UNIVERSIDADE FEDERAL DA PARAÍBA
NÚCLEO DE TECNOLOGIA DA INFORMAÇÃO
COORDENAÇÃO DE APOIO TÉCNICO-CIENTÍFICO

ADMINISTRAÇÃO

DE SISTEMAS

GNU/LINUX

(Conectiva 10)

Prof. Sérgio de Albuquerque Souza

JOÃO PESSOA - PARAÍBA
5 de dezembro de 2004

Sumário

Parte I

Instalando e Usando o GNU/Linux

CAPÍTULO 1

Introdução

A administração de sistemas GNU/Linux e do Unix em geral, sempre foi concebida como algo que só podia ser feita por pessoal altamente especializado, que dominava centenas de comandos e procedimentos, tanto é que os administradores de rede/sistemas eram geralmente chamados de magos (*wizards*) ou gurus, mas este panorama mudou bastante com o surgimento de ferramentas de gerenciamento mais amigáveis, muitos delas baseadas em interfaces gráficas[1], porém neste curso será dada ênfase ao modo texto, pois neste modo, tem-se uma melhor visão do que está, de fato, acontecendo com o sistema e não ficando apenas a cargo de clicar aqui e/ou ali para ver se algo ocorre.

Mesmo com todo esse avanço, a administração de sistemas está longe de ser uma atividade trivial para qualquer que seja o sistema operacional, dada a quantidade de tarefas desempenhadas pelo administrador, dentre as quais, podemos citar:

- Instalação do sistema operacional.
- Criação, remoção e gerenciamento das contas dos usuários.
- Fazer *Backups*/Restaurações.
- Instalação e atualizações de programas e serviços.
- Instalação de novos hardware no sistema (impressora, discos rígidos, etc).
- Fazer manutenção do sistema como um todo.
- Monitorar a segurança do sistema.
- Montar e administrar uma rede.

Portanto este texto visa dar uma "brevíssima" introdução em cada um desse tópicos, tendo sempre em vista, o fato de se destinar a pessoas iniciantes, ou seja, que tenham pelo menos uma noção de como funciona um computador em rede, mesmo os que tiveram pouco ou nenhum contato com os sistemas GNU/Linux.

Como ponto de partida, para este curso, utilizaremos o dentro de um GNU/Linux, no caso o Debian, programa *VMware* (ver anexos A.1 na página 97 e A.3 na página 98) que será de grande utilidade na criação e instalação do GNU/Linux, bem como da rede, tudo de maneira virtual, dando a cada aluno a capacidade de instalar, modificar e gerenciar o seu próprio servidor e sua própria rede com os seus próprios usuários (clientes).

[1] Aplicativos gráficos: *Webmin, Swat, Linuxconf, Synaptic*, etc

Neste ambiente virtual podemos instalar qualquer Windows, bem como qualquer uma das grandes distribuições[2] para o GNU/Linux, no qual destacamos o *Conectiva*, *Red Hat/Fedora*, *Kurumin*, *Debian*, *SuSE* e *Mandrake*.

No curso, usaremos o *Conectiva Linux 10* que tem quase a totalidade de seus aplicativos traduzidos para o português[3], espanhol e inglês, tendo o português como sua base, facilitando a integração com o usuário brasileiro. O que não quer dizer que outras distribuições não estejam disponíveis na língua portuguesa.

A seguir daremos algumas noções básicas, dos termos mais utilizados.

1.1 Sistema Operacional

O Sistema Operacional, ou simplesmente *SO* é o conjunto de programas que fazem o elo do usuário e seus programas com o computador.

É chamado GNU/Linux o conjunto do *kernel* e demais programas, como *shells*, compiladores, bibliotecas de funções, etc.

O GNU/Linux, como sistema operacional é composto por três partes:

1.1.1 O Núcleo (kernel)

Embora o Núcleo (*kernel*) seja uma parte importante do Linux, ele sozinho não constitui o sistema GNU/Linux. O *kernel* do sistema é responsável pelas funções de mais baixo nível, no sentido de está mais próximo ao equipamento, como o gerenciamento de memória, gerenciamento de processos e da CPU, pelo suporte aos sistemas de arquivos, dispositivos e periféricos conectados ao computador, como placas controladoras, placas de rede, de som, portas seriais, etc.

O *kernel* do Linux pode ser compilado[4] para se adequar melhor ao tipo de máquina e ao tipo de tarefa que essa máquina vai executar. Por exemplo, se no servidor não houver a necessidade de usar um determinado tipo de placa de rede, é possível compilar o *kernel* sem suporte a essa placa, resultando assim em um *kernel* de menor tamanho e mais eficiente.

1.1.2 Aplicações do Sistema

O *kernel* faz muito pouco sozinho, uma vez que ele só provê os recursos que são necessários para que outros programas sejam executados. Logo, é necessária a utilização de outros programas para implementar os vários serviços necessários ao sistema operacional.

No *kernel*, as aplicações do sistema, bem como qualquer outro programa, rodam no que é chamado “modo usuário”, logo, a diferença entre aplicações de sistema e aplicações do usuário se dá pelo propósito de cada aplicação. Aplicações do sistema são necessárias para fazer o sistema funcionar, enquanto as aplicações do usuário são todos programas utilizados pelo usuário para realizar uma determinada tarefa (como um processador de texto, por exemplo).

Entre as aplicações de sistema do GNU/Linux, pode-se citar:

init - é o primeiro processo lançado após o carregamento do *kernel* na memória, e é ele o responsável por continuar o processo de inicialização (*boot*), ativando os outros programas. É o *init* o responsável, também, por garantir que o *getty* esteja sendo executado (para que os usuários possam entrar no sistema) e por adotar processos órfãos (processos filhos no qual o pai morreu), pois no GNU/Linux todos os processos devem estar em uma mesma árvore, e possuírem um pai (excluindo o processo *init*, que não tem pai).

[2] O que diferencia uma distribuição de outra é a maneira como são organizados e pré-configurados os aplicativos que cada uma contém.

[3] Português do Brasil

[4] Compilação é o processo de transformação de um código-fonte em um programa executável.

getty - provê o serviço responsável pelo *login* dos usuários em terminais textos (virtuais ou não). É ele que lê o nome do usuário e a senha e chama o programa *login* para validá-los; caso estejam corretos é lançado um *shell*, caso contrário o processo todo é reiniciado.

syslog - é responsável por capturar as mensagens de erro geradas pelo *kernel* ou por outras aplicações de sistema, e por mostrá-las posteriormente quando o administrador do sistema solicitá-las.

1.1.3 Aplicações do Usuário

As aplicações do usuário são todas aquelas utilizadas pelo usuário para executar uma determinada tarefa. Editores de texto, editores de imagens, navegadores e leitores de *e-mail* se encaixam nessa categoria.

1.2 Ambiente Gráfico

No GNU/Linux a responsabilidade pelo ambiente gráfico não é do *kernel* e sim de um programa especial, o *XFree86*. No entanto, este programa provê apenas as funções de desenho de elementos gráficos e interação com a placa de vídeo. A interação final do usuário com a interface gráfica se dá através de programas gerenciadores de janelas, como o *KDE*, o *WindowMaker* e o *GNOME* dentre outros, e são eles os responsáveis pela "aparência" gráfica do seu GNU/Linux.

A separação do ambiente gráfico do resto do sistema apresenta muitas vantagens. Como o ambiente gráfico consome recursos do sistema, é possível desativá-lo, principalmente em servidores, resultando assim em um melhor desempenho de outras aplicações, uma vez que a quantidade de processamento da *CPU* que seria utilizado para o *XFree86*, poderá ser utilizado para essas aplicações. Além do mais, o desenvolvimento do ambiente gráfico pode ocorrer de maneira independente ao do *kernel*.

O GNU/Linux também pode funcionar em modo texto (ver capítulo 3 na página 19). Nesse caso a interação com o usuário se dá por meio de um *shell*, como o *bash*, que é capaz de interpretar e executar comandos digitados pelo usuário.

1.3 O GNU/Linux

O GNU/Linux é um sistema operacional criado em 1991 por *Linus Torvalds* na Universidade de Helsinki na Finlândia. É um sistema operacional de código aberto distribuído gratuitamente pela Internet. Seu código fonte é liberado como *Free Software* (software livre).

Isto quer dizer que você não precisa pagar nada para usar o GNU/Linux, e não é crime fazer cópias para instalar em outros computadores, e inclusive você deve fazer isto. Ser um sistema de código aberto pode explicar a performance, estabilidade e velocidade em que novos recursos são adicionados ao sistema.

Para rodar o GNU/Linux você precisa, no mínimo, de um computador *386 SX* com *2 MB* de memória e 40MB disponíveis em seu disco rígido para uma instalação básica e funcional ou um disquete para as chamadas mini-distribuições, no qual destaco o *Freesco*.

O sistema segue o padrão *POSIX* que é o mesmo usado por sistemas *UNIX* e suas variantes. Assim, aprendendo o GNU/Linux você não encontrará muita dificuldade em operar um sistema do tipo *UNIX*, *FreeBSD*, *HPUX* ou *SunOS*, bastando apenas aprender alguns detalhes encontrados em cada sistema.

O código fonte aberto permite que qualquer pessoa veja como o sistema funciona (útil para aprendizado), corrija algum problema ou faça alguma sugestão sobre sua melhoria, sendo esse um dos motivos de seu rápido crescimento, do aumento da compatibilidade de periféricos (como novas placas sendo suportadas logo após seu lançamento) e de sua estabilidade.

Outro ponto em que ele se destaca é o suporte que oferece a placas, CD-Roms e outros tipos de dispositivos de última geração e mais antigos (a maioria deles já ultrapassados e sendo completamente suportados pelo sistema operacional). Este é um ponto forte para empresas que desejam manter seus micros em pleno funcionamento e pretendem investir em novas tecnologias.

Hoje o GNU/Linux é desenvolvido por milhares de pessoas espalhadas pelo mundo, cada uma dando sua contribuição ou mantendo alguma parte gratuitamente, como por exemplo do *kernel*. Linus Torvalds ainda trabalha no desenvolvimento do *kernel* e também ajuda na coordenação entre os desenvolvedores.

O suporte ao sistema também se destaca como sendo o mais eficiente e rápido do que qualquer programa comercial disponível no mercado. Existem centenas de consultores especializados espalhados ao redor do mundo. Você pode se inscrever em uma lista de discussão e relatar sua dúvida ou alguma falha, e sua mensagem será vista por centenas de usuários na Internet e algum irá te ajudar ou avisará as pessoas responsáveis sobre a falha encontrada para devida correção.

1.4 Linux e o Projeto GNU

A idéia do *Richard Stallman*, em 1984, para *Projeto GNU* (*GNU's Not Unix*, ou *GNU* não é *Unix*) era desenvolver um sistema operacional livre similar ao *Unix*, pois todo usuário de computadores necessita de um sistema operacional; se não existe um sistema operacional livre, você não pode nem mesmo iniciar o uso de um computador sem recorrer a software proprietário. Portanto, o primeiro ítem na agenda do software livre é um sistema operacional livre.

A palavra *livre* (*free* no original) está relacionada com liberdade, em vez de preço. Você pode ou não pagar um preço para obter software *GNU*. De qualquer modo, uma vez que você tenha o software você tem quatro liberdades específicas na sua utilização:

- A liberdade de executar o programa, para qualquer propósito.
- A liberdade de estudar como o programa funciona, e adaptá-lo para as suas necessidades. O aceso ao código-fonte é um pré-requisito para esta liberdade.
- A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo.
- A liberdade de aperfeiçoar o programa, e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie. O acesso ao código-fonte é um pré-requisito.

A maioria dos projetos de software livre tem por objetivo desenvolver um programa em particular para uma tarefa em particular. Por exemplo, *Linus Torvalds* escreveu um núcleo similar ao *Unix* (Linux); *Donald Knuth* escreveu um formatador de textos (Tex/LaTex)[5]; *Bob Scheifler* escreveu um sistema de janelas (X Windows). É natural medir a contribuição deste tipo de projeto pelos programas específicos que vieram daquele projeto.

À parte do *GNU*, um outro projeto desenvolveu um sistema livre similar ao *UNIX*, este sistema é conhecido como *BSD* e foi desenvolvido na Universidade da Califórnia em Berkeley. Os desenvolvedores do *BSD* foram inspirados a fazer seu trabalho como software livre seguindo o exemplo do *Projeto GNU*, e ocasionalmente encorajaram os ativistas do *GNU*. Um sistema operacional livre que existe hoje é quase com certeza ou uma variante do sistema *GNU* ou um tipo de sistema *BSD*.

Utilizamos sistemas *GNU* baseados em Linux hoje para a maioria dos nossos trabalhos. Mas não utilize o nome "Linux" de forma ambígua. Linux é o núcleo, um dos principais componentes essenciais do sistema e o sistema como um todo é mais ou menos o sistema *GNU*.

Para enfatizar o que estamos utilizando, chamaremos de GNU/Linux, ou seja, um sistema operacional livre com o núcleo Linux.

[5]Formatador de textos utilizado para a geração deste material

CAPÍTULO 2

Instalando o GNU/Linux

Neste capítulo daremos uma visão geral, passo a passo, da instalação de um sistema operacional GNU/Linux, que no caso será o Conectiva Linux 10 da Conectiva SA, que é uma empresa brasileira, com sede em Curitiba - Paraná, fundada em 1995.

2.1 Requisitos do sistema

Os requisitos mínimos necessários para instalação do Conectiva Linux - Desktop, recomendados pela própria Conectiva, são:

- Processador 100 MHz
- 2GB de espaço em disco
- 64 de memória RAM, para instalação e utilização do sistema
- Adaptador de vídeo VGA

Estes requisitos permitem que você utilize a sua máquina com o Conectiva Linux, porém com certas restrições quanto à interface gráfica e agilidade.

É importante frisar que estes requisitos são aplicados a uma instalação padrão. Instalações personalizadas e instalação de servidores podem requerer mais ou menos hardware, dependendo de cada caso.

Daremos ênfase a instalação no modo gráfico que é mais prático, mas caso você deseje instalar via modo texto, perceberá que as opções de instalação são bastante similares, só um pouco mais trabalhosa.

O "instalador" foi programado com uma série de janelas que irão servir como guia. Cada janela irá configurar uma ou mais opções, que você deve selecionar. Os ítens que estiverem sendo configurados aparecem na área de seleção. Você pode selecionar as características nessa área, pressionando as teclas direcionais ou a tecla *Tab*, ou ainda, usar o *mouse* (caso seja detectado automaticamente, o que ocorre na maioria dos casos).

Para prosseguir, ao final de cada operação, basta clicar no botão Próximo, situado no canto inferior direito da tela. Caso queira fazer algumas modificações nas configurações já feitas, utilize o botão Anterior.

Caso esteja com dúvidas sobre como proceder, você pode utilizar o botão de Ajuda. Para visualizá-lo, mova o *mouse* até o lado esquerdo da sua tela ou tecle *F1*.

2.2 Nosso Caso

No caso, o *computador virtual* (ver anexo A.3 na página 98) terá uma configuração bem mais simples e modesta, mas que atenderá completamente, o objetivo de se montar um servidor com os principais serviços de rede, ou seja, vamos utilizar:

- Processador 500MHz;
- Um HD com 700 MB de espaço em disco, definido virtualmente como um arquivo;
- 48 MB de memória RAM, para instalação no modo gráfico e utilização do sistema;
- Adaptador de vídeo VGA;
- Adaptador de rede (no modo bridged);
- Um CD-ROM utilizando a imagem do primeiro CD do *Conectiva 10*

2.3 Iniciando a Instalação

Após configurar o *computador virtual* (ver A.3 na página 98) e com o CD 1 do Conectiva Linux em "mãos", você deverá iniciar seu computador com a função de inicialização pelo CD, função esta que pode ser ativada através da BIOS[1] (teclando *F2*), que no nosso caso já está habilitado por padrão.

Assim que o computador for iniciado, com inicialização pelo CD-ROM, o monitor deve ficar escuro por alguns segundos, até que seja ativada a *Tela do Modo de Instalação*. Ela permite que você escolha em qual modo do Conectiva Linux deverá ser instalado.

Se em 30 segundos você não pressionar a tecla *Enter* nem mudar de opção, o instalador vai escolher a opção *Conectiva Linux 10 (Graphic Interface)*, para iniciar a instalação padrão (modo gráfico).

Das principais opções, destacamos:

- *Conectiva Linux 10 (Graphic Interface)*

  Instalação via modo gráfico.

- *Conectiva Linux 10 (Text Interface)*

  Instalação via modo texto, seguindo a mesma seqüência de telas do modo gráfico, porém apenas usando o teclado.

- *Conectiva Linux 10 (Expert)*

  Selecione esta opção somente se você já possui alguma experiência com o Conectiva Linux. O instalador não fará nenhuma tentativa de detecção de hardware, e portanto, você deverá escolher as opções adequadas.

- *Conectiva Linux 10 (vesa driver)*

  Caso o sistema não reconheça sua placa de vídeo, tente esta opção. Drivers VESA podem ser reconhecidos pela maior parte das placas de vídeo.

- *Conectiva Linux 10 Rescue Image*

  Quando esta opção for selecionada, o instalador em modo gráfico não é iniciado, e em vez dele, um ambiente modo texto será mostrado. Ele serve como recuperação, e fornece apenas ferramentas básicas para correções ou recuperações do sistema. Acesse esta opção somente se você souber o que está fazendo.

[1] *Basic Input Output System*

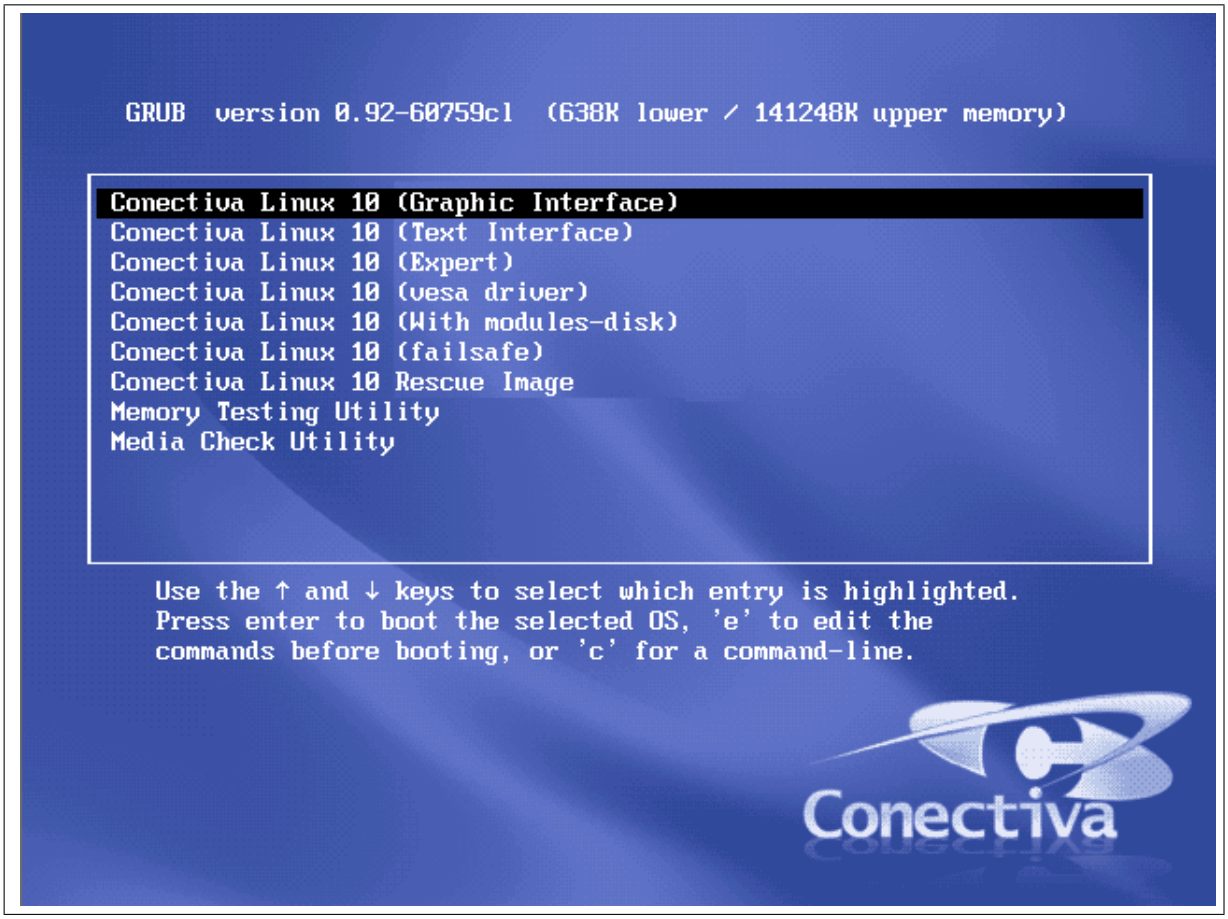


Figura 2.1: Tela inicial de instalação

Vamos utilizar a primeira opção e seguiremos o roteiro padrão, do instalador Conectiva, e definir:

2.3.1 Escolha do idioma

Escolheremos como idioma o português, por motivos óbvios.

2.3.2 Configuração do mouse

O *mouse* normalmente é detectado sem problemas, mas caso contrário, marque a opção *Selecionar mouse manualmente* e localize na lista (usando o teclado) o modelo do seu *mouse*; caso não saiba qual é o modelo, experimente a opção *Generic Mouse*.

Marque opção *Emulação do terceiro botão*, para que o botão do meio se comporte como se fosse a execução simultânea dos botões esquerdo e direito do *mouse*.

2.3.3 Configuração do teclado

Na tela de *Seleção do Teclado* já existem opções pré-selecionadas, tanto na lista de modelo, como na de *layout*. Isso porque o modelo *Brazilian ABNT2* e o *layout Brazilian*, são configurações mais comuns em teclados brasileiros (com a tecla ”ç”).

Se o *layout* do seu teclado não é brasileiro, a configuração mais provável é a *Generic 105-key (intl) PC*, na lista de modelos, e *English w/ deadkeys*, na lista de *layout*.

Teste se sua configuração esta correta, na *Área de teste do teclado* e digite qualquer palavra.

2.3.4 Tabela de partições

No caso de não existir nenhuma tabela de partições no HD, ou seja, um HD novo, será mostrado uma janela mostrando que não foi possível encontrar a tabela de partições e que não poderá continuar sem a mesma, logo indicaremos ao instalador, que será criada uma tabela de como será particionado o nosso HD de 700 MB.

2.3.5 Quantos CDs você possui?

Esta opção é importante para o sistema saber quais pacotes poderá instalar, sem que surjam problemas de dependências de pacotes. No nosso caso apenas o CD 1.

Caso já tenha algum sistema operacional Conectiva Linux no seu disco, aparecerá uma tela perguntando se você deseja fazer uma nova instalação ou apenas atualizar a versão anterior. Selecione a opção correta e prossiga.

2.3.6 Escolha seu perfil

A tela de perfil permite a escolha do tipo de instalação que você poderá utilizar no Conectiva Linux. Suas opções, apenas com o CD 1, são as seguintes (**utilizaremos a instalação mínima**):

Instalação Padrão - Serão instalados todos os recursos necessários para um computador doméstico. Este perfil instala o *ConectivaOffice*, programas de multimídia, gráficos, jogos e principais recursos para um ambiente doméstico.

Instalação Mínima - Neste perfil serão instalados apenas programas básicos necessários para o funcionamento do sistema, além de um simples editor de textos chamado `vi` (ver anexo A.2 na página 97) e o programa para instalação de pacotes `apt-get` (ver capítulo 7 na página 34). Ideal para servidores, onde o próprio usuário instalaria os serviços e seus respectivos pacotes, de acordo com sua necessidade. **Este será o perfil utilizado no curso.**

Notebook - Semelhante a instalação padrão, porém com drivers específicos para notebooks.

Instalação Personalizada - Permite que seja feita uma seleção manual dos pacotes a serem instalados. Recomendado para usuários avançados, que possuam uma grande familiaridade com a sistema.

E nas opções de instalação, são mostradas:

- *Forçar particionamento manual*, permite que o usuário faça o particionamento, em vez da sua execução automática. Caso você não possua outros sistemas instalados na máquina, ou seja, seu HD está livre para a instalação do Conectiva Linux, então isto é opcional, más é o que utilizaremos, pois estamos instalando em um HD com apenas 700 MB.

- *Não instalar documentação*, possibilita aos usuários mais experientes, deixar mais espaço em disco, pelo simples fato de não instalar os documentos referentes à manuais, *faqs* e etc. Não deverá ser marcada.

Com a utilização do CD 2 do Conectiva, além dessas opções acima, ainda seriam mostrado as opções:

Desktop Corporativo - Perfil totalmente voltado para o usuário corporativo, atendendo suas principais necessidades num ambiente enxuto e conservador. Ideal para estações de trabalho de empresas, que desejam otimizar seus processos.

Servidor Web - Perfil totalmente otimizado para a criação de um servidor web, com a instalação dos principais pacotes para sua configuração.

Servidor Samba - Este perfil permite instala os pacotes para configuração de um servidor Samba.

Servidor de Correio Eletrônico - Os principais pacotes para a configuração de um servidor de e-mail serão instalados neste perfil.

Servidor Firewall - Ideal para administradores, pois permite a otimização do processo de configuração de um servidor de *firewall* básico, instalando os principais pacotes para esta configuração.

Desenvolvimento - Inclui diversas ferramentas de desenvolvimento, fontes do kernel, ferramentas de *debug* e programa de controle de versão, entre outros programas, utilizados por programadores.

Todos os Perfis - Instalação de todos os perfis disponíveis.

2.3.7 Particionando o HD

O HD deve estar separado em áreas, tanto para o próprio sistema GNU/Linux, como para outros sistemas operacionais. Isto é feito dividindo o disco em partições, e através do particionador, é possível que existam outros sistemas operacionais junto com o Linux. Podem existir os seguintes casos de particionamento:

- O disco não contém nenhum outro sistema operacional instalado, ou seja, há espaço livre não-particionado. Neste caso, as partições deverão ser criadas.
- O disco já contém outros sistemas operacionais instalados, está dividido em várias partições e ainda existe uma partição livre. O particionador reconhece as partições existentes, e você deve configurar a partição livre para instalar o Linux.
- O disco já possui o Linux instalado, mas você deseja fazer um novo particionamento. Esta ação pode ser destrutiva, onde todos os dados serão apagados e as partições refeitas. Neste caso, você deve salvar as informações para que não sejam apagadas e reformular as partições. O redimensionamento pode ser ainda não-destrutivo, onde são alterados os tamanhos das partições na *Tabela de Alocação de Arquivos* (FAT). É recomendável você fazer uma cópia de segurança dos dados.

No nosso caso, temos apenas um HD vazio, ou seja, temos que criar as partições necessárias para a instalação e execução do GNU/Linux, ou seja, utilizaremos duas partições simples, para efeito didático na qual destacamos:

- **Partição de Troca (Swap)**

  Destinada ao suporte de memória virtual. O tamanho da partição deverá ser de, no mínimo[2], igual à quantidade de memória do equipamento. No nosso caso utilizaremos por exemplo 50 MB com uma partição do tipo primário e com a opção de formatar.

- **Partição Raiz**

  Uma partição raiz é montada como "/" (pasta raiz)[3] quando o GNU/Linux inicia, e contém os ítens necessários à inicialização do sistema de arquivos de configuração. Ela pode ser, inclusive, a única partição do sistema; se isto ocorrer, você deve colocar o tamanho da partição igual ao tamanho do disco. No nosso caso de 550 MB com uma partição do tipo primário, com sistema de arquivo *ext3*, inicializável e com a opção de formatar.

[2]Esse número não é exato, pois depende de vários fatores.

[3]Equivalente ao "C:" do Windows.

Note que deixamos uma parte do HD livre, que futuramente será incorporado ao sistema, com o comando `fdisk`.

Após a formatação, toda a estrutura para leitura/gravação de arquivos e diretórios pelo sistema operacional estará pronta para ser usada.

Cada sistema de arquivos tem uma característica em particular mas seu propósito é o mesmo o de oferecer ao sistema operacional a estrutura necessária para ler/gravar os arquivos e diretórios.

Entre os principais sistemas de arquivos existentes podemos citar:

Ext2 - Usado em partições nativas do Linux para o armazenamento de arquivos. É identificado pelo código 83. Seu tamanho deve ser o suficiente para acomodar todo os arquivos e programas que deseja instalar no GNU/Linux.

Ext3 - É totalmente compatível com o *ext2* em estrutura e possui melhorias como o recurso de *jornaling*[4], e é identificado pelo tipo 83.

Reiserfs - É um alternativo ao *ext2/3* que também possui suporte a *journaling*. Entre suas principais características, estão que ele possui tamanho de blocos variáveis, suporte a arquivos maiores que 2 GB (esta é uma das limitações do *ext3*) e o acesso *mhash* a árvore de diretórios é um pouco mais rápida que o *ext3*.

Swap - Usado para oferecer memória virtual ao sistema. Note que é altamente recomendado o uso de uma partição *Swap* no sistema (principalmente se você tiver menos que 16 MB de memória RAM). Este tipo de partição é identificado pelo código 82.

Proc - Sistema de arquivos do *kernel*.

FAT12 - Usado em disquetes no DOS.

FAT16 - Usado no DOS e oferece suporte até discos de 2 GB.

FAT32 - Também usado no DOS e oferece suporte a discos de até 2 TB.

NTFS - Usado nos sistemas baseados no Windows (NT/2000/XP).

2.3.8 Configurando a rede

Nesta janela você deverá obrigatoriamente escolher um nome qualquer para referenciar sua máquina.

Exemplo 2.3.1 *Um nome do tipo* `servidorXX.cursoXX.nti.ufpb.br`*, onde* `XX` *é um número dado pelo instrutor, na hora da instalação e* `cursoXX.linux.nti.ufpb.br` *será o domínio.*

Depois de escolhido o nome da sua máquina, escolha a interface da sua rede. No nosso caso a opção deve ser *Configurar rede via DHCP* (ver capítulo 16 na página 72).

Aguarde alguns segundos até o surgimento da próxima tela. Durante este tempo, o instalador ativará os módulos compatíveis com seu hardware e pegará as informações necessárias para o protocolo TCP/IP (IP, mascara, roteador e servidor de nomes) na rede para que o mesmo "entre" na rede.

[4]O *journal* mantém um *log* de todas as operações no sistema de arquivos, caso aconteça uma queda de energia elétrica (ou qualquer outra anormalidade que interrompa o funcionamento do sistema), o `fsck` verifica o sistema de arquivos no ponto em que estava quando houve a interrupção, evitando a demora para checar todo um sistema de arquivos (que pode levar minutos em sistemas de arquivos muito grandes).

2.3.9 Instalando os programas

Agora, o conjunto de programas começará a ser copiado e o GNU/Linux será automaticamente instalado seu computador. O tempo deste processo varia de acordo com cada máquina e com as opções escolhidas no inicio da instalação. Você pode acompanhar o andamento da instalação nas barras de progresso situadas na base da tela.

A partir desse momento será ativado o *Synaptic*, ferramenta para instalação dos pacotes correspondentes ao perfil selecionado. Todo processo do *Synaptic* será automático, exceto se você tiver selecionado a opção *Instalação Personalizada*. Caso se tenha os outros CDs de instalação, será necessário trocar os discos quando solicitado pelo sistema.

2.3.10 Seleção da placa de vídeo

Nesta etapa, o instalador tentará detectar automaticamente as configurações de sua placa de vídeo[5]. Se você estiver executando a instalação em modo gráfico, isso significa que o instalador já detectou seu hardware automaticamente, e neste caso, basta marcar a opção *Usar as configurações detectadas automaticamente* e clicar em Próximo.

Caso o instalador não tenha conseguido detectar o modelo de sua placa corretamente, ou você deseje selecionar uma placa diferente da configurada, você deverá fazer a seleção manualmente. Clique no ítem *Selecione* a placa de vídeo manualmente e procure pelo modelo de sua placa. Assim que o achar, clique sobre ele, e depois selecione o driver da sua placa na área direita.

Caso não consiga, use o modo *VGA Padrão*, para só depois procurar pelo *driver* correto.

2.3.11 Seleção do monitor

Na maioria das vezes, o monitor é automaticamente detectado, fazendo com que você apenas precise clicar em Próximo.

Entretanto, caso não tenha sido detectado corretamente, use a barra de rolagem, situada ao lado direito da tela, para encontrar o nome do fabricante. Em seguida, clique no marcador "+", ao lado do nome do fabricante, para abrir uma lista com os modelos disponíveis. Selecione o modelo de seu monitor, e passe para a próxima tela.

Se seu monitor não estiver listado, você pode escolher um modelo similar do mesmo fabricante. Caso não funcione corretamente, opte por um modelo genérico.

2.3.12 Configuração da área de trabalho

Neste momento o programa de instalação exibirá uma janela com as opções que permitem escolher a resolução de vídeo para seu monitor e também o número de cores padrão do modo gráfico.

Selecione as resoluções de vídeo desejadas na caixa *Resolução do Monitor*. Se nenhuma opção for escolhida, podem ser adotadas como padrão as resoluções que estão habilitadas na tela.

Clique em Tentar Configuração para testar suas configurações. Quando você utilizar esta opção, uma nova instância do "X" será aberta em sua máquina. Infelizmente, alguns hardwares mais antigos não suportam a execução de duas instâncias do *XServer* e travam. Caso isto ocorra, reinicie a instalação, e não teste a configuração da próxima vez.

2.3.13 O superusuário ou root

O *Superusuário* ou *root* é a conta com o poder de administrador do sistema. Ele possui acesso exclusivo e total ao sistema, podendo fazer qualquer tipo de configuração. Por uma questão de segurança, o *root* é o único usuário que tem permissão para instalar e remover programas,

[5]Responsável pela montagem e exibição das imagens na tela do monitor.

criar novos usuários, configurar os periféricos, dentre outras funções. Este procedimento evita intervenções desordenadas de outros usuários e garantem a integridade do sistema.

Para definir a senha do superusuário, vá ao ítem *Definição da senha de root* e digite, no campo *Senha*, uma senha de pelo menos seis caracteres, misturando letras e números. Redigite a mesma senha no campo *Confirmação*.

Exemplo 2.3.2 *Esta senha por exemplo* (@12da.silva?) *é bastante razoável, mas para evitar complicações futuras iremos colocar como senha do root a palavra* **instalando**, *que não é nada boa, mas num futuro iremos modificá-la (ver capítulo 4 página 23).*

Observação 2.3.1 *Lembre-se de que esta senha é super importante, e é um pouco complicado recuperá-la. É aconselhável que seja anotada em algum lugar seguro. Também é recomendado que o administrador do sistema crie um usuário para seu uso comum, e utilize o usuário root somente quando for necessário fazer alguma alteração ou configuração do sistema.*

2.3.14 Configuração de usuários

Na mesma tela de definição da senha do superusuário, você poderá definir os usuários que irão utilizar o computador no qual você está instalando o GNU/Linux. O sistema permite que cada pessoa que acesse o computador possua uma área personalizada, acessada através de sua conta (nome de *login + senha*).

No espaço *Criação de Usuários*, digite seu nome completo, ou de uma pessoa que irá utilizar o computador com você. O campo *Nome de Acesso* (que é o nome de login para a conta) será preenchido automaticamente, mas se você preferir, poderá modificar este campo.

Faremos toda esta parte de criação e remoção de usuários em modo texto, mais adiante, no capítulo 4 na página 23.

2.3.15 Gerenciando os sistemas da máquina

Gerenciadores de inicialização são programas que carregam um sistema operacional e/ou permitem escolher qual será iniciado. Normalmente este programas são gravados no setor de inicialização (*boot*) da partição ativa ou no *MBR* (*Master Boot Record*) do HD.

O gerenciador padrão utilizado pelo Conectiva Linux é o *GRUB* (*Grand Unified Boot Loader*), pois apresenta alguns recursos extras com relação as outras opções disponíveis. Ele é flexível, funcional e poderoso, podendo inicializar sistemas operacionais como o Windows (9x, ME, NT, 2000 e XP), DOS, Linux, GNU Hurd, *BSD, OS/2 e etc. Podemos destacar também o suporte a vários sistemas de arquivos (ver 2.3.7 anterior).

Por utilizar o padrão *Multiboot* ele é capaz de carregar diversas imagens de *boot* e módulos. O GRUB também permite buscar imagens do *kernel* pela rede, por cabo seriais, suporta discos rígidos IDE e SCSI, detecta toda a memória RAM disponível no sistema, tem interface voltada para linha de comandos ou menus de escolha, além de suportar sistemas sem discos e terminais remotos.

Observação 2.3.2 *Utilizaremos o GRUB instalado no MBR.*

2.3.16 Instalação concluída

Parabéns! Você concluiu a instalação do Conectiva Linux em seu computador.

Agora é só reiniciar o "computador virtual", "retirando" o CD-ROM e pronto, o próximo passo é colocar a mão massa e trabalhar.

CAPÍTULO 3

Modo Texto no GNU/Linux

Neste capítulo serão estudados os principais comandos do GNU/Linux em modo texto, assim como será feita uma breve introdução sobre o *shell*, que é o interpretador que está por trás destes comandos, e sobre o *bash*, que é um dos tipos de it shell mais comuns no mundo Linux.

Apesar de todas as facilidades dos ambientes gráficos, é interessante entender um pouco do GNU/Linux modo texto, pois com certeza esse modo de se trabalhar pode trazer grandes benefícios aos administradores, como por exemplo em forma de agilidade em certos tipos de serviços que se tornam muito mais rápidos ao serem executados desta maneira e principalmente se você estiver acessando o servidor remotamente (ver capítulo 9 na página 40).

Você perceberá que a maior parte das ações que você executa no ambiente gráfico podem ser feitas através de linhas de comandos, de um modo muitas vezes mais simples e rápido do que você possa imaginar.

O modo texto do GNU/Linux é composto por terminais. Um terminal, ou console, é o conjunto formado pelo teclado e o monitor, que constitui o dispositivo padrão de entrada e saída de dados, local ou remoto, do tipo:

```
Conectiva Linux 10 [tty1]
Kernel 2.6.5-63077cl (i686)

curso login:
```

onde será pedido o nome da conta do usuário (*login*) e em seguida a senha (*passwd*).

No GNU/Linux, existem terminais virtuais que podem ser ativados e alternados pela seqüência de teclas *Ctrl-Alt-Fn*, onde ($1 \leq n \leq 7$) é o número do terminal virtual no qual está trabalhando.

Imagine o seguinte: enquanto você está digitando um texto num desses terminais, no outro está executando um processo demorado, como uma procura no disco e em um outro ativando algum serviço.

No nosso caso iremos trabalhar basicamente com 3 ou 4 terminais, da seguinte maneira:

Terminal	F1	F2	F3	F4
Usuário	*root*	*root*	*root*	*teste*
Tarefa	editar os arquivos de configuração	ativar e/ou desativar os serviços	checar os arquivos de *logs*	testar
Comando	`pico` ou `vi`	`service`	`tail -f`	

3.1 Shell

Um *shell* é um interpretador de comandos que analisa o texto digitado na linha de comandos e executa esses comandos produzindo algum resultado.

O *shell* pode ser considerado como um ponto a partir do qual você pode iniciar todos os comandos do GNU/Linux, inclusive o modo gráfico. Da mesma maneira que existem várias aplicações para editor de textos (ver anexo A.2 na página 97), cada uma com suas funcionalidades, existem também vários *shells*, cada um com suas configurações e funcionalidades específicas, onde podemos destacar, dentre outros: *bash* (o mais utilizado e em muitos sistemas o padrão), *csh*, *ksh*, *zsh* e *tcsh*.

Quando você entra no modo texto, o *shell*, já lhe deixa dentro de um diretório, caso tenha permissão de usá-lo, que é pasta local do usuário na máquina, também chamada de diretório *home*. É lá que você deverá colocar os seus arquivos, onde você tem permissão de criar, modificar e apagar seus arquivos e pastas.

A partir de agora será estudado mais detalhadamente o *shell bash*.

3.2 O bash

No GNU/Linux existe um recurso muito interessante chamado de *Tab Completion*. Quando você está digitando um comando ou nome de arquivo muito longo, basta que você digite os primeiros caracteres desse comando e aperte a tecla *Tab* que o *shell* se encarrega de completar o resto do nome. Nem todos os nomes podem ser completados até o final pelo fato de existirem outras opções no processo de complementarão, então, o *shell* para de completar e emite um *bip* avisando que não conseguiu completar totalmente a sentença. Caso você aperte *Tab* novamente, o *shell* irá lhe mostrar as opções que você tem para completar a sentença.

Outro recurso, muito importante, é a de poder redirecionar a saída de um comando para a entrada de outro. Esse truque, na realidade é um comando chamado *pipe* e tem como sintaxe uma barra vertical "|". A utilização do *pipe* é da seguinte forma:

```
[root@curso root]# comando1 | comando2 | comando3 ...
```

Ele irá redirecionar a saída do `comando1` para a entrada do `comando2`, o resultado deste será redirecionado para o `comando3` e assim por diante (ver 3.3.5 página 22).

3.3 Comandos

Nesta seção serão mostrados os vários comandos básicos (os mais utilizados) e uma breve descrição dos mesmos. Em todos os casos, o usuário **não** deve ter a obrigatoriedade de saber todos os parâmetros de cada comando, pois para cada um deles, em geral, existe a opção de uma ajuda (*help*) na linha de comando, do tipo:

```
[root@curso root]# nome.do.comando --help
[root@curso root]# info nome.do.comando
[root@curso root]# help nome.do.comando
```

ou utilizar o manual do referido comando/programa/configuração, com o comando *man*, escrevendo o seguinte:

```
[root@curso root]# man nome.do.comando
```

A seguir estão listados e separados por tipo os principais comandos:

3.3.1 Manipulação de Arquivos e Diretórios

Comando	Descrição
ls	Lista os arquivos de um diretório.
cd	Entra em um diretório. Você precisa ter a permissão de execução para entrar no diretório.

pwd	Mostra o nome e caminho do diretório atual.
rm	Apaga arquivos. Também pode ser usado para apagar diretórios e sub-diretórios vazios ou que contenham arquivos.
cp	Copia arquivos.
mv	Move ou renomeia arquivos e diretórios.
mkdir	Cria um diretório no sistema.
rmdir	Remove um diretório do sistema.
tree	Lista os diretórios em forma de arvore.

3.3.2 Relacionado ao Sistema

Comando	Descrição
history	Exibe uma listagens dos últimos comandos executados no *shell*.
ps	Exibe uma lista os processos ativos, indicando o número do processo (PID).
top	Exibe uma tabela (em tempo real) dos processos ativos no sistema.
date	Permite ver/modificar a Data e Hora do Sistema.
df	Mostra o espaço livre/ocupado de cada partição (ver anexo A.4 página 99).
du	Mostra o espaço ocupado por arquivos e sub-diretórios do diretório atual.
free	Mostra detalhes sobre a utilização da memória RAM do sistema.
uptime	Mostra o tempo de execução do sistema desde que o computador foi ligado.
dmesg	Mostra as mensagens de inicialização do kernel.
reboot	Reinicia o computador.
shutdown	Desliga (reinicia) o computador imediatamente ou após determinado tempo, de forma segura.
uname	Mostra qual é a versão do *kernel* utilizada, indicados por números, onde números pares identificam versões testadas e consideradas estáveis, enquanto que números ímpares identificam versões de desenvolvimento, onde novos recursos estão sendo testados.

3.3.3 Diversos

Comando	Descrição
clear	Limpa a tela e posiciona o cursor no canto superior esquerdo do vídeo.
ln	Cria *links* para arquivos e diretórios no sistema. O *link* é um mecanismo que faz referência a outro arquivo ou diretório em outra localização.
find	Procura por arquivos/diretórios no disco através de sua data de modificação, tamanho, etc, ao contrário de outros programas, usa opções longas através de um “-”.
grep	Procura por um texto dentro de um arquivo(s) ou no dispositivo de entrada padrão, ou seja, é utilizado como um filtro.
cat	Mostra o conteúdo de um arquivo binário ou texto.
more	Permite fazer a paginação de arquivos ou da entrada padrão. Para sair do pressione *q*.
less	Semelhante ao more, mas permite que você pressione as setas para cima e para baixo ou *Page UP/Page Down* para fazer o rolamento da página. Para sair do pressione *q*.

sort	Ordena as linhas de um arquivo texto ou da entrada padrão.
tail	Mostra as linhas finais de um arquivo texto.
echo	Mostra mensagens. Este comando é útil na construção de *scripts* para mostrar mensagens na tela para o usuário acompanhar sua execução.
su	Permite o usuário mudar sua identidade para outro usuário sem fazer o *logout*.
wc	Conta o número de palavras, bytes e linhas em um arquivo ou entrada padrão.
cut	Mostra seções de cada linha do arquivo dependendo das opções passadas ao programa.

3.3.4 Rede

Comando	Descrição
who	Mostra quem está atualmente conectado no computador.
ftp	Permite a transferência de arquivos do computador remoto/local e vice-versa
hostname	Mostra ou muda o nome de seu computador na rede.
ping	Verifica se um computador está na rede, enviando pacotes para um computador, que quando recebe o pacote retorna uma resposta ao endereço de origem avisando que está disponível na rede.
traceroute	Mostra o caminho percorrido por um pacote para chegar ao seu destino.
netstat	Mostra conexões de rede, tabela de roteamento, estatísticas de interfaces, conexões *masquerade*, e mensagens.
wall	Envia uma mensagem a todos os usuários conectados ao sistema.

3.3.5 Alguns Exemplos

Exemplo 3.3.1 *Criando o diretório* public_html *na pasta do usuário* teste*:*

```
[root@curso root]# mkdir /home/teste/public_html
```

Exemplo 3.3.2 *Listando todos os arquivos, com detalhes, contidos no diretório* /home*, bem como nos sub-diretórios:*

```
[root@curso root]# ls -la /home/*
```

Exemplo 3.3.3 *Visualizar em qual linha do arquivo* smb.conf *do diretório* etc/samba *se encontra a palavra* security

```
[root@curso root]# grep security -n /etc/samba/smb.conf
```

Exemplo 3.3.4 *Verificar quantas vezes o computador foi ligado:*

```
[root@curso root]# last reboot | wc -l
```

Exemplo 3.3.5 *Exibir uma listagem das vezes que o usuário root se logou no servidor:*

```
[root@curso root]#  cat /var/log/messages | grep 'user root'
```

CAPÍTULO 4

Criando Usuários no GNU/Linux

Este capítulo traz comandos usados para manipulação de conta de usuários e grupos em sistemas GNU/Linux, bem como um detalhamento dos arquivos relacionados às contas. Entre os assuntos descritos aqui estão adicionar usuários ao sistema, adicionar grupos, incluir usuários existente em novos grupos, etc.

Como o GNU/Linux é um sistema operacional multi-usuário, vários usuários podem utilizar o sistema ao mesmo tempo, e para isto ocorra é necessário que ter uma conta, juntamente com uma senha.

Nas subseções seguintes serão dadas os principais comando a manipulação das contas dos usuários e portando deve ser utilizado com o usuário *root*.

4.1 Criando e Removendo Contas

Comando	Descrição
adduser	Adiciona um usuário ao sistema, criando um grupo com o mesmo nome do usuário e dentro do diretório `/home` um diretório com o nome do usuário.
passwd	Muda a senha do usuário. Um usuário somente pode alterar a senha de sua conta.
userdel	Apaga todos os dados da conta do usuário especificado, dos arquivos de contas do sistema.

4.2 Criando e Removendo Grupos

Comando	Descrição
addgroup	Adiciona um novo grupo de usuários no sistema.
passwd	Muda a senha do grupo. Os donos de grupos também podem alterar a senha do grupo com este comando.
groupdel	Apaga todos os dados do grupo especificado, dos arquivos de contas do sistema.

4.3 Outros Comandos para Contas e Grupos

Comando	Descrição

lastlog	Mostra o último *login* dos usuários cadastrados no sistema.
last	Mostra uma listagem de entrada e saída de usuários no sistema. A listagem é obtida do arquivo `/var/log/wtmp`.
chfn	Muda os dados usados.
id	Mostra a identificação atual do usuário, grupo primário e outros grupos que pertence.
logname	Mostra seu *login* (username).
groups	Mostra os grupos que o usuário pertence.

4.4 Arquivo passwd

As informações relacionadas com as contas de todos os usuários do sistema, são guardadas no arquivo `/etc/passwd`. Este arquivo tem o seguinte formato:

```
username:password:UID:GID:GECOS:diretory:shell
```

Onde cada campo tem o seguinte significado:

Campo	Descrição
username	nome da conta.
password	corresponde a senha criptografada, que nos sistemas atuais é apenas o `x` indicando que a senha está no arquivo `/etc/shadow`.
UID	equivale a um número que identifica o usuário no sistema, cada usuário tem um número diferente.
GID	equivale a um número que identifica o grupo principal do usuário.
GECOS	Este campo é opcional, e contém informações extras a respeito do usuário tais como nome, telefone, etc.
diretory	corresponde ao diretório padrão do usuário.
shell	especifica o nome do interpretador de comando (*shell*) do usuário (ver a seção 3.1 na página 19).

4.5 Arquivo shadow

Além de aumentar a segurança, o `shadow` proporciona ao administrador alguns recursos de gerenciamento de senhas. O formato de cada linha do arquivo `/etc/shadow` é da forma:

```
username:password:changed:min:max:warn:inactive:close:reserved
```

Onde cada campo tem o seguinte significado:

Campo	Descrição
username	é o nome do usuário para *login*
password	é a senha criptografada
hanged	é data em que a senha foi modificada pela última vez, escrita na forma do número de dias, desde 1 de janeiro de 1970 até à data da alteração.
min	é o número de dias, após a última modificação da senha, que a mesma não pode ser alterada.
max	é o número de dias após a última modificação da senha, que a mesma expira, ou seja, após este prazo, o usuário deve altera-la.
warn	é o número de dias antes da expiração da senha, no qual notifica-se ao usuário que a senha está expirando.
inactive	é o número de dias após a expiração da senha antes que a conta seja bloqueada. Uma vez bloqueada a conta, o usuário não pode efetuar *login* e, conseqüentemente, alterar a senha.

close	é a data em que a conta será fechada, expressa em número de dias após 1 de janeiro de 1970.
reserved	é um campo reservado para uso do sistema.

4.6 Arquivo group

As informações relacionadas com todos os grupos das contas dos usuários são guardadas no arquivo `/etc/passwd`. Este arquivo tem o seguinte formato:

`groupname:password:GID:users`

Onde cada campo tem o seguinte significado:

Campo	Descrição
groupname	nome de um grupo de usuários.
password	corresponde a senha criptografada do grupo, caso esteja utilizando senhas ocultas para grupos, as senhas estarão em `/etc/gshadow`.
GID	identificação numérica do grupo de usuário.
users	lista de usuários que também fazem parte daquele grupo. Caso exista mais de um nome de usuário, eles devem estar separados por vírgula.

Exemplo 4.6.1 *Para acrescentar o usuário joao ao grupo root, acrescente o nome no final da linha:* `root:x:100:joao`.

4.7 Exercícios

Nos exercícios abaixo, utilizar apenas os comandos, ou seja, sem utilizar diretamente os arquivos `passwd` e `group`.

Exercício 4.7.1 *Criar três grupos:* `chefe`, `diretores` *e* `outros`.

Exercício 4.7.2 *Criar a conta* `teste` *com a senha* `teste`, *pertencente ao grupo* `outros` *e com shell* `bash`.

Exercício 4.7.3 *Criar a sua conta particular, pertencente ao grupo* `chefe` *e colocando todos os seus dados.*

Exercício 4.7.4 *Criar uma super conta com poderes igual ao do usuário* `root` *e tenha como diretório base o mesmo diretório do* `root`. *Não deve ser sua conta pessoal.*

Exercício 4.7.5 *Criar dois usuários sem permissão de acessar o servidor e que pertença ao grupo* `outros`.

CAPÍTULO 5

Acesso à Arquivos e Diretórios

Este capítulo pode se tornar um pouco difícil de se entender, então recomendo ler e ao mesmo tempo praticá-la para uma ótima compreensão. Use os exemplos para ajudá-lo a entender o sistema de permissões de acesso do ambiente GNU/Linux.

A permissão de acesso ao sistema de arquivos GNU/Linux é muito confiável e seguro, pois protege o sistema do acesso indevido de pessoas ou programas não autorizados, impedindo por exemplo que um programa mal intencionado, apague um arquivo que não deve, envie arquivos para outra pessoa ou forneça acesso da rede para que outros usuários invadam o sistema.

5.1 Donos, Grupos e Outros Usuários

O princípio da segurança no sistema de arquivos GNU/Linux é definir o acesso aos arquivos por *donos*, *grupos* e *outros* usuários respectivamente, onde:

Tipo	Descrição
dono	É a pessoa que criou o arquivo ou o diretório. O nome do dono do arquivo ou diretório é o mesmo do usuário usado para entrar no sistema GNU/Linux. Somente o dono pode modificar as permissões de acesso do arquivo. As permissões de acesso do dono de um arquivo somente se aplicam ao dono do arquivo ou diretório. A identificação do dono também é chamada de *user id* (UID). A identificação de usuário e o nome do grupo que pertence são armazenadas respectivamente nos arquivos `/etc/passwd` e `/etc/group` (ver seções 4.4 e 4.6 do capítulo anterior).
grupo	Para permitir que vários usuários diferentes tivessem acesso a um mesmo arquivo (já que somente o dono poderia ter acesso ao arquivo), este recurso foi criado. Cada usuário pode fazer parte de um ou mais grupos e então acessar arquivos que pertençam ao mesmo grupo que o seu (mesmo que estes arquivos tenham outro dono).
outros	É a categoria de usuários que não são donos ou não pertencem ao grupo do arquivo, ou seja, os demais usuários.

Cada um dos tipos acima possuem três tipos básicos de permissões de acesso que são:

Tipo	Descrição
r	Permissão de leitura para arquivos. Caso for um diretório, permite listar seu conteúdo.

w	Permissão de gravação para arquivos. Caso for um diretório, permite a gravação de arquivos ou outros diretórios dentro dele.
x	Permite executar um arquivo (caso seja um programa executável). Caso seja um diretório, permite que seja acessado.

As permissões de acesso a um arquivo/diretório podem ser visualizadas com o uso do simples comando:

```
[root@curso root]# ls -l
```

```
total 8
-rw-r--r--  1 root root   69 2004-09-08 14:10 inicial.txt
lrwxrwxrwx  1 root root   11 2004-09-08 22:49 teste.txt -> inicial.txt
drwx------  3 root root 4096 2004-09-08 22:01 tmp
```

As 3 principais letras (*rwx*) são agrupadas da seguinte forma:

$$l \underbrace{r\,w\,x}_{dono}\ \underbrace{r\,w\,x}_{grupo}\ \underbrace{r\,w\,x}_{outros}$$

- A primeira letra ou símbolo diz qual é o tipo do arquivo. Caso tiver um "`d`" é um diretório, um "`l`" um *link* a um arquivo no sistema e um "`-`" quer dizer que é um arquivo comum.
- Da segunda a quarta letra, dizem qual é a permissão de acesso ao dono do arquivo.
- Da quinta a sétima letra, diz qual é a permissão de acesso ao grupo do arquivo.
- Da oitava a décima letra, diz qual é a permissão de acesso para os outros usuários.

5.2 Comandos para Permissões de Arquivos

Comando	Descrição
chmod	Muda a permissão de acesso a um arquivo ou diretório. Com este comando você pode escolher se usuário ou grupo terá permissões para ler, gravar, executar um arquivo ou arquivos
chgrp	Muda o grupo de um arquivo/diretório.
chown	Muda dono de um arquivo/diretório. Opcionalmente pode também ser usado para mudar o grupo.
mv	Move ou renomeia arquivos e diretórios.

Exemplo 5.2.1 *Para dar permissão a que outro usuário do grupo possa modificar e que os demais usuários possam ler o arquivo, usaremos o comando:*

```
[root@curso root]# chmod g+w o+r arquivo
```

Exemplo 5.2.2 *Para retirar a permissão executar a para o grupo e os demais nada possam, usaremos o comando:*

```
[root@curso root]# chmod g-x a-rwr arquivo
```

Exemplo 5.2.3 *Para modificar o grupo de um arquivo, usaremos o comando:*

```
[root@curso root]# chgrp usuário arquivo
```

Exemplo 5.2.4 *Para modificar o dono e o grupo de um diretório e todos os seus arquivos, usaremos o comando:*

```
[root@curso root]# chown -r usuário:grupo diretório
```

5.2.1 Modo de permissão octal

Ao invés de utilizar os modos de permissão *g+r*, *u-rw*, no comando `chmod`, pode ser usado o modo *octal* para se alterar a permissão de acesso a um arquivo, que é um conjunto de oito números onde cada número define um tipo de acesso diferente.

É mais flexível gerenciar permissões de acesso usando o modo *octal* ao invés do comum, pois você especifica diretamente a permissão do dono, grupo, outros ao invés de gerenciar as permissões de cada um separadamente.

Abaixo a lista de permissões de acesso *octal*:

Número	Descrição
0	Nenhuma permissão de acesso. Equivalente a *-rwx*.
1	Permissão de execução (*x*).
2	Permissão de gravação (*w*).
4	Permissão de leitura (*r*).
1 + 2 = 3	Permissão de gravação e execução (*wx*).
1 + 4 = 5	Permissão de leitura e execução (*rx*).
2 + 4 = 6	Permissão de leitura e gravação (*rw*).
1+2+4 = 7	Permissão de leitura, gravação e execução. Equivalente a *+rwx*.

O uso de 3 deste números consecutivos, definem a permissão de acesso do dono, grupo ou outros usuários, respectivamente.

Exemplo 5.2.5 *Para definir a permissão de acesso dos outros usuários para somente execução, o acesso do grupo como leitura e execução e o acesso do dono como leitura, gravação e execução, basta executar:*

```
[root@curso root]# chmod 751 teste
```

Exemplo 5.2.6 *Para as permissões de acesso especiais:*

- *1000 = Salva imagem do texto no dispositivo de troca*
- *2000 = Ajusta o bit setgid na execução*
- *4000 = Ajusta o bit setuid na execução*

5.3 Exercícios

Exercício 5.3.1 *Criar no diretório* `/home` *os seguintes diretórios, com as seguintes características:*

Diretório	*dono*	*grupo*	*permissões do grupo*	*permissões dos outros*
netlogon	*root*	*root*	*total*	*apenas leitura*
arquivos	*root*	*chefe*	*total*	*nenhuma*
secreto	*root*	*diretores*	*escrita e execução*	*nenhuma*
lixo	*root*	*outros*	*total*	*total*
profiles	*root*	*root*	*total*	*total*
profilesNT	*root*	*root*	*total*	*nenhuma*

5.3.1 Permissões de Acesso Especiais

Além das três permissões básicas (*rwx*), existem permissões de acesso especiais (*stX*) que afetam arquivos executáveis e diretórios, como descrito abaixo:

Tipo	Descrição
s	Quando é usado na permissão de acesso do Dono, ajusta a identificação efetiva do usuário do processo durante a execução de um programa, também chamado de *bit setuid*. Não tem efeito em diretórios. Quando "s" é usado na permissão de acesso do Grupo, ajusta a identificação efetiva do grupo do processo durante a execução de um programa, chamado de *bit setgid*. É identificado pela letra "s" no lugar da permissão de execução do grupo do arquivo ou diretório. Em diretórios, força que os arquivos criados dentro dele pertençam ao mesmo grupo do diretório, ao invés do grupo primário que o usuário pertence. Ambos *setgid* e *setuid* podem aparecer ao mesmo tempo no mesmo arquivo ou diretório. A permissão de acesso especial "s" somente pode aparecer no campo Dono e Grupo.
S	Idêntico a "s". Significa que não existe a permissão "x" (execução ou entrar no diretório) naquele lugar.
t	Salva a imagem do texto do programa no dispositivo *swap*, assim ele será carregado mais rapidamente quando executado, também chamado de *stick bit*. Em diretórios, impede que outros usuários removam arquivos dos quais não são donos. Isto é chamado de colocar o diretório em modo *append-only*. Um exemplo de diretório que se encaixa perfeitamente nesta condição é o `/tmp`, todos os usuários devem ter acesso para que seus programas possam criar os arquivos temporários lá, mas nenhum pode apagar arquivos dos outros. A permissão especial "t", pode ser especificada somente no campo outros usuários das permissões de acesso.
T	Idêntico a "t". Significa que não existe a permissão "x" naquela posição.
X	Se você usar "X" ao invés de "x", a permissão de execução somente é afetada se o arquivo já tiver permissões de execução. Em diretórios ela tem o mesmo efeito que a permissão de execução "x".

Parte II

Instalando, Configurando e Executando Serviços

CAPÍTULO 6

Ativar e Desativar Serviços

Neste capítulo daremos um breve entendimento dos procedimentos de ativar (*startup*) e desativar (*shutdown*), bem como dos arquivos de configuração envolvidos dão uma idéia do que pode estar errado caso o sistema não inicialize ou não desligue corretamente.

A inicialização do computador faz-se da seguinte maneira: primeiramente o BIOS, inicializa todos os dispositivos do sistema, e lê o primeiro setor, chamado de setor de *boot*, de um disquete ou do disco rígido[1]. Neste setor está contido o carregador de *boot*, que se encarrega de colocar o sistema operacional de algum lugar do disco (ou de outro dispositivo) na memória e executá-lo.

É possível ter várias partições e com isso vários sistemas operacionais no mesmo disco, cada qual com o seu setor de *boot*, e por conseguinte o seu carregador de *boot*. Para isto é necessário um programa que nos permita escolher qual dos setores de *boot* queremos inicializar. Este programa é chamado de gerenciador de *boot*, e em geral cada sistema operacional tem um gerenciador próprio, sendo que no caso do GNU/Linux existem o GRUB (no nosso caso) e o LILO[2].

Após o GNU/Linux ser carregado, ele inicializa o hardware e os *device drivers* e roda o programa *init*, que inicializa outros processos que fazem com que o sistema fique pronto para ser usado. O conjunto de processos que serão inicializado pelo programa *init*, definem o que chamamos de níveis de execução (*runlevels*). No arquivo `/etc/inittab` (mais especificamente na linha *id:número:initdefault:*) definimos sob qual nível irá iniciar, em geral existem seis níveis. Na tabela a seguir temos uma breve descrição de cada um deles.

Nível	Descrição
0	Faz com que *init* desligue o computador
1	É usado para colocar o sistema em modo monousuário (*single*), que é usado pelo administrador para fazer manutenção.
2	O sistema entra no modo multiusuário com maior parte dos serviços ativos, com exceção dos processos de rede.
3	É o modo padrão, onde todos os serviços estão disponíveis.
4	Este modo não é usado pela maioria das distribuições. É reservado para que possamos desenvolver o nosso próprio *runlevel*.
5	Semelhante ao nível 3, com todos o processo ativos, porém com uma interface gráfica de *login*.
6	Reinicializa o sistema.

Para saber em que nível o seu sistema esta rodando, basta executar o comando:

```
[root@curso root]# /sbin/runlevel
```

[1] No caso dos discos rígidos tem a denominação de MBR (*Master Boot Record*)

[2] *Linux Loader*

Este comando irá consultar o arquivo `/var/run/utmp` para determinar o estado atual e o anterior. Caso o estado anterior não possa ser determinado é exibida a letra *N* em seu lugar. Também é possível ao superusuário alterar o nível de execução do sistema, bastando para isso executar o comando `telinit` seguido de um número que indica em que nível o sistema operacional ficará.

Exemplo 6.0.1 *Para rebootar o sistema usaríamos o seguinte comando:*

```
[root@curso root]# telinit 6
```

Exemplo 6.0.2 *Para saber quais são os serviços, que serão ativados ou desativados no nível 6:*

```
[root@curso root]# ls -l /etc/rc.d/rc6.d
```

Para conhecer que conjunto de serviços são executados, em cada nível, basta listar o conteúdo dos diretórios `/etc/rc.d/rc<N>.d`, onde `N` é um dos seis níveis existentes.

Como se pode ver, todo o conteúdo do diretório `/etc/rc.d/rc6.d` consiste de *links* simbólicos apontando para *scripts* dentro do diretório `/etc/rc.d/init.d`. A primeira letra dos nomes dos *links* simbólicos pode ser ou "`S`" ou "`K`", indicando se o processo para o qual aponta deve ser ativado (*Started*) ou desativado (*Killed*). O número que se segue a esta letra indica a ordem em que os processos devem ser encerrados ou ativados.

Cada um dos *scripts* aceita geralmente três principais parâmetros: `start`, `stop`, `restart`. Estes parâmetros indicam, respectivamente, a ativação, desativação e reinicialização do processo. Esses parâmetros são úteis quando alguma modificação precisa ser feita na configuração de algum dos serviços e queremos que elas façam efeitos sem termos que reinicializar o computador.

Exemplo 6.0.3 *Suponhamos que fizemos uma alteração na configuração da rede, e queremos que elas surtam efeito, para tanto usaríamos o seguinte comando:*

```
[root@curso root]# service network restart
```

6.1 Definição ou Remoção de Processos Residentes

Para desativar um serviço de um determinado nível de execução basta remover o *link* simbólico do diretório apropriado.

Exemplo 6.1.1 *Para desativar o serviço httpd, do nível de execução 3, basta remover o link S85httpd, com o comando:*

```
[root@curso root]# rm /etc/rc.d/rc3.d/S85httpd
```

Exemplo 6.1.2 *Para inserir um novo serviço, basta criar um link no diretório que se deseja, apontando para o script correspondente em* `/etc/rc.d/init.d`*, usando o comando:*

```
[root@curso root]# ln -s /etc/rc.d/init.d /etc/rc.d/rc3.d/S100test
```

Este *script* é fictício, mas caso existisse pela numeração (100) seria o último a ser ativado. É importante, certificar-se na escolha de uma numeração que posicione a ativação do *script* na ordem correta. Se o serviço é dependente do funcionamento da rede ele deve necessariamente ser ativado após estes serviços estarem ativos.

6.2 Utilitários para Configuração dos Níveis de Execução

Até agora foi abordada configuração manual dos *scripts* de inicialização. Existem entretanto diversos utilitários para realizar este trabalho. Os mais usados são:

Comando	Descrição
ntsysv	Utilitário de linha de comando, com uma interface amigável, que exibe e permite alterar os serviços ativados ou não no nível de execução corrente.
chkconfig	Utilitário invocado a partir da linha de comando que recebe como parâmetros o nível de execução, o serviço e se o mesmo deve ficar ativo ou não no nível especificado.
linuxconf	Ferramenta genérica de configuração que possui um módulo para o gerenciamento dos níveis de execução.

Exemplo 6.2.1 *Para visualizar quais e em qual nível estão configurados os serviços, basta usar o comando:*

```
[root@curso root]# chkconfig --list
```

6.3 Procedimento de shutdown

Depois dessa longa descrição do *startup* do GNU/Linux se faz necessário algumas palavras a respeito do que acontece durante o procedimento de `shutdown`. Quando o sistema recebe o comando `shutdown` para fazer o desligamento, todos os sistema de arquivos (exceto o "`/`") são desmontados, os *logins* são desabilitados, os processos de usuários e os serviços são desativados. Logo após é mostrada uma mensagem na tela dizendo que você pode desligar o computador.

Esta seqüência é justificável devido a natureza multitarefa do GNU/Linux, onde existe uma grande quantidade de tarefas sendo executadas em *background*[3], muita dessas tarefas estão escrevendo ou lendo dados do disco rígido para a memória, dessa forma um desligamento sem que se faça o descarregamento para o disco do conteúdo da memória pode ser desastroso.

Exemplo 6.3.1 *Para desligar o computador em 10 minutos e mandando para os usuário uma mensagem, usaríamos o seguinte comando:*

```
[root@curso root]# shutdown -h +10 'Sistema suspenso por 10 min.'
```

Existe ainda variações do `shutdown`. Dois exemplo delas são o `reboot` e o `halt`, que correspondem ao `shutdown -r now` e ao `shutdown -h now`. Pode-se ainda usar o `init` com os argumentos 0 ou 6, conforme foi visto anteriormente.

Exemplo 6.3.2 *Para simplesmente desligar o servidor:*

```
[root@curso root]# poweroff
```

[3]São processos que não interagem com o usuário.

CAPÍTULO 7

Instalando Pacotes via APT

Neste capítulo daremos uma noção de como instalar e adaptar programas (pacotes), usando o `apt-get`, que é uma ferramenta de fácil utilização.

O APT (*Advanced Package Tool*), é um conjunto de ferramentas utilizadas para gerenciar pacotes de forma automatizada, capaz de resolver automaticamente dependências de pacotes, utilizar e criar repositórios de pacotes. O APT pode ser utilizado para instalar, remover ou atualizar um aplicativo, verificar conflitos de versões de aplicativos instalados, e até mesmo fazer a atualização automática de todo o seu sistema.

7.1 Configurando o APT

O APT utiliza repositórios de arquivos RPM (ver anexo A.6 na página 103) ao instalar ou atualizar pacotes no sistema. Esses repositórios podem estar na Internet, no CD-ROM da distribuição, ou podem ser construídos pelo administrador. A configuração da localização desses repositórios é feita através do arquivo `/etc/apt/sources.list`.

7.1.1 Nosso caso

O nosso arquivo `sources.list`, utilizará o repositório espelho da UNICAMP, pois a mesma também pertence à RNP (Rede Nacional de Pesquisa), o que garante uma maior facilidade e velocidade na instalação via rede e terá as seguintes linhas, que deverão ser modificadas com o editor `vi` (ver anexo A.2 na página 97), com o comando:

```
[root@curso root]# vi /etc/apt/sources.list
```

Exemplo 7.1.1 *Arquivo que utilizaremos no curso:*

```
rpm cdrom:[Conectiva Linux 10 CD 1]/ conectiva 001
rpm [cncbr] ftp://ftp.unicamp.br/pub/conectiva 10/i386 all extras
rpm-src [cncbr] ftp://ftp.unicamp.br/pub/conectiva 10/i386 all extras
rpm [cncbr] ftp://ftp.unicamp.br/pub/conectiva/atualizacoes 10/i386 updates
rpm-src [cncbr] ftp://ftp.unicamp.br/pub/conectiva/atualizacoes 10/i386 updates
```

7.2 Usando o apt-cdrom

O `apt-cdrom` é utilizado para adicionar o CD-ROM ao arquivo `sources.list`, fazendo com que o `apt-get` procure no seu CD-ROM por pacotes. Sua sintaxe é:

```
[root@curso root]# apt-cdrom add
```

O apt-cdrom usa informação do arquivo /etc/fstab, caso você não especifique onde está o drive de CD-ROM. Após a execução do comando, uma linha será incluída no seu arquivo sources.list (ver o exemplo anterior).

7.3 Utilizando o apt-get

O apt-get é uma das ferramentas do APT e provavelmente é o comando que será utilizado com mais freqüência no nosso curso. Os comandos do apt-get são muito fáceis e intuitivos. Eles seguem uma estrutura muito simples:

```
[root@curso root]# apt-get opções pacote [pacote2 pacote3...]
```

A linha de comando pode ter uma variação das opções básicas a seguir:

Opções	Descrição
update	Atualiza o banco de dados local do apt-get com os arquivos (*pkglist*), que contém a lista de pacotes, dependências e informações encontradas no servidor. Este comando deve ser executado sempre antes de se utilizar o comando apt-get com outro parâmetro.
check	Verifica a integridade do seu sistema. Execute este comando quando tiver dúvidas quanto à integridade dos pacotes do seu sistema. É recomendável executá-lo antes de executar uma atualização de distribuição.
install	Instala o pacote, solucionando e carregando automaticamente os pacotes dos quais o aplicativo a ser instalado depende. Caso o pacote já esteja instalado, o apt-get tentará atualizá-lo.
source	Faz o download dos fontes do pacote (SRPM). Note que é necessário que haja uma linha com o tipo *rpm-src* no arquivo sources.list para que este comando seja executado.
upgrade	Procura por pacotes desatualizados no sistema e os atualiza automaticamente. Atualizará todos os pacotes antigos no sistema. Para atualizar apenas um pacote, utilize o parâmetro install.
dist-upgrade	Semelhante ao upgrade, mas instala todos os pacotes básicos e tenta atualizar tudo, instalando novos pacotes caso seja necessário. É uma maneira mais fácil de fazer uma atualização de sua distribuição.
remove	Remove o pacote e todos os demais pacotes que dele dependam.
clean	Remove os arquivos encontrados no diretório *cache*, localizado em /var/cache/apt/archives, liberando um pouco de espaço no seu disco de sistema. É uma maneira automática de apagar os arquivos que já foram instalados e que não são mais necessários ao sistema.

Exemplo 7.3.1 *Para atualizar o banco de dados de todos os pacotes do Conectiva 10, deveremos utilizar o comando:*

```
[root@curso root]# apt-get update
```

Exemplo 7.3.2 *Para instalar os pacotes dos editores de texto* pico *e* joe *e o utilitário* ntsysv*, em uma única linha de comando, deveremos utilizar:*

```
[root@curso root]# apt-get install pico joe ntsysv
```

CAPÍTULO 8

Configurando a Rede

Como a instalação foi toda baseada na rede, a configuração do acesso à rede externa, já está configurada, porém vale ressaltar que para o desenvolvimento do curso, sempre estaremos olhando para o nosso servidor GNU/Linux, com duas placas de rede, no sentido de que, ele será a ponte entre a rede externa e interna, conforme ilustrado no diagrama abaixo:

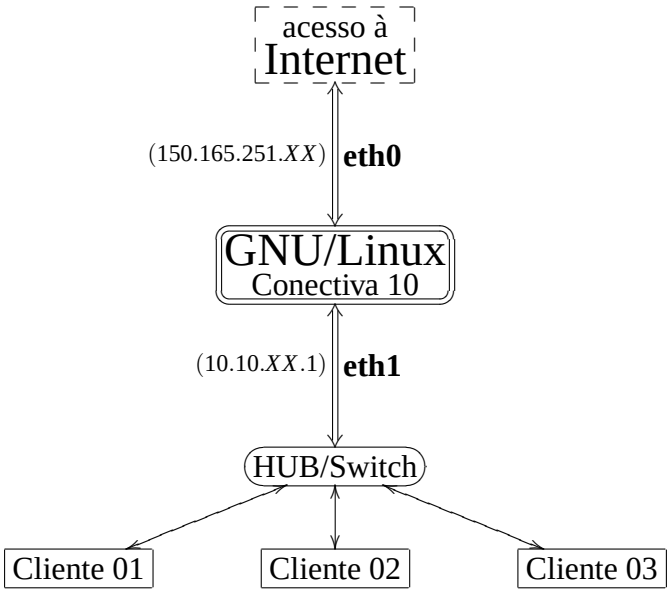


Figura 8.1: Diagrama da nossa rede

onde **eth0** é a primeira placa de rede (rede externa), **eth1** é a segunda placa de rede (rede interna).

8.1 Rede Externa - eth0

A rede externa terá todos os números, dado pelo servidor de DHCP (ver capítulo 16 na página 72) do NTI, ou seja:

- Número IP: **150.165.250.YYY** fornecido pelo servidor DHCP.
- Mascara: **255.255.255.0** fornecida pelo servidor DHCP.
- Placa de rede: **eth0**

Para checar qual foi o número fornecido, usaremos o comando:

```
[root@curso root]# ifconfig
```

Mas para efeito didático e de configurações futuras, iremos redefinir a nossa rede (placa) externa com as seguintes características:

- Número IP: **150.165.251.XX**, onde **XX** será um número escolhido pelo instrutor durante a instalação.
- Mascara: **255.255.255.0**
- Placa de rede: **eth0**
- Rota padrão: **150.165.251.1**
- Nome do domíno: **cursoXX.linux.nti.ufpb.br**

Estas modificações serão feitas nos arquivo `network` e`ifcfg-eth1` com os respectivos comandos:

```
[root@curso root]# pico /etc/sysconfig/network

NETWORKING=yes
HOSTNAME="curso.cursoXX.linux.nti.ufpb.br"
GATEWAY="150.165.251.1"

[root@curso root]# pico /etc/sysconfig/network-script/ifcfg-eth1

DEVICE=eth0
ONBOOT=yes
BOOTPROTO=none
IPADDR=150.165.251.XX
NETMASK=255.255.255.0
NETWORK=150.165.251.0
BROADCAST=150.165.251.254
```

8.2 Rede Interna - eth1

Consideraremos que a rede interna terá as seguintes características:

- Número IP: **10.10.XX.1**, onde **XX** será um número escolhido pelo instrutor durante a instalação.
- Mascara: **255.255.255.0**
- Placa de rede: **eth1**

8.3 Instalando a Segunda Placa de Rede

Antes de mais nada devemos configurar o VMware (ver anexo A.3 página 98) para que adicione uma segunda placa de rede no nosso servidor, onde esta placa deverá ser do tipo *host only*, pois queremos uma rede privada, conforme o diagrama acima (ver o diagrama 8.1).

Para configurar, editaremos o arquivo com o comando:

```
[root@curso root]# pico /etc/sysconfig/network-script/ifcfg-eth1
```

```
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=none
IPADDR=10.10.XX.1
NETMASK=255.255.255.0
NETWORK=10.10.XX.0
BROADCAST=10.10.10.254
```

Vamos comparar com os arquivos de configuração das outras redes existentes, da seguinte maneira:

```
[root@curso root]# more /etc/sysconfig/network-script/ifcfg-*
```

E para finalizar, devemos informar ao *kernel* que deve adicionar o módulo da placa de rede *pcnet32*, adicionando a linha `alias eth1 pcnet32` ao arquivo `/etc/modprob.conf`.

8.4 Ativando a Segunda Placa de Rede

Depois de feitas as configurações, vamos ativá-las com o comando:

```
[root@curso root]# ifup eth1
```

ou reativando todo o serviço de rede, com:

```
[root@curso root]# service network restart
```

Verificar se tudo ocorreu como o esperado, executando o seguinte comando:

```
[root@curso root]# ifconfig
```

Se tudo estiver certo aparecerá algo do tipo:

```
lo  Encapsulamento do Link: Loopback Local
    inet end.: 127.0.0.1  Masc:255.0.0.0
    UP LOOPBACKRUNNING  MTU:16436  Métrica:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
    colisões:0
    RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

eth0  Encapsulamento do Link: Ethernet  Endereço de HW 00:50:56:C0:00:01
      inet end.: 150.165.251.XX  Bcast:150.165.251.255  Masc:255.255.255.0
      UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      colisões:0
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

eth1  Encapsulamento do Link: Ethernet  Endereço de HW 00:50:56:C0:00:01
      inet end.: 10.10.XX.1  Bcast:10.10.XX.255  Masc:255.255.255.0
      UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      colisões:0
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

8.5 Configurando a rede do cliente Windows

Esta seção descreve a reconfiguração da rede de um cliente Windows 98, para tanto, basta:

1. configurar o VMware (ver anexo A.3 página 98), trocando a placa de rede da opção *bidged* para *host only*;

2. iniciar o Windows, e em *Painel de Controle / Rede / Configuração / TCP-IP / Propriedades / Endereço IP* modificar a opção *obter IP* por especificando um IP com o número IP **10.10.XX.10** e máscara **255.255.255.0** e rota padrão **10.10.XX.1**;

3. salvar as modificações e reiniciar o Windows.

Para testar no Windows usar o comando de linha:

```
C:\WINDOWS> winipcfg
```

8.6 Testando a conexão entre o GNU/Linux e Windows

No GNU/Linux basta usar o comando `ping` que serve para mandar um pacote para o computador destino e caso receba um pacote, significa que o computador destino está na rede, e para tanto basta utilizar da seguinte forma:

```
[root@curso root]# ping 10.10.XX.10
```

No Windows é o mesmo comando, em um *prompt-DOS*, ou seja:

```
C:\WINDOWS> ping 10.10.XX.1
```

8.7 Outros Comandos de Rede

Comando	Descrição
ifconfig	configura e exibe o estado das placas de rede.
ifup/ifdown	ativa e desativa uma determinada placa de rede.
route	exibe e ou modifica a tabela de roeamento IP.
netstat	mostra as conexões de rede, tabelas de roteamento e estatísticas das placas de rede.

8.8 Exercícios

Exercício 8.8.1 *Abrir no Windows, uma janela com o prompt-DOS, executando um ping infinito para o servidor. Ao mesmo tempo desativar e ativar a placa de rede interna do servidor.*

Exercício 8.8.2 *No servidor, executar um ping infinito para o computador do seu vizinho, em outro terminal um ping para um computador externo e em um outro terminal desativar e ativar a placa de rede externa.*

Exercício 8.8.3 *Visualize todas as rotas existentes em seu servidor e no seu Windows com o comando* `route`*, claro que colocando alguns parâmetros.*

CAPÍTULO 9

Servidor de Acesso Remoto - SSH

O SSH (*Secure Shell*) é um programa de acesso remoto criptografado (ver seção 9.9), que substitui o `telnet`, `rshell` e `rsh`, pois estes não possuem criptografia, ou seja, a transferência de dados, inclusive o *login* e senha, pode ser capturada e lida por terceiros sem nenhum problema. Já com o SSH pode-se capturar os dados, mas estes estão criptografados com uma chave aleatória, e que dificulta a leitura por terceiros.

9.1 Pré-requisitos

Para uma implementação bem-sucedida de um servidor de SSH é necessário apenas que sua rede esteja corretamente configurada.

9.2 Instalação

Para instalarmos o SSH, precisamos dos seguintes pacotes:

- openssh: Arquivos necessários ao SSH cliente e servidor.
- openssh-clients: Cliente.
- openssh-server: Servidor.
- openssl: Biblioteca de criptografia.

A instalação se dá utilizando o comando:

```
[root@curso root]# apt-get install openssh-server openssh-clients
```

9.3 Configuração

A configuração padrão já é funcional, porém existem algumas opções importantes no arquivo `sshd_config` no diretório `/etc/ssh`, que devem ser consideradas:

Opção	Descrição
Port	Especifica a porta na qual o servidor SSH receberá as conexões dos cliente (padrão é **22**).
Allow Users	Especifica a lista de usuários que podem conectar ao servidor.
Permitrootlogin	permitir o acesso remoto com a conta *root*, que é altamente recomendável que não seja permitido.

9.4 Ativando

O *script* de inicialização do servidor SSH, contido no diretório `/etc/rc.d/init.d`, é o `sshd` e para ativar o serviço basta executar o comando:

```
[root@curso root]# service sshd start
```

Observação 9.4.1 *Não esquecer de colocar o serviço SSH nos níveis de execução com o comando* `ntsysv`*, para que na próxima vez que o servidor for ligado, garantir que o serviço será ativo automaticamente.*

9.5 Cliente SSH - GNU/Linux

O cliente é usado da através do comando:

```
[root@curso root]# ssh teste@150.165.250.yyy
```

onde *teste* é o nome do usuário que deve ter permissão de acesso e *150.165.250.yyy* é o endereço IP do servidor SSH a qual se deseja acessar.

Após a validação do usuário, obtém-se um terminal remoto, ou seja, os comandos digitados são executados no servidor SSH.

9.6 Cliente SSH - Windows

Para acessar um servidor a partir de um Windows, deve-se usar algum programa *SSH-Client*, do qual recomendo o *PuTTY*.

O *PuTTY* é uma implementação gratuita de cliente para plataformas *Win32* de:

- *SSH* com acesso normal ou criando um túnel entre as redes.
- *telnet*
- *rlogin*
- emulador de terminal *xterm*.

Ele é escrito e mantido principalmente por *Simon Tatham*[1] e está na versão *0.55 beta*. Outros utilitários complementares do pacote *PuTTY*[2]:

Utilitário	Descrição
PSCP	cliente SCP (cópia segura).
Plink	executar um programa no servidor.
PSFTP	um cliente de SFTP (ftp seguro).
PuTTYtel	Telnet somente de um cliente.
PuTTYgen	autenticador do SSH para PuTTY, PSCP e Plink.
Pageant	geração de uma chave RSA e de DSA.

O programa *PuTTY*, bem como o pacote completo para Windows (9x/ME/NT/2000/XP) em *Intel x86* e também disponível para NT em Alpha, pode ser baixado da internet da página oficial do *PuTTY*, que é `http://www.chiark.greenend.org.uk/~sgtatham/putty`.

[1] `anakin@pobox.com`

[2] O uso do *PuTTY*, *PSCP*, *PSFTP* e *Plink* é ilegal em países onde a lei restringe o uso de criptografia. O programa somente de *Telnet* (*PuTTYtel*) tem uso irrestrito, pois ele não utiliza criptografia.

9.7 Cliente SCP

Este programa é usado na transferência de arquivos. A sua forma de uso é semelhante ao comando `cp`, no qual pode-se informar como origem ou destino, mas não ambos, qual servidor SSH e qual arquivo.

Exemplo 9.7.1 *Para copiar o arquivo* `seunome.txt` *do seu servidor para o usuário teste existente no servidor do instrutor (IP 150.165.250.PPP), basta usar o seguinte comando:*

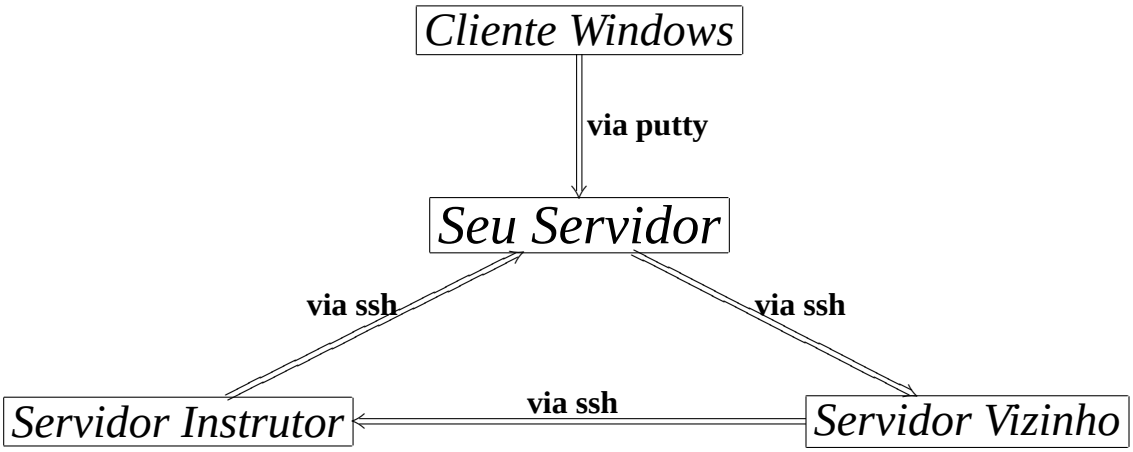
```
[root@curso root]# scp seunome.txt teste@instrutor:/home/teste/.
```

Exemplo 9.7.2 *Para copiar o arquivo* `remoto.txt` *localizado no diretório da conta teste no servidor do instrutor (IP 150.165.250.PPP), para o seu diretório no seu servidor basta usar o seguinte comando:*

```
[root@curso root]# scp teste@instrutor:/home/teste/remoto.txt .
```

9.8 Exercícios

Exercício 9.8.1 *Fazer uma triangulação com o acesso remoto com a conta teste, do tipo:*



Exercício 9.8.2 *Mudar a porta padrão do SSH para 2004 e testar.*

Exercício 9.8.3 *Entrar no servidor do instrutor com o login teste e trocar para o usuário root, com o comando* `su - root` *e criar sua conta pessoal neste servidor, pertencente ao grupo aluno. Teste a sua nova conta, acessando remotamente.*

9.9 Criptografia

A utilização de criptografia em comunicações não é algo novo, e já era utilizada antes mesmo de se existirem computadores. Criptografia é um conjunto de técnicas que permitem tornar incompreensível uma mensagem originalmente escrita com clareza, de forma a permitir que apenas o destinatário a decifre e compreenda, ou seja, na impossibilidade de manter informações de acessos não autorizados, a criptografia visa tornar uma mensagem ininteligível a quem não deveria vê-la.

É exatamente isso que o SSH faz ao se comunicar, ou seja, os pacotes contendo as informações a serem transmitidas passam por um processo de criptografia que visa proteger a informação contida neles, já que não há formas de conhecer os caminhos por onde esses pacotes trafegarão (principalmente em se tratando de Internet), nem o quão seguro será este caminho.

CAPÍTULO 10

Servidor de Nomes - DNS

O DNS é o *Servidor de Nomes do Domínio*, que converte os nomes das máquinas para números IP, que são os endereços das máquinas, fazendo uma correspondência de nome para endereço IP e de endereço IP para nome, ou seja, é simplesmente uma associação entre duas informações, neste caso um nome de máquina, como *ftp.linux.org*, e o número IP da máquina, como por exemplo *199.249.150.4*. Portanto o DNS é um banco de dados distribuído por toda a rede.

O DNS é um banco de dados distribuído por toda a rede. É necessário ter-se extremo cuidado com tudo o que for colocado nele. Ao se colocar dados sem significado, outros utilizarão estes dados e certamente tudo ficará um pouco *estranho*. O DNS deve estar sempre atualizado e arrumado, evitando-se assim problemas desagradáveis. Deve-se aprender a usá-lo, administrá-lo, depurá-lo para tornar-se um bom administrador da rede, evitando sobrecargas geradas por problemas de administração.

Porém, nesse curso, será dada uma breve introdução aos principais arquivos e a idéia básica do seu funcionamento.

10.1 Pré-requisitos

Para configurar um domínio, que no nosso caso, será o domínio `cursoXX`, você precisará de:

- Um servidor com uma placa de rede configurada. É importante que você certifique-se de que o nome da máquina e o domínio pertençam ao domínio que está sendo configurado;
- Um domínio registrado (opcional); quando o servidor DNS for usado somente para uma rede interna, não há necessidade de que o domínio (.com.br) seja registrado.

10.2 Instalação

Para configurar um domínio, você precisará usar dos pacotes `bind` e `bind-utils` e para instalá-los usaremos o seguinte comando:

```
[root@curso root]# apt-get install bind bind-utils
```

10.3 Configuração

Nesta seção serão exibidos os arquivos e diretórios que devem trabalhados, com a breve explicação para cada um:

Opção	Descrição

/var/named	este é o diretório principal. Nele ficam os arquivos das zonas e reversos (`localhost.zone, named.ca, named.local`).
named.conf	neste arquivo são definidas as zonas e os reversos para as mesmas.
named.local	Arquivo de Zone para o *loopback* (ou *localhost*), servindo apenas para configuração do mesmo.
localhost.zone	Reverso para a zona do *localhost* (*nslookup 127.0.0.1* deverá voltar como *localhost*).
named.ca	Arquivo usado para o *named* acessar servidores de DNS básicos da Internet.

Observação 10.3.1 *O domínio que será configurado será o* `cursoXX.linux.nti.ufpb.br`*, ou seja, será um sub-domínio do domínio* `nti.ufpb.br` *do qual o laboratório faz parte.*

Observação 10.3.2 *Em todos os arquivos de configuração, trocar o nome do computador* `curso.` *pelo nome* `servidorXX.`*, onde* `XX` *será o seu número.*

10.4 Criando Zonas

Nesta parte usaremos o `named.conf` e criaremos uma base de dados com informações sobre o domínio que queremos criar. Abaixo segue um exemplo:

```
options{
        directory "/var/named";
        listen-on { 127.0.0.1/32; };
};
zone "." {
        type hint;
        file "named.ca";
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "named.local";
};
```

Onde:

- a linha `directory` indica onde os arquivos devem estar localizados para o sistema, mas na realidade por motivos de segurança ele se encontra no diretório `/var/named/var/named`;
- o arquivo `named.ca` descreve o nome dos servidores raiz no mundo;
- o arquivo `named.local` define as características para o *localhost*.

Desta forma, o servidor de DNS será informado da existência da zona *localhost* e a seu reverso (*0.0.127.in-addr.arpa*).

Resta agora informar qual é a ordem de busca no computador local, que definiremos no arquivo `/etc/resolv.conf`, como mostrado abaixo:

```
search cursoXX.linux.nti.ufpb.br linux.nti.ufpb.br
nameserver 127.0.0.1
```

Será adicionado o domínio `cursoXX.linux.nti.ufpb.br`[1], que é fictício, pois, um domínio real exige um cadastro prévio no setor responsável. Portanto para criarmos tal domínio, adicionaremos ao arquivo `named.conf` as seguintes linhas:

[1] O *XX* será um número dado pelo instrutor do curso.

```
zone "cursoXX.linux.nti.ufpb.br" {
        notify no;
        type master;
        file "cursoXX.linux.zone";
};
zone "10.10.10.in-addr.arpa" {
        notify no;
        type master;
        file "cursoXX.linux.rev";
};
```

10.5 Criando um Domínio

Para editar o arquivo da base de dados do domínio que queremos, é necessário criar um arquivo auxiliar (para simplificar), com o nome `cursoXX.linux.soa` com as seguintes linhas:

```
; Arquivo cursoXX.linux.soa
;
@   IN  SOA     curso.cursoXX.linux.nti.ufpb.br. root.cursoXX.linux.nti.ufpb.br. (
                2004110302                          ; Data + Serial
                28800                               ; Atualização (segundos)
                7200                                ; Tentativa (segundos)
                604800                              ; Expiração (segundos)
                86400 )                             ; Minimum TTL (segundos)
    IN  NS      curso.cursoXX.linux.nti.ufpb.br     ; Servidor de nomes
    IN  A       150.165.251.XX                      ; IP do servidor
    IN  MX      10 curso.cursoXX.linux.nti.ufpb.br. ; Servidor de E-mail
```

As zonas de fato serão criados no arquivo `cursoXX.linux.zone`, exibido abaixo:

```
;       Definições locais (zonas)
;
$INCLUDE cursoXX.linux.soa
;
curso         A       150.165.251.XX
mail          A       10.10.XX.1
www           A       10.10.XX.1
win10         A       10.10.XX.10
              HINFO   "Celeron" "Win 98"
              TXT     "Cliente Win"
win11         A       10.10.XX.11
win12         A       10.10.XX.12
win13         A       10.10.XX.13
win14         A       10.10.XX.14
```

Uma vez editado e salvo o *named* deverá ser iniciado, com o comando:

```
[root@curso root]# service named start
```

Verifique se foi tudo configurado corretamente executando o comando:

```
[root@curso root]# nslookup -sil www.conectiva.com.br

Server:         127.0.0.1
Address:        127.0.0.1#53
```

```
Non-authoritative answer:
www.conectiva.com.br    canonical name = barney.conectiva.com.br.
Name:   barney.conectiva.com.br
Address: 200.140.247.101
```

ou o comando:

```
[root@curso root]# dig www.conectiva.com.br

; <<>> DiG 9.2.3 <<>> www.conectiva.com.br
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29446
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 0

;; QUESTION SECTION:
;www.conectiva.com.br.          IN      A

;; ANSWER SECTION:
www.conectiva.com.br.   35443   IN      CNAME   barney.conectiva.com.br.
barney.conectiva.com.br. 35442  IN      A       200.140.247.101

;; AUTHORITY SECTION:
conectiva.com.br.       35443   IN      NS      golfinho.telepar.net.br.
conectiva.com.br.       35443   IN      NS      ns.netbank.com.br.
conectiva.com.br.       35443   IN      NS      ns.conectiva.com.br.
conectiva.com.br.       35443   IN      NS      slave.convoy.com.br.

;; Query time: 26 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Nov  3 17:19:21 2004
;; MSG SIZE  rcvd: 179
```

Perfeito, se a saída for essa, sem erro algum. Execute o comando abaixo tentando resolver o domínio recém criado.

```
[root@curso root]# nslookup -sil cursoXX.linux.nti.ufpb.br

Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   cursoXX.linux.nti.ufpb.br
Address: 10.10.XX.1
```

Como podemos verificar, o domínio `cursoXX.linux.nti.ufpb.br` assumiu o IP 10.10.XX.1 definido em sua base de dados, assim como os nomes dos seus computadores da sua rede local (`win10.cursoXX.linux.nti.ufpb.br`) que poderão ser testados da mesma forma.

10.6 Criando Reverso

Após o domínio ter sido criado e ter assumido seu respectivo IP, é mais do que natural que desejemos que o mesmo faça o inverso (ou reverso), ou seja, dizemos o IP, e recebemos o domínio. Para isso é necessária a edição de mais um arquivo. O definido no `named.conf`, chamado de `cursoXX.linux.rev`, como o exemplo mostrado abaixo:

```
; Reverso para cursoXX.linux.nti.ufpb.br
;
$INCLUDE cursoXX.linux.soa
;
1             PTR     curso.cursoXX.linux.nti.ufpb.br.
10            PTR     win10.cursoXX.linux.nti.ufpb.br.
11            PTR     win11.cursoXX.linux.nti.ufpb.br.
12            PTR     win12.cursoXX.linux.nti.ufpb.br.
13            PTR     win13.cursoXX.linux.nti.ufpb.br.
14            PTR     win14.cursoXX.linux.nti.ufpb.br.
```

Assim como os outros, este arquivo ficará no diretório `/var/named`. Reinicie o *named* e tente executar o `nslookup` com o IP do domínio criado, e veja abaixo o resultado, caso sua configuração tiver sido feita com sucesso.

```
[root@curso root]# nslookup -sil 10.10.XX.1

Server:         127.0.0.1
Address:        127.0.0.1#53

10.XX.10.10.in-addr.arpa        name = win10.cursoXX.linux.nti.ufpb.br.
```

Observe que o IP retornou o domínio e você já pode também testar os sub-domínios criados, pois eles também estarão com o reverso funcionando.

Observação 10.6.1 *Deve-se sempre modificar o serial dentro do arquivo* `cursoXX.linux.soa` *no* `named` *quando os arquivos* `cursoXX.linux.rev` *e* `cursoXX.linux.zone` *forem modificados.*

CAPÍTULO 11

Servidor de Web - Apache

Devido ao importância que a Web tem hoje no mundo moderno, nenhuma discussão sobre administração de sistemas estará completa, sem que mencione algo sobre a instalação de servidor Web.

O servidor Web que será abordado e instalado neste curso, é o *Apache*[1], que é largamente utilizado no mundo todo. Esta liderança deve-se ao fato de ter como principais características: ser de baixo custo, ser altamente configurável, pode ser executado em diferentes plataformas, é flexível, está sempre em desenvolvimento para a inclusão dos protocolos mais atualizados, fornece o código-fonte completo e não possui licenças restritivas, pode ser configurado para diferentes funções, é composto de módulos, cada um implementando uma característica diferente e aumentando a funcionalidade do servidor, além de várias outras características.

11.1 Pré-requisitos

Para implantar esta solução é necessário que:

- a rede esteja corretamente configurada e funcionando;
- o serviço de DNS esteja corretamente instalado e configurado.

11.2 Instalação

Para instalação do servidor *web Apache* com a inclusão do módulo PHP4, são necessários os seguintes pacotes: `apache`, `php4` e `openssl-progs`.

Para instalá-los basta usar o seguinte comando:

```
[root@curso root]# apt-get install apache php4 openssl-progs
```

e caso se queira instalar a documentação[2], usar o comando:

```
[root@curso root]# apt-get install apache-doc
```

11.3 Configurando o Apache

A configuração do apache é um pouco complexa devido a enorme quantidade de opções disponíveis, mas a configuração instalada por padrão já lhe permite rodar o servidor com total fun-

[1] A palavra significa *A PAtCHy*, pois foi baseado em um código juntamente com uma série de arquivos patch. Para muitos desenvolvedores, porém, a palavra faz referência aos nativos americanos, ou seja, os índios Apache.

[2] A documentação está na língua inglesa, alemã, japonesa, coreana e russa.

cionalidade, bastando que você edite o arquivo `httpd.conf` (que contém mais de mil linhas), contido no diretório `/etc/apache/conf`, e configure o endereço de e-mail do administrador Web e o nome do seu servidor.

11.4 O arquivo httpd.conf

O arquivo `httpd.conf` é responsável pela configuração do serviço, e suas opções mais importantes são:

Opção	Descrição
ServerRoot	indica o diretório onde os arquivos de configuração, de log e de erros devem ser colocados, que no caso é o `/etc/apache` (linha 50).
ServerAdmin	indica o endereço de e-mail que é usado em caso de algum problema no serviço (linha 322).
ServerName	esta opção define o nome e a porta que o servidor usa par identificá-lo, que por padrão é o nome do computador na porta 80 (linha 336).
DocumentRoot	define o diretório do conteúdo da URL `http://seuservidor`, ou seja, é no diretório `/srv/www/default/html`, que se encontram os arquivos para a home-page (linha 352).
UserDir	define qual o diretório que conterá as páginas pessoais no diretório dos usuários, que geralmente é `public_html` (linha 412).
DirectoryIndex	define o nome do arquivo que é retornado, caso o cliente não especifique um nome de arquivo na URL, o padrão é `index.html` (linha 439).
AccessFileName	indica o nome do arquivo onde o apache procurará por informações sobre controle de acesso de um determinado diretório. Basicamente este arquivo serve para negar o acesso a determinadas paginas ou então condicionar o acesso a uma senha. O nome padrão do arquivo de acesso é `.htaccess` (linha 446).
ErrorLog	indica o arquivo onde o log das transações WWW será guardado, no caso, no arquivo *error_log* no diretório `/var/log/apache` (linha 519).

11.5 Inicializando

Para ativar o serviço utilizar o comando:

```
[root@curso root]# service httpd start
```

Para se certificar que o apache está funcionando, após a instalação e inicialização, inicie o navegador e entre com IP *10.10.XX.1* na barra de endereço, você deverá ver uma página de apresentação do Apache. Se esta página não for mostrada, verifique se existem erros contidos no arquivo `/etc/apache/logs/error_log` e tente configurar o Apache novamente.

Exemplo 11.5.1 *Para que o usuário* teste *possua uma página na internet, basta que o mesmo crie um diretório* `public_html` *e que dentro deste diretório exista o arquivo* `index.html`[3]*, como esse pequeno exemplo abaixo:*

```
<html><body>
   Bem vindo à minha primeira página.
</body></html>
```

[3]Note que a página tem acesso negado, e portanto o administrador têm que liberar o acesso, via comando `chmod` (ver seção 3.3 na página 20).

CAPÍTULO 12

Servidor de Proxy - Squid

Um servidor de *proxy* de modo geral, é um computador que intercepta as requisições dos clientes locais para um determinado serviço (*http, ftp,* etc) e faz ele mesmo o pedido ao *host* remoto e devolvendo as informações obtidas ao cliente, de maneira semelhante ao mostrado na figura abaixo:

Como todos os pedidos passam pelo *proxy*, é possível usá-lo como *cache* das informações pedidas. Assim se são feitos vários pedidos pelos clientes para a mesma informação, o proxy fará um único pedido ao servidor, retornará aos clientes esta informação e a armazenará.

No GNU/Linux existem vários programas para *proxy*, mas será abordado neste capítulo o SQUID. O programa SQUID é um proxy cache de HTTP e FTP, ou seja, ele armazenará as informações dos pedidos, para servidores HTTP (WWW) e FTP. O squid poderá ser configurado de 2 maneiras:

Proxy ordinário - No qual os clientes (ou melhor, os navegadores WEB) devem ser configurados para usar o proxy.

Proxy transparente - Neste caso, o *proxy* é instalado no roteador da rede, para que qualquer pedido seja redirecionado (ver seção 18.4.5 na página 82), pelo sistema operacional do roteador ao SQUID. Desta forma, não é necessário configurar os clientes.

12.1 Pré-requisitos

Para implementar o Squid é necessário:

- que o acesso à Internet esteja configurado corretamente;
- recomenda-se que o servidor tenha uma boa quantidade de memória (por exemplo, 128 MB) e que seja usado um disco rígido SCSI para permitir um acesso mais rápido aos arquivos armazenados no *buffer*.

12.2 Instalação

Para instalação do servidor proxy são necessários os seguintes pacotes: `squid`, `squid-auth` e `squid-extra-templates`.

Para instalá-los, basta usar o seguinte comando:

```
[root@curso root]# apt-get install squid squid-.*
```

12.3 Configurado o SQUID

O arquivo responsável pela configuração é o `squid.conf` que fica localizado no diretório `/etc/squid/`. Ele é um arquivo extenso contendo muitas opções agrupadas em seções, mas que contém também uma grande quantidade de comentários.

As opções padrão funcionam bem, mas a título de ilustração faremos algumas modificações.

Primeiramente será observada a opção `http_port 3128` que serve para especificar a porta que será usada pelo *squid* para receber as requisições dos clientes www (browsers)[1].

Modificar a opção `cache_mem` que especifica a quantidade de memória que estará disponível para o *squid*, usando o valor 8 MB, porém quanto maior for este valor melhor será a performance do servidor e modificar o terceiro campo da opção `cache_dir`, que especifica o tamanho do espaço em disco destinado ao *cache* do *squid*, para 10 MB, pois o padrão é 100 MB. É interessante também modificar a opção `error_directory`, que serve para exibir as massagens dadas pelo *squid* para a língua portuguesa, ou seja, trocando `English` por `Portuguese`.

O *squid* tem capacidade de permitir e restringir o acesso a determinadas *URLs* através da criação de *Listas de Controle de Acesso* (*ACL*) e do uso da opção it http_access. Para permitir o acesso ao serviço do *squid* à rede interna e negar o acesso a um conjunto de sites, basta adicionar as linhas, na seção *Access control*:

```
acl CURSO src 10.10.XX.0/255.255.255.0
acl sitesnegados dstdomain "/etc/sites-negados"
http_access deny sitesnegados
http_access allow CURSO
http_access deny all
```

e colocar no arquivo `/etc/sites-negados` o nome dos sites que terão os acessos negados. As 2 primeiras linhas criam listas de controle de acesso chamada `CURSO` e `sites-negados` do tipo `src` e `dstdomain`, a primeira contendo como regra, toda a rede interna e a segunda aponta para um arquivo que contém uma lista negra de sites.

Essas regras tornam-se válidas pelas linhas `http_access` seguintes, que neste caso é para negar o acesso a *acl* `sitesnegados` e liberado para `CURSO`. É bom lembrar que como estamos no modo *ordinary proxy*, o usuário pode burlar as *acls* bastando para isso, não usar o servidor de proxy, uma vez que as *ACL's* só tem efeito para os pedidos feito via *squid*.

Observação 12.3.1 *A ordem na qual são colocadas as linhas dos* `http_access` *devem ser consideradas.*

12.4 Configurando o Cliente

Deve-se configurar os navegadores (Opera, Mozilla, Internet Explorer) dos clientes para utilizar o proxy para os protocolos FTP, HTTP e HTTPS utilizando-se da porta padrão, que no caso é a porta *3128*. Além disso, configure o navegador para não utilizar o proxy para endereços do domínio local (`cursoXX.nti.ufpb.br`).

12.5 Exercícios

Exercício 12.5.1 *Tente criar uma ACL que libere o Windows (10.10.XX.10) para navegar em qualquer página.*

[1] Pode-se modificar o valor para 8080, facilitando aos usuários que desejam utilizar o *squid*, uma vez que a porta padrão para um servidor de *www* é 80.

CAPÍTULO 13

Servidor de FTP - ProFTP

O FTP *File Transfer Protocol* significa Protocolo de Transferência de Arquivos. O protocolo FTP permite a transferências de arquivos binários e arquivos texto com alta eficiência através de uma rede. É muito interessante para empresas que desejam compartilhar arquivos com os usuários na rede, ou servidores *web* que desejam disponibilizar áreas para usuários fazerem transferência de arquivos.

Neste capítulo será explicado a configuração do servidor FTP, que no caso, será o *Proftpd* (*Professional File Transfer Protocol Daemon*), que é o programa que implementa o servidor padrão e realiza os serviços de FTP no Conectiva Linux. O *Proftpd* possui uma configuração muito simples, parecida com a configuração do servidor *web Apache*, ou seja, em um único arquivo, com diretivas.

Além disso, implementações de segurança podem ser feitas em diretórios, utilizando para isso o arquivo `.ftpaccess`, permitindo restrições a diretórios que serão acessados por usuários.

13.1 Pré-requisitos

Para implementar esta solução você não precisará de nenhum pré-requisito especial, basta que sua máquina possua uma placa de rede.

13.2 Instalação

Para instalação do servidor FTP instale os seguintes pacotes: `proftpd` e `anonftp`

Para instalá-los, basta usar o seguinte comando:

```
[root@curso root]# apt-get install anonftp proftpd ftp
```

13.3 Inicializando

Para ativar o serviço utilizar o comando:

```
[root@curso root]# service proftpd start
```

Para testar o acesso ao servidor FTP com a conta *teste*, usarmos o comando `ftp` a partir do servidor, como a partir do Windows, como mostrado abaixo:

```
[root@curso root]# ftp 10.10.XX.1

Connected to 10.10.XX0.1.
220 ProFTPD 1.2.9 Server ready.
```

```
Name (10.10.XX.1:root): teste
331 Password required for teste.
Password:
230 User teste logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

13.4 Dando acesso a usuário anônimo

Para dar acesso ao servidor FTP anônimo, deveremos do arquivo `/etc/proftpd.conf`, retirar os comentários na configuração, deixando na seguinte forma:

```
# Default configuration for anonymous ftp, without an incoming directory.
#
<Anonymous /srv/ftp>
  User                    ftp
  Group                   ftp
  DirFakeUser             on
  DirFakeGroup            on
  RequireValidShell       off
  UserAlias               anonymous ftp
  MaxClients              10 "Sorry, max %m users -- try again later"
  MaxClientsPerHost       2 "Too many simultaneous connections"
  DisplayLogin            welcome.msg
  DisplayFirstChdir       .message
  AccessGrantMsg          "Anonymous access granted for %u."
  <Limit WRITE>
    DenyAll
  </Limit>
</Anonymous>
```

Para testar o acesso ao FTP como um anônimo, use o mesmo procedimento para o usuário *teste* utilizado anteriormente, com a diferença de fornecer o usuário *anonymous* e uma *string* qualquer (um endereço de e-mail, por exemplo), como senha.

Note que a senha não é mostrada. O usuário *anonymous* não precisa (nem deve) ser cadastrado em seu GNU/Linux, já que ele é um usuário especial para o servidor FTP. Quando é feita uma tentativa de acesso com o usuário *anonymous*, o servidor automaticamente trata o acesso como anônimo, aceitando o endereço de correio eletrônico como senha. Se algo não der certo, verifique se o pacote para acesso anônimo está instalado e reveja as configurações.

Você pode utilizar um arquivo `.message` para dar breves explicações sobre os propósitos dos diretórios que estão sendo acessados. Além disso, você poderia ajudar o usuário a encontrar o que ele está procurando. Para testar, crie um arquivo `.message` no `/srv/ftp/pub`, como um arquivo texto e coloque-o no diretório onde ele deve ser mostrado, com mensagens úteis sobre o acesso ao servidor FTP. Reinicialize o *Proftpd* e acesse o servidor novamente.

O servidor FTP pode ser acessado com qualquer navegador, bastando para tanto, colocar como endereço a ser acessado, como por exemplo: `ftp://10.10.XX.1`.

CAPÍTULO 14

Servidor de E-mail - Posfix

O correio eletrônico (*e-mail*) é um dos serviços mais utilizados na Internet, e cada vez mais pessoas e empresas utilizam-no para trocar informações de maneira rápida e eficiente.

Nesta seção será visto como funciona e como implementar um serviço de correio eletrônico.

14.1 Funcionamento do Correio Eletrônico

Antes de implementar um serviço de correio eletrônico é importante que o administrador entenda como funciona a troca de mensagens, seja na Internet, seja em uma rede local. Para uma simples troca de mensagens entre dois usuários, pode ser necessária a utilização de vários protocolos e de várias aplicações. Será visto a seguir como isso acontece.

Um usuário que queira enviar uma mensagem para outro utilizará um aplicativo cliente de *e-mail*, também conhecido como Agente de Mensagens do Usuário ou *MUA* (*Mail User Agent*). Ao terminar de redigir a sua mensagem o *MUA* enviará a mensagem a um Agente Transportador de Mensagens *MTA* (*Mail Transport Agent*) que se encarregará então de entregar a mensagem ao *MTA* do destinatário, caso ele se encontre em outra máquina ou simplesmente colocar a mensagem na caixa postal do destinatário, caso ele se encontre no mesmo servidor.

A transferência da mensagem entre o *MUA* e o *MTA* se efetua utilizando um protocolo chamado *SMTP* ou Protocolo Simples de Transferência de Mensagens (*Simple Mail Transfer Protocol*). O protocolo *SMTP* será utilizado também entre o *MTA* do remetente e o *MTA* do destinatário.

Veja a figura 14.1 que resume todo esse processo.

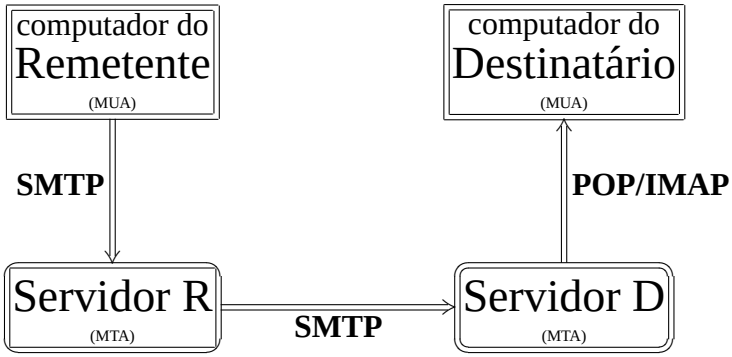


Figura 14.1: Transferência de Mensagens

O servidor de e-mail do destinatário, ao receber uma mensagem para um dos seus usuários, simplesmente a coloca na caixa postal deste usuário. A transferência de mensagens recebidas entre o servidor e o cliente de e-mail requer a utilização de outros programas e protocolos. Usualmente é utilizado o protocolo *POP* ou Protocolo de Agência de Correio (*Post Office Protocol*), que recebe

este nome por agir como uma agência de correios mesmo, que guarda as mensagens dos usuários em caixas postais e aguarda que estes venham buscar suas mensagens.

Outro protocolo que pode ser utilizado para este mesmo fim é o *IMAP* Protocolo para Acesso de Mensagens via Internet (*Internet Message Access Protocol*), que implementa, além das funcionalidades fornecidas pelo *POP*, muitos outros recursos.

Os protocolos *POP* e *IMAP* são protocolos para recebimentos de mensagens, ao contrário do protocolo *SMTP*, que serve para enviar mensagens, logo, possuem funcionalidades diferenciadas, como por exemplo, autenticação do usuário.

Para a utilização dos protocolos *POP* e *IMAP* é necessária a instalação do servidor apropriado, que vai ser o responsável por atender as solicitações do cliente de e-mail por novas mensagens. O recebimento de mensagens pelo cliente se dá através da solicitação do *MUA* do usuário ao seu servidor de e-mail, que após a autenticação do usuário vai informar se existem mensagens em sua caixa postal e quantas são. A seguir o MUA solicita a transferência das mensagens para a máquina local, finalizando assim o processo de troca de mensagens entre dois usuários.

14.2 O Postfix

O *Postfix* é o *MTA* padrão do Conectiva Linux e será o utilizado, pois apresenta bastante robustez, desempenho e maior facilidade na manutenção e configuração. Além disso, Postfix é capaz de emular várias funções do *Sendmail*[1], evitando assim modificações nas aplicações que o utilizam.

Outra característica importante do Postfix é a sua construção modular, facilitando a manuteção do código e permitindo a implementação de novas funcionalidades mais facilmente.

14.2.1 Pré-requisitos

Para uma implementação bem-sucedida do Postfix é necessário:

- uma interface de rede instalada e configurada (ver capítulo 8 página 36);
- um servidor DNS instalado e configurado (ver capítulo 10 página 43).

14.2.2 Instalação

Para instalar o Postfix instale o pacote utilizando o `apt-get`, da seguinte maneira:

```
[root@curso root]# apt-get install postfix
```

14.2.3 Configuração

Os valores padrões da configuração do Postfix exigem pouca modificação para se ter um servidor de e-mail funcional.

14.2.4 Testando a Configuração

Para testar a configuração é necessário certificar-se de que as mensagens estão chegando corretamente ao seu servidor e que as mensagens com destino fora do seu servidor estão sendo enviadas corretamente.

O administrador poderá iniciar os seus testes experimentando enviar uma mensagem a partir do próprio servidor para um outro usuário, também localizado no servidor. Ele pode utilizar o comando *mail* através de um terminal, digitando-se na linha de comando:

```
[root@curso root]# echo teste | mail usuario@minhaorganizacao
```

[1] Outro servidor de e-mail bastante utilizado

Para utilizar este comando, verifique se o pacote `mailx` está instalado em seu sistema.

Verifique se o usuário recebeu a mensagem e se o campo *From:* é preenchido corretamente, também verifique o arquivo de *logs*, em `/var/log/maillog`. Se a mensagem não foi recebida, certifique-se de que o serviço está rodando. Se o arquivo de registro acusar *mail from xx.xx loops back to mysel* verifique se o domínio local está configurado corretamente nas opções do Postfix (lembre-se também de que o servidor DNS deve estar corretamente configurado).

Pode-se então tentar enviar uma mensagem para um usuário localizado em outro servidor. Novamente, confira o arquivo de registro para verificar se houve algum problema no envio.

Você pode também utilizar, no modo terminal, o `pine`, que é um gerenciador de e-mail, de fácil utilização e em modo texto.

Após isto, seu servidor estará pronto para funcionar. Mas ainda nos resta um pequeno comentário sobre o arquivo `/etc/aliases`, que serve para que se defina apelidos para os endereços de e-mail da sua rede.

Exemplo 14.2.1 *Para que o usuário* ze.lezin *possa receber e-mails, também como* lezin*, bastaria adicionar no* `/etc/aliases` *a linha* `lezin: ze.lezin`*, e executado o comando* `newaliases`*. Depois disto os e-mails mandados para* lezim@domínio.br *seriam na verdade mandados para* ze.lezim@dominio.br.

Exemplo 14.2.2 *Se houvesse uma linha* `chefes: mickey pateta donald pluto` *no arquivo* `/etc/aliases` *então um e-mail direcionado a* `chefes@cursoXX.nti.ufpb.br` *seria retransmitido para os usuários* mickey*,* pateta*,* donald *e* pluto*.*

Exemplo 14.2.3 *Para pequenas listas, basta colocar a linha:*

```
todos: include /home/lista_com_os_nomes_dos_usuarios
```

e um e-mail mandado para todos@dominio.br *seria encaminhado para todos da lista.*

14.3 IMAP e POP3

Os protocolos IMAP e POP3 são responsáveis pelo transporte de mensagens recebidas do servidor de e-mail para o cliente de e-mail do usuário. O protocolo POP3 é mais antigo e mais simples e portanto mais popular e praticamente todos os programas clientes de e-mail o suportam. O protocolo IMAP é mais novo e possui mais funções que o POP3, no entanto nem todos os programas clientes de e-mail o suportam e que não será coberto neste texto.

O suporte a estes protocolos não é feito pelo Postfix, mas sim por outros servidores. Para implementar um servidor POP/IMAP é necessário somente que um servidor de e-mail esteja instalado e configurado.

14.3.1 Instalação do IMAP

Para instalar o IMAP instale-o via `apt-get`:

```
[root@curso root]# apt-get install cyrus-imapd
```

14.3.2 Configuração do IMAP

Para colocar os protocolos POP e IMAP em funcionamento no servidor, siga os seguintes passos:

- Inicie os serviços `cyrus-master` e `saslauthd`, utilizando o comando `service`.
- Modifique a senha do administrador IMAP (*cyrus*), com o comando `passwd`.
- Verifique se o arquivo `/etc/imapd.conf` contém no mínimo, estas linhas de configuração:

```
configdirectory: /var/lib/imap
partition-default: /var/spool/imap
admins: cyrus
sasl_pwcheck_method: saslauthd
```

Existem muitas outras opções disponíveis; verifique o que pode ser incluído no arquivo na página de manual (`imapd.conf`).

Neste ponto, o servidor IMAP já está configurado, mas sem contas para administrar. O próximo passo é incluir as caixas de correios de usuários. Primeiramente, entre no sistema como usuário *cyrus* (administrador IMAP), e em seguida, execute o comando `cyradm`, conforme abaixo:

```
[root@curso root]# su - cyrus

[/var/lib/imap]> cyradm -a plain curso.cursoXX.nti.ufpb.br
IMAP Password:
     curso.cursoXX.linux.nti.ufpb.br>
```

Você agora está na área de administração do IMAP. A seguir, estão descritos os comandos para administrar as *mailboxes*:

```
curso.cursoXX.linux.nti.ufpb.br> cm users.teste
curso.cursoXX.linux.nti.ufpb.br> lm
users.teste (\HasNoChildren)
curso.cursoXX.linux.nti.ufpb.br> dm users.teste
```

O comando `cm` cria a caixa de correio (no exemplo, do usuário teste). O uso de *users* é obrigatório. O `lm` lista as caixas, e `dm` remove (delete) a caixa de correio especificada.

Na página de documentação do IMAP você poderá consultar os outros comandos disponíveis.

Para configurar uma conta POP, não é necessária mais nenhuma configuração: ela usa as contas de usuários do sistema, ou as contas criadas pelo administrador IMAP.

14.3.3 Testando a Configuração do IMAP

Para testar a configuração do IMAP, será necessário configurar um cliente de e-mail localizado em alguma máquina de sua rede para buscar suas mensagens no servidor. Mande algumas mensagens para um usuário e depois tente recuperar as mensagens deste usuário através do cliente de e-mail. Caso não seja possível recuperar essas mensagens verifique os arquivos de registro `/var/log/messages` e `/var/log/maillog` em busca de mensagens de erro.

O teste de funcionamento da configuração do POP pode ser feito da mesma forma: configure um cliente de e-mail para que ele busque as mensagens de uma conta POP; envie uma mensagem para esta conta e depois tente recuperá-la. Caso não funcione, verifique se a senha foi digitada corretamente e se a configuração do cliente de e-mail está correta.

Outro teste da configuração do POP também é simples, e pode ser feito no próprio servidor:

```
[root@curso root]# telnet localhost 110

Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
+OK curso.cursoXX.linux.nti.ufpb.br Cyrus POP3 v2.2.4 server ready
<1271837990.1099938796@curso.cursoXX.nti.ufpb.br>
user cyrus senha_de_cyrus
+OK Name is a valid mailbox
user teste senha_do_teste
+OK Name is a valid mailbox
quit
```

As linhas iniciadas com *user* foram digitadas para verificar se as caixas de correio daquele usuário estão corretas. Se algo mais aparecer, algo está errado, e você deverá verificar os arquivos de *log* para obter os detalhes.

Se desejar, pode ainda fazer um último teste com o IMAP, também no próprio servidor:

```
[root@curso root]# su - cyrus

[/var/lib/imap]> cyradm -a plain curso.cursoXX.linux.nti.ufpb.br
IMAP Password:
     cursoXX.linux.nti.ufpb.br>
curso.cursoXX.linux.nti.ufpb.br> imtest -m login -p imap curso
S: * OK curso.cursoXX.linux.nti.ufpb.br Cyrus IMAP4 v2.2.4 server ready
C: C01 CAPABILITY
S: * CAPABILITY IMAP4 IMAP4rev1 ACL QUOTA LITERAL+ MAILBOX-REFERRALS
NAMESPACE UIDPLUS ID NO_ATOMIC_RENAME UNSELECT CHILDREN MULTIAPPEND
BINARY SORT THREAD=ORDEREDSUBJECT THREAD=REFERENCES ANNOTATEMORE IDLE
AUTH=DIGEST-MD5 AUTH=CRAM-MD5 SASL-IR LISTEXT LIST-SUBSCRIBED X-NETSCAPE
S: C01 OK Completed
Please enter your password:
C: L01 LOGIN cyrus {7}
S: + go ahead
C: <omitted>
S: L01 OK User logged in
Authenticated.
Security strength factor: 0
. logout
* BYE LOGOUT received
. OK Completed
Connection closed.
curso.cursoXX.linux.nti.ufpb.br> quit
curso [/var/lib/imap] > exit
```

Se algo diferente que isto, verifique os *logs* do *cyrus*.

14.3.4 Instalação e Configuração do servidor de POP3

Será usado como servidor de POP3, o `ipop3d` que faz parte do pacote *imap*. Este pacote contém também um servidor *imap*, que é outro protocolo para leitura/envio remoto de e-mails, que não será coberto neste texto.

Após a instalação, será habilitado o serviço pop-3, bastando para isso executar o comando `chkconfig ipop3 on`, que irá alterar o arquivo `/etc/xinetd.d/ipop3`, ou seja, troca a opção `disable = yes` por `no`. É necessário após isto reinicializar o *inetd*, para que as alterações tenham efeito, usando o seguinte comando:

```
[root@curso root]# service xinetd restart
```

Por fim, será testado o funcionamento do servidor de POP3 usando o comando a seguir:

```
[root@curso root]# telnet localhost 110
```

Se estiver configurado corretamente, deverá aparecer a seguinte resposta:

```
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
+OK POP3 localhost.localdomain v7.59 server ready.
```

CAPÍTULO 15

Servidor SAMBA

O SAMBA é um aplicativo que torna possível o compartilhamento de recursos com máquinas rodando Windows. O nome SAMBA é derivado do protocolo utilizado pelo Windows para compartilhar discos e impressoras: o protocolo *SMB* (*Server Message Block*).

Através da utilização do SAMBA é possível criar redes mistas, utilizando servidores Linux e clientes Windows. O SAMBA também permite que o GNU/Linux acesse discos exportados de máquinas Windows. Além de compartilhar recursos, o SAMBA é capaz de executar várias funções de um servidor Windows, como por exemplo autenticação de clientes, servidor *WINS*, e até mesmo agir como um *Controlador Primário de Domínio* (*PDC*).

15.1 Pré-requisitos

Para a implementação de uma solução utilizando o SAMBA:

- a rede deve estar corretamente instalada e configurada;
- as estações Windows devem utilizar o protocolo TCP/IP e que possam acessar o servidor GNU/Linux.

15.2 Instalação

Para instalar o SAMBA e seus aplicativos de configuração, deve-se utilizar os seguintes pacotes: `samba-common`, `samba-server`, `samba-clients` e `smbfs` (para usar como um cliente) e `samba-doc` (opcional).

Instale os pacotes anteriores digitando em um terminal os seguintes comandos:

```
[root@curso root]# apt-get install samba-common samba-server
[root@curso root]# apt-get install samba-doc samba-clients smbfs
```

15.3 Configuração

Toda a configuração relacionada com *nomes*, *grupo de trabalho*, *tipo de servidor*, *log*, *compartilhamento de arquivos* e *impressão* do SAMBA é colocada no arquivo de configuração `smb.conf` no diretório `/etc/samba`. Este arquivo é dividido em *seções* e *parâmetros*.

15.3.1 Seções

Uma seção no arquivo de configuração do SAMBA é definido por um nome entre "[ ]". As seções tem o objetivo de organizar os parâmetros pra que tenham efeito somente em algumas configurações de compartilhamento do servidor[1].

Linhas iniciadas por "#" ou ";" são tratadas como comentário. Quebras de linha pode ser especificadas com uma no final da linha.

Alguns nomes de seções foram reservados para configurações específicas do samba, eles são os seguintes:

[global] - define configurações que afetam o servidor samba como um todo, fazendo efeito em todos os compartilhamentos existentes na máquina. Por exemplo, o grupo de trabalho, nome do servidor, restrições de acesso por nome, etc (ver seção 15.4, na pagina 60).

[homes] - especifica opções de acesso aos diretórios dos usuários (`/home`). O diretório `home` é disponibilizado somente para seu dono, após se autenticar no sistema (ver seção 15.4, na pagina 64).

[printers] - define opções gerais para controle das impressoras do sistema. Este compartilhamento mapeia os nomes de todas as impressoras encontradas no `/etc/printcap`. Configurações especiais podem ser feitas separadamente (ver seção 15.6, na pagina 64).

E outras duas seções necessárias para que o SAMBA seja um PDC, são:

[netlogon] - define um compartilhamento no servidor SAMBA, necessário pelos clientes Windows (ver seção 15.7, na pagina 64).

[profiles] - define um perfil ambulante para os clientes Windows (ver seção 15.8, na pagina 65).

Qualquer outro nome de `[seção]` no arquivo `smb.conf` que não sejam as acima, são tratadas como um compartilhamento ou uma impressora.

15.3.2 Parâmetros

Um parâmetro é definido no formato `nome = valor`. Na configuração de booleanos, a seguinte sintaxe pode ser usada:

- **0** ou **1**
- **yes** ou **no**
- **true** ou **false**

15.4 A seção [global]

Os parâmetros especificados nesta seção tem efeito em todo o servidor SAMBA incluindo os compartilhamentos. Caso o parâmetro seja novamente especificado para o compartilhamento, o valor que valerá é o do compartilhamento.

A seguir serão detalhadas os principais parâmetros:

[1] Exceto os da seção `[global]` que não especificam compartilhamentos, mas suas diretivas podem ser válidas para todas os compartilhamentos do arquivo de configuração.

15.4.1 Nomes e grupos de trabalho

netbios name - especifica o nome NetBIOS primário do servidor samba. Caso não seja ajustado, ele usará o it hostname de sua máquina como valor padrão.
Ex.: `netbios name = ServidorXX`

workgroup - diz qual o nome do grupo de trabalho/domínio que a máquina samba pertencerá.
Ex.: `workgroup = CURSOXX`

netbios aliases - permite o uso de nomes alternativos ao servidor, separados por espaços.
Ex.: `netbios aliases = serv1 principal`

server string - identificação enviada do servidor samba para o ambiente de rede. A string padrão é Samba.
Ex.: `server string = Servidor SAMBA`

name resolve order - define a ordem de pesquisa para a resolução de nomes no samba. A ordem padrão é: it lmhosts, *host*, *wins* e *bcast*, que é a melhor para resolução rápida e que tente gerar menos tráfego broadcast possível.
Ex.: `name resolve order = host wins`

netbios name - especifica o nome NetBIOS primário do servidor samba. Caso não seja ajustado, ele usará o it hostname de sua máquina como valor padrão.
Ex.: `netbios name = ServidorXX`

15.4.2 Impressoras

printcap name - este parâmetro define a localização do arquivo `printcap` (que contém as definições da impressoras) ou se utilizará o gerenciador de impressão *CUPS*.
Ex.: `printcap name = cups`

load printers - este parâmetro controla se o SAMBA deve ou não apresentar todas as impressoras presente no arquivo `printcap` como compartilhadas. O valor padrão é `yes`, já o valor `no` significa que não queremos compartilhar todas as impressoras e nesse caso devemos especificar quais impressora queremos compartilhar individualmente.
Ex.: `load printers = yes`

printing - controla como será interpretada as informações da impressora no sistema.
Ex.: `printing = cups`

15.4.3 Caracteres e página de código

character set - seleciona o conjunto de caracteres dos arquivos exibidos pelo servidor SAMBA. Para os idiomas de língua latina, sempre utilize *iso8859-1*.
Ex.: `character set = iso8859-1`

client code page - seleciona a página de código do servidor SAMBA para tratar os caracteres. Para os idiomas de língua latina, sempre utilize *850*.
Ex.: `client code page = 850`

preserve case - seleciona se arquivos com nomes extensos criados serão criados com os caracteres em maiúsculas/minúsculas definidos pelo cliente (no) ou se será usado o valor de default case (caso seja especificado yes).
Ex.: `preserve case = yes`

short preserve case - seleciona se os arquivos com nomes curtos no formato 8.3, serão criados com os caracteres mixto enviados pelo cliente (no) ou se será usando o valor de default case (caso seja especificado yes).
Ex.: `short preserve case = yes`

default case - define se os arquivos criados terão seus nomes todos em minúsculas (lower) ou maiúsculas (upper).
Ex.: `default case = upper`

valid chars - define caracteres válidos nos nomes de arquivos. Este parâmetro DEVERÁ ser sempre especificado depois do `client code page`.
Ex.: `valid chars = á:Á é:É í:Í ó:Ó ú:Ú â:Â ê:Ê ô:Ô ã:Ã õ:Õ à:À ò:Ò`

15.4.4 Restrições de acesso/mapeamento de usuários

guest account - define a conta local de usuário que será mapeada quando um usuário se conectar sem senha (usuário guest).
Ex.: `guest account = visitante`

invalid users - define uma lista de usuários que não terão acesso aos recursos do servidor ou compartilhamento. É seguro restringir o acesso samba a usuários com grande poder no sistema (como o *root*).
Ex.: `invalid users = root`

valid users - semelhante a opção invalid users mas permite que somente os usuários especificados tenham acesso ao sistema.
Ex.: `valid users = jose`

username map - especifica um arquivo que faz o mapeamento entre nomes fornecidos por clientes e nomes de contas locais.
Ex.: `username map = /home/lista`

Exemplo 15.4.1 *Arquivo de mapeamento:*

```
chico = "chico da silva"
nobody = root adm
samba = @smb-users
!chico = "chico da silva"
!samba = @smb-users
nobody = *
```

15.4.5 Níveis de autenticação

Define o nível de segurança do servidor. Os seguintes valores são válidos para o parâmetro `security`:

share - usada principalmente quando apenas a senha é enviada por compartilhamento acessado para o servidor, caso muito típico em sistemas *Lan Manager* e *Windows for Workgroups*.
Ex.: `security = share`

user - este é o padrão. O usuário precisa ter uma conta de usuário no GNU/Linux para acessar seus compartilhamentos. A mesma conta de usuário/senha deverá ser usada no Windows para acessar seus recursos ou realizado um mapeamento de nomes de usuários.
Ex.: `security = user`

domain - neste nível, o acesso só será permitido quando a máquina for adicionada ao domínio com o `smbpasswd` e a conta do usuário será validada em um servidor PDC (controlador de domínio) e o acesso aos recursos das máquinas que fazem parte do domínio será feito a partir do PDC.
Ex.: `security = domain`

server - a validação do usuário é redirecionada para outro computador que pode ser um servidor Windows ou outro servidor SAMBA. Uma vez validado, os compartilhamentos do GNU/Linux ficam disponíveis, desde que o *login* usado na validação exista no GNU/Linux.
Ex.: `security = server`

15.4.6 Registros de acessos e serviços

log file - define a localização e nome do arquivo de log gerado pelo SAMBA.
Ex.: `log file = /var/log/samba/%m.log`

max log size - especifica o tamanho máximo em Mb do arquivo de log gerado pelo SAMBA. O valor padrão é 50 Mb (5MB).
Ex.: `max log size = 10`

debug level - aumenta o nível de depuração dos serviços do SAMBA.
Ex.: `debug level = 1`

15.4.7 WINS

O WINS (*Windows Internet Naming Service*) é um serviço de resolução de nomes que funciona de forma semelhante ao *DNS*, só que voltado para o *NetBIOS*. Quando uma máquina cliente *NetBIOS* entra na rede, o servidor WINS pega o nome e o IP e inclui em uma tabela para futura consulta pelos clientes da rede. Os principais parâmetros são:

win support - informa se o SAMBA deve agir como servidor WINS.
Ex.: `win support = yes`

wins server - informa ao SAMBA qual é o servidor WINS da rede.
Ex.: `wins server = x.y.z.w`

wins proxy - se tiver algum cliente mais velho na rede, que não possua o WINS implementado, pode fazer com que o SAMBA delegue sua consulta para o servidor WINS.
Ex.: `wins proxy = no`

wins proxy - permite que o SAMBA use consultas *DNS* para verificar nomes *CIFS*.
Ex.: `dns proxy = no`

15.4.8 Navegação no servidor e tipo de servidor

os leve - especifica o nível do sistema operacional. Este número é usado para as eleições NetBIOS para definir o navegador de grupo local e controlador de domínio.
Ex.: `os level = 32`

announce as - selecione o nome que o samba (*nmbd*) se anunciará na lista de pesquisa de rede.
Ex.: `announce as = NT Server (ou NT)`

domain master - diz se o servidor tentará se tornar o navegador principal de domínio.
Ex.: `domain master = yes`

local master - diz se o servidor participará ou não das eleições para navegador local do grupo de trabalho (workgroup).
Ex.: `local master = yes`

preferred master - diz se o servidor samba terá ou não vantagens de ganhar uma eleição local.
Ex.: `preferred master = yes`

15.5 A seção [homes]

Esta seção tem a função especial de disponibilizar o diretório `/home` do usuário. Quando o usuário envia seu nome de *login* como compartilhamento é feita uma busca no arquivo `smb.conf` procurando por um nome de compartilhamento que confira. Caso nenhum seja encontrado, é feita uma busca por um nome de usuário correspondente no arquivo `/etc/passwd`, se um nome conferir e a senha enviada também, o diretório de usuário é disponibilizado como um compartilhamento com o mesmo nome do usuário local. Quando o caminho do compartilhamento não for especificado, o SAMBA utilizará o diretório `/home` do usuário definido em `/etc/passwd`.

Exemplo 15.5.1 *Seção homes padrão*

```
[homes]
        comment = Diretório Pessoal
        browseable = no
        writable = yes
```

15.6 A seção [printers]

Esta seção tem a função de disponibilizar as impressoras existentes no sistema (*lp*, *lp1*, *lp2*, etc) existentes no `/etc/printcap` como compartilhamento de sistemas Windows.

Exemplo 15.6.1 *Seção homes padrão*

```
[printers]
        comment = All Printers
        path = /var/spool/samba
        browseable = no
        guest ok = no
        writable = no
        printable = yes
        printer admin = root

[hp-printer]
        path = /tmp
        printer name = HP DeskJet 690C
        printable = yes
        print command = lpr -r -h -P %p %s
        valid users = winuser winuser2
        create mode = 0700
```

15.7 A seção [netlogon]

É um compartilhamento procurado pelo cliente Windows no *login* de um usuário, para utilizá-lo deve-se ter no `smb.conf` os seguintes parâmetros na seção global e o compartilhamento abaixo:

```
[global]
        domain logons = yes
        logon script = inicio.bat

[netlogon]
        writeable = yes
        path = /home/netlogon
        force create mode = 0777
        create mask = 0777
        comment = Diretório para o logon
        public = yes
```

O parâmetro `ogon script = inicio.bat` informa qual comando localizado no compartilhamento `netlogon` será executado pelo cliente Windows após o *logon*.

Normalmente é utilizado um arquivo de lote (DOS), mas existe outras possibilidade de *scripts* como por exemplo (recomendo) o *KiXtart* encontrado em `www.kixtart.org` que tem muito mais opçoes e portanto um poder maior sobre os clientes Windows.

Usaremos para o nosso servidor o seguinte arquivo:

```
rem Logon script padrão para a rede: inicio.bat
cls
@echo off
if %OS%.==Windows_NT. goto WinNT
:Win95
  net use U: \\servidorxx\HOMES
  net use N: \\servidorxx\netlogon
  net time \\servidorxx /set /yes
  goto end
:WinNT
  net use P: \\servidorxx\arquivos /persistent:no
  net use N: \\servidorxx\netlogon /persistent:no
:end
```

15.8 A seção [profiles]

É um compartilhamento utilizado pelo cliente Windows para manter o mesmo perfil em qualquer máquina que ele se autentique na rede, para utilizá-lo deve-se ter no `smb.conf` os seguintes parâmetros na seção global e o compartilhamento abaixo:

```
[global]
        security = user
        encrypt passwords = yes
        domain logons = yes
        logon drive = U:
        logon path = \\%L\profiles\%U
        logon home  = \\%N\%u
        preserve case = yes
        short preserve case = yes
        case sensitive = no

[profiles]
        comment = Profile de %u
        path = /home/profiles
```

```
        read only = no
        create mask = 0777
        force create mode = 0770
```

15.9 Seções de compartilhamento.

As opções mais usadas nas seções de compartilhamento são:

comment - fornece uma do compartilhamento, que será mostrada na janela de procura de rede.
Ex.: `comment = Programas e Arquivos`

path - especifica o caminho do diretório que se quer compartilhar.
Ex.: `path = /home/arquivos`

public - permitem aos usuários utilizarem do compartilhamento sem fornecer uma senha.
Ex.: `public = no`

browsable - define se o compartilhamento será ou não exibido na janela de procura de rede.
Ex.: `browsable = yes`

read only - especifica se o compartilhamento é somente para leitura (`yess`) ou não (`no`) para todos os usuários. O parâmetro `writable` é um antônimo equivalente a este parâmetro, só que utiliza as opções invertidas.
Ex.: `read only = no`
Ex.: `writable = yes`

printable - especifica se o compartilhamento é uma impressora.
Ex.: `printable = no`

write list - lista de usuários separados por espaço ou vírgula que poderão ler e gravar no compartilhamento. Caso o nome for iniciado por "@´´, o nome especificado será tratado como um grupo do GNU/Linux (`/etc/group`) e todos os usuários daquele grupo terão acesso de gravação.
Ex.: `write list = @especial`

valid users - especifica quais os usuários podem usar esse compartilhamento
Ex.: `valid users = @users`

create mask - especifica a permissão que terão os arquivos criados no compartilhamento. Os valores são semelhante aqueles usados no comando `chmod`.
Ex.: `create mask = 0770`

diretory mode - especifica a permissão que será dada aos diretórios criados no compartilhamento.
Ex.: `directory mode = 0770`

Exemplo 15.9.1 *Compartilhar os arquivos do diretório* `/home/arquivos` *como* arquivos*:*

```
[arquivos]
        comment = Programas e Arquivos
        path = /home/arquivos
        public = no
        writable = yes
        browseable = yes
        printable = no
        write list = @especial
        valid users = @users
```

```
        create mask = 0770
        force directory mode = 0770
        force create mode = 0770
```

15.10 Utilizando o SAMBA com PDC

Para utilizar o GNU/Linux como servidor de arquivos e PDC de uma rede com clientes Microsoft Windows 95/98/2000/XP (Professional e Server) **exceto XP Home Edition**.

A configuração do `smb.conf` deve seguir o modelo abaixo:

```
[global]
        workgroup = CURSOXX
        netbios name = SERVIDORXX
        server string = Servidor de Arquivos (PDC)
        bind interfaces only = yes
        interfaces = loopback, eth1
        hosts equiv = /etc/hosts
        username map = /etc/samba/smbusers
        log level = 1
        log file = /var/log/samba/%m.log
        max log size = 10
        name resolve order = host wins bcast
        hostname lookups = Yes
        socket options = TCP_NODELAY SO_SNDBUF=8192 SO_RCVBUF=8192
        printcap name = /etc/printcap
        add machine script = /usr/sbin/adduser -n -r -g win
                                    -d /dev/null -s /bin/false %u
        logon script = inicio.bat
        logon path = \\%L\profiles\%u
        logon drive = U:
        logon home = \\%L\%u
        domain logons = Yes
        os level = 65
        domain master = Yes
        dns proxy = No
        wins support = Yes
        remote announce = 10.10.10.255
        admin users = root
        hosts allow = 10.10.10., 127.
        time server = yes
        wins support = yes
        wins proxy = no
        dns proxy = no
        max wins ttl = 518400

[homes]
        comment = Home Directories
        read only = No
        browseable = No

[netlogon]
        comment = Network Logon Service
        path = /home/netlogon
        read only = No
        create mask = 0777
        force create mode = 0777
```

```
        guest ok = Yes

[profiles]
        comment = Profile de %u
        path = /home/profiles
        read only = no
        create mask = 0777
        force create mode = 0770

[arquivos]
        comment = Programas e Arquivos
        path = /home/arquivos
        write list = @especial
        read only = No
        create mask = 0770
        force create mode = 0770
        force directory mode = 0770
```

Use o `testparm`, para testar sua configuração (essa ferramenta é importante para descobrir possíveis falhas na sua configuração.

```
[root@curso root]# testparm
```

15.11 Criando um usuário no SAMBA

Um usuário do SAMBA DEVE SER necessariamente um usuário local do GNU/Linux, logo para a criação de usuários no SAMBA, deve-se primeiro criá-los no GNU/Linux, com o comando:

```
[root@curso root]# useradd -g especial -s /bin/false joao
```

Esse comando cria um usuário sem *shell* e isto mantém o sistema mais seguro e não interfere no funcionamento do SAMBA, pois somente é necessário para fazer o mapeamento de UID e GID dos usuários com as permissões do sistema GNU/Linux.

Para criar essa conta no SAMBA o comando é:

```
[root@curso root]# smbpasswd -a joao
```

Para mais detalhes do comando `smbpasswd` olhar no manual.

15.12 Clientes com Windows 2000/XP

Para autilização de clientes com Windows 2000/XP, deve-se seguir os seguintes passos:

1. Crie um grupo para os computadores com Windows no GNU/Linux.

   ```
   [root@curso root]# groupadd win
   ```

2. É necessário criar uma conta no GNU/Linux para cada computador com Windows (é preciso ter o $ no final do nome), sem *shell* e sem diretório, como por exemplo, criar com o comando:

   ```
   [root@curso root]# useradd -g win -s /bin/false
   -d /dev/null nome-win2k$
   ```

3. Faça *LOCK* na senha da conta de máquina Windows2K.

   ```
   [root@curso root]# passwd -l nome-win2k$
   ```

Mesmo tendo criado o seu usuário de máquina no grupo *win*, adicione-o manualmente no `/etc/group` como `win:x:537:nome-win2k$`

4. Agora, crie a conta do computador Windows no samba, com o comando:

   ```
   [root@curso root]# smbpasswd -a -m nome-win2k$
   ```

   e ative essa conta com o comando:

   ```
   [root@curso root]# smbpasswd -e nome-win2k$
   ```

Para que o Windows faça parte do domínio do servidor GNU/Linux, deve-se configurar o Windows para se juntar (*join*) ao domínio, com as seguintes observações:

- Instale o *Service Pack* mais recente, no computador com Windows, para que a mesma consiga realizar o *join* no domínio samba.
- Não esqueça que cada usuário do sistema precisa ter uma pasta correspondente no `/home`.
- Não esqueça as permissões da pasta `home` do usuário.
- Às vezes, nenhuma os computadores com Windows, não conseguem fazer *login* no domínio, mas adicionar as entradas das máquinas presentes na rede no `/etc/hosts` ajuda.

  ```
  127.0.0.1 localhost.localdomain localhost
  10.10.XX.1 server01 server01
  10.10.XX.20 desktop20 desktop20
  10.10.XX.30 desktop30 desktop30
  ```

Com estas configurações, você está pronto para usar seu PDC Linux.

15.13 Utilizando um cliente SAMBA Linux

Anteriormente viu-se como configurar um servidor SAMBA, de maneira que outros computadores, sejam eles Linux ou Windows, pudessem ter acesso ao arquivos e impressoras nele contido. No entanto também é possível configurá-lo de modo que ele seja um cliente para um servidor Windows NT ou 9x. Para essas situações, é necessário utilizar as ferramentas do cliente SAMBA, mas especificamente o `smbclient` e `smbmount`.

15.13.1 Utilizando o smbclient

O programa `smbclient` é um cliente de acesso à servidores Windows/SAMBA com uma interface de cliente FTP. Um exemplo do uso do `smbclient` é mostrado a seguir, onde estamos acessando um computador chamado *curso*, que tem o seu disco rígido compartilhado com o nome de *docs*.

```
[root@curso root]# smbclient //servidorxx/arquivos

Added interface ip=10.10.XX.1 bcast=10.10.XX.255 nmask=255.255.255.0
Password:
Domain=[CURSOXX] OS=[Windows NT 4.0] Server=[NT LAN MANAGER]

smb : \> ls
    .              D        0  Sat Dec 10 10:22:22  2004
    ..             D        0  Sat Dec 10 10:22:22  2004
    TEXTO.DOC      A   147024  Sat Dec 10 10:22:22  2004
    ARQUIVO.DOC    A   147024  Sat Dec 10 10:22:22  2004
```

No exemplo acima conectou-se ao compartilhamento *arquivos* e listamos o seu conteúdo. Temos um conjunto vasto de comandos tais como `ls`, `put`, `get`, etc. Podemos obter um `help` completo de todos os comandos digitando *h*.

15.13.2 Utilizando o smbmount

O `smbmount` permite você montar um compartilhamento e utilizá-lo como se fosse parte do sistema de arquivo do Linux. Os compartilhamentos são montados utilizando o comando `smbmount` e desmontados utilizando o comando `umount`. Para montar, por exemplo, o compartilhamento chamado *docs* que esta localizado no computador chamado *curso*, digite:

```
[root@curso root]# smbmount //servidorxx/arquivos /mnt/montado
```

A sintaxe do comando `smbmount` é similar a do comando `mount`, onde o primeiro argumento é o nome do computador e compartilhamento que desejamos montar e o segundo argumento é o ponto de montagem. Já para desmontar, basta usarmos o comando `umount` seguido do caminho do ponto de montagem.

Para desmontar o sistema de arquivo montado anteriormente, usaríamos:

```
[root@curso root]# umount /mnt/montado
```

15.14 Variáveis usadas pelo SAMBA

O SAMBA inclui uma lista completa de variáveis para determinar as características do servidor SAMBA e dos clientes que se conectam a ele. Cada uma dessas variáveis começa com o símbolo de porcentagem (%), seguidas de uma única letra maiúscula ou minúscula e pode ser usada apenas do lado direito da opção de configuração.

15.14.1 Variáveis do Cliente

Variável	Definição
%a	Arquitetura do cliente (SAMBA, WfWg, WinNT, Win95, ou UNKNOWN)
%I	Endereço IP do cliente
%m	Nome NetBIOS do cliente
%M	Nome DNS do cliente

15.14.2 Variáveis do Usuário

Variável	Definição
%g	Grupo primário de %u
%G	Grupo primário de %U
%H	Diretório base de %u
%u	Nome do usuário no Linux, pode não ser o mesmo nome usado na conexão.
%U	Nome do usuário usado na conexão.

15.14.3 Variáveis de Compartilhamento

Variável	Definição
%p	Caminho para o diretório raiz compartilhado, caso seja diferente de %P
%P	Atual diretório raiz compartilhado

%S	Nome do compartilhamento

15.14.4 Variáveis do Servidor

Variável	Definição
%d	ID do processo do smbd que está gerenciando a conexão.
%h	Nome DNS do servidor SAMBA
%L	Nome NetBIOS do servidor SAMBA
%N	Diretório base do servidor, para o `automount map`
%v	Versão do SAMBA

15.14.5 Variáveis Gerais

Variável	Definição
%R	O nível do protocolo SMB que foi negociado
%T	Data e hora atual

15.15 Exercícios

Exercício 15.15.1 *Configurar o seu servidor como um servidor de logon, com as seguintes características:*

a) *Que o servidor compartilhe 4 pastas, sendo uma do* **usuário**, *uma do* **netlogon**, *uma pasta* **pública** *e outra* **documentos**, *com as seguintes características*[2]*:*

Pasta	*Visível na rede*	*Diretório*	*Quem pode escrever*
Usuário	*não*	*/home*	*apenas o usuário*
Netlogon	*não*	*/home/netlogon*	*apenas o super*
Pública	*sim*	*/home/publico*	*apenas o grupo chefe*
Documentos	*sim*	*/home/doc*	*todos*

b) *Criar o script que monte os compartilhamentos usuário, pública e documentos, nas respectivas letras,* **U:**, **P:** *e* **T:**, *e que atualize a hora.*

[2]Ver outros exemplos contidos no *smb.conf*

CAPÍTULO 16

Servidor DHCP

Todo computador conectado uma redes que usa TCP/IP precisa, para se comunicar, de uma identificação numérica. Esta identificação é conhecida como endereço IP. O endereço IP pode ser atribuído de forma estática ou dinâmica.

Endereços IP atribuídos estaticamente possuem algumas desvantagens. Sempre que um equipamento for movido de uma rede para outra o endereço IP tem que ser alterado manualmente, o que pode envolver uma consulta ao administrador de redes. Adicionalmente, cada rede IP possui uma rota distinto, que também precisa ser indicado na configuração do equipamento. Em pequenas redes isso é possível de ser feito (por uma tabela escrita), mas em grandes redes isso se torna uma tarefa trabalhosa, e bastante sujeita a falhas.

Endereços atribuídos dinamicamente oferecem uma flexibilidade maior, pois libertam o usuário de conhecer detalhes sobre a configuração de sua máquina, permitindo-lhes uma maior mobilidade dentro da rede. Tudo o que é necessário é desconectar o equipamento de um ponto e ligá-lo em outro e tudo continuará funcionando normalmente. Usuários de computadores portáteis se beneficiam ainda mais, pois ficam livres de constantemente terem que identificar endereços IP livres nas redes em que irão trabalhar.

A atribuição dinâmica de endereços IP é feita através do protocolo DHCP ou Protocolo de Configuração de Hospedeiro Dinâmico (*Dynamic Host Configuration Protocol*). Seu uso e configuração, tanto do lado do cliente como do servidor, é extremamente simples.

O procedimento do servidor de DHCP é verificar qual o endereço IP disponível, numa tabela cadastrada previamente dentro do mesmo, e informar ao solicitante esse endereço e o tornar indisponível para futuras solicitações. Desta maneira, a administração dos endereços IPs é feita automaticamente e não existem problemas de conflito. Quando a máquina solicitante sai da rede o servidor DHCP torna seu endereço IP disponível novamente.

Este protocolo baseia-se num conjunto de mensagens emitidas em modo de difusão (*broadcast*). Assim sendo torna-se vital garantir que os roteadores encaminhem estas mensagens ou que exista um servidor de DHCP por cada subdomínio.

Em sistemas GNU/Linux é preciso instalar o software DHCP, que é desenvolvido e mantido pelo *Internet Software Consortium* (`www.isc.org`).

16.1 Pré-requisitos

Para a instalação do serviço DHCP é necessário que:

- a rede deve estar corretamente instalada e configurada;
- os computadores dos clientes utilizem o protocolo TCP/IP e que possam acessar o servidor GNU/Linux.

16.2 Instalação

Para instalar DHCP e seus aplicativos de configuração, deve-se instalar os seguintes pacotes: `dhcp` para o servidor e `dhcpcd` para usar como um cliente (opcional).

Instale os pacotes digitando em um terminal os seguintes comandos:

```
[root@curso root]# apt-get install dhcp
[root@curso root]# apt-get install dhcpcd dhcp-doc
```

16.3 Configuração

Como todo serviço do GNU/Linux, a configuração do DHCP é através do arquivo `dhcpd.conf` que se encontra no diretório `/etc/`, vejamos o exemplo abaixo:

Exemplo 16.3.1

```
                                # Nome do domínio e do servidor
server-identifier               roteador.curxoxx.nti.ufpb.br;
option domain-name              "curxoxx.nti.ufpb.br";
                                # Servidor de nomes DNS
option domain-name-servers      10.10.XX.1,
                                150.165.251.200;
ddns-update-style ad-hoc;
authoritative;
                                # mascara da rede
option subnet-mask              255.255.255.0;
                                # libertará o IP por 600s
default-lease-time              600;
                                # o tempo máximo permitido é de 7200s
max-lease-time                  7200;
                                # Broadcast da rede
option broadcast-address        10.10.XX.255;
                                # IP do roteador, que no nosso caso é:
option routers                  10.10.XX.1;
                                # Usado pelos clientes Windows
option netbios-node-type        8;
option netbios-name-servers     10.10.XX.1;
option netbios-dd-server        10.10.XX.1;

# Definindo uma sub-rede em 10.10.XX.0
# variando de 10.10.XX.10 até 10.10.XX.50
subnet 10.10.XX.0 netmask 255.255.255.0 {
range 10.10.XX.10 10.10.XX.50;
}
# Definindo IP para determinados clientes (pelo MAC)
host sergio.curxoxx.nti.ufpb.br{
hardware ethernet             00:C0:DF:E8:9E:81;
fixed-address                 10.10.XX.5 ;
}
host teste.curxoxx.nti.ufpb.br{
hardware ethernet             00:30:9D:E8:2E:22;
fixed-address                 10.10.XX.51 ;
}
```

Falta criar um arquivo, onde serão colocadas pelo servidor DHCP, todas as informações dos pedidos dos clientes, para tanto, criaremos com o seguinte comando:

```
[root@curso root]# echo > /var/state/dhcp/dhcpd.leases
```

bem como dizer ao DHCP qual placa de rede ele irá aceitar os pedidos, e isso é feito editando o arquivo `/etc/init.d/dhcpd`, acrescentando à linha `daemon /usr/sbin/dhcpd` a placa de rede a qual o DHCP irá atender, ou seja, no nosso caso a `eth1`.

Após digitado essa configuração e criado o arquivo, basta ativá-lo, como todo serviço do Linux, ou seja:

```
[root@curso root]# service dhcpd start
```

Mais detalhes (como sempre) devem ser obtidas dos manuais do `dhcpd` e `dhcpd.conf` e os *logs* no arquivo `/var/log/messages`.

16.4 Configurando o cliente Windows

16.4.1 Windows 9x

Abra o painel de controle e abra o ícone rede, adicione o protocolo TCP/IP, caso não exista e depois nas propriedades do protocolo ativar o *obter um número IP automaticamente*.

Feito isso, o cliente assim que ligado vai fazer o pedido na rede de um IP, e é nessa hora que as configurações feitas anteriormente serão repassadas ao cliente.

Para checar se o DHCP forneceu um IP da sua rede com as suas configurações, por você definido, execute o seguinte comando:

```
C\:> winipcfg
```

Renove o aluguel do seu IP, modifique a configuração do *max-lease-time* para 10 dias (lembre-se que deve ser colocado em segundos).

16.4.2 Windows 2000/XP

Abra *conexões de rede* e veja o *status / suporte / detalhes* de sua placa de rede, para verificar as informações obtidas pelo computador, do servidor DHCP.

Parte III

Protegendo a Rede

CAPÍTULO 17

Roteamento entre Redes

Nesse ponto temos uma rede local *10.10.XX.0*, o GNU/Linux com as duas placas de rede ativas e o DHCP em funcionamento, restando colocar o serviço de roteamento em funcionamento, para que se possa enxergar as outras redes.

Teremos que informar ao núcleo do GNU/Linux que ele fará o papel de um roteador, portanto, iremos modificar o arquivo `/etc/sysctl.conf` na segunda linha para `net.ipv4.ip_forward = 1` e depois ativar as modificações com o comando:

```
[root@curso root]# sysctl -p
```

Deste modo o GNU/Linux já está preparado para fazer roteamento e para tal, precisamos criar uma tabela de rotas para todas as redes que queremos acessar.

Por exemplo queremos criar uma rota entre a rede *10.10.XX.0* e a rede *10.10.YY.0* através da internet, usando os roteadores *Servidor XX* e *Servidor YY* como mostra a figura abaixo:

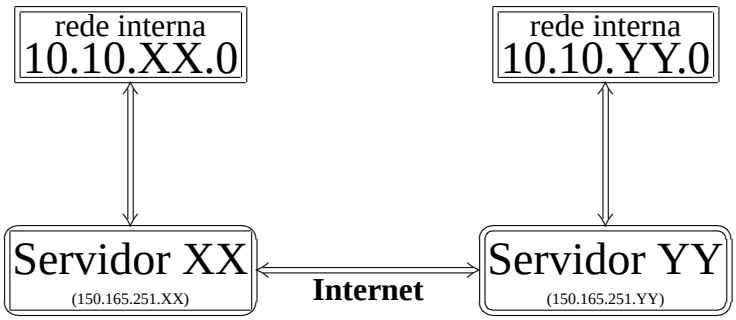


Figura 17.1: Rota entre a rede XX e YY

A seguir são dadas as possibilidades de criar essas rotas.

17.1 Rotas definidas Manualmente

Adicionaremos manualmente, ou seja adicionando as rotas, uma a uma, com o comando *route*, da seguinte forma:

```
[root@curso root]# route add -net REDE netmask MÁSCARA dev PLACA
gw DESTINO
```

Que no nosso caso será:

```
[root@curso root]# route add -net 10.10.YY.0 netmask 255.255.255.0
dev eth0 gw 150.165.251.YY
```

onde `YY` é a rede a qual queremos criar uma rota e `eth1` é por onde sairá os pacotes com destino à rede `10.10.YY.0`.

Para verificar se foi adicionado a rota na tabela de rotas, use um dos seguintes comandos:

```
[root@curso root]# route -n
[root@curso root]# netstat -rn

Tabela de Roteamento IP do Kernel
Destino         Roteador        MáscaraGen.     Opções Métrica Ref   Uso Ifac
10.10.YY.0      150.165.251.YY  255.255.255.0   U     0      0        0 eth0
10.10.XX.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
150.165.251.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        0 lo
0.0.0.0         150.165.251.XX  0.0.0.0         UG    0      0        0 eth0
```

Para se retirar uma tabela, usar o parâmetro `del` no lugar de `add`.

Exercício 17.1.1 *Criar as tabelas de rotas para as redes vizinhas a sua.*

Exercício 17.1.2 *Use o comando* `ping` *para um computador interno do seu vizinho.*

17.2 Rotas definidas Automaticamente

Para que as rotas sejam ativadas ao inicializar a rede, deve-se editar apenas uma vez o arquivo `/etc/sysconfig/static-routes`, como por exemplo:

```
route -add 10.10.YY.0 netmask 255.255.255.0 gw 150.165.251.YY
route -add 10.10.ZZ.0 netmask 255.255.255.0 gw 150.165.251.ZZ
route -add 10.10.WW.0 netmask 255.255.255.0 gw 150.165.251.WW
```

E quando a rede for ativada, essa tabela será toda implementada no kernel, ou seja basta executar o comando:

```
[root@curso root]# service network restart
```

Agora é só testar para ver se a tabela de rotas está realmente funcionando, fazendo a partir do cliente windows, o `ping` para um computador da rede vizinha, ou seja:

```
C\:> ping 10.10.YY.1
```

Verifique se o windows está visualizando os outros computadores windows das redes vizinhas, clicando em ambiente de rede.

CAPÍTULO 18

NAT

Neste ponto já estamos com a rede interna montada, o GNU/Linux com as duas placas de rede ativas e o DHCP em funcionamento, restando colocar o NAT e firewall em funcionamento.

O NAT não é um protocolo nem um padrão. O NAT é apenas uma série de tarefas que um roteador (ou equipamento equivalente) deve realizar para converter endereços IPs entre redes distintas.

Um equipamento que tenha o recurso de NAT (*Network Address Translation*) Tradução de Endereço de Rede) deve ser capaz de analisar todos os pacotes de dados que passam por ele e trocar os endereços desses pacotes de maneira adequada.

O tamanho de cada pacote é definido de acordo com as características da sua rede, geralmente nas redes TCP/IP o tamanho máximo é de 1500 bytes. Então se por exemplo for feito um *download* de um arquivo de 1 MB, o que realmente estará acontecendo é que o seu computador receberá vários pequenos pacotes (fragmentos) de 1500 bytes e no final irá agrupá-los para formar o arquivo original. Esses pacotes são compostos por duas partes:

Header (cabeçalho) - é nessa parte do pacote que o nat/firewall vai atuar, possui informações importantes como: origem do pacote, destino do pacote dentre outras.

Body (corpo) - é a parte que contem os dados propriamente ditos.

É importante conhecer esses conceitos sobre pacotes pois o nat/firewall no GNU/Linux irá atuar diretamente sobre eles fazendo a filtragem dos mesmos.

A seguir são dadas algumas utilidades para um NAT:

Conexões com a Internet via modem - A maioria dos provedores de acesso te dá um único endereço IP quando você conecta. Você pode mandar pacotes com qualquer endereço de origem que você quiser, mas apenas respostas para pacotes com esse endereço de origem retornarão para você. Se você quiser utilizar mais de uma máquina (uma rede local na sua casa) para conectar na Internet através deste único *link*, você precisará de NAT.

Esta é a maneira de NAT mais utilizada atualmente, e é conhecida como **Masquerading** no mundo GNU/Linux. Também conhecido de **Source-NAT (S-NAT)**, porque você altera o endereço de origem do primeiro pacote.

Múltiplos Servidores - Às vezes, você quer mudar o destino dos pacotes que chegam na sua rede. Freqüentemente isto ocorre porque (como dito acima) você tem apenas um endereço IP, mas quer que as pessoas alcancem servidores atrás do que tem o endereço IP real. Reescrevendo o destino dos pacotes que chegam, é possível implementar isso.

Uma variação comum disto é *load-sharing*, na qual a carga de pacotes é dividida entre máquinas. este tipo de NAT era chamado de *port-forwarding* em versões anteriores do GNU/Linux.

Proxy Transparente - Às vezes você pretende que cada pacote que atravesse sua máquina Linux é destinado a um programa específico. Isto é utilizado para fazer *proxies transparentes*[1]. O termo transparente refere-se ao fato de que sua rede não saberá que está lidando com um proxy, a não ser, é claro, que seu proxy não funcione.

18.1 Compreendendo o NAT

A Troca de Endereço de Rede (**N**etwork **A**ddress **T**ranslation), ou simplesmente **NAT**, trocar os endereços de entrada e/ou de saída de uma rede, tornando essa troca um componente importante de proteção da sua rede. Existem dois tipos de NAT, distintas:

NAT de Origem (Source NAT - SNAT) - ocorre quando o endereço de origem do primeiro pacote é alterado.

Por exemplo, quando se muda de onde vem uma conexão. Source NAT sempre ocorre no momento de *post-routing*, logo antes que o pacote deixe o frewall. O *Masquerading* que será visto mais adiante (ver 18.3.1 na página 81) é uma forma especializada de SNAT.

NAT de Destino (Destination NAT - DNAT) - ocorre quando o endereço de destino do primeiro pacote é alterado.

Por exemplo, quando se altera o destino final de uma conexão.

Destination NAT sempre ocorre no momento de *pre-routing*, quando o primeiro pacote está prestes a entrar no firewall. *Port forwarding*, *load sharing* e *proxy transparente* são exemplos de DNAT.

18.1.1 Criando regras NAT

Você precisa criar regras NAT as quais dirão ao kernel quais conexões serão alteradas, e como alterá-las. Para fazer isso, utiliza-se o versátil `iptables` e dizemos a ele para alterar a tabela NAT ao especificar a opção `-t nat`.

Observação 18.1.1 *Mas antes de usá-lo deve-se instalar o mesmo com o comando:*

```
[root@curso root]# apt-get install iptables
```

As regras da tabela NAT contém três listas chamadas *chains*, onde cada regra é examinada em ordem, até que alguma delas corresponda-se com o pacote analisado. O diagrama abaixo tenta ilustrar as *chains*.

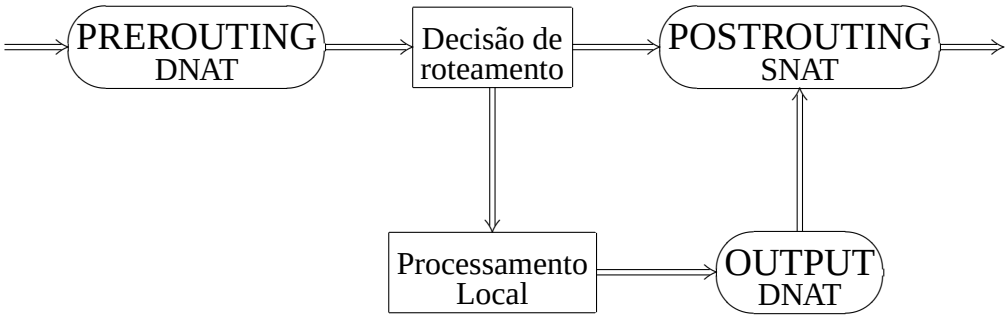


Figura 18.1: Esquema do NAT

As três *chains* são chamadas:

- **PREROUTING**: para Destination NAT, quando os pacotes estão prestes a entrar;

[1] Um proxy é um programa que fica entre sua rede e o mundo externo, controlando a comunicação entre os mesmos

- **POSTROUTING**: para Source NAT, assim que os pacotes saem;
- **OUTPUT**: para Destination NAT de pacotes gerados localmente.

Em cada um dos pontos acima, quando um pacote passa, é verificado a conexão com a qual ele está associado. Se é uma nova conexão, é verificada a *chain* correspondente na tabela NAT para ver o que fazer com a mesma. A resposta dada será idêntica para todos os outros pacotes relacionados com tal conexão.

18.2 Opções mais utilizadas do iptables

O `iptables` tem um número de opções padrão conforme a lista abaixo. Todas as opções com dois hífens (`--`) podem ser abreviadas, desde que o `iptables` ainda possa diferenciá-las das demais opções disponíveis.

Observação 18.2.1 *Se seu kernel tem suporte a* `iptables` *via módulos, você precisará carregar o módulo* `ip_tables.o` *antes com o seguinte comando:*

```
[root@curso root]# insmod ip_tables
```

Alguns parâmetros do iptables, que usaremos:

- Seleção da tabela: `-t`. Para todas as operações NAT, será necessária a opção `-t nat` para a tabela NAT.
- Adicionar uma nova regra no fim da chain: `-A`. Exemplo (`-A POSTROUTING`), ou `-I` para adicioná-la no início.
  Ex.: `-I PREROUTING`
- Especificar a origem dos pacotes que sofrerão NAT: `-s` ou `--source`

  Se o endereço de origem foi omitido, a regra servirá para qualquer origem.
- Especificar o destino dos pacotes que sofrerão NAT: `-d` ou `--destination`

  As opções `-s` e `-d` podem ser seguidas de um simples endereço IP `10.10.XX.1`, um nome (`www.meudominio.net`), ou um endereço de uma rede `10.10.XX.0/255.255.255.0` ou `10.10.XX.0/24`.

  Se o endereço de destino for omitido, a regra servirá para qualquer destino.
- Especificar as interfaces de entrada: `-i` ou `--in-interface`. Utilizado com a *chain* PREROUTING.
- Especificar as interfaces de saída: `-o` ou `--out-interface`. Utilizado com a *chain* POSTROUTING e OUTPUT.
- Especificar um protocolo: `-p` ou `--protocol`. Como por exemplo o TCP ou o UDP. Ao selecionar um protocolo, apenas pacotes do mesmo se encaixarão na regra. A principal razão para selecionar um protocolo, é que isso permite algumas opções extras.
- Especificar porta de destino e origem de um protocolo: `--destination-port` (`--dport`) e `--source-port` (`--sport`) respectivamente.

  Essas opções permitem especificar que apenas pacotes com certa porta de origem e destino se encaixarão na regra. Isso é útil para redirecionamento de requisições *web* (porta TCP 80 ou 8080).

  Podem ser usados os números das portas ou um nome do arquivo `/etc/services`. Todas as outras formas de se selecionar um pacote estão descritas em detalhes na página de manual do `iptables`.

Agora já sabemos como escolher os pacotes a serem tratados. Para completar, precisamos dizer ao kernel o que fazer com estes pacotes.

18.3 Source NAT

Você quer fazer SNAT mudar o endereço de origem das conexões para algo diferente. Isso é feito na *chain* POSTROUTING, logo antes de que o pacote saia da máquina. Isso é um detalhe muito importante, pois significa que qualquer tarefa da sua máquina GNU/Linux verá o pacote inalterado. Isso também significa que a opção `-o` (interface de saída) pode ser utilizada.

O SNAT é especificado com `-j SNAT`, e a opção `--to-source` demonstra um endereço IP, um conjunto de endereços IP, e uma porta opcional ou um conjunto de portas (apenas para os protocolos TCP e UDP).

Vejamos alguns exemplos:

Exemplo 18.3.1 *Mudando o endereço de origem para 1.2.3.4.*

```
[root@curso root]# iptables -t nat -A POSTROUTING -o eth1 -j SNAT
--to 1.2.3.4
```

Exemplo 18.3.2 *Mudando o endereço de origem para 1.2.3.4, 1.2.3.5 ou 1.2.3.6.*

```
[root@curso root]# iptables -t nat -A POSTROUTING -o eth1
-j SNAT --to 1.2.3.4-1.2.3.6
```

Exemplo 18.3.3 *Mudando o endereço de origem para 1.2.3.4, portas 1-1023.*

```
[root@curso root]# iptables -t nat -A POSTROUTING -p tcp -o eth1
-j SNAT --to 1.2.3.4:1-1023
```

Exemplo 18.3.4 *Listando as regras criadas nos exemplos anteriores:*

```
[root@curso root]# iptables -t nat -L
```

18.3.1 Mascaramento (Masquerading)

É um caso especial de SNAT, que só deve ser utilizado para endereços IP dinâmicos, como por exemplo, nas conexões através de provedores.

No mascaramento, não é necessário explicitar o endereço de origem: será utilizado o endereço de origem da interface de saída do pacote, porém, o mais importante é que se o *link* cair, o endereço de origem que estava sendo utilizado é descartado, dando lugar ao novo endereço de origem da interface quando o *link* for restabelecido.

Exemplo 18.3.5 *Por enquanto sem muitos detalhes, do mascaramento da rede interna, via modem.*

```
[root@curso root]# iptables -t nat -A POSTROUTING -o ppp0
-j MASQUERADE
```

Exemplo 18.3.6 *Para mascarar a nossa rede 10.10.XX.0, ou seja, a partir da execução do comando:*

```
[root@curso root]# iptables -t nat -A POSTROUTING -o eth0
-j MASQUERADE
```

os nossos computadores internos com IP 10.10.XX.yy, já poderão usar a internet.

Observação 18.3.1 *Lembre-se de verificar o arquivo* `/etc/sysctl.conf` *(ver capítulo 17 página 76).*

Após executar esse comando, a rede interna já pode utilizar a internet, pois foi informado ao kernel que todos os pacotes vindo da rede *10.10.XX.0* será enviado com tendo sido enviado pela placa *eth0*, ficando uma tabela de conversão, que na volta do pacote manda para o endereço IP original.

Exemplo 18.3.7 *se o computador 10.10.XX.5 manda um pacote para 150.165.250.1, esse pacote receberá um novo remetente, ficando a cargo do kernel fazer essa correspondência bionívuca entre quem mandou e quem respondeu ao pacote enviado.*

18.4 Destination NAT

O DNAT é feito na *chain* PREROUTING, assim que o pacote chega, isso significa que todas as outras tarefas da sua máquina GNU/Linux verão o pacote já indo para seu destino verdadeiro. A opção `-i` (interface de entrada) pode ser utilizada.

Para alterar o destino de pacotes gerados localmente, a chain OUTPUT deve ser utilizada, mas isso não é muito utilizado.

O DNAT é especificada utilizando `-j DNAT`, e a opção `--to-destination` especifica um endereço IP, um conjunto de IPs, e uma porta opcional ou um conjunto (*range*) de portas (apenas para protocolos TCP e UDP).

Exemplo 18.4.1 *Mudando destino para 5.6.7.8.*

```
[root@curso root]# iptables -t nat -A PREROUTING -i eth1
-j DNAT --to 5.6.7.8
```

Exemplo 18.4.2 *Mudando destino para 5.6.7.8, 5.6.7.9 ou 5.6.7.10.*

```
[root@curso root]# iptables -t nat -A PREROUTING -i eth1
-j DNAT --to 5.6.7.8-5.6.7.10
```

Exemplo 18.4.3 *Mudando destino do tráfego web para 5.6.7.8, porta 8080.*

```
[root@curso root]# iptables -t nat -A PREROUTING -p tcp --dport 80
-i eth1 -j DNAT --to 5.6.7.8:8080
```

Exemplo 18.4.4 *Redirecionar pacotes locais com destino a 1.2.3.4 para loopback.*

```
[root@curso root]# iptables -t nat -A OUTPUT -d 1.2.3.4
-j DNAT --to 127.0.0.1
```

18.4.1 Redirecionamento para um servidor PROXY

Há um caso especializado de DNAT chamado redirecionamento, que é uma simples conveniência, a qual equivale a fazer DNAT para o endereço da própria interface de entrada.

Exemplo 18.4.5 *Mandando tráfego web da porta 80 para squid proxy (transparente) na maquina local:*

```
[root@curso root]# iptables -t nat -A PREROUTING -i eth1
-p tcp --dport 80 -j REDIRECT --to-port 3128
```

Observação 18.4.1 *Lembre-se que o Squid precisa ser configurado para que o mesmo, funcione como sendo um proxy transparente e para tanto deve-se acrescentar as linhas no arquivo* `squid.conf` *(ver capítulo 12 na página 50):*

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Exemplo 18.4.6 *Mandando os pacotes web da porta 80 para um squid proxy em outra máquina na rede, por exemplo:*

```
[root@curso root]# iptables -t nat -A PREROUTING -i eth1
-p tcp --dport 80 -j DNAT --to 150.165.250.1:3128
```

18.5 Regras mais elaboradas de NAT

Há alguns refinamentos do NAT os quais a maioria das pessoas nunca utilizará. Eles estão aqui documentados apenas para os mais curiosos.

Seleção de múltiplos endereços em um conjunto - Se um conjunto de endereços IP é dado, o IP escolhido para uso é o que está sendo menos utilizado pelas conexões conhecidas. Isso porvê um balanceamento de carga (*load-balancing*) bem primitivo.

Criando mapeamentos NAT nulos - Você pode utilizar `-j ACCEPT` para aceitar uma conexão sem que haja nenhuma espécie de NAT.

Comportamento padrão do NAT - O comportamento padrão é alterar a conexão o mínimo possível, modificando apenas o definido pelo usuário nas suas regras. Isso significa que as portas não serão "remapeadas" a não ser que as regras mandem que elas sejam.

Mapeamento de portas de origem implícita - Mesmo quando nenhum tipo de NAT é requisitado para uma conexão, a alteração da porta de origem pode ocorrer de forma implícita, quando uma conexão requisita uma porta que está em uso. Considere um caso de mascaramento, que gera esse tipo de situação:

1. Uma conexão web estabelecida pela máquina 192.1.1.1 vinda da porta 1024 para *yahoo.com* porta 80.
2. Tal conexão é mascarada pela máquina de masquerading a fim de usar o endreço de IP de origem 1.2.3.4, que é o endereço de origem da máquina que realiza o masquerading.
3. A máquina que faz o masquerading tenta realizar a conexão web com *yahoo.com* porta 80 vinda de 1.2.3.4 (endereço de origem da interface de saída) porta 1024.
4. O código de NAT alterará a porta de origem da segunda conexão para a porta 1025, assim as duas conexão não ficam em conflito.

Quando esse mapeamento de origem implícito ocorre, as portas são divididas em três classes:

- Portas menores que 512
- Portas entre 512 e 1023
- Portas acima de 1024.

Uma porta **NUNCA** será mapeada implicitamente para uma classe diferente.

O que ocorre quando NAT falha - Se não há formas de mapear uma conexão conforme requisitado pelo usuário, a mesma será descartada (DROP). Isso também se aplica a pacotes que não foram classificados como parte de qualquer conexão, devido a malformação, ou falta de memória da máquina, etc.

Mapeamentos múltiplos, overlap e conflitos - Você pode ter regras NAT que mapeiam pacotes para o mesmo conjunto, o código NAT suficientemente inteligente para evitar conflitos. Não há nenhum problema em mapear os endereços de origem 192.168.1.1 e 192.168.1.2 para 1.2.3.4.

Além disso, você pode mapear para endereços IP reais e utilizados, desde que tais endereços passem pela máquina responsável pelo mapeamento. Então, se você tem uma rede válida (1.2.3.0/24), mas possui uma rede interna que utilize um IP privado (192.168.1.0/24), você pode realizar SNAT do endereço 192.168.1.0/24 para a rede 1.2.3.0, sem medo de quaisquer conflitos:

```
[root@curso root]# iptables -t nat -A POSTROUTING
-s 192.168.1.0/24 -o eth1 -j SNAT --to 1.2.3.0/24
```

A mesma lógica se aplica aos endereços utilizados pela própria máquina que realiza NAT: essa é a forma que o mascaramento funciona (compartilhando o endereço da interface de saída entre os pacotes mascarados e os gerados localmente).

Você também pode mapear os mesmos pacotes para alvos distintos, e estes serão compartilhados.

Exemplo 18.5.1 *Se você não quer mapear NADA para o endereço 1.2.3.5, você faria:*

```
[root@curso root]# iptables -t nat -A POSTROUTING
-s 192.168.1.0/24 -o eth1 -j SNAT --to 1.2.3.0-1.2.3.4
--to 1.2.3.6-1.2.3.254
```

Alterando destino de pacotes gerados localmente - Se o destino de um pacote gerado localmente é alterado (pela chain OUTPUT), e isso leva o pacote a passar por uma interface diferente, e então o endereço de origem também é alterado para o endereço da interface de saída.

Exemplo 18.5.2 *Mudando o destino de um pacote loopback para sair por eth1 irá resultar na mudança do endereço de origem de 127.0.0.1 para o endereço da eth1; ao contrário dos outros mapeamentos de origem (que são feitos depois de tudo, na chain POSTROUNTING) esse é realizado imediatamente. Naturalmente, todos esses mapeamentos são revertidos no momento que os pacotes de resposta chegam.*

CAPÍTULO 19

Firewall

A idéia de se ter um *firewall* é de colocar um "porteiro" que gerencia as entradas e saídas de pacotes de sua rede, utilizando um filtro de pacotes.

Um filtro de pacotes é um software que analisa o cabeçalho (*header*) dos pacotes enquanto eles passam, e decide o destino do pacote como um todo. Ele pode decidir entre descartar (*DROP*) o pacote (descartando-o como se nunca o tivesse recebido), aceitar (*ACCEPT*) o pacote (deixar o pacote seguir seu caminho), ou algo mais complicado que isso.

No GNU/Linux, filtragem de pacotes está implementada diretamente no kernel (como um módulo ou diretamente compilado), e há várias coisas interessantes que podem ser feitas com os pacotes, mas o princípio geral é analisar o cabeçalho dos pacotes e decidir o que ser feito com cada pacote.

As principais características de um *firewall* são:

Controle - quando uma máquina GNU/Linux é utilizada para conectar sua rede interna em outra rede (a Internet, por exemplo) há a oportunidade de permitir certo tipo de tráfego, e rejeitar outro. Por exemplo, o cabeçalho do pacote contém o endereço de destino do pacote, logo você pode evitar que os pacotes saiam em direção a um endereço.

Segurança - quando uma máquina GNU/Linux é a única coisa entre o caos da Internet e sua calma e organizada rede, é bom saber que é possível restringir o tráfego vindo do mundo externo.

Por exemplo, você pode permitir que qualquer tipo de pacote saia da sua rede, mas você é preocupado com o famoso ataque do "ping da morte" (*Ping of Death*) vindo de mal-feitores do mundo externo.

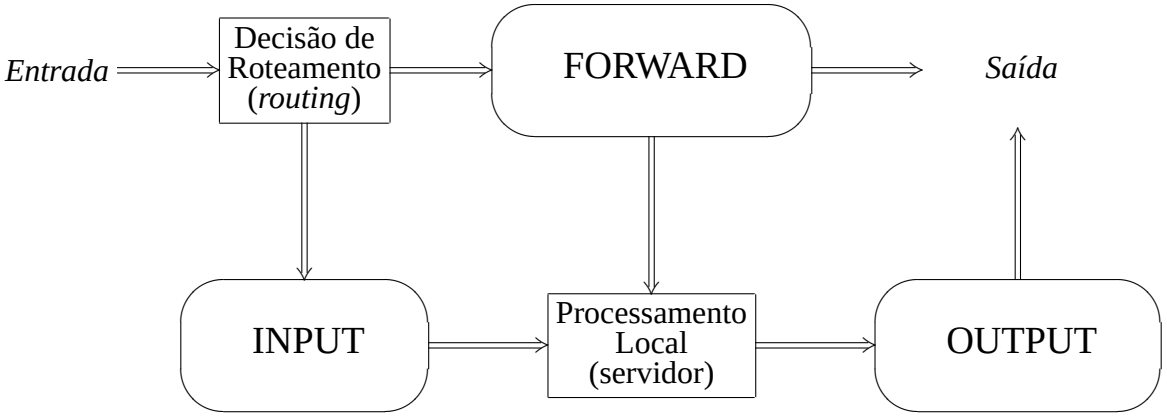
Como outro exemplo, você pode não gostar da idéia de permitir que qualquer um da rede externa conecte-se via telnet na sua máquina GNU/Linux, mesmo que todas as contas da máquina tenham senhas. Talvez você queira (como a maioria das pessoas) ser apenas um observador, e não um servidor. Simplesmente não deixe ninguém conectar, dizendo ao filtro de pacotes para rejeitar pacotes requisitando a criação de novas conexões.

Vigilância - às vezes uma máquina mal configurada na rede local pode decidir enviar pacotes descontroladamente para o mundo externo. É interessante dizer ao filtro de pacotes para te informar se algo anormal ocorrer.

19.1 Filtrando pacotes no GNU/Linux

A ferramenta que conversa com o kernel e fala quais são os pacotes a serem filtrados é também o *iptables* e portanto também utiliza *chains*, onde cada *chain* é uma lista de regras.

Cada regra diz *se o cabeçalho do pacote se parece com isso, então aqui está o que deve ser feito com o pacote*. Se a regra não associa-se com o pacote, então a próxima regra na *chain* é consultada. Se não há mais regras a consultar, o kernel analisa a política da *chain* para decidir o que fazer, ou seja a *chain* tem uma política geral caso não faça parte de algumas das regras. Em um sistema preocupado com segurança, a política diz ao kernel para rejeitar (DROP) o pacote. Veja no diagrama abaixo:



1. Quando o pacote chega, pela placa de rede por exemplo, o kernel analisa o destino do pacote, o chamado roteamento (*routing*).

2. Se ele é destinado a própria máquina, o pacote desce no diagrama, indo para a *chain* INPUT. Se ele passar pela *chain* INPUT, então a máquina recebe o pacote.

3. Depois, se o kernel não tem suporte a *forwarding*, ou não sabe como repassar (*forward*), o pacote é descartado.

4. Se há suporte a *forwarding* e o pacote é destinado a outra interface de rede (se existir outra), o pacote vai para a *chain* FORWARD. Se ele for aceito (ACCEPT), ele será enviado.

5. Finalmente, um programa rodando na máquina *firewall* pode enviar pacotes. Esses pacotes passam pela *chain* OUTPUT imediatamente, se ela aceitar o pacote, ele continua seu caminho, caso contrários ele é descartado.

Para definir o que e como será filtrado, utilizaremos os seguintes parâmetros do iptables:

- **INPUT**: o que pode entrar
- **OUTPUT**: o que pode sair
- **FORWARD**: o que pode passar

Há várias formas de manipular regras dentro de uma chain:

- Adicionar uma nova regra na *chain*: `-A`
- Inserir uma nova regra em alguma posição da *chain*: `-I`
- Substituir uma regra em alguma posição da *chain*: `-R`
- Apagar uma regra em alguma posição da *chain*: `-D`
- Apagar a primeira regra que associa (com alguma condição) numa *chain*: `-D`

Abaixo seguem opções para se gerenciar as *chain*.

- Criar nova *chain*: `-N`
- Apagar uma *chain* vazia: `-X`
- Mudar a política de uma *chain built-in*: `-P`
- Listar as regras de uma *chain*: `-L`
- Apagar todas as regras de uma *chain*: `-F`
- Zerar os contadores de pacotes e bytes de todas as regras de uma *chain*: `-Z`

Antes dos comandos do *iptables* rodarem[1], não haverão regras em quaisquer *chain* (*INPUT*, *FORWARD* e *OUTPUT*) e todas as *chain* terão a política *ACCEPT*. A política padrão da *chain* FORWARD pode ser alterada para o parâmetro `forward=0` para o módulo `iptable_filter`.

19.2 Operações em uma única regra

Este é o arroz com feijão da filtragem de pacotes, ou seja, manipular as regras. Usualmente, você provavelmente utilizará os comandos para adicionar (`-A`) e apagar (`-D`). Os outros (`-I` para inserir e `-R` para substituir) são apenas extensões destes conceitos.

Cada regra especifica uma série de condições que o pacote deve atender, e o que fazer com o mesmo, se ele atendê-las (um alvo *target*).

Exemplo 19.2.1 *Você pode desejar descartar todos os pacotes ICMP vindos do IP 127.0.0.1. Então neste caso nossas condições são que o protocolo precisa ser ICMP e o endereço de origem deve ser 127.0.0.1. Nosso alvo (target) é DROP. O IP 127.0.0.1 é a interface "loopback", que existirá mesmo que você não tenha nenhuma conexão de rede real. Pode-se utilizar o programa "ping" para gerar esse tipo de pacote*[2].

```
[root@curso root]# ping -c 1 127.0.0.1 ; iptables -L
[root@curso root]# iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP
[root@curso root]# ping -c 1 127.0.0.1 ; iptables -L
```

Pode-se ver, no exemplo acima, que o primeiro *ping* é bem sucedido (a opção `-c 1` diz ao `ping` para mandar um único pacote). Então foi adicionada (`-A` à *chain* `INPUT`, uma regra especificando que pacotes vindos de 127.0.0.1 (`-s 127.0.0.1`) com o protocolo ICMP (`-p icmp`) devem ser mandados para DROP (`-j DROP`), ou seja descartados.

Logo depois, testamos nossa regra, usando um segundo `ping`. Haverá uma pausa antes que o programa desista de esperar pelo pacote que nunca virá.

Pode-se apagar regras de duas maneiras. Primeiramente, desde que sabemos que existe apenas uma regra na *chain* INPUT, podemos utilizar um número para designar a regra.

Exemplo 19.2.2 *Para apagar a regra número 1 na chain INPUT.*

```
[root@curso root]# iptables -D INPUT 1 ; iptables -L
```

A segunda forma, é fazer o mesmo que faríamos para adicionar a regra, trocando `-A` por `-D`. Isso é útil quando você tem uma *chain* muito complexa e não quer contar para descobrir que a regra 37 deve ser apagada. Neste caso usaria-se:

[1] Cuidado, algumas distribuições rodarão *iptables* em seus *scripts* de inicialização

[2] Simplesmente manda um ICMP tipo 8 (requisição de eco) que é respondido pelos hosts com um ICMP tipo 0 (resposta de eco)

Exemplo 19.2.3 *Para remover a regra criada no primeiro exemplo, basta usar o comando:*

```
[root@curso root]# iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP
```

Observação 19.2.1 *A sintaxe do* `-D` *deve ter exatamente as mesmas opções que seriam passadas para a opção* `-A` *(ou* `-I` *ou* `-R`*). Se existem várias regras idênticas na mesma chain, apenas a primeira será apagada.*

19.3 Operações em uma chain

Uma funcionalidade muito importante do iptables é a habilidade de juntar regras em *chains* (cadeias). Você pode dar o nome que quiser para as *chains*, mas é recomendada a utilização de minúsculas para evitar confusão com as *chains* padrões ou com os alvos (*targets*). Nomes de *chains* podem ter até 31 letras.

19.3.1 Criando uma nova chain

Criemos uma nova *chain* chamada de teste. Utiliza-se as opções `-N` ou `--new-chain`, com o seguinte comando:

```
[root@curso root]# iptables -N teste
```

É tão simples assim. Agora é possível acrescentar regras conforme descrito em todo o documento.

19.3.2 Apagando uma chain

Apagar uma *chain* é tão simples quanto criá-la, utilizando as opções `-X` ou `--delete-chain`.

```
[root@curso root]# iptables -X teste
```

Há uma série de restrições ao se apagar uma *chain*: ela deve estar vazia (veja em esvaziando uma *chain*, logo abaixo) e ela não deve ser alvo de NENHUMA regra.

Se uma *chain* não for especificada, todas as *chains* definidas pelo usuário serão apagadas, desde que seja possível.

19.3.3 Esvaziando uma chain

Há uma forma muito simples de retirar todas as regras de uma *chain*, utilizando as opções `-F` ou `--flush`.

```
[root@curso root]# iptables -F FORWARD
```

Se uma *chain* não for especificada, todas as *chains* serão esvaziadas.

19.3.4 Listando uma chain

Pode-se listar todas as regras de uma *chain* utilizando as opções *-L* ou *–list*.

O valor *refcnt* listado para cada *chain* definida por usuário é o número de regras que têm tal *chain* como alvo (*target*). Este valor deve ser zero (e a *chain* deve estar vazia) antes que essa *chain* possa ser apagada. Se o nome da *chain* é omitido, todas as *chains* são listadas, até as vazias.

Há três opções que podem acompanhar `-L`.

- `-n` (numérico) é muito útil uma vez que previne que o iptables tente resolver os endereços IP, o que (se você utiliza DNS assim como a maioria das pessoas) causaria grandes atrasos se o DNS não está configurado corretamente, ou você está filtrando requisições DNS. Essa opção também faz as portas TCP e UDP serem impressas como números em vez de nomes.

- `-v` mostra todos os detalhes das regras, como os contadores de pacotes e bytes, as comparações TOS, e as interfaces. Caso tal opção não seja passada, esses valores são omitidos.

  Note que os contadores de pacotes e bytes são impressos com os sufixos “K”, “M” ou “G” para 1.000, 1.000.000 e 1.000.000.000 respectivamente.

- `-x` (expandir números) os números serão impressos inteiros, independente de seu tamanho.

19.3.5 Zerando contadores

É útil zerar os contadores. Isso pode ser feito com a opção `-Z` ou `--zero`.

Exemplo 19.3.1 *Considere os seguintes comandos:*

```
[root@curso root]# iptables -L FORWARD
[root@curso root]# iptables -Z FORWARD
```

alguns pacotes passariam pelos comandos `-L` *e* `-Z`*. Por isso, você pode utilizar* `-L` *e* `-Z` *em conjunto, para zerar os contadores enquanto estes são lidos.*

19.4 Especificações para filtragem

Foi visto o uso do `-p` para especificar o protocolo, e `-s` para especificar o endereço de origem, mas existem outras opções que podem ser utilizadas para especificar outras características dos pacotes. O que segue é uma explicação exaustiva de cada especificação:

Endereços IP de origem e destino - `-s`, `--source` ou `--src` para origem e `-d`, `--destination` ou `--dst` para destino.

Endereços IP podem ser especificados em quatro formas diferentes.

A mais comum é utilizar o nome completo, como `localhost` ou `www.meudominio.net`. A segunda é dizer o IP, como `127.0.0.1`.

A terceira e a quarta formas permitem a especificação de um grupo de endereços IP, como `10.10.10.0/24` ou `10.10.10.0/255.255.255.0`.

Ambas especificam qualquer endereço IP de `10.10.10.0` até `10.10.10.255`; os dígitos depois da `/` dizem quais partes do endereço IP são significantes[3]. Para especificar qualquer endereço IP pode-se utilizar */0*[4], conforme descrito abaixo:

```
[root@curso root]# iptables -A INPUT -s 0/0 -j DROP
```

Isso raramente é utilizado, uma vez que o efeito da regra acima é o mesmo que se não fosse especificada nenhuma origem.

Inversão - `!` (negação)

Muitas flags, incluindo `-s` (ou `--source`) e `-d` (ou `--destination`) podem ter seus argumentos precedidos de `!` (negação) para associar-se com endereços DIFERENTES aos passados à opção.

Exemplo 19.4.1 *O* `-s ! localhost` *associa-se com qualquer pacote que não venha de* `localhost`.

[3] */32* ou */255.255.255.255* é o padrão (associando o endereço IP inteiro)
[4] NOTA: *-s 0/0* é redundante.

Protocolo - `-p` ou `--protocol`

O protocolo pode ser um número (se você sabe o valor numérico dos protocolos) ou um nome para casos especiais como *TCP*, *UDP* ou *ICMP*. Digitar em maiúsculas ou minúsculas não faz diferença, `tcp` funciona tão bem como *TCP*.

Se o nome do protocolo é precedido de `!`, a fim de invertê-lo, como `-p! TCP`, a regra especificará todos os pacotes que não são TCP.

Interface de entrada e saída - `-i` ou `--in-interface` para entrada e `-o` ou `--out-interface` para saída.

Pacotes atravessando a *chain* INPUT não possuem interface de saída, logo qualquer regra utilizando `-o` nessa *chain* nunca aplicar-se-á.

Da mesma forma, pacotes atravessando a *chain* OUTPUT não têm interface de entrada, logo qualquer regra utilizando `-i` nessa *chain* nunca aplicar-se-á.

Apenas pacotes passando pela *chain* FORWARD têm interfaces de entrada e saída.

Não há nenhum problema em especificar uma interface que ainda não existe; a regra não associar-se-á com quaisquer pacotes até que a interface torne-se ativa. Isso é extremamente útil para links discados PPP (usualmente com a interface `ppp0`) e similares.

Como um caso especial, um nome de interface terminando com um `+` vai associar-se com todas as interfaces (existam elas ou não) que comecem com aquela *string*.

Exemplo 19.4.2 *Para especificar uma regra que se associa com todas as interfaces PPP, a opção* `-i ppp+` *deve ser criada.*

O nome da interface pode ser precedido de `!` para se associar com um pacote que não é representado pelas interface(s) especificada(s).

Fragmentos - às vezes um pacote é muito grande para ser enviado todo de uma vez. Quando isso ocorre, o pacote é dividido em fragmentos, e é enviado como múltiplos pacotes. Quem o recebe, remonta os fragmentos reconstruindo o pacote.

O problema com os fragmentos é que o fragmento inicial tem os campos de cabeçalho completos (IP + TCP, UDP e ICMP) para examinar, mas os pacotes subseqüentes só têm um pequeno grupo de cabeçalhos (somente IP, sem os campos dos outros protocolos). Logo examinar o interior dos fragmentos subseqüentes em busca de cabeçalhos de outros protocolos (como extensões de TCP, UDP e ICMP) é impossível.

Se você está fazendo acompanhamento de conexões ou NAT, todos os fragmentos serão remontados antes de atingirem o código do filtro de pacotes, então você não precisa se preocupar sobre os fragmentos.

Caso contrário, é importante entender como os fragmentos são tratados pelas regras de filtragem. Qualquer regra de filtragem que requisitar informações que não existem não será válida. Isso significa que o primeiro fragmento é tratado como um pacote qualquer. O segundo e os seguintes não serão. Então uma regra `-p TCP --sport www` (especificando `www` como porta de origem) nunca se associar-se-á com um fragmento (exceto o primeiro). O mesmo se aplica à regra oposta `-p TCP --sport ! www`.

De qualquer forma, você pode especificar uma regra especial para o segundo e os fragmentos que o sucedem, utilizando a *flag* `-f` (ou `--fragment`). Também é interessante especificar uma regra que não se aplica ao segundo e os outros fragmentos, precedendo a opção `-f` com `!`.

Geralmente é reconhecido como seguro deixar o segundo fragmento e os seguintes passarem, uma vez que o primeiro fragmento será filtrado, logo isso vai evitar que o pacote todo seja remontado na máquina destinatária. De qualquer forma, há *bugs* que levam à derrubada de uma máquina através do envio de fragmentos. A decisão é sua.

Observação 19.4.1 *Os pacotes mal formados (pacotes TCP, UDP e ICMP muito pequenos para que o código do firewall possa ler as portas ou o código e tipo dos pacotes ICMP) são descartados quando ocorrem tais análises. Então os fragmentos TCP começam na posição 8.*

Exemplo 19.4.3 *A regra a seguir vai descartar quaisquer fragmentos destinados ao endereço 10.10.XX.1:*

```
[root@curso root]# iptables -A OUTPUT -f -d 10.10.XX.1 -j DROP
```

19.4.1 Extensões TCP

As extensões TCP são automaticamente carregadas se é especificada a opção `-p tcp`. Elas provém as seguintes opções (nenhuma associa-se com fragmentos).

- **--tcp-flags** - seguida por uma `!` (opcional), e por duas *strings* indicando *flags*, permite que sejam filtradas *flags* TCP específicas. A primeira *string* de *flags* é a máscara: uma lista de *flags* que serão examinadas. A segunda *string* diz quais flags devem estar ativas para que a regra se aplique.

  Exemplo 19.4.4 *Regra que indica que todas as flags devem ser examinadas (*`ALL` *é sinônimo de* `SYN, ACK, FIN, RST, URG, PSH`*), mas apenas* `SYN` *e* `ACK` *devem estar ativadas.*

  ```
  [root@curso root]# iptables -A INPUT --protocol tcp
  --tcp-flags ALL SYN,ACK -j DROP
  ```

  Também há um argumento `NONE` que significa nenhuma *flag*.

- **--syn** - opcionalmente precedido por `!`, é um atalho para `--tcp-flags SYN, RST, ACK SYN`.

- **--source-port** - seguido por `!` (opcional), e uma única porta TCP, ou um conjunto de portas. As portas podem ser descritas pelo nome, conforme listado em `/etc/services`, ou pelo número. Conjuntos são dois nomes de portas separados por `:`, ou (para especificar uma porta maior ou igual à especificada) uma porta sucedida por `:`, ou (para especificar uma porta menor ou igual à especificada), uma porta precedida por `:`.

- **--sport** - é sinônimo de `--source-port`.

- **--destination-port** e **--dport** funcionam de forma idêntica a `--source-port`, a única diferença é que elas indicam a porta de destino, e não a porta de origem.

- **--tcp-option** - seguida por `!` (opcional) e um número, associa-se com um pacote com a opção TCP igual ao do número passado. Um pacote que não tem um cabeçalho TCP completo é automaticamente descartado se há uma tentativa de examinar suas opções TCP.

Uma explicação sobre as *flags* TCP

Às vezes é util permitir conexões TCP em uma única direção, mas não nas duas. Por exemplo, você permitirá conexões em um servidor web externo, mas não conexões vindas deste servidor.

Uma tentativa ingênua seria bloquear pacotes TCP vindos do servidor. Infelizmente, conexões TCP necessitam de pacotes bidirecionais para um funcionamento perfeito.

A solução é bloquear apenas os pacotes utilizados para requisitar uma conexão. Tais pacotes são chamados pacotes SYN (tecnicamente eles são pacotes com a *flag* SYN marcada, e as *flags* RST e ACK desmarcadas, mas dizemos pacotes SYN como atalho). Ao não permitir tais pacotes, nós impedimos conexões vindas do servidor.

A opção `--syn` é utilizada para isso mas, só é válida para regras que especificam TCP como protocolo. Por exemplo, para especificar conexões TCP vindas do endereço `192.168.1.1: -p TCP -s 192.168.1.1 --syn`.

Essa opção pode ser invertida se precedida por `!`, o que significa qualquer pacote exceto os que iniciam conexões.

19.4.2 Extensões UDP

Essas extensões são automaticamente carregadas se a opção `-p udp` é especificada. Ela provê as opções `--source-port`, `--sport`, `--destination-port` e `--dport` conforme detalhado para o TCP acima.

19.4.3 Extensões ICMP

Essas extensões são automaticamente carregadas se a opção `-p icmp` é especificada. Ela só possui uma opção diferente das demais:

- **-icmp-type** - seguida por `!` (opcional), e um nome de tipo ICMP (`host-unreachable`), ou um tipo numérico (exemplo `3`), ou um tipo numérico e código separados por `/` (exemplo `3/3`). Uma lista de tipos ICMP é passada utilizando-se `-p icmp --help`.

19.4.4 Outras extensões

Outras extensões no pacote *netfilter* são extensões de demonstração, que (caso instaladas) podem ser chamadas com a opção `-m`.

mac - este módulo deve ser explicitamente especificado com `-m mac` ou `--match mac`. Ele é usado para associar-se com o endereço *ethernet* (MAC) de origem do pacote, e logo só é útil nas chains PREROUTING e INPUT. Só provê uma única opção:

- `--mac-source` seguida por `!` (opcional) e um endereço *ethernet* passado em notação hexa, exemplo `--mac-source 00:60:...:B7`.

limit - este módulo deve ser explicitamente especificado com `-m limit` ou `--match limit`. É usado para restringir a taxa de pacotes, e para suprimir mensagens de log. Vai fazer com que a regra seja válida apenas um número de vezes por segundo (por padrão 3 vezes por hora, com um limite máximo def 5). Possui dois argumentos opcionais:

- `--limit` seguido de um número; especifica a média máxima de pacotes (ou LOGs, etc) permitida por segundo. Pode-se especificar a unidade de tempo, usando `/second`, `/minute`, `/hour` ou `/day`, ou abreviações dos mesmos (assim, `5/second` é o mesmo que `5/s`).
- `--limit-burst` seguido de um número, indicando o máximo de entradas antes do limite tornar-se válido.

Essa extensão pode ser usada com o alvo (target) LOG para criar registros (logs) limitados por taxa de incidência. Para entender o funcionamento disso, olhe a regra abaixo, que loga pacotes com os parâmetros padrão de limite:

```
[root@curso root]# iptables -A FORWARD -m limit -j LOG
```

Na primeira vez que essa regra é analisada, o pacote será logado; na realidade, uma vez que o padrão máximo é 5, os cinco primeiros pacotes serão logados. Depois disso, passar-se-ão vinte minutos antes de que essa regra faça um novo logo, independente do número de pacotes que entrarem.

Além disso, a cada vinte minutos que se passam sem que um pacote associe-se com a regra, o contador é diminuído em uma unidade. Logo, se nenhum pacote associar-se com a regra em 100 minutos, o limite estará zerado, voltando ao estágio inicial.

Observação 19.4.2 *Não é possível criar uma regra com tempo de recarga superior a 59 horas, então se você configura uma taxa média de um por dia, seu limite máximo deve ser menor que 3.*

Esse módulo também pode ser usado para evitar uma série de ataques do tipo negativa de serviço (*denial of service 'DoS'*) com uma taxa mais rápida, a fim de aumentar a sensibilidade do sistema.

Exemplo 19.4.5 *Proteção contra Syn-flood:*

```
[root@curso root]# iptables -A FORWARD -p tcp --syn
-m limit --limit 1/s -j ACCEPT
```

Exemplo 19.4.6 *Port scanner suspeito:*

```
[root@curso root]# iptables -A FORWARD -p tcp --tcp-flags
SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT
```

Exemplo 19.4.7 *Ping da morte:*

```
[root@curso root]# iptables -A FORWARD -p icmp --icmp-type
echo-request -m limit --limit 1/s -j ACCEPT
```

owner - esse módulo tenta associar-se com várias características do criador do pacote, para pacotes gerados localmente. Só é válido na *chain* OUTPUT, e mesmo assim alguns pacotes (como respostas de ping ICMP) podem não ter dono, e nunca serão analisados pela regra.

- `--uid-owner userid` Associa-se com o pacote, se este foi criado por um processo com o *userid* igual ao passado na opção.
- `--gid-owner groupid` Associa-se com o pacote, se este foi criado por um processo com o *groupid* igual ao passado na opção.
- `--pid-owner processid` Associa-se com o pacote, se este foi criado por um processo com o *proccessid* igual ao passado na opção.
- `--sid-owner sessionid` Associa-se com o pacote, se este foi criado por um processo com o *sessionid* igual ao passado na opção.

unclean - esse módulo *experimental* deve ser explicitamente especificado com `-m unclean` ou `--match unclean`. Ele faz diversas checagens de sanidade nos pacotes. Esse módulo não foi auditado, e não deve ser utilizado como dispositivo de segurança (ele provavelmente torna as coisas piores, uma vez que provavelmente está cheio de bugs). Não possui opções.

19.4.5 Checagens de estado dos pacotes (state match)

O critério para filtragem de pacotes mais útil é provido pela extensão `state` que interpreta a análise do controle da conexão feita pelo módulo *ip_conntrack*. Essa extensão é altamente recomendada.

Especificando `-m state` a opção `--state` torna-se disponível, a qual é uma lista dos estados possíveis dos pacotes separada por vírgula (a opção `!` indica para que a regra não se associe com os estados passados). Os estados são:

- **NEW** Um pacote que cria uma nova conexão.
- **ESTABLISHED** Um pacote que pertence a uma conexão existente (um pacote de resposta, um pacote saindo por uma conexão na qual já houveram respostas).
- **RELATED** Um pacote relacionado, mas que não faz parte, de uma conexão existente, como um erro ICMP, ou (com o módulo FTP carregado), um pacote estabelecendo uma conexão de dados FTP.
- **INVALID** Um pacote que não pôde ser identificado por alguma razão: isso inclui falta de memória e erros ICMP que não correspondem a qualquer conexão conhecida. Geralmente tais pacotes devem ser descartados.

19.4.6 Especificações de alvo (target)

Agora que sabemos quais as análises podem ser feitas em um pacotes, precisamos de uma forma de dizer o que fazer com os pacotes que passam nas condições que estabelecemos. Isso é chamado de alvo (target) da regra.

Há dois alvos bem simples: DROP (descartar) e ACCEPT (aceitar). Se a regra se associa com o pacote e seu alvo é um desses dois, nenhuma outra regra é consultada: o destino do pacote já foi decidido.

Há dois tipos de alvos diferentes dos descritos acima, as extensões e as *chains* definidas por usuários.

Chains definidas por usuários - uma funcionalidade que o *iptables* herdou do ipchains é a possibilidade do usuário criar novas *chains*, além das três padrão (INPUT, FORWARD e OUTPUT). Por convenção, *chains* definidas pelo usuário são escritas em minúsculas, para diferenciá-las.

Quando um pacote associa-se com uma regra cujo alvo (target) é uma *chain* definida pelo usuário, o pacote passa a ser analisado pelas regras dessa *chain* definida pelo usuário. Se a *chain* não decide o que deve ser feito com o pacote, o mesmo volta a ser analisado pela *chain* padrão que continha a regra que o levava para a *chain* definida pelo usuário.

Chains definidas por usuário podem ter como alvo outras *chains* definidas por usuário (mas não faça loops, pois os pacotes serão descartados se entrarem em loop).

Extensões ao iptables: Novos alvos (targets) - uma extensão-alvo consiste em um módulo do kernel, e uma extensão opcional ao *iptables* para prover opções de linha de comando. Há diversas extensões na distribuição padrão do *netfilter*:

- **LOG** - Esse módulo provê *logging* dos pacotes submetidos. Possui as seguintes opções adicionais:
  - `--log-level` seguido de um número de nível ou nome. Os nome válidos (sensíveis a maiúsculas/minúsculas) são `debug`, `info`, `notice`, `warning`, `err`, `crit`, `alert` e `emerg`, correspondendo a números de 7 até 0. Veja a página de manual do *syslog.conf* para uma explicação mais detalhada desses níveis.

  - `--log-prefix` seguido de uma *string* de até 29 caracteres, esta será adicionada no início da mensagem de log, permitindo melhor identificação da mesma.

  Esse módulo é últil após de uma comparação por limite (*limit match*), assim, você não enche seus logs.

- **REJECT** - Esse módulo tem o mesmo efeito de DROP, a não ser que o remetente tenha enviado uma mensagem de erro ICMP *port unreachable* (porta inacessível). A mensagem de erro ICMP não será enviada se (veja RFC 1122):
  - O pacote que está sendo filtrado era uma mensagem de erro ICMP no início, ou um tipo desconhecido de ICMP
  - O pacote sendo filtrado era um fragmente sem cabeçalho
  - Já enviamos muitas mensagens de erro ICMP para o mesmo destinatário recentemente.

  O REJECT também tem o argumento opcional `-reject-with` que altera o pacote de resposta utilizado (veja a página de manual).

19.5 Escolhendo política

Já discutimos sobre o que ocorre quando um pacote chega ao fim de uma *chain* padrão quando o caminho dos pacotes através das *chains*. Nesse caso, a política da *chain* determina o destino do pacote. Apenas as *chains* padrão (INPUT, OUTPUT e FORWARD) têm políticas, porque se um pacote chega ao fim de uma *chain* definida por usuário, o caminho do pacote continua pela *chain* anterior.

A política pode ser tanto ACCEPT (aceitar), DROP (descartar) quanto REJECT (rejeitar mandando uma massagem), por exemplo:

```
[root@curso root]# iptables -P FORWARD DROP
```

19.6 Alguns exemplos

Exemplo 19.6.1 *O arquivo* `/etc/sysconfig/iptables` *(criado pelo* `iptables-save`*) temos as seguintes regras:*

- *Pacotes vindos da rede 10.10.YY.0 com destino a rede 10.10.XX.0, serão aceitos;*
- *Pacotes com destino a rede 10.10.XX.0 com protocolo TCP e UDP com portas variando de 0 a 1024, serão rejeitadas como se fossem portas não reconhecidas, para o remetente.*

```
# Generated by iptables-save v1.2.4 on Thu Nov 28 16:50:17 2004
*filter
:INPUT ACCEPT [1544:164023]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1030:77438]
-A FORWARD -s 10.10.YY.0/255.255.255.0 -d 10.10.XX.0/255.255.255.0
                                                        -j ACCEPT
-A FORWARD -d 10.10.XX.0/255.255.255.0 -p tcp -m tcp --dport 0:1024
                      -j REJECT --reject-with icmp-port-unreachable
-A FORWARD -d 10.10.XX.0/255.255.255.0 -p udp -m udp --dport 0:1024
                -j REJECT --reject-with icmp-port-unreachable COMMIT
# Completed on Thu Nov 28 16:50:17 2004
```

Exemplo 19.6.2 *Neste mais elaborado, que usa regras definidas pelo administrador (snmp, snort, srvwwwque), ficará a cargo do leitor, descobrir o que faz esse firewall.*

```
:snmp
:snort
:srvwww
snort -i eth1 -s 150.165.0.0/16 -d 150.165.254.2 -j ACCEPT
snort -i eth1 -d 150.165.128.0/17 -p udp -m multiport
--dports 161,162,163,164 -j snmp
snort -i eth1 -d 150.165.128.0/17 -p tcp -m multiport
--dports 161,162,163,164 -j snmp
snort -i eth1 -d 150.165.254.2 -j DROP
snort -i eth1 -p tcp --dport 80 -j srvwww
snmp -j DROP
srvwww -d 150.165.250.1 -j ACCEPT
srvwww -d 150.165.135.1 -j ACCEPT
srvwww -d 150.165.250.40 -j ACCEPT
srvwww -j DROP
INPUT -p tcp -s 150.165.255.0/24  --dport 616 -j ACCEPT
INPUT -p tcp -s 150.165.254.64/29 --dport 616 -j ACCEPT
INPUT -p tcp --dport 616 -j DROP
INPUT -p 89 -s 150.165.255.0/24  -j ACCEPT
INPUT -p 89 -s 150.165.254.64/29 -j ACCEPT
INPUT -p 89 -j DROP
FORWARD -s ! 150.165.128.0/17 -i eth0 -j DROP
FORWARD -i eth0 -o eth1 -m multiport -p tcp --dports 3128 -j DROP
FORWARD -j snort -o eth0
FORWARD -p udp --dport 137:139 -j DROP
FORWARD -p tcp --dport 137:139 -j DROP
FORWARD -p tcp --dport 1433 -j DROP
FORWARD -p tcp --dport 1434 -j DROP
```

Exemplo 19.6.3 *Um exemplo de NAT, que também ficará para o leitor, tentar descobrir:*

```
PREROUTING -s 150.165.255.130  -i eth0 -p tcp --dport 80 -j ACCEPT
PREROUTING -s 150.165.255.133  -i eth0 -p tcp --dport 80 -j ACCEPT
PREROUTING -s 150.165.255.135  -i eth0 -p tcp --dport 80 -j ACCEPT
PREROUTING -s 150.165.182.5    -i eth0 -p tcp --dport 80 -j ACCEPT
PREROUTING -s 150.165.128.0/17 -i eth0 -p tcp --dport 80 -j REDIRECT
                                                      --to-ports 3128
```

Apêndice

A.1 Conta no Terminal

Ao ligar o computador será mostrado quatro opções de sistemas e programas, que são:

Microsoft Windows NT (tm) - uma partição com o sistema operacional Windows NT.

Debian Serge - uma partição com o sistema operacional Linux Debian Serge, que será a base para o nosso curso e portanto a nossa opção.

Teste de memória - programa para fazer teste na memória do computador.

Recuperar Disco - uma partição especial, de acesso restrito, que serve para reconfigurar e reinstalar as duas primeiras partições.

Para utilizar o terminal do laboratório `CPU - XX NTI/CTC` usaremos a conta **linux** e senha **cursonti**. Note que o sistema instalado é o *Debian*.

Para usar os programas instalados, neste simples ambiente gráfico, basta clicar o botão esquerdo do *mouse*, e aparecerá um menu com as opções, donde destacamos os seguintes:

VMware - programa que será bastante utilizado, para a criação dos nossos servidores virtuais (ver anexo A.3 na página 98).

Terminal - abre uma janela do modo terminal, que nos utilizaremos para montar o CD de instalação do Conectiva Linux 10, com o seguinte comando: `mount /cd-inst`

Kill - para parar algum processo que eventualmente tenha travado.

Exit - para fechar a conta.

A.2 Editor de Texto vi

Nesta seção serão vistos os comandos básicos do editor de texto mais utilizado e comum no meio UNUX/Linux, o **vi**. Mas existem outros, que para a primeira utilização, seja mais fácil de se utilizar, que são:

- **pico**: Um editor bastante simples de usar, pois os principais comandos estão colocados na parte inferior do editor.

  Para instalá-lo, basta usar o seguinte comando:

  ```
  [root@curso root]# apt-get install pico
  ```

- **joe**: Outro editor bastante simpático, principalmente para aqueles com um pouco mais de idade, que já utilizou o editor *WordStar*, onde os comando são baseados nas teclas *Ctrl+K*.

  Para instalá-lo, basta usar o seguinte comando:

  ```
  [root@curso root]# apt-get install joe
  ```

- **emacs**: Editor também muito utilizado.

  Para instalá-lo, basta usar o seguinte comando:

  ```
  [root@curso root]# apt-get install emacs
  ```

A.2.1 Comandos básicos

Teclas	Descrição
:help	Ajuda para o vi.
i	Inserir um texto, deve-se teclar *esc* ao terminar a digitação
teclas	Movimentação dentro do texto.
:q	Sair do Vi (o arquivo deve estar salvo).
:q!	Sair do vi forçado sem salvar.
:wq	Sair do vi salvando o arquivo que está sendo editado.
:e nome	Abrir o arquivo nome.
:w	Salvar o Arquivo.
:w nome	Salvar o arquivo corrente com o nome escolhido.
dd	Apaga a linha corrente colocando-a na memória.
p	Cola a última entrada colocada na memória.
:r nome	Insere o arquivo nome no ponto onde está o cursor.
/padrão	Procura padrão em todo o texto corrente.
:u	Desfaz a última ação.

A.3 VMware

O VMware da *VMware, Inc.* é um software realmente fantástico, do tipo que realmente vale à pena testar[1] e dependendo do caso, até comprá-lo[2]. A página oficial na internet é a: `http://www.vmware.com`

Roda em vários sistemas e cria máquinas virtuais que simulam um PC completo dentro de uma janela (ou em tela cheia), permitindo instalar praticamente qualquer sistema operacional para a plataforma *x86*. É possível abrir várias máquinas virtuais simultaneamente e rodá-los lado a lado, com várias versões do Linux e Windows, BeOS, DOS e o que mais você tiver em mãos.

Essa facilidade permite a criação de redes heterogêneas com diversos sistemas operacionais o que possibilita fazer testes em rede, sistemas, proteção, simulações e outras coisinhas mais.

[1] Na página oficial pode-se baixar uma versão para teste, com validade de 30 dias, para tanto, basta fazer o cadastro

[2] Preço na internet U$ 189,00

No nosso curso, utilizaremos a versão *VMware Workstation 4.5* em um sistema com processador *Intel Celeron 500 MHz*, com 160 *MB* de memória *RAM*, *HD* de 4.0 *GB* e com o sistema Linux.

A.3.1 Usando o VMware

Cada computador virtual é composto por arquivos e que portanto, caso tenha dado errado basta simplesmente apagá-los.

Para criar um novo computador virtual basta clicar em *File/New Virtual Machine...* ou simplesmente *Ctrl+N*, donde aparecerá uma janela de ajuda, para a instalação, que poderá ser típica (*Typical*) ou otimizada (*Custom*) de acordo com o desejado que é dado na seção 2.2 na página 12.

As opções serão dadas por:

- Sistema: *Linux*
- Versão: *Other Linux 2.6x Kernel*
- Nome do computador virtual: *Conectiva*
- Localização: */home/linux/conectiva*
- Memória: *48 MB*
- Rede: *Use bridged network* - dá o acesso direto à rede na qual o computador está ligado.
- Adaptador de entrada/saída: *Buslogic*
- Disco: *Create a new virtual disk* - para a criação de um HD virtual.
- Tipo do HD virtual: *IDE* - o mais comum e mais barato no mercado.
- Capacidade do HD: *0.7 GB* - 700 MB é o suficiente.
- Arquivo do HD: *hd.vmdk* - o HD será um arquivo cujo o nome o usuário escolhe.

Feito isso o computador virtual foi criado, agora falta instalar o CD-ROM, do qual iniciaremos (*boot*) para a instalação do Conectiva.

Isto é feito clicando no CD-ROM, onde colocaremos a opção *Use ISO image* localizado em */mnt/cds/cl10_cd1.iso*.

Agora é só clicar em *Start virtual machine* e esperar que o computador virtual seja ligago, com uma BIOS e dando *boot* pelo CD-ROM, começando a instalação (ver 2.3 página 12).

A.4 Discos e Partições

O Linux como todos os sistemas operacionais modernos tem suporte aos discos, sejam eles fixos ou removíveis, mas a forma como os discos são tratados difere um pouco dos utilizados nos sistemas Windows (95/98/2000/NT). Para ilustrar tais diferenças de maneira prática, ao final desse capítulo será feito um estudo de caso com a adição de um disco rígido no RedHat/Linux . Para que um disco seja usado no Linux eles precisam antes passar por um processo que inclui: particionamento, formatação, criação de um sistema de arquivos e montagem do sistema de arquivos, cada um desses passos será explicado com detalhes a seguir.

A.4.1 Particionamento

Um disco rígido pode ser dividido em partes menores chamadas de partições, onde cada partição funciona como se fosse um outro pequeno disco rígido. Com essa idéia cada uma das partições são independentes, isto permite por exemplo que os arquivos essenciais ao sistema sejam separados dos arquivos dos usuários, ou ter vários sistemas operacionais no mesmo disco rígido e como as partições são independente qualquer problema numa delas não afetará, a princípio, as demais. Dito isso sempre vem a questão: Quantas partições devo ter no disco rígido? A resposta é no mínimo uma e o número máximo fica por conta do administrador. No caso de servidor recomenda-se uma partição para raiz (esta é a única que é obrigatória), outra para a área de troca (swap), uma para o */var* (a área onde ficam os mails e arquivos de log) e outra para o */usr* (onde ficarão a maioria dos programas), podendo haver ainda uma área reservada ao arquivos dos usuários (*/home*).

Cada disco rígido é referenciado por um arquivo no Linux, por exemplo: o primeiro disco rígido IDE está associado ao arquivo */dev/hda*, o segundo será o */dev/hdb* e assim por diante, acontecendo o mesmo para os discos SCSI, onde o primeiro seria o */dev/sda*. Como foi dito antes, cada partição é como se fosse um pequeno disco rígido, portanto é razoável que cada partição também tenha um arquivo associado, por isso a primeira partição do primeiro disco IDE está associada ao arquivo */dev/hda1*, a segunda partição seria o */dev/hda2*.

A.4.2 Formatação e Criação do Sistema de Arquivos

Após o processo de particionamento, segue-se a formatação onde todos os dados, caso existam, serão apagados do disco rígido . Note que cada uma das partições criadas deve ser formatada.

O próximo passo é a criação, no disco, de um sistema de arquivos, mas antes de aprender como se cria um sistema de arquivo (filesystem) no Linux é importante definir um sistema de arquivo como sendo a maneira como o Sistema Operacional organiza os arquivos no disco, ou seja o sistema operacional escreve no disco uma série de informações que ele mesmo modificará e usará para saber onde os arquivos se encontram, para saber quanto de espaço existe numa determinada partição, etc.

A criação de um sistema de arquivo é a escrita de tais informações no disco, que estava vazio devido ao processo de formatação.

O Linux tem seu próprio sistema de arquivo (ext2), mas ele suporta muitos outros sistema de arquivos tais como: vfat (Win95), msdos, hpfs (OS/2), ntfs (NT), nfs, iso9660 (cd-rom), etc.

A.4.3 Montagem do Sistema de Arquivos

A criação de um sistema de arquivo lhe garante que o sistema operacional pode e sabe como gravar e ler os arquivos do disco rígido, mas para que isso possa ser efetivamente feito é preciso associar o sistema de arquivo do disco a um diretório, processo este que damos o nome de montagem do sistema de arquivo. Na verdade o que acontece é que com a montagem o conteúdo da mídia (os arquivo e diretórios), ficam disponíveis (visíveis) abaixo deste diretório, que é comumente chamado de ponto de montagem.

A.4.4 Estudo de Caso

No caso de ser necessário adicionar um segundo disco rígido IDE de 3.0 GB, a um computador com GNU/Linux já instalado no primeiro disco rígido IDE, que conterá os arquivos de dois usuários chamados *Chico* e *Zefa* em partições diferentes. Este disco rígido contém uma partição FAT32 que será destruída. Afim de melhorar a compreensão, seguiremos os passos abaixo

1. O primeiro passo é o particionamento do HD, para isso utilizaremos um programa chamado fdisk, e com ele iremos destruir a partição FAT32, e dividir o HD em duas partições (ext2) iguais de 1.5 GB, onde cada uma conterá o `home` de um dos usuários.

```
[root@curso root]# fdisk /dev/hda

Command (m for help): p    (listaremos a partições existentes no HD)

Disk /dev/hda: 128 heads, 63 sectors, 781 cylinders
Units = cylinders of 8064 * 512 bytes

   Device Boot   Start      End   Blocks        Id     System
/dev/hda1   *        1         782    30208+      83   FAT32 Win95


Comand (m for help): d     (apagaremos a partição FAT32)
Partition number:  1

Comand (m for help): n     (Iniciaremos a criação da  primeira partição)
First cylinder (1-782, default 782): (Digite apenas Enter para usar o default)
Using default value 1
Last cylinders or +size or +sizeM  or sizeK: +1500M
```

O mesmo processo deverá ser repetido para a segunda partição e em seguida gravar as alterações da seguinte maneira:

```
Comand (m for help): w
```

2. Acabado o processo de particionamento será feita a criação do sistema de arquivos, através do comando `mkfs` (*make filesystem*). A opção `-c` será usada para checar as partições, procurando por blocos defeituosos.

   ```
   [root@curso root]# mkfs -c /dev/hdb1 ; mkfs -c /dev/hdb2
   ```

3. Agora será criado o ponto de montagem do sistema de arquivo, que equivale a criar os diretórios sob os quais estes sistemas de arquivos poderão ser acessados:

   ```
   [root@curso root]# mkdir /home/chico ; mkdir /home/zefa
   ```

4. E finalmente faremos a montagem dos sistemas de arquivos através do comando `mount`:

   ```
   [root@curso root]# mount -t ext2 /dev/hda1 /home/chico

   [root@curso root]# mount -t ext2 /dev/hda2 /home/zefa
   ```

Pronto, já podemos usar o novo disco, mas do jeito que fizemos até agora teríamos que a todo *reboot* do computador fazer novamente a montagem dessas partições. Para que isso seja feito de modo automático devemos adicionar as seguinte linha no arquivo `/etc/fstab`:

```
/dev/hdb1 /home/chico      ext2      defaults 1  2
/dev/hdb2 /home/zefa       ext2      defaults 1  2
```

Onde o primeiro campo representa o arquivo de dispositivo associado ao dispositivo físico. O segundo campo corresponde ao ponto de montagem. O terceiro é o tipo do sistema de arquivos. O quarto começa especificar as opções de montagem associadas ao sistema de arquivos (filesystem), podendo ser especificada várias opções desde que separadas por vírgula. A opção padrão corresponde a uma abreviação para as opções exibida na tabela abaixo.

Tipo	Descrição
rw	Pode-se ler e escrever no sistema de arquivos.

suid	Os arquivos executáveis com atributos set-user-id e set-group-id terão estes atributos interpretados corretamente.
dev	Os arquivos especiais (bloco ou caracter) que estiverem definidos no sistema de arquivos não podem ser ignorados.
nouser	Proíbe-se que usuários comuns monte o sistema de arquivos.
auto	O sistema de arquivos é montado no boot.
exec	Os programas contidos podem ser executados.
async	A escrita e leitura ao sistema de arquivos é assíncrona; ou seja, ao se modificar um arquivo, a atualização não é imediata.

O quinto e sexto campos contém informações associadas com a checagem do sistema de arquivo.

A.5 Estrutura dos Diretórios

A estrutura dos sistemas de arquivos do Linux prevê um agrupamento que permite maior organização de dados, o que aumenta a funcionalidade do sistema. Os comandos estão todos em uma determinada área, todos os arquivos de dados em uma outra, documentação em uma terceira, e assim por diante. Além disso, o diretório raiz geralmente não contém nenhum arquivo, exceto, em algumas distribuições, pela imagem de inicialização do sistema. Todos os outros arquivos estão em subdiretórios do raiz.

Veja a estrutura básica de diretórios abaixo:

Diretório	Descrição
/	chamado de diretório raiz é específico de cada máquina. Pode ficar tanto em um disco físico quanto na memória da máquina ou em uma unidade de rede. É o diretório principal, que contém todos os arquivos e diretórios do sistema.
/bin	contém o mínimo necessário para funcionar e poder ser manuseado pelo administrador. Serão necessárias ferramentas que se encontram em outros diretórios para que a máquina fique operacional. A maioria dos programas possui o seu arquivo executável neste diretório.
/dev	é o local onde ficam armazenadas as referências aos dispositivos presentes na máquina, para o controle destes dispositivos. Esse diretório contém apontadores para, por exemplo, o drive de disquetes, os discos da máquina, terminais virtuais, portas de acesso seriais e paralelas, etc. Os controladores são automaticamente criados durante a instalação do sistema e posteriormente podem ser criados através do comando MAKEDEV.
/home	contém os diretórios pessoais dos usuários e suas configurações.
/proc	fornece informações sobre o kernel e sobre os processos que estão rodando no momento, além de informações sobre a utilização de alguns dispositivos. Esse diretório não ocupa espaço nenhum em disco e as informações ali presentes são geradas apenas quando solicitadas.
/usr	contém comandos, bibliotecas, programas, páginas de manual e outros arquivos que não mudam mas que se fazem necessários para a operação normal do sistema.
/boot	contém informações para o gerenciador de inicialização do sistema. É aqui que normalmente fica o arquivo contendo o kernel da máquina e informações para o carregador do sistema operacional.

/etc	é um dos mais importantes diretórios da máquina. Nele ficam a maioria dos arquivos de configuração e manipulação dos serviços essenciais ao sistema, a maioria dos arquivos de configuração de acesso a rede e de comunicação, arquivos de configuração do sistema de janelas X, arquivos de configuração do idioma do sistema, de atualizações, enfim, de muitas funcionalidades da máquina.
/lib	ficam as bibliotecas básicas do sistema. Elas são compartilhadas por diversos programas, principalmente os que se encontram no diretório raiz.
/var	contém arquivos que possuem dados variáveis. Neste diretório estão arquivos e diretórios de *spool*, arquivos de *log*, arquivos de configuração de correio eletrônico, entre outros.
/sbin	contém ferramentas de interesse do superusuário e que geralmente são usadas por serviços básicos da máquina. Ficam nesse diretório programas como os responsáveis pela carga de módulos do kernel, ativação e interrupção das interfaces de rede, manutenção dos sistemas de arquivos e outras atividades.
/mnt	utilizado para o acesso a dispositivos de mídia, como disquetes e CD-ROM. Ele é utilizado como ponto de montagem para a maioria destes dispositivos.
/tmp	serve como repositório para arquivos temporários, sendo utilizado para programas que são executados após a ativação do sistema, ou seja, este diretório serve como espaço extra para vários programas e aplicações.
/opt	alguns programas são projetados para serem instalados sob este diretório.

Apesar das diferentes partes acima serem chamadas de diretórios, não há obrigatoriedade que elas estejam separadas. Elas podem estar facilmente no mesmo sistema de arquivos em uma pequena máquina utilizada por um único usuário que deseje mantê-las de uma forma mais simplificada.

Alguns destes diretórios podem ser montados em suas próprias partições. Suponha que, por questões de espaço ou segurança, é desejável que o diretório dos usuários esteja em uma outra partição. Então, é criada uma outra partição que será montada no diretório /home e que terá, por exemplo, o mesmo sistema de arquivos utilizado pelo sistema de arquivos raiz. Essa separação em um sistema de arquivos à parte é interessante pois facilita determinadas tarefas administrativas como gerenciamento da quantidade de espaço que cada usuário pode utilizar e a manutenção de cópias de segurança[4].

Portanto, a árvore de diretórios pode estar dividida ainda em diferentes sistemas de arquivos, dependendo do tamanho de cada disco e de quanto espaço será alocado para cada finalidade.

A.6 Instalação e Atualização de Programas com RPM

A instalação e atualização de programas é uma tarefa que foi bastante facilitada com a introdução dos sistema de gerenciamento de pacotes, no caso do Conectiva (RedHat Linux) é o *RPM*[3]. Com esse sistema, os administradores não precisam mais fazer o download dos fontes dos programas, depois descompactá-los, compilá-los e finalmente instalá-los. Agora tem-se num único arquivo o programa já compilado pronto para ser instalado e o que é melhor, o sistema de gerenciamento checa se as dependências para este pacote estão instaladas, mantém um banco de dado de todos os pacotes instalados e facilitam as desinstalações. Enfim, os sistemas gerenciadores de pacotes facilitam a vida de um administrador de sistemas, ainda mais quando o número de hosts a se administrar é grande. O rpm é o comando usado para gerenciar os pacotes: instalar, desinstalar,

[3] *RedHat Package Manager* Gerenciador de Pacotes RedHat)

atualizar, etc.

Entendendo os Pacotes

Na maioria dos casos, um aplicativo não é formado apenas por um único arquivo executável, mas sim por um grande número de arquivos, como arquivos de tradução (potfiles), arquivos auxiliares[4], arquivos de documentação e arquivos de configuração. Assim, é bastante complexa a tarefa de instalação de um aplicativo manualmente; ainda mais difícil é a manutenção de seus arquivos. Muitas vezes, uma nova versão do aplicativo torna alguns de seus arquivos obsoletos e então, o administrador do sistema tem de apagar o arquivo antigo para evitar um acúmulo de arquivos inúteis.

Um pacote é um arquivo que, além de conter os arquivos necessários para a instalação de um determinado aplicativo, contém também as informações necessárias para que o gerenciador de pacotes possa instalar, manter e remover programas.

Um arquivo típico de pacote se parece com o seguinte:

`emacs-21.3-37707cl_i386.rpm`

Estes nomes de arquivos seguem um padrão. Os nomes dos arquivos de pacotes RPM contém informações sobre a versão e a arquitetura às quais se destinam. No caso acima, o arquivo diz que se trata do pacote do *Emacs*, versão *21.3*, release *37707* no Conectiva Linux (cl) e que ele foi criado para plataforma Intel 386. Todos os arquivos de pacotes RPM têm o formado *pacote-versao-release.arquitetura.rpm*, para permitir reconhecer-se visualmente o arquivo. Você poderá encontrar também pacotes com nomes no formato *pacote-versao.src.rpm*. Esses pacotes não contém os binários de um aplicativo para serem instalados, mas sim os seus arquivos fontes e são (na grande maioria dos casos) independentes de arquitetura. Será mostrado mais adiante como utilizar esse tipo de pacote.

Apesar de várias distribuições de Linux utilizarem pacotes do tipo RPM, isso não significa que eles sejam iguais e que um pacote feito originalmente para uma distribuição irá funcionar perfeitamente em outra. Além dos arquivos que compõem um determinado aplicativo, o pacote RPM contém também informações de como instalar, em que local copiar os arquivos, como configurar, etc. Essas informações adicionais podem ser diferentes de uma distribuição para outra, então, para evitar problemas, o administrador deve procurar sempre utilizar pacotes feitos especificamente para a sua distribuição.

A.6.1 Pacotes e Dependências

Uma das características de um sistema Linux é a modularização de seus componentes e o uso intensivo de bibliotecas compartilhadas, e isso se aplica também aos pacotes de aplicativos. É bastante comum que um programador ao escrever um aplicativo utilize bibliotecas de funções já existentes, mas não as inclua no pacote do seu aplicativo uma vez que, se todos fizessem isso, seria um tremendo desperdício de espaço e de banda de rede no caso de se disponibilizar esse pacote na Internet.

Assim, ao se instalar um determinado pacote, pode ser necessário primeiro instalar um outro pacote que contenha os ”pré-requisitos”para a sua instalação. Pode-se dizer então que o primeiro pacote depende, ou tem como dependência o segundo pacote. Os pacotes RPM contém a informação das dependências de cada aplicativo, e essa informação é utilizada pelo gerenciador de pacotes ao se instalar um novo pacote no sistema, ou remover um pacote instalado. Caso a instalação de um novo pacote requeira um outro, ou se tente remover um pacote instalado do qual outro depende, o gerenciador de pacotes alertará o administrador sobre isso.

Outra característica do uso de pacotes é que podem haver aplicativos que se encontram distribuídos por vários pacotes. O XFree86 é um exemplo disto. As bibliotecas se encontram em um pacote, os binários em outro, e os módulos específicos a cada grupo de placas de vídeo em outros. A organização de um aplicativo em vários pacotes permite que o administrador instale somente

[4] Arquivos diversos como imagens de ícones ou arquivos de dados.

aqueles pacotes que são realmente necessários, evitando ter que instalar ou fazer o *download* de componentes que não serão utilizados.

A.6.2 O Banco de Dados RPM

Quando o administrador instala um pacote RPM, o gerenciador de pacotes não se limita a instalar os arquivos, ele também mantém um banco de dados que armazena informações sobre todos os arquivos instalados. O banco de dados registra onde estão os arquivos e quais versões estão instaladas, assim, quando o usuário precisar instalar um pacote que necessite de um arquivo específico, ele saberá se esse pacote já existe ou precisa ser instalado. Ao remover um pacote, o gerenciador consultará o banco de dados, e assim obterá uma lista de quais arquivos que podem ser removidos.

Em algumas situações especiais, o banco de dados pode ficar corrompido. Para isso, o gerenciador de pacotes conta com funções especiais para recuperar a integridade deste banco, como será mostrado mais adiante.

A.6.3 O Gerenciador de Pacotes RPM

O RPM é um poderoso gerenciador de pacotes que permite ao administrador instalar, remover e obter informações sobre pacotes. Com o RPM é possível também reparar um banco de dados danificado, construir pacotes a partir de arquivos fonte, verificar a assinatura digital de pacotes RPM, simular uma instalação, entre outras coisas. O RPM oferece uma grande gama de funcionalidades, no entanto, serão mostrados aqui apenas as mais utilizadas, devendo o administrador consultar a documentação do aplicativo para obter mais detalhes.

Instalando e Atualizando Pacotes RPM

Instalar pacotes utilizando utilizando o comando rpm é bastante simples. Abra um terminal e, estando no mesmo diretório onde está o pacote que se deseja instalar, digite:

```
[root@curso root]# rpm -i pacote-versao.i386.rpm
```

A opção -i informa ao comando que você deseja instalar um pacote. É recomendado que o administrador utilize também as opções v (verbose - modo detalhado) e h (inclui linhas de progresso) ao instalar um pacote, ou seja, rpm -ivh mostra mais informações sobre o andamento do processo.

Caso o pacote a ser instalado necessite que outro pacote tenha sido previamente instalado, o RPM apresentará uma mensagem de erro mostrando quais dependências não foram atendidas para a instalação deste pacote. Será necessário então instalar os pacotes indicados para poder então instalar com sucesso o pacote que originalmente se queria instalar. Será mostrado mais adiante um modo de fazer isso automaticamente.

Caso se deseje atualizar um pacote já instalado no sistema por uma versão mais nova em vez do -i utilize a opção -U, assim o pacote mais antigo será removido, o pacote novo será instalado e as configurações serão mantidas.

Removendo Pacotes RPM

Para remover um pacote com o rpm abra um terminal e digite:

```
[root@curso root]# rpm -e nome_do_pacote
```

Ao remover um pacote não é necessário utilizar o nome do pacote completo, isto é, não utilize nome-versão-release.rpm, mas sim apenas o nome do pacote. Utilize a versão apenas se existirem duas versões do pacote instalado e se desejar remover uma delas.

Caso algum pacote instalado no sistema dependa do pacote que se deseja remover, o RPM não fará a desinstalação e emitirá uma mensagem de erro informando que dependências seriam

quebradas com isto. Caso seja realmente necessário, o administrador poderá utilizar a opção –nodeps para evitar que o RPM faça essa verificação, mas isso não é recomendado, pois poderá danificar o sistema. Obtendo Mais Informações Sobre os Pacotes

O RPM pode ser utilizado para obter mais informações sobre os pacotes, tanto os já instalados quanto os não instalados. A forma básica para o modo de consulta é rpm -q[opção] pacote. Será mostrado a seguir, algumas das consultas mais comuns utilizando-se o RPM:

Obtendo informações de um pacote:

```
[root@curso root]# rpm -qi bash

Name        : bash         Relocations: (not relocatable)
Version     : 2.05b        Vendor: Conectiva
Release     : 42313cl      Architecture: i386
Group       : Base         License: GPL
Size        : 772573
Install Date: Qui 04 Mar 2004 11:26:13 BRT
Build Date  : Sex 26 Dez 2003 01:52:22 BRT
Build Host  : mapi8.distro.conectiva
Source RPM  : bash-2.05b-42313cl.src.rpm
Signature   : (none)
Packager    : Conectiva S.A. <security at conectiva.com.br>
URL         : http://www.gnu.org/software/bash
Summary     : GNU Bourne Again Shell (bash)
Description :
Bash é um interpretador de comandos compatível com sh,
que executa comandos lidos da entrada padrão ou de um arquivo.
Bash também incorpora características úteis das shells Korn e
C (ksh e csh). O Bash tem sido desenvolvido para ser uma
implementação compatível com a especificação IEEE Posix para
shells e ferramentas (IEEE Working Group 1003.2).
```

Para obter informações de um pacote que não está instalado, utilize a opção -p, seguida do nome do arquivo do pacote.

O administrador poderá utilizar o rpm para obter informações sobre as dependências de um pacote. Poderá descobrir que outros pacotes dependem do pacote que se quer consultar, bem como descobrir de que pacotes o pacote consultado necessita. Os dois exemplos abaixo mostram como descobrir essas duas informações, respectivamente:

```
[root@curso root]# rpm -q --whatrequires glib

gtk+-1.2.10-45456cl
xmms-1.2.10-52293cl
bonobo-1.0.22-46388cl
```

```
[root@curso root]# rpm -q --requires glib

/sbin/ldconfig
/sbin/ldconfig
libc.so.6
libc.so.6(GLIBC_2.0)
libc.so.6(GLIBC_2.1.2)
libc.so.6(GLIBC_2.1.3)
libc.so.6(GLIBC_2.3)
libdl.so.2
libdl.so.2(GLIBC_2.0)
libdl.so.2(GLIBC_2.1)
libpthread.so.0
libpthread.so.0(GLIBC_2.0)
```

```
libpthread.so.0(GLIBC_2.3.2)
rpmlib(CompressedFileNames) <= 3.0.4-1
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(ScriptletInterpreterArgs) <= 4.0.3-1
```

Para descobrir a qual pacote pertence um arquivo do sistema utilize a opção -qf arquivo, como no exemplo abaixo:

```
[root@curso root]# rpm -qf /bin/bash
bash-2.05b-42313cl
```

E como um último exemplo, veja como listar todos os pacotes instalados no sistema:

```
[root@curso root]# rpm -qa
[root@curso root]# rpm -qa | grep xfree
```

Recuperando o Banco de Dados RPM

Se por um motivo ou outro o banco de dados de pacotes corromper-se, o rpm pode recuperá-lo. Caso seja necessário fazer isso, basta utilizar o comando abaixo:

```
[root@curso root]# rpm --rebuilddb
```

Utilizando Pacotes de Fontes

Usualmente, além dos pacotes que contém os binários dos aplicativos, encontram-se também pacotes contendo os arquivos fontes[5] dos aplicativos. O pacote fonte, ao ser instalado, copia seus arquivos para a o diretório */usr/src/rpm*, permitindo assim que os usuários do sistema possam estudar como determinado programa é feito e até mesmo alterá-lo de acordo com sua vontade.

É possível utilizar um pacote fonte para construir um pacote RPM contendo os arquivos binários adequados à arquitetura da máquina em que ele será utilizado. Se você possui o arquivo fonte e deseja construir um pacote contendo os binários para que o programa possa efetivamente ser instalado, utilize o comando `rpm --rebuild pacote.src.rpm`. O pacote será construído e colocado no diretório */usr/src/rpm/RPMS/arquitetura*, onde arquitetura é o processador para o qual o pacote foi compilado, normalmente i386. Proceda então a instalação como faria normalmente.

[5] Aqueles que contém o código que deve ser compilado para gerar o programa em si.

Referências Bibliográficas

[1] Ian Soboroff, *Mail Filtering with Procmail*, 22/10/1997.

[2] Peter Norton e Arthur Griffith, *Guia Completo do Linux*, Ed. Berkley, 2000;

[3] Bill Ball, *Usando Linux*, Ed. Campus, 1999;

[4] Robert Eckstein, David Collier-Brown, Peter Kelly, *Using Samba*, Ed. O'Reilly, Novembro de 1999.

[5] CIPISGA, *Guia Foca GNU/Linux*, `focalinux.cipsga.org.br`

[6] Conectiva S.A.,*Guia do Servidor Conectiva Linux*, `www.conectiva.com.br`

[7] Conectiva S.A.,*Guia Rápido de Instalação e Uso*, `www.conectiva.com.br`

[8] Conectiva S.A.,*Guia do Usuário do Conectiva Linux*, `www.conectiva.com.br`

[9] ZAGO,*Tuturial com todos os passos pra instalar e configurar o servidor Samba no CL10*, `http://www.zago.eti.br`

[10] André Assad, Clístenes Inácio, Eric Mello, Laudemiro Neto e Sérgio Souza, *Administração de Sistemas Linux*, NTI-UFPB, 2002.

[11] André Assad, Clístenes Inácio e Sérgio Souza, *Linux como Servidor de Intranet*, NTI-UFPB, 2002.

[12] Sérgio Souza, *Linux - Firewall e Roteador*, NTI-UFPB, 2003.

[13] Nicolai Langfeldt, *COMO FAZER DNS*, `br.tldp.org/projetos/howto`, 1999.