O sistema operacional NetBSD

um breve guia do usuário

Federico Lupi

O sistema operacional NetBSD: um breve guia do usuário

por Federico Lupi

Copyright © 1999, 2000, 2001, 2002 por Federico Lupi

Traduzido por Paulo Aukar (ptaaukar@terra.com.br)

Índice

Lista de Figuras

Lista de Exemplos

Objetivo deste guia

Este guia descreve a instalação e a configuração do sistema operacional NetBSD e se destina principalmente a pessoas provenientes de outros sistemas operacionais, na esperança de poder ajudar na resolução de alguns daqueles pequenos problemas que se encontram quando nos envolvemos com um sistema que nos é pouco familiar. As instruções contidas a seguir foram definidas a partir da versão para i386 do NetBSD.

Serão, no decorrer do texto, pressupostos alguns conhecimentos mínimos por parte do leitor: o que é um arquivo, um diretório, um programa; como se faz para editar um arquivo e assim por diante. Nas livrarias e bibliotecas podemos encontrar muitos textos que tratam dessas questões (em particular para os sistemas Unix). Estes livros têm freqüentemente um caráter geral e são um pré-requisito essencial para o uso de um sistema como o NetBSD. Em particular, quem instala um sistema BSD deverá encontrar-se (ou se desencontrar?), pelo menos nas etapas iniciais, com o editor vi: sem um pouco de documentação isso poderia ser um obstáculo insuperável. Em um segundo momento, naturalmente, cada um poderá instalar no sistema o editor (e os programas) que preferir.

O guia ainda não está completo. Alguns argumentos não estão terminados ou requerem revisão. Outros faltam integralmente. Procurarei levá-lo adiante tendo em conta a minha disponibilidade de tempo. Se você quiser ser avisado quando houver uma atualização ou se está interessado em temas específicos envie-me um e-mail. Contate-me também se estiver interessado em colaborar no desenvolvimento do guia, já traduzido para o inglês, para o francês e para o português do Brasil; ou em sua tradução para outras línguas.

`flupi@mclink.it`

Capítulo 1. Licença

Copyright (c) 1999, 2000, 2001 Federico Lupi. Todos os direitos reservados.

Redistribuição e uso na forma binária ou do código fonte, com ou sem modificação, são permitidos desde que fiquem asseguradas as seguintes condições:

1. Redistribuições do código fonte devem manter a notificação do copyright acima, esta lista de condições e a seguinte renúncia.
2. Redistribuições na forma de binários devem reproduzir o copyright acima, esta lista de condições e a seguinte renúncia na documentação e/ou materiais fornecidos com a distribuição.
3. Todo material de propaganda mencionando características ou uso deste software deve mostrar o seguinte reconhecimento: Este produto inclui software desenvolvido por Federico Lupi para o Projeto NetBSD.
4. O nome do autor não pode ser usado para endossar ou promover produtos derivados deste software sem uma prévia permissão escrita.

ESTE SOFTWARE É FORNECIDO PELO AUTOR "TAL QUAL" E QUALQUER GARANTIA EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS, GARANTIAS DE COMERCIALIZAÇÃO E ADEQUAÇÃO PARA UM PROPÓSITO PARTICULAR FICA EXCLUÍDA. EM NENHUMA CIRCUNSTÂNCIA DEVE O AUTOR SER RESPONSÁVEL POR QUALQUER DANO DIRETO, INDIRETO, INCIDENTAL, SINGULAR, EXEMPLAR OU CONSEQÜENTE (INCLUINDO, MAS NÃO SE LIMITANDO AO PROVIMENTO DE BENS E SERVIÇOS DE SUBSTITUIÇÃO; PERDA POR INUTILIZAÇÃO, PERDA DE DADOS OU LUCROS; OU INTERRUPÇÃO DE NEGÓCIOS), DE QUALQUER MODO CAUSADO E SEGUNDO QUALQUER TEORIA DE RESPONSABILIDADE, SEJA EM CONTRATO, RESPONSABILIDADE ESTRITA OU ILÍCITO (INCLUINDO NEGLIGÊNCIA OU QUALQUER OUTRA COISA) DECORRENDO DE QUALQUER MANEIRA DO USO DESTE SOFTWARE, AINDA QUE SOB A ADVERTÊNCIA DA POSSIBILIDADE DE TAL DANO.

Capítulo 2. O que é o NetBSD

2.1. As origens do NetBSD

A primeira versão do NetBSD (0.8) saiu em 1993, derivando do sistema operacional 4.3BSD Lite, uma versão do Unix desenvolvida na Universidade da Califórnia de Berkeley (BSD = Berkeley Software Distribution) e do sistema 386BSD, a primeira *conversão* para sistemas Intel 386. Em seqüência foram incorporadas as modificações provenientes do sistema 4.4BSD Lite, a última versão oficial do grupo de desenvolvedores de Berkeley antes de sua dissolução. O ramo BSD do Unix teve uma grande importância na história desse sistema operacional, para o qual contribuiu com variadas inovações que hoje fazem parte de todos os sistemas Unix (o editor vi, a shell C (csh), o job control, o fast file system de Berkeley, a integração do TCP/IP, apenas para mencionar algumas). Essa tradição de desenvolvimento e de pesquisa sobrevive hoje nos sistemas BSD (sejam livres ou comerciais) e, em particular, no NetBSD.

2.2. Características do NetBSD

O NetBSD funciona sobre uma vasta gama de plataformas de hardware e é muito portável. Com o NetBSD é fornecido o código-fonte do sistema operacional inteiro, para todas as plataformas suportadas. Sem alongar-me em excessivos detalhes, para os quais remeto o leitor para o site oficial do ProjetoNetBSD (http://www.netbsd.org/),as características fundamentais do sistema operacional são as seguintes:

- Extrema portabilidade (mais de 20 plataformas suportadas)
- Qualidade e correção do código
- Adesão aos padrões
- Pesquisa e inovação

As características que acabam de ser mencionadas também trazem vantagens indiretas. Por exemplo, quem trabalha em uma só plataforma poderia não estar interessado na portabilidade. Na realidade, todavia, a portabilidade está estritamente ligada à qualidade do código. Não seria possível suportar todas essas plataformas se o código não fosse bem escrito e bem organizado. A atenção para com o aspecto arquitetônico e qualitativo do sistema é recompensada com as grandes potencialidades do seu código e a qualidade de seus drivers. Portanto, interessa a todos os usuários.

Uma das características do sistema é a de não se contentar com implementações incompletas: “se deve ser feito, deve ser bem feito”.No panorama informático já há uma triste abundância de exemplos de programas e sistemas operacionais super-desenvolvidos e cheios de erros que entram em colapso sob o seu próprio peso.

Um elenco detalhado de características do NetBSD pode ser encontrado no endereço http://www.netbsd.org/Misc/features.

2.3. Plataformas suportadas

A versão 1.5.2, por exemplo, suporta plataformas como:

- Digital Alpha (64bit)
- Commodore Amiga, MacroSystem DraCo
- Acorn RiscPC/A7000, CATS, Digital Shark, EBSA-285, VLSI RC7500
- Atari TT030, Falcon, Hades
- Hewlett-Packard 9000/300 e 400
- PCs IBM da família i386 e clones
- Apple Macintosh
- Apple Power Macintosh
- Motorola MVME 68k SBCs
- NeXT 68k ’black’ hardware
- PC532
- Digital MIPS-based DECstations e DECsystems
- Sun SPARC
- Sun 3 e Sun3x
- Digital VAX
- Sharp X680x0

No site do NetBSD podem ser encontrados os detalhes técnicos para todas as plataformas precedentes.

2.4. A quem se destina o NetBSD

De acordo com o que se apresenta no site do NetBSD, os seus destinatários são os profissionais, os amadores e os pesquisadores que querem um sistema estável que privilegie a qualidade. Também quem deseja aprender a usar Unix encontrará no NetBSD a plataforma ideal, sobretudo pela sua aderência aos padrões (um dos objetivos do projeto). Enfim, quem tem necessidade de uma plataforma Unix disponível para uma grande variedade de máquinas, não pode encontrar melhor aliado que o NetBSD.

Uma outra característica interessante é que com o NetBSD podem-se utilizar sistemas de hardware considerados obsoletos para a maior parte dos sistemas operacionais. Isso torna-o uma ótima plataforma para a aprendizagem do Unix. Pode-se dizer que “não há necessidade de comprar um novo hardware para ter a sua versão de Unix em funcionamento. Você pode reutilizar aquele velho MacIIcx que está encostado no fundo do armário.”

2.5. Aplicativos

Quem instala o NetBSD tem à disposição abundância de aplicativos à sua escolha. Além do conjunto de aplicativos padrões de produtividade pessoal e de desenvolvimento (C/C++) de todo sistema Unix, está disponível um grande número de pacotes adicionais (mais de mil, e o número está em constante aumento), instaláveis através de cômodo sistema de gerenciamento de pacotes. Em geral, todos os principais pacotes para Unix estão disponíveis para o NetBSD. Além disso, o NetBSD é capaz de emular o Linux e executar muitos programas nativos desse sistema operacional (Netscape, Acrobat Reader... e até mesmo Doom e Quake!). A emulação do Linux é muito cuidada e considerada de grande importância: quase todos os usuários de NetBSD a utilizam para alguns programas.

Além de executar programas Linux, o NetBSD é capaz de emular também outros sistemas, em particular o FreeBSD, o BSDI e ainda outros.

2.6. A filosofia da distribuição NetBSD

Ao contrário de muitos sistemas contemporâneos, a distribuição NetBSD não tem nada de mastodôntica, já que se fundamenta no conceito de produzir um sistema de base estável,completo e eficiente, mas sem redundâncias. Ao final da instalação deparamo-nos com um sistema funcionando, mas que ainda não tem uma série de programas utilitários (por exemplo, um browser para a Internet). Todos estes aplicativos não fazem parte do núcleo básico. Cabe ao usuário a decisão de se quer o Netscape ou o Lynx; qual versão de Perl lhe é útil e assim por diante. Uma outra vantagem é que o sistema básico funciona de modo auto-suficiente, sem ter necessidade desses pacotes "adicionais". Por exemplo, o sistema não depende do Perl para funcionar. Se decidimos instalar uma nova versão do Perl, não se expõe a risco qualquer componente padrão do sistema. Quando se instala o NetBSD, portanto, não se encontram seleções pré-estabelecidas de centenas de pacotes que vão entupir o disco rígido e que acabarão ficando, na maioria dos casos, sem uso. O usuário BSD deve estar sempre consciente daquilo que instala e por que. O processo de instalação dos programas, entre outras coisas, foi muito simplificado pelo sistema de gerenciamento de pacotes.

Mesmo quando se começa a conhecer um pouco o sistema e a se orientar bem em seu interior, uma das coisas que continuam a nos admirar é a extrema coerência e logicidade de seu conjunto. Nada é casual ou gratuito e tudo parece pensado com grande atenção aos detalhes. Pensando bem, somando tudo, considero que esta seja a característica mais interessante do NetBSD. Podemos passar dias discutindo sobre os méritos relativos dos vários sistema operacionais (e alguns até conseguem tempo para fazê-lo), mas somente quando se prova e se chega a conhecer um sistema é que se pode fazer uma escolha motivada. Estou convencido, por tê-lo observado tantas vezes nas mailing lists, que quem experimenta o NetBSD não pode deixar de ser conquistado pelo equilíbrio perfeito entre complexidade e eficácia realizado por este sistema. O NetBSD consegue sempre realizar as coisas mais complexas da maneira mais simples. É uma atitude mental contagiosa. Depois de pouco tempo fica difícil tolerar as inúteis complicações (freqüentemente travestidas de "melhoramentos" ou de "cômodas interfaces gráficas") de outros sistemas.

2.7. Como obter o NetBSD

Até hoje o NetBSD não tem um revendedor oficial, ainda que existam vários pontos de distribuição. Para um conjunto atualizado é bom consultar a página (http://www.netbsd.org/Sites/cdroms.html) correspondente no site do Projeto. Naturalmente, é sempre possível fazer o download do sistema de um dos vários "mirror sites" na Internet.

Se você não pode recorrer a nenhuma destas opções, entre em contato comigo e verei se lhe posso produzir um CD com uma cópia. Disponho das versões 1.3.3, 1.4, 1.4.1, .1.4.2, 1.4.3, 1.5, 1.5.1 e 1.5.2, a última no momento para i386.

Capítulo 3. Instalação

3.1. Documentação

A documentação do NetBSD está em sua maior parte no formato das páginas de manual (man) e compreende sobretudo documentação técnica de referência, muito válida e completa mas não particularmente adaptada a objetivos introdutórios. Isso sem dizer que é impossível ler uma página man antes de se ter instalado o sistema...

> **Nota:** depois da instalação também podem ser encontrados vários guias de Unix/BSD no diretório `/usr/share/doc`. Algumas têm caráter histórico, enquanto outras são permanentemente atualizadas. Estão divididas em três grupos: *psd* (UNIX Programmer's Supplementary Documents), *smm* (UNIX System Manager's Manual) e *usd* (UNIX User's Supplementary Documents). Infelizmente alguns diretórios estão vazios, provavelmente por motivos relacionados à licença dos conteúdos. O texto pode ser lido na tela com o nroff. Por exemplo:
>
> ```
> $ cd /usr/share/doc/smm/09.sendmail
> $ nroff -me 09.sendmail/intro.me | more
> ```
>
> Usando os makefiles pode-se até mesmo gerar output em formato Postscript, pronto para ser impresso.

É, todavia, inegável uma certa carência de guias para usuários e dos assim chamados HOWTO e, portanto, é necessário desfrutar ao máximo dos existentes. Uma série de documentos encontra-se nas releases do NetBSD, em formato texto. Informações adicionais podem ser encontradas no site do NetBSD (www.netbsd.org (http://www.netbsd.org) e mirrors), em particular as Frequently Asked Questions (FAQ).

> **Documentação original:** o site do NetBSD contém várias páginas de documentação e de HOWTO, tanto genéricos como específicos por plataforma. Trata-se de informações úteis e expostas de maneira simples (por exemplo, como ver uma partição Windows a partir do NetBSD ou como inicializar o NetBSD com o seletor de boot do Windows NT, etc.). No presente guia por ora não são duplicadas as informações já constantes do site do NetBSD.

Mais ou menos em todas as versões do NetBSD encontram-se os seguintes arquivos:

INSTALL

Notas para a instalação do NetBSD/i386. Para ler (e reler) com o máximo de atenção. Contém uma descrição do sistema, o elenco de dispositivos de hardware suportados e as instruções para a instalação.

README.first

Não é propriamente essencial para a instalação, mas convém lê-lo.

release.man

Descreve a estrutura da versão do NetBSD em formato man (pré-formatado: pode ser lido com um editor de textos comum).

No site do NetBSD podem ser encontrados vários guias online, muito úteis e muito sintéticos.

NetBSD FAQ

Contém algumas informações de caráter geral (por exemplo, como recompilar o kernel) e indicadores para outras FAQ.

NetBSD/i386 FAQ

Temas específicos do NetBSD/i386.

Gerenciamento básico de redes em NetBSD

Guia para a configuração de redes e do uso do PPP.

3.2. A estrutura de uma distribuição NetBSD

A *disposição* dos arquivos em uma distribuição NetBSD é descrito no fim do arquivo `INSTALL`. A título de recordação, os arquivos binários para instalação encontram-se (para a plataforma *i386*) no subdiretório `i386/binary/sets`. Os códigos fonte correspondentes podem ser encontrados no diretório `source/sets`. No diretório `source/patches` estão localizados os *patches* para os códigos fonte, a fim de que se corrijam problemas eventuais (comumente problemas de segurança descobertos depois de liberada uma nova versão do sistema).

3.3. Instalação

A primeira coisa a fazer para instalar o NetBSD é ler as notas de instalação da versão desejada, que se encontram no arquivo INSTALL. O procedimento de instalação é ali descrito em detalhe. Nesse ponto, pode-se decidir o meio a ser utilizado para a instalação, dentre aqueles que são suportados pela sua máquina:

- ftp
- nfs
- CD-ROM
- disquete
- partição não montada
- diretório local

3.3.1. Teclado

O programa de instalação sysinst não permite mudar a disposição do teclado e, portanto, a instalação deve ser efetuada com a utilização do teclado na disposição americana (us). Na realidade não se trata de um grande problema, já que para instalar o NetBSD com CDROM necessita-se simplesmente escolher entre uma série de opções de tipo alfa-numérico e, portanto, as teclas correspondem. Se a instalação for feita por ftp o problema pode ser mais enjoado. Penso que as próximas versões do programa de instalação possam permitir modificar a disposição do teclado. Por enquanto, todavia, pode-se orientar pela seguinte tabela:

US	IT	DE	FR
-	’	ß	)
/	-	-	!
=	ì	’	-
:	ç	Ö	M
;	ò	ö	m
#	£	§	3
"	°	Ä	%
*	(	(	8
(	)	)	9
)	=	=	0
’	à	ä	ù
‘	\	^	@
\	ù	#	‘

Ao fim da instalação, uma das primeiras coisas a fazer será mudar a disposição do teclado.

3.3.2. Geometrias

Em alguns pontos do programa de instalação se faz referência à geometria do disco rígido e, em particular a

- geometria real
- geometria da BIOS

A geometria real é identificada pelo sistema e se refere à geometria efetiva do disco. A geometria da BIOS é a geometria com que a BIOS vê o disco e poderia ser diferente da geometria real (por exemplo, poderia ser remapeada com LBA). O disco rígido que será usado no exemplo de instalação tem a seguinte geometria:

```
real:  6232 cyl,   16 heads,  63 sec
BIOS:   779 cyl,  128 heads,  63 sec   (LBA)
```

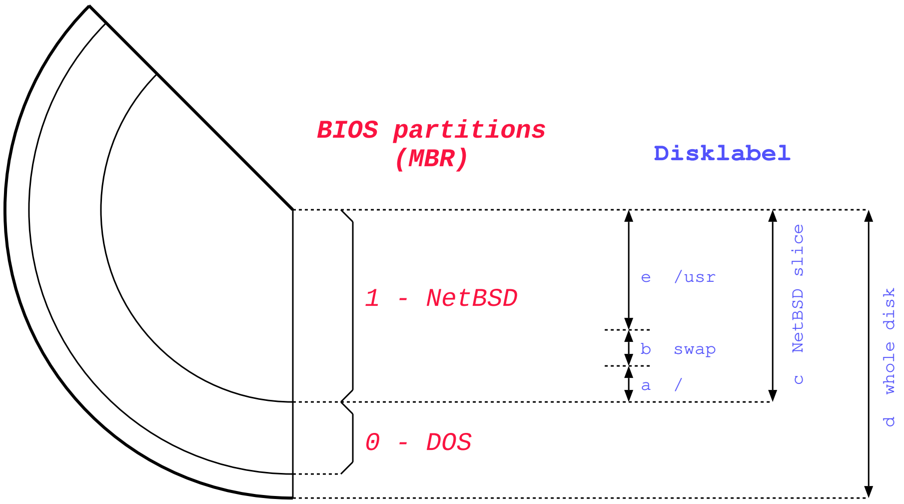
Neste caso a BIOS usa a modalidade LBA para remapear a geometria do disco, diminuindo o número de cilíndros e aumentando o de trilhas (mas o produto mantém-se o mesmo: 6232*16 = 779*128 = 99712). Lembrando que um setor é composto de 512 bytes, a dimensão do disco é: 6232*16*63*512 = 3 GB. O NetBSD não tem nenhuma necessidade de remaperar a geometria do disco, e é também possível mudar a geometria manualmente se a que foi revelada pelo sistema não fosse correta.

3.3.3. Partições

A terminologia usada pelo NetBSD no que diz respeito às partições é diferente daquela dos sistemas DOS/Windows. NetBSD instala-se em uma das partições primárias do disco rígido (partições BIOS, aquelas definidas nas tabelas das partições do disco). No interior da própria partição BIOS o NetBSD define suas próprias partições, que poderemos chamar partições BSD, escrevendo um *disklabel* ou grade divisória interna. As partições BSD são visíveis apenas pelo NetBSD e são identificadas por meio de letras (de "a" em diante. Por exemplo, wd0a indica a partição "a" do primeiro disco IDE (wd0)). No exemplo da Figura 3-1 há duas partições primárias. A primeira é ocupada pelo DOS e a segunda pelo NetBSD, que a subdivide, por sua vez, em partições BSD, escrevendo um disklabel.

> **Nota:** na realidade não é de todo exato dizer que o disklabel subdivide a partição BIOS do NetBSD em subpartições. Como se pode ver pela figura as partições "c" e "d" não são subpartições. O disklabel é, para todos os efeitos, a tabela de partições própria do NetBSD: o kernel do NetBSD não consulta a tabela da BIOS mas o disklabel.

Figura 3-1. Partições



> **Nota:** a disposição descrita na Figura 3-1 é típica dos sistemas i386 no que se refere ao significado das partições “c” e “d”.

> **Nota:** se o NetBSD não é o único sistema operacional presente no disco rígido (como no exemplo precedente), é necessário instalar um *gerenciador de inicialização (boot manager)*, isto é, um programa que na fase da inicialização do computador permite a escolha do sistema operacional que se quer inicializar. O programa de instalação do NetBSD pode instalar automaticamente um gerenciador de boot se o usuário assim o quiser. Se o outro sistema operacional presente no disco rígido é Windows NT, é possível a utilização do gerenciador de inicialização do NT, acrescentando o NetBSD ao menu. O procedimento (muito simples) é descrito no site do NetBSD.

3.3.4. Espaço necessário

O espaço a ser destinado ao NetBSD depende claramente do uso que se quer fazer dele (por exemplo, servidor ou estação de trabalho) e dos programas que se quer instalar. A título de exemplo, considere-se um sistema usado como desktop com um disco rígido de 420 MB, no qual foram instalados o X, as fontes do kernel e alguns programas. À partição de memória adicional (swap) foram dedicados 32 MB. **df** mostra o que se segue:

```
Filesystem  1K-blocks     Used    Avail Capacity  Mounted on
/dev/wd1a       31887    16848    13444    56%    /
/dev/wd1e      363507   173202   172129    50%    /usr
```

Como se vê, cerca de 180 MB ficaram livres no sistema.

3.3.5. Tentar de novo

As primeiras instalações, em geral, quase nunca dão totalmente certo. Isto vale para todos os sistemas operacionais e o NetBSD não é exceção. O importante é não se desencorajar e provar de novo. Aquilo que de início parece incompreensível torna-se logo claro devido às provas feitas e à experiência adquirida (assim como às numerosas releituras do arquivo INSTALL).

Durante as primeiras instalações convém aceitar todos os valores definidos por default que são propostos pelo programa de instalação, evitando aventurar-se em modificações do disklabel ou similares. A única decisão real a tomar é quanto ao espaço a dedicar à partição NetBSD com o fdisk.

3.4. Exemplo de instalação

Na parte restante deste capítulo vejamos em detalhe um exemplo real de instalação para um dos casos mais comuns: a instalação por CDROM. Os outros tipos de instalação não diferem nos conceitos, mas somente no modo em que são encontrados os arquivos a instalar. Note-se que alguns detalhes da instalação podem apresentar diferenças segundo tal ou qual versão do NetBSD que se instala. O exemplo refere-se à versão 1.5.

No exemplo seguinte sempre serão tomados os caminhos mais "difíceis":

- Tabela das partições sem espaço disponível: é necessário eliminar uma ou mais partições para abrir espaço para o NetBSD.
- Particionamento usando setores ao invés de megabytes como unidade de medida.
- Modificação manual do disklabel criado pelo programa de instalação (também aqui usando os setores como unidade de medida).
- Instalação de tipo "custom".

Estas escolhas fazem a instalação parecer muito mais trabalhosa, embora na realidade não o seja. Por outro lado, não teria muito sentido escrever um guia que ilustre apenas as situações mais fáceis. Tenha-se em todo caso presente que escolhendo-se o uso dos default da instalação as operações são *muito* mais simples.

3.4.1. Preparando a instalação

Antes de proceder à instalação é bom planejar cuidadosamente as operações a realizar. Em primeiro lugar é necessário ler atentamente o já citado INSTALL, que descreve atentamente os passos necessários à instalação, controlando particularmente o elenco das compatibilidades de hardware. Depois é necessário liberar espaço suficiente para acolher o NetBSD no disco rígido. Se o NetBSD compartilhar o disco com outros sistemas operacionais, será necessário criar uma nova partição (é nisso que "pensa" o programa de instalação), eventualmente reduzindo uma já existente (é nisso que deve pensar o usuário). Para esse objetivo pode-se usar um produto comercial (por exemplo, o Partition Magic) ou um software livre como o FIPS ou o pfdisk (ambos presentes na distribuição do NetBSD).

A instalação é dividida logicamente em duas partes. Na primeira parte cria-se uma partição para o NetBSD e se cria o disklabel para a partição. Na segunda parte decide-se que coisa se quer instalar e se extraem os arquivos na partição recém criada. A primeira parte é independente do meio a ser adotado para a instalação (CDROM, FTP, NFS, etc.). Ao término da primeira parte é solicitada a confirmação das operações efetuadas. Até aquele momento ainda não foi gravado *nada* no disco rígido. Abortando-se a instalação tudo permanece como era antes de começar.

3.4.2. Criando o disquete de instalação

> **Nota:** quando se dispõe de um CD-ROM inicializável do NetBSD não é necessário criar o disquete de instalação. Bastará inicializar o computador depois de tê-lo configurado para dar o boot a partir do drive do CD (esta opção não é disponível em todos os PCs).

A primeira operação a realizar é a criação do disquete de instalação. Trata-se de copiar a imagem do disquete do CD para o floppy. Se esta operação é efetuada em ambiente DOS, usar-se-á o programa rawrite.exe, que se encontra no diretório `i386/installation/misc`. O arquivo a ser copiado no disquete é `i386/installation/floppy/boot.fs`.

> Certificar-se sempre de que os disquetes utilizados para os discos de instalação não apresentem defeitos. Este simples cuidado pode evitar muitos dissabores na fase de instalação.

1. Formatar o floppy em ambiente DOS.
2. Deslocar-se para o diretório \NETBSD-1.3.2\I386\INSTALLATION\FLOPPY del CD-ROM.
3. Executar **..\MISC\RAWRITE** indicando `BOOT.FS` como *Source file name* e A: como *Destination drive*.

Se a criação do disquete for efetuada em ambiente Unix, usar-se-á o o comando **dd**. Por exemplo:

```
# cd i386/installation/floppy
```

```
# dd if=boot.fs of=/dev/fd0a bs=36b
```

dd copia blocos de 512 bytes. A opção `bs=36b` indica a cópia de 36 blocos por vez, para acelerar a operação.

> **Nota:** um disquete de 1440k contém 1474560 bytes e é composto de 80 cilíndros, 2 trilhas, 18 setores de 512 bytes, isto é, de 80*2*18=2880 blocos de 512 bytes. Portanto, com a opção `bs=36b` copia-se um cilíndro por vez.

3.4.3. Últimas operações preparatórias

Agora tudo está pronto para a instalação propriamente dita mas, antes de começar, é bom obter algumas informações sobre o hardware do PC.

A coisa mais importante a se descobrir é o tipo de disco rígido instalado no sistema (IDE, SCSI, ...) e a sua geometria. Estas informações podem ser extraídas do manual do disco, de muitos programas de diagnóstico para DOS ou da própria BIOS, na inicialização do computador. Em alguns discos a geometria está impressa em uma etiqueta colada nele próprio. Uma outra possibilidade válida é entrar no site do fabricante na Internet e procurar aí as especificações técnicas do disco rígido. Se a intenção for instalar via ftp ou via NFS, é bom que se anote também os dados da placa de rede. Uma das mais freqüentes causas de problemas é que o IRQ da placa de rede não corresponde àquele registrado no kernel. Vejamos, a título de exemplo, os registros para uma categoria de placas de rede muito difundidas, as NE2000 e compatíveis de barramento ISA. O kernel é pré-configurado para localizar placas com estas duas configurações:

```
ne0     at isa? port 0x280 irq 9        # NE[12]000 ethernet cards
ne1     at isa? port 0x300 irq 10
```

Portanto, se as configurações de uma NE2000 instalada no sistema não coincidem com *ne0* ou com *ne1*, o kernel de instalação não a identificará (depois da instalação, naturalmente, é possível compilar um kernel personalizado capaz de reconhecer até mesmo outras configurações).

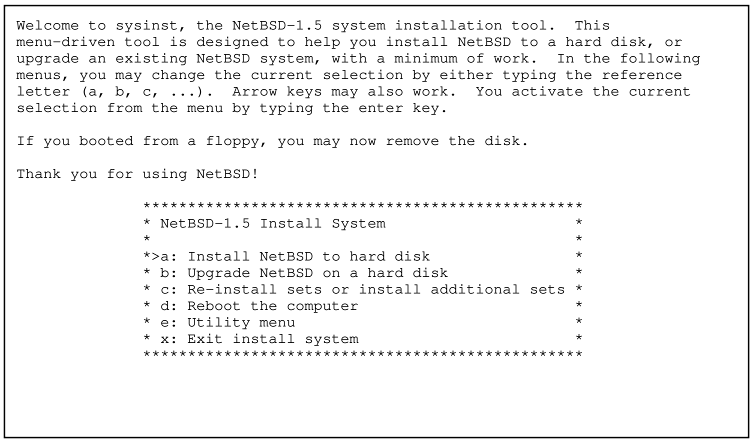
Já que estamos nisso, convém descobrir também algumas informações sobre o resto do sistema. Por exemplo, o tipo de placa de áudio, o número de portas seriais e paralelas, etc. Todas estas informações podem ser úteis em um segundo momento, mesmo que habitualmente não sirvam na fase de instalação. É muito importante individuar eventuais configurações não usuais dos periféricos. Para fazer isso, confrontar as configurações do seu sistema (IRQ, portas de I/O, etc.) com as configurações default reportadas no arquivo INSTALL.

> **Nota:** pode-se proceder à instalação mesmo sem saber a geometria do disco. A informação sobre a geometria serve unicamente para controle. O programa de instalação está em condição de determiná-la automaticamente.

3.4.4. Iniciemos a instalação

Inserir o disquete de instalação recém criado no drive A: e inicializar o computador (ou fazer o boot a partir do CD-ROM). Com o disquete se inicializa o kernel comprimido `netbsd.gz`. Durante o boot do disquete de instalação recém criado (para certas versões do NetBSD os disquetes poderiam ser dois), quando da inicialização do kernel são mostradas as informações sobre os dispositivos encontrados e configurados. É provável que ocorram muitas indicações de dispositivos não encontrados. Isso é devido ao fato de que o kernel presente no disquete compreende o suporte para uma grande quantidade de periféricos. É natural que nem todos estejam presentes no sistema.

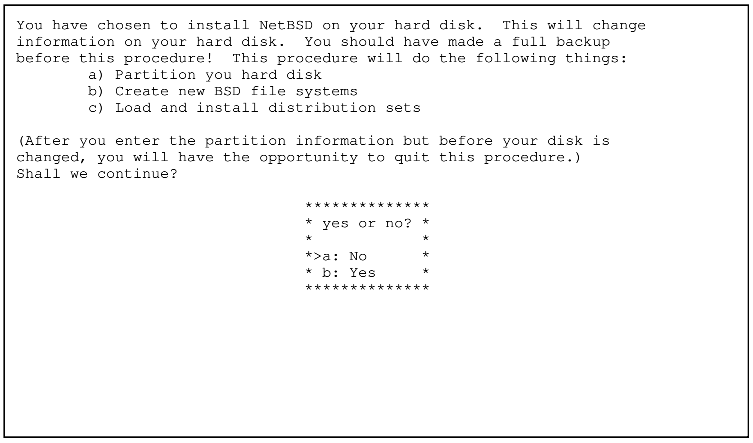
Figura 3-2. Início da instalação



```
Welcome to sysinst, the NetBSD-1.5 system installation tool.  This
menu-driven tool is designed to help you install NetBSD to a hard disk, or
upgrade an existing NetBSD system, with a minimum of work.  In the following
menus, you may change the current selection by either typing the reference
letter (a, b, c, ...).  Arrow keys may also work.  You activate the current
selection from the menu by typing the enter key.

If you booted from a floppy, you may now remove the disk.

Thank you for using NetBSD!

              *************************************************
              * NetBSD-1.5 Install System                     *
              *                                               *
              *>a: Install NetBSD to hard disk                *
              * b: Upgrade NetBSD on a hard disk              *
              * c: Re-install sets or install additional sets *
              * d: Reboot the computer                        *
              * e: Utility menu                               *
              * x: Exit install system                        *
              *************************************************
```

Terminada a inicialização, visualizamos na tela o menu principal do programa de instalação, mostrado na Figura 3-2. Não se pode deixar enganar pelo aspecto espartano do sysinst. Trata-se de um programa muito potente e flexível. Daqui em diante é só seguirmos as instruções mostradas no vídeo, tendo presente as explicações do arquivo INSTALL. As telas do sysinst estão todas estruturadas do mesmo modo: na parte superior se visualiza uma pequena explicação da operação corrente ou uma mensagem de ajuda; na parte central é mostrada a situação atual tal como a identifica o NetBSD (não presente nessa tela de amostra); e na parte inferior aparece o menu com as escolhas possíveis. Escolhendo a opção de instalação “a” chega-se à tela de confirmação, mostrada na Figura 3-3.

Figura 3-3. Confirmando a instalação

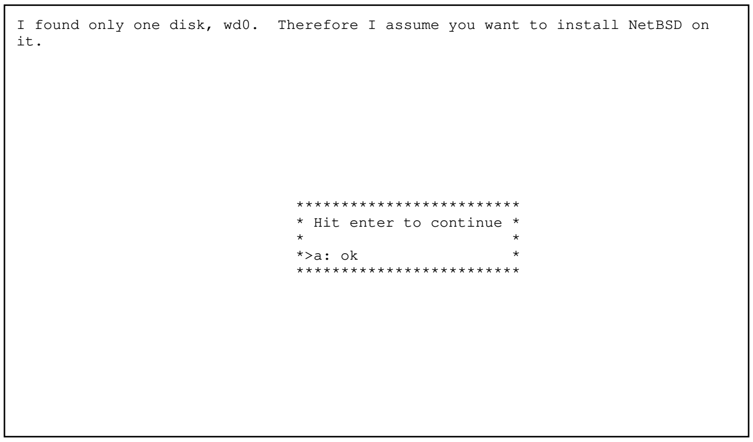


```
You have chosen to install NetBSD on your hard disk.  This will change
information on your hard disk.  You should have made a full backup
before this procedure!  This procedure will do the following things:
        a) Partition you hard disk
        b) Create new BSD file systems
        c) Load and install distribution sets

(After you enter the partition information but before your disk is
changed, you will have the opportunity to quit this procedure.)
Shall we continue?

                                ***************
                                * yes or no? *
                                *            *
                                *>a: No      *
                                * b: Yes     *
                                ***************
```

Depois de ter decidido prosseguir com a opção "b", chega o momento de indicar sobre qual disco rígido se quer instalar o NetBSD. Se no sistema de hardware estão presentes mais de um disco rígido é mostrada uma relação para fazermos a escolha. Em nosso caso há no computador apenas um disco rígido IDE e, portanto, o programa de instalação não propõe escolhas, limitando-se a mostrar uma tela informativa que pode ser vista na Figura 3-4.

> **Nota:** as informações mostradas neste tela mudam de acordo com o tipo e com o número de discos rígidos instalados no micro.

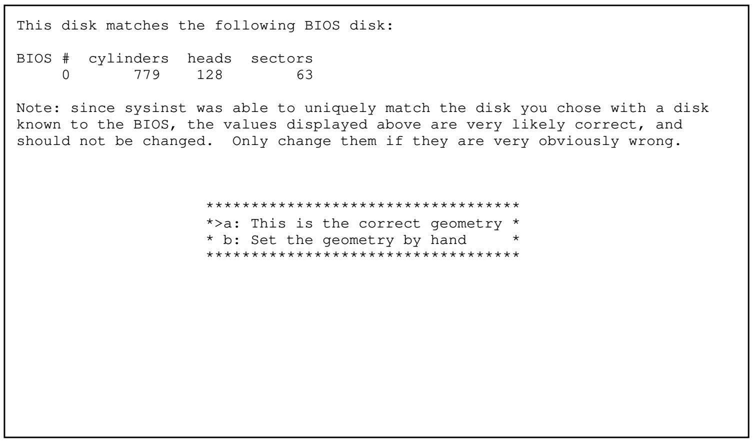
Figura 3-4. Escolha do disco rígido



```
I found only one disk, wd0.  Therefore I assume you want to install NetBSD on
it.









                               *************************
                               * Hit enter to continue *
                               *                       *
                               *>a: ok                 *
                               *************************
```

Subseqüentemente (Figura 3-5) pode-se visualizar a geometria registrada pela BIOS para o disco selecionado na tela precedente. Pode-se confirmar a geometria mostrada ou modificá-la manualmente. Esta segunda opção serve no caso em que o sistema não tenha reconhecido corretamente a geometria.

Figura 3-5. Confirmação da geometria



```
This disk matches the following BIOS disk:

BIOS #  cylinders  heads  sectors
     0       779    128        63

Note: since sysinst was able to uniquely match the disk you chose with a disk
known to the BIOS, the values displayed above are very likely correct, and
should not be changed.  Only change them if they are very obviously wrong.



                     **********************************
                     *>a: This is the correct geometry *
                     * b: Set the geometry by hand     *
                     **********************************
```

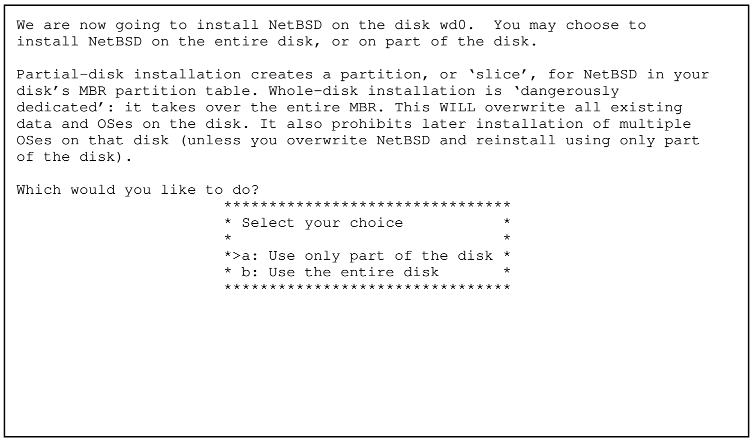
3.4.5. Partições

Chegou o primeiro momento importante da instalação: o particionamento do disco rígido. A primeira coisa a fazer é indicar se o NetBSD ocupará o disco inteiro (escolha temerária). Mesmo escolhendo a primeira opção, entretanto, pode-se criar uma partição que cobre o disco inteiro (Figura 3-6). A diferença é que na primeira forma de instalação a tabela das partições é mantida em uma modalidade compatível com outros sistemas operacionais.

Em nosso exemplo usaremos um disco com a seguinte geometria real, correspondente à geometria da BIOS mostrada na Figura 3-5.

```
6232 cyl, 16 heads, 63 sec  (6232 x 16 x 63 = 6281856 setores totais)
1 setor = 512 bytes
1 trilha = 63 setores = 63 * 512 bytes = 32 K
1 cilíndro = 16 * 63 * 512 bytes = 504 K
```

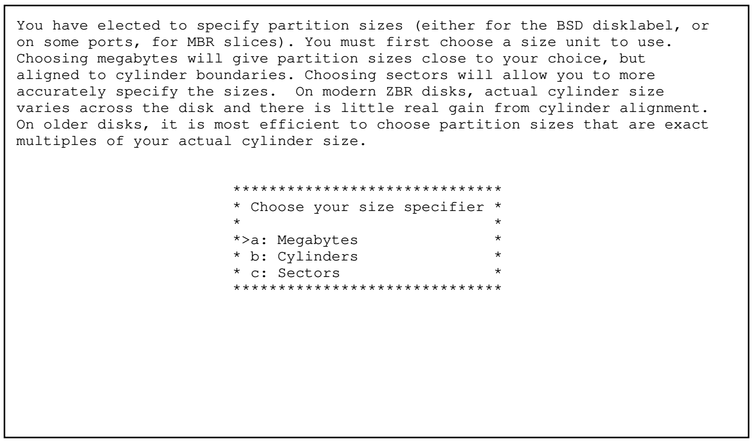
Figura 3-6. Escolhendo o esquema de particionamento



```
We are now going to install NetBSD on the disk wd0.  You may choose to
install NetBSD on the entire disk, or on part of the disk.

Partial-disk installation creates a partition, or ‘slice’, for NetBSD in your
disk’s MBR partition table. Whole-disk installation is ‘dangerously
dedicated’: it takes over the entire MBR. This WILL overwrite all existing
data and OSes on the disk. It also prohibits later installation of multiple
OSes on that disk (unless you overwrite NetBSD and reinstall using only part
of the disk).

Which would you like to do?
                       ********************************
                       * Select your choice           *
                       *                              *
                       *>a: Use only part of the disk *
                       * b: Use the entire disk       *
                       ********************************
```

O passo seguinte, ilustrado na Figura 3-7 é a indicação das unidades de medida a serem utilizadas para particionar o disco. Os setores são aqueles que permitem a maior precisão, lembrando que sobre discos mais antigos convém alinhar as partições em múltiplos da dimensão do cilíndro. Os megabytes são mais simples de utilizar porque não exigem cálculos manuais, sendo mais “intuitivos”.

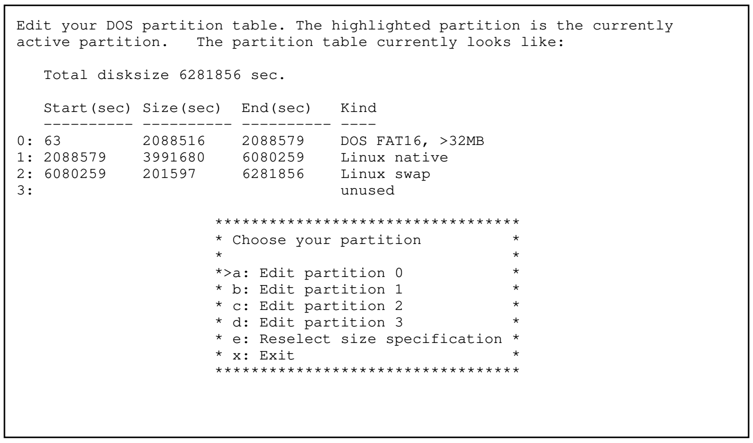
Figura 3-7. Escolha da unidade de medida



```
You have elected to specify partition sizes (either for the BSD disklabel, or
on some ports, for MBR slices). You must first choose a size unit to use.
Choosing megabytes will give partition sizes close to your choice, but
aligned to cylinder boundaries. Choosing sectors will allow you to more
accurately specify the sizes.  On modern ZBR disks, actual cylinder size
varies across the disk and there is little real gain from cylinder alignment.
On older disks, it is most efficient to choose partition sizes that are exact
multiples of your actual cylinder size.


                        ******************************
                        * Choose your size specifier *
                        *                            *
                        *>a: Megabytes               *
                        * b: Cylinders               *
                        * c: Sectors                 *
                        ******************************
```

Na seqüência utilizaremos setores porque são mais úteis para fins demonstrativos. Escolhendo a opção “c”, chega-se à tela de interface do fdisk.

Figura 3-8. fdisk



```
Edit your DOS partition table. The highlighted partition is the currently
active partition.   The partition table currently looks like:

   Total disksize 6281856 sec.

   Start(sec) Size(sec)  End(sec)   Kind
   ---------- ---------- ---------- ----
0: 63         2088516    2088579    DOS FAT16, >32MB
1: 2088579    3991680    6080259    Linux native
2: 6080259    201597     6281856    Linux swap
3:                                  unused

                      **********************************
                      * Choose your partition          *
                      *                                *
                      *>a: Edit partition 0            *
                      * b: Edit partition 1            *
                      * c: Edit partition 2            *
                      * d: Edit partition 3            *
                      * e: Reselect size specification *
                      * x: Exit                        *
                      **********************************
```

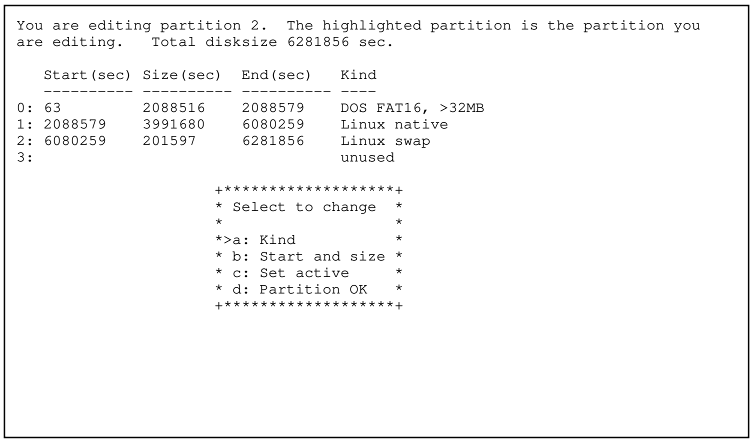
A Figura 3-8 mostra a situação do disco rígido antes da instalação do NetBSD. Há quatro partições primárias. Uma para DOS/Windows, duas para Linux e uma não utilizada. Mesmo havendo uma partição livre, não há espaço livre no disco. A coluna End(sec) da partição 2 mostra que todos os 6281856 setores do disco estão ocupados. A

> **Nota:** na tela do fdisk vale sempre a fórmula
>
> ```
> Start(sec) + Size(sec) = End(sec)
> ```

além disso, a coluna End(sec) de uma partição corresponde à coluna Start(sec) da partição seguinte. Isto não é muito intituitivo porque o setor visualizado na coluna End(sec) na realidade não pertence à partição, mas é o primeiro setor da partição seguinte. No disklabel, ao contrário, é utilizada uma convenção diferente (e mais lógica).

Para abrir espaço é necessário sacrificar (ainda que com dor no coração) as duas partições do Debian GNU/Linux, começando pela última (opção “c”). Será visualizada uma tela que pode ser usada para modificar os dados de uma partição e a Figura 3-9 mostra-nos os dados correntes da partição 2.

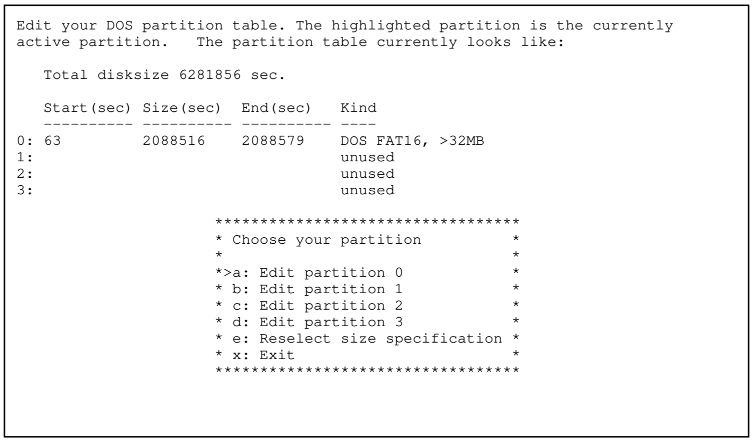
Figura 3-9. Deletando uma partição



```
You are editing partition 2.  The highlighted partition is the partition you
are editing.   Total disksize 6281856 sec.

   Start(sec) Size(sec)  End(sec)   Kind
   ---------- ---------- ---------- ----
0: 63         2088516    2088579    DOS FAT16, >32MB
1: 2088579    3991680    6080259    Linux native
2: 6080259    201597     6281856    Linux swap
3:                                  unused

                      +*******************+
                      * Select to change  *
                      *                   *
                      *>a: Kind           *
                      * b: Start and size *
                      * c: Set active     *
                      * d: Partition OK   *
                      +*******************+
```

Para deletar uma partição seleciona-se primeiro o tipo *unused* com a opção "a: Kind" e depois se escolhe a opção “b”, deixando-se vazios os campos “Start” e “Size” (tecle Enter deixando os campos em branco). Enfim, confirma-se tudo com a opção “d”. Quando se volta à tela principal do fdisk, a partição se mostra livre. Do mesmo modo são deletadas as partições 2 e 1, até que fique apenas a partição 0 (Figura 3-10).

Figura 3-10. Partições deletadas



```
Edit your DOS partition table. The highlighted partition is the currently
active partition.   The partition table currently looks like:

   Total disksize 6281856 sec.

   Start(sec) Size(sec)  End(sec)   Kind
   ---------- ---------- ---------- ----
0: 63         2088516    2088579    DOS FAT16, >32MB
1:                                  unused
2:                                  unused
3:                                  unused

                      **********************************
                      * Choose your partition          *
                      *                                *
                      *>a: Edit partition 0            *
                      * b: Edit partition 1            *
                      * c: Edit partition 2            *
                      * d: Edit partition 3            *
                      * e: Reselect size specification *
                      * x: Exit                        *
                      **********************************
```

Permanece somente a partição DOS/Windows de 2088516 setores, ou seja, 1029 MB (cerca de 1 GB). O espaço livre é dado pela diferença entre o número total de setores, já calculado acima, e o ponto final da partição DOS (coluna “End”).

```
6281856 - 2088579 = 4193277 setores = 2047 MB livres no disco
```

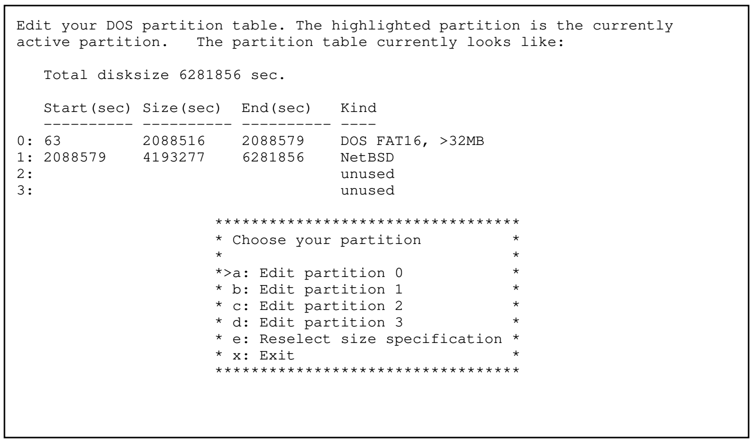
> **Nota:** note-se que a partição DOS inicia-se no setor 63 e não no setor 0, como se poderia esperar. De fato, a primeira trilha (isto é, os primeiros 63 setores) fica reservada. No cilíndro 0, trilha 0, setor 1 do disco rígido encontra-se o *Master Boot Record* (MBR). Quando da inicialização do computador, a BIOS carrega na memória o MBR do disco rígido, determina qual é a partição ativa e carrega na memória o boot sector daquela partição, à qual cede o controle. O boot sector da partição provê a inicialização do sistema operacional correspondente.

Agora procede-se à criação de uma nova partição para o NetBSD, que deve ter início no fim da partição DOS, escolhendo a opção “b”. Para criar uma nova partição é necessário especificar

- o tipo da nova partição
- o setor inicial da nova partição
- a dimensão em setores da nova partição

Em primeiro lugar indica-se que a nova partição dever ser do tipo “NetBSD” (opção "a: Kind") e depois se inserem os dados recém calculados: início = 2088579 e tamanho = 4193277 (utilizando a opção “b”). No fim atenta-se para que tudo esteja correto e se confirma a criação com a opção "d" que remete ao menu principal do fdisk. O resultado final é visualizado na Figura 3-11 que mostra a tabela completa das partições. Nesse ponto pode-se selecionar a opção "x" para passar para o menu seguinte.

Figura 3-11. Partição criada



```
Edit your DOS partition table. The highlighted partition is the currently
active partition.   The partition table currently looks like:

   Total disksize 6281856 sec.

   Start(sec) Size(sec)  End(sec)   Kind
   ---------- ---------- ---------- ----
0: 63         2088516    2088579    DOS FAT16, >32MB
1: 2088579    4193277    6281856    NetBSD
2:                                  unused
3:                                  unused

                      **********************************
                      * Choose your partition          *
                      *                                *
                      *>a: Edit partition 0            *
                      * b: Edit partition 1            *
                      * c: Edit partition 2            *
                      * d: Edit partition 3            *
                      * e: Reselect size specification *
                      * x: Exit                        *
                      **********************************
```

> **Nota:** a versão do sysinst do NetBSD 1.5 verifica também o início e o fim das partições não utilizadas (aquelas marcadas *unused*), mesmo se esses dados não sejam mostrados na tela. Portanto, poderia acontecer que o programa indique que existam superposições nas partições,

enquanto que no vídeo pareça tudo correto. Aconselho portanto que se definam também os dados do início e do fim das partições não utlizadas.

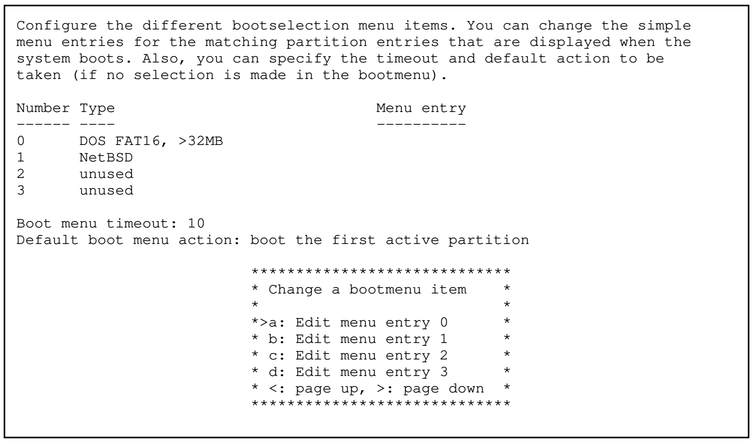
A tela que se mostra agora depende de vários fatores. Se foi cometido algum erro no particionamento (por exemplo foi criada uma partição que se superpõe a uma outra já existente) o sysinst mostrará uma mensagem e proporá que se retorne à tela de particionamento. Se os dados inseridos estiverem corretos, de acordo com a versão do NetBSD e os dados inseridos, agora poderia aparecer uma tela que avisa que a partição NetBSD encontra-se em uma posição em que a BIOS poderia não conseguir inicializá-la e pergunta se se quer prosseguir mesmo assim. Este fato poderia ser um problema nos PCs menos recentes. O PC utilizado para o exemplo inicializa-se perfeitamente. Não é possível termos uma regra geral (depende da BIOS). No momento, se aparecer este aviso, sugiro ignorá-lo e prosseguir.

Nota: esta não é uma limitação do NetBSD, mas de algumas BIOS mais velhas que não estão em condições de inicializar a partir de uma partição que se encontre além do limite de 1024 cilíndros. Para aprofundar este argumento é necessário estudar os diversos tipos de BIOS e as diversas modalidades de endereçamento utilizados (CHS físicos, CHS lógicos, LBA, etc.), argumentos que ultrapassam as finalidades deste guia e, em todo caso, não essenciais para a instalação.

Com as BIOS mais recentes, que suportam extensões int13, é possível instalar e inicializar o NetBSD sobre partições > 8 GB, contanto que o código de boot do NetBSD seja instalado no MBR, coisa que se pode obter instalando o gerenciador de boot do NetBSD.

Também a tela posterior depende da versão do NetBSD: a 1.5, se detecta a presença de outros sistemas operacionais, pergunta se queremos instalar o *gerenciador de inicialização*. Além de instalar o gerenciador de inicialização, o sysinst procede à sua configuração. Pode-se especificar que linhas escritas aparecerão quando se inicializa o computador, qual sistema terá inicialização por default e com que tempo de espera. A tela de configuração do gerenciador de boot é mostrada na Figura 3-12.

Figura 3-12. Configurando o seletor de inicialização

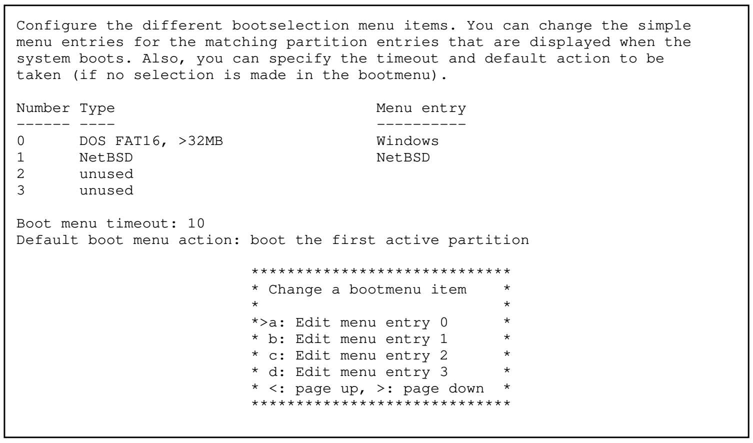


```
Configure the different bootselection menu items. You can change the simple
menu entries for the matching partition entries that are displayed when the
system boots. Also, you can specify the timeout and default action to be
taken (if no selection is made in the bootmenu).

Number Type                             Menu entry
------ ----                             ----------
0      DOS FAT16, >32MB
1      NetBSD
2      unused
3      unused

Boot menu timeout: 10
Default boot menu action: boot the first active partition

                          *****************************
                          * Change a bootmenu item    *
                          *                           *
                          *>a: Edit menu entry 0      *
                          * b: Edit menu entry 1      *
                          * c: Edit menu entry 2      *
                          * d: Edit menu entry 3      *
                          * <: page up, >: page down  *
                          *****************************
```

Nota: nessa tela podem-se usar as teclas < e > para movimentar-se pela lista de opções, se as teclas de setas não funcionam por algum motivo.

Indicar as partições que devem aparecer no menu do gerenciador de boot quando da inicialização do PC, selecionando uma das opções de "a" a "d". Na coluna "Menu entry" devem aparecer os nomes de todas as partições que se quer inicializar através do seletor de boot, como mostrado na Figura 3-13.

Figura 3-13. Configuração do seletor de inicialização



```
Configure the different bootselection menu items. You can change the simple
menu entries for the matching partition entries that are displayed when the
system boots. Also, you can specify the timeout and default action to be
taken (if no selection is made in the bootmenu).

Number Type                             Menu entry
------ ----                             ----------
0      DOS FAT16, >32MB                 Windows
1      NetBSD                           NetBSD
2      unused
3      unused

Boot menu timeout: 10
Default boot menu action: boot the first active partition

                          *****************************
                          * Change a bootmenu item    *
                          *                           *
                          *>a: Edit menu entry 0      *
                          * b: Edit menu entry 1      *
                          * c: Edit menu entry 2      *
                          * d: Edit menu entry 3      *
                          * <: page up, >: page down  *
                          *****************************
```

Escolhendo a opção "e", pode-se especificar um prazo de espera (timeout) para o menu de inicialização: transcorridos os segundos especificados o sistema especificado como default mediante a opção "f" inicializa-se automaticamente. Como default pode-se indicar

- uma partição específica
- um outro disco rígido
- a primeira partição ativa

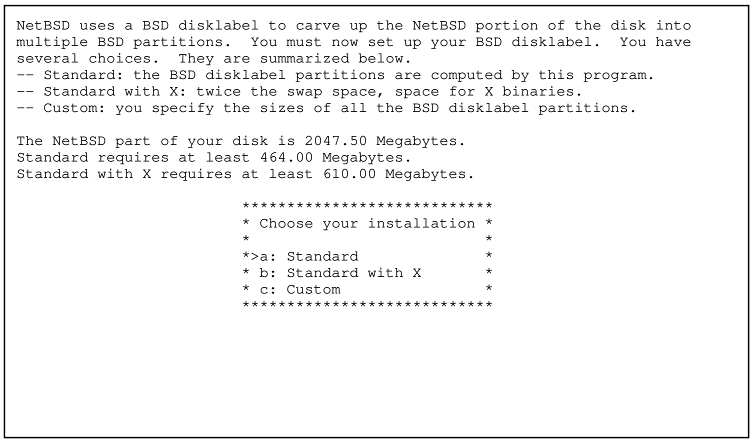
Ao sairmos dessa tela pode-se dizer terminada a parte da instalação relacionada com o particionamento "BIOS" do disco.

Até aqui falamos de partições referindo-nos àquelas que o NetBSD denomina partições Bios ou *slice*. Agora chegou o momento de definir as partições BSD.

3.4.6. Disklabel

O programa de instalação apresenta três alternativas para a criação das partições, como está ilustrado na Figura 3-14.

Figura 3-14. Tipo de disklabel



```
NetBSD uses a BSD disklabel to carve up the NetBSD portion of the disk into
multiple BSD partitions.  You must now set up your BSD disklabel.  You have
several choices.  They are summarized below.
-- Standard: the BSD disklabel partitions are computed by this program.
-- Standard with X: twice the swap space, space for X binaries.
-- Custom: you specify the sizes of all the BSD disklabel partitions.

The NetBSD part of your disk is 2047.50 Megabytes.
Standard requires at least 464.00 Megabytes.
Standard with X requires at least 610.00 Megabytes.

                         ****************************
                         * Choose your installation *
                         *                          *
                         *>a: Standard              *
                         * b: Standard with X       *
                         * c: Custom                *
                         ****************************
```

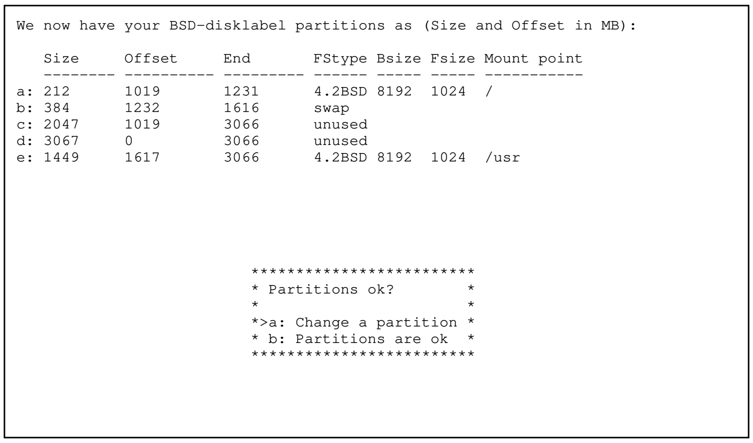
Para uma primeira instalação convém selecionar "a" ou "b", deixando ao sistema a tarefa de decidir sobre as partições e suas dimensões. Naturalmente, neste exemplo não nos contentaremos com uma solução tão simples e procuraremos complicar a nossa vida (para fins puramente didáticos, entenda-se) modificando manualmente as partições.

> **Nota:** quando se deixa o sistema decidir é necessária, mesmo assim, uma certa cautela. Se o espaço em disco for pouco, controlar as indicações da tela sobre o espaço necessário para a instalação padrão (standard). Se não é suficiente, será necessário modificar o disklabel ou criar diretamente um disklabel personalizado. O programa de instalação da versão 1.5 reconhece automaticamente essas incongruências e mostra uma tela de advertência.

3.4.7. Criação de um disklabel

A estratégia que seguiremos será de deixar o sistema criar automaticamente um disklabel e depois modificá-lo manualmente. Escolhendo a opção "b" da Figura 3-14 é mostrada a tela da Figura 3-15.Figura 3-15.

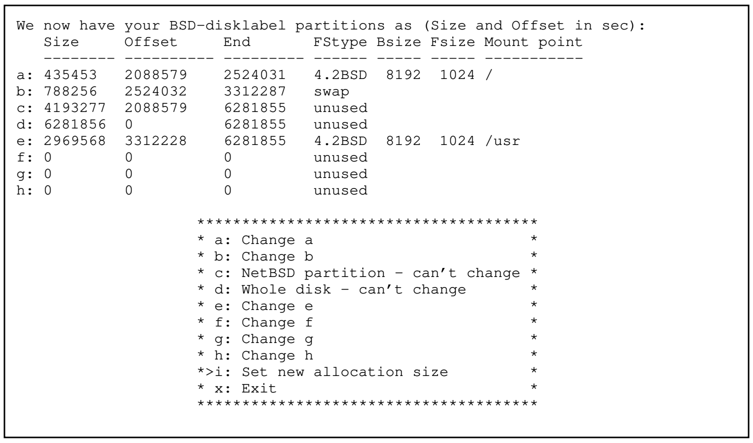
Figura 3-15. Disklabel padrão



```
We now have your BSD-disklabel partitions as (Size and Offset in MB):

   Size     Offset     End       FStype Bsize Fsize Mount point
   -------- ---------- --------- ------ ----- ----- -----------
a: 212      1019       1231      4.2BSD 8192  1024  /
b: 384      1232       1616      swap
c: 2047     1019       3066      unused
d: 3067     0          3066      unused
e: 1449     1617       3066      4.2BSD 8192  1024  /usr





                          *************************
                          * Partitions ok?        *
                          *                       *
                          *>a: Change a partition *
                          * b: Partitions are ok  *
                          *************************
```

Nesse ponto poder-nos-íamos limitar à confirmação (escolhendo a opção "b") e o trabalho estaria terminado. Suponhamos todavia que a partição de swap é muito grande e que queiramos reduzí-la para aumentar a partição `usr`. Escolhendo a opção "a" mudamos o tamanho da partição de memória adicional (swap). Na nova tela mudamos a unidade de medida, passando para setores, com o resultado mostrado na Figura 3-16.

Figura 3-16. Modificação do disklabel (sec)



```
We now have your BSD-disklabel partitions as (Size and Offset in sec):
   Size     Offset     End       FStype Bsize Fsize Mount point
   -------- ---------- --------- ------ ----- ----- -----------
a: 435453   2088579    2524031   4.2BSD  8192  1024 /
b: 788256   2524032    3312287   swap
c: 4193277  2088579    6281855   unused
d: 6281856  0          6281855   unused
e: 2969568  3312228    6281855   4.2BSD  8192  1024 /usr
f: 0        0          0         unused
g: 0        0          0         unused
h: 0        0          0         unused

                    **************************************
                    * a: Change a                        *
                    * b: Change b                        *
                    * c: NetBSD partition - can’t change *
                    * d: Whole disk - can’t change       *
                    * e: Change e                        *
                    * f: Change f                        *
                    * g: Change g                        *
                    * h: Change h                        *
                    *>i: Set new allocation size         *
                    * x: Exit                            *
                    **************************************
```

A seqüência das partições (isto é, as letras atribuídas às partições) é padrão. Algumas letras têm um significado bem preciso.

- *a* é a partição *root*.
- *b* é a partição de *swap*.
- *c* cobre a partição NetBSD inteira (e portanto não cobre a parte DOS).
- *d* cobre o disco todo. Note-se que a área coberta por esta partição "extrapola" a parte destinada ao NetBSD. Com um mecanismo análogo a este (ou seja, criando uma partição que "extrapola" a fatia do NetBSD) será possível permitir ao NetBSD "ver" as partições DOS, Linux, etc.

- *e* é a primeira partição livre, sobre a qual é comumente montado `/usr`.

> **Nota:** o significado das letras atribuídas às partições não é o mesmo sobre todas as plataformas suportadas pelo NetBSD. O elenco precedente é específico do "port" i386.

As partições *b* e *c* deveriam ser deixadas inalteradas (são modificadas apenas em caso de definição errada por parte do sysinst). Pode-se mudar tranqüilamente a dimensão da partição *a*, mas é melhor manter o ponto de *montagem*. Podem ser criadas até 8 partições, para as quais são disponibilizadas as letras de *e* a *h*.

Para modificar a partição adicional de memória (swap) é necessário mudar também a partição *e*, para fazê-la iniciar depois do fim da partição *b*. As partições *c* e *d* não podem ser modificadas (coisa suficientemente óbvia, dado o seu significado).

Suponhamos querer atribuir 150 MB para o swap (correspondentes a 307200 setores). A partição *b*, portanto, iniciará no setor 2524032 e terminará no setor 281231 (2524032+307200-1).

```
id:      Size    Offset       End FStype Bsize Fsize Mount point
---      ----    ------       --- ------ ----- ----- -----------
 a:    435453   2088579   2524031 4.2BSD  8192  1024 /
 b:    307200   2524032   2831231   swap
 ...
```

O espaço remanescente destinamos à partição *e*: início no setor que imediatamente segue o setor final da partição *b*. O fim é fixo (corresponde ao último setor da partição NetBSD) e o tamanho é dado por *End - Offset + 1*.

```
id:      Size    Offset       End FStype Bsize Fsize Mount point
---      ----    ------       --- ------ ----- ----- -----------
 a:    435453   2088579   2524031 4.2BSD  8192  1024 /
 b:    307200   2524032   2831231   swap
 ...
 e:   3450624   2831232   6281855 4.2BSD  8192  1024 /usr
```

O exemplo precedente mostra a situação que se quer obter com o disklabel. Para modificar as duas partições usam-se as opções "b" e "e", e inserimos os dados calculados.

A tela de modificação é mostrada na Figura 3-17.

Figura 3-17. Modificação de uma partição BSD

```
You should set the file system (FS) kind first.  Then set the other values.

The current values for partition b are:

    Size      Offset    End       FStype Bsize Fsize Mount point
    --------- --------- --------- ------ ----- ----- -----------
 b: 788256    2524032   3312287   swap







                               ******************
                               * Change what?   *
                               *                *
                               *>a: FS kind     *
                               * b: Offset/size *
                               * c: Bsize/Fsize *
                               * d: Mount point *
                               * x: Exit        *
                               ******************
```

O disklabel modificado pode ser visto na Figura 3-18.

Figura 3-18. Disklabel modificado

```
We now have your BSD-disklabel partitions as (Size and Offset in sec):
   Size     Offset     End       FStype Bsize Fsize Mount point
   -------- ---------- --------- ------ ----- ----- -----------
a: 435453   2088579    2524031   4.2BSD  8192  1024 /
b: 307200   2524032    2831231   swap
c: 4193277  2088579    6281855   unused
d: 6281856  0          6281855   unused
e: 3450624  2831232    6281855   4.2BSD  8192  1024 /usr
f: 0        0          0         unused
g: 0        0          0         unused
h: 0        0          0         unused

                     **************************************
                     * a: Change a                        *
                     * b: Change b                        *
                     * c: NetBSD partition - can't change *
                     * d: Whole disk - can't change       *
                     * e: Change e                        *
                     * f: Change f                        *
                     * g: Change g                        *
                     * h: Change h                        *
                     *>i: Set new allocation size         *
                     * x: Exit                            *
                     **************************************
```

> **Dimensões das partições:** é muito difícil fornecer critérios gerais para se decidir quantas partições criar e como dimensioná-las. Depende muito do uso que se fará do computador (servidor, workstation, servidor de correio eletrônico, etc.) Para um posto de trabalho (workstation) podem bastar as partições *a* (para `/`) de pelo menos 50 MB, *b* (para o swap), pouco maior que a RAM disponível e *e* (para `/usr`), o maior possível. Para um servidor será utilizada uma subdivisão mais sofisticada. As estratégias de subdivisão do disco são entretanto muitíssimas e exorbitam o objetivo deste guia.

Quando o resultado é satisfatório pode-se selecionar "Exit" para salvar e sair, retornando-se à tela da Figura 3-15 onde, desta vez, se escolhe a opção "b".

3.4.8. Operações finais

A criação das partições (BIOS e BSD) era sem dúvida o maior obstáculo da instalação. A partir de agora a estrada só tem descida. Para completar o processo é necessário dar um nome (ao bel prazer) ao disco (por default é proposto *mydisk*) e confirmar as operações executadas.

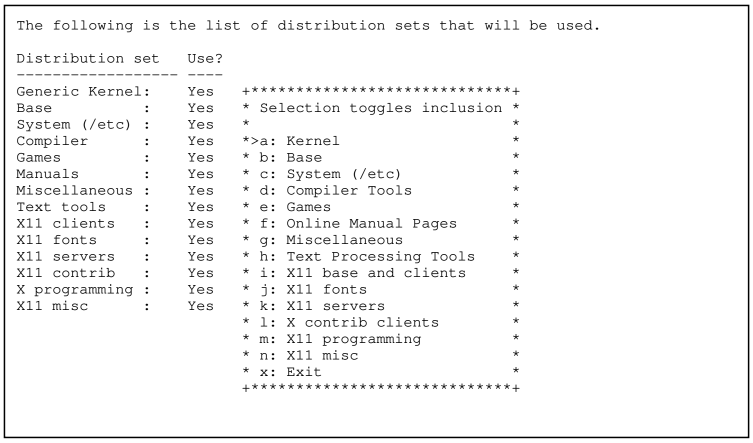
> **Nota:** tudo o que foi feito até agora não foi gravado no disco. É ainda possível mudar de idéia e abandonar a instalação.

O sistema provê agora à criação das partições e dos sistemas de arquivos (filesystems) executando os comandos **fdisk**, **newfs**, **fsck** e **installboot**. Depois, se tudo vai bem, passa-se à instalação dos conjuntos de arquivos das distribuições e do kernel.

3.4.9. Selecionando o tipo de meio para a instalação

A primeira parte da instalação está terminada. O próximo passo é a escolha do tipo de instalação, que pode ser *full* ou *custom*. A opção *full* instala a distribuição inteira, enquanto que a opção *custom* instala os conjuntos básicos (obrigatório) e permite selecionar os sets restantes. É aconselhável escolher a opção *full*, desde que não se tenha particulares problemas de espaço em disco. A seguir escolheremos a opção *custom*, que leva à telinha da Figura 3-19.

Figura 3-19. Seleção dos itens (sets) a instalar



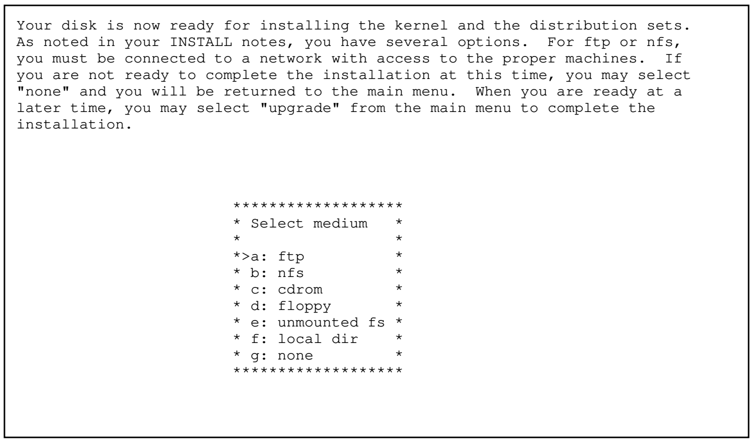
```
The following is the list of distribution sets that will be used.

Distribution set   Use?
------------------ ----
Generic Kernel:    Yes   +*****************************+
Base          :    Yes   * Selection toggles inclusion *
System (/etc) :    Yes   *                             *
Compiler      :    Yes   *>a: Kernel                   *
Games         :    Yes   * b: Base                     *
Manuals       :    Yes   * c: System (/etc)            *
Miscellaneous :    Yes   * d: Compiler Tools           *
Text tools    :    Yes   * e: Games                    *
X11 clients   :    Yes   * f: Online Manual Pages      *
X11 fonts     :    Yes   * g: Miscellaneous            *
X11 servers   :    Yes   * h: Text Processing Tools    *
X11 contrib   :    Yes   * i: X11 base and clients     *
X programming :    Yes   * j: X11 fonts                *
X11 misc      :    Yes   * k: X11 servers              *
                         * l: X contrib clients        *
                         * m: X11 programming          *
                         * n: X11 misc                 *
                         * x: Exit                     *
                         +*****************************+
```

Os primeiros três itens são obrigatórios porque sem eles o sistema não pode funcionar. Os outros podem ser ativados ou desativados inserindo-se a opção no menu. Inicialmente todos os itens aparecem selecionados. Nesse caso a instalação seria equivalente à opção de instalação *full* do menu anterior. Escolhamos instalar tudo e prossigamos com a opção "x: Exit".

Sucessivamente o sysinst pergunta se desejamos ver os nomes dos arquivos extraídos durante o processo de extração

Nesse ponto o programa de instalação tem necessidade de encontrar os arquivos da distribuição (os arquivos `.tgz`): é necessário indicar-lhe onde eles se encontram. O menu oferece várias possibilidades:

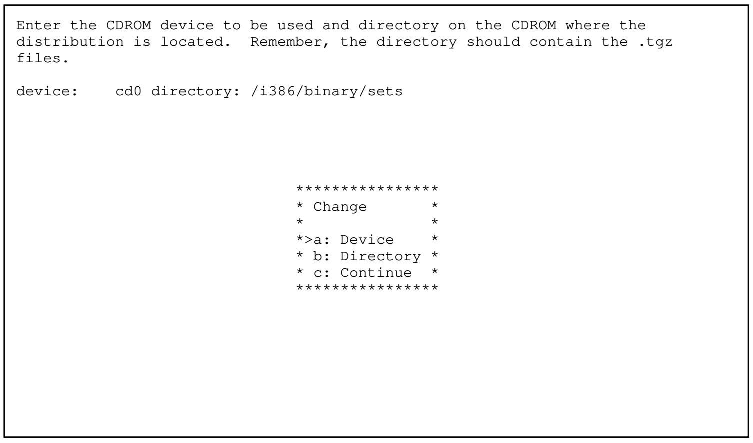
Figura 3-20. Meios para instalação



```
Your disk is now ready for installing the kernel and the distribution sets.
As noted in your INSTALL notes, you have several options.  For ftp or nfs,
you must be connected to a network with access to the proper machines.  If
you are not ready to complete the installation at this time, you may select
"none" and you will be returned to the main menu.  When you are ready at a
later time, you may select "upgrade" from the main menu to complete the
installation.



                        *******************
                        * Select medium   *
                        *                 *
                        *>a: ftp          *
                        * b: nfs          *
                        * c: cdrom        *
                        * d: floppy       *
                        * e: unmounted fs *
                        * f: local dir    *
                        * g: none         *
                        *******************
```

As várias opções são explicadas em detalhe no habitual arquivo `INSTALL`. Note-se que é possível instalar também a partir de uma partição não montada com sistema de arquivos diferente, como por exemplo uma partição MSDOS. Isto significa que é possível copiar todos os arquivos da distribuição em uma partição MSDOS e utilizá-los depois para a instalação.

Figura 3-21. Instalando com CD-ROM



```
Enter the CDROM device to be used and directory on the CDROM where the
distribution is located.  Remember, the directory should contain the .tgz
files.

device:    cd0 directory: /i386/binary/sets




                               ****************
                               * Change       *
                               *              *
                               *>a: Device    *
                               * b: Directory *
                               * c: Continue  *
                               ****************
```

Ao selecionarmos "cdrom" o sistema pede o nome do dispositivo (por exemplo, cd0) e realiza automaticamente a sua montagem. Ademais, é necessário especificar o caminho (path) dos arquivos da distribuição binária: o sysinst apresenta um caminho por default. No caso do CD-ROM criado por mim, o NetBSD encontra-se no diretório `NetBSD-1.5` e, portanto, é necessário modificar o caminho utilizando a opção "b". O novo percurso será:

```
/NetBSD-1.5/i386/binary/sets
```

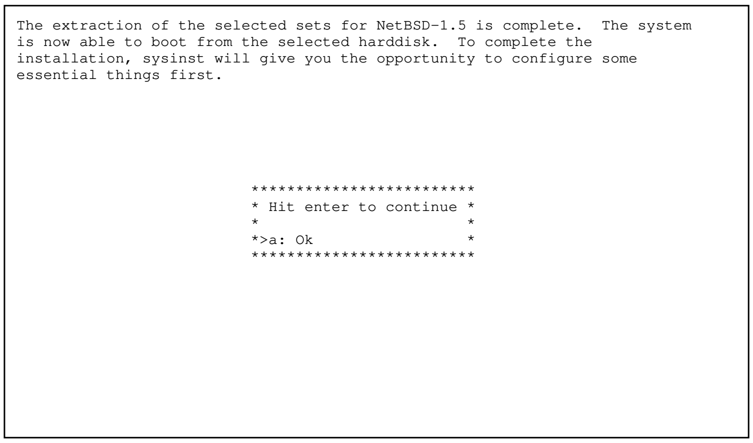
> **Nota:** nesse momento apresenta-se o problema do teclado americano, que já apontei antes. Ver Secção 3.3.1.

O nome do dispositivo: também é possível conferir o nome do dispositivo correspondente ao CD-ROM.

1. Teclar Ctrl-Z para suspender o sysinst: passamos ao prompt da shell.
2. Dar o comando **cat /kern/msgbuf**. Desse modo podemos ver as mensagens de inicialização do kernel. Entre elas aparece também a detecção do CD-ROM com o nome do dispositivo (por exemplo, *cd0*).
3. Se as linhas escritas deslizam pela tela muito rapidamente, pode-se também utilizar **ed /kern/msgbuf**, (supondo-se o conhecimento do ed).
4. Voltar ao programa de instalação com o comando **fg**.

Ao término da instalação é mostrada uma mensagem que confirma se as operações foram bem sucedidas. Enfim, selecionando "ok", são criados os device files (arquivos de dispositivo).

Figura 3-22. Instalação terminada!



```
The extraction of the selected sets for NetBSD-1.5 is complete.  The system
is now able to boot from the selected harddisk.  To complete the
installation, sysinst will give you the opportunity to configure some
essential things first.






                          *************************
                          * Hit enter to continue *
                          *                       *
                          *>a: Ok                 *
                          *************************
```

A instalação está terminada. O sysinst oferece a possibilidade de se efetuar algumas configurações e a timezone (isto é, o fuso horário em que se está). (Para o Brasil podemos selecionar quatro timezones possíveis. A hora oficial, de Brasília, é Brazil/East. Nota do tradutor.). Na tela seguinte pode-se definir a senha do root. Nesse ponto retorna-se ao menu principal e o sistema pode ser inicializado pela primeira vez.

Capítulo 4. A primeira inicialização

Ao término da instalação procede-se à primeira inicialização do sistema a partir do disco rígido. Se tudo correu bem, nos encontraremos diante de um prompt de login. Quem está habituado a encontrar um sistema pré-configurado ao fim da instalação do sistema operacional poderá sentir-se um pouco deslocado com o NetBSD. Seguindo as intruções deste guia, no entanto, poder-se-á configurar rapidamente o sistema, ao mesmo tempo em que se aprende como funciona.

4.1. Se alguma coisa deu errado

Se alguma coisa deu errado na instalação, pode ocorrer que o sistema não se inicialize. Um dos problemas mais comuns é a instalação incorreta do gerenciador de boot. Nesse caso, inserir o disquete de instalação no drive e reinicializar a máquina. Quando aparece a linha

```
booting fd0a:netbsd - starting in ...
```

pressionar a barra de espaço durante a contagem regressiva de 5 segundos. O boot se interrompe e aparece o prompt. Junto com o ponto de interrogação se visualiza a ajuda.

```
type "?" or "help" for help.
> ?
commands are:
boot [xdNx:][filename] [-adrs]
     (ex. "sd0a:netbsd.old -s")
ls [path]
dev xd[N[x]]:
help|?
quit
> boot wd0a:netbsd
```

Com o comando precedente indica-se ao sistema para se inicializar a partir do disco rígido ao invés do floppy. Se o sistema inicializa-se normalmente do disco rígido, trata-se apenas de um problema do gerenciador de boot. Caso contrário, provavelmente será necessário repetir a instalação. Para instalar novamente o gerenciador de boot ou modificar a sua configuração, usa-se o comando **fdisk** com a opção `-B` (ver Secção 20.4 para uma explicação detalhada).

4.2. Login

Para o primeiro login é preciso utilizar o usuário *root* (a conta do administrador do sistema), o único definido no sistema ao término da instalação. Se durante a instalação foi definida uma senha para o root, é necessário inserí-la. Do contrário basta pressionar a tecla Enter.

```
NetBSD/i386 (Amnesiac) (ttyE0)
login: root
password
...
We recommend creating a non-root account and using su(1) for root access.
```

```
#
```

4.3. Mudar a disposição do teclado

Até aqui a disposição do teclado permaneceu aquela da instalação, isto é, o teclado americano. Antes de começar a configurar o sistema, convém mudar a disposição para utilizar o teclado italiano executando o comando

```
# wsconsctl -k -w encoding=it
encoding -> it
```

A lista completa das configurações do teclado suportadas encontra-se em `/sys/dev/wscons/wsksymdef.h`, mas as mais comuns são:

- de
- dk
- fr
- it
- jp
- sv
- uk
- us
- no
- es

Esta definição durará até a próxima inicialização. Para restabelecê-la automaticamente, inserir o comando acima no fim do arquivo `/etc/wscons.conf`: uma olhada nos comentários contidos no arquivo deveria ser suficiente para compreender seu mecanismo de funcionamento. Para aprofundar o argumento: wscons.conf(5). Por exemplo, para utilizar o teclado italiano basta acrescentar a seguinte linha ao arquivo `/etc/wscons.conf`:

```
encoding it
```

Podemos também personalizar algumas teclas, para acrescentar as chaves, o acento til etc., criando o arquivo `/etc/itmap.it` (o nome nós escolhemos à vontade) com o seguinte conteúdo:

```
keycode 12 = apostrophe question grave
keycode 13 = igrave asciicircum asciitilde
keycode 8 = 7 slash braceleft
keycode 9 = 8 parenleft bracketleft
keycode 10 = 9 parenright bracketright
keycode 11 = 0 equal braceright
```

Enfim, acrescentar a linha que carrega o mapa no habitual arquivo `/etc/wscons.conf`:

```
mapfile /etc/itmap.it
```

Mais à frente veremos como fazer para que o NetBSD use por default o teclado italiano compilando um novo kernel e, também, como personalizar a disposição do teclado.

4.4. O comando mais importante

Ainda que possa parecer estranho, o comando mais importante nesse momento é o **man**, que mostra uma página de manual.

Comando **man** para ter informações sobre um comando e **man -k tópico** para ter um conjunto de comandos que têm a ver com um 'tópico' (pode-se também usar **apropos tópico**).

Naturalmente o primeiro comando a acionar é

```
# man man
```

O manual está subdividido em nove seções que não contêm informações apenas sobre comandos, mas também descrições de algumas características do sistema. Por exemplo, podemos consultar a página do hier(7) que descreve detalhadamente a arquitetura do sistema de arquivos do NetBSD

```
# man hier
```

Para quem tenha ficado um pouco perplexo, *hier* é a abreviação de hierarchy. Outras páginas informativas desse tipo são, por exemplo, release(7) e packages(7). Cada seção do manual tem ademais uma página *intro* que lhe descreve o conteúdo. Experimentar, por exemplo

```
# man 8 intro
```

Exemplo 4-1. As seções do manual

1. comandos gerais do usuário do ambiente BSD
2. chamadas do sistema
3. funções da biblioteca C
4. dispositivos de drivers (device drivers) e suas funções
5. formatos dos arquivos
6. jogos
7. informações variadas (por exemplo, os pacotes de macros groff)
8. manutenção do sistema
9. estrutura do kernel

Às vezes um argumento está presente em mais de uma seção do manual. Para visualizar o argumento da seção que interessa, é necessário indicar o número da seção no comando man. Por exemplo, *time* está presente na seção 1 (o comando de usuário time), na seção 3 (a função time na biblioteca C) e na seção 9 (a variável de sistema time). Se nos interessa uma função determinada, devemos especificar

```
# man 3 time
```

Para visualizar todas as seções

```
# man -a time
```

4.5. Mudando a senha do root

Ao término da instalação, o único usuário definido no sistema é o root, o administrador do sistema. Se no decorrer da instalação não foi definida uma senha para o root (o que não era possível nas versões anteriores à 1.5), chegou o momento de fazê-lo usando o comando **passwd**.

```
# passwd
Changing local password for root.
New password:
Retype new password:
```

As senhas não são vistas sobre a tela quando são escritas. Mais â frente veremos como adicionar outros usuários.

4.6. Mudando a shell

A shell por omissão (default) do root é csh. Se isto não lhe diz nada, convém mantê-la e começar a estudar-lhe o funcionamento (em **man csh**). Trata-se de uma shell ótima para o uso interativo, ainda se lhe falta a *history editing* (falta remediada por tcsh, bash, ksh e da própria `/bin/sh` do NetBSD). Todavia, se a csh não lhe agrada, você pode mudar a shell do usuário root com o comando **chsh**. No todo, as shells disponíveis no sistema são

- csh
- sh
- ksh

A nova shell será utilizada no próximo login. Enquanto isso não acontece convém dar o comando

```
# set filec
```

que habilita o completamento dos nomes na linha de comando (com a tecla Esc).

Naturalmente é possível instalar muitas outras shells (tcsh, bash, zsh, etc.) utilizando o sistema de gerenciamento de pacotes, que veremos mais adiante.

Este é um bom momento para criar/copiar os arquivos de configuração da shell (.cshrc, .login, etc.).

4.7. Mudando a hora do sistema

O NetBSD, como todos os sistemas Unix, utiliza um relógio de sistema ajustado à hora de Greenwich e não à hora local. Infelizmente isso não é possível se usamos uma dupla inicialização com, por exemplo, Windows, que exige que a hora do sistema seja ajustada à hora local. Para resolver este problema é necessário informar o NetBSD, agindo sobre a variável rtc_offset do kernel. Pode-se recompilar o kernel depois de ter modificado o arquivo de configuração ou modificar diretamente o kernel existente (a modificação tornar-se-á efetiva na próxima inicialização) com a operação ilustrada pelo próximo exemplo.

```
# gdb --write /netbsd
GNU gdb 4.17
Copyright 1998 Free Software Foundation, Inc.
...
This GDB was configured as "i386--netbsd"...(no debugging symbols found)...
(gdb) set rtc_offset=-60
(gdb) quit
```

Note-se que o valor é indicado em minutos (-60) e representa, no caso, um fuso horário a oeste de Greenwich.

Para visualizar o valor corrente da variável:

```
# sysctl kern.rtc_offset
kern.rtc_offset = -60
```

Nesse ponto simplesmente indicamos ao kernel como fazer para converter em UTC a hora do relógio do sistema. Se durante a instalação não foi definido o fuso horário, deve-se fazê-lo agora. Em todo caso convém verificar se a configuração está correta. No Brasil há quatro fusos horários, todos eles a oeste de Greenwich. Para a hora de Brasília, a configuração seria a seguinte (Nota do Tradutor):

```
# rm -f /etc/localtime
# ln -s /usr/share/zoneinfo/Brasil/East /etc/localtime
```

Uma vez que o sistema estiver configurado corretamente, podem-se mudar a data e a hora com o comando seguinte:

```
# date [[[[[cc]yy]mm]dd]hh]mm
```

4.8. Configurando o `/etc/rc.conf`

Nos sistemas BSD o arquivo `/etc/rc.conf` é quem controla o que é executado automaticamente na inicialização do sistema. Em um certo sentido pode-se dizer que ele é o coração da configuração do sistema.

O mecanismo de funcionamento desse arquivo mudou com a versão 1.5 do NetBSD. Nas versões anteriores à 1.5, todos os valores a definir encontravam-se no arquivo `/etc/rc.conf`. Inicialmente ele continha as configurações por default que o usuário deveria personalizar depois, modificando diretamente o próprio arquivo. A partir da versão 1.5 as configurações por default estão definidas no arquivo `/etc/defaults/rc.conf`. O `/etc/rc.conf` contém apenas as opções modificadas pelo usuário com respeito aos defaults. As opções modificadas ficam portanto definidas no `/etc/rc.conf`, usando-se o `/etc/defaults/rc.conf` (que não deve ser modificado) como referência.

Um exame deste arquivo é altamente instrutivo e a leitura da página de manual correspondente, que descreve todas as opções, é enfaticamente aconselhada.

```
# man rc.conf
```

Ao término da instalação o sysinst já deveria ter efetuado as modificações fundamentais no arquivo. Atentar para que estejam presentes as seguintes linhas:

```
rc_configured=YES
wscons=YES
```

Se “rc_configured” não está em YES o sistema entre em modo monousuário no fim do boot. A diretriz “wscons” configura o driver de console. Já que estamos aqui, se se que imprimir, convém inserir a diretriz:

```
lpd=YES
```

para ativar o dæmon de impressão.

4.9. Estabelecendo o nome do host

Chegou o momento de indicar ao sistema o nome do nosso host. Há dois modos simples de fazê-lo. Se a máquina faz parte de uma rede, o nome da rede será obrigatório e o nome do host ser-nos-á provavelmente indicado pelo administrador. Se, ao contrário, se trata de um micro isolado (por exemplo, o computador de casa) pode-se escolher um nome de fantasia. Por exemplo, ape.insetti.net. Para ulteriores aprofundamentos ver o Capítulo 9.

A primeira maneira de se indicar o nome do host é inserindo a seguinte diretiva no arquivo `/etc/rc.conf`:

```
hostname=ape.insetti.net
```

No lugar de indicar o nome do host diretamente no arquivo de configuração, pode-se escrevê-lo no arquivo `/etc/myname`. Os dois métodos são inteiramente equivalentes.

```
# echo ape.insetti.net > /etc/myname
```

4.10. Reinicializando o sistema

Nesta primeira seção

- Configuramos o teclado
- Mudamos a senha do root
- Mudamos a shell do root (opcional)

- Mudamos a hora do sistema
- Definimos o fuso horário de localização
- Verificamos o arquivo `/etc/rc.conf`
- Definimos o nome do host

Ao término destas primeiras operações é necessário reinicializar o sistema:

```
# reboot
```

Capítulo 5. Operações subseqüentes

O primeiro boot já é história passada e a configuração básica do sistema já está completa. Nesta seção descreveremos, sem uma ordem muito rígida, algumas operações comuns e alguns comandos úteis.

5.1. dmesg

Durante a inicialização do sistema o kernel escreve no vídeo uma longa série de mensagens relativas às operações executadas e aos periféricos reconhecidos. Estas mensagens são fundamentais seja para diagnosticar eventuais problemas, seja para determinar os nomes dos dispositivos de driver associados aos periféricos. O comando **dmesg** reapresenta as mensagens explicitadas pelo kernel durante a inicialização do sistema, permitindo examiná-las com toda a calma. Em particular, quando se solicita a assistência de umas das mailing lists do NetBSD porque algum periférico não funciona a contento, sempre se deve anexar a mensagem de output do dmesg (ou pelo menos a parte relativa ao periférico em questão).

```
# dmesg | more
```

Se alguma coisa no sistema não funciona como deveria e você decide recorrer à ajuda de uma das mailing lists do NetBSD, lembre-se de anexar à sua mensagem o output do dmesg relevante para o seu problema. Estas informações serão úteis para que outros diagnostiquem o seu problema.

> Desde a versão 1.4.2 do NetBSD o processo de inicialização escreve uma cópia do output do comando dmesg no arquivo `/var/run/dmesg.out`. Esta característica é particularmente útil para os sistemas (por exemplo, servidores) que raramente são reinicializados. No curso do funcionamento, efetivamente, as novas mensagens acrescentadas causam a eliminação das mais velhas.

5.2. Montando o cdrom

Uma das questões que reaparecem com maior freqüência nas mailing lists é "como se faz para montar o CD-ROM?". Vejamos agora o procedimento a adotar. O dispositivo correspondente ao CD-ROM pode-se identificar através do já citado comando **dmesg**. Suponhamos, por exemplo, que o dmesg torne visíveis as seguintes informações:

```
# dmesg | grep ^cd
cd0 at atapibus0 drive 1: <ASUS CD-S400/A, , V2.1H> type 5 cdrom removable
```

Neste caso o CD-ROM poderá ser montado com

```
# mkdir /mnt/cdrom
# mount -t cd9660 -o ro /dev/cd0a /mnt/cdrom
```

Para evitar a repetição destes comandos toda vez, convém inserir uma linha no `/etc/fstab`.

```
/dev/cd0a /mnt/cdrom cd9660 ro,noauto 0 0
```

A essa altura, sem necessidade de reinicializar, para montar o CD-ROM basta dar o comando

```
# mount /mnt/cdrom
```

Se o CD-ROM está montado não é possível ejetá-lo manualmente (isto é, pressionando o botãozinho do driver). É necessário que antes seja dado o comando

```
# umount /mnt/cdrom
```

Alternativamente pode-se ejetá-lo via software (mesmo se estiver montado) com o comando

```
# eject cd0
```

5.3. Montando o disquete

Para usar o floppy disk pode-se utilizar um mecanismo similar ao do CD-ROM. Em primeiro lugar cria-se o ponto de montagem. Por exemplo:

```
# mkdir /mnt/floppy
```

Para ler e escrever um disquete em formato MSDOS pode-se agora executar o comando:

```
# mount -t msdos /dev/fd0a /mnt/floppy
```

Se muitos disquetes em formato MSDOS são usados, a minha sugestão é que se instale o pacote “mtools”, que acessa diretamente o disquete (e também uma partição DOS) de modo muito cômodo.

5.4. Acessando uma partição MS-DOS/Windows

Se o NetBSD compartilha o disco rígido com o MS-DOS ou com Windows, é possível tornar visíveis as partições de cada sistema modificando o disklabel. A primeira coisa a fazer é determinar a geometria da partição DOS. Por exemplo, com o comando **fdisk**.

```
# fdisk wd0
NetBSD disklabel disk geometry:
cylinders: 6232 heads: 16 sectors/track: 63 (1008 sectors/cylinder)
...
Partition table:
0: sysid 6 (Primary 'big' DOS, 16-bit FAT (> 32MB))
    start 63, size 2088516 (1019 MB), flag 0x80
        beg: cylinder    0, head   1, sector  1
        end: cylinder  259, head   0, sector  4
1: sysid 169 (NetBSD)
    start 2088579, size 4193277 (2047 MB), flag 0x0
        beg: cylinder  259, head   0, sector  4
        end: cylinder  779, head   0, sector  1
2: <UNUSED>
3: <UNUSED>
```

> **Nota:** neste exemplo foi usado o disco *wd0*. Cada um deve substituir a identificação correta para seu próprio disco (por exemplo, *sd0*, etc.)

O exame do output do **fdisk** mostra que a partição DOS inicia-se no setor 63 e tem o tamanho de 2088516. A partição NetBSD inicia-se no setor 2088579 (note-se que 2088579 = 2088516 + 63). Esses dados servem para modificar o disklabel BSD, usando o editor especificado pela variável do ambiente $EDITOR. É necessário acrescentar uma linha ao disklabel, especificando-se os dados da partição MS-DOS que acabamos de descobrir, usando a primeira letra livre como identificador da partição.

```
# disklabel -e wd0
...
#        size   offset     fstype  [fsize bsize  cpg]
...
  e:  3450624  2831232    4.2BSD    1024  8192    16   # (Cyl.  2808* - 6231)
  f:  2088516       63    MSDOS
```

As partições de "a" a "e" estavam ocupadas e, portanto, o identificador escolhido para a linha foi “f”. Os campos “size” e “offset” foram preenchidos com os dados encontrados antes. Agora se cria o ponto de montagem, por exemplo:

```
# mkdir /msdos
```

enfim, acrescenta-se a linha seguinte em `/etc/fstab`

```
/dev/wd0f /msdos msdos rw,noauto 1 3
```

Para acessar a partição MS-DOS basta executar o comando:

```
# mount /msdos
```

Com o método descrito acima podem-se ler partições FAT e FAT32. Se queremos que a partição MS-DOS seja montada automaticamente na inicialização, basta eliminar a opção “noauto”, da seguinte maneira:

```
/dev/wd0f /msdos msdos rw 1 3
```

5.5. Acrescentar um novo usuário

O gerenciamento dos usuários e dos grupos realiza-se por meio dos programas **useradd(8)** e **groupadd(8)**. Por exemplo, para acrescentarmos um novo usuário carlo designando-o para um grupo com seu próprio nome, criemos em primeiro lugar o grupo com:

```
# groupadd carlo
```

depois criemos o novo usuário:

```
# useradd -G wheel -g carlo -m -s /bin/ksh carlo
```

Com o comando acima fizemos várias operações de uma vez só:

-G wheel

indica o adicionamento do usuário ao grupo “wheel” (o grupo do root), o que permitirá a execução do comando **su**.

-g carlo

especifica o grupo a que pertence o usuário. No caso, o grupo precedentemente criado.

-m

ordena a criação do diretório home para o usuário.

-s /bin/ksh

escolhe a shell ksh para o usuário.

carlo

o nome do usuário a adicionar.

Neste exemplo escolhemos manualmente a shell ksh, mas os valores utilizados por default pelo useradd podem ser configurados. Ver a página de manual useradd(8). O comando

```
# useradd -D
```

mostra os valores correntes por default.

O comando:

```
# useradd -D -s /bin/ksh
```

estabelece a ksh como shell por default do comando useradd.

> **Nota:** o instrumento a se utilizar para acrescentar ou modificar os usuário manualmente é o **vipw**, que permite modificar de modo seguro o arquivo das senhas.

5.6. Shadow password

As shadow password são o default para o NetBSD, e não podem ser desabilitadas. Todas as senhas de `/etc/passwd` contêm um ’*’. As senhas criptografadas estão contidas em um arquivo separado, `/etc/master.passwd`, que pode ser lido somente pelo root. Quando se lança o **vipw** para editar o arquivo das senhas, o programa abre uma cópia do `/etc/passwd`. Quando é encerrado, vipw verifica a validade da cópia, gera `/etc/passwd` e instala o novo `/etc/master.passwd`. Por fim, executa **pwd_mkdb** que cria os arquivos `/etc/pwd.db` e `/etc/spwd.db`, que são bancos de dados equivalentes a `/etc/passwd` e `/etc/master.passwd`, permitindo um acesso mais veloz.

A conseqüência de tudo isso é que as shadow password são geridas de modo automático (transparente, para usar um termo que está em moda hoje) pelo NetBSD. Enquanto for usado o vipw para se editar o arquivo das senhas, não é necessária qualquer mudança nos procedimentos padrão de administração do sistema.

É importante lambrar que a administração das contas deve *sempre* ser efetuada usando-se o **vipw** (ou chfn, chsh, chpass, passwd). Não é necessário *jamais* modificar diretamente o `/etc/master.passwd`.

5.7. Localização

A localização é um setor em que o NetBSD infelizmente ainda está um tanto débil. Seu suporte ainda é carente, embora venha a ser estendido nas próximas edições do sistema, já que o código necessário já está presente na versão current. Nesse meio tempo podemos dar um jeito. Para os usuários europeus, a primeira coisa a fazer é obter o tarball `locale.tgz` no site do NetBSD e instalá-lo com os comandos:

```
# cd /
# gunzip < /path/to/locale.tgz | tar xvf -
```

Subseqüentemente acrescentemos as instruções seguintes aos arquivos de configuração da shell (por exemplo `/etc/profile`):

```
LANG=it
export LANG
LC_CTYPE=iso_8859_1
export LC_CTYPE
```

Estas configurações permitem a utilização das letras acentuadas e os menus em italiano em alguns programas (por exemplo o mutt).

5.8. Encerrando e reinicializando o sistema

O comando a usar é **shutdown**. Por exemplo:

```
# shutdown -h now
# shutdown -r now
```

Como alternativa podem-se usar os comandos:

```
# halt
# reboot
```

halt/reboot e **shutdown** não são sinônimos. Este último permite programar o encerramento do sistema, notificar os usuários, etc. Para os detalhes, consultar a página do manual respectiva.

Capítulo 6. Impressão

Neste capítulo é descrita uma configuração simples para uma impressora utilizando, a título de exemplo, uma HP Deskjet 690C conectada à primeira porta paralela. Em um primeiro momento é configurada a impressora de documentos em modo texto. Depois, a impressão de documentos PostScript com o programa Ghostscript.

6.1. Ativando o dæmon de impressão

Ao fim da instalação ainda não será possível imprimir porque o dæmon de impressão não está habilitado. Para ativá-lo é suficiente editar o arquivo `/etc/rc.conf`, mudando a linha

```
lpd=NO
```

para

```
lpd=YES
```

Para verificar se **lpd** está ativo, executar o seguinte comando:

```
# ps ax | grep lpd
  179 ??  Is     0:00.01 lpd
```

Se o comando precedente não produz um output, quer dizer que o **lpd** não está ativo. Nesse caso, pode-se ativá-lo manualmente com o seguinte comando:

```
# lpd -s
```

Efetuemos agora uma prova de impressão, usando o programa **lptest**.

```
# lptest 20 10 > /dev/lpt0
```

O output dever-se-ia apresentar com alinhamento em colunas. Se ao invés disso obtém-se um efeito de “escadinhas”, provavelmente a impressora está configurada para utilizar um caracter <CR> (carriage return, ASCII 13) para voltar ao início da linha, e um sucessivo caracter <LF> (line feed, ASCII 10) para avançar uma linha, enquanto o NetBSD (e o Unix em geral) usa somente o caracter <LF>. Pode-se resolver este problema:

- mudando a configuração da impressora
- utilizando um simples filtro de output (ver abaixo)

> **Nota:** no exemplo precedente, na realidade, a spooler (lançadeira) lpd não é utilizada porque o output do lptest foi enviado diretamente ao driver da impressora (`/dev/lpt0`).

6.2. Configurando o `/etc/printcap`

Nesta seção é explicado como configurar a impressora que usamos como exemplo para a impressão de documentos de texto.

Em primeiro lugar é necessário modificar o arquivo `/etc/printcap`, criando uma descrição para a impressora usando um identificador (o nome da impressora) padrão *lp*.

Exemplo 6-1. `/etc/printcap`

```
lp|local printer|HP DeskJet 690C:\
        :lp=/dev/lpa0:sd=/var/spool/lpd/lp:lf=/var/log/lpd-errs:\
        :sh:pl#66:pw#80:if=/usr/local/libexec/lpfilter:
```

O formato do arquivo e as opções estão descritas em detalhe em printcap(5). A única coisa a assinalar é que, para resolver o problema das “escadinhas”, foi inserido o indicador para um filtro com a instrução:

```
if=/usr/local/libexec/lpfilter
```

> **Driver de impressão:** no Exemplo 6-1 é usado o dispositivo *lpa#*, que é o driver “polled” da impressora no lugar de *lpd#*, que é o driver padrão para interrupção administrada (interrupt driven driver). Algumas impressoras não administram corretamente as interrupções, com a conseqüência de que levam horas para imprimir uma página. Com esse tipo de impressora (e a HP Deskjet 690C é uma delas) basta usar o driver polled (note-se que também é possível compilar um kernel indicando que lpd deve funcionar em modo polled).

Depois é necessário criar os diretórios de spool especificados com as opções `sd`, que serão usadas pelo **lpd** para acumular os dados.

```
# cd /var/spool/lpd
# mkdir lp
# chown daemon:daemon lp
# chmod 770 lp
```

Nesse ponto é necessário criar o filtro de impressão `lpfilter`. A sua única função é configurar a impressora, eliminando o já citado problema das escadinhas, antes de enviar o texto para impressão. Para a Deskjet 690C a seqüência para enviar é "ESC&k2G". Depois de ter criado o filtro é preciso fazê-lo funcionar.

Exemplo 6-2. `/usr/local/libexec/lpfilter`

```
#!/bin/sh
# Treat LF as CR+LF
printf "\033&k2G" && cat && exit 0
exit 2
```

```
# cd /usr/local/libexec
# chmod 755 lpfilter*
```

> **Nota:** um outro filtro que se pode utilizar é
>
> ```
> :if=/usr/libexec/lpr/lpf:
> ```
>
> cujas fontes encontram-se em `/usr/src/usr.sbin/lpr/filters`. Com respeito ao exemplo apresentado acima, trata-se de um filtro muito mais elaborado, concebido para produzir o output do **nroff** que, além de converter LF em CR + LF, controla subscrito e sobrescrito, expandindo os caracteres de tabulação em colunas múltiplas de 8.

Agora pode-se tentar executar novamente o programa **lptest**. Dessa vez, utilizando-se da spooler **lpd**.

```
# lptest 20 10 | lpr -h
```

lpr é um programa de impressão que utiliza a spooler para enviar os dados para a impressora (a opção `-h` serve para eliminar a impressão da página de cabeçalho).

> **Nota:** a opção `-h` não é necessária se no `/etc/printcap` foi especificada a opção `sh`.

Se tudo funcionou bem, as escadinhas foram eliminadas. Este não é o único método para eliminar o problema das escadinhas. Podem-se utilizar muitos tipos de filtros, inclusive programas escritos em C. Mas esta solução tem a vantagem de ser muito simples.

6.3. Configurando o Ghostscript

Agora que a impressão básica funciona, pode-se pensar em configurar a impressão de arquivos em formato PostScript. Uma vez que a impressora utilizada no exemplo não suporta nativamente a impressão de arquivos PostScript, é necessário usar um programa que converta o PostScript para um formato compreensível para a impressora. Este programa é o Ghostscript, que se pode instalar a partir da coleção de pacotes (ver Capítulo 8). Nesta seção será mostrado um exemplo de configuração para o uso do Ghostscript com a impressora HP Deskjet 690C. No exemplo é criado um segundo identificador para a impressora. Esse novo nome utiliza um filtro diferente, capaz de invocar o Ghostscript para a impressão dos arquivos PostScript. Portanto, os documentos de texto serão impressos na impressora *lp*, enquanto os documentos PostScript se-lo-ão na *lps*.

Como em muitas outras situações, a solução não é a única possível. Por exemplo, poder-se-ia usar um único filtro de impressão capaz de reconhecer o tipo de arquivo que está sendo impresso e de agir de acordo com isso. Com esta abordagem basta definir uma só impressora, um só diretório de spool e um único filtro, ao preço de uma maior complexidade do próprio filtro. Enfim, existem programas capazes de instalar filtros “inteligentes” e de simplificar muito a fase de configuração. Veja-se, por exemplo, o magicfilter na coleção de pacotes.

O arquivo `/etc/printcap` é modificado assim:

Exemplo 6-3. `/etc/printcap`

```
lp|local printer|HP DeskJet 690C:\
        :lp=/dev/lpa0:sd=/var/spool/lpd/lp:lf=/var/log/lpd-errs:\
        :sh:pl#66:pw#80:if=/usr/local/libexec/lpfilter:
```

```
ps|Ghostscript driver:\
        :lp=/dev/lpa0:sd=/var/spool/lpd/ps:lf=/var/log/lpd-errs:\
        :mx#0:sh:if=/usr/local/libexec/lpfilter-ps:
```

Note-se, em particular, a opção `mx#0`, que elimina as restrições de tamanho dos arquivos de output. Esta opção é necessária porque os arquivos PostScript são habitualmente de notáveis dimensões. Ademais, a opção `if` faz referência ao novo filtro, que se chama `lpfilter-ps`.

Agora podemos utilizar dois nomes de impressora no sistema: lp e ps que, na realidade, fazem referência à mesma impressora física mas utilizam filtros e diretórios diferentes. Para imprimir um arquivo Postscript, naturalmente, não basta definir uma expressão no `/etc/printcap`. É necessário também instalar o programa Ghostscript.

Ainda precisam ser criados o diretório da spool e o filtro.

```
# cd /var/spool/lpd
# mkdir ps
# chown daemon:daemon ps
# chmod 770 ps
```

O filtro para a impressão em formato PostScript é mais elaborado. O arquivo a ser impresso alimenta o intérprete que o transforma em uma seqüência de comandos adaptada à impressora. Desse modo transforma-se uma impressora econômica de jato de tinta em uma potente impressora PostScript. Para uma descrição dos comandos do filtro veja-se a documentação do Ghostscript. A chave do processo de impressão é a escolha do driver da impressora usado pelo Ghostscript, que no nosso caso é a `cdj550`.

Exemplo 6-4. `/usr/local/libexec/lpfilter-ps`

```
#!/bin/sh
# Treat LF as CR+LF
printf "\033&k2G" || exit 2
# Print the postscript file
/usr/pkg/bin/gs -dSAFER -dBATCH -dQUIET -dNOPAUSE -q -sDEVICE=cdj550 \
-sOutputFile=- -sPAPERSIZE=a4 - && exit 0
exit 2
```

6.4. Comandos úteis para a impressão

Nesta seção estão apresentados alguns comandos úteis para a gerenciamento da impressão típicos do ambiente BSD. Além dos já citados **lpr** e **lpd**, temos:

lpq

examina a fila dos trabalhos de impressão.

lprm

elimina trabalhos de impressão da fila.

lpc

controla o sistema de impressão, permitindo habilitar/desabilitar as impressoras e suas propriedades.

6.5. Impressão remota

É possível configurar o sistema de impressão de tal modo que possamos imprimir em uma impressora ligada a um host remoto. Suponhamos, por exemplo, trabalhar no host *wotan* e querer imprimir na impressora conectada ao host *loge*. Seja o arquivo `/etc/printcap` de loge aquele do Exemplo 6-3. A partir de wotan, portanto, poderemos imprimir até arquivos PostScript, utilizando o Ghostscript instalado em loge.

A primeira operação a cumprir é habilitar em loge as ordens de impressão provenientes do host wotan. Para fazer isso é necessário inserir o nome do host wotan no arquivo `/etc/hosts.lpd` de loge. O formato do arquivo é muito simples: em cada linha aparece o nome de um host a habilitar.

Depois é necessário configurar `/etc/printcap` em wotan, de modo a enviar as ordens de impressão para loge. Por exemplo:

```
lp|line printer on loge:\
 :lp=:sd=/var/spool/lpd/lp:lf=/var/log/lp-errs:\
 :rm=loge:rp=lp

ps|Ghostscript driver on loge:\
 :lp=:sd=/var/spool/lpd/lp:lf=/var/log/lp-errs:\
 :mx#0:\
 :rm=loge:rp=ps
```

Há quatro diferenças entre estas configurações e as do Exemplo 6-3.

1. A definição de "lp" está vazia.
2. A expressão "rm" define o nome do host a que está ligada a impressora.
3. A expressão "rp" especifica o nome da impressora remota no host remoto.
4. Não é necessário especificar filtros de input porque são utilizados os filtros definidos em loge.

Uma vez efetuadas as operações precedentes, os trabalhos de impressão de wotan são enviados automaticamente ao host loge.

Capítulo 7. Compilando o kernel

A maior parte dos usuários do NetBSD termina, cedo ou tarde, por compilar um kernel personalizado. Desse modo, obtêm-se vários resultados apreciáveis.

- reduzem-se consideravelmente as dimensões do kernel (por ex. de 2.5 MB para 1MB) e, portanto, o uso de memória. Na versão 1.5 do NetBSD, criando um kernel personalizado, pode-se passar de 4.7 MB a menos de 2 MB.
- melhoram-se as performances.
- otimiza-se o sistema.
- são resolvidos os problemas de conflitos/reconhecimentos de periféricos.
- aprofunda-se o conhecimento do sistema.
- personalizam-se algumas opções (por exemplo o tipo de teclado, o offset do relógio da BIOS, etc.)

7.1. Instalando o código-fonte do sistema

> Se você não tem o arquivo do código-fonte do sistema operacional, pode obtê-lo por download no mesmo lugar de origem em que obteve o sistema instalado. Por exemplo: ftp://ftp.netbsd.org. É importante fazer o download do código-fonte correspondente à versão do sistema instalado!

O programa de instalação (sysinst) não instala as fontes do kernel que devem, portanto, ser extraídas a mão. O arquivo que as contém é `syssrc.tgz`, que se encontra no diretório `source/sets`. O arquivo se descompacta com

```
# gzip -dc syssrc.tgz | (cd / ; tar xvf -)
```

É preciso ter um pouco de paciência porque os arquivos a extrair são muitos e a operação dura alguns minutos. As fontes são extraídas no diretório `/usr/src/sys`. Para acessá-lo pode-se inclusive usar o *link simbólico* `/sys`. Portanto os comandos

```
# cd /usr/src/sys
# cd /sys
```

são equivalentes.

Terminada a instalação é possível remover as fontes relativas às arquiteturas que não nos interessam e poupar um pouco de espaço em disco (se isso for problema). Para fazer isso é necessário entrar no diretório `/sys/arch` e eliminar os subdiretórios desnecessários. Por exemplo, para a arquitetura i386 é suficiente conservar o diretório `i386`. Para outras versões de hardware do NetBSD é necessário conservar mais de um diretório de fontes, já que podem conter dependências.

Tudo está pronto para a criação de um kernel personalizado. Compilar um novo kernel não tem nada de difícil. As operações a executar (quatro ou cinco ao todo) são descritas no site do NetBSD.

7.2. Modificando a disposição do teclado

Uma vez instaladas as fontes do kernel, antes de recompilar convém efetuar uma pequena modificação no layout do teclado italiano contido no arquivo `/sys/dev/pckbc/wskbdmap_mfii.c`. Na disposição por default faltam alguns caracteres úteis para os programadores, em particular parênteses, haspas e til. Eu utilizo estes recursos.

```
static const keysym_t pckbd_keydesc_it[] = {
...
KC(8),   KS_7,          KS_slash,       KS_braceleft,
KC(9),   KS_8,          KS_parenleft,   KS_bracketleft,
KC(10),  KS_9,          KS_parenright,  KS_bracketright,
KC(11),  KS_0,          KS_equal,       KS_braceright,
KC(12),  KS_apostrophe, KS_question,    KS_grave,
KC(13),  KS_igrave,     KS_asciicircum, KS_asciitilde,
KC(26),  KS_egrave,     KS_eacute,      KS_bracketleft, KS_braceleft,
KC(27),  KS_plus,       KS_asterisk,    KS_bracketright,KS_braceright,
...
```

Com o código precedente obtêm-se estas correspondências

Combinação	Caracter
Alt Gr + 7	{
Alt Gr + 8	[
Alt Gr + 9	]
Alt Gr + 0	}
Alt Gr + ’	‘
Alt Gr + ì	~
Alt Gr + é	[
Alt Gr + +	]
Shift + Alt Gr + è	{
Shift + Alt Gr + +	}

Driver de console: desde a versão 1.4 o NetBSD utiliza o driver multiplataforma wscons para o gerenciamento da tela e do teclado. As versões precedentes utilizavam pccons ou pcvt. Para maiores detalhes ver o Capítulo 13.

7.3. Recompilando o kernel

A recompilação do kernel é descrita de modo simples e preciso no site do NetBSD. Assim, sugiro que se faça referência àquelas instruções. Todavia, a seguir reporto-me a algumas indicações apenas para dar uma idéia do procedimento.

Para poder recompilar o kernel é necessário ter instalado o set do compilador (`comp.tgz`). A recompilação do kernel consiste dos seguintes passos:

Compilação do kernel

1. Modificar o arquivo de configuração do kernel
2. Configurar o kernel
3. Gerar as dependências
4. Recompilar
5. Instalar o novo kernel

7.4. Modificar o arquivo de configuração do kernel

O arquivo de configuração do kernel define o tipo, o número e as características dos dispositivos que devem ser suportados pelo kernel. O arquivo encontra-se no diretório `/sys/arch/i386/conf`. Para criar um arquivo personalizado convém utilizar como base uma das configurações existentes. Para a maior parte das plataformas a configuração GENERIC é um bom ponto de partida: remeta-se aos exemplos presentes no diretório `/sys/arch/<arch>/conf`. Muitas opções estão descritas nos comentários dos arquivos de configuração. Muitas outras estão descritas na página de manual options(4).

```
# cd /sys/arch/i386/conf/
# cp GENERIC MYKERNEL
# vi MYKERNEL
```

Nomes: tradicionalmente os nomes utilizados para os arquivos de configuração estão em maiúsculas.

Quando se modifica um arquivo de configuração do kernel, são realizados basicamente dois tipos de operação.

1. desabilita-se o suporte para os periféricos não presentes no sistema e se habilita o suporte para aqueles que estão presentes (por exemplo, quem não tem dispositivos SCSI pode desabilitar o suporte para a interface SCSI no kernel).
2. habilita-se e se desabilita o suporte das características suportadas pelo kernel (por exemplo, o suporte para a compatibilidade com o Linux, etc.).
3. otimizam-se os parâmetros do kernel.

Para desabilitar uma linha do arquivo de configuração é necessário acrescentar um caracter "#" no início dessa mesma linha. Um bom ponto de partida para determinar o que se pode eliminar é o estudo do output do comando **dmesg** para cada linha do tipo

```
<XXX> at <YYY>
```

É necessário que no arquivo de configuração estejam presentes tanto XXX como YYY. Antes de chegar a uma configuração mínima, um pouco de experimentação é necessária. Para um sistema de desktop, sem periféricos SCSI e placas PCMCIA, pode-se chegar a produzir um kernel de dimensões bem menores que o genérico.

Também é bom dar uma olhada nas opções presentes no arquivo de configuração, desabilitando aquelas que não nos interessam. Cada opção é acompanhada de uma breve descrição. Muitas opções estão descritas mais aprofundadamente em options(4). Antes de recompilar convém definir as opções para o teclado italiano e para o fuso horário. (Há necessidade de desenvolvimento da opção para o teclado br-abnt2 e us com acentos (nota do tradutor).)

```
options RTC_OFFSET=-60
...
options PCKBD_LAYOUT="KB_IT"
```

Existe um script Perl que gera automaticamente um arquivo de configuração reduzido, analisando o output do **dmesg**. Esse script pode ser encontrado na URL http://www.feyrer.de/Misc/adjustkernel. Para poder executá-lo, naturalmente, é necessário ter instalado o Perl no sistema, o que é bem fácil de se fazer. Ver o Capítulo 8. A título de antecipação, um modo muito simples de instalar o Perl consiste em obtermos o pacote pré-compilado `perl-5.00404.tgz` e depois executar o comando:

```
# pkg_add perl-5.00404.tgz
```

Neste ponto o Perl já está instalado, configurado e pronto para funcionar. Mais fácil que isso...

O script se executa com:

```
# cd /sys/arch/i386/conf
# perl adjustkernel GENERIC > MYKERNEL
```

Pessoalmente usei esse script apenas uma vez e deu ótimos resultados. Preste-se atenção, contudo, no fato de que o script configura as linhas relativas aos dispositivos e não as das opções.

7.5. Configurando o kernel

Depois de ter modificado à vontade o arquivo de configuração do kernel (que estaremos supondo chamar-se `MYKERNEL`), dar o comando

```
# config MYKERNEL
```

Se o `MYKERNEL` não contém erros, o programa criará os arquivos necessários para poder compilar o kernel. Caso contrário deve-se voltar ao ponto anterior e corrigir os erros. O programa **config** analisa o arquivo de configuração indicado como parâmetro e cria os arquivos necessários para a compilação, habilitando os dispositivos selecionados.

7.6. Gerar as dependências e recompilar

Deslocar-se para o diretório criado pelo config no passo anterior, executar o comando **make depend** e, enfim, o comando **make**.

```
# cd ../compile/MYKERNEL
```

```
# make depend
# make
```

Na compilação podem-se verificar erros, caso em que ela própria se interrompe. Se isso ocorre, significa que o arquivo de configuração não foi modificado corretamente e é necessário corrigí-lo e repetir o procedimento. Um exemplo típico é o seguinte: a opção B, que exige a opção A, está ativa; a opção A, todavia, está comentada.

Uma compilação completa pode durar de poucos minutos a muitas horas, de acordo com o hardware utilizado. A tabela seguinte contém alguns exemplos:

CPU	**RAM (MB)**	**Tempo aprox.**
486 DX2 50	20	1 hora
P166	96	15 minutos
PIII	128	5 minutos
68030/25	8	4 horas

Ao término da compilação será criado no diretório corrente o arquivo `netbsd`. O último passo a ser dado é a cópia do novo kernel em `/` (não sem antes renomear o kernel precedente).

```
# mv /netbsd /netbsd.old
# mv netbsd /
```

Como já dito, personalizando o kernel podêmo-lo reduzir consideravelmente de tamanho. Por exemplo, ao término das operações precedentes se tem:

```
-rwxr-xr-x  1 root  wheel  1342567 Nov 13 16:47 /netbsd
-rwxr-xr-x  1 root  wheel  3111739 Sep 27 01:20 /netbsd.old
```

Neste ponto não resta senão reinicializar o sistema com o comando **reboot**.

7.7. Se alguma coisa deu errado

É sempre possível que, ao reinicializirmos, alguma coisa não corra como o esperado. No limite, o novo kernel poderia até mesmo não conseguir reinicializar. Neste caso não precisa se desesperar. Basta inicializar com o kernel precedente, que tinhamos providencialmente salvado, em modo monousuário, da seguinte maneira:

- Reinicializar a máquina
- Pressionar a barra de espaço quando aparecer o prompt do boot, durante a contagem regressiva de 5 segundos

  ```
  boot:
  ```

- Executar o comando

  ```
  > boot netbsd.old -s
  ```

- Executar os seguintes comandos

  ```
  fsck /
  mount /
  mv netbsd.old netbsd
  exit
  ```

Capítulo 8. Os pacotes

O sistema de pacotes do NetBSD é um conjunto de instrumentos que permitem compilar e instalar de maneira simples uma grande quantidade de softwares de domínio público para os sistemas Unix. Em geral bastam um ou dois comandos para instalar qualquer pacote configurado e pronto para funcionar.

O primeiro contato com os pacotes gera um pouco de confusão porque aparentemente há vários comandos que fazem a mesma coisa. Na realidade a questão é muito simples. Há *dois modos* de se instalar um programa.

- Compilar as fontes do programa em nosso sistema. Esta é a tarefa da coleção de pacotes que é capaz de fazer o download automático das fontes, compilá-las e instalar e configurar o programa e a documentação com um ou dois comandos. Como se não bastasse, ainda é capaz de determinar automaticamente as dependências e instalar os pacotes necessários. O sistema de pacotes não é instalado no sistema de base pela razão de que sofre freqüentes atualizações, para manter-se de acordo com as novas versões dos programas a instalar. O sistema dos pacotes é descrito de modo completo e exaustivo no site do NetBSD.
- Instalar uma versão pré-compilada e pré-configurada do programa em um formato adequado. É essa a tarefa dos “pkgtools”, uma série de programas instalados com a base do sistema. Este método é mais veloz mas menos flexível que o precedente. Os pkgtools também estão em condição de reconhecer e gerir (listar, desinstalar, etc.) os programas instalados com o sistema dos pacotes. Trata-se, por assim dizer, de ferramentas complementares. Se decidimos instalar exclusivamente programas em forma pré-compilada, não é necessário instalar a coleção de pacotes.

Os dois métodos ilustrados acima são os mais "cômodos", que aproveitam o trabalho de configuração feito por alguém. Se o programa que nos interessa não está incluído na coleção dos pacote, ninguém nos impede de compilá-lo a mão e instalá-lo no sistema. Nesse caso, é boa norma fazer um pequeno esforço a mais e criarmos nós mesmos os pacotes, de modo que outros possam beneficiar-se do nosso trabalho.

8.1. Instalação da coleção de pacotes

Antes de poder instalar um pacote é necessário instalar a coleção de pacotes, como se descreve em detalhe no site do NetBSD. Em substância, trata-se de:

1. Fazer o download da última versão das fontes da coleção, que incluem todos os arquivos de configuração, de `ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/tar_files/`. O nome do arquivo a obter é: `pkgsrc.tar.gz`.
2. Remover a coleção anterior, se presente, com

   ```
   # cd /usr
   # rm -rf pkgsrc
   ```

3. Instalar a nova coleção com

   ```
   # tar -xzvpf pkgsrc.tar.gz -C /usr
   ```

A execução do comando acima dura um certo tempo porque o número dos arquivos a extrair é notável. Ao fim da operação o *framework* para a instalação dos pacote está em seu lugar. Fica para decidir apenas quais pacotes instalar no sistema.

Nota: imagino que neste ponto esteja claro que com a operação anterior foi instalado no sistema o conjunto dos arquivos de configuração necessários para a instalação individual dos programas, mas não os próprios programas nem as suas fontes. Basicamente agora o sistema dispõe da lista dos programas instaláveis automaticamente e as instruções para instalá-los.

Uma vez instalada a coleção dos pacotes, com um browser de HTML, como o Lynx ou o Netscape, podem-se ler os detalhes e as descrições de todos os pacotes disponíveis. Por exemplo:

```
$ cd /usr/pkgsrc
$ lynx README.html
```

SidebarPara não perder os distfiles

Quando se instala uma nova coleção de pacotes é necessário remover a coleção precedente, que geralmente se encontra em `/usr/pkgsrc`, o que comporta a perda do subdiretório `distfiles` se nós nos esquecemos de salvar o seu conteúdo. O diretório `distfiles` é o que contém todos os pacotes descarregados da Internet e, portanto, pode ser um aborrecimento perdê-lo. O problema pode ser resolvido pondo-se o diretório `distfiles` em um outro lugar (fora do diretório `/usr/pkgsrc`).

```
# mkdir /usr/pkgsrc_distfiles
```

e depois acrescentar ao arquivo `/etc/mk.conf` a seguinte linha

```
DISTDIR=/usr/pkgsrc_distfiles
```

Obviamente pode-se usar um diretório diferente do que foi proposto no exemplo, `/usr/pkgsrc_distfiles`.

No arquivo `/etc/mk.conf` podemos configurar muitos aspectos do comportamento do sistema de pacotes. Para uma descrição de todas as opções, veja-se o arquivo `/usr/pkgsrc/mk/mk.conf.example`.

8.2. Atualização da coleção de pacotes

A coleção dos pacotes é atualizada muito freqüentemente. No site de ftp encontra-se uma nova cópia quase toda semana. Para atualizar a coleção no próprio computador é necessário seguir as instruções dadas para a instalação.

Ás vezes, além da coleção dos pacotes é necessário atualizar também os programas de instalação (pkgtools). É fácil compreender se está sendo necessário realizar a atualização: quando se tenta instalar um pacote o sistema de pacotes avisa que as pkgtools devem ser atualizadas. Por exemplo:

```
# make
===> Validating dependencies for gqmpeg-0.6.3
Your package tools need to be updated to 2000/02/02 versions.
The installed package tools were last updated on 1999/01/01.
Please make and install the pkgsrc/pkgtools/pkg_install package.
*** Error code 1
```

O método de atualização mais simples é:

```
# cd /usr/pkgsrc/pkgtools/pkg_install
# make install
```

Feito isso, pode-se retomar a instalação do pacote original (aquele que tinha dado lugar à mensagem de erro.)

> **Nota:** tecnicamente a mensagem de erro indica que a versão dos programas de instalação está contida em `pkg_install-20000202.tar.gz`, que se pode encontrar no site de ftp em `packages/distfiles/LOCAL_PORTS`. O pkg_install instala-se como qualquer outro pacote (`pkgsrc/pkgtools/pkg_install`).
>
> Para determinar exatamente a versão exigida, examinar o arquivo `pkgsrc/mk/bsd.pkg.mk` e procurar a linha com
>
> ```
> PKGTOOLS_REQD = 20000202
> ```
>
> (a data 20000202 é só um exemplo).

8.3. Instalação de um programa

A título de exemplo, tentemos instalar o programa addnerd, que serve para adicionar usuários ao sistema de maneira automática (mais ou menos como faz o programa adduser em outros sistemas). A primeira coisa é irmos para o diretório `/usr/pkgsrc/sysutils/addnerd`.

8.3.1. Fazendo o download das fontes

Se dispomos de uma conexão à Internet o Makefile está preparado para fazer automaticamente o download do pacote, e podemos pular para a seção seguinte.

SidebarFazendo o download em outra máquina

Um cenário muito comum é aquele em que não se dispõe de uma conexão veloz à Internet, como por exemplo no computador que temos em casa. Nesse caso pode ser conveniente descarregar as fontes em uma outra máquina que dispõe de uma conexão melhor (por exemplo, no trabalho).

Em caso contrário é preciso obter o pacote fonte e copiá-lo no diretório `/usr/pkgsrc/distfiles`. Examinando o Makefile, encontra-se o nome do pacote a obter (DISTNAME = addnerd-1.6). O nome completo do pacote é `addnerd-1.6.tar.gz`.

O mesmo resultado pode-se obter de modo mais simples com os comandos:

```
# cd /usr/pkgsrc/sysutils/addnerd
# make fetch-list
```

que, ademais, também mostram os endereços dos sites dos quais pode-se fazer o download do programa.

8.3.2. Compilando as fontes

Para compilar o pacote escrever

```
# cd /usr/pkgsrc/sysutils/addnerd
# make
```

O comando precedente descarrega o pacote da Internet (se ele ainda não estiver presente no diretório dos distfiles), extrai as fontes, aplica os patches necessários para poder compilá-lo sob o NetBSD e, por fim, compila o pacote.

Para instalá-lo

```
# make install
```

A instalação do pacote é registrada pelo sistema. Pode-se verificar escrevendo **pkg_info -a**. Com a instalação completada, pode-se fazer um pouco de limpeza com

```
# make clean
# make clean-depends
```

O segundo comando é mostrado somente a título de exemplo. Nesse caso não estavam instalados pacotes dependentes e, portanto, não era necessário fazer a faxina.

8.4. Instalação de um pacote em formato binário

Como expliquei na introdução, o sistema dos pacotes foi concebido para instalar programas na forma do código-fonte. Entretanto, ele também é habilitado a instalar pacotes pré-compilados, gerados por qualquer um utilizando a própria coleção. Nesse caso, a instalação é mais veloz (não é necessário compilar o programa a instalar).

Os programas pré-compilados distinguem-se geralmente daqueles que estão na forma de fonte por terem a extensão `.tgz` ao invés de `.tar.gz`.

Nota: na realidade a extensão dos tarballs de fontes não é sempre `.tar.gz`. O sistema de pacotes, todavia, é capaz de gerir também extensões diferentes como (`.zip`, `.bz2`, etc.).

Não é estritamente necessário fazer manualmente o download de um pacote binário para poder instalá-lo. O pkg_add também é capaz de fazer-lhe o download autonomamente, quando são utilizados URL FTP (o uso de HTTP é possível mas desaconselhado). Por exemplo, pode-se especificar o nome do pacote com:

```
ftp://ftp.netbsd.org/pub/NetBSD/packages/1.4.2/i386/All/tcsh-6.09.00.tgz
```

Se não se conhece o número de versão do pacote disponível no site de FTP, também é possível omitir esta informação da linha de comando e o **pkg_add** utilizará automaticamente a última versão disponível no servidor de FTP. Por exemplo:

```
# pkg_add ftp://ftp.netbsd.org/pub/NetBSD/packages/1.4.2/i386/All/tcsh
```

É possível também configurar a variável ambiental PKG_PATH com uma lista de caminhos e URLs separados por *;* e omitir o caminho (path) no comando pkg_add:

```
# PKG_PATH="/cdrom;/usr/pkgsrc/packages/All;ftp://ftp.netbsd.org/pub/NetBSD/packages/1.4.2/i
export PKG_PATH
# pkg_add tcsh
```

O comando precedente instala o primeiro pacote binário de tcsh encontrado nos caminhos especificados.

A título de exemplo, vejamos como procede a instalação do programa texinfo em forma pré-compilada. A instalação do pacote binário é muito simples:

1. Copiar o `gtexinfo-3.12.tgz` em um diretório temporário.
2. Executar o comando

   ```
   # pkg_add -v gtexinfo-3.12.tgz
   ```

3. Aferir a instalação feita com o comando

   ```
   # pkg_info
   ```

4. Remover o arquivo `gtexinfo-3.12.tgz` do diretório temporário.

Os pacotes pré-compilados são muito cômodos de usar porque exigem tempo e esforço mínimos para a instalação. Todavia, os pacotes em forma de código-fonte são mais flexíveis ainda porque permitem a personalização de eventuais opções de configuração e de instalação. A instalação é mais longa mas, em todo caso, não se instalam pacotes a toda hora...

Quando se instala um pacote binário com o comando **pkg_add**, convém como primeira coisa a fazer examiná-lo com o comando **pkg_info**. Por exemplo:

```
# pkg_info -f jpeg-6b.tgz
```

Em particular convém observar o primeiro comando CWD para ver onde será instalado o pacote. Se o diretório base não é o desejado, será necessário usar a opção `-p` do **pkg_add**. Por exemplo, o pacote `jpeg-6b.tgz` vai ser instalado em `/usr/pkg`, enquanto em minha opinião, deveria ir para `/usr/X11R6`. (Em geral convém extrair em `/usr/X11R6` todos os pacotes da hierarquia X). Portanto é preferível extraí-lo com:

```
# pkg_add -p /usr/X11R6 -v jpeg-6b.tgz
```

8.5. Comandos para a gestão de pacotes

Os principais comandos para a administração do sistema de pacotes são

pkg_add

Utilitário para a instalação de pacotes pré-compilados.

pkg_delete

Utilitário para a desinstalação de pacotes pré-compilados. Os nomes dos pacotes podem ser especificados com ou sem o número de versão. O pkg_delete determina automaticamente o número de versão do pacote instalado. Podem ser utilizados caracteres curinga à condição de serem destacados do interpretador de comandos (a shell). Por exemplo:

```
# pkg_delete "*emacs*"
```

A opção `-r` permite a remoção recursiva dos pacotes. Primeiro remove todos os pacotes que exigem o indicado e depois remove o próprio pacote. Por exemplo:

```
# pkg_delete -r jpeg
```

remove o pacote jpeg e todos os que o utilizam. Pode ser útil para efetuar a atualização de um pacote.

pkg_info

Utilitário para a visualização de informações sobre pacotes de software instalados e a instalar.

pkg_create

Utilitário para a criação de pacotes de software. Este é o programa usado para a criação de pacotes pré-compilados. É chamado automaticamente pelo sistema de pacotes e jamais é necessário executá-lo manualmente.

pkg_admin

Programa utilitário para a execução de várias funções administrativas no sistema de pacotes.

8.6. Guia rápido para a criação de pacotes

O autor desta seção (Guia rápido para a criação de pacotes) é Jason R. Fink

Esta seção descreve um método simples e veloz para a criação de pacotes simples para a coleção de pacotes do NetBSD. Para ulteriores aprofundamentos remete-se à documentação oficial contida no documento `Packages.txt`.

8.6.1. Ferramentas

Os principais instrumentos a utilizar para a construção de um simples pacote a ser acrescentado à coleção dos pacotes do NetBSD são três:

- url2pkg
- um protótipo de pacote
- pkglint

8.6.1.1. url2pkg

O programa utilitário url2pkg pode ser instalado da coleção de pacotes. Trata-se de uma ferramenta que permite ao construtor de pacotes preparar e verificar velozmente as estruturas de base da construção de um pacote.

8.6.1.2. Protótipo de pacote

A escolha do protótipo é deixada ao desenvolvedor. `Packages.txt` mostra um exemplo de construção de pacotes na seção 11 'A simple example of package: bison'. Note-se que muitos dos pacotes da coleção não têm nada de complicados.

8.6.1.3. pkglint

O utilitário pkglint pode ser instalado a partir da coleção de pacotes. Trata-se de um instrumento para se aferir a correção dos pacotes.

8.6.2. Primeiros passos

Os passos iniciais são muito simples, contanto que se tenha o discernimento de verificar se o programa se compila corretamente no NetBSD a partir das fontes. Caso contrário a construção do pacote poderia complicar-se notavelmente. Note-se que usualmente é possível gerar "patches" para aplicar às fontes e incluir no pacote para resolver os problemas de compilação (ou, mais em geral, de compatibilidade), mas esse tópico ultrapassa os objetivos desta simples introdução.

8.6.2.1. Uso do url2pkg

A primeira coisa a fazer é utilizar o url2pkg.

O processo seguinte detalha os passos a seguir para os arquivos de um novo pacote.

1. Criar um novo diretório para o novo pacote, escolhendo uma posição apropriada em `pkgsrc`. Deixar o novo diretório inicialmente vazio.
2. Entrar no novo diretório.
3. Executar o comando

   ```
   $ url2pkg
   ```

4. Nesse ponto o programa pede para inserir uma URL: inserir a URL solicitada e pressionar **Enter**.
5. Uma sessão do editor vi se inicia para criar o Makefile para o novo pacote. É necessário inserir o nome e a categoria a que pertence o pacote, além do endereço de e-mail da pessoa que cuidará da manutenção do próprio pacote.
6. Salvar o arquivo e sair.
7. url2pkg faz automaticamente o download do pacote e o deposita no subdiretório `work`
8. Enfim url2pkg gera o arquivo MD5.

Com isso fica concluída a seção do url2pkg. Note-se que o url2pkg não preenche nenhum arquivo à exceção do Makefile. Além disso gera um PLIST vazio.

8.6.3. Operações subseqüentes

Agora que o Makefile foi gerado, é necessário criar os arquivos restantes aproveitando um pacote para usar como protótipo. Escolher um pacote existente e copiar os seguinte arquivos do subdiretório `pkg`, modificando-os oportunamente.

DESCR

Uma descrição do novo pacote em poucas linhas.

COMMENTS

Uma descrição do pacote em uma linha. Não é necessário citar o nome do próprio pacote: ele será adicionado automaticamente pelas várias ferramentas pkg_* quando forem invocadas.

PLIST

Este arquivo descreve a posição dos arquivos a instalar no sistema. Para um pacote não muito complexo (por exemplo um executável e uma ou duas páginas de manual) deveria ser suficiente dar uma olhada no Makefile e nos scripts de instalação para compreender onde pôr os arquivos a instalar.

8.6.4. Verificação com o pkglint

Agora que todos os arquivos exigidos estão prontos, é chegado o momento de verificar a correção dos pacotes com o programa pkglint. Acontece freqüentemente que uma seção do Makefile deva ser deslocada, modificada ou acrescida de informações adicionais. É bom executar logo o pkglint para que se encontre de imediato aquilo que deverá ser mudado. Vejamos agora um exemplo de output do pkglint obtido no Packages.txt:

```
$ pkglint
OK: checking pkg/COMMENT.
OK: checking pkg/DESCR.
OK: checking Makefile.
OK: checking files/md5.
OK: checking patches/patch-aa.
looks fine.
```

Em caso de erro é geralmente simples levantar as causas. Eis um exemplo de mensagem de erro obtida durante a construção de um pacote:

```
extract suffix not required
```

A mensagem indica simplesmente que não era necessário definir no Makefile o sufixo para a extração.

8.6.5. Testes finais

Nesse ponto, ultrapassada a verificação do pkglint, é aconselhável efetuar um “ciclo” completo de fetch (download), compilação e instalação. Para um desenvolvimento correto é necessário antes deletar o subdiretório `work` e os distfiles já descarregados (que geralmente encontram-se em `/usr/pkgsrc/distfiles`). Desse modo garante-se que o teste efetuado será completo.

8.6.6. Submetendo um pacote ao send-pr

Em primeiro lugar é necessário criar um “tarball” (arquivo tar comprimido) da árvore de diretórios do próprio pacote (compreendido o subdiretório `work`). Por exemplo:

```
$ tar -czf packagename.tgz package_dir
```

Sucessivamente o arquivo deve ser copiado em um local acessível aos gerenciadores da coleção de pacotes do NetBSD. Se isso não é possível, convém contatar os responsáveis para perguntar se existem outros métodos que se podem usar para tornar o arquivo disponível.

O método preferido para informar os responsáveis pela coleção de pacotes do NetBSD é usar o utilitário send-pr, especificando a categoria “pkg”, incluindo o nome, a versão e a URL do pacote, além de uma breve descrição do mesmo.

Pode-se utilizar tanto a versão do send-pr incluída no NetBSD quanto a disponível online no endereço http://www.NetBSD.org/cgi-bin/sendpr.cgi?gndb=netbsd.

8.6.7. Notas conclusivas

Como já foi dito, este breve guia limita-se a considerar pacotes simples que instalam um número limitado de arquivos no sistema NetBSD, com a suposição adicional de que os próprios pacotes não tenham dependências de outros pacotes.

Para um aprofundamento posterior aconselha-se a leitura do já citado `Packages.txt`.

Capítulo 9. Introdução às redes TCP/IP

9.1. Redes TCP/IP

O autor deste capítulo (Introdução às redes TCP/IP) é Hubert Feyrer `<hubert@feyrer.de>`.

9.1.1. Introdução

O presente capítulo do guia é dedicado às redes, das quais explica os elementos fundamentais para os leitores que não têm uma grande familiaridade com o assunto. O capítulo seguinte, por sua vez, contém uma introdução prática às configurações de rede do NetBSD. Quem deseja configurar rapidamente a rede pode iniciar com o próximo capítulo e depois voltar vom calma à leitura do capítulo atual.

Da parte do leitor é pressuposto o conhecimento de elementos de base da administração do sistema: como tornar-se "root", como editar arquivos, como mudar as permissões dos arquivos, como encerrar processos, etc. Ver por exemplo [AeleenFrisch] para aprofundar estas questões. Além disso, é necessário saber utilizar alguns programas utilitários como telnet, ftp, etc. Tais programas não serão descritos aqui e, portanto, para informações adicionais remete-se às páginas de manual, à bibliografia e aos outros capítulos desse guia.

Esta introdução às redes TCP/IP não tem a pretensão de ser completa. Quem deseja aprofundar os assuntos tratados, para ter um quadro mais completo pode fazer referência a [CraigHunt]. Este livro, além de descrever os elementos de base, explica todos os conceitos e detalhes necessários para a configuração de todos os serviços. É um ótimo livro.

9.1.2. Protocolos de rede suportados

O NetBSD suporta vários protocolos de rede, a maior parte dos quais herdados de seu predecessor, o 4.4BSD, e posteriormente estendidos e melhorados. O primeiro protocolo implementado é o que hoje tem maior importância, o Transmission Control Protocol/Internet Protocol (TCP/IP) do DARPA. Outros protocolos disponíveis no NetBSD são o Xerox Network System (XNS), implementado para a UCB para coligar em rede máquinas isoladas, o AppleTalk da Apple, o ISO, o CCITT X.25 e o ARGO TP. Estes últimos protocolos são usados hoje apenas para algumas aplicações especiais.

Hoje o TCP/IP é o protocolo mais difundido entre os mencionados no parágrafo precedente. Além de ser implementado para todo sistema operacional e em todo tipo de hardware, é também o protocolo mais usado em ambientes heterogêneos. Portanto, o TCP/IP é a escolha mais justa para conectar seu próprio computador a outras máquinas domésticas ou a redes empresariais ou universitárias.

A versão corrente do protocolo TCP/IP é a 4 (*IPv4*). A versão 6 (*IPv6*) está em fase de desenvolvimento e o código IPv6 do projeto KAME está integrado nas edições do NetBSD já a partir da 1.5.

Também existem protocolos que não são suportados pelo NetBSD, como DECNET, IPX/SPX da Novell e NetBIOS da Microsoft. Estes protocolos, diferentemente dos precedentes, são proprietários e, portanto, não são definidos por algum RFC público como os outros padrões abertos.

9.1.3. Interfaces suportadas

O protocolo TCP/IP pode ser utilizado com uma ampla gama de interfaces. Entre as suportadas pelo NetBSD recordemos a Ethernet (10/100/1000MBd), a linha serial Arcnet, ATM, Fiber Channel, USB, HIPPI, FireWire (IEEE 1394), Token Ring, linhas seriais e ainda outras.

9.1.3.1. Linhas seriais

Há várias razões que nos levam a aconselhar o uso de TCP/IP em linhas seriais.

- se um host remoto é acessável apenas por telefone, pode-se contatá-lo por modem.
- praticamente todos os computadores dispõem de uma saída serial e o cabo necessário é muito barato.

A desvantagem das conexões por linha serial é que são mais lentas que os outros métodos. Nessa conexão o NetBSD pode utilizar no máximo 115300 bit/s, o que a torna muito mais lenta em comparação com os 10 Mbit/s da Ethernet e aos 4 Mbit/s do Arcnet.

Para conectar um host com NetBSD a um outro host por via serial, eventualmente também por linha telefônica, podem-se usar dois métodos:

- Serial Line IP (SLIP)
- Point to Point Protocol (PPP)

A escolha depende do tipo de conexão que se quer efetuar. Para uma conexão telefônica por modem, convém utilizar o PPP, que oferece a opção de auto-negociar os endereços IP e trajetos (routes), coisas complicadas de configurar manualmente. Ao contrário, para conectar-se diretamente a uma outra máquina, por exemplo com um cabo null-modem ou uma linha dedicada, o protocolo SLIP é mais simples de configurar com endereços e trajetos (routes), e é suportado praticamente por todos os sistemas operacionais.

PPP em uma linha direta é mais difícil de configurar se compararmos com SLIP por causa do handshake inicial, que no SLIP não está presente. Isto significa que com o SLIP inicia-se a conexão de um lado e o outro lado já pode enviar o primeiro pacote assim que estiver pronto.

[RFC1331] e [RFC1332] descrevem os protocolos PPP e TCP/IP em PPP. SLIP é descrito em [RFC1055].

9.1.3.2. Ethernet

Ethernet é a interface mais utilizada no âmbito das redes locais LAN (Local Area Network) de máquinas interconectadas em uma área limitada, que pode ser um escritório, uma empresa ou uma universidade. Ethernet baseia-se em um bus a que várias máquinas podem conectar-se e a comunicação ocorre sempre entre dois nódulos (nodes) por vez. Quando vários pares de nódulos tentam comunicar-se ao mesmo tempo, verifica-se um conflito e os pares procurarão comunicar-se de novo ao fim de um prazo (timeout). O termo técnico utilizado para esse procedimento é CSMA/CD (Carrier Sense w/ Multiple Access and Collision Detection).

Inicialmente o hardware Ethernet consistia em um grosso cabo (amarelo) que era ligado às máquinas usando conectores especiais que furavam a camada de proteção externa do próprio cabo. Posteriormente chegou o tipo 10base5, que usava conectores do tipo BNC para inserir em conectores especiais em T da própria máquina, com terminais em ambas as extremidades da linha. Hoje o bus ethernet está contido dentro dos comutadores (switches) e dos hubs, e os cabos são do tipo twisted pair, dando o nome a este tipo de interface: 10baseT para as redes a 10 Mbit/s e 100baseT para as redes a 100Mbit/s. Nas redes configuradas com switch distinguem-se os dois casos em que a comunicação entre nódulo e switch ocorre em modo full-duplex ou half-duplex.

9.1.4. O formato dos endereços TCP/IP

A atual implementação do protocolo TCP/IP (IPv4) usa os identificadores de 4 bytes (32 bits) (chamados também endereços de IP) para endereçar os hosts.

Estes endereços IP são únicos em nível mundial. Para atingir esse objetivo, sua administração é delegada a um organismo central, o InterNIC, que designa grupos de endereços diretamente aos sites que queiram ligar-se à Internet ou aos provedores que os redistribuirão aos próprios clientes.

As universidades e empresas ligadas à Internet têm pelo menos um desses endereços, geralmente não atribuído diretamente pelo InterNIC, mas obtido de um Provedor de Serviço de Internet (Internet Service Provider ISP).

Para configurar uma rede privada doméstica basta “inventar” os próprios endereços privados, como se explica à frente. Para conectar a própria máquina à Internet, todavia, é necessário obter um “verdadeiro” endereço de IP do próprio administrador de rede ou do provedor.

Para a escrita dos endereços IP é usada particularmente uma notação chamada *dotted quad* (quadripontuada), na qual os quatro bytes do endereço são escritos um de cada vez (a partir do mais significativo), separando-os com um ponto. Por exemplo 132.199.15.99 é um endereço válido. Um endereço pode também ser escrito utilizando-se uma palavra de 32 bits em hexadecimal, por exemplo: 0x84c70f63. Este formato não é tão intuitivo como o precedente, mas às vezes pode-se revelar útil como será visto nas seções seguintes.

Atribuir uma rede a um host significa simplesmente estabelecer valores determinados para alguns dos 32 bits do endereço do host. Estes bits, usados para identificar a rede, são chamados “bits de rede”. Os bits restantes servem para identificar o host no interior da rede, e são portanto denominados “bits de host”.

No exemplo precedente o endereço de rede é 132.199.0.0 (em um endereço de rede os bits de host são definidos como zero), enquanto o endereço do host na rede é 15.99.

Como se faz para saber que o endereço do host ocupa 16 bits? Simples: é indicado pelo provedor do qual se obtém o próprio endereço. No estradamento inter-domínios sem classificação (Classless Inter-Domain Routing,CIDR) utilizado hoje, o número de bits reservados ao host vai de 2 a 16 e o número de bits reservado à rede é escrito depois do endereço IP, separado por uma “/”. Por exemplo, 132.199.0.0/16 indica que a rede em questão tem 16 bits. Quando nos referimos à “dimensão” de uma rede, o fazemos em geral simplesmente com “/16”, “/24”, etc.

Antes do advento do CIDR as redes eram divididas em quatro classes. Cada classe iniciava-se com uma bem precisa seqüência de bits que a identificava. Em detalhe:

- A *Classe A* inicia-se com “0” como o bit mais significativo. Os próximos 7 bits de um endereço de Classe A identificam a rede e os restantes 24 bits são usados para endereçar o host. Portanto, em uma rede de Classe A pode haver um host 2^{24}. Não é fácil, todavia, para uma firma ou para uma universidade obter um endereço de Classe A completo.

  A notação CIDR para uma rede de Classe A com os seus 8 bits de rede é “/8”.

- A *Classe B* inicia-se com "10" como os bits mais significativos. Os próximos 14 bits identificam a rede e os restantes 16 bits servem para endereçar os hosts (mais de 65000). Os endereços de Classe B, que já foram muito comuns para firmas e universidades, são hoje muito difíceis de se obter, por causa da atual escassez de endereços IPv4 disponíveis.

  A notação CIDR para uma rede de Classe A com os seus 16 bits de rede é "/16".

  Retornando ao exemplo precedente, pode-se ver que o host 132.199.15.99 (ou o equivalente hexadecimal 0x84c70f63, que aqui é mais cômodo) encontra-se em uma rede de Classe B, já que 0x84 = **10**00... (em binário).

  Portanto, o endereço 132.199.15.99 pode ser dividido na parte de rede que é 132.199.0.0 e na parte do host que é 15.99.

- A *Classe C* é identificada pelos bits mais significativos (chamados também MSB, Most Significant Bits) iguais a "110", permitindo apenas 256 hosts em cada uma das 2^{21} possíveis redes de Classe C (na realidade, como será visto depois, o número de hosts possíveis é 254). Os endereços de Classe C são mais comumente encontrados em pequenas empresas.

  A notação CIDR para uma rede de classe C com os seus 24 bits de rede é "/24".

- Existe também uma ulterior série de endereços que se iniciam com "111", e que são muito usados para outros fins (por exemplo, multicast) e que não nos interessam aqui.

Note-se que os bits utilizados para identificar a rede fazem, não obstante, parte do próprio endereço de rede.

Quando se separa a parte de rede da parte de host é cômodo utilizar a assim chamada *netmask* (máscara de rede). Trata-se de um valor a ser usado como "mascára", em que todos os bits de rede são assentados em "1" e todos os bits de host em "0". Pondo juntos com um AND lógico o endereço com a netmask obtém-se o endereço da rede.

Reportando-se ao exemplo precedente, 255.255.0.0 é uma possível netmask de 132.199.15.99. Quando se aplica a netmask ao endereço, permanece a parte de rede 132.199.0.0.

Para os endereços em notação CIDR, o número indicado de bits de rede especifica também quais dos bits mais significativos devem ser postos em "1" para se obter a netmask da rede correspondente. Para os endereços de rede Classe A/B/C existe uma netmask default:

- Class A (/8): netmask default: 255.0.0.0, primeiro byte do endereço: 1-127
- Class B (/16): netmask default: 255.255.0.0, primeiro byte do endereço: 128-191
- Class C (/24): netmask default: 255.255.255.0, primeiro byte do endereço: 192-223

Um tipo particular de endereço que é bom conhecer é o "endereço de broadcast". As mensagens enviadas a este endereço são recebidas por *todos* os hosts da rede correspondente. O endereço de broadcast é caracterizado pelo fato de ter todos os bits de host colocados em "1".

Por exemplo, dado o endereço 132.199.15.99 com a máscara de rede 255.255.0.0, o endereço de broadcast é 132.199.255.255.

Nesse ponto podem-nos perguntar se podemos utilizar um endereço de host com todos os bits em “0” ou em “1”. A resposta é não, porque o primeiro é o endereço da rede e o segundo o endereço de broadcast, e esses dois endereços devem estar sempre presentes. Agora se pode compreender porque uma rede de Classe B pode conter no máximo 2^{16}-2 hosts e uma rede de Classe C pode conter 2^{8}-2 = 254.

Além dos vários tipos de endereço já citados, há um outro que é especial. Trata-se do endereço 127.0.0.1, que se refere sempre ao host local (*localhost*). Isso significa que quando se “fala” com o 127.0.0.1, comunica-se na realidade consigo mesmo, sem dar lugar a nenhuma atividade de rede, o que pode ser útil para utilizar os serviços instalados na própria máquina ou para fazer simulações e testes quando não há outro host na rede.

Resumindo o que se discutiu até agora:

Endereço IP

endereço de 32 bits, comprendendo a parte de rede e a parte do host.

Endereço de rede

endereço IP com todos os bits de host postos em “0”.

Netmask (máscara de rede)

máscara de 32 bits em que os bits que se referem à rede são colocados em “1”, enquanto os que se referem ao host são postos em “0”.

Endereço de broadcast

endereço IP com todos os bits de host postos em “1”.

Localhost

o endereço IP do host local. É sempre 127.0.0.1

9.1.5. Sub-rede e estradamento (routing)

A seção precedente descreveu em detalhe as máscaras de rede e os endereços de host e de rede. O discurso sobre redes, entretanto, está longe de terminar.

Consideremos a situação de uma típica universidade, com um endereço de Classe B (/16) que lhe permite ter até 2^{16} ~= 65534 hosts na rede. Mesmo que fôsse cômodo pensarmos em ter todos estes hosts em uma mesma rede, isto não pode ser realizado por causa das limitações físicas do hardware em uso atualmente.

Por exemplo, o comprimento máximo de um cabo Ethernet thinwire é de 185 metros. Ainda que se ponham repetidores que amplifiquem o sinal entre os pontos de rede, isto não basta para se atingir todos os pontos de rede em que se encontram as máquinas. Além disso, o número máximo de hosts em um cabo Ethernet thinwire é de 1024 e, avizinhando-se desse limite, tem-se uma perda de eficiência.

Portanto, encontramo-nos em uma situação em que temos um endereço que permitiria ter 60000 hosts mas estamos limitados pelo cabeamento, que nos permite atingir um número muito menor de pontos de rede.

Naturalmente existe a solução, e consiste em subdividir a “rede” de Classe B em redes menores, usualmente denominadas *sub-redes*. Estas sub-redes podem hospedar, por exemplo, até 254 hosts cada (isto é, subdividimos a grande rede de Classe B em várias sub-redes de Classe C).

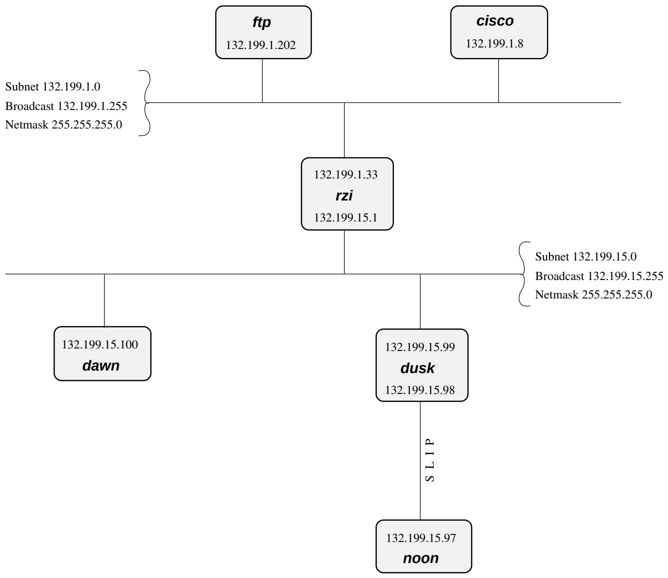
Para se obter esse resultado, é necessário modificar a máscara de rede para se ter mais bits de rede e menos bits de host. Isto geralmente se faz trabalhando sobre os bytes da máscara (mesmo que seja possível descer ao nível de um único bit). Portanto, a coisa mais simples é transformar a máscara de rede de 255.255.0.0 (Classe B) em 255.255.255.0 (Classe C).

Usando a notação CIDR, escreve-se agora "/24" ao invés do precedente "/16" para indicar que são usados 24 bits do endereço para identificar a rede e a sub-rede,em vez dos 16 usados antes.

Esta modificação permite-nos ter um byte de rede a mais para cada uma das redes físicas. Todos os 254 hosts de cada rede podem comunicar-se diretamente e podem ser criadas 256 redes desse tipo (Classe C). Esta nova configuração deveria ser adequada às exigências da nossa universidade hipotética.

Para esclarecer melhor o conceito, continuemos com o exemplo precedente. Suponhamos que o nosso host seja 132.199.15.99, que se chame dusk e que tenha a netmask 255.255.255.0, o que significa que se encontra na sub-rede 132.199.15.0/24. Naturalmente há outros hosts,como se vê pela Figura 9-1.

Figura 9-1. Exemplo de rede



Na rede do exemplo,dusk pode falar diretamente com dawn, já que ambos encontram-se na mesma sub-rede. Existem ainda outros hosts na sub-rede 132.199.15.0/24, mas por ora não os consideraremos.

O que acontece se dusk quer falar com um host que se encontra em uma outra sub-rede? Nesse caso, o tráfego ocorrerá através de um ou mais gateways (routers) anexados às duas sub-redes. Por esse motivo um gateway tem sempre dois endereços diferentes: um para cada uma das subredes a que está conectado. Do ponto de vista funcional, o router (estradador) é totalmente transparente para os hosts. Ou seja, não é necessário conhecer-lhe o endereço para atingir os hosts que estão do "outro lado". É suficiente endereçar diretamente aos hosts e os pacotes alcançarão o destino correto.

Suponhamos por exemplo que dusk queira fazer o download de arquivos do servidor FTP local. Já que dusk não pode alcançar ftp diretamente (porque se encontra em uma sub-rede diferente) todos os seus pacotes serão encaminhados ao seu "router por default" rzi (132.199.15.1), que sabe para onde deve enviá-los para que atinjam a destinação.

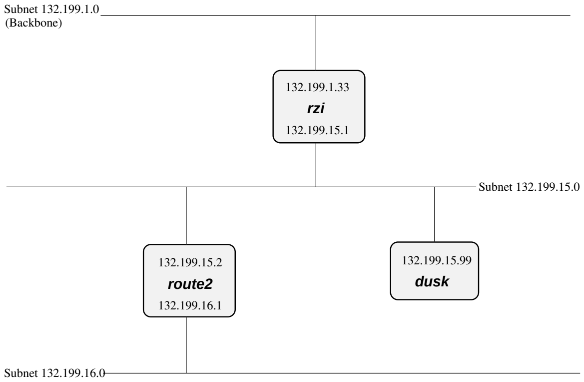
dusk conhece o endereço do router por default da sua rede (rzi, 132.199.15.1), e lhe encaminha todos os pacotes que não são destinados à mesma sub-rede. Isto é, neste caso, todos os pacotes IP que têm o terceiro byte do endereço diferente de 15.

O router por default (defaultrouter) envia os pacotes ao host apropriado, visto que se encontra na sub-rede do servidor de FTP.

Neste exemplo *todos* os pacotes são enviados à rede 132.199.1.0/24 simplesmente porque se trata do *backbone* da rede, a parte mais importante da rede, que transporta todo o tráfego de passagem entre as várias sub-redes. Quase todas as sub-redes além de 132.199.1.0/24 estão conectadas ao backbone de modo similar.

Que acontece se conectássemos uma outra sub-rede a 132.199.15.0/24 ao invés de 132.199.1.0/24? Provavelmente algumas coisa parecida à situação descrita na Figura 9-2.

Figura 9-2. Conectando uma sub-rede a uma outra sub-rede



Agora, para dusk atingir um host que se encontra na sub-rede 132.199.16.0/24, os pacotes não podem ser encaminhados a rzi. É necessário enviá-los diretamente a route2 (132.199.15.2). dusk deve saber que deve encaminhar estes pacotes para route2 e enviar todos os outros para rzi.

Quando se configura dusk, dizemos a ele para enviar todos os pacotes para a sub-rede 132.199.16.0/24, a route2, e todos os outros a rzi. Ao invés de explicitar este default com 132.199.1.0/24, 132.199.2.0/24, etc., pode-se usar o valor 0.0.0.0 para definir a rota default.

Retornando à Figura 9-1, verifica-se um problema parecido quando dawn quer enviar a noon, que está ligado a dusk via cabo serial. Do exame de seu endereço IP,noon pareceria estar conectado à rede 132.199.15.0. Esse realmente não é o caso. Ao contrário, dusk é usado como gateway e dawn deve enviar seus pacotes a dusk que, por sua vez, os encaminhará a noon. O sistema graças ao qual dusk é forçado a aceitar pacotes que não são destinados a ele, mas a um outro host (noon), é definido "proxy arp".

O mesmo discurso vale quando um host de uma outra sub-rede que enviar a noon. Deverá enviar os seus pacotes a dusk (eventualmente encaminhados por rzi).

9.1.6. Conceitos de Name Service

Nas seções precedentes, fazendo referência aos hosts utilizamos freqüentemente seus endereços IP. Isto era necessário para explicar as diferenças entre os vários tipos de endereços. Em geral, todavia, quando se fala de host é mais cômodo utilizar seu “nome”, como fizemos na seção sobre routing.

Para a maior parte das aplicações não importa se os hosts são especificados usando o nome ou o endereço IP. Todavia, internamente elas usam endereços IP. Nesta seção serão introduzidos os conceitos que estão na base de cada um desses métodos. Sucessivamente será explicado como efetuar-lhes a configuração.

A conversão dos nomes dos hosts (e dos domínios) em endereços de IP é realizada por um módulo de software chamado *resolver*. Não se trata de um serviço complementar, mas de uma série de funções de biblioteca que são articuladas com cada aplicação que usa funções de rede. O resolver procura “resolver” (daí o nome) os nomes dos hosts em endereços IP. Veja-se [RFC1034] e [RFC1035] para maiores detalhes sobre o resolver.

Os nomes dos hosts têm geralmente um comprimento máximo de 10 caracteres e podem conter letras, números, hífens (“-”) e traços de subscrito (“_”). Maiúsculas e minúsculas são consideradas equivalentes.

Assim como para as redes e sub-redes, é possível (mais que isso, desejável) reagrupar os hosts em domínios e subdomínios. Quando se obtém um endereço de rede do provedor, geralmente se obtém também um nome de domínio. Como para as sub-redes, é deixada ao usuário a tarefa de criar sub-domínios. Os sub-domínios não estão em relação direta com as sub-redes. Por exemplo, um domínio pode conter hosts conectados a diversas sub-redes.

Figura 9-1 mostra as sub-redes 132.199.1.0/24 e 132.199.15.0/24 (e as outras) que fazem parte do sub-domínio rz.uni-regensburg.de. O domínio que a Universidade de Regensburg obteve de seu provedor de IP é “uni-regensburg.de” (“.de” indica a Alemanha, Deutschland); o sub-domínio “rz” indica Rechenzentrum (Centro de Cálculo).

Os nomes dos hosts, dos sub-domínios e dos domínios são separados por pontos (“.”. Também é possível, embora não seja comum, usar mais de um nível de sub-domínio. Por exemplo: fox_in.socs.uts.edu.au.

Um nome de host que compreende o (sub)domínio é denominado domínio completamente qualificado (FQDN, Fully Qualified Domain Name). Por exemplo, o endereço IP 132.199.15.99 pertence ao host cujo FQDN é dusk.rz.uni-regensburg.de.

Anteriormente foi dito que o endereço IP 127.0.0.1 pertence sempre ao host local, qualquer que seja o “verdadeiro” endereço IP do próprio host. Portanto, 127.0.0.1 é sempre mapeado pelo nome localhost.

Os três métodos diferentes de traduzir um nome de host em endereço IP são: `/etc/hosts`, o Domain Name Service (DNS) e o Network Information Service (NIS).

9.1.6.1. `/etc/hosts`

O primeiro e mais simples dos três métodos para traduzir um nome de host para endereço IP consiste no uso de uma tabela que põe em correspondência os nomes e os endereços correspondentes. Esta tabela encontra-se no arquivo `/etc/hosts` e tem o seguinte formato:

```
endereço IP        nome host [alias [...]]
```

As linhas que se iniciam com um sustenido (cerquinha) ("`#`") são considerados comentários e são ignoradas. As outras linhas contêm um endereço IP e o/os correspondentes nomes de host.

Não é possível que um nome de host corresponda a mais de um endereço IP. rzi, por exemplo, tem dois nomes diferentes para os seus dois endereços: rzi e rzia.

Pode ser cômodo dar vários nomes (alias) a um host. Isto permite especificar um nome alternativo para o host em relação a um serviço particular, como é feito, por exemplo, com o servidor de FTP. O primeiro nome (o mais à esquerda) é usualmente o nome real (canônico) do host.

Também pode ser útil usar o nome completo (FQDN) do host como nome canônico, e o seu hostname (sem o domínio) como alias.

Importante: devemos *sempre* convencionar o mapping do localhost em 127.0.0.1!

9.1.6.2. O Domain Name Service (DNS)

O uso do `/etc/hosts` comporta um problema, especialmente nas redes de grandes dimensões: quando se acrescenta um host ou se altera o endereço de um host, todos os arquivos `etc/hosts` de todas as máquinas devem ser atualizadas. Isso não só implica uma perda de tempo, mas também é uma possível causa de erros e inconsistências que inevitavelmente trarão problemas.

Um outro método é manter uma só tabela de nomes de hosts (banco de dados) para a rede inteira e fazer com que todos os clientes interroguem esse “nameserver”.

Esta é a idéia de base sobre a qual se funda o *Domain Name Service* (DNS).

Usualmente há um servidor de nomes para cada domínio (daí o nome Domain Name Server) e cada host no domínio sabe a qual domínio pertence e qual servidor de nomes deve interrogar para seu próprio domínio.

Quando o DNS recebe uma interrogação para um host que não se encontra em seu domínio a encaminha ao DNS do domínio correspondente ou a um DNS que sabe a qual DNS solicitar a informação. Este último encaminha a interrogação a um DNS de nível superior e assim por diante. Este processo não vai adiante ao infinito. Há vários servidores “root” que possuem as informações de todos os domínios.

Ver Capítulo 11 para maiores detalhes sobre o DNS.

9.1.6.3. O Network Information Service (NIS/YP)

A Yellow Pages (YP) foram inventadas pela Sun Microsystems. O nome depois foi mudado para Network Information Service (NIS) porque YP já era uma marca da British Telecom.

Nos sistemas Unix estão habitualmente presentes um grande número de configurações e é, portanto, muito desejável limitar a manutenção destes arquivos a uns poucos hosts. Os hosts são, assim, agrupados em domínios NIS (que não têm nada a ver com os domínios DNS) e em geral fazem parte de um cluster (cacho) de workstations.

Alguns dos arquivos de configuração compartilhados por estes hosts são `/etc/passwd`, `/etc/group` e `/etc/hosts`.

É, portanto, possível “abusar” do NIS para obter uma única tradução de nomes para endereços em todos os hosts de um domínio NIS.

Há apenas uma desvantagem que impede ao NIS de ser efetivamente utilizado para esse tipo de tradução: diferentemente do DNS, o NIS não fornece nenhum método para resolver os nomes dos hosts que não fazem parte da tabela dos hosts. Não existem hosts de nível superior que o servidor NIS possa interrogar e, portanto, a tradução fica fadada ao fracasso. O NIS+, também da Sun, resolve esse problema, mas como o NIS+ está disponível apenas para os sistemas Solaris, não nos será de grande ajuda.

Do que foi dito não quer dizer que o NIS não seja um método excelente para gerenciar, por exemplo, as informações sobre os usuários (`/etc/passwd`, ...) em clusters de workstations. Simplesmente não é muito útil para resolver os nomes dos hosts.

9.1.6.4. Outros métodos

Os métodos descritos até aqui para a resolução dos nomes dos hosts em endereços IP são aqueles atualmente de uso mais comum, mas não são os únicos. Por definição pode-se utilizar um banco de dados qualquer, mas não há desenvolvimentos desse tipo no NetBSD.

O NIS+, como já foi dito, procura remediar a falta de hierarquia nas estruturas de dados do NIS. As tabelas podem ser estabelecidas de modo que se uma interrogação não puder ser satisfeita pelo servidor de um domínio, pode haver um outro domínio de nível superior em condição de dar uma resposta. Por exemplo, pode-se definir um domínio que relacione todos os hosts (usuários, grupos, etc.) pertencentes a uma firma inteira, um outro que define os mesmos dados para cada divisão, e assim por diante. O NIS+ não é atualmente muito usado. Até a própria Sun retomou o fornecimento do NIS inicial.

No passado, o padrão X.500 foi projetado para gerenciar bancos de dados simples, como o arquivo `/etc/hosts`, e estruturas hierárquicas complexas, como por exemplo o atual DNS. O X.500 jamais teve muito sucesso, em parte porque tentava fazer muitas coisas ao mesmo tempo. Uma versão reduzida chamada LAP (Lightweight Directory Access Protocol) está hoje disponível e a popularidade está crescendo nos últimos anos, para gerir os dados dos usuários (mas também dos hosts) em organizações de pequenas e médias proporções.

9.1.7. IPv6, o protocolo de última geração da Internet

9.1.7.1. O futuro da Internet

A juízo dos experts, a Internet na sua forma atual deverá enfrentar um problema sério dentro de poucos anos. Por causa de seu rápido crescimento e das limitações em seu projeto, chegar-se-á a um ponto em que não estarão mais disponíveis endereços para conter novos hosts. Nesse momento, não se poderão acrescentar novos servidores da web, os usuários não poderão obter contas dos ISPs, não se poderão conectar novas máquinas em rede para acessar a web ou participar de jogos online. Em resumo, um problema muito sério.

Foram pensadas várias abordagens para resolver este problema. Um método muito difundido consiste em não atribuir endereço válido unívoco para cada máquina conectada, mas atribuir endereços “privados” e “esconder” várias máquinas por trás de um único endereço oficial global. Este método, que se chama Network Address Translation (NAT, conhecido ainda como IP Masquerading), apresenta problemas porque as máquinas escondidas atrás do endereço único global não podem ser endereçadas diretamente e, portanto, não é possível abrir conexões com elas, como exigido, por exemplo, pelos jogos online, pelas redes peer-to-peer, etc. Para uma discussão sobre as desvantagens do NAT, ver [RFC3027].

Uma abordagem diferente ao problema da escassez de endereços da Internet é abandonar o velho protocolo da Internet e a sua limitada capacidade de endereçamento, em favor de um novo protocolo que não sofre estas limitações. O protocolo, ou melhor, o conjunto de protocolos usado pelas máquinas conectadas para formar a atual Internet é conhecido como suite TCP/IP (Transmission Control Protocol/Internet Protocol), e a sua versão 4, a que está em uso atualmente, tem todos os problemas descritos há pouco. Para poder passar a uma nova versão do protocolo que resolva esses problemas é necessário, obviamente, que a tal versão “melhor” exista. De fato, a versão 6 do Internet Protocol (IPv6) existe e responde a todas as possíveis exigências futuras de espaço de endereçamento, enfrentando ainda outros problemas como a segurança de dados, a criptografia e um suporte melhor à computação móvel.

Esta seção introduz o protocolo IPv6. Para uma boa compreensão será exigido um conhecimento básico do funcionamento do protocolo IPv4. São descritas as mudanças no formato do endereço e na resolução dos nomes. Usufruindo dessas informações, a Secção 10.2.4 mostrará como utilizar o IPv6 mesmo se ele não for oferecido pelo ISP, utilizando um mecanismo de transição simples mas eficaz denominado *6to4*. O objetivo final será por-se online com o IPv6, mostrando um exemplo de configuração do NetBSD.

9.1.7.2. Por que passar ao IPv6?

Quando se aconselha alguém a passar do IPv4 para o IPv6, a primeira pergunta que se ouve em retorno é: “por que”? Há várias boas razões.

- Maior espaço de endereçamento.
- Suporte à “mobile computing”.
- Gestão integrada da segurança.

9.1.7.2.1. Maior espaço de endereçamento

A ampliação do espaço de endereçamento oferecido é a primeira das vantagens oferecidas pelo IPv6 em comparação com o IPv4. A atual arquitetura da Internet está baseada em endereços de 32 bits, enquanto que a nova versão dispõe de endereços de 128 bits. Graças a esta ampliação, não será mais necessário utilizar soluções como a NAT e levará a uma completa e ilimitada conectividade para todas as máquinas atuais baseadas em IP, assim como para os novos dispositivos móveis como PDA e telefones celulares, que poderão dispor de acesso IP completo graças a GPRS e a UTMS.

9.1.7.2.2. Mobilidade

Quando se fala de dispositivos móveis e IP, um ponto importante a se ter presente é que é necessário um protocolo especial para suportar a mobilidade. A implementação deste protocolo, chamado *IP móvel*, é um dos requisitos de cada stack IPv6. Portanto, quando se ativa o IPv6 obtém-se o suporte para o roaming entre redes diferentes, com atualização completa quando se deixa uma rede e se transfere para uma outra. O suporte ao *roaming* é possível também com o IPv4, mas é preciso resolver alguns problemas antes de se obter um sistema que funcione. Com o IPv6 não é necessário nenhum artifício particular, uma vez que o suporte à mobilidade é um dos requisitos desse projeto. Ver [RFC3024] para maiores informações sobre os problemas a resolver para suportar o “IP móvel” no IPv4.

9.1.7.2.3. Segurança

Além do suporte para a mobilidade, a segurança era um dos requisitos do sucessor à atual versão do Internet Protocol. Como conseqüência, os stacks do protocolo IPv6 devem incluir o *IPsec*. O IPsec permite autenticar, criptografar e comprimir qualquer tipo de tráfico IP. Diferentemente dos protocolos em nível de aplicação, como o SSL ou o SSH, pode-se gerenciar todo o tráfico IP entre dois nódulos sem ter que fazer modificações nas próprias aplicações. A vantagem dessa abordagem é que todas as aplicações em uma máquina podem beneficiar-se da criptografia e da autenticação, e que as políticas correspondentes possam ser estabelecidas, de modo geral, no nível do host ou até mesmo da rede, e não simplesmente no nível do aplicativo/serviço. Uma introdução ao IPsec que compreende também os links com a documentação pode-se encontrar em [RFC2411], enquanto que o protocolo de base é descrito em [RFC2401].

9.1.7.3. Modificações no IPv4

Agora que vimos as várias características importantes do IPv6, podemos passar ao exame das bases do protocolo. Para uma melhor compreensão é preferível um conhecimento elementar do IPv4, em comparação com o qual serão descritas as diferenças. Comecemos com os endereços IPv6, explicando como estão estruturados, quais são os tipos de endereço e o que foi feito dos broadcasts. Assim, depois de ter discutido o "substrato" do IP, passaremos às mudanças relativas à resolução dos nomes e às novidades do IPv6 para o DNS.

9.1.7.3.1. Endereçamento

Um endereço IPv4 é um valor de 32 bits, escrito usualmente no formato "dotted quad", onde cada um dos valores separados por pontos é um valor compreendido entre 0 e 255. Por exemplo:

```
127.0.0.1
```

Esta estrutura permite um número máximo teórico de 2^{32} ou ~4 bilhões de hosts serem conectados à atual Internet. Por causa dos agrupamentos, todavia, nem todos os endereços estão realmente disponíveis.

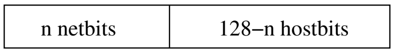
Os endereços IPv6 usam 128 bits, que nos levam a 2^{128} de hosts teoricamente endereçáveis. Isto permite endereçar um número verdadeiramente enorme de máquinas, cobrindo completamente sem problemas todos os requisitos atuais e futuros mesmo de PDAs e telefones celulares dotados de endereço IP. Para escrever os endereços IP, dividimo-os em grupos de 16 bits escritos com quatro algarismos hexadecimais, separando os grupos com o caracter ":" (dois pontos). Por exemplo:

```
fe80::2a0:d2ff:fea5:e9f5
```

Este exemplo mostra também que uma série de zeros consecutivos pode ser abreviada com um único "::" no endereço IPv6. O endereço precedente é pois equivalente a fe80:0:00:000:2a0:d2ff:fea5:e9f5. Os zeros iniciais dentro do grupo podem ser omitidos.

Para simplificar o gerenciamento, os endereços são divididos em duas partes: os bits que identificam a rede a que a máquina pertence e os bits que identificam a própria máquina na (sub)rede. Os primeiros são definidos como "bits de rede" (*netbits*) os segundos como "bits de host" (*hostbits*) e, tanto no IPv4 quanto no IPv6, os bits de rede são os que estão mais à esquerda (bits mais significativos), enquanto que os bits de host estão mais à direita (bits menos significativos), como se mostra na Figura 9-3.

Figura 9-3. Endereços: bits de rede e bits de host



No IPv4 os bits de rede e os bits de host são separados graças à máscara de rede. Exemplos típicos são 255.255.0.0, que usa 16 bits para o endereço de rede, e 255.255.255.0, que usa outros 8 bits, permitindo, por exemplo, endereçar 256 sub-redes em uma rede de classe B.

Quando se passa do endereçamento por classes ao formato CIDR, a separação entre bits de rede e bits de host deixa de cair necessariamente sobre o limite dos 8 bits e, por conseqüência, as máscaras de rede começam a ficar mais difíceis de gerenciar. No lugar das máscaras já se começa a indicar o limite usando o número de bits de rede para um dado endereço. Por exemplo:

```
10.0.0.0/24
```

corresponde a uma máscara de rede de 255.255.255.0 (24 bits por 1). O mesmo esquema é usado no IPv6. O valor:

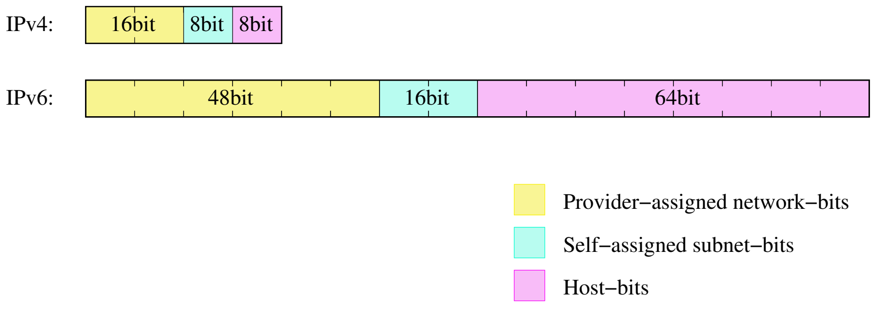
```
2001:638:a01:2::/64
```

indica que no endereço especificado os primeiros 64 bits (os que estão mais à esquerda) dizem respeito à rede, enquanto os últimos 64 bits (os mais à direita) identificam a máquina na rede. Os bits de rede são freqüentemente chamados "prefixo" (da rede) e aqui o comprimento do prefixo é de 64 bits.

Entre os esquemas de endereçamento mais comuns do IPv4, estavam as redes de classe B e de classe C. Com uma rede de classe C (/24) o provedor atribui 24 bits, deixando 8 bits disponíveis. Se acrescentamos as sub-redes acaba-se tendo uma máscara de rede "irregular" não muito prática de se administrar. Neste caso uma rede de classe B (/16) seria mais fácil de administrar porque são fixados só 16 bits pelo provedor, o que permite criar sub-redes com mais facilidade, dividindo em duas partes os restantes 16 bits. A primeira parte endereça a sub-rede local e a segunda endereça o host no interior da própria sub-rede. Geralmente se "interrompe" no extremo dos oito bits (um byte). Usando uma máscara de rede de 255.255.255.0 (/24) consegue-se administrar de modo mais flexível mesmo redes de grandes dimensões, continuando-se a manter o limite de 256 sub-redes com 254 máquinas cada uma.

Com os 128 bits disponíveis para o endereçamento IPv6, o esquema utilizado é o mesmo, apenas que com os campos maiores. Os provedores usualmente estabelecem redes /48, o que deixa 16 bits para as sub-redes e 64 bits para os hosts.

Figura 9-4. Os endereços IPv6 têm uma estrutura parecida com os endereços de classe B



Enquanto o espaço para as rede e as sub-redes parece adequado, o uso de 64 bits para os hosts parece um desperdício de espaço. Visto que é improvável a utilização de vários bilhões de hosts na mesma sub-rede, porque usar essa estrutura?

A idéia que está na base de um identificador de host de comprimento fixo de 64 bits é que os identificadores de host não são mais estabelecidos manualmente, como é feito hoje com o IPv4. No lugar disso, recomenda-se (é só uma recomendação, não uma obrigação) contruí-los partindo dos assim chamados endereços EUI64. Os endereços EUI64 ocupam, como diz o nome, 64 bits e derivam do endereço MAC da placa de rede. Por exemplo, para a Ethernet, ao endereço de rede de 6 bytes (42 bits) são usualmente acrescentados os bits "fffe" no meio e um bit é definido para indicar que o endereço é único (o que é verdadeiro para a Ethernet). Por exemplo, o endereço MAC

```
01:23:45:67:89:ab
```

torna-se o endereço EUI64

```
03:23:45:ff:fe:67:89:ab
```

o que leva os bits de host do endereço IPv6 a ter a forma

```
::0323:45ff:fe67:89ab
```

Estes bits de host podem ser usados para designar o endereço IPv6 a um host suportando a auto-configuração dos hosts IPv6. Tudo o que é preciso para se ter um endereço IPv6 completo são os primeiros bits (rede/sub-rede), mas o IPv6 oferece ainda o sistema para determiná-los automaticamente.

Nas redes de máquinas IP há geralmente um router que age como gateway para com a redes externas. Nas redes IPv6 o router envia uma série de informações de "router advertisement", que os clientes deveriam receber enquanto estão em funcionamento ou solicitar explicitamente enquanto estão em fase de inicialização. Estas informações compreendem o endereço do router e os prefixos de rede geridos pelo próprio router. Com estas informações e o endereço EUI64 gerado automaticamente, um host IPv6 pode determinar seu próprio endereço IP sem uma atribuição manual. Naturalmente os routers devem ainda ser explicitamente configurados.

As informações enviadas pelo router fazem parte do protocolo NDP Neighbor Discovery Protocol (ver [RFC2461]), que é o sucessor do protocolo ARP do IPv4. Diferentemente do ARP, o NDP não apenas busca os endereços IP correspondentes aos endereços MAC, mas efetua também um serviço parecido para os routers e seus prefixos correspondentes das redes geridas, o que é utilizado pela auto-configuração dos hosts IPv6 recém descritos.

9.1.7.3.2. Endereços múltiplos

Com o IPv4, um host tem geralmente um endereço IP para cada interface de rede ou, se o stack o suporta, até mesmo por máquina. Somente alguns aplicativos específicos, como os servidores web, exigem máquinas que têm mais de um endereço IP. Com o IPv6, tudo isso é diferente. Para cada interface, não só há um endereço global, mas também dois outros endereços interessantes: o endereço "link local" e o endereço "site local". O endereço link local tem prefixo fe80::/64 e os bits de host derivam do endereço EUI64: é usado só para contactar os hosts e os routers da mesma rede e não é visível ou alcançável a partir de outras sub-redes. Se assim o desejarmos, pode-se escolher entre usar endereços globais (designados pelo provedor) ou endereços site local. Estes últimos têm o endereço de rede fex0::/10 e as sub-redes e os hosts podem ser endereçados exatamente como no caso das redes designadas pelo provedor. A única diferença é que o endereço não será visível pelas máquinas externas porque se encontram em uma rede diferente e seus endereços site local, se existem, estão em uma rede física diferente. Para o IPv6 é prática comum atribuir aos hosts o endereço global e o endereço link local.

Os endereços site local estão-se tornando cada vez mais raros. Até porque para a conectividade global é exigido um endereço global único.

9.1.7.3.3. Multicasting

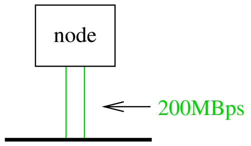
No domínio IP há três modos para se atingir um host: *unicast*, *broadcast* e *multicast*. A forma mais comum de conversação usufrui do endereço unicast que, no IPv4, é o endereço IP "normal" atribuído a um host, compreendendo os bits de rede. O endereço de broadcast é utilizado para se alcançar todos os hosts da mesma sub-rede IP e tem os bits de rede definidos pelo endereço da rede e os bits de host todos em "1". Os endereços de tipo multicast servem para se atingir os hosts que fazem parte do grupo multicast, que se podem encontrar em um ponto qualquer da Internet. As máquinas devem unir-se explicitamente a um grupo multicast para poder participar e há endereços especiais IPv4 usados para o multicast, alocados na sub-rede 224/8. O multicast não é muito utilizado sob o IPv4. Apenas alguns aplicativos como os utilitários MBone de áudio e vídeo broadcast o usam.

Os endereços unicast IPv6 são usados do mesmo modo que os IPv4. Não há nenhuma mudança. Assim, todos os bits de rede e de host são designados para identificar a rede e a máquina. O broadcast não está mais disponível em IPv6, do modo em que o era no IPv4. É aqui que entra em jogo o multicasting. Os endereços da rede ff::/8 são reservados para as aplicações multicast e há dois endereços multicast especiais que tomam o lugar do endereço de broadcast do IPv4. Um é o endereço multicast " all routers" (todos os estradadores) e o outro é o "all hosts" (todos os hospedeiros). Estes endereços são específicos da sub-rede. Ou seja, um router conectado a duas sub-redes pode endereçar todos os hosts/routers de qualquer uma das duas sub-redes. Nesse caso os endereços são:

- ff0*X*::1 para todos os hosts
- ff0*X*::2 para todos os routers

onde "*X*" é o ID da conexão que identifica a rede e que, geralmente, é iniciado em "1" para o âmbito local, em "2" para o primeiro link, etc. Note-se que é possível que duas interfaces de rede sejam vinculadas ao mesmo link, resultando na duplicação da amplitude da rede.

Figura 9-5. Várias interfaces conectadas a um link: um único ID para link



Um uso do endereço de multicast "all hosts" está no código NDP, quando uma máquina que se quer comunicar com uma outra máquina envia uma solicitação ao grupo all hosts e a máquina em questão deve responder.

9.1.7.3.4. A resolução dos nomes em IPv6

Depois desta longa introdução ao endereçamento em IPv6, é lícito esperar que haja um modo para continuar a usar nomes para nos referirmos aos hosts, como se faz com o IPv4, no lugar de desfiar os longos endereços IPv6. Naturalmente é possível.

A tradução do nome de host para o endereço IP em IPv4 geralmente se faz em um destes três modos: usando uma simples tabela no `/etc/hosts`, usando o Network Information Service (NIS, antes YP) ou com o Domain Name System (DNS).

No momento o NIS/NIS+ em IPv6 está disponível apenas para o Solaris 8, tanto para o conteúdo do banco de dados como para o transporte, usando uma extensão RPC.

Um simples mapa do tipo endereço<->nome, do tipo do `/etc/hosts` é suportado por todos os stacks IPv6. Com a implementação KAME usada no NetBSD, o arquivo `/etc/hosts` contém tanto os endereços IPv4 como os IPv6. A título de simples exemplo, vejamos a expressão "localhost" na versão default para uma instalação do NetBSD.

```
127.0.0.1               localhost
::1                     localhost
```

Para o DNS não são introduzidos conceitos fundamentalmente novos. A resolução dos nomes IPv6 é feita com registro de tipo AAAA que, como indica o nome, apontam para uma entidade que é quatro vezes maior que um registro do tipo A. Um registro AAAA tem um nome de host na parte esquerda, do mesmo jeito que para A, e o endereço IPv6 na parte direita. Por exemplo:

```
noon            IN      AAAA    3ffe:400:430:2:240:95ff:fe40:4385
```

Para a resolução inversa o IPv4 usa a zona `in-addr.arpa`, à qual acrescenta os bytes (decimais) em ordem inversa, vale dizer com os bytes mais significativos mais à direita. Com o IPv6 o método é o mesmo, só que são usados os algarismos hexadecimais (um a cada 4 bits) ao invés dos números decimais e os registros dos recursos encontram-se em um domínio diferente: ip6.int.

Portanto, para a resolução inversa do host precedente, deve-se por no arquivo `/etc/named.conf` uma expressão parecida com a seguinte:

```
zone "0.3.4.0.0.0.4.0.e.f.f.3.IP6.INT" {
        type master;
        file "db.reverse";};
```

e, no arquivo de zona `db.reverse`, além dos usuais registros SOA e NS:

```
5.8.3.4.0.4.e.f.f.f.5.9.0.4.2.0.2.0.0.0 IN   PTR   noon.ipv6.example.com.
```

Aqui o endereço é escrito como uma cifra hexadecimal e, ao mesmo tempo, em ordem inversa, iniciando-se pela menos significativa (a que está mais à direita) e separando os números com pontos, como nos costumeiros arquivos de zona.

Quando se instala o DNS para o IPv6, é preciso prestar atenção à versão do software DNS que se utiliza. BIND 8.x compreende os registros AAAA mas não oferece a resolução dos nomes via IPv6. Porisso é necessário utilizar o BIND 9.x que, entre outras coisas, suporta alguns tipos de registro de recursos ainda em fase de discussão e, portanto, não introduzidos oficialmente. O mais interessante é o registro A6, que facilita a mudança do provedor/prefixo.

Resumindo, nesta seção foram discutidas as diferenças técnicas entre o IPv4 e o IPv6 para o endereçamento e resolução dos nomes. Alguns aspectos, como as opções dos títulos IP, QoS, etc. não foram intencionalmente tratados para não avolumar demais a matéria, dado também o caráter introdutório deste guia.

Bibliografia

[AeleenFrisch] Aeleen Frisch, 1991, O'Reilly & Associates, *Essential System Administration*.

[CraigHunt] Craig Hunt, 1993, O'Reilly & Associates, *TCP/IP Network Administration*.

[RFC1034] P. V. Mockapetris, 1987, *RFC 1034: Domain names - concepts and facilities*.

[RFC1035] P. V. Mockapetris, 1987, *RFC 1035: Domain names - implementation and specification*.

[RFC1055] J. L. Romkey, 1988, *RFC 1055: Nonstandard for transmission of IP datagrams over serial lines: SLIP*.

[RFC1331] W. Simpson, 1992, *RFC 1331: The Point-to-Point Protocol (PPP) for the Transmission of Multi-protocol Datagrams over Point-to-Point Links*.

[RFC1332] G. McGregor, 1992, *RFC 1332: The PPP Internet Protocol Control Protocol (IPCP)*.

[RFC1933] R. Gilligan e E. Nordmark, 1996, *RFC 1933: Transition Mechanisms for IPv6 Hosts and Routers*.

[RFC2004] C. Perkins, 1996, *RFC 2003: IP Encapsulation within IP*.

[RFC2401] S. Kent e R. Atkinson, 1998, *RFC 2401: Security Architecture for the Internet Protocol*.

[RFC2411] R. Thayer, N. Doraswamy, e R. Glenn, 1998, *RFC 2411: IP Security Document Roadmap*.

[RFC2461] T. Narten, E. Nordmark, e W. Simpson, 1998, *RFC 2461: Neighbor Discovery for IP Version 6 (IPv6)*.

[RFC2529] B. Carpenter e C. Jung, 1999, *RFC 2529: Transmission of IPv6 over IPv4 Domains without Explicit Tunnels*.

[RFC3024] G. Montenegro, 2001, *RFC 3024: Reverse Tunneling for Mobile IP*.

[RFC3027] M. Holdrege e P. Srisuresh, 2001, *RFC 3027: Protocol Complications with the IP Network Address Translator*.

[RFC3056] B. Carpenter e K. Moore, 2001, *RFC 3056: Connection of IPv6 Domains via IPv4 Clouds*.

Capítulo 10. Configurações de rede

10.1. Prática

10.1.1. Opções de configuração do kernel

Antes de adentrarmos os meandros das configurações de rede, é bom dar uma olhada nos correspondentes parâmetros presentes na configuração do kernel. Para maiores detalhes sobre o tópico veja-se Capítulo 7. Aqui nos concentraremos apenas sobre os aspectos relativos à montagem e gerenciamento de redes, tomando como exemplo o arquivo de configuração i386/GENERIC. Os arquivos de configuração para as outras plataformas contêm opções similares, geralmente acompanhadas de comentários dentro do próprio arquivo que lhes esclarecem o uso. Ademais, as opções de compilação do kernel estão documentadas na página de manual options(4) e, geralmente, há uma página de manual para cada driver (por exemplo tlp(4)).

```
#       $NetBSD: GENERIC,v 1.354.2.15 2001/05/06 15:18:54 he Exp $
```

A primeira linha do arquivo de configuração mostra o número de versão que, neste caso, é 1.354.2.15 e que pode ser útil para confrontar com outras versões via CVS ou para reportar eventuais bugs.

```
options         NTP             # NTP phase/frequency locked loop
```

Se quisermos usar o Network Time Protocol (NTP), pode-se ativar esta opção para obter a máxima precisão. Se esta opção não está ativa, o NTP funciona do mesmo modo. Ver ntpd(8) para detalhes ulteriores.

```
file-system     NFS             # Network File System client
```

Para utilizar o disco rígido de uma outra máquina com o Network File System (NFS), é necessário ativar esta opção. Ver Secção 10.2.2 para maiores informações.

```
options         NFSSERVER       # Network File System server
```

Esta opção compreende o lado servidor do protocolo NFS e deve ser habilitada para permitir a outras máquinas usarem o disco rígido local. Ver Secção 10.2.2 para maiores informações.

```
#options        GATEWAY         # packet forwarding
```

Para instalar um estradador (router) que encaminha os pacote entre várias redes ou entre várias interfaces de rede, é preciso habilitar a opção GATEWAY. Não apenas ativa o estradamento dos pacotes mas aumenta ainda alguns buffers. Ver options(4) para outros detalhes.

```
options         INET            # IP + ICMP + TCP + UDP
```

Ativa o suporte ao TCP/IP no kernel. Mesmo se não se pretende usar redes, esta opção é ainda assim necessária para as comunicações internas à máquina por parte de sub-sistemas como o X Window. Ver inet(4) para os detalhes.

```
options         INET6           # IPV6
```

Quem quer usar o IPv6 deve ativar esta opção que foi introduzida com a versão 1.5 do NetBSD. Quem não quer IPv6 pode comentá-la. Ver a página de manual inet6(4) e Secção 9.1.7 para ulteriores informações sobre o protocolo Internet de nova geração.

```
#options        IPSEC           # IP security
```

Inclui o suporte para o protocolo IPsec: autenticação, compressão, chaves, etc. Esta opção pode ser utilizada mesmo sem ativar o IPv6, quando se deseja usar o IPsec apenas com o IPv4, o que é possível. Ver ipsec(4).

```
#options        IPSEC_ESP       # IP security (encryption part; define w/IPSEC)
```

Esta opção é utilizada em complemento ao IPSEC para ativar a criptografia IPsec.

```
#options        MROUTING        # IP multicast routing
```

Incluir esta opção para ativar o estradamento de serviços multicast como o MBone. Note-se que o routing é depois controlado pelo dæmon mrouted(8).

```
options         NS              # XNS
#options        NSIP            # XNS tunneling over IP
```

Estas opções habilitam a família de protocolos Xerox Network Systems(tm). Não estão em relação com o stack do protocolo TCP/IP e não são muito utilizados hoje. A página de manual ns(4) contém alguns detalhes.

```
options         ISO,TPIP        # OSI
#options        EON             # OSI tunneling over IP
```

Estas opções ativam o stack do protocolo OSI que, depois de ter sido considerado o futuro das redes, tem hoje um interesse prevalentemente histórico. Ver a página de manual iso(4).

```
options         CCITT,LLC,HDLC  # X.25
```

Estas opções ativam o conjunto de protocolos X.25 para a transmissão de dados por linha serial. São, ou melhor, eram, usadas juntamente com o protocolo OSI ou em redes WAN.

```
options         NETATALK        # AppleTalk networking protocols
```

Inclui o suporte para o stack do protocolo AppleTalk. Para poder usá-lo são necessários os programas servidores que podem ser encontrados, por exemplo, em `pkgsrc/net/netatalk` e `pkgsrc/net/netatalk-asun`. Maiores informações sobre o protocolo AppleTalk e seu stack correspondente podem ser encontrados na página do manual atalk(4).

```
options         PPP_BSDCOMP     # BSD-Compress compression support for PPP
options         PPP_DEFLATE     # Deflate compression support for PPP
options         PPP_FILTER      # Active filter support for PPP (requires bpf)
```

Estas opções controlam vários aspectos do protocolo PPP (Point-to-Point-Protocol). As duas primeiras determinam os algoritmos de compressão usados/disponíveis, enquanto a terceira habilita o código para a filtragem de alguns pacotes.

```
options         PFIL_HOOKS      # pfil(9) packet filter hooks
```

```
options         IPFILTER_LOG    # ipmon(8) log support
```

Estas opções habilitam o uso do firewall com IPfilter. Ver as páginas de manual ipf(4) e ipf(8) para maiores informações sobre o uso do IPfilter, assim como a Secção 10.2.1 para um exemplo de configuração.

```
# Compatibility with 4.2BSD implementation of TCP/IP.  Not recommended.
#options        TCP_COMPAT_42
```

Esta opção é necessária somente se na rede ainda estão presentes máquinas que usam 4.2BSD ou um stack de rede derivado desta versão. Na presença de máquinas 4.2BSD é necessário prestar atenção na correta configuração do endereço de broadcast. Isso porque o 4.2BSD tem um bug no código de networking relativamente ao endereço de broadcast. Este bug força pôr todos os bits de host em "0" no endereço de broadcast. A opção TCP_COMPAT_42 serve para isso.

```
options         NFS_BOOT_DHCP,NFS_BOOT_BOOTPARAM
```

Estas opções habilitam a busca de informações com DHCP ou com o protocolo BOOTPARAM quando o kernel usa um sistema de arquivos "root" do tipo NFS. Ver a página de manual diskless(8) para informações adicionais.

```
# Kernel root file system and dump configuration.
config          netbsd  root on ? type ?
#config         netbsd  root on sd0a type ffs
#config         netbsd  root on ? type nfs
```

Estas linhas indicam ao kernel onde procurar o próprio sistema de arquivos "root" e de qual tipo de sistema de arquivos se trata. Se, por exemplo, quer-se criar um kernel que usa um sistema de arquivos (filesystem) root do tipo NFS através da interface tlp0, pode-se fazê-lo com a diretriz "`root on tlp0 type nfs`". Se usamos `?` no lugar do tipo de dispositivo, o kernel procura determinar o dispositivo autonomamente.

```
# ISA serial interfaces
com0    at isa? port 0x3f8 irq 4        # Standard PC serial ports
com1    at isa? port 0x2f8 irq 3
com2    at isa? port 0x3e8 irq 5
```

Para usar o PPP ou SLIP é necessário ter uma interface serial (com). Também funciona uma que se encaixe em USB, PCMCIA ou PUC.

```
# Network Interfaces
```

A lista das interfaces de rede é bastante longa e contém dispositivos de todo tipo. É necessário simplesmente selecionar aquela que corresponde ao próprio hardware, baseando-se nos comentários presentes no arquivo. A maior parte dos drivers tem sua própria página de manual. Por exemplo, tlp(4), ne(4), etc.

```
# MII/PHY support
```

Esta seção elenca as interfaces de rede independentes do dispositivo. Em caso de dúvida, habilitar todas elas e deixar a escolha para o kernel. Ver a página de manual mii(4) para maiores informações.

```
# USB Ethernet adapters
aue*    at uhub? port ?         # ADMtek AN986 Pegasus based adapters
cue*    at uhub? port ?         # CATC USB-EL1201A based adapters
kue*    at uhub? port ?         # Kawasaki LSI KL5KUSB101B based adapters
```

Os adaptadores ethernet USB têm uma banda limitada a apenas 2 Mbits/s mas são práticos de usar. Naturalmente, para funcionar, é necessário configurar também as opções para USB que, contudo, não são descritas aqui, além, é claro, de dispor do hardware necessário. Vejam-se as várias páginas de manual.

```
# network pseudo-devices
pseudo-device   bpfilter        8       # Berkeley packet filter
```

Este *pseudo-dispositivo* permite o “sniffing (farejamento)” de pacotes de todos os tipos. É necessário para tcpdump, mas também para rarpd e para outros aplicativos que devem examinar o tráfico da rede. Ver bpf(4) para informações adicionais.

```
pseudo-device   ipfilter                # IP filter (firewall) and NAT
```

Esta linha ativa a interface IPfilter do kernel para a filtragem dos pacotes usada pelo firewalling, NAT (também chamado IP masquerading), etc. Ver ipf(4) e Secção 10.2.1 para maiores informações.

```
pseudo-device   loop                    # network loopback
```

Esta é a interface “loopback” de rede, usada tanto por alguns programas quanto para efetuar algum tipo de routing. Deve ser habilitada. Ver a página de manual lo(4) para os detalhes.

```
pseudo-device   ppp             2       # Point-to-Point Protocol
```

Esta opção é necessária para se poder usar o protocolo PPP tanto via interface serial quanto via ethernet (PPPoE). Ver ppp(4).

```
pseudo-device   sl              2       # Serial Line IP
```

O “Serial Line IP” é simplesmente o protocolo IP encapsulado para viajar por uma linha serial. Já que não inclui opções como a negociação dos endereços IP, já não é muito usado atualmente. Veja-se sl(4).

```
pseudo-device   strip           2       # Starmode Radio IP (Metricom)
```

Ver a página de manual strip(4) para o suporte aos velhos dispositivos de rede Metricon Ricochet. Se você tem um, habilite esta opção.

```
pseudo-device   tun             2       # network tunneling over tty
```

Este dispositivo de rede serve para predispor um *túnel* para os pacotes de rede, na direção de um dispositivo `/dev/tun*`. Os pacotes encaminhados à interface tun0 podem ser lidos em `/dev/tun0`, e os dados escritos no `/dev/tun0` são enviados à interface de rede tun0. Com este sistema pode-se implementar, por exemplo, o estradamento QoS em modo usuário. Ver tun(4) para outros detalhes.

```
pseudo-device   gre             2       # generic L3 over IP tunnel
```

O encapsulamento GRE pode ser usado para efetuar um túnel de pacotes arbitrários de nível 3 no IP, por exemplo pela implementação do VPN. Ver gre(4) para detalhes ulteriores.

```
pseudo-device   ipip            2       # IP Encapsulation within IP (RFC 2003)
```

Um outro dispositivo para encapsular IP sobre IP, com um formato diferente de encapsulamento. Ver a página de manual ipip(4) para os detalhes.

```
pseudo-device   gif             4       # IPv[46] over IPv[46] tunnel (RFC 1933)
```

A interface GIF serve para ativar um túnel, por exemplo de IPv6 sobre Ipv4, que pode ser usado para se obter conectividade IPv6 na ausência de uma conexão IPv6 com o ISP. São possíveis também outras combinações de operações: ver os exemplos contidos na página de manual gif(4).

```
#pseudo-device  faith           1       # IPv[46] tcp relay translation i/f
```

A interface faith captura o tráfico TCP IPv6 para implementar o relay de IPv6 a IPv4, por exemplo para transições de protocolo. Ver a página de manual faith(4).

```
#pseudo-device  stf             1       # 6to4 IPv6 over IPv4 encapsulation
```

Esta diretriz acrescenta um dispositivo de rede que se pode usar para criar um túnel IPv6 em IPv4 sem necessidade de configurar o próprio túnel. O remetente dos pacotes contém na saída o endereço IPv4, o que permite enviar as respostas via IPv4. Ver a página de manual stf(4) e a Secção 10.2.4 para maiores detalhes.

```
pseudo-device   vlan                    # IEEE 802.1q encapsulation
```

Esta interface fornece o suporte para as LAN virtuais IEEE 802.1Q, que permitem marcar os pacotes ethernet com uma ID “vlan”. Configurando oportunamente os switches que suportam as VLAN, podem ser criadas VLAN nas quais um grupo de máquinas não vê o tráfico dos outros grupos. A página de manual vlan(4) contém maiores informações a propósito.

10.1.2. Os arquivos de configuração da rede

A lista seguinte contém um elenco dos arquivos de configuração da rede, alguns dos quais já vistos em capítulos precedentes. A descrição destes arquivos será objeto das próximas seções.

`/etc/hosts`

O banco de dados dos hosts locais. Cada linha descreve um host e contém o endereço IP, o nome do host e os eventuais codinomes (aliases). As redes de modestas dimensões podem ser configuradas usando-se apenas os arquivos dos hosts, sem necessidade de um *name server*.

`/etc/resolv.conf`

Este arquivo especifica as modalidades operacionais para as funções de biblioteca do Internet Domain Name System. Geralmente contém os endereços dos “name servers”.

`/etc/ifconfig.xxx`

Este arquivo é usado para a configuração automática da placa de rede na inicialização.

`/etc/mygate`

Contém o endereço IP do gateway.

`/etc/nsswitch.conf`

Arquivo de configuração do *Name Service Switch*. Controla as modalidades de acesso aos vários bancos de dados que contêm informações sobre hosts, usuários, grupos, etc. Na prática este arquivo especifica a ordem com base na qual são consultados os bancos de dados. Por exemplo, a linha:

```
hosts:    files dns
```

especifica que as informações sobre os hosts vêm de duas fontes: o arquivo local `/etc/hosts` e o *DNS* (Internet Domain Name System), e que os arquivos locais são consultados *antes* do DNS.

Geralmente não é necessário modificar este arquivo.

10.1.3. Conectando-se à Internet

Há muitos tipos de conexão à Internet. Esta seção explica como conectar-se a um provedor usando um modem através da linha telefônica, com o protocolo PPP, uma das instalações mais comuns. As operações a realizar, pela ordem, são:

1. Solicitar as informações necessárias ao provedor.
2. Editar o `/etc/resolv.conf` e aferir o arquivo `/etc/nsswitch.conf`.
3. Criar os diretórios `/etc/ppp` e `/etc/ppp/peers` se não existem.
4. Criar o script de conexão, o chat file e o arquivo de opções pppd.
5. Criar o arquivo de autenticação de usuário e senha.

A julgar pela lista acima parece um procedimento complexo, que exige muito trabalho. Na realidade, cada um dos pontos elencados é muito simples. Trata-se apenas de criar, modificar ou simplesmente aferir alguns pequenos arquivos de texto. No exemplo seguinte suporemos que o modem esteja conectado à segunda porta serial `/dev/tty01` (COM2 no DOS).

> Além dos modems externos (ou internos) conectados a uma porta serial (usando `/dev/tty0[012]` em i386, `/dev/tty[ab]` em sparc, ...) podem-se usar até modems de USB (`/dev/ttyU*`) e pcmcia/cardbus (`/dev/tty0[012]`).

10.1.3.1. Buscando as informações de conexão

a primeira coisa a fazer é solicitar ao provedor, que suponhamos chamar-se BigNet, para nos fornecer os dados que servem para se pode conectar. Em particular:

- O número de telefone do POP (Point Of Presence) mais vizinho.
- O tipo de autenticação utilizada.
- O nome de usuário e a nossa senha para usar na conexão.
- Os endereços IP dos nameserver.

10.1.3.2. resolv.conf e nsswitch.conf

O arquivo `/etc/resolv.conf` deve ser configurado com o uso de algumas informações fornecidas pelo provedor, em particular o DNS. Suponhamos que os dos DNS sejam "194.109.123.2" e "191.200.4.52".

Exemplo 10-1. resolv.conf

```
nameserver 194.109.123.2
nameserver 191.200.4.52
#lookup file bind
```

> **Nota:** a linha "lookup file bind" indica que os servidores de nomes são utilizados para as expressões não presentes em `/etc/hosts`. A linha está comentada porque a partir da versão 1.4 do NetBSD não é mais utilizada. As suas funções são desenvolvidas agora pelo arquivo `/etc/nsswitch.conf`. O novo Name Server Switch permite mudar os métodos de acesso aos bancos de dados usados pelos programas para acessar as informações básicas do sistema.

Exemplo 10-2. nsswitch.conf

```
# /etc/nsswitch.conf
group:          compat
group_compat:   nis
hosts:          files dns
netgroup:       files [notfound=return] nis
networks:       files
passwd:         compat
passwd_compat: nis
```

> **Nota:** a única linha modificada neste arquivo é a que se inicia com "hosts:". Note-se que nas versões posteriores à 1.4.1 não deveria ser mais necessário efetuar essa modificação manualmente.

10.1.3.3. Criando o diretório para o ppp

Os diretórios `/etc/ppp` e `/etc/ppp/peers` deverão conter os arquivos de configuração para a conexão. Ao término de uma nova instalação do NetBSD não existem e, assim, é preciso criá-los (chmod 700).

10.1.3.4. Script de conexão e chat file

O script de conexão é o que vem especificado na linha de comando do pppd. Encontra-se em `/etc/ppp/peers` e tem geralmente o nome do provedor. Por exemplo, supondo que o provedor se chame BigNet e que o nosso nome de usuário junto ao provedor seja alan, assim poderia ser o script para a conexão:

Exemplo 10-3. Script de conexão ao provedor

```
# /etc/ppp/peers/bignet
connect '/usr/sbin/chat -v -f /etc/ppp/peers/bignet.chat'
noauth
user alan
remotename bignet.it
```

No exemplo precedente, o script indica o *chat file* a ser utilizado para a conexão. Para uma explicação das opções: pppd(8).

> **Nota:** em caso de problemas de conexão, é possível adicionar ao arquivo precedente duas linha contendo as opções
>
> ```
> debug
> kdebug 4
> ```
>
> Para informações adicionais: pppd(8), syslog.conf(5).

O script de conexão chama o programa chat para efetuar a conexão física verdadeira e própria (inicialização do modem, composição do número, etc.). Os parâmetros a passar para o chat são bastante numerosos e convém pô-los em um arquivo, ainda que seja possível especificá-los inline no script de conexão. Por exemplo, supondo que o número do telefone do POP mais vizinho seja 02 99999999

Exemplo 10-4. Chat file

```
# /etc/ppp/peers/bignet.chat
ABORT BUSY
ABORT "NO CARRIER"
ABORT "NO DIALTONE"
" ATDT0299999999
CONNECT "
```

> **Nota:** ocorrendo problemas com o chatfile, pode ser cômodo conectar-se manualmente ao provedor com o cu: desse modo podemos ver com exatidão as linhas que recebemos. Para informações adicionais: cu(1).

10.1.3.5. Autenticação

Durante a fase de autenticação cada um dos dois sistemas verifica a identidade do outro. Na realidade, o provedor geralmente não espera ser autenticado, mas pretende fazê-lo com o usuário que se conecta.

Geralmente os provedores, para providenciar a nossa autenticação, utilizam um destes dois métodos:

- login
- PAP/CHAP

A maior parte dos provedores utiliza uma autenticação do tipo PAP/CHAP. Esta informação é geralmente fornecida pelo próprio provedor.

10.1.3.5.1. Autenticação com PAP/CHAP

É suficiente criar o arquivo `/etc/ppp/pap-secrets` (ou `/etc/ppp/chap-secrets`), cujas linhas têm o seguinte formato:

```
usuário * password
```

Por exemplo:

```
alan * pZY9o
```

> **Nota:** por motivos de segurança é bom que o arquivo `pap-secrets` seja de propriedade do *root* e tenha permissões ’600’.

10.1.3.5.2. Autenticação com login

Trata-se de uma forma de autenticação cada vez menos usada. Se o provedor usa a autenticação login, o nome do usuário e a senha não devem ser especificados no `pap-secrets` ou no `chap-secrets`, mas no chatfile (naturalmente, nesse caso, é bom adotar permissões adequadas para o chatfile). Neste caso o chatfile simula um login interativo.

Exemplo 10-5. Chat file com login

```
# /etc/ppp/peers/bignet.chat
ABORT BUSY
ABORT "NO CARRIER"
ABORT "NO DIALTONE"
” ATDT0299999999
CONNECT ”
TIMEOUT 50
ogin: alan
ssword: pZY9o
```

10.1.3.6. Opções de pppd

Resta para configurar apenas o arquivo das opções do **pppd**, que é o `/etc/ppp/options` (chmod 644).

Exemplo 10-6. /etc/ppp/options

```
/dev/tty01
lock
crtscts
57600
modem
```

```
defaultroute
noipdefault
```

10.1.3.7. Teste do modem

Antes de ativar a conexão pode ser conveniente realizar um teste do modem para verificar se tudo funciona corretamente. Para o teste pode-se usar o programa cu.

1. Criar o arquivo `/etc/uucp/port` que contém as seguintes linhas:

   ```
   type modem
   port modem
   device /dev/tty01
   speed 115200
   ```

   (naturalmente, no lugar de `/dev/tty01` é necessário substituir o dispositivo apropriado).

2. Executar o comando **cu -p modem** para iniciar um teste interativo. Neste ponto podem-se enviar os comandos ao modem. Por exemplo:

   ```
   # cu -p modem
   Connected.
   ATZ
   OK
   ~.

   Disconnected.
   #
   ```

   No exemplo precedente é enviado ao modem o comando de reset (ATZ) e o modem responde com OK. Como se vê, para sair do cu é necessário executar o comando **~.** (acento til seguido de ponto).

Se o modem parece não funcionar, a primeira coisa a verificar é se o modem está efetivamente conectado à porta usada pelo **cu**. Uma outra causa freqüente de problemas são os cabos.

> **Nota:** se quando se inicia o **cu** aparece uma mensagem que diz “Permission denied”, verificar quem é o proprietário dos arquivos `/dev/tty##`: deve ser uucp. Por exemplo:
>
> ```
> $ ls -l /dev/tty00
> crw-------  1 uucp  wheel  8, 0 Mar 22 20:39 /dev/tty00
> ```
>
> Se, contudo, o proprietário é root:
>
> ```
> $ ls -l /dev/tty00
> crw-------  1 root  wheel  8, 0 Mar 22 20:39 /dev/tty00
> $ cu -p modem
> cu: open (/dev/tty00): Permission denied
> cu: All matching ports in use
> ```

SidebarCOM, com e tty

Não precisa fazer confusão entre *com*, *COM* e *tty*. Para o NetBSD, “com” é o nome do driver (aquele que é mostrado pelo **dmesg**) e “tty” é o nome da porta. Já que a numeração parte de 0, com0 será o driver tty00. Nos ambientes DOS, todavia, COM1 indica a primeira porta (aquela que geralmente tem o endereço 0x3f8), COM2 a segunda e assim por diante. Portanto COM1 (DOS) corresponde a tty00 (NetBSD).

10.1.3.8. Ativando a conexão

Finalmente tudo está pronto para ativar a conexão com o provedor, executando-se o comando

```
# pppd call bignet
```

onde `bignet` é o nome do script de conexão já descrito. Para ver as mensagens de conexão do **pppd** e do **chat**, executar o comando

```
# tail -f /var/log/messages
```

Para desconectar-se é preciso fazer um **kill -HUP** do **pppd**.

10.1.3.9. Automatizando a conexão e a desconexão

Quando a conexão está pronta e funcionando corretamente, chega o momento de criar alguns scripts que aumentam a nossa comodidade, permitindo a conexão e a desconexão automática. Por exemplo, para se conectar:

Exemplo 10-7. ppp-up

```
#!/bin/sh
MODEM=tty01
POP=bignet
if [ -f /var/spool/lock/LCK..$MODEM ]; then
  echo ppp ja\’ esta\’ ativo...
else
  pppd call $POP
  tail -f /var/log/messages
fi
```

e para se desconectar:

Exemplo 10-8. ppp-down

```
#!/bin/sh
MODEM=tty01
if [ -f /var/spool/lock/LCK..$MODEM ]; then
  echo -f killing pppd...
  kill -HUP ‘cat /var/spool/lock/LCK..$MODEM‘
```

```
  echo done
else
  echo ppp ja\' esta\' inativo 1>&2
fi
```

Estes dois scripts usufruem do fato de que o **pppd** quando está ativo cria o arquivo `/var/spool/locl/LDK..tty01`, que contém o *pid* (identificador de processo) do **pppd**

Os dois scripts devem ser executáveis:

```
# chmod u+x ppp-up ppp-down
```

> **Nota:** para acionar o pppd é preciso ser *root* ou pertencer ao grupo *dialer* ou ao grupo *operator*. Portanto, para o uso dos scripts por parte do usuário normal pode-se utilizar o sudo (`/usr/pkgsrc/security/sudo`) ou atribuir ao usuário um dos grupos acima citados.

10.1.4. Criando uma pequena rede doméstica

Operar redes é seguramente um dos pontos de força do Unix em geral e do NetBSD em particular, sendo a criação de uma rede doméstica uma operação muito simples (e econômica). Entre outras coisas, na Secção 10.2.1 será visto como configurar uma máquina NetBSD para fazê-la de gateway e de firewall da Internet para o resto da rede, permitindo a todas as máquinas da rede o acesso à Internet com uma só conexão ao provedor. Também aqui não há necessidade de aquisição de nenhum software complementar. O NetBSD já dispõe de todo o necessário. A única perspicácia necessária diz respeito à aquisição de uma placa de rede suportada pelo NetBSD (consultar o arquivo `INSTALL` para a relação dos dispositivos suportados).

A primeira coisa a fazer é instalar a placa de rede Ethernet na máquina e conectar-se a um hub, a um switch ou conectar-se diretamente entre si com os adequados cabos coaxiais e terminadores. Para este exemplo faça-se referência à figura Figura 10-1.

Uma vez instaladas as placas de rede é necessário atentar para que sejam reconhecidas corretamente pelo kernel estudando o output do **dmesg**. Por exemplo:

```
...
ne0 at isa0 port 0x280-0x29f irq 9
ne0: NE2000 Ethernet
ne0: Ethernet address 00:c2:dd:c1:d1:21
...
```

Naturalmente é necessário que a placa esteja habilitada no kernel. Caso contrário, habilitá-la, recompilar o kernel e reinicializar. Por exemplo:

```
...
ne0 at isa? port 0x280 irq 9 # NE[12]000 ethernet cards
...
```

Se a placa não for reconhecida pelo kernel, é necessário examinar o arquivo de configuração do kernel para corrigir o IRQ da placa de rede. O IRQ especificado no arquivo deve ser aquele que a placa realmente utiliza. Senão, na maior parte dos casos, o kernel não poderá reconhecê-la na inicialização. Se o IRQ for diferente, pode-se mudar a linha do arquivo de configuração e recompilar o kernel ou reconfigurar a placa (usualmente com um disquete de setup ou, para as placas mais antigas, um jumper).

Experimentemos agora executar o seguinte comando:

```
# ifconfig ne0
ne0: flags=8822<BROADCAST,NOTRAILERS,SIMPLEX,MULTICAST> mtu 1500 media: Ether-
net 10base2
```

O output do comando precedente mostra a atual configuração da placa de rede

Neste ponto pode-se passar à configuração de software, que é muito simples. Antes de tudo convém experimentar a placa, começando por configurá-la:

```
# ifconfig ne0 inet 192.168.1.1 netmask 0xffffff00
```

O efeito do comando pode ser verificado com:

```
# ifconfig ne0
ne0: flags=8863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST> mtu 1500 me-
dia: Ethernet 10base2 inet 192.168.1.1 netmask 0xffffff00 broadcast 192.168.1.255
```

Confrontando com o output do comando ifconfig anterior, pode-se notar que apareceu o endereço de rede especificado e as flags “UP” e “RUNNING”. Se uma interface não estiver “UP”, o sistema não a inicializará para transmitir pacotes.

Ao host foi atribuído o endereço IP 192.168.1.1, que pertence aos endereços de rede considerados “internos”, ou seja, reservados às redes não visíveis da Internet. Neste ponto pode-se dizer que a configuração terminou e não resta senão testá-la. Se já existem outros hosts em rede, pode-se testar fazendo um *ping* para um outro host. Por exemplo, se o endereço IP de um host ativo da rede é 192.168.1.2, podemos prová-lo com:

```
# ping 192.168.1.2
PING ape (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=255 time=1.286 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=255 time=0.649 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=255 time=0.681 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=255 time=0.656 ms
^C
----ape PING Statistics----
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.649/0.818/1.286/0.312 ms
```

Na próxima reinicialização a configuração da placa será efetuada novamente. Para automatizar a operação convém criar o arquivo `/etc/ifconfig.ne0`, que contém a linha

```
inet 192.168.1.1 netmask 0xffffff00
```

Depois, no arquivo `/etc/rc.conf` é necessário configurar a opção

```
auto_ifconfig=YES
```

enfim se inserem os endereços dos hosts em /etc/hosts. Por exemplo:

Exemplo 10-9. /etc/hosts

```
#       $NetBSD: hosts,v 1.4 1997/01/09 05:33:14 mikel Exp $
#
# Host Database
# This file should contain the addresses and aliases
# for local hosts that share this file.
# It is used only for "ifconfig" and other operations
# before the nameserver is started.
#
#
127.0.0.1               localhost
#
# RFC 1918 specifies that these networks are "internal".
# 10.0.0.0    10.255.255.255
# 172.16.0.0  172.31.255.255
# 192.168.0.0 192.168.255.255

192.168.1.1   ape.insetti.net ape
192.168.1.2   vespa.insetti.net vespa
192.168.1.0   insetti.net
```

O arquivo `/etc/nsswitch.conf` deve ser modificado como já foi explicado no Exemplo 10-2.

Na próxima reinicialização tudo será configurado automaticamente.

> **Nota:** neste exemplo foi criado o arquivo `/etc/ifconfig.ne0` para que a placa de rede fosse reconhecida como *ne0*. Para outras placas de rede é necessário substituir pelo nome adequado.

Para configurar a rede é necessário, portanto, simplesmente conectar fisicamente os computadores, atribuir a cada um deles um endereço IP e configurar o arquivo `/etc/hosts` com os endereços de todos em cada host e modificar o arquivo `/etc/nsswitch.conf`. Trata-se de uma política de gestão elementar, adaptada somente a redes de pequenas dimensões.

10.1.5. Conectando dois PCs por via serial

Acontece às vezes (ao menos comigo) encontrar-me na necessidade de transferir arquivos de um portátil para um PC. Se o portátil não se pode conectar em rede e não tem o drive do disquete (ou talvez o arquivo é muito grande para um disquete), uma solução simples e eficaz é conectar as duas máquinas com um cabo *null modem*. Nas seções seguintes vejamos alguns exemplos de possíveis configurações.

10.1.5.1. Conectando duas máquinas BSD

Se ambas as máquinas a se coligar utilizam o NetBSD, encontramo-nos na situação mais favorável. A realização de uma conexão SLIP é pra lá de simples. Na primeira máquina

```
# slattach /dev/tty00
# ifconfig sl0 inet 192.168.1.1 192.168.1.2
```

na segunda máquina

```
# slattach /dev/tty00
# ifconfig sl0 inet 192.168.1.2 192.168.1.1
```

Nesse ponto pode-se experimentar a conexão com o ping. Por exemplo, a partir do segundo PC executar

```
# ping 192.168.1.1
```

Se tudo andou bem as duas máquinas estão *em rede* e podemos usar ftp, telnet, etc. Também pode ser conveniente inserir os nomes e os endereços dos hosts no arquivo `/etc/hosts`.

- No exemplo precedente ambas as máquinas usavam a primeira porta serial (/dev/tty00).
- Os endereços 192.169.x.x estão entre os reservados para as redes privadas. A primeira máquina tem o endereço 192.168.1.1 e a segunda tem o endereço 192.168.1.2.
- Para se obter uma conexão mais veloz é necessário especificar o parâmentro `-s` *`velocidade`* para slattach.
- Para poder executar ftp deve ter sido inicializado o inetd e deve estar habilitado o servidor ftpd.

> **Nota:** se um dos dois PCs utiliza Linux, os comandos neste último são ligeiramente diferentes. Supondo dar ao PC Linux o endereço 192.168.1.2:
>
> ```
> # slattach -p slip -s 115200 /dev/ttyS0 &
> # ifconfig sl0 192.168.1.2 pointopoint 192.168.1.1 up
> # route add 192.168.1.1 dev sl0
> ```
>
> Não se pode esquecer do "&" no primeiro comando.

10.1.5.2. Conectando Windows NT a NetBSD

Windows NT e NetBSD podem ser postos em rede de modo simples, com um cabo serial null modem. O procedimento consiste em se criar uma conexão de "acesso remoto" no NT e em inicializar o pppd no NetBSD. Vejamos na prática como se faz.

Supondo inicializar o pppd como root, é necessário criar um arquivo `.ppprc` em `/root`. O conteúdo do arquivo deve ser parecido com o seguinte:

```
connect '/usr/sbin/chat -v CLIENT CLIENTSERVER'
local
tty00
115200
crtscts
lock
noauth
nodefaultroute
:192.168.1.2
```

O significado da primeira linha será explicado depois. 192.168.1.2 é o endereço IP que será designado pelo NetBSD ao host NT. tty00 é a porta serial usada para a conexão.

Do lado do NT é necessário antes instalar um modem do tipo *null modem* (conexão direta) e depois criar uma conexão de acesso remoto que use este modem. O driver null modem é padrão no NT, mas não se trata de um verdadeiro e próprio null modem. De fato, na ativação da conexão o NT envia a linha CLIENT e espera receber a resposta CLIENTSERVER (eis o significado da primeira linha do `.ppprc`: o chat deve responder ao NT, do contrário a conexão não chegará a bom termo). Para a criação da conexão utilizar o null modem, indicar um número telefônico qualquer (ele não será usado), escolher um servidor ppp, habilitar apenas o protocolo TCP/IP, usar endereço IP e nameserver indicados pelo servidor (o NetBSD, no caso). Ademais, selecionar o fluxo de controle de hardware e a porta 115200 8N1.

Nesse ponto pode-se efetuar a conexão:

- Conectar fisicamente os seriais das duas máquinas.
- Ativar o pppd no NetBSD. (Para ver as mensagens enviadas pelo pppd: **tail -f /var/log/messages**).
- Ativar a conexão de acesso remoto no NT.

10.1.5.3. Conectando Windows 95 ao NetBSD

Como para o caso do Windows NT, convém usar a conexão de acesso remoto do Windows e o servidor ppp do BSD. Infelizmente o Windows 95 não dispõe de um driver *null modem*, o que complica um pouco as coisas. A coisa mais simples é procurar um na Internet (trata-se de um arquivo `.INF` banal) e seguir as mesmas instruções do Windows NT, com a única advertência de remover a linha que reclama o chat do arquivo `.ppprc`.

Na falta de um driver null modem pode-se usar um pequeno truque:

- Criar uma conexão de acesso remoto como a descrita na Secção 10.1.5.2, mas utilizando o "Modem Standard".
- No `.ppprc` substituir a linha que aciona o chat com a seguinte

  ```
  connect '/usr/sbin/chat -v ATH OK AT OK ATE0V1 OK AT OK ATDT CONNECT'
  ```
- Para a conexão proceder como no caso do Windows NT.

Desse modo o programa chat, chamado na conexão, emula o comportamento de um modem padrão e restitui ao Windows as linhas de resposta que receberia de um modem: para cada comando AT enviado, o chat responde com um OK.

10.2. Tópicos avançados

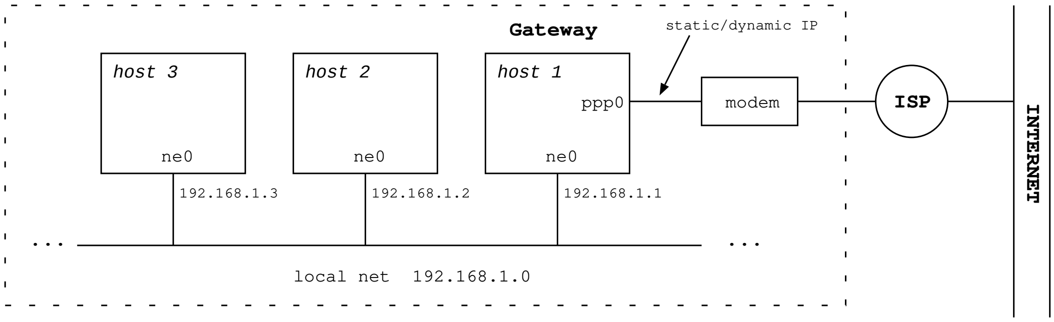
10.2.1. IPNAT

Por trás da misteriosa sigla IPNAT esconde-se o Internet Protocol Network Address Translation, que permite o estradamento (routing) de uma rede IP fictícia (local) para uma verdadeira (Internet). Isto significa que com um só IP fornecido pelo provedor podem-se conectar à Internet vários computadores, criando uma falsa rede IP interna e estradando os dados através de um computador conectado à Internet e no qual funciona o IPNAT.

Para alguns exemplos de uso do IPNAT, ver no subdiretório `/usr/share/examples/ipf`, em particular os arquivos BASIC.NAT e nat-setup.

O exemplo apresentado nesta seção é ilustrado pela Figura 10-1. O host 1 pode conectar-se à Internet com o modem chamando o provedor, que designa um endereço IP dinâmico. Os hosts 2 e 3 não podem comunicar-se com a Internet. Para tornar possível a conexão também a eles, usam-se o IPF e o IPNAT.

Figura 10-1. Rede com gateway



10.2.1.1. Configuração do gateway/firewall

Se o kernel foi compilado com a opção GATEWAY ativa, esta fica habilitada por default sem que se executem operações adicionais de configuração. Para verificar se está habilitada:

```
# sysctl net.inet.ip.forwarding
net.inet.ip.forwarding = 1
```

Se o resultado for “1”, como no exemplo precedente, a opção está habilitada. Além disso, no arquivo de configuração do kernel deve estar habilitado o "pseudo-device ipfilter". Se o resultado todavia for “0”, então a opção não está habilitada e há duas possibilidades:

1. Compilar um novo kernel com a opção GATEWAY ativa.
2. Ativar a opção no kernel atual com o seguinte comando:

   ```
   # sysctl -w net.inet.ip.forwarding=1
   ```

   As definições do sysctl são adicionadas ao arquivo `/etc/sysctl.conf` de modo que sejam restabelecidas automaticamente na reinicialização do sistema. Neste caso é necessário acrescentar

   ```
   net.inet.ip.forwarding=1
   ```

As instruções seguintes explicam como criar uma configuração do IPNAT que entre em funcionamento automaticamente cada vez que nos conectamos com o provedor e se ativa o link PPP. Com esta configuração todos os PCs da rede doméstica poderão compartilhar a conexão à Internet, mesmo que não utilizem o NetBSD.

Criar o arquivo `/etc/ipnat.conf` contendo as seguintes regras:

```
map ppp0 192.168.1.0/24 -> 0/32 proxy port ftp ftp/tcp
map ppp0 192.168.1.0/24 -> 0/32 portmap tcp/udp 40000:60000
map ppp0 192.168.1.0/24 -> 0/32
```

192.168.1.0/24 representa o conjunto de endereços internos a mapear. A primeira linha permite o funcionamento do tcp ativo através do NAT e é opcional. A segunda linha assegura que os pacotes tcp e udp sejam gerenciados corretamente pelo NAT (o portmapping é necessário devido à relação um/muitos). A terceira linha faz funcionar todo o resto (ICMP, ping, etc.).

Criar o arquivo `/etc/ppp/ip-up` que será chamado automaticamente na ativação do link PPP.

```
#!/bin/sh
# /etc/ppp/ip-up
/etc/rc.d/ipnat forcestart
```

Criar o arquivo `/etc/ppp/ip-down` que será acionado automaticamente na desativação do link PPP.

```
#!/bin/sh
# /etc/ppp/ip-down
/etc/rc.d/ipnat forcestop
```

`ip-up` e `ip-down` devem ser executáveis:

```
# chmod u+x ip-up ip-down
```

10.2.1.2. Configuração dos clientes

Criar o arquivo `/etc/resolv.conf` (como o do gateway).

Executar o comando **route add default 192.168.1.1** (192.168.1.1 é o endereço IP do gateway anteriormente configurado).

Naturalmente é muito desconfortável repetir este comando a cada inicialização. Para tornar automática a configuração do gateway podem-se utilizar dois métodos equivalentes:

- especificar o endereço do gateway utilizando a opção “defaultroute” do arquivo `/etc/rc.conf`
- escrever o endereço do gateway no arquivo `/etc/mygate`

A cada inicialização, o conteúdo de `/etc/mygate` (ou da opção defaultroute) são usados como argumento para o comando **route add default**.

Se o cliente não é uma máquina NetBSD, a sua configuração será naturalmente diferente. Em uma máquina Windows, por exemplo, é necessário definir a propriedade “gateway” do protocolo TCP/IP.

10.2.1.3. Alguns comandos úteis

Para se efetuar verificações podem ser úteis os seguintes comandos:

ping

netstat -r

Mostra as tabelas de estradamento (parecido com o **route show**).

traceroute

No cliente, mostra o caminho seguido pelos pacotes para chegar ao destino.

tcpdump

Para executar no gateway.

10.2.2. NFS

Uma vez posta para funcionar a rede, é possível compartilhar arquivos e diretórios entre computadores graças ao NFS. Do ponto de vista do compartilhamento de arquivos, o computador que oferece acesso às suas hierarquias de diretórios é o computador *servidor*. O computador que acessa é o *cliente*. Um computador pode ser contemporaneamente cliente e servidor. Vejamos quais são as operações a cumprir.

- Para o cliente como para o servidor deve ser compilado um kernel com a opção necessária habilitada (não é difícil de encontrar...).
- O servidor deve habilitar os serviços RPC no `/etc/inetd.conf`.
- No `/etc/rc.conf` o servidor deve habilitar o dæmon inetd e portmap, além da opção nfs_server.
- Em `/etc/rc.conf` o cliente deve habilitar inetd e nfs_client.
- O servidor deve elencar as exportações em `/etc/exports` e depois executar o comando **kill -HUP ‘cat /var/run/mountd.pid**.

O acesso a um diretório remoto baseia-se em dois pressupostos:

- O computador remoto exporta o diretório.
- O computador local monta o diretório remoto com o comando **mount server:/xx/yy /mnt**

O comando mount é dotado de um rico sortimento de opções, não propriamente simples.

10.2.2.1. Exemplo de instalação do NFS

Este exemplo, bastante elaborado, foi tomado de uma situação real apresentada em uma mailing list. Suponhamos ter o seguinte cenário: cinco máquinas cliente (cli1, ..., cli5) devem acessar alguns diretórios em um servidor (buzz.toy.org). Alguns dos diretórios exportados pelo servidor são específicos dos clientes individuais, outros compartilhados por todos os clientes. Todos os clientes inicializam a partir do servidor e devem montar os diretórios. Os diretórios exportados pelo servidor são:

`/export/cli?/root`

deve servir de diretório root para cada uma das máquinas cli?.

`/export/cli?/swap`

área de swap para cada uma das máquinas cli?.

`/export/common/usr`

diretório `/usr` comum a todas as máquinas cli?.

`/usr/src`

para montar como `/usr/src` em cada uma das màquinas cli? (comum).

No servidor estão presentes os seguintes sistemas de arquivos

```
/dev/ra0a su /
/dev/ra0f su /usr
/dev/ra1a su /usr/src
/dev/ra2a su /export
```

Nos clientes queremos obter os seguintes sistemas de arquivos

```
buzz:/export/cli?/root su /
buzz:/export/common/usr su /usr
buzz:/usr/src su /usr/src
```

A configuração do servidor é a seguinte:

```
# /etc/exports
/usr/src -network 123.45.67.0 -mask 255.255.255.0
/export  -alldirs -maproot=root -network 123.45.67.0 -mask 255.255.255.0
```

Obviamente 123.445.67.0 deve ser substituído com o endereço real da rede, como também a netmask. Ademais, pode ser necessário acrescentar: **-maproot=root** à linha com "/usr/src".

Nos clientes `/etc/fstab` deve conter

```
buzz:/export/cli?/root / nfs rw
buzz:/export/common/usr /usr nfs rw,nodev,nosuid
buzz:/usr/src /usr/src rw,nodev,nosuid
```

Naturalmente para cada cliente é necessário substituir o número correspondente ao lugar do caracter "?" em `/etc/fstab`.

10.2.3. Instalando `/net` com amd

O autor deste seção (Instalando ...) è Hubert Feyrer <`hubert@feyrer.de`>.

10.2.3.1. Introdução

O problema com o mount NFS (e com os outros tipos de montagem) é que geralmente podem ser realizadas apenas pelo usuário “root”, o que pode resultar em muito desconforto para os outros usuários. Graças ao **amd** pode-se selecionar um diretório (no nosso caso será usado o diretório `/net`) sob o qual mesmo os usuários comuns possam efetuar a montagem do NFS, à condição de que o sistema de arquivos a que se deve acessar seja efetivamente exportado pelo servidor NFS.

Para verificar quais são os sistemas de arquivos exportados por um servidor de NFS, usar o comando **showmount** com a opção `-e`:

```
# showmount -e wuarchive.wustl.edu
Exports list on wuarchive.wustl.edu:
/export/home                       onc.wustl.edu
/export/local                      onc.wustl.edu
/export/adm/log                    onc.wustl.edu
/usr                               onc.wustl.edu
/                                  onc.wustl.edu
/archive                           Everyone
```

Para montar um diretório e acessar todo o seu conteúdo, incluindo-se os sub-diretórios (por exemplo `/archive/systems/unix/NetBSD`), basta entrar no diretório com **cd**:

```
# cd /net/wuarchive.wustl.edu/archive/systems/unix/NetBSD
```

O sistema de arquivos é montado automaticamente pelo **amd** e é possível acessar todos os arquivos, exatamente como se o diretório tivesse sido montado pelo administrador do sistema.

10.2.3.2. Práticas de instalação

O diretório `/net` do nosso exemplo pode ser instalado com as seguintes operações (que incluem o setup do **amd**:

1. no `/etc/rc.conf`, selecionar a seguinte variável:

   ```
   amd=yes
   ```

2. **mkdir /amd**
3. **mkdir /net**
4. Usando `/usr/share/examples/amd/master` como exemplo, acrescentar a seguinte diretriz `/etc/amd/master`:

   ```
   /net            /etc/amd/net
   ```

5. Usando `/usr/share/examples/amd/net` como exemplo, acrescentar as seguintes linhas a `/etc/amd/net`:

   ```
   /defaults       type:=host;rhost:=${key};fs:=${autodir}/${rhost}/root
   *               host==${key};type:=link;fs:=/                            \
     host!=${key};opts:=ro,soft,intr,nodev,nosuid,noconn
   ```

6. Reinicializar a máquina, ou reinicializar o **amd** manualmente:

```
# sh /etc/rc.d/amd restart
```

10.2.4. Conectividade IPv6 e passagem para o IPv6 com o 6to4

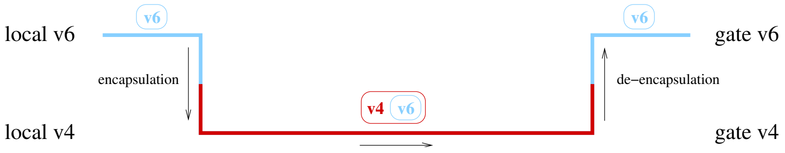
O autor deste seção (Conectividade ...) é Hubert Feyrer `<hubert@feyrer.de>`

Esta seção explica de forma prática como chegar à conectividade IPv6 e, já que esta última ainda não é fácil de se obter de modo nativo, descreve amplamente as alternativas à conectividade nativa IPv6 e um método para se efetuar a transição até que as conexões IPv6 se tornem mais comuns.

Encontrar um ISP que ofereça a conectividade IPv6 nativa requer uma boa dose de sorte. O passo seguinte é encontrar um router capaz de gerenciar o tráfico. No momento nem todos os produtores de routers introduzem suporte a IPv6 em suas máquinas e, quando o fazem, é improvável que ofereçam estradamento ou switching IPv6 com aceleração de hardware. Uma alternativa econômica e de uso comum ao router hardware consiste no uso de um PC padrão, configurando-o como router. Por exemplo, utilizando um sistema operacional como o NetBSD e um software do tipo do Zebra para gerenciar os protocolos de estradamento. Esta solução é hoje bastante difundida para os sites que queiram logo a conectividade IPv6. A única desvantagem é que só serve um ISP que suporte IPv6 e uma conexão dedicada apenas para IPv6.

Na falta de conectividade nativa, pode-se ainda utilizar o IPv6 graças aos *túneis*. No lugar de enviá-los diretamente, os pacotes IPv6 são encapsulados dentro de pacotes IPv4, como mostrado na Figura 10-2. Usando a infraestrutura IPv4, os pacotes encapsulados são enviados a uma conexão real IPv6, que remove o encapsulamento e o envia em modo nativo.

Figura 10-2. Um túnel de pacotes IPv6 dentro do IPv4



Para usar os túneis há duas possibilidades. A primeira consiste no uso de um túnel que se chama “configurado”. A segunda no uso de um túnel “automático”. O túnel configurado caracteriza-se pelo fato de requerer uma preparação de ambos os lados do próprio túnel, geralmente sujeita a alguma forma de registro a fim de intercambiar os dados de configuração. Um exemplo desse túnel é o encapsulamento IPv6-sobre-Ipv4 descrito em [RFC1933] e implementado, por exemplo, pela interface gif(4) do NetBSD.

Um túnel automático requer um servidor público dotado de uma forma qualquer de conectividade IPv6, por exemplo via 6bone. Este servidor torna públicos os próprios dados de conectividade e ativa o protocolo de “tunneling” que não requer um registro explícito dos sites que o utilizam para se conectar. Um exemplo popular desse protocolo é o mecanismo 6to4 descrito em [RFC3056] e implementado pelo dispositivo stf(4) do NetBSD. Um outro mecanismo que não requer registro de informações IPv6 é o assim chamado 6over4, que implementa o transporte do IPv6 em uma rede IPv4 habilitada para o multicast ao invés de, por exemplo, ethernet ou FDDI. O 6over4 está documentado em [RFC2529]. A sua principal desvantagem é que é necessária a presença de uma infraestrutura multicast existente. Se não há uma estrutura desse tipo, a sua criação e configuração requer um esforço comparável à configuração de túnel IPv6 direto e, portanto, nesse caso, trata-se de uma hipótese que não vale a pena considerar.

10.2.4.1. Utilizando o IPv6 com o 6to4

O 6to4 oferece uma solução simples para a obtenção da conectividade IPv6 para os host que são dotados de uma conexão IPv4 (verSecção 9.1.7. Pode-se utilizar com endereços IPv4 estáticos e dinâmicos (o tipo mais comum no caso de conexão "dial-up" via modem). Quando se usa um endereço Ipv4 dinâmico, as mudanças de endereço são um problema para o tráfico de entrada, o que implica que não se pode usar este tipo de disposição para um servidor web.

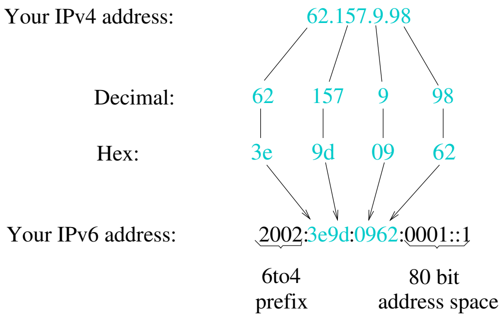
A configuração do exemplo descrita nesta seção refere-se ao NetBSD 1.5.2.

10.2.4.2. Obtendo o espaço de endereços IPv6 para o 6to4

O setup IPv6 do 6to4 não consiste de um endereço só do lado do usuário. De fato, obtém-se uma rede /48 inteira! Os endereços IPv6 são extraídos a partir de um simples endereço IPv4 original. O prefixo "2002:" é reservado aos endereços de tipo 6to4 (isto é, aos endereços IPv6 derivados do IPv4). Os 32 bits seguintes contém o endereço IPv4. Isto corresponde a ter a disponibilidade de uma rede /48 inteira, deixando 16 bits de espaço para 2^{64} nódulos. A Figura 10-3 mostra a construção do endereço (ou melhor, do conjunto de endereços) IPv6 a partir do IPv4.

Graças ao prefixo 6to4 e à unicidade do endereço IPv4, o bloco de endereços IPv6 obtidos é também único e atribuído à máquina que tinha o endereço IPv4 de partida.

Figura 10-3. 6to4 deriva um endereço IPv6 de um IPv4

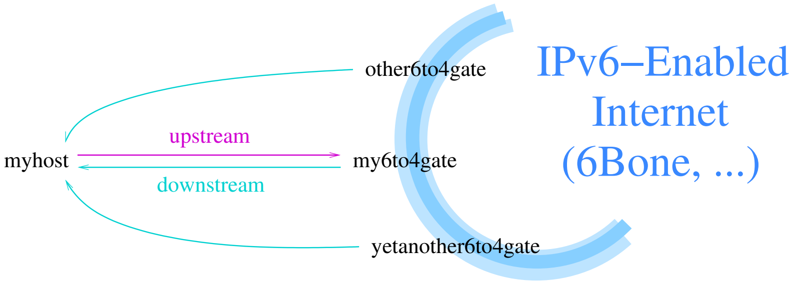


10.2.4.3. Como conectar-se

Diferentemente do setup configurado do túnel "IPv6-over-IPv4", não é necessário registrar-se em um gateway 6bone que encaminhe o tráfico IPv6. Em lugar disso, já que o endereço IPv6 deriva do IPv4, as respostas são enviadas ao usuário do gateway 6to4 mais próximo. O encapsulamento dos pacotes é eliminado por uma interface de rede habilitada para 6to4 que depois providencia seu encaminhamento de acordo com o estradamento instalado localmente (caso haja mais de uma máquina conectada à rede 6to4 que foi estabelecida).

No que diz respeito ao envio de pacotes IPv6 (tráfico de saída), a interface de rede habilitada para 6to4 pega o pacote IPv6 e o encapsula em um pacote IPv4. É necessária uma conexão no gateway 6to4, por sua vez conectado ao 6bone, que elimine o encapsulamento dos pacotes e os encaminhe ao 6bone, como se ilustra na Figura 10-4.

Figura 10-4. Solicitação e resposta podem passar por dois gateways diferentes no setup do 6to4



10.2.4.4. Segurança

Diferentemente do setup com um "túnel pré-configurado", neste caso não se consegue usualmente instalar filtros para os pacotes, a fim de bloquear pacotes 6to4 provenientes de fontes não autorizadas, exatamente por causa do método de funcionamento do mecanismo 6to4. Os pacotes com as seguintes características não deveriam ser considerados pacotes 6to4 válidos e, assim, é justificado filtrá-los:

- endereço fonte/destino IPv4 não especificado: 0.0.0.0/8
- endereço de loopback (v4) como fonte/destinação: 127.0.0.0/8
- multicast IPv4 na fonte/destinação:224.0.0.0/4
- broadcast limitado: 255.0.0.0/8
- endereço de broadcast de sub-rede na fonte/destinação: depende do setup do IPv4

A página de manual stf(4) do NetBSD relata alguns erros comuns de configuração que são interceptados pelo stack KAME e contém também algumas sugestões para a filtragem de pacotes, mesmo se, por causa dos requisitos desses filtros, o 6to4 não é completamente seguro. Todavia, se os pacotes 6to4 fictícios tornam-se um problema, pode-se utilizar a autenticação IPsec para assegurar-se de que os pacotes IPv6 não sejam modificados.

10.2.4.5. Dados necessários para a configuração do 6to4

Para configurar o IPv6 via 6to4 são necessárias algumas informações que tais:

- O endereço IPv4 local, que se pode determinar com o comando **netstat -i** ou com o **netstat -i** na maior parte dos sistemas Unix. Quando se usa um gateway com NAT ou similar, é preciso utilizar o endereço oficial, visível do exterior, e não o privado (10/8 ou 192.168/16).

Neste exemplo será usado o endereço IP local 62.224.57.114.

- O endereço IPv6 local derivado do endereço IPv4. Ver Figura 10-3.

  Neste exemplo o endereço é 2002:3ee0:3972:0001::1 (62.224.57.114 == 0x3ee03972, 0001::1 foi escolhido arbitrariamente).

- O endereço IPv6 do gateway 6to4 a utilizar. O endereço IPv6 está bom porque contém o endereço IPv4, na habitual tradução do 6to4.

  Neste exemplo usaremos 2002:c25f:6cbf::1 (== 194.95.108.191 == 6to4.ipv6.fh-regensburg.de).

10.2.4.6. Preparação do kernel

Para elaborar pacotes do tipo 6to4, o kernel do sistema operacional deve ser posto em condição de reconhecê-lo e gerenciá-lo. É necessário, portanto, habilitar um driver específico.

Para preparar o kernel do NetBSD para o uso do IPv6 e do 6to4, é necessário ativar as seguinte linhas no arquivo de configuração do kernel:

```
options INET6               # IPv6
pseudo-device stf           # 6to4 IPv6 over IPv4 encapsulation
```

> **Nota:** o dispositivo stf(4) não está ativo por default.

Recompilar o kernel e reinicializar a máquina para ativar o novo kernel. Ver Capítulo 7 para informações adicionais sobre a configuração, compilação e instalação de um novo kernel.

10.2.4.7. Setup do 6to4

A presente seção descreve os comandos necessários para o setup do 6to4. Em resumo, as operações a cumprir são:

1. configurar a interface
2. estabelecer a rota por default
3. instalar o Router Advertisement, se assim for desejado.

O primeiro passo a dar para instalar o 6to4 é atribuir um endereço IPv6 à interface 6to4, usando o comando ifconfig(8). Dada a configuração descrita há pouco, o comando é:

```
# ifconfig stf0 inet6 2002:3ee0:3972:1::1 prefixlen 16 alias  (local)
```

Depois de ter configurado o dispositivo 6to4 com este comando, precisa instalar o routing para encaminhar o tráfico IPv6 ao gateway 6to4. O melhor método consiste no fixar uma rota:

```
# route add -inet6 default 2002:cdb2:5ac2::1 (remote)
```

Note-se que o dispositivo stf(4) do NetBSD determina o endereço do gateway 6to4 da tabela de estradamento. Usufruindo desta característica é simples instalar um gateway 6to4 próprio, se se dispõe de uma verdadeira conexão IPv6, por exemplo, via 6bone.

Depois de ter dado estes comandos a conexão ao mundo do IPv6 estará ativa. Congratulações! Assumindo que a resolução dos nomes seja sempre efetuada por IPv4, pode-se agora efetuar um "ping" de um site IPv6 como www.kame.net ou www6.netbsd.org:

```
# /sbin/ping6 www.kame.net
```

O último passo na instalação do IPv6 via 6to4 consiste na configuração do Router Advertisement, para o caso em que haja vários hosts na rede. Se bem que seja possível repetir as instalações 6to4 para cada nódulo, esta operação comporta custos de estradamento onerosos de um nódulo para outro. Os pacotes são enviados ao gateway 6to4 que os encaminha ao nódulo vizinho. Em vez disso é preferível instalar o 6to4 em uma única máquina e estabelecer o IPv6 nativo para a rede local.

Inicia-se com a designação de um endereço IPv6 à interface de rede. No exemplo seguinte assumimos que a sub-rede "2" da rede IPv6 seja a rede ethernet local e que o endereço MAC da interface ethernet seja 12:34:56:78:9a:bc. Ou seja, o endereço IP da interface ethernet do gateway local é 2002:3ee0:3972:2:1234:56ff:fe78:9abc. Atribuimos este endereço à interface ethernet:

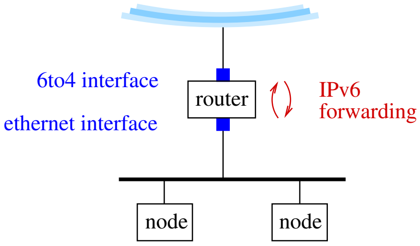
```
# ifconfig ne0 inet6 alias 2002:3ee0:3972:2:1234:56ff:fe78:9abc
```

No comando precedente, "ne0" é a interface que se refere à placa de rede ethernet. Naturalmente será diferente de acordo com a placa instalada no sistema.

Para o setup do router é preciso depois verificar se encaminha efetivamente os pacotes do dispositivo 6to4 local à interface ethernet e vice-versa. Para habilitar o forwarding dos pacote IPv6, definir "ip6mode=router" no arquivo `/etc/rc.conf`, que fixa em "1" o "sysctl" do net.inet6.ip6.forwarding.

```
# sysctl -w net.inet6.ip6.forwarding=1
```

Figura 10-5. Para um router 6to4 é preciso habilitar o forwarding dos pacotes



Para configurar o Router Advertisement é preciso aferir o arquivo `/etc/rtadvd.conf`. Este arquivo permite configurar muitas coisas mas, geralmente, a configuração por default de não conter dados vai bem. Com esta instalação os endereços IPv6 de todas as placas de rede serão tornadas públicas.

Depois de ter verificado que a configuração do Router Advertisement está correta e que o forwarding IPv6 está habilitado, pode-se inicializar o dæmon correspondente que no NetBSD chama-se **rtadvd**. Da primeira vez ele pode ser inicializado manualmente para efeito de teste. Depois convém usar os scripts de inicialização do sistema. Em qualquer hipótese, veremos todos os nódulos locais configurarem magicamente a sub-rede, tornada pública em virtude de seus endereços link-local pré-existentes.

```
# rtadvd
```

10.2.4.8. Breve introdução ao uso do pkgsrc/net/6to4

Até agora descrevemos o funcionamento e instalação do 6to4. Para automatizar estas operações, por exemplo quando se conecta “online”, convém utilizar o pacote “6to4”, que determina o endereço IPv6 a partir do endereço IPv4 atribuído pelo provedor e ainda provê às necessárias configurações.

A instalação do pacote pkgsrc/net/6to4 requer as seguintes operações:

1. Instalar o pacote compilando-o a partir do pkgsrc ou usando o pacote pré-compilado 6to4-1.nb1 com o comando **pkg_add**.

   ```
   # cd /usr/pkgsrc/net/6to4
   # make install
   ```

2. Verificar se o pseudo-dispositivo stf(4) esteja incluso no kernel (ver acima).
3. Configurar o pacote “6to4”. Em primeiro lugar, copiar `/usr/pkg/etc/6to4.conf-example` em `/usr/pkg/etc/6to4.conf`, e depois ordenar as variáveis. Note-se que o arquivo utiliza a sitaxe perl.

   ```
   # cd /usr/pkg/etc
   # cp 6to4.conf-example 6to4.conf
   # vi 6to4.conf
   ```

   A página de manual 6to4(8) descreve todas as variáveis que se possam assentar no `6to4.conf`. Quando se tem uma conexão IP dialup via PPP e não se está interessado no Router Advertising para as outras máquinas IPv6 da rede local doméstica ou do escritório, não é necessário efetuar nenhuma configuração. Se, ao contrário, se quer utilizar o Router Advertising, é necessário definir a variável `in_if` de acordo com o valor da interface ethernet interna. Por exemplo:

   ```
   in_if="rtk0";             # Inside (ethernet) interface
   ```

4. Agora já nos podemos conectar e depois executar o comando 6to4 manualmente:

   ```
   # /usr/pkg/sbin/6to4.pl start
   ```

   Com a conexão efetuada, usar o comando ping6(8) para verificar se tudo funciona corretamente. Em caso afirmativo, inserir as seguintes linhas no script `/etc/ppp/ip-up` que é ativado toda vez que nos conectamos:

   ```
   logger -p user.info -t ip-up Configuring 6to4 IPv6
   /usr/pkg/sbin/6to4.pl stop
   /usr/pkg/sbin/6to4.pl start
   ```

5. Para ter o routing IPv6 também na rede interna, pode-se indicar ao **6to4.pl** para fazer funcionar automaticamente o Router Advertising:

   ```
   # /usr/pkg/sbin/6to4 rtadvd-start
   ```

   Também este comando se pode por no script `/etc/ppp/ip-up` para torná-lo permanente.

6. Se mudamos o `/etc/ppp/ip-up` para acionar automaticamente o 6to4, convém mudar também `/etc/ppp/ip-down` para encerrá-lo quando se corta a conexão e se passa offline. Os comandos a inserir em `/etc/ppp/ip-down` são:

   ```
   logger -p user.info -t ip-down Shutting down 6to4 IPv6
   /usr/pkg/sbin/6to4.pl rtadvd-stop
   /usr/pkg/sbin/6to4.pl stop
   ```

10.2.4.9. Gateway 6to4

No momento não estão ainda disponíveis muitos gateways 6to4. Uma lista dos atualmente utilizáveis encontra-se em http://www.kfu.com/~nsayer/6to4/. Naturalmente convém escolher o mais vizinho. Nos testes parece que apenas 6to4.kfu.com e 6to4.ipv6.microsoft.com funcionaram. A Cisco tem um gateway junto ao qual é necessário registrar-se antes de poder usá-lo. Ver http://www.cisco.com/ipv6/.

Há ainda um servidor experimental 6to4 na Alemanha: 6to4.ipv6.fh-regensburg.de. Este servidor usa o NetBSD-1.5 e foi instalado com o uso do método descrito neste capítulo. A configuração da máquina pode-se ver no endereço http://www.feyrer.de/IPv6/netstart.local.

10.2.4.10. Conclusões e aprofundamentos

Se o confrontamos com o nível alcançado hoje pelo IPv4, o IPv6 ainda está dando os seus primeiros passos: funciona, muitos serviços e clientes já estão disponíveis mas ainda falta uma base de usuários. As informações contidas neste capítulo podem constituir um ponto de partida que permita um começo de experimentação e uso do IPv6.

Alguns links úteis:

- Um exemplo de script para instalar o 6to4 em máquinas BSD pode-se encontrar em http://www.netbsd.org/packages/net/6to4/. O script determina o endereço IPv6 da máquina e instala, se for desejado, o Router Advertising. O script foi projetado pensando nos ambientes com conexão dial-up, com endereço local que muda periodicamente.
- As implementações dos Unix BSD têm a própria documentação. Os links mais interessantes podem ser encontrados em http://www.netbsd.org/Documentation/network/ipv6/ para o NetBSD, http://www.freebsd.org/doc/en_US.ISO_8859-1/books/handbook/ipv6-implementation.html para o FreeBSD ver as páginas 61 e 62 do Administrator's Guide do BSD/OS em http://www.bsdi.com/products/internet/release-notes/rn42.pdf.
- Os projetos que trabalham para implementar os stacks do protocolo IPv6 são o KAME para o BSD e o USAGI para o Linux. Os sites correspondentes são http://www.kame.net/ e http://www.linux-ipv6.org/. Uma lista de hosts e routers pode ser encontrada em http://playground.sun.com/pub/ipng/html/ipng-implementations.html.

- Além do arquivo oficial RFC no endereço ftp://ftp.isi.edu/in-notes, podem-se encontrar informações sobre o IPv6 em muitos sites da web. O primeiro e mais importante é a página 6Bone em http://www.6bone.net/. O 6Bone foi criado para servir como banco de testes para o IPv6, e agora é uma parte importante do mundo dos usuários do IPv6. Outras páginas da web contendo informações sobre o IPv6 são http://www.ipv6.org/, http://playground.sun.com/pub/ipng/html/ e http://www.ipv6forum.com/. A maior parte destes sites, que vale a pena visitar, contêm outros links úteis.

Capítulo 11. O Domain Name System

Neste capítulo nos ocuparemos do DNS (Domain Name System). Utilizando como exemplo a rede descrita no Capítulo 10, será configurado um name server local.

No Capítulo 10 vimos como elaborar os arquivos de configuração de uma rede local simples. Um dos problemas a enfrentar era a tradução dos nomes dos hosts da rede para endereços IP e vice-versa. No contexto daquele capítulo o problema foi resolvido utilizando-se o arquivo `/etc/hosts`, em que eram indicadas as correspondências entre hosts e endereços IP, e que deveria estar presente (e atualizado) em cada um dos hosts da própria rede. Por exemplo:

```
192.168.1.1 ape.insetti.net ape
192.168.1.2 vespa.insetti.net vespa
```

Um outro arquivo, `/etc/nsswitch.conf`, indicava ao sistema a ordem em que consultar os recursos para resolver os nomes dos hosts.

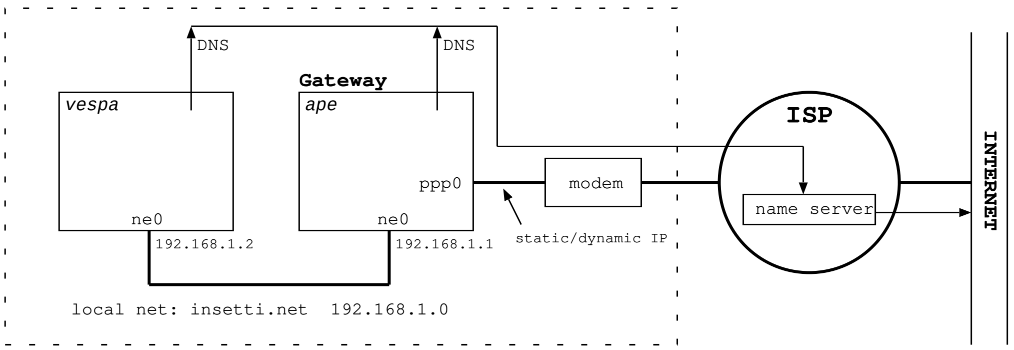
```
HOSTS:    files dns
```

Com este sistema o arquivo `/etc/hosts` era consultado primeiro e só em caso de malogro lançava-se mão do DNS. Uma vez que a rede não tinha seu próprio name server, eram utilizados os fornecidos pelo provedor, especificando-os no arquivo `/etc/resolv.conf` com a diretriz “nameserver”. Na prática o arquivo `/etc/hosts` era usado para a resolução dos nomes na rede local (não visível ao exterior) e os servidores DNS do provedor permitiam resolver os nomes pertencentes à Internet.

> **Nota:** com o termo *resolver* indica-se a conversão do nome em endereço IP e vice-versa.

A situação é ilustrada na Figura 11-1: ape é o gateway da rede e a diretriz nameserver dos arquivos `/etc/resolv.conf` de ambos os hosts apontam para o name server do provedor. Todas as interrogações DNS são endereçadas a este último. Os nomes dos hosts locais são resolvidos com o exame do arquivo `/etc/hosts`.

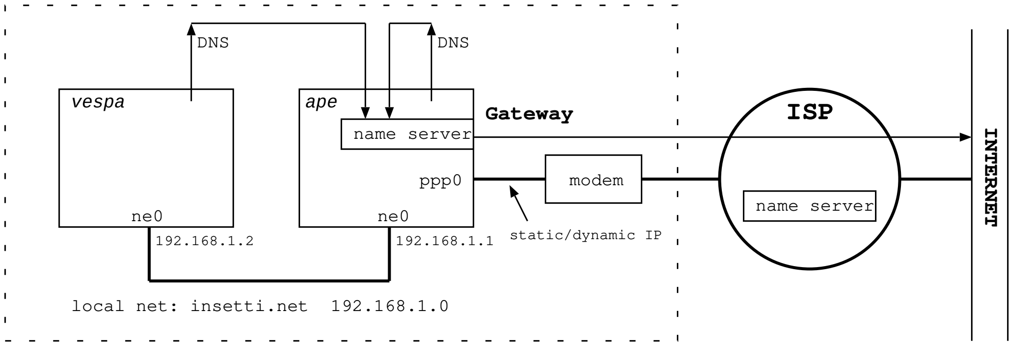
Figura 11-1. Uso do DNS do provedor



Neste capítulo veremos como preparar um servidor DNS local, sem necessidade de recorrermos aos servidores do provedor, que permita resolver todo tipo de nome, seja interno seja externo, e evitar o recurso ao arquivo hosts. Serão descritos dois tipos de abordagem, cujo significado será esclarecido no decorrer: um *caching server* e um servidor propriamente dito. O ambiente em que será configurado o name server será o mesmo daquele descrito no Capítulo 10.

A situação que se quer obter é ilustrada pela Figura 11-2. O name server encontra-se no host ape e é utilizado para resolver tanto os nomes internos quanto os externos. O arquivo `/etc/resolv.conf` do ape aponta para si mesmo, enquanto o do vespa aponta para o ape. O name server do provedor não será utilizado (exceto quando não usamos a diretriz *forwarders* ilustrada a seguir).

Figura 11-2. Uso do DNS interno



> **Nota:** podemo-nos perguntar quando é oportuno seguir a abordagem descrita no Capítulo 10 e quando convém utilizar um servidor DNS próprio. Como em muitos outros casos, não existe uma resposta unívoca. Em geral, para computadores isolados ou para casos de redes locais muito simples, como aquela descrita anteriormente, que se apoiam no servidor DNS de um provedor, convém seguir a abordagem já descrita, reservando o uso de um servidor DNS próprio para situações mais complexas. Ao término da leitura do capítulo deveria estar suficientemente claro quando é que o uso de um servidor local pode ser mais cômodo.

11.1. O que é o DNS?

Quando se faz referência a um host é decididamente mais confortável utilizar um nome do tipo www.netbsd.org que um endereço IP numérico como 110.111.112.113. Em um nível de comunicação mais baixo, entretanto, não são usados os nomes mas os endereços em forma numérica. Parece, pois, evidente, a exigência de um serviço de tradução dos nomes em endereços numéricos e vice-versa. O DNS (Domain Name Server) é precisamente isto (e ainda algumas coisas mais). Trata-se de um banco de dados hierárquico e distribuído. *Hierárquico* significa que o espaço dos nomes dos hosts não é “chato”, mas apresenta uma hierarquia dos nomes. Isso torna possível a existência de dois hosts com o mesmo nome, desde que pertençam a domínios diferentes. Se assim não fosse, poderia haver um só host de nome www conectado à Internet. Com uma estrutura hierárquica, www.netbsd.org e www.mydomain.org são dois hosts diferentes. *Distribuído* significa que o banco de dados não está inteiramente concentrado em um ponto (ou seja, não existe um host que conhece os nomes e os endereços de todos os outros hosts) mas “estilhaçado”: cada servidor possui informações relativas ao próprio domínio. Portanto, para poder desenvolver a sua tarefa é necessário que os servidores possam por-se em contato entre si para trocar informações.

O aplicativo de referência no que diz respeito ao gerenciamento do DNS é o BIND (Berkeley Internet Name Domain System), um software distribuído livremente pelo ISC (Internet Software Consortium) que implementa o protocolo DNS. O BIND faz parte da distribuição básica do NetBSD e, portanto, não é necessário instalar nenhum software adicional. A distribuição do BIND compreende vários componentes, entre os quais o servidor DNS, representado pelo dæmon named e alguns programas utilitários como host, dig, nslookup e dnsquery. Ademais, fazem parte da distribuição do BIND também as funções de biblioteca para a resolução das queries do DNS.

> **Nota:** na coleção dos pacotes geralmente tem uma versão mais recente do BIND que a instalada no sistema. O uso desta versão usualmente não é necessária, a menos que existam exigências específicas.

11.1.1. Como funciona uma interrogação do DNS?

Para efetuar uma interrogação do DNS, um aplicativo contata um servidor DNS que provê, por sua vez, à contatação dos servidores responsáveis pelo domínio ao qual se refere o nome a ser resolvido. Na realidade, a interrogação segue um percurso de tipo hierárquico, partindo dos servidores de nível mais alto, os assim chamados *root servers*, até chegar ao nível de detalhamento desejado. Por exemplo, suponhamos querer encontrar o endereço IP de www.netbsd.org. Primeiro o nosso name server contata um root server que fornece os endereços dos servidores para o domínio “org” (org é um dos TLD, ou seja, Top Level Domain, que são os domínios de nível mais alto como “edu”, “org”, “com”, etc.) Assim o nosso servidor contata um dos servidores para o domínio org, que por seu turno lhe fornece os endereços dos servidores para o domínio netbsd.org. Enfim o nosso servidor contata um dos servidores do netbsd.org para saber o endereço de www. Aqui conclui-se o ciclo de inquirições e respostas.

Este exemplo deveria ter esclarecido o que se entende por “hierárquico e distribuído”. O nome do host foi analisado de acordo com uma hierarquia precisa e as interrogações foram realizadas dirigindo-se em cada momento aos servidores que tinham conhecimento de cada uma das suas várias partes.

> **Nota:** note-se que para o DNS a ordem hierárquica dos nomes procede da direita para a esquerda, ou seja, em www.netbsd.org, org é o nível mais alto (o TLD, aquele imediatamente abaixo dos root servers); netbsd pertence ao domínio org e www pertence ao domínio netbsd.org.

11.2. Produzindo um caching server

Com o termo *caching server* entende-se um servidor que não gerencia em si mesmo a conversão dos nomes de host em endereços IP, mas sempre faz referência a outros servidores. A particularidade desse tipo de servidor é que memoriza as respostas que lhe chegam e se torna capaz, desse modo, de efetuar as interrogações posteriores mais velozmente. Já que um servidor desse tipo não gerencia os nomes internos da nossa rede, para obter os endereços daqueles nomes será ainda necessário fazer referência ao habitual arquivo `/etc/hosts`.

Como o NetBSD já fornece uma série de arquivos de configuração pré-definidos, a constituição de um servidor desse tipo é, para dizer pouco, elementar, como veremos em breve.

Nota: o conteúdo e o número dos arquivos de configuração pode variar de acordo com a versão do NetBSD.

O servidor propriamente dito é constituído pelo dæmon named, que faz referência a um arquivo de configuração que por default é `/etc/named.conf`. Exemplo desse arquivo de configuração no NetBSD pode ser encontrado no diretório `/etc/namedb` e, como primeiro passo, criemos um link ao arquivo:

```
# ln -s /etc/namedb/named.conf /etc/named.conf
```

O nosso name server está pronto para o uso. O passo seguinte será efetuar os testes para verificar-lhe o correto funcionamento mas, antes, devemos indicar ao sistema para usar o nosso servidor, inserindo no arquivo `/etc/resolv.conf` uma linha do tipo:

```
nameserver 127.0.0.1
```

Nesse momento tudo está pronto para inicializar o named.

```
# named
```

Nota: aqui o name server foi inicializado manualmente. Uma vez verificado o correto funcionamento é possível reinicializá-lo automaticamente no boot, agindo sobre a opção especificada do arquivo `/etc/rc.conf`.

11.2.1. Teste do servidor

Executamos o **nslookup** para examinar a situação e verificar o bom funcionamento do servidor.

```
# nslookup
Default server: localhost
Address: 127.0.0.1

>
```

Agora efetuemos uma inquirição a (a título de exemplo) www.mclink.it.

```
> www.mclink.it
Server:  localhost
Address:  127.0.0.1

Name:    www.mclink.it
Address:  195.110.128.8
```

Repetindo a interrogação uma segunda vez, obtém-se um resultado ligeiramente diferente:

```
> www.mclink.it
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
```

```
Name:    www.mclink.it
Address:  195.110.128.8
```

Como se vê, os dados são os mesmos mas agora apareceu a expressão “Non-authoritative answer”, que indica que o servidor não saiu pela rede para descobrir o endereço, mas utilizou os dados que tinha guardado no cache (depósito) depois da primeira interrogação. A resposta não se origina, portanto, do servidor mestre do domínio mclink.it, como no caso precedente, mas do cache do nosso servidor.

Estes primeiros resultados nos confortam pelo bom funcionamento do servidor.

Pode ser uma boa ocasião para experimentar também o comando **host**.

```
# host www.mclink.it
www.mclink.it has address 195.110.128.8
```

11.3. Um name server para o domínio local

No Secção 11.2 viu-se como estruturar um caching server, o que foi bastante simples. Experimentemos agora solicitar ao servidor algumas informações sobre um host da nossa rede local.

```
# nslookup
Default server: localhost
Address: 127.0.0.1

> ape.insetti.net
Server:  localhost
Address:  127.0.0.1

*** localhost can’t find ape.insetti.net: Non-existent host/domain
```

O servidor não sabe nada do nosso host local. De fato, os dados correspondentes ainda estão confinados no arquivo `/etc/hosts`, que não é consultado pelo servidor. Portanto o servidor de nomes efetua o habitual giro de interrogações partindo dos root servers, mas naturalmente não conclui nada. Isso não significa que a instalação do servidor esteja errada. Indica apenas que a resolução dos nomes locais poderá ser efetuada apenas por meio do arquivo `/etc/hosts`.

Na seqüência do capítulo será visto como fazer para o servidor de nomes gerenciar o nosso domínio local. Para fazer isso será necessário tanto modificar o `named.conf` como adicionar os *arquivos de zona*. Isto é, os arquivos que contêm os registros do DNS que descrevem os hosts (e outras características) da rede local. Na realidade, já estão presentes na configuração atual os arquivos de zona, que podem ser encontrados no diretório `/etc/named.db`. Sobre isto voltaremos em um segundo momento.

11.4. Os arquivos de configuração do named

O dæmon named lê dois tipos de arquivo de configuração. O primeiro, já citado, é o `named.conf`, que contém as opções de configuração do programa. O segundo tipo de arquivo é constituído pelos assim chamados arquivos de zona, que contêm os registros do DNS para os hosts da rede local. Os registros do DNS permitem transformar os nomes dos hosts em endereços IP e vice-versa, definir os servidores de correio e especificar muitas outras informações. Neste capítulo não serão descritos em detalhe nem o arquivo de configuração, nem os arquivos de zona. Vamo-nos limitar a apresentar e comentar exemplos que deveriam ser fáceis de modificar e adaptar.

> **Nota:** é possível que no arquivo `named.conf` fornecido com o NetBSD estejam presentes também outras seções, por exemplo IPv6.

11.4.1. Named.conf

O exemplo seguinte mostra uma configuração mínima para o arquivo `named.conf`. Grande parte do conteúdo já estava presente no arquivo por default do NetBSD. Apenas as linhas de 17 a 25 foram acrescentadas para gerenciar o domínio local.

> **Nota:** os números de linha e o caracter “|” no início de cada linha não fazem perte do arquivo e estão presentes apenas como referência.

Exemplo 11-1. named.conf

```
 1| # Created by carlo
 2|
 3| options {
 4|         directory "/etc/namedb";
 5| };
 6|
 7| zone "." {
 8|         type hint;
 9|         file "root.cache";
10| };
11|
12| zone "127.IN-ADDR.ARPA" {
13|         type master;
14|         file "127";
15| };
16|
17| zone "insetti.net" {
18|         type master;
19|         file "insetti.net";
20| };
21|
22| zone "1.168.192.IN-ADDR.ARPA" {
23|         type master;
24|         file "1.168.192";
25| };
```

Em uma primeira olhada já se nota que o arquivo está dividido em seções (ou mais precisamente statements) distintas, com os corpos das várias seções postos entre haspas. Vejamos agora em detalhe o significado das várias partes do arquivo (para uma descrição completa remete-se à página de manual named.conf(5).

Linha 1

As linhas que se iniciam com o caracter “#” são consideradas comentários e são ignoradas, assim como as linhas vazias. São reconhecidos também os comentários em estilo C e C++.

Linhas 3-5

A seção *options* controla as opções de configuração global do servidor e os defaults que serão usados nas outras seções do arquivo. A diretriz *directory* define o diretório de trabalho do servidor. Todos os percursos não absolutos presentes nas outras seções do arquivo de configuração serão interpretados como relativos a este diretório. Por exemplo, o arquivo `root.cache` (linha 7) será procurado em */etc/namedb/root.cache*.

Linhas 7-10

Esta seção define uma zona. Trata-se de uma zona especial, pré-definida, contendo os já citados root servers dos quais partem as buscas. A diretriz *hint* indica que esta zona é precisamente a dos root servers e a linha 7 especifica o nome do arquivo em que se encontram os dados da zona (o arquivo foi cortesmente fornecido pelo NetBSD).

Linhas 12-15

Esta seção define a zona inversa local para o endereço de loopback (127.0.0.1). Ver a explicação das linhas 22-25.

Linhas 17-20

Definição da zona relativa ao domínio local. A opção “type master” indica que o servidor é o *master server* para esta zona, ou seja, é aquele que tem a cópia master do arquivo de zona (no arquivo indicado na linha 19). O arquivo de zona contém o registro do DNS para efetuar a conversão do nome do host para o endereço IP, assim como outras informações que serão examinadas em seguida.

Note-se que o nome da zona deve corresponder ao nome do domínio, enquanto o nome do arquivo que contém o registro do DNS é livre, mesmo se normalmente convém ater-se a alguma convenção.

Linhas 22-25

Estas linhas definem a *zona inversa* para o domínio local. Isto é, o arquivo que contém o registro do DNS para efetuar a conversão de endereço IP para nome do host. Mesmo nesse caso o servidor é do tipo master. A linha 24 indica o nome do arquivo que contém o registro do DNS.

O nome da zona merece uma explicação. A primeira parte é o endereço IP da rede escrito ao contrário (192.168.1 torna-se 1.168.192), enquanto que a segunda refere-se ao domínio pré-definido “in-addr.arpa” que serve para a *resolução inversa*, isto é, a transformação dos endereços IP em nomes.

Server master e server slave: No exemplo precedente o nosso servidor era do tipo master para todas as zonas. Isto é, era o servidor que possuia a cópia master dos arquivos de zona. Um servidor de tipo slave, ao contrário, possui uma cópia (réplica) dos arquivos de zona do master correspondente (que não é escrita a mão pelo administrador do sistema, como a cópia master).

Entre o server master e o slave existem métodos de sincronização automática cuja descrição excede os objetivos dessa breve introdução.

A configuração do `named.db` terminou. Nas seções seguintes examinaremos o conteúdo dos arquivos de zona, descrevendo os registro do DNS. Os arquivos para as zonas "." e ".127" são pré-definidos e não exigem modificações. Portanto, não serão examinados. A sua estrutura é, mesmo assim, a mesma dos arquivos para o domínio local.

Antes de passar ao exame dos arquivos de zona, vejamos uma outra diretiva que pode ser inserida na seção options, ou seja: *forwarders*. Por exemplo:

```
options {
        directory "/etc/namedb";
        forwarders {
         1.2.3.4;
  1.2.3.5;
        };
};
```

No exemplo precedente suponha-se que 1.2.3.4 e 1.2.3.5 sejam o name server do provedor. O efeito desta configuração é que o nosso servidor efetuará as interrogações servindo-se dos name servers do provedor. Desse modo serão eles a se encarregar do caching das informações, ao invés do servidor local. No caso em que os forwarders não consigam responder a uma interrogação, então o servidor procurará encontrar a resposta por conta própria. Esta articulação reduz a carga de trabalho do servidor local, que se limita a gerenciar sozinho as zonas de que é master (em definitivo, o domínio local).

11.4.2. O arquivo para a zona insetti.net

Pelo modo como foi definido em `named.conf`, o arquivo para a zona insetti.net recebe o nome de `/etc/namedb/insetti.net`. O seu conteúdo é o seguinte:

```
1| ; Local hosts
 2|
 3|  @ IN SOA ape.insetti.net. hostmaster.ape.insetti.net. (
 4|     2000120301 ; Serial
 5|     3600  ; Refresh
 6|     300  ; Retry
 7|     3600000  ; Expire
 8|     3600 )  ; Minimum
 9|  IN NS ape.insetti.net.
10|  IN MX 10 ape
11| ape  IN A 192.168.1.1
12| vespa IN A 192.168.1.2
```

O arquivo é o verdadeiro banco de dados do DNS, contendo os *registros de DNS*. Na seqüência desta seção examinemos os registros presentes no arquivo.

Linha 1

Esta linha é um comentário.

Linhas 3-8

Este grupo de linhas define o registro SOA (Start Of Authority) que delimita o início de uma zona de autoridade. Cada zona tem um só registro SOA.

O caracter @ no início da linha é um sinônimo para o nome do domínio especificado na diretriz “zone” do arquivo `named.conf` e, assim, neste caso equivale a “insetti.net”.

O subseqüente “IN” indica estar na Internet e define o tipo de rede.

Depois da SOA vêm, na ordem, o name server e o endereço de e-mail do responsável pela zona (no endereço de e-mail o caracter @, que para o DNS tem um significado particular, deve ser substituído por um ponto).

> **Nota:** note-se que ambos os endereços precedentes terminam em um ponto. Todos os nomes não terminados em um ponto são considerados *relativos* ao domínio corrente e o servidor o completa acrescentando o domínio (@) no fim. Portanto, nos arquivos de zona é necessário prestar atenção em como se escrevem os nomes. Se, por exemplo, escrevemos “ape.insetti.net” (no lugar de “ape.insetti.net.”) o servidor modifica internamente o nome para “ape.insetti.net.insetti.net”, que certamente não é aquele que se quer obter.
>
> O registro SOA continua, delimitado por parênteses, até a linha 8. A linha 4 define o número de série do arquivo. O número pode ser escolhido à vontade, com a condição de que seja atualizado cada vez que se modifica o arquivo de zona. Nesse caso foi escolhida uma codificação que se baseia na data da última modificação do arquivo, expressa no formato ano-mês-dia seguido de um número de série. Por exemplo, se o arquivo for novamente modificado em 03/12/2000 o número ficará 2000120302. Se houver modificação dois dias depois, o número ficará 2000120501. É muito importante lembrar-se de atualizar esse número de série quando se modifica o arquivo.
>
> As linhas de 5 a 8 contêm os tempos em segundos que se referem à interação entre o master os seus slaves. Para os fins desta introdução não nos interessa examiná-la em detalhe e tomaremos por bons os valores definidos por default no exemplo.

Linha 9

A linha 9 é um registro do tipo *NS* que indica qual é o name server autorizado para a zona. Pode haver mais de uma linha do tipo NS. Note-se que o nome termina com um ponto.

Linha 10

Esta linha contém um registro do tipo *MX*, que contribui para acelerar a transferência da correspondência indicando qual é (ou quais são) os servidores de correio da zona. O número “10” que aparece depois de NS exprime uma preferência: quanto menor for o número, maior será a preferência dada ao servidor. No caso da presença de vários registros MX, é utilizado em primeiro lugar o que tem o número mais baixo. Não importa o valor do número. Conta apenas a relação maior-menor com os números dos outros registros MX. No exemplo seguinte o mail hub ape será provado antes de vespa.

```
IN MX  10 ape
 IN MX  20 vespa
```

Note-se que como depois do nome do host não aparece o ponto, ao próprio nome será articulado o domínio.

Linhas 11-12

Finalmente chegamos aos registros que são usados para transformar os nomes dos hosts em endereços IP (mas não vice-versa: essa é a tarefa do arquivo para a zona 1.168.192). Os dois registros de tipo *A* associam os endereços aos nomes dos dois hosts da nossa rede. Se estivessem presentes outros hosts seriam adicionados outros registros A. Note-se que os nomes dos hosts não são seguidos de ponto.

11.4.3. O arquivo para a zona 1.168.192.in-addr.arpa

Segundo o que foi definido no `named.conf`, o arquivo para a zona 1.168.192.in-addr.arpa chama-se `/etc/namedb/1.168.192`. O seu conteúdo é o seguinte:

```
1| @ IN SOA ape.insetti.net. hostmaster.ape.insetti.net. (
 2|    2000120301 ; Serial
 3|     3600  ; Refresh
 4|     300  ; Retry
 5|     3600000  ; Expire
 6|     3600 )  ; Minimum
 7|  IN NS ape.insetti.net.
 8| 1 IN PTR ape.insetti.net.
 9| 2 IN PTR vespa.insetti.net.
```

A primeira parte deste arquivo é de todo equivalente àquela do arquivo para a zona insetti.net e, portanto, não a examinaremos de novo. As únicas novidades são as linhas 8 e 9, que têm a função inversa ao registro A. Trata-se dos registros PTR, que permitem transformar os endereços IP em nomes de host. À luz de tudo o que foi dito até aqui, o seu significado deveria estar claro o suficiente. Note-se que os nomes dos hosts terminam em ponto.

No início da linha 8 aparece o número “1”. Já que não está seguido de um ponto, o DNS o concatena ao nome da zona (1.168.192.IN-ADDR.ARPA), obtendo 1.1.168.192.IN-ADDR.ARPA, que é o endereço completo a ser resolvido. O mesmo discurso vale para a linha 9, onde se obtém 2.1.168.192.IN-ADDR.ARPA.

In-addr.arpa: O domínio pré-definido (trata-se de um TLD) in-addr.arpa serve para transformar os endereços IP em nomes de host. No interior deste domínio os endereços IP devem ser escritos com as mesmas convenções dos nomes dos hosts, isto é, com a parte mais significativa à direita. Por esse motivo 192.168.1.2 torna-se 2.1.168.192.IN-ADDR.ARPA. Desse modo o BIND pode analisar esse nome exatamente como faz com os nomes dos hosts, procedendo da direita para a esquerda.

11.4.4. Outros tipos de registro para os arquivos de zona

Nos exemplos precedentes foram inseridos nos arquivos de zona os registros estritamente necessários para o funcionamento da configuração que nos tínhamos proposto. Na realidade, existem muitos outros tipos de registro que permitem especificar em maior detalhe e com maior flexibilidade a estrutura da nossa rede. Na seqüência desta seção, vejamos alguns dos outros tipos de registro. Para uma descrição mais completa e aprofundada, entretanto, remete-se à literatura especializada.

Os registros *CNAME* servem para criar codinomes (alias) de nomes existentes, ou seja, atribuir nomes adicionais a um host. Por exemplo:

```
www      IN       CNAME   ape
ftp      IN       CNAME   ape
```

Desse modo indicamos ao BIND que os nomes www e ftp fazem, ambos, referência ao host ape.

Os registros *TXT* servem para acrescentar texto descritivo aos registros de DNS. Por exemplo:

```
IN TXT "A minha rede local"
```

11.5. Inicializar o name server

O servidor se inicializa, como já foi explicado, pelo caching server. Se o servidor já estiver ativo, agora que os dados de configuração foram modificados é necessário reinicializá-lo, coisa que se efetua com o comando **ndc** (name dæmon control program).

```
# ndc reload
```

Agora que o name server está ativo, não é mais necessário inserir os nomes dos hosts no arquivo `/etc/hosts` porque pensamos que o named vai resolver os nomes. Naturalmente todos os hosts da rede devem `apontar` para o name server usando o arquivo `/etc/resolv.conf`. Particularmente em ape teremos:

```
nameserver 127.0.0.1
```

enquanto em vespa teremos:

```
nameserver 192.168.1.1
```

Capítulo 12. Correio e news

Neste capítulo é explicado como preparar o sistema para o o uso do correio eletrônico e das news. Trata-se de um tópico que, por causa da sua complexidade, se for tratado de forma geral, corre o risco de se tornar incompreensível para quem se aproxima pela primeira vez. Isto deve-se ao fato de que existe uma grande multiplicidade de situações possíveis (seja no que diz respeito às tipologias das conexões, seja no que se refere ao modo de trabalho) e que não se trata apenas de configurar um programa, mas de compor um sofisticado grupo de programas que cooperam para o funcionamento do sistema. Por esse motivo o tratamento da questão é ilustrado por um exemplo que descreve uma possível configuração para um PC de uso doméstico conectado à Internet através de um provedor, uma situação muito comum que, ainda que não apresente dificuldades específicas, pode-se revelar bastante espinhosa de se manejar quando não se sabe onde se deve por as mãos.

Além da preparação do sistema, descreve-se a configuração necessária para dois dentre os mais difundidos e apreciados programas de correio eletrônico e news: o mutt e o tin. Os programas em si mesmos não são descritos, até porque são simples de usar e bem documentados. Naturalmente, tanto o mutt quanto o tin não são escolhas obrigatórias e o usuário é livre para instalar os aplicativos que preferir. Em particular, quem não gosta de programas em modo texto, pode encontrar alternativas válidas no X.

Em resumo, os programas exigidos pela configuração descrita neste capítulo são:

- sendmail
- fetchmail
- m4
- mutt
- tin

Destes, apenas o sendmail e o m4 já estão presentes no sistema básico. Os outros podem ser instalados a partir da coleção de pacotes.

Antes de prosseguir, gostaria de lembrar ainda uma vez que nenhum dos programas apresentados neste capítulo é uma escolha obrigatória. Existem numerosas outras alternativas para cada um deles mas, mesmo assim, constituem um bom ponto de partida. Também a estratégia utilizada para o envio e recebimento de correspondência não é único. Cada um pode modificá-la segundo a própria configuração e as próprias exigências.

No extremo oposto dos programas e da estratégia ilustrados neste capítulo há o uso de um programa faz-tudo como o Netscape Communicator. Com esse tipo de programa pode-se navegar na Internet, receber e enviar mensagens e ler artigos nos newsgroups. Além disso, a sua configuração é muito simples. A desvantagem desse tipo de abordagem é que o Netscape é um sistema fechado, que não coopera com os outros programas do NetBSD.

Uma outra abordagem muito seguida é a de ler a correspondência e as news utilizando-se dos recursos específicos do emacs. O emacs certamente não tem necessidade de apresentação para os apaixonados pelo Unix mas, para quem não saiba, trata-se de um editor avançado (na verdade, chamá-lo de editor é um pouco reducionista) que se transforma em um verdadeiro ambiente de trabalho dentro do qual é possível ler a correspondência e as news, além de efetuar muitas outras operações. A configuração do emacs para correio e news está descrita no manual do programa.

Na seqüência deste capítulo será feita referência a um host ligado à Internet através de uma conexão serial PPP via modem com um provedor. Os dados para a configuração são:

- o host local chama-se "ape" e a rede é "insetti.net", para a qual o FQDN (Fully Qualified Domain Name) é "ape.insetti.net".
- O nome do usuário do ape no exemplo é "carlo".
- O provedor utilizado chama-se "BigNet".
- O servidor de correio do provedor é "mail.bignet.it".
- O nome de login do usuário "carlo" junto ao provedor é "alan" e a sua senha é "pZY9o".

Antes de começar, um pouco de terminologia:

MUA

(mail user agent): programa que permite escrever e ler a correspondência. Por exemplo: mutt, elm e pine, mas também o simples mail pré-instalado.

MTA

(mail transfer agent): programa que responde pelo encaminhamento da correspondência, transferindo-a de um computador a outro e localmente. O MTA é o programa que decide o percurso que a correspondência deve seguir para chegar ao seu destino. Nos sistemas BSD (mas não só) o padrão é o sendmail.

MDA

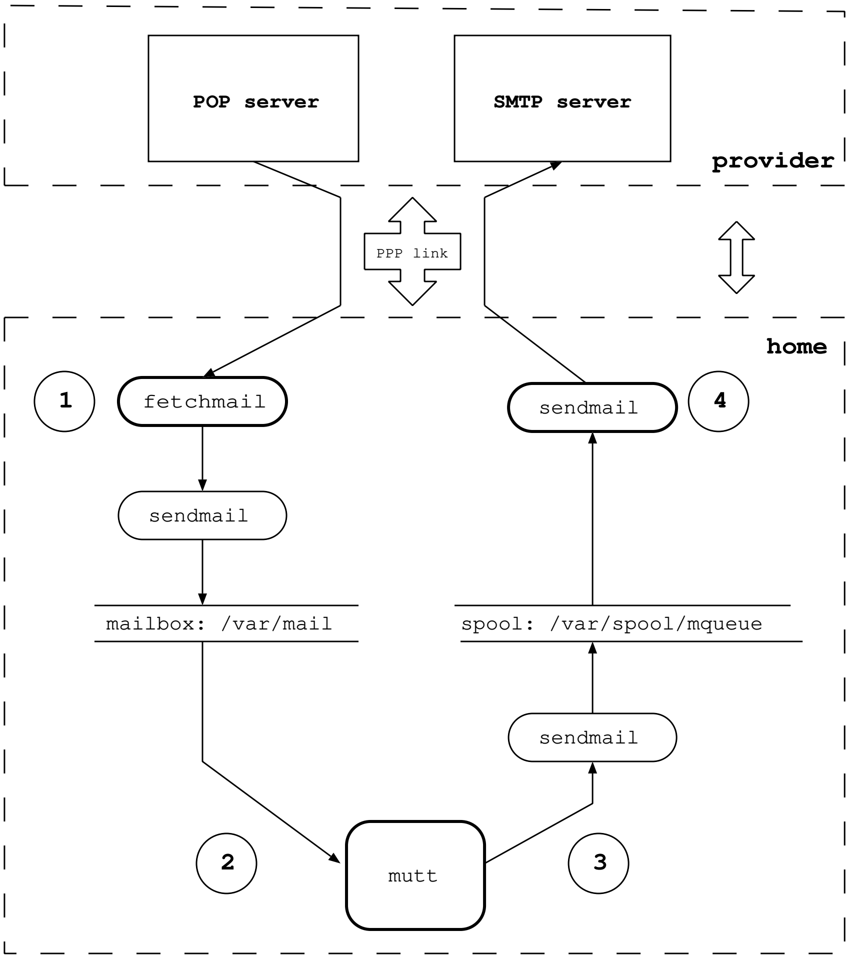
(mail delivery agent): programa, geralmente usado pelo MTA, que serve para entrega da correspondência. Por exemplo, para inserir fisicamente as mensagens nas caixa postal do usuário destinatário. O sendmail, por exemplo, utiliza um ou mais MDAs para fazer a entrega da correspondência.

A Figura 12-1 ilustra o sistema de correspondência que se quer organizar. Entre a rede doméstica (ou mesmo um host isolado) e o provedor há uma conexão PPP via modem. Os "balões" de borda larga referem-se a comandos executados explicitamente pelo usuário. Os "balões" restantes são os programas executados automaticamente. Os números nos círculos referem-se às várias fases lógicas do processo de gerenciamento do ciclo da correspondência:

1. Na fase (1) a correspondência é tirada (por download) do servidor POP do provedor mediante o fetchmail, que usa o sendmail para pô-la na caixa postal do usuário.
2. Na fase (2) o usuário executa o mutt (ou um outro cliente de sua escolha) para ler a correspondência, responder às mensagens recebidas e escrever outras novas.
3. Na fase (3) o usuário envia a correspondência pelo mutt. As mensagens são acumuladas na área da spool (lançadeira).
4. Na fase (4) o usuário chama o sendmail para transferir as mensagens para o servidor SMTP do provedor de onde serão expedidas aos destinatários.

A conexão com o provedor dever estar ativa somente durante as fases (1) e (4). Para as restantes operações pode ser desativada tranqüilamente.

Figura 12-1. Estrutura do sistema de correspondência



12.1. sendmail

Quando um MTA, o sendmail no caso, recebe uma mensagem de correio para expedir, em se tratando de uma mensagem local, expede-a diretamente. Do contrário, se a mensagem está destinada a um outro domínio, deve procurar o endereço do servidor de correio eletrônico do domínio em questão. Para fazer isso o MTA apoia-se no serviço de DNS (descrito no Capítulo 11). O servidor DNS do domínio de destinação conhece o endereço do servidor de correio (record MX) e o fornece ao MTA que, nesse ponto, pode expedir a mensagem ao servidor destinatário.

O *MTA* mais difundido, pelo menos no ambiente BSD, é o sendmail. O comportamento do sendmail é controlado pelo arquivo de configuração `/etc/mail/sendmail.cf` e por uma série de arquivos acessórios. Em geral, a menos que se saiba o que se está fazendo, não é aconselhável escrever ou modificar este arquivo diretamente. O arquivo de configuração pode ser automaticamente gerado com o uso de um conjunto de macros M4 que facilita muito a tarefa.

> **Nota:** nas versões do NetBSD precedentes à 1.5, os arquivos de configuração do sendmail encontravam-se no `/etc` ao invés do `/etc/mail`.

Mesmo utilizando as macros, a configuração do sendmail é um tópico bastante hostil (para dizer pouco) e na seqüência limito-me a mostrar um exemplo básico, que pode ser usado como ponto de partida e modificado para gerir a própria correspondência frente a configurações diferentes. Para uma conexão à Internet via modem, o arquivo apresentado pode ser usado tal qual, sem necessidade de qualquer modificação (salvo naturalmente inserir os próprios dados no lugar dos que estão no exemplo).

O primeiro problema a resolver é que a rede doméstica é totalmente fictícia. Ou seja, não existe realmente na Internet e, portanto, os nomes e os endereços internos não têm sentido para o mundo externo. Em suma, o host "ape.insetti.net" não é alcançável por nenhum outro host da Internet e, se enviamos um e-mail com o nome local do host no remetente, ninguém poderá responder (não somente. Alguns destinatários rejeitarão a mensagem porque o remetente não existe). O verdadeiro endereço, aquele que é visível para todos, é o designado pelo provedor "alan@bignet.it". Disso se vai ocupar o sendmail, devidamente configurado, no momento de transferir a correspondência.

O segundo aspecto a considerar é que convém instruir o sendmail para fazê-lo enviar a correspondência ao servidor de mail do provedor. Na configuração descrita nesta seção, de fato, o sendmail não se põe diretamente em contato com os servidores de correio dos destinatários das nossas mensagens (como decrito no início desta seção), mas se apoia somente no servidor do provedor, utilizando-o como *relay*. Isso torna muito mais veloz o envio da correspondência, transferindo todo o trabalho de busca e de conexão ao servidor de mail do provedor.

> **Nota:** é dito que um servidor de correio age como *relay* quando se encarrega de encaminhar a correspondência que não é de sua competência direta. Neste caso o servidor de correio relay age como intermediário, providenciando o encaminhamento da correspondência ao servidor destinatário (ou a um outro servidor relay).

Uma vez que a conexão com o provedor não está sempre ativa, pode-se desabilitar a inicialização do sendmail como dæmon no `/etc/rc.conf` com a linha 'sendmail=NO'. A correspondência local será mesmo assim expedida corretamente, enquanto que para transferir correspondência ao provedor será necessário executar o sendmail manualmente (quando a conexão com o provedor estiver ativa, naturalmente).

Comecemos pois o trabalho de configuração do sendmail.

12.1.1. Configuração com genericstable

Este tipo de configuração utiliza o arquivo `/etc/mail/genericstable` que contém as regras explícitas para se reescrever os endereços internos à rede local.

A primeira coisa, portanto, será criar o arquivo `genericstable`. Por exemplo:

```
carlo:          alan@bignet.it
root:           alan@bignet.it
news:           alan@bignet.it
```

Por motivos de eficiência o genericstable deve ser transformado com o comando:

```
# /usr/sbin/sendmail -bi -oA/etc/mail/genericstable
```

Agora é o momento de criar o protótipo que será utilizado para gerar o novo arquivo de configuração do sendmail. A primeira coisa a fazer é deslocar-se para o diretório em que será criado o novo arquivo de configuração:

```
# cd /usr/share/sendmail/m4
```

Suponhamos um protótipo chamado `mycf.mc`. O conteúdo do arquivo é o seguinte:

```
divert(-1)dnl
include(`../m4/cf.m4')dnl
VERSIONID(`mycf.mc criado por carlo@ape.insetti.net May 18 2001')dnl
OSTYPE(bsd4.4)dnl

dnl # Definições para masquerading.  São reescritos os
dnl # endereços do tipo
dnl #     carlo@ape.insetti.net
dnl #     carlo@ape
GENERICS_DOMAIN(ape.insetti.net ape)dnl
FEATURE(genericstable)dnl
FEATURE(masquerade_envelope)dnl

define(`SMART_HOST',`mail.bignet.it')dnl

FEATURE(redirect)dnl
FEATURE(nocanonify)dnl

dnl # A característica (feature) seguinte serve sobretudo se o sendmail for usado
dnl # junto com o fetchmail. Fetchmail chama o sendmail para expedir a
dnl # correspondência. Se o sendmail não consegue resolver o nome do emitente
dnl # a correspondência são é enviada. Por exemplo:
dnl # MAIL FROM:<www-owner@netbsd.org>... Sender domain must exist
FEATURE(`accept_unresolvable_domains')dnl

dnl # accept_unqualified_senders satisfaz alguns MUA, que enviam
dnl # a correspondência como
dnl # MAIL FROM:<carlo>
FEATURE(`accept_unqualified_senders')dnl

dnl # A correspondência para o mailer 'smtp' é posta na fila: o sendmail não
dnl # tenta enviá-la diretamente (a flag 'e' indica 'expensive').
dnl # O sendmail começa a lamentar-se depois de 16 horas de expedição frustrada.
define(`SMTP_MAILER_FLAGS',`e')dnl
define(`confCON_EXPENSIVE',`True')dnl
define(`confTO_QUEUEWARN', `16h')dnl

dnl # Para os usuários europeus
define(`confDEF_CHAR_SET',`ISO-8859-1')dnl

dnl # Habilitar as três linhas seguintes para utilizar o procmail para
dnl # o envio da correspondência local (a terceira linha é opcional,
dnl # somente as duas primeiras são necessárias)
dnl # define(`PROCMAIL_MAILER_PATH', /usr/pkg/bin/procmail)dnl
dnl # FEATURE(local_procmail)dnl
```

```
dnl # MAILER(procmail)dnl

dnl # Os dois mailers seguintes devem ser sempre definidos
MAILER(local)dnl
MAILER(smtp)dnl
```

> **Nota:** no exemplo precedente as linhas que se iniciam com “dnl” são comentários e, portanto, não são necessárias para a geração do arquivo de configuração (a sigla “dnl” indica ao m4 para ignorar o resto da linha e aparece também após as várias diretrizes com o fim de se evitar efeitos indesejados).
>
> As linhas relativas ao uso do procmail são deixadas comentadas (ou seja, o uso do procmail fica desabilitado).

Esta configuração aponta para a transformação dos endereços do tipo "ape.insetti.net" nos nomes reais, através de busca no arquivo `/etc/mail/genericstable`. Além disso fica especificado que a correspondência deva ser enviada ao host "mail.bignet.it". O significado das opções está descrito em detalhe no arquivo `/usr/share/sendmail/README`.

Para criar o próprio arquivo personalizado é suficiente inserir os próprios dados modificando apenas duas linhas do exemplo precedente, isto é:

```
GENERICS_DOMAIN(ape.insetti.net ape)dnl
define('SMART_HOST','mail.bignet.it')dnl
```

Geremos o novo arquivo de configuração, salvando antes o arquivo anterior.

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.bak
# m4 mycf.mc > /etc/mail/sendmail.cf
```

> **Nota:** em `/usr/share/sendmail/cf` fica o arquivo `netbsd-proto.mc`, que é utilizado para criar a versão `/etc/mail/sendmail.cf` que é distribuída por default com o NetBSD. Jamais deveria ser necessário, mas com o comando **make** podemos construí-lo.

Um outro arquivo importante é o `/etc/mail/aliases` que, todavia,pode ser mantido com a configuração original. Mesmo assim é necessário executar o comando:

```
# newaliases
```

Agora está tudo pronto para expedir a correspondência.

12.1.2. Teste da configuração

Agora que a configuração do sendmail está completa, pode-se proceder a algumas verificações simples para confirmarmos se tudo funciona corretamente. O primeiro teste poderá ser expedir uma mensagem local:

```
$ sendmail -v carlo
Subject: teste

Prova
.
carlo... Connecting to local...
carlo... Sent
```

Seguir o esquema do exemplo, deixando uma linha vazia depois de Subject: e concluindo a mensagem com uma linha que contém só um ponto. A mensagem enviada deveria ser legível utilizando-se o cliente de correio eletrônico. Em particular, verifique-se o conteúdo do campo

```
From: alan@bignet.it
```

no qual deve constar a reescrita do endereço.

Uma segunda verificação importante é o teste das regras de reescrita definidas no arquivo de configuração do sendmail. Para fazer isso executa-se o sendmail em modo de teste: o sendmail lê um endereço e mostra todas as passagens realizadas para transformá-lo. Nesse modo podem-se também efetuar outros tipos de teste, assim como visualizar várias informações.

Inicializar o sendmail em modo de teste usando a opção `-bt`

```
$ /usr/sbin/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Para ver a ajuda, utilizar o comando **’?’**.

Em primeiro lugar verifiquemos o funcionamento correto do arquivo genericstable dando o seguinte comando:

```
/map generics carlo
map_lookup: generics (carlo) returns alan@bignet.it
```

Até aqui tudo bem. O sendmail encontrou o nome local e o seu correspondente real no mapa generics.

Agora verifiquemos a reescrita do remetente no envelope executando os comandos:

```
/tryflags ES
/try smtp carlo@ape.insetti.net
```

O resultado deveria ser similar ao seguinte:

```
Trying envelope sender address carlo@ape.insetti.net for mailer smtp
rewrite: ruleset   3   input: carlo @ ape . insetti . net
rewrite: ruleset  96   input: carlo < @ ape . insetti . net >
rewrite: ruleset  96 returns: carlo < @ ape . insetti . net . >
rewrite: ruleset   3 returns: carlo < @ ape . insetti . net . >
rewrite: ruleset   1   input: carlo < @ ape . insetti . net . >
rewrite: ruleset   1 returns: carlo < @ ape . insetti . net . >
rewrite: ruleset  11   input: carlo < @ ape . insetti . net . >
rewrite: ruleset  51   input: carlo < @ ape . insetti . net . >
```

```
rewrite: ruleset  51 returns: carlo < @ ape . insetti . net . >
rewrite: ruleset  61   input: carlo < @ ape . insetti . net . >
rewrite: ruleset  61 returns: carlo < @ ape . insetti . net . >
rewrite: ruleset  94   input: carlo < @ ape . insetti . net . >
rewrite: ruleset  93   input: carlo < @ ape . insetti . net . >
rewrite: ruleset   3   input: alan @ bignet . it
rewrite: ruleset  96   input: alan < @ bignet . it >
rewrite: ruleset  96 returns: alan < @ bignet . it >
rewrite: ruleset   3 returns: alan < @ bignet . it >
rewrite: ruleset  93 returns: alan < @ bignet . it >
rewrite: ruleset  94 returns: alan < @ bignet . it >
rewrite: ruleset  11 returns: alan < @ bignet . it >
rewrite: ruleset   4   input: alan < @ bignet . it >
rewrite: ruleset   4 returns: alan @ bignet . it
Rcode = 0, addr = alan@bignet.it
>
```

Como se pode ver o endereço local foi reescrito como endereço real, que aparecerá nas mensagens quando deixarem o sistema.

Um resultado equivalente pode-se obter com o comando:

```
/try smtp carlo
```

Para verificar a reescrita do remetente no cabeçalho, repitamos os comandos precedentes mudando as flags. O resultado é equivalente.

```
/tryflags HS
/try smtp carlo@ape.insetti.net
```

12.1.3. Usando um outro MTA

A partir da versão 1.4 do NetBSD o sendmail não é chamado de modo direto:

```
$ ls -l /usr/sbin/sendmail
lrwxr-xr-x  1 root  wheel  21 Nov  1 01:14 /usr/sbin/sendmail@ -> /usr/sbin/mailwrapper
```

O objetivo do mailwrapper é simplificar o uso dos MTAs alternativos ao sendmail (por exemplo, o postfix) mas, para se ter uma idéia precisa do que se trata, aconselho a leitura das páginas de manual mailwrapper(8) e mailer.conf(5), que são muito claras.

12.2. fetchmail

A correspondência chega ao endereço, depositando-se no provedor. Nesse momento é necessário transferí-la (fisicamente) para o computador local. O fetchmail é um programa que captura a correspondência de um servidor de correio remoto e a encaminha ao sistema local para entrega (geralmente através do sendmail). O fetchmail é simples de usar e de configurar. Depois de instalá-lo basta criar o arquivo `.fetchmailrc` no próprio diretório home (dentro vai a senha; atenção às permissões...). Eis um exemplo:

```
poll mail.bignet.it
protocol POP3
username alan there with password pZY9o is carlo here
flush
mda "/usr/sbin/sendmail -oem %T"
```

> **Nota:** a linha "mda ..." serve apenas se o sendmail não estiver ativo como um dæmon do sistema. Ademais, note-se que o servidor de correio especificado neste arquivo poderia ser diferente do relay usado pelo sendmail.

Para transferir a correspondência basta executar o comando:

```
$ fetchmail
```

e depois podemos lê-la com o mutt.

12.3. Correspondência com o mutt

O mutt é um programa de correio que recebe muitas adesões tanto porque é *rápido* quanto porque é simples de usar e poderoso ao mesmo tempo. O mutt tem uma página de manual muito enxuta. A verdadeira documentação encontra-se em `/usr/pkg/share/doc/mutt/`, em particular no `manual.txt`.

A configuração do mutt se faz através do arquivo `.muttrc`, que deve ser criado no próprio diretório home (usualmente copiando o arquivo de exemplo fornecido com o mutt e depois modificando-o). O exemplo à frente mostra como modificar algumas linhas do arquivo para obter os seguintes efeitos:

- Salvar as mensagens enviadas.
- Definir um diretório para a correspondência recebida e enviada salva pelo mutt (será posta no subdiretório `~/Mail` nos arquivos `incoming` e `outgoing`).
- Definir algumas cores.
- Definir um codinome (alias).

```
set copy=yes
set edit_headers
set folder="~/Mail"
unset force_name
set mbox="~/Mail/incoming"
set record="~/Mail/outgoing"
unset save_name

bind pager <up> previous-page
bind pager <down> next-page
```

```
color normal white black
color hdrdefault blue black
color indicator white blue
color markers red black
color quoted cyan black
color status white blue
color error red white
color underline yellow black

mono quoted standout
mono hdrdefault underline
mono indicator underline
mono status bold

alias pippo Pippo Verdi <pippo.verdi@pluto.net>
```

Para executar o mutt é suficiente executar o comando

```
$ mutt
```

> **Nota:** de acordo com o terminal o mutt suporta o uso de cores diversas para os diversos elementos de sua tela. Sob X pode-se usar o xterm-color. Por exemplo:
>
> ```
> TERM=xterm-color mutt
> ```

12.4. Como receber a correspondência

Esta seção descreve uma estratégia simples para receber e ler a correspondência. A conexão com o provedor é ativada apenas durante o tempo necessário à transferência da correspondência, que depois pode ser lida *offline*.

1. Ativar a conexão com o provedor.
2. Executar o comando **fetchmail**.
3. Fechar a conexão com o provedor.
4. Ler a correspondência com o mutt.

12.5. Como enviar a correspondência

Uma vez escrita e enviada a correspondência com o mutt, deve-se remetê-la ao servidor do provedor com o sendmail. O envio da correspondência com o mutt, que se efetua com o comando **y**, limita-se a enfileirar as missivas na spool (lançadeira). Neste ponto, se o sendmail não está ativo como dæmon, é preciso tomar a iniciativa de invocá-lo explicitamente. Os passos a seguir são:

1. Escrever a correspondência com o mutt, executar o comando de envio e sair do mutt.
2. Ativar a conexão com o provedor.
3. Executar o comando **/usr/sbin/sendmail -a -v** para transferir a correspondência ao provedor.
4. Fechar a conexão com o provedor.

12.6. Ferramentas avançadas para correio eletrônico

Quando se começa a usar o correio eletrônico, geralmente não se tem exigências muito sofisticadas e, portanto, a configuração padrão descrita acima é mais que suficiente. Com o passar do tempo, entretanto, para muitos usuários o volume diário de correspondência cresce e se diferencia. Nessa fase torna-se necessário organizar racionalmente as mensagens, separando-as em “caixinhas” diferentes, e nos damos conta de precisar realizar um grande número de operações manuais repetitivas todo dia para classificar a correspondência. Por exemplo, quem se inscreve em uma *mailing list* pode encontrar-se na situação de ter que selecionar cotidianamente um notável número de mensagens para transferí-las da mailbox geral para aquela mailbox que criamos para dedicar-se apenas à *mailing list*.

Porque efetuar à mão operações que podem ser agilmente desenvolvidas por um programa? Existem vários instrumentos complementares que se inserem no fluxo normal da correspondência e permitem realizar várias operações sobre a mesma, com base em configurações pré-fixadas pelo usuário. Entre as ferramentas mais conhecidas e usadas encontramos:

- procmail, um mail delivery agent e filtro avançado para a correspondência local, que permite elaborar automaticamente a correspondência com base em regras pré-estabelecidas pelo usuário redefinindo-lhe, por exemplo, a destinação. O melhor, contudo, é que ele se integra completamente com o sendmail.
- metamail, um instrumento para elaborar os anexos (attachments) contidos nas mensagens de correio eletrônico.
- formail, um programa para reformatar as mensagens de correio.

Na seqüência desta seção será descrito um exemplo de uso do procmail, sem todavia descer aos detalhes da configuração e das características do programa, já que se trata de um instrumento que comporta imensa variedade de usos. Limitar-nos-emos, assim, a descrever a configuração do sendmail e do procmail para um caso muito comum: a separação da correspondência proveniente de uma mailing list. Para esse tipo de configuração far-se-á de modo que o sendmail utilize o procmail como um mailer local. O procmail, por sua vez, lerá um arquivo de configuração especificado para conter as regras de separação da correspondência.

Em primeiro lugar é preciso instalar o procmail, recorrendo ao sistema de pacotes (`mail/procmail`).

Depois é necessário configurar o sendmail para que use o procmail. Para fazer isso é suficiente tirar o comentário das linhas relativas ao procmail no arquivo de configuração do sendmail já descrito:

```
define(‘PROCMAIL_MAILER_PATH’, /usr/pkg/bin/procmail)dnl
FEATURE(local_procmail)dnl
MAILER(procmail)dnl
```

A primeira linha especifica o percurso do programa procmail e deve ser alterada segundo o local em que ele estiver instalado (o que pode ser determinado com o comando **which procmail**). A segunda linha indica ao sendmail para usar o procmail como *mailer* local. A terceira adiciona o procmail à relação dos mailers do sendmail (esta linha é opcional).

Enfim cria-se o arquivo de configuração do procmail, `.procmailrc`, nos próprios diretórios home. Este arquivo contém as regras para a expedição da correspondência em caráter local e é utilizado pelo programa para decidir a destinação das mensagens.

A título de exemplo, imagine-se estar inscrito em uma mailing list sobre rosas, cujo endereço é `<roses@flowers.org>`. Suponhamos, além disso, que todas as mensagens que chegam desta mailing list contenham, no cabeçalho, a espressão

```
Delivered-To: roses@flowers.org
```

O arquivo de configuração do procmail tem um aspecto do seguinte tipo:

```
PATH=/bin:/usr/bin:/usr/pkg/bin
MAILDIR=$HOME/Mail
LOGFILE=$MAILDIR/from

:0
* ^Delivered-To: roses@flowers.org
roses_list
```

O arquivo precedente contém uma só regra de redistribuição, que começa com a linha ":0" que o usuário terá oportunamente criado no diretório $MAILDIR. Note-se que o $MAILDIR é o mesmo diretório indicado na diretriz do arquivo de configuração do Mutt.

```
set folder="~/Mail"
```

As mensagens provenientes da mailing list são identificadas pela linha:

```
* ^Delivered-To: roses@flowers.org
```

contida no cabeçalho. As mensagens remanescentes irão para a mailbox geral.

A mailing list é naturalmente apenas um exemplo, e o procmail pode ser usado para redistribuir qualquer tipo de mensagem com base, por exemplo, na proveniência, no conteúdo, etc.

Para assegurar que o procmail seja usado como um mailer local pelo sendmail, pode-se executar este último em modo de teste, da seguinte maneira:

```
$ /usr/sbin/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Agora se invoca a lista dos mailers reconhecidos pelo sendmail com o comando

```
> =M
```

No output deveria aparecer uma linha desse tipo:

```
mailer 3 (local): P=/usr/pkg/bin/procmail S=EnvFromL/HdrFromL ...
```

Como sempre, para ulteriores detalhes e para se ter uma idéia mais precisa das potencialidades do procmail, consultar as páginas de manual procmail(1), procmailrc(5) e procmailex(5), contendo, a última, arquivos de configuração.

12.7. Como verificar um endereço de correio eletrônico

Antes de passar aos *newsgroups* decidí inserir este pequeno "truque" que pode ser útil com o correio: trata-se de verificar interativamente a veracidade de um endereço de e-mail.

> **Nota:** nem todos os servidores aceitam o comando VRFY, que pode ser desabilitado.

Suponhamos querer verificar o usuário carlo@dominio.com. O método é o seguinte:

```
$ telnet dominio.com smtp
Trying aa.bb.cc.dd...
Connected to dominio.com.
Escape character is '^]'.
220 dominio.com ESMTP Sendmail 8.9.3/8.9.3; Fri, 30 Apr 1999
VRFY carlo
250 Carlo Rossi
VRFY abcde
550 abcde... User unknown
^]
telnet> quit
Connection closed.
```

12.8. News com tin

Com o termo *news* indica-se o conjunto dos artigos dos newsgroups de USENET, um serviço disponível na Internet. Cada newsgroup recolhe artigos relacionados com um tópico específico. O mecanismo de leitura dos artigos é diferente em relação às mailing lists. Quando nos inscrevemos em uma mailing list, os artigos são-nos enviados pelo correio e para lê-los (e respondê-los) usa-se um programa de correio (como, por exemplo, o mutt. A consulta das news, por outro lado, ocorre de modo direto, utilizando um programa chamado *newsreader* como, por exemplo, o tin, que permite que se faça a inscrição nos grupos que nos interessam e seguir-lhes os *threads (enredos, encadeamentos)*.

> **thread:** um thread é uma seqüência de mensagens (réplicas e tréplicas) derivadas de uma mensagem que poderemos definir como "inicial". Na prática, qualquer um envia uma mensagem ao grupo, outros enviam sua resposta à mensagem original, outros ainda (talvez os mesmos) respondem àqueles que responderam e assim por diante, até criar uma estrutura ramificada de mensagens e respostas. Sem um newsreader é impossível desembaraçar-se do emaranhado (leia-se "seqüências lógicas") dos threads.

Uma vez instalado o programa, a única coisa a fazer é estabelecer o nome do servidor de NNTP, escrevendo-o no arquivo `/usr/pkg/etc/nntp/server`. Por exemplo;

```
news.bignet.it
```

Feito isso, pode-se executar o programa com o comando **rtin**. Na tela vai aparecer uma série de escritos desse tipo:

```
$ rtin
Connecting to news.bignet.it...
news.bignet.it InterNetNews NNRP server INN 1.7.2 08-Dec-1997 ready (posting ok).
Reading groups from active file...
Checking for new groups...
Reading attributes file...
Reading newsgroups file...
Creating newsrc file...
Autosubscribing groups...
Reading newsrc file...
```

É necessário ter paciência quando se conecta pela primeira vez, porque é descarregada uma lista interminável de newsgroups nos quais podemos nos inscrever: a operação dura vários minutos. Terminado o descarregamento aparece a tela do programa, geralmente vazia. Para ver a lista dos grupos pressione-se a tecla **y**. Para subscrever-se em um grupo, colocar-se sobre o nome do grupo e pressionar **s**.

Para acelerar as inicializações posteriores pode-se usar o comando **rtin -Q**. Desse modo evita-se a busca de novos newsgroups, carregam-se apenas os grupos ativos (`-n`) e não se carregam as descrições dos grupos (`-d`). Naturalmente não será possível usar o comando **y** (yank) durante a sessão do tin. Quando se inicia o tin desse modo, o programa não é capaz de saber quais newgroups são regulares.

> **Nota:** se nos conectamos à Internet através de um provedor com um computador que não pertence pois a um “verdadeiro” domínio, quando se envia a mensagem o endereço no cabeçalho será errado (porque inexistente). Para resolver o problema pode-se usar a opção “mail_address” no arquivo de configuração do tin (`~/.tin/tinrc`) ou se pode definir a variável de ambiente REPLYTO.

Capítulo 13. Driver de console

Nas versões anteriores à 1.4, para gerenciar a tela do monitor e o teclado podia-se escolher entre pccons (específico para i386) e pcvt. Na versão 1.4 foi introduzido o novo driver wscons que, com o tempo, destinava-se a substituir completamente os dois primeiros.

13.1. wscons

wscons é o novo driver de console do NetBSD que deveria substituir pccons e pcvt. Oferece terminais virtuais, suporte para teclados internacionais, gerenciamento do mouse, etc. As capacidades do wscons alteram-se segundo a arquitetura. O i386 é o mais rico de funcionalidade.

wscons é ativado por default ao término da instalação e, portanto, não é necessário fazer nada para poder utilizá-lo. Para habilitar os consoles virtuais ver Capítulo 4. No resto da seção serão descritas as opções do arquivo de configuração do kernel que dizem respeito ao wscons.

Para habilitar wscons no arquivo de configuração do kernel é necessário ativar a opção adequada e desabilitar aquelas que dizem respeito ao pcvt e ao pcccons. (Atenção, somente um dos três drivers deve ser habilitado).

```
#pc0    at isa? port 0x60 irq 1   # pccons generic PC console driver
#vt0    at isa? port 0x60 irq 1   # PCVT console driver
```

Para habilitar o teclado italiano

```
options     PCKBD_LAYOUT="KB_IT"
```

A disposição do teclado italiano não é a ideal, em especial para os programadores. Para uma modificação ver Capítulo 4.

O número de consoles virtuais alocados automaticamente é controlado por

```
options     WSDISPLAY_DEFAULTSCREENS=4
```

Os consoles adicionais, isto é, aqueles a mais com relação aos consoles auto-alocados, são habilitados no arquivo `/etc/wscons.conf`, tirando-se os indicadores de cometário (#) nas linhas do tipo "screen `x`". No exemplo seguinte é reservado um console adicional em relação aos alocados de modo automático.

```
# screens to create
#       idx     screen  emul
#screen 0       -       vt100
screen 1        -       vt100
screen 2        -       vt100
screen 3        -       vt100
screen  4       -       -
#screen 4       80x25bf vt100
#screen 5       80x50   vt100
```

O script `rc.wscons` transforma cada linha não comentada desta tabela em uma chamada ao comando **wsconscfg**. As colunas tornam-se parâmetros do próprio comando. A coluna *idx* torna-se o parâmetro `index`; a coluna *screen* torna-se o parâmetro `-t type` (que define o tipo de tela, ou seja, o formato, o número de cores, etc.) e a coluna *emul* torna-se o parâmetro `-e emul` (que define a emulação). Por exemplo, se "screen 3" estivesse ativo, tornar-se-ia uma chamada a:

```
wsconscfg -e vt100 3
```

> **Nota:** note-se que pode ocorrer um conflito (inócuo) entre consoles alocados automaticamente pelo kernel e os consoles alocados no momento da inicialização através do `/etc/wscons.conf`. Se na inicialização tenta-se alocar uma tela já auto-alocada pelo kernel, será visualizada uma mensagem do tipo:
>
> ```
> wsconscfg: WSDISPLAYIO_ADDSCREEN: Device busy
> ```
>
> A solução consiste em se comentar as linhas inúteis do `/etc/wscons.conf`.

Os consoles criados são naturalmente também ativados em `/etc/ttys`. Por exemplo:

```
console "/usr/libexec/getty Pc"         pc3     off secure
ttyE0   "/usr/libexec/getty Pc"         vt220   on secure
ttyE1   "/usr/libexec/getty Pc"         vt220   on secure
ttyE2   "/usr/libexec/getty Pc"         vt220   on secure
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
...
```

A linha

```
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
```

do `/etc/ttys` permite ao servidor X encontrar um terminal livre. Para usar uma tela diferente da número 4 é necessário fazer de tal modo que se passe um parâmetro na forma vt*n* ao servidor X (onde *n* é o número da tecla de função associada à tela a ser reservada para a interface gráfica).

Pode-se, por exemplo, habilitar a "screen 7" no `/etc/wscons.conf` e depois inicializar o X com “vt8”. No caso do xdm é necessário editar o `/usr/X11R6/lib/X11/xdm/Xserver`. Por exemplo:

```
:0 local /usr/X11R6/bin/X +kb dpms -bpp 16 dpms vt8
```

Se, no entanto, é usado o xdm3d, o percurso é diferente: `/usr/X11R6/share/xdm3d/Xservers`.

> **Nota:** no wscons a combinação Ctrl+Alt+Backspace reseta o emulador do terminal. Pode ser útil em caso de problemas.

13.1.1. Modo texto em 50 linhas com wscons

A modalidade de texto em 50 linhas pode ser utilizada a partir da versão 1.4.1 do NetBSD. Para ativar esta modalidade, além das operações precedentemente descritas, é necessário editar o `/etc/wscons.conf`, tirando o comentário da linha:

```
font ibm  -  8  ibm  /usr/share/pcvt/fonts/vt220l.808
```

No fim do arquivo, modificar as linhas do seguinte modo:

```
#screen 0       80x50   vt100
screen  1       80x50   vt100
screen  2       80x50   vt100
screen  3       80x50   vt100
screen  4       80x50   vt100
screen  5       80x50   vt100
screen  6       80x50   vt100
screen  7       80x50   vt100
```

Deste modo obtém-se oito telas, às quais se acessa com a combinação Ctrl+Alt+Fn (sendo que `n` vai de 1 a 8). Os dispositivos correspondentes são os que vão de ttyE0 a ttyE7. Para habilitá-los e ter um prompt de login, modificar o `/etc/ttys`:

```
ttyE0   "/usr/libexec/getty Pc"         vt220   on secure
ttyE1   "/usr/libexec/getty Pc"         vt220   on secure
ttyE2   "/usr/libexec/getty Pc"         vt220   on secure
ttyE3   "/usr/libexec/getty Pc"         vt220   on secure
ttyE4   "/usr/libexec/getty Pc"         vt220   on secure
ttyE5   "/usr/libexec/getty Pc"         vt220   on secure
ttyE6   "/usr/libexec/getty Pc"         vt220   on secure
ttyE7   "/usr/libexec/getty Pc"         vt220   on secure
```

Ao que parece, não é possível modificar a definição de 80x25 da tela 0, provavelmente para evitar, em caso de problemas, que se perca a única tela de console que está funcionando.

13.1.2. wsmouse

13.2. pccons

Pccons é o driver de console utilizado pelos disquetes de instalação da versão de hardware i386 (e talvez também de outras arquiteturas). Trata-se de um driver muito compacto (exige pouco espaço), mas que não dispõe de consoles virtuais ou de outras características avançadas.

13.3. pcvt

pcvt é um emulador de terminal VT220 e é dotado de funcionalidade muito mais avançada se o comparamos com o simples pccons. Entre outras coisas, suporta mais configurações de teclado e consoles virtuais múltiplos (com Ctrl+Alt+F1...F8 ou com as teclas de função F9-F12). Para ativar o pcvt é necessário recompilar o kernel. É preciso que se tenha as seguintes linhas no próprio arquivo de configuração:

```
# Ativar somente uma das duas linhas seguintes
#pc0    at isa? port 0x60 irq 1
vt0     at isa? port 0x60 irq 1
# Options for PCVT console driver
```

```
#options FAT_CURSOR
options PCVT_NETBSD=132
options PCVT_NSCREENS=3
```

Ademais, é necessário ativar o teclado italiano na inicialização e escolher o tipo certo de terminal:

```
/usr/local/bin/kcon -m i2
TERM=pcvt25;    export TERM
```

Simultaneamente é necessário modificar o `/etc/ttys`. Por exemplo:

```
#console "/usr/libexec/getty Pc"   pcvt25  on secure
ttyv0    "/usr/libexec/getty Pc"   pcvt25  on secure
```

> **Nota:** Na definição do teclado i2 há um erro. É necessário modificar o arquivo `/sys/arch/i386/isa/pcvt/Util/keycap/keycap.src`. A versão correta é:
>
> ```
> i2|italy142|Italian 142 mapping:\
>      :A8={:A9=[:A10=]:A11=}:\
>      :A12=`:A13=~:\
>      :A17=@:A18=#:\
>      :tc=italy141:
> ```

Na inicialização é necessário carregar as definições para o teclado italiano modificando, por exemplo, o arquivo `/etc/rc.local`:

```
KCONP=/usr/local/bin
SCONP=/usr/local/bin
LDFNP=/usr/local/bin
ISPCP=/usr/sbin
CURSP=/usr/local/bin

set_keybd=YES

#------------------------------------------------------
# if desired, setup keyboard for italian keyboard layout
#------------------------------------------------------

if [ X${set_keybd} = X"YES" -a -x $KCONP/kcon ]
then
  echo
  echo 'switching to italian keyboard layout'
  $KCONP/kcon -m i2
fi

echo '.'
```

Além disso, é preciso modificar o `/etc/ttys`:

```
#console  "/usr/libexec/getty Pc" pcvt25  on secure
ttyv0   "/usr/libexec/getty Pc" pcvt25  on secure
ttyv1   "/usr/libexec/getty Pc" pcvt25  on secure
```

```
ttyv2   "/usr/libexec/getty Pc" pcvt25  on secure
```

É necessário também compilar e instalar os programas utilitários do pcvt.

```
cd /sys/arch/i386/isa/pcvt/Util
make
make install
```

> **Nota:** No caso de erros de compilação, atentar para que no `/usr/include/machine` esteja presente o arquivo `pcvt_ioctl.h`.

13.3.1. Mudando as dimensões da tela

Com o pcvt é possível alterar o número de linhas e de colunas sobre a tela. O seguinte programa, por exemplo, executa a operação automaticamente, permitindo escolher entre várias configurações:

```
#!/bin/sh
# Set the screen to # lines
case $1 in
  25)
    /usr/local/bin/scon -s 25
    /usr/local/bin/cursor -s13 -e14
    ;;
  28)
    /usr/local/bin/loadfont -c1 -f
    /usr/share/misc/pcvtfonts/vt220l.814
    /usr/local/bin/loadfont -c2 -f
    /usr/share/misc/pcvtfonts/vt220h.814
    /usr/local/bin/scon -s 28
    /usr/local/bin/cursor -s12 -e14
    ;;
  40)
    /usr/local/bin/loadfont -c3 -f
    /usr/share/misc/pcvtfonts/vt220l.810
    /usr/local/bin/loadfont -c4 -f
    /usr/share/misc/pcvtfonts/vt220h.810
    /usr/local/bin/scon -s 40
    /usr/local/bin/cursor -s8 -e10
    ;;
  50)
    /usr/local/bin/loadfont -c5 -i
    /usr/share/misc/pcvtfonts/vt220l.808
    /usr/local/bin/loadfont -c6 -i
    /usr/share/misc/pcvtfonts/vt220h.808
    /usr/local/bin/scon -s 50
    /usr/local/bin/cursor -s6 -e8
    ;;
  *)
    echo "Invalid # of lines (25/28/40/50)"
    ;;
```

```
esac
```

Capítulo 14. Editando

Este capítulo descreve o editor padrão do NetBSD, isto é, o editor vi. A primeira parte do capítulo introduz as bases do uso desse editor, enquanto as seções posteriores descrevem-lhe algumas características avançadas, com referência particular à versão do vi instalada com o NetBSD. Quem deseja aprofundar o argumento pode fazer referência à bibliografia no final deste capítulo e também aos links apresentados nas seções seguintes.

14.1. Introdução ao vi

Não há necessidade de apresentação para o editor vi. Desde quando foi desenvolvido por Bill Joy, na Universidade da Califórnia de Berkeley, tornou-se o editor padrão de quase todas as versões do Unix; o companheiro e amigo fiel dos administradores de sistema, assim como o bicho-papão de quase todos os principiantes.

Vale, pois, a pena dedicar algumas palavras ao uso desse editor, até porque ao final da instalação do NetBSD é o único disponível no sistema (exceção feita ao ed) e é, portanto, o instrumento que deverá ser utilizado para a configuração inicial do próprio sistema. Em um segundo momento, naturalmente, cada um estará livre para instalar o editor que preferir e conhecer melhor, escolhendo na nutrida série dos integrantes da coleção de pacotes (ver o Capítulo 8).

Esta seção contém uma introdução ao uso do vi, que deveria ser suficiente para permitir a um principiante criar e modificar os arquivos de texto. Apenas as características básicas e alguns dos comandos de uso mais comum são descritos sem nenhuma pretensão de completude. Para um abordagem mais exaustiva remete-se aos documentos descritos na Secção 14.2.2.

Os principiantes não devem cometer o erro de julgar o vi a partir dessas poucas instruções. Usado de modo tão “minimalista”, vi é sem dúvida muito incômodo. As suas virtudes podem ser descobertas quando nos apropriamos de todos (!) os seus comandos e as funções avançadas começam a ser usadas. Nesse ponto o vi já não é mais o editor hostil do início, para se tornar um instrumento cômodo, potente e confiável. Este guia foi escrito, inútil dizê-lo, utilizando o editor vi.

14.1.1. Primeiros passos

Para iniciar o vi, executar o comando:

```
$ vi nomedoarquivo
```

onde *`nomedoarquivo`* é o nome do arquivo a ser editado (que também pode não existir ainda). Se o *`nomedoarquivo`* é omitido, vi começa com um arquivo inicialmente vazio (e ainda sem nome. Será necessário dar-lhe um no momento de salvá-lo). Neste exemplo suponhamos a criação de um arquivo novo:

```
$ vi meuarquivo
```

onde *`meuarquivo`* é o nome de um arquivo que não existe no diretório corrente.

Uma vez inicializado o vi, a tela mostra uma coluna de acentos til à esquerda e, na última linha, o nome do arquivo junto a algumas informações de significado evidente. Algo desse tipo (mas com mais linhas):

```
~
~
~
~
~
~
meuarquivo: new file: line 1
```

Os acentos til no início da linha indicam que a linha em questão está vazia, isto é, que a linha não contém texto. Uma vez que o arquivo carregado (meuarquivo) não existia, todas as linhas sobre a tela do vi estarão vazias.

Nesse ponto a pressão da maior parte das teclas provoca um sinal acústico ou a visualização de uma mensagem de erro (tente-se, por exemplo, pressionar o **j** ou o **k**). Isso é devido ao fato de que o vi é um editor *modal* e, portanto, as teclas pressionadas têm um efeito diferente de acordo com o "modo" em que se encontra o editor. Na inicialização o vi encontra-se sempre em *modo de comando* e todas as teclas pressionadas são interpretadas como comandos para o editor ao invés de texto para inserir no arquivo. Portanto, pressionando **j** o vi não insere um "j" no arquivo, mas procura executar o comando **j** que, como será visto em breve, significa "desloca o cursor para a próxima linha". Como a arquivo ainda está vazio, não há nenhuma linha seguinte para se deslocar o cursor e será assinalado um erro.

Alguns dos comandos do vi permitem passar do modo de comando para o *modo de inserção*. Uma vez que se entre nesse modo, tudo o que se escreve é interpretado como texto a ser inserido no arquivo. Vejamos agora um exemplo.

Pressionar **i** para passar ao modo de inserção e digitar as seguintes linhas de texto (terminar cada linha com **Enter**):

```
Primeira linha do texto inserido,
segunda linha do texto inserido,
terceira e última linha do texto inserido.
```

ao término da inserção pressionar **ESC** para voltar ao modo de comando.

Neste exemplo vimos como passar do modo de comando para o modo de inserção pressionando **i**; como inserir algumas linhas de texto, terminando cada linha com um **Enter**; e como voltar, por fim, ao modo de comando pressionando **ESC**. Agora que o arquivo contém um pouco de texto, alguns comandos podem ser experimentados. Os primeiros que veremos são os que permitem o deslocamento no interior do arquivo.

> **Nota:** o texto inserido pertence logicamente ao arquivo `meuarquivo` (o nome transmitido como parâmetro ao editor) mas, nesse momento, reside somente na memória do editor porque ainda não foi salvo em um arquivo. Todas as operações efetuadas no texto (acréscimos, modificações, etc.) não são salvas no disco enquanto o usuário não o exige explicitamente com um comando de salvamento do arquivo. Portanto é sempre possível abandonar as modificações saindo do editor sem salvar, ficando a cópia do arquivo invariada. Os comandos para salvar/abandonar/sair são descritos na Secção 14.1.9.

14.1.2. Comandos para deslocar-se no arquivo

Quando nos encontramos em modo de comando, alguns comandos permitem-nos o deslocamento no interior de um arquivo (ou seja, de modificar a posição do cursor). A lista seguinte mostra os principais comandos de deslocamento.

```
h       cursor à esquerda       (seta para  esquerda)
j       cursor para baixo       (seta para baixo)
k       cursor para cima        (seta para cima)
l       cursor à direita        (seta para a direita)
^F      página seguinte         (PgDn)
^B      página anterior         (PgUp)
0       cursor no início da linha
$       cursor no fim da linha
b       palavra anterior
w       palavra seguinte
Enter   primeiro caracter da linha seguinte   (espaços excluídos)
+       primeiro caracter da linha seguinte   (espaços excluídos)
-       primeiro caracter da linha anterior   (espaços excluídos)
```

(o símbolo ^ indica o pressionamento da tecla Control em conjunto com a tecla que representa a letra que segue. Portanto, **^F** significa Control+F). De acordo com a configuração do terminal também é possível que o editor consiga utilizar as teclas de setas e as tecla PgUp e PgDn.

Note-se que, como já foi dito, essas teclas servem para se deslocar quando nos encontramos em modo de comando. Se pressionadas em modo de inserção, contudo, causam a inserção do texto no interior do arquivo (isto é, pressionando **h** o caracter 'h' é inserido no arquivo). Observemos assim um primeiro fato: em modo de inserção pode-se apenas inserir texto seqüencialmente. Para que se possa deslocar no interior do arquivo é necessário sair do modo de inserção (com **ESC**), retornando ao modo de comando. Esta é uma das conseqüências do fato de que o vi é um editor modal.

Em modo de comando é possível deslocar-se no arquivo até mesmo especificando o número da linha para a qual queremos ir. Por exemplo:

```
12G     vai para a linha 12
```

O comando **G** sem prefixo numérico leva o cursor ao fim do arquivo.

> **Prefixos numéricos:** muitos comandos do vi aceitam um *prefixo numérico* que lhes modifica a interpretação por parte o editor. Mesmo se o uso do prefixo numérico não é descrito aqui, mencionamos alguns exemplos para dar uma idéia dos possíveis usos.
>
> ```
> 9k      cursor 9 linhas para cima
> 12dd    deleta doze linhas
> 3w      cursor à frente de três palavras
> 5x      delete 5 caracteres
> ```

Nesta seção descrevem-se alguns dos principais comandos de deslocamento, que podem ser experimentados nas linhas de um texto já inserido. O editor vi dispõe de muitas outras funções sofisticadas para deslocar-se no interior do arquivo. Estas, contudo, não são descritas nesse breve tutorial.

14.1.3. Comandos para mudar o modo

No exemplo inicial já vimos que para passar do modo de inserção para o modo de comando é necessário pressionar a tecla **ESC**. Se pressionamos a tecla Esc quando o vi já está em modo de comando, o editor emite um sinal acústico inócuo e, portanto, se não nos recordamos qual é o modo corrente, sempre se pode pressionar ESC, com a garantia de se confirmar no modo de comando.

Se ESC é o único método para voltar para o modo de comando, os comandos que permitem entrar em modo de inserção são muitos. Segundo o comando escolhido, o efeito será diferente, como se pode observar da seguinte tabela.

`i`	`insere texto antes do cursor`
`a`	`insere texto depois do cursor`
`A`	`insere texto no fim da linha`
`I`	`insere texto no início da linha`
`o`	`abre uma linha vazia depois da linha corrente`
`O`	`abre uma linha vazia antes da linha corrente`

A presença de numerosos comandos que fazem coisas similares é uma constante do editor vi. O efeito é o de se ter muitos modos para fazer a mesma coisa e, portanto, muitas possíveis soluções. Um expert em vi geralmente consegue encontrar a solução que lhe permite levar a cabo a função desejada com o menor número possível de comandos. Por exemplo, para inserir a palavra "fim" no fim da linha podem-se utilizar as duas seguintes seqüências equivalentes:

```
$afimESC
AfimESC
```

na primeira, vamos para o fim da linha com o comando **$**, depois se passa ao modo de inserção com o comando **a**, acrescenta-se a palavra "fim" e se retorna ao modo de comando com a pressão da tecla **ESC**. Na segunda seqüência, o comando **A** combina as funções de **$** e de **a** em uma só tecla.

14.1.4. Comandos para deletar o texto

A tabela seguinte reúne alguns comandos para a deleção de texto.

`dd`	`apaga a linha corrente`
`x`	`apaga o caracter sob o cursor`
`X`	`apaga o caracter antes do cursor`
`dw`	`apaga até o fim da palavra`
`D`	`apaga do cursor até o fim da linha`

também estes comandos só podem ser dados em modo de comando, e não em modo de inserção. Daqui para a frente não será mais indicado o fato de que os comandos operam em modo de comando, uma vez que isso já deve ter ficado evidente. Quando forem descritas algumas funções que podemos acionar no modo de inserção, este fato será indicado explicitamente.

14.1.5. Exemplo

Vejamos agora um exemplo prático que retoma o uso de alguns dos comandos apresentados nas seções precedentes. Suponhamos que o arquivo em curso de edição seja composto das três linhas inseridas no exemplo inicial e que se queira transformar a terceira linha do seguinte modo:

```
terceira linha de texto iserida no arquivo.
```

Como primeira operação cancelamos da última linha as palavras “e última” e depois inserimos o novo texto “inserida no arquivo” ao fim da linha, antes do ponto.

Suponhamos que o cursor encontre-se sobre a primeira linha (se assim não for, deslocar-se para a primeira linha com o comando **1G**). Para ir da primeira para a terceira linha, pode-se usar uma das seqüências seguintes (mas ainda existem muitas outras):

```
<Enter><Enter>
jj
3G
/terceira/<Enter>
```

Uma vez chegados ao início da terceira linha, postamo-nos sobre a primeira das duas palavras a apagar com

```
w
```

Enfim as duas palavras são apagadas com:

```
dwdw
```

> **Nota:** recordando tudo o que foi acenado sobre prefixos numéricos, pode-se intuir que os dois comandos **dw** recém executados podem ser substituídos por um único comando **2dw**, que apaga duas palavras de um só golpe; ou ainda, o que pode parecer surpreendente, por **d2w**

Para adicionar o novo texto vamos para o fim da linha com o comando **$**, entra-se em modo de inserção com o comando **i** e depois se escreve:

```
inserida no arquivo
```

Pressionando **ESC** retorna-se ao modo de comando.

14.1.6. Busca de texto

Para procurar uma expressão dentro do arquivo corrente usam-se os seguintes comandos:

```
/expressão/
?expressão?
```

o primeiro comando (seguido de Enter) procura a expressão no arquivo andando para a frente, ou seja, na direção do fim do arquivo a partir da posição do cursor. O segundo comando (também esse seguido de Enter) procura a expressão andando para trás, ou seja, na direção do início do arquivo.

Por exemplo, retornando às três linhas inseridas inicialmente, experimente-se dar os seguintes comandos:

```
1G/última/
```

e teclar Enter. O comando **1G** faz o cursor deslocar-se sobre a primeira linha do arquivo (note-se que não é feito o eco do comando). A seqüência **/última** (que é vista na última linha da tela) seguida do **Enter** deflagra a busca da expressão "última" dentro do texto e o deslocamento do cursor para a linha correspondente.

> **Nota:** os comandos de busca constituem assim um método a mais de deslocamento dentro do arquivo, que tem em vista o próprio conteúdo do arquivo.

Para repetir a última busca efetuada sem que seja preciso reescrever, podem-se usar os comandos

`n`	`repete a busca na mesma direção`
`N`	`repete a busca na direção oposta`

14.1.7. Comandos que funcionam em modo de inserção

Alguns comandos, geralmente obtidos com a tecla Control, estão disponíveis mesmo quando se está inserindo um texto.

`^H`	`apaga o caracter à esquerda do cursor`
`BS`	`apaga o caracter à esquerda do cursor`
`^W`	`apaga a palavra precedente`
`^U`	`apaga até o início da linha`
`ESC`	`volta para o modo de comando`

14.1.8. Anulando o efeito de uma ação

Se, uma vez retornados ao modo de comando constatamos ter cometido um erro, é possível utilizar o comando *undo* (a tecla **u**) para anular o último comando executado.

14.1.9. Comandos para sair do vi

Os principais comandos para encerrar o trabalho e sair do vi ou simplesmente para salvar e continuar trabalhando são:

`:w`	`salva o arquivo (sem sair)`
`:q`	`sai`

```
:wq     salva o arquivo e sai
:q!     sai sem salvar (as modificação são perdidas)
:w abc  salva o arquivo dando-lhe 0 nome 'abc'
```

Para mudar o nome do arquivo corrente (ou para dar-lhe um se ainda não tem):

```
:file nomedoarquivo
```

Os comandos descritos nesta seção exemplificam uma ulterior modalidade de funcionamento do vi, o *modo ex*, que nesta breve introdução não será descrito. O editor vi nasce de fato da combinação de dois editores distintos, dos quais um (precisamente o vi) serve de interface *full screen* para o outro (o ex), que fornece as funções básicas de edição. O ex pode ser considerado uma versão evoluída do ed.

14.1.10. Conclusões

Como já foi acenado na introdução, nesta seção foram descritos apenas os comandos mais elementares do editor vi, para que se possam editar arquivos do NetBSD ao término da instalação. Para poder usar esse editor de modo produtivo (e cômodo), é necessário aprofundar o conhecimento dos comandos lendo, por exemplo, os documentos citados na Secção 14.2.2.

14.2. Configurando o vi

O editor padrão de que o NetBSD é dotado é, como já foi dito, o vi. Se você não utiliza o vi, esta seção não lhe interessa. Caso contrário, leia-a antes de correr para instalar o vim ou o elvis. O vi do NetBSD é, de fato, o nvi, uma versão histórica do vi com muitas extensões poderosas escrita por Keith Bostic da Universidade da Califórnia de Berkeley nos primeiros anos 90, para ter uma versão livremente redistribuível do vi e tornada, em seguida, a versão oficial do BSD.

Entre as extensões mais interessantes encontramos

- Expressões regulares estendidas (habilitadas com a opção `extended`).
- Tag stacks.
- Undo infinito (ativado de modo um pouco curioso: no primeiro *undo* pressiona-se o **u**. Os undo sucessivos são obtidos pressionando-se o caracter ponto (**.**)).
- Busca ampliada (habilitada com a opção `searchincr`).
- Rolagem das linhas (habilitada pela opção `leftright`. O número das colunas é definido por `sidescroll`).
- Histórico da linha de comando (controlado pela opção `cedit`).
- Completamento da linha de comando (controlado pela opção `filec`).
- Telas que podem ser postas em primeiro plano ou em plano de fundo.
- Tela dividida (split screen).

14.2.1. Extensões do `.exrc`

No exemplo seguinte vejamos algunas comandos de configuração que simplificam um pouco a nossa vida.

```
set showmode ruler
set filec=^[
set cedit=^[
```

A primeira linha habilita a visualização da posição do cursor (linha e coluna) e do modo corrente (Command, Insert, Append) sobre a linha de status. A segunda linha (^[é o caracter ESC) habilita o completamento dos nomes dos arquivos e a terceira a *history* da linha de comando. Ambas utilizarão o comando ESC. Por exemplo, escrevendo ':' e depois pressionando-se ESC, abre-se uma janela com a lista dos comandos anteriores. A relação pode ser modificada como se fosse um arquivo. Quando se pressiona Enter sobre uma linha o comando correspondente é executado.

14.2.2. Documentação

No *tarball* do código-fonte (`src.tgz`) está contida uma notável quantidade de documentação original do vi/nvi. Ela se encontra em `/usr/src/usr.bin/vi/docs`. Em particular podemos encontrar:

- Edit: A tutorial
- Ex Reference Manual
- Página de manual do vi
- An Introduction to Display Editing with Vi de William Joy e Mark Horton
- Ex/Vi Reference Manual de Keith Bostic, o manual de referência do nvi, que descreve em detalhe todos os comandos do editor.
- Vi Command & Function Reference
- Vi tutorial (iniciante e avançado)

Quem deseja aprender a usar o vi deve começar lendo *An Introduction to Display Editing with Vi* de William Joy e Mark Horton, para depois passar a *Ex/Vi Reference Manual* de Keith Bostic.

14.3. O uso de tags no vi

Este tópico não é estritamente ligado ao NetBSD mas, já que estamos aqui, merece um pequeno aprofundamento; pode ser cômodo para examinar, por exemplo, o código-fonte do kernel ou de um programa complexo, composto de vários diretórios.

Os *tags* são uma daquelas características que tornam o vi um potente instrumento de edição para programadores. Trata-se, em substância, de um arquivo que contém uma lista de nomes de procedimentos, macros, etc., presentes nas fontes de um programa. Quando se editam as fontes, graças aos tags é possível saltar de um ponto em que uma função é chamada para o ponto em que a própria função é definida. Por exemplo, suponhamos examinar um código-fonte com as seguintes linhas:

```
for (i = 0; i < len; i++)
```

```
        funcXY(i);
```

posicionando o cursor sobre a primeira letra de “funcXY” e pressionando ^] (Control+]), o vi abre automaticamente a fonte em que está presente a definição de funcXY e se posiciona sobre a própria função.

```
void
 funcXY(i)
        int i;
 {
  funcZ();
        ...
```

Uma vez examinada/modificada funcXY, pressionando-se ^T (Control+T) retorna-se ao arquivo e à posição inicial (aquela do ciclo for). Note-se que estes “saltos” podem ser aninhados. Se com ^] saltamos à função funcXY, e depois com mais um ^] salta-se à função funcZ, com dois ^T retornamos ao arquivo e à posição inicial.

Para se obter o resultado descrito no parágrafo anterior é necessário que o vi encontre um arquivo (geralmente) chamado `tags` no diretório corrente ou em uma posição especificada por nós com o comando **:set tags=/percurso/nomedoarquivo** (este comando também se pode inserir no arquivo `.exrc`). Tal arquivo, que é um arquivo de texto, é gerado pelo programa ctags. Por exemplo, o comando:

```
$ ctags *.c *.h
```

gera o arquivo `tags` no diretório corrente. Quando o vi é aberto, o arquivo `./tags` é encontrado automaticamente.

Quando examinamos as fontes que se encontram em uma hierarquia de diretórios e subdiretórios, podem-se igualmente gerar e utilizar os *tags* do vi. O processo é o seguinte.

1. Deslocamo-nos para o diretório de base das fontes com o comando

   ```
   $ cd /percurso
   ```

2. Constrói-se o arquivo dos tags em duas passagens: primeiro cria-se uma lista dos arquivos-fonte a examinar e depois se a submete ao programa ctags para criar o arquivo `tags`.

   ```
   $ find . -name "*.[ch]" > filelist
   $ ctags -L filelist
   ```

3. Em `./.exrc` insere-se a linha

   ```
   set tags=/percurso/tags
   ```

   substituindo o percurso apropriado no lugar de “percurso”.

14.4. Alternativas ao nvi

O nvi é a versão do vi nativa do NetBSD, mas não é a única existente. Desde logo, é uma das mais próximas do espírito da versão original do programa e, portanto, se você é um purista, não quererá usar outro, como de resto fazem muitos dos próprios desenvolvedores do NetBSD. Todavia, existem muitos “clones” desse programa, alguns dos quais muito difundidos e apreciados. Portanto, para quem se dá bem trabalhando com o vi mas procura “algo mais”, há uma ampla gama de escolhas. Para uma panorâmica completa os sites “The VI pages” (http://www.math.fu-berlin.de/~guckes/vi) e “Vi Lovers Home Page” (http://www.thomer.com/thomer/vi/vi.html) são o melhor ponto de partida. Além da descrição e dos links para todos os clones existentes do vi, há links para tutoriais e informações variadas.

Para começar a se desvencilhar da selva de programas aconselho dar uma olhada no vim e no vile. Trata-se de duas versões “turbinadas” do vi, a que foram adicionados muitos comandos, suporte para mouse (nas versões X), evidenciação da sintaxe dos arquivos fonte com cores diversas, possibilidade de se deslocar dentro de um arquivo mesmo quando se está no modo de inserção e muitas, mas muitas mesmo, outras funcionalidades. A filosofia dos dois programas, ambos ótimos e merecedores de serem experimentados, é diferenciada. O vile, em particular, é programável, sendo dotado de uma verdadeira macro-linguagem que permite definir processos e atribuí-los às teclas. Agradará, portanto, aos amantes da personalização extrema. O vim, por sua vez, é talvez o clone mais difundido e amado, dotado de um riquíssimo (até excessivo) sortimento de novos comandos, muitos dos quais estudados especificamente para os programadores.

Bibliografia

[LambRobbins] Linda Lamb e Arnold Robbins, O’Reilly & Associates, 1-56592-426-6, *Learning the vi Editor, 6th Edition*.

[Robbins] Arnold Robbins, O’Reilly & Associates, 1-56592-497-5, *vi Editor Pocket Reference*.

Capítulo 15. X

15.1. O que é o X?

O sistema X Window é um ambiente de trabalho gráfico disponível para o NetBSD e para muitos outros sistemas Unix (e não Unix). Com efeito, o X é mais que um simples ambiente gráfico. Graças ao uso do protocolo X, o sistema X Window é “network transparent” (transparente à rede) e é capaz de executar aplicações distribuídas (cliente-servidor). Isto significa, a uma primeira aproximação, que é possível lançar um programa em um host “client” e visualizar-lhe o output gráfico em um outro host “server” de modo totalmente transparente (transparente significa, nesse caso, que não é necessário modificar a aplicação para obter este resultado). O sistema X Window é produto do “X Consortium” e a edição corrente é a X11R6. A versão do X usada pelo NetBSD é a XFree86, uma implementação open source livremente redistribuível do sistema X Window.

Quando começamos a usar o X encontramos alguns termos cujo significado pode parecer inicialmente obscuro. Os elementos de base do X são:

- Hardware de vídeo suportado pelo XFree86.
- Um *servidor X* interfaciando-se com o hardware. O servidor X fornece um modo padrão para abrir janelas, efetuar operações gráficas (por exemplo, utilizar fontes para a visualização do texto) e para ler o input do mouse, do teclado ou de outros periféricos. X é transparente à rede e, portanto, é possível acionar cliente X em uma máquina e o servidor de X correspondente (isto é, o monitor com o hardware de vídeo) em outra.
- Um *window manager* (gerenciador de janelas) que utiliza o servidor X. O window manager é basicamente um tipo especial de cliente X a que é permitido manipular o posicionamento e o aspecto das janelas. Pode-se “decorar” as janelas com “widgets” padrões, que geralmente permitem deslocar, redimensionar, reduzir a ícone, etc., as janelas. Um gerenciador de janelas pode ser dotado também de outras funcionalidades (fechar as janelas, encerrar os programas, mostrar o menu dos programas, etc.) ou de efeitos especiais (sombreamento, decorações, etc.) e assim por diante.
- Um *desktop manager* ou gerenciador das telas de trabalho (opcional). KDE e GNOME são dois exemplos de desktop managers. Trata-se de conjuntos de programas (suites) mais ou menos integrados e coordenados, projetados para fornecer ao usuário um conjunto de aplicações de base dotadas de uma interface comum. O gerenciador da área de trabalho (desktop manager) pode permitir o acesso ao sistema de arquivos utilizando a metáfora do desktop (mesa de trabalho), pode dispor de um browser de navegação no help, terminais que substituem o clássico xterm, aplicativos de gerenciamento de áudio, ambiente de desenvolvimento, editor e assim por diante.
- Todos os outros aplicativos (clientes X de terceiros) instalados no sistema. Estes “falam” com o servidor X e com o gerenciador de janelas (geralmente o desktop manager não está envolvido de modo particular naquilo que os aplicativos executam).

Recapitulando, para que se possa utilizar o ambiente gráfico é necessário:

- o sistema XFree86;
- um gerenciador de janelas (o XFree86 já dispõe de um gerenciador de janelas simples chamado twm. Não é muito sofisticado mas muitos o acham mais que suficiente);

- quem prefere um ambiente mais estruturado deverá instalar também um desktop manager, embora isso não seja uma necessidade absoluta. Os desktop managers oferecem características que podem facilitar a aproximação ao sistema de usuários provenientes de outros ambientes, como os Macintosh ou uma das tantas versões do MS-Windows.

> **Nota:** aqui já deveria estar claro que os desktops como GNOME e KDE têm seu próprio window manager, embora também possam usar um externo compatível.

Normalmente se utiliza apenas um window manager por vez em cada servidor X (mas é possível ativar mais de um servidor X em um mesmo computador).

15.2. Configuração

Se você não escolheu uma configuração mínima para a instalação do NetBSD, ao fim da instalação o X, na sua encarnação do XFree86, já está presente no sistema e pronto para funcionar, contanto que se crie o arquivo de configuração, o temido `/etc/XF86Config`. Para se ter uma idéia do aspecto deste arquivo pode-se dar uma olhada em `/usr/X11R6/lib/X11/XF86Config.eg`. A estrutura do arquivo de configuração está descrita formalmente em XF86Config(4/5), que pode ser examinado com:

```
# man XF86Config
```

Antes de configurar o sistema é aconselhável uma atenta leitura da documentação do XFree86 em `/usr/X11R6/lib/X11/doc`: há arquivos README específicos para as placas de vídeo e para o mouse, e também um `README.NetBSD`. Aconselho começar com a leitura do `QuickStart.doc`. O tempo dedicado à leitura destas informações não é de modo algum desperdiçado. O conhecimento da própria configuração e de como utilizá-la com o X é de grande importância para se aproveitar da melhor maneira o próprio hardware.

O arquivo `/etc/XF86Config` pode ser escrito manualmente com qualquer editor ou usufruindo de algum programa que o crie automaticamente, partindo da configuração especificada pelo usuário. Os programas mais conhecidos são xf86config e XF86Setup. O primeiro funciona em modo texto e já vem instalado no X. O segundo é um programa gráfico que pode ser encontrado na coleção de pacotes.

Ambos os aplicativos requerem ao usuário especificar a configuração do próprio sistema e, em particular:

- o tipo de mouse e o device driver a utilizar
- o tipo de teclado e o seu layout
- o tipo de placa de vídeo
- o tipo de monitor

Antes de escrever o arquivo de configuração, à mão ou com um dos dois programas já mencionados, é necessário que se obtenha estas informações.

15.3. O mouse

No que diz respeito ao mouse, é necessário verificar se ele é do tipo serial ou PS/2 e se estamos usando o dispositivo *wsmouse*, que exige um protocolo diferente. Para o mouse serial é preciso escolher o protocolo correto e especificar a porta serial. Por exemplo, para um mouse na primeira porta serial:

```
Section "Pointer"
  Protocol    "Microsoft"
  Device      "/dev/tty00"
EndSection
```

Para um mouse no dispositivo wsmouse:

```
Section "Pointer"
  Protocol    "wsmouse"
  Device      "/dev/wsmouse0"
EndSection
```

No campo "Device" também se pode especificar `/dev/mouse`, contanto que se crie o symbolic link apropriado no sistema. Por exemplo:

```
# ln -sf /dev/wsmouse0 /dev/mouse
```

15.4. O teclado

Mesmo que você já tenha configurado o teclado para o wscons, para poder usar o X é necessário efetuar uma configuração do teclado específica para esse ambiente. Isso, em todo caso, não tem nada de complexo.

Na falta de contraindicações convém utilizar o protocolo XKB, selecionando o tipo de teclado e a sua disposição.

A configuração do teclado é um dos pontos fracos de todos os programas de configuração automática. Geralmente convém deixar definida a configuração original para o inglês americano e depois retocar manualmente o arquivo de configuração gerado pelo programa. Por exemplo:

```
# XkbDisable
# XkbKeymap   "xfree86(us)"

XkbModel        "pc102"
XkbLayout       "it"
XkbVariant      "nodeadkeys"
```

Substituindo pc105 por pc102 no XkbModel podem ser utilizados até mesmo os símbolos dos teclados Windows.

15.5. O monitor

É fundamental especificar corretamente os valores da freqüência horizontal e vertical do monitor. Uma definição correta protege o monitor de configurações incompatíveis da placa de vídeo, evitando que seja danificado. No diretório da documentação está presente um arquivo com as especificações de um grande número de monitores.

15.6. A placa de vídeo e o servidor

A placa de vídeo pode ser escolhida no banco de dados dos programas de configuração. Uma vez escolhida a placa, o próprio programa cuida das configurações restantes.

Uma vez escolhida a placa é preciso definir o servidor a ser utilizado. Os programas de configuração normalmente são capazes de escolher o servidor adequado com base na placa gráfica selecionada. Algumas placas, entretanto, são suportadas por mais de um servidor. Nesse caso é preciso decidir com base na documentação. Servidores diferentes têm geralmente características e níveis de suporte para as placas de vídeo. Um exemplo típico é a placa S3 Virge que é suportada tanto pelo servidor SVGA quanto pelo servidor S3V.

15.7. Inicializando o X

Ao se finalizar, o programa de configuração cria o arquivo `/etc/XF86Config`, que pode ser examinado e modificado a mão.

Antes de executar o X convém:

- verificar se o link simbólico `/usr/X11R6/bin/X` aponta para o servidor certo:

  ```
  # ls -l /usr/X11R6/bin/X
  ```

- Para assegurar que tudo corra bem executar:

  ```
  # X -probeonly
  ```

  verificando atentamente o output.

Neste ponto pode-se executar o X com o comando:

```
# startx
```

Se o X não se inicializa, será preciso compreender o motivo observando as mensagens sobre a tela e corrigindo o arquivo de configuração.

Se o X inicializa-se mas alguma coisa não funciona (por exemplo, o mouse), pode-se sair do X com a combinação de teclas Ctrl+Alt+Backspace (não disponível para todas as versões de hardware). Se tudo vai bem, entretanto, adentramos o ambiente X com o gerenciador de janelas nativo que é o twm. Trata-se de um window manager simples mas eficaz, tanto que muita gente não sente a necessidade de procurar outros e continuam a usá-lo. Naturalmente, é possível instalar um outro window manager, assim como personalizar a aparência do twm. Por exemplo, tente dar o seguinte comando em um xterm:

```
# xsetroot -solid DarkSeaGreen
```

15.8. Personalizando o X

Há vários maneiras de personalizar a configuração (isto é, o aspecto das janelas) do X já na inicialização. Uma das mais simples é criar um arquivo `.xinitrc` no próprio diretório raiz, copiando o arquivo pré-configurado do sistema e o modificando.

```
# cp /usr/X11R6/lib/X11/xinit/xinitrc ~/.xinitrc
# vi .xinitrc
```

Eis um exemplo de modificação que inicializa o gerenciador de janelas, mostra um relógio em baixo à direita e abre dois terminais gráficos (xterms). Além do mais, define uma cor para o fundo.

```
a primeira parte é igual
...
# start some nice programs
twm &
xclock -geometry 50x50-1-1 &
xterm -geometry 80x34-1+1 -bg OldLace &
xsetroot -solid Bisque4 &
exec xterm -geometry 80x44+0+0 -bg AntiqueWhite -name login
```

Com esse tipo de configuração, para sair do X é necessário que o último xterm (aquele com o nome de “login”) seja fechado.

Além das preferências pessoais quanto à disposição das janelas e das cores, parece-me evidente que somente com estas pequenas modificações o ambiente de trabalho torna-se mais agradável.

Para melhorar ainda mais o aspecto do desktop podem ser instalados alguns programas utilitários da coleção de pacotes, entre os quais:

xcolorsel

mostra todas as cores definidas em `rgb.txt`.

xpmroot

permite utilizar um pixmap como fundo do X.

xscreensaver

protetor de tela para o X.

xdaemon

nenhum desktop pode dizer-se completo sem este programa que mostra o bitmap do demônio que simboliza os *BSD, selecionável em duas grandezas. A imagem pode ser deslocada à vontade.

15.9. Outros window managers

Quem não se contenta com o twm pode instalar um outro gerenciador de janelas entre os muitos disponíveis na coleção de pacotes. Entre os mais populares estão: fvwm2, olwm/olvwm (Open Look Window Manager), WindowMaker, Enlightenment e AfterStep.

A seguir vejamos, a título de exemplo, a instalação do WindowMaker. Para adicionar o programa utilizemos o pacote pré-compilado `windowmaker-0.61.tgz`, que depende de uma série de pacotes que devem ser instalados. Como sempre, seja o **pkg_add** seja o **make install** administrarão automaticamente todas as dependências e, portanto, na realidade não é necessário individuá-los e instalá-los manualmente.

```
# cd /usr/pkgsrc/x11/windowmaker
# make depends-list
xpm-3.4k
jpeg-6b
pkglibtool-1.2p2
giflib-3.0
libproplist-0.9.1
tiff-3.5.2
```

> **Nota:** as dependências poderiam ser vistas também com:
>
> ```
> # pkg_info -f windowmaker-0.61.0.tgz | grep depends
> ```

Depois de ter instalado os pacotes exigidos acrescentemos o WindowMaker e, já que estamos nisso, também alguns temas:

```
# pkg_add windowmaker-0.61.0.tgz wmthemes-0.6x.tgz
```

O WindowMaker agora está instalado e para inicializá-lo basta modificar o `.xinitrc` e/ou o `.xsession`, substituindo a chamada ao window manager anterior (twm) com a chamada ao comando `wmaker`. Por exemplo:

```
# start some nice programs
# start WindowMaker
wmaker &
xclock -geometry 50x50-1-1 &
xdaemon2 -geometry +0-70 &
...
```

Nesse exemplo também o programa xdaemon, que “personaliza” o desktop, é executado automaticamente.

Antes de inicializar o WindowMaker, é necessário executar o programa de configuração:

```
$ wmaker.inst
$ startx
```

15.10. Exemplo de uso dos recursos do X

A título de exemplo sobre como são personalizados os recursos dos programas X, vejamos como mudar as fontes de caracteres (fonts) pré-instaladas no XTerm.

Para personalizar os recursos utilizados por um programa, no nosso caso o XTerm, pode-se agir em nível global, modificando-se o arquivo `/usr/X11R6/lib/app-defaults/Xterm`. É preferível, entretanto, modificar o arquivo ${HOME}/.Xresources, até porque, fazendo assim, a configuração não se perde no caso de uma atualização do X ou do sistema operacional.

Uma vez tomada essa decisão, é preciso escolher uma nova fonte de caracteres usando os programas xfontsel e xlsfonts. Também é bom consultar o arquivo que define os codinomes (aliases) das fontes de caracteres, que é o `/usr/X11R6/lib/X11/fonts/misc/font.alias`. Por exemplo, descobre-se que o tipo pré-instalado de fonte de caracteres “fixed” é na realidade o codinome de “-misc-fixed-medium-r-semicondensed-13-120-75-c-60-iso8859-1”. Mesmo as fontes de caracteres como, por exemplo, “9x15” são na realidade codinomes.

Uma vez escolhida a fonte de caracteres, antes de a instalar no `.Xresources`, pode-se testá-la com um comando similar ao seguinte:

```
$ xterm -fn '-bitstream-courier-medium-r-normal-*-15-*-*-*-*-*-*-*'
```

Uma vez encontrado o tipo desejado de fonte de caracteres, que vamos supor seja “9x15”, o inserimos no arquivo `.Xdefaults` com uma linha do seguinte tipo:

```
XTerm.VT100.font: 9x15
```

Para tornar imediatamente efetiva a mudança, executa-se o comando:

```
$ xrdb -merge ~/.Xresources
```

Na próxima reinicialização do NetBSD não será mais necessário executar o comando precedente porque o arquivo `.xinitrc` comumente realiza automaticamente a fusão (merge) do .Xresources com o funcionamento do X.

> **Nota:** note-se que
>
> ```
> XTerm.VT100.font: 9x15
> ```
>
> é diferente de
>
> ```
> XTerm*font: 9x15
> ```
>
> No segundo caso o caracter “*” age como caracter curinga (wildcard) e, portanto, a diretriz tem efeito sobre todas as fontes de caracteres (por exemplo, também sobre aquelas dos menus). A escolha de uma das duas formas depende, portanto, do efeito que se quer obter.

Examinando a página do manual do XTerm (como também da maior parte dos programas do X) encontram-se indicações sobre os recursos que são passíveis de personalização.

Quase todos os programas X (aqueles escritos com os assim chamados “X Toolkit Intrinsics”), aceitam algumas opções comuns da linha de comando, tais como:

- -display display
- -geometry geometry
- -bg color, -fg color

- -fn font
- -xrm resourcestring

Veja-se a página de manual do X para detalhes adicionais.

15.11. Login gráfico com xdm

Quando se utiliza sempre o X para trabalhar, pode ser mais agradável inicializá-lo diretamente com a inicialização do computador e ter um login gráfico. Nada mais simples:

1. Criar o arquivo `.xsession` no próprio diretório home. Este arquivo faz as vezes do `~/.xinitrc` e pode ser também simplesmente um link para esse último.
2. Modificar o `/etc/rc.conf` desse modo:

   ```
   xdm=YES       xdm_flags=""       # x11 display manager
   ```

   Alternativamente pode-se acrescentar a seguinte linha no fim do `/etc/rc.local`:

   ```
   /usr/X11R6/bin/xdm
   ```

   Este método pode ser utilizado para inicializar, por exemplo, o kdm ou o gdm no lugar do xdm.

Os arquivos de configuração do xdm encontram-se em `/usr/X11R6/lib/X11/xdm`. Em Xservers o X é inicializado no terminal virtual vt05. Se se quer inicializá-lo em um outro terminal esse é o momento certo para vermos isso. Para um bom funcionamento do xdm, a fim de se evitar que dois processos entrem em conflito pelo teclado, convém inicializar o xdm em um terminal virtual em que o getty não está ativo. Na prática se no `Xservers` temos:

```
:0 local /usr/X11R6/bin/X :0 vt04
```

no `/etc/ttys` é oportuno ter:

```
ttyE3   "/usr/libexec/getty Pc"          vt220   off secure
```

(note-se que vt04 corresponde a ttyE3, porque a numeração dos vt começa em 1, enquanto que a dos ttyE começa em 0).

Para tornar a tela de login do xdm um pouco menos "miserável", basta intervir no arquivo de configuração do xdm. Por exemplo, pode-se mudar a cor do fundo adicionando ao arquivo `Xsetup_0` a seguinte linha:

```
xsetroot -solid SeaGreen
```

Ou, no lugar da cor, como fundo pode-se por uma imagem, usando xpmroot. Por exemplo:

```
xpmroot /path_to_xpm/netbsd.xpm
```

Jogando com os arquivos de configuração, podem-se obter muitos efeitos agradáveis.

Capítulo 16. Emulação do Linux

O NetBSD reescrito para as máquinas i386 é capaz de executar também uma grande parte dos programas do Linux em modo de emulação. Geralmente, quando se fala de emulação, pensa-se em algo de muito lento e ineficiente já que, habitualmente, as emulações comportam a reprodução através de softwares de arquiteturas de hardware muito diferentes daquela em que se trabalha. No caso da emulação do Linux, todavia, trata-se simplesmente do acréscimo de um sutil estrato de softwares já "adaptados" a uma série de chamadas do sistema que já são muito similares ao sistema emulado. O código do próprio aplicativo já foi elaborado nativamente para a mesma CPU e, portanto, não se deve temer redução da performance. Há até mesmo quem, não sem uma dose de exagero, chega a dizer que os programas para Linux definitivamente rodam melhor no NetBSD.

No presente capítulo explica-se como configurar a emulação do Linux para instalar um programa nativo desse sistema que, no exemplo, será o conhecidíssimo Acrobat Reader versão 4.

16.1. Instalando a emulação

A página do manual compat_linux(8) descreve em detalhe a instalação do emulador de Linux em um sistema NetBSD. Trata-se de uma das páginas mais amigáveis do manual: fica clara a intenção de ajudar o usuário a realizar uma operação, ao invés de fornecer um pacote de especificações e de detalhes técnicos. Para instalar a emulação Linux são necessários dois passos:

1. Configurar o kernel.
2. Instalar as bibliotecas Linux.

16.1.1. Configuração do kernel

A primeira operação a cumprir é a compilação de um kernel que suporte a compatibilidade com o Linux. Quem usa o kernel GENERIC não deve fazer nada porque ele já está habilitado para a compatibilidade com o Linux. Quem usa um kernel personalizado, todavia, deve ativar as opções:

```
option COMPAT_LINUX
option EXEC_ELF32
```

quando o kernel estiver pronto pode-se passar à instalação do software necessário.

16.1.2. Instalação das bibliotecas Linux

As bibliotecas Linux podem ser retiradas de uma distribuição Linux, contanto que suficientemente atualizada. Para simplificar a operação convém recorrer ao sistema de gerenciamento dos pacotes, que efetua a instalação de modo automático, utilizando bibliotecas do Suse Linux. As principais operações que serão realizadas são:

- A criação de um diretório *root secundário*, que será utilizado para os programas do Linux. Este diretório é `/emul/linux/`. Os programas Linux em emulação no NetBSD farão referência a esse diretório como se fosse o diretório raiz do sistema.

- A instalação das bibliotecas compartilhadas para Linux. A maior parte dos programas são, de fato, ligados dinamicamente e esperam, portanto, encontrar as bibliotecas necessárias instaladas no sistema. No que diz respeito ao Acrobat Reader, entrando-se no diretório `/usr/pkgsrc/print/acroread` e escrevendo **make depends**, obtém a seguinte mensagem:

  ```
  ===>  acroread-4.0 requires Linux glibc2 libraries - see compat_linux(8).
  ```

Ambas as operações serão realizadas pelo sistema de gerenciamento dos pacotes, sem necessidade de intervenções manuais por parte do usuário.

Antes de passar à instalação das bibliotecas propriamente ditas, carece instalar o pacote necessário para a gestão dos arquivos em formato *RPM*. Trata-se do `rpm-2.5.4`, que permitirá a extração dos arquivos das bibliotecas do Suse.

A seguir é necessário instalar o pacote `suse_base`. Os arquivos RPM Suse podem ser obtidos por download diretamente do sistema de pacotes ou, quem tem um CD-ROM Suse pode copiá-los no diretório `/usr/pkgsrc/distfiles/suse` e depois executar **make** e **make install**.

Com o mesmo método instalam-se `suse_compat`, `suse_libc5` e `suse_x11`, obtendo-se ao fim esta configuração:

```
# pkg_info -a | grep suse
suse_base-6.1p1     Linux compatibility package
suse_x11-6.1p1      Linux compatibility package for X11 binaries
suse_compat-6.1p1   Linux compatibility package with old shared libraries
suse_libc5-6.1p1    Linux compatibility package for libc5 binaries
```

16.1.3. Instalação do Acrobat Reader

Agora tudo está pronto para a instalação do programa Acrobat Reader (ou de outros programas Linux, naturalmente). Basta entrar no diretório `/usr/pkgsrc/print/acroread` e executar os comandos usuais:

```
make
make install
```

O script de instalação do Acrobat pede que se aceite as condições da licença. Feito isso, o programa está pronto para ser executado.

16.2. Estrutura dos diretórios

Examinando os resultados da instalação das bibliotecas e do programa, constatamos que `/emul/linux` é um link simbólico que aponta para `/usr/pkg/emul/linux`, onde se encontram os seguintes diretórios:

```
bin/
boot/
cdrom/
dev/
etc/
```

```
floppy/
home/
lib/
mnt/
opt/
proc/
root/
sbin/
usr/
```

> **Nota:** note-se que é sempre bom fazer referência a `/emul/linux`. O fato de que se trate de um link simbólico com `/usr/pkg/emul/linux` é um detalhe de implementação que poderia mudar em futuras versões do NetBSD.

Quanto espaço é necessário para instalar a emulação do Linux? Com a instalação já feita a estimativa do espaço dá os seguintes resultados:

```
# cd /usr/pkg/emul
# du -k linux
...
60525   linux/
```

Acrobat Reader, o programa propriamente dito, foi instalado no diretório usual dos pacotes pré-compilados, `/usr/pkg/bin/`.

Capítulo 17. Áudio

Manolo De Santis

O autor deste capítulo (Áudio) é Manolo De Santis

Este capítulo propõe-se a ser um guia breve para o uso de áudio no NetBSD, visto que jamais alguém gostaria de um computador que fosse mudo...

17.1. Um pouco de conhecimento do hardware

Antes de mais nada é necessário saber com que tipo de chipset de áudio estamos a tratar. De fato, não basta saber quem produziu a placa de áudio para conseguir fazê-la funcionar com o NetBSD porque, às vezes, os chipsets de uma certa placa são produzidos por terceiros. Mas não há o que temer! Na maioria dos casos o kernel do NetBSD é capaz de reconhecer o dispositivo. Para checarmos isso basta dar uma olhada no output do dmesg.

Portanto, executar um bom:

```
# dmesg | more
```

para ver qual chipset de áudio foi identificado. Em muitos casos, não se deve nem mesmo intervir no kernel porque o NetBSD identifica e autoconfigura (quase!) toda placa de áudio com muita facilidade.

Às vezes não se consegue fazer funcionar o áudio somente porque o chipset da placa não é suportado pelo NetBSD (e por isso, muita gente bem que pensou em adquirir uma nova placa) ou então porque é necessário perder um pouco de tempo com isso.

17.2. Configuração da BIOS

Pois bem, esta seção será útil apenas aos possuidores de PCs e compatíveis, já que em outras arquiteturas (tipo Amiga) não serve muito, se é que serve. A coisa mais importante para a utilização de uma placa de áudio em um PC com NetBSD é a configuração da BIOS. De fato, é necessário saber qual é o tipo de interface da placa de áudio que se tem.

A escolha é entre PCI e ISA.

Uma placa de tipo ISA é, em geral, mais difícil de configurar, sobretudo porque a BIOS, até certo ponto, quer meter o bedelho nisso.

Nos novos computadores disponíveis no comércio (aqueles produzidos em geral depois do fim de 1997) há uma opção da BIOS que cria grandes problemas para as placas de áudio ISA no NetBSD. Trata-se da ’PNP OS Installed’(encontra-se geralmente em ’PNP/PCI Configuration’). Pois bem, se esta opção está presente, é necessário desabilitá-la. Portanto, definí-la como ’NO’.

> **Nota:** pode também ocorrer que tudo funcione, ainda que deixemos a tal opção habilitada. Nunca se sabe!

17.3. Configuração dos dispositivos

Durante a instalação do NetBSD são criados todos os dispositivos de driver em `/dev`. Interessam-nos principalmente:

```
/dev/audio
/dev/sound
/dev/mixer
```

Se eles não estão presentes, podemos criá-los assim:

```
# cd /dev
# ./MAKEDEV all
```

Dessa forma são recriados todos os dispositivos de driver, incluindo os relacionados ao áudio.

Pois bem, nesse ponto é muito provável que a placa de áudio já esteja pronta para o uso.

Agora podemos testá-la, enviando um arquivo qualquer (mesmo de texto ou binário) diretamente ao `/dev/audio` ou ao `/dev/sound`, da seguinte maneira:

```
# cat NomeDoArquivo > /dev/audio
```

ou ainda

```
# cat NomeDoArquivo > /dev/sound
```

Se for possível ouvir alguma coisa, então não há problemas. Tudo funciona corretamente porque a placa é diretamente suportada pelo kernel do nosso sistema.

Em caso contrário, há necessidade de configurar o kernel definindo os dispositivos correspondentes à nossa placa de áudio.

17.4. Conselhos sobre a configuração do kernel

As placas de áudio suportadas pelo NetBSD são muitíssimas. De fato, no kernel GENERIC da versão 1.5 do sistema operacional, já estão habilitadas todas as opções correspondentes.

Alguns PCs não têm placa de som, mas um chipset de áudio integrado à placa-mãe, e tal função não está habilitada no kernel GENERIC. É necessário identificar estas linha no próprio arquivo de configuração do kernel GENERIC:

```
# Plug-and-Play BIOS and attached devices

#pnpbios*       at mainbus?

# mainboard audio chips
```

```
#ess*           at pnpbios? index ?     # ESS AudioDrive
#sb*            at pnpbios? index ?     # NeoMagic 256AV in sb mode
#wss*           at pnpbios? index ?     # NeoMagic 256AV in wss mode
#ym*            at pnpbios? index ?     # OPL3-SA3
```

Nesse ponto é necessário retirar os comentários para habilitar os chipsets de áudio integrados e recompilar o kernel.

> **Nota:** pessoalmente creio que seja melhor fazer uma cópia do arquivo de configuração GENERIC e trabalhar sobre ela, deixando portanto invariado o arquivo original, como se explica no Capítulo 7.

Voltando às placas de som, eventualmente devem ser configurados manualmente também os IRQ e os DMA.

Para um teste, podem-se deixar todas as opções habilitadas porque muitos chipsets de áudio somente podem funcionar emulando um outro dispositivo.

Geralmente muitos chipsets desfrutam de compatibilidade com SoundBlaster e OPL, mas na maioria dos casos o áudio funciona aproveitando o WSS.

OPL é um sintetizador MIDI produzido pela Yamaha e existem diferentes variantes dele no comércio (entre os quais OPL2, OPL3SA, OPL3SA2, etc.). Posso dizer que muitíssimas placas de som do mercado possuem este microchip ou um totalmente compatível. Por exemplo, todos os chips produzidos pela Crystal (entre os quais os comuns CS423X) possuem este chipset, e é exatamente este último que lhes permite o funcionamento no NetBSD!

WSS não é propriamente um microchip, antes... é o acrônimo de Windows Sound System. De fato, wss é um driver do kernel do NetBSD que suporta o sistema de áudio do Microsoft Windows. Muitas placas de áudio funcionam com o Windows exatamente porque aderem ao padrão WSS. A mesma coisa afortunadamente vale para o NetBSD.

Pessoalmente tenho um modo de provar diversas placas no NetBSD e posso dizer que muitas delas apenas funcionam se opl* e wss* estão habilitados no kernel.

Nenhum problema com as Sound Blaster Creative originais. Todas funcionam, inclusive a Sound Blaster Live 1024!

Uma vez que tudo funciona corretamente, podem-se desabilitar do kernel as opções relativas aos dispositivos de áudio que não nos interessam.

17.5. Comandos avançados

O NetBSD possui utilitários projetados exclusivamente para o controle e uso dos dispositivos de áudio. Eles são:

- audioctl
- mixerctl
- audioplay
- audiorecord

17.5.1. audioctl

Tendo aparecido pela primeira vez no NetBSD 1.3, serve para configurar determinadas variáveis de I/O, entre as quais a parametrização de freqüências para reprodução e gravação. Para ver o que é possível configurar dar um bom:

```
# audioctl -a | more
```

Por exemplo, para escutar música com a qualidade de CD, executar:

```
# audioctl -w play=44100,2,16,slinear_le
```

Isto define a freqüência em 44100Hz, 2 canais de áudio, 16 bits e mudança do encoding para slinear_le.

Para ver quais são os encodings suportados:

```
# audioctl encodings
```

aparecerá uma lista de todos os encondings suportados pela placa de áudio em uso.

17.5.2. mixerctl

Server claramente para regular o mixing audio e funciona de modo análogo ao audioctl.

17.5.3. audioplay

Com este é possível executar arquivos de áudio, mas aconselho a procura de alguns pacotes que dão a possibilidade de escutar formatos variados.

17.5.4. audiorecord

Útil para a gravação de arquivos de áudio.

Capítulo 18. Obtendo o código-fonte com CVS

Reinoud Koornstra

O autor deste capítulo (Obtendo o código-fonte com CVS) é Reinoud Koornstra

CVS (Concurrent Versions System) é um sistema de controle de versões que pode ser usado para manter atualizada a árvore das fontes do sistema presente em locais com modificações realizadas nas fontes "oficiais" do NetBSD. Há três árvores (ou mais precisamente, três ramos de uma mesma árvore) que podem ser usadas como referência para se manter atualizadas as fontes locais: o ramo *current*, com o qual se pode seguir a linha de desenvolvimento corrente (isto é, aquela mais avançada mas potencialmente menos estável), o ramo relativo à *release 1.5*, ao qual são aplicadas as correções relativas à versão 1.5 para eliminar erros e resolver problemas de segurança, e o ramo da *release 1.4*, ao qual são acrescentadas as correções relativas à versão 1.4. Essa última árvore não terá nenhum desenvolvimento ulterior.

> **Nota:** os três ramos recém descritos são os que existem no momento da elaboração desse capítulo. No futuro os números de versão das releases mudarão.

18.1. Obtendo as fontes do sistema e do userland

> **Nota:** pelo termo "userland" entende-se o conjunto de programas que fazem parte do sistema operacional NetBSD mas que não fazem parte do "kernel" propriamente dito.

O sistema de controle de versões CVS não integra as versões 1.4.x e 1.5.x do NetBSD (mas vai fazer parte da versão 1.6) e, portanto, se você ainda não o fez, é necessário instalá-lo.

```
% pkg_add ftp://ftp.netbsd.org/pub/NetBSD/packages/<OS Ver>/<arch>/All/cvs-1.11nb2.tgz
```

Os parâmetros <OS Ver> e <arch> são obtidos com o seguinte comando:

```
% sysctl kern.osrelease hw.machine_arch
```

Para fazer o download das fontes partindo do zero, isto é, sem ter nada no diretório `/usr/src`:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs checkout -rnetbsd-1-5 -PA src
```

Também é possível utilizar ssh, transferindo os dados criptografados:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
```

```
% cd /usr
% cvs checkout -rnetbsd-1-5 -PA src
```

Os comandos precedentes buscam as fontes no ramo relativo à release 1.5. Para fazer o download das fontes correntes basta omitir o parâmetro “-rnetbsd-1-5” do último comando. Para fazer o download do ramo relativo à release 1.4, utiliza-se o parâmetro “-rnetbsd-1-4” na última linha.

Para atualizar uma árvore de fontes já existente, por exemplo para a release 1.5, executar os seguintes comandos:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

ou, com ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

Para atualizar uma árvore das fontes com a versão corrente é necessário omitir o parâmetro “-rnetbsd-1-5”. Para atualizar uma árvore das fontes com a release 1.4, utilizar o seguinte parâmetro: “-rnetbsd-1-4”.

Para fazer a atualização de uma árvore de fontes que não esteja “limpa” (porque foi feito um “build” de uma parte ou de todas as fontes, ou do próprio kernel) e na qual não foi executado o **make cleandir**, é necessário executar um **make obj** no diretório `/usr/src`:

```
% makedir /usr/obj
% make obj
```

> **Nota:** os diretórios object são necessários para efetuar uma operação de atualização de uma árvore que não esteja limpa. Quando é realizada uma operação de construção (“build”) em uma árvore, ou em uma parte dela, sem acompanhá-la de um “make clean” ou de um “make cleandir”, dizemos que a árvore permanece *suja*. Nesse caso o CVS poderia tentar criar diretórios que têm o mesmo nome de alguns arquivos binários já existentes e falhar (por exemplo, onde havia o diretório “groff” agora está construído um binário que se chama “groff”, mas o CVS deve criar todos os diretórios vazios antes de eliminá-lo).

Agora é possível realizar a atualização com o “cvs update”. Ou então pode-se dar um **make cleandir** no `/usr/src` antes de usar o CVS. Este método é mais simples e mais veloz quanto ao make dos objectdir quando se deve atualizar uma árvore suja.

A execução é notavelmente mais veloz em comparação com o sup e as atualizações são mais freqüentes.

Não é possível definir com exatidão o tempo necessário para o download de todas as fontes. A título de exemplo, em uma linha T1 a operação exige um pouco mais de uma hora, de acordo com a qualidade da conexão. Para utilizar um modem convém aproveitar-se da compressão dos dados. Por exemplo:

```
% cvs -z5 checkout ...
```

ou

```
% cvs -z5 -d $CVSROOT update ...
```

Nos exemplos precedentes o número 5 indica o nível de compressão desejado. Pode-se especificar um número compreendido entre 1 e 9, onde 1 representa a compressão mais veloz e 9 a mais eficaz, embora mais lenta. Tenha-se presente que a compressão aumenta a carga de trabalho do servidor.

> **Nota:** é necessário deslocar-se para o diretório `/usr/src/sys/arch/$arch/compile/$kernel_conf_name`, executar o comando **make clean** e remover o diretório `$kernel_conf_name`, porque ele também poderia não estar “limpo”. Desse modo garante-se que a atualização da parte relativa ao kernel chegará a bom termo.

18.2. Obtendo o pkgsrc

Pkgsrc (as fontes do conjunto dos pacotes) compreende um conjunto de utilitários e de arquivos de configuração que permitem compilar e utilizar no NetBSD um alentado grupo de aplicativos, tornando simples a instalação e a desinstalação de softwares no sistema.

Para fazer o download de todo o pkgsrc partindo do zero:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
(the password is: "anoncvs")
% cvs checkout -PA pkgsrc
```

ou, com ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs checkout -PA pkgsrc
```

Os comandos anteriores criam o diretório `pkgsrc` no `/usr` e gravam as fontes do sistema de pacotes inteiro no `/usr/pkgsrc`

Para atualizar o pkgsrc executar os seguintes comandos:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
```

```
(the password is: "anoncvs")
% cvs -d $CVSROOT update -PAd pkgsrc
```

ou, com ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -PAd pkgsrc
```

Antes de efetuar a atualização é bom assegurar-se de que o diretório `pkgsrc` esteja “limpo”. Não é seguro dar o comando **make clean** no **`/usr/pkgsrc`**.

Capítulo 19. CCD: configuração

Brian A Seklecki

O autor deste capítulo (CCD: configuração) é Brian A. Seklecki <lavalamp@burghcom.com>

O driver CCD permite ao usuário “concatenar” vários discos físicos entre si, formando um único pseudo-volume. CCD faculta, além disso, superar uma limitação do CMU RAIDFrame, que não permite configurar RAID0 (sistema de arquivos em vários discos) em discos de geometria diferente. CCD permite ainda utilizar um “interleave” para melhorar a performance, diminuindo o desperdício de espaço (esta característica não é descrita no presente exemplo).

Os passos necessários para a configuração do CCD são os seguintes:

1. Instalação dos discos
2. Configuração do suporte ao CCD no kernel
3. Criação do disklabel para cada volume membro do CCD
4. Criação do arquivo de configuração do CCD
5. Inicialização do dispositivo CCD
6. Criação de um sistema de arquivos 4.4BSD/UFS no novo dispositivo CCD
7. Montagem do sistema de arquivos CCD

Este exemplo faz referência à configuração do CCD no NetBSD/Sparc 1.5. O CCD residirá em quatro discos SCSI em um chassis externo da Sun, conectado à porta externa SCSI de 50 pin.

19.1. Instalação dos discos

Esta fase depende naturalmente da plataforma e do hardware utilizado.

Exemplo de **dmesg**:

```
Disk #1:
  probe(esp0:0:0): max sync rate 10.00MB/s
  sd0 at scsibus0 target 0 lun 0: <SEAGATE, ST32430N SUN2.1G, 0444> SCSI2 0/direct fixed
  sd0: 2049 MB, 3992 cyl, 9 head, 116 sec, 512 bytes/sect x 4197405 sectors

Disk #2
  probe(esp0:1:0): max sync rate 10.00MB/s
  sd1 at scsibus0 target 1 lun 0: <SEAGATE, ST32430N SUN2.1G, 0444> SCSI2 0/direct fixed
  sd1: 2049 MB, 3992 cyl, 9 head, 116 sec, 512 bytes/sect x 4197405 sectors

Disk #3
  probe(esp0:2:0): max sync rate 10.00MB/s
```

```
  sd2 at scsibus0 target 2 lun 0: <SEAGATE, ST11200N SUN1.05, 9500> SCSI2 0/di-
rect fixed
  sd2: 1005 MB, 1872 cyl, 15 head, 73 sec, 512 bytes/sect x 2059140 sectors

Disk #4
  probe(esp0:3:0): max sync rate 10.00MB/s
  sd3 at scsibus0 target 3 lun 0: <SEAGATE, ST11200N SUN1.05, 8808 > SCSI2 0
  sd3: 1005 MB, 1872 cyl, 15 head, 73 sec, 512 bytes/sect x 2059140 sectors
```

19.2. Configuração do kernel

Para o suporte aos dispositivos CCD é necessário ativar a seguinte opção no arquivo de configuração do kernel (ativada por default no GENERIC):

```
pseudo-device  ccd  4    # concatenated disk devices
```

No meu arquivo de configuração do kernel, codifico diretamente a associação entre os dispositivos correspondentes `/dev`, para ter o máximo controle:

```
sd0     at scsibus0 target 0 lun ?
# SCSI disk drives
sd1     at scsibus0 target 1 lun ?
# SCSI disk drives
sd2     at scsibus0 target 2 lun ?
# SCSI disk drives
sd3     at scsibus0 target 3 lun ?
# SCSI disk drives
sd4     at scsibus0 target 4 lun ?
# SCSI disk drives
sd5     at scsibus0 target 5 lun ?
# SCSI disk drives
sd6     at scsibus0 target 6 lun ?
# SCSI disk drives
```

19.3. Escrevendo os disklabels

É necessário criar um sistema de arquivos especial para cada disco membro do CCD. No presente exemplo, dever-se-á criar o disklabel para:

```
/dev/rsd0c
/dev/rsd1c
/dev/rsd2c
/dev/rsd3c
```

Recorde-se de criar sempre o disklabel para o dispositivo de caracteres, não para o dispositivo de blocos. Por exemplo: `/dev/r{s,w}d*`

> **Nota:** em todas as plataformas, a repartição `c` é reservada e representa a partição do NetBSD inteira.

Na plataforma i386 a repartição `d` é reservada e representa o disco inteiro.

Antes de criar novos disklabels, podem-se remover os pré-existentes com uma série de comandos similares aos seguintes:

Nota: na plataforma i386, substituir `rsd#c` por `rsd#`

```
# dd if=/dev/zero of=/dev/rsd0c bs=8k count=1
# dd if=/dev/zero of=/dev/rsd1 bs=8k count=1
# dd if=/dev/zero of=/dev/rsd2 bs=8k count=1
# dd if=/dev/zero of=/dev/rsd3 bs=8k count=1
```

O disklabel por default para os discos será parecido com o seguinte:

```
# disklabel -r /dev/rsd0c
[...snip...]
bytes/sector: 512
sectors/track: 116
tracks/cylinder: 9
sectors/cylinder: 1044
cylinders: 3992
total sectors: 4197405
[..snip...]
3 partitions:
#        size   offset    fstype   [fsize bsize   cpg]
  c:  4197405       0     unused     1024  8192        # (Cyl.    0 - 4020*)
```

É necessário criar uma "repartição (slice)" na partição NetBSD que ocupe a partição inteira. A repartição deve iniciar-se pelo menos um cilindro depois do início do disco ou da partição para deixar espaço ao disklabel especial exigido pelo CCD. O offset deve ser 1 x setores/cilíndro. Portanto, o valor do "tamanho (size)" deve ser igual ao número de setores totais menos o número de setores presentes em um cilíndro.

Nota: certificar-se de que o offset de uma repartição do tipo "ccd" seja um múltiplo do valor "setores/cilíndro".

Edite seus disklabels adequadamente. Assegure-se de especificar o caminho (path) para o device de caracteres, não o device de blocos.

Nota: antes de iniciar a edição dos disklabels, verificar para que seja exportada a variável de ambiente EDITOR.

```
# disklabel -e /dev/rsd0c
```

Nota: o tipo ("fstype") da repartição deve ser `ccd`.

Uma vez que a partição conterá uma só repartição, pode-se reciclar a repartição `c`, que normalmente fica reservada. O disklabel fica, portanto, assim colocado:

```
3 partitions:
#        size   offset    fstype   [fsize bsize   cpg]
  c:  4196361     1044       ccd                     # (Cyl. 1 - 4020*)
```

Também é possível estabelecer uma repartição diferente da `c`, como no exemplo seguinte:

```
3 partitions:
#        size   offset    fstype   [fsize bsize   cpg]
  a:  4196361     1044       ccd                     # (Cyl. 1 - 4020*)
  c:  4197405        0    unused     1024  8192      # (Cyl. 0 - 4020*)
```

Ao término da operação asseguremo-nos de gravar o disklabel, que deve passar pela checagem de validade, sob pena de rejeição do comando **disklabel**.

19.4. Configuração do CCD

Uma vez terminada a criação dos disklabels para todos os discos, pode-se passar à geração do arquivo de configuração, que reside por default no diretório `/etc`. O formato é o seguinte:

```
#ccd    ileave    flags   component    devices
```

> **Nota:** no que diz respeito ao valor do "ileave", se é definido como zero, então os discos são concatenados. Mas se usamos um valor igual ao número de setores/cilíndro, então os discos ficam intercalados.

No exemplo corrente, o conteúdo do arquivo é:

```
# more /etc/ccd.conf
ccd0  0  none /dev/sd0c /dev/sd1c /dev/sd2c /dev/sd3c
```

> **Nota:** o arquivo de configuração do CCD faz referência aos sistemas de arquivos CCD recém criados. É necessário utilizar os dispositivos de caracteres e não os de blocos.

19.5. Inicialização do CCD

Depois de ter verificado a correção das configurações recém efetivadas, pode-se finalmente inicializar o dispositivo usando o comando **ccdconfig**:

```
# ccdconfig -c -f /etc/ccd.conf
```

A configuração pode ser cancelada com a flag `-u`:

```
# ccdconfig -u -f /etc/ccd.conf
```

A inicialização do CCD ativará os dispositivos `/dev`: `/dev/{,r}ccd#`:

```
# ls -la  /dev/{,r}ccd0*
brw-r-----  1 root  operator   9, 0 Apr 28 21:35 /dev/ccd0a
brw-r-----  1 root  operator   9, 1 Apr 28 21:35 /dev/ccd0b
brw-r-----  1 root  operator   9, 2 May 12 00:10 /dev/ccd0c
brw-r-----  1 root  operator   9, 3 Apr 28 21:35 /dev/ccd0d
brw-r-----  1 root  operator   9, 4 Apr 28 21:35 /dev/ccd0e
brw-r-----  1 root  operator   9, 5 Apr 28 21:35 /dev/ccd0f
brw-r-----  1 root  operator   9, 6 Apr 28 21:35 /dev/ccd0g
brw-r-----  1 root  operator   9, 7 Apr 28 21:35 /dev/ccd0h
crw-r-----  1 root  operator  23, 0 Jun 12 20:40 /dev/rccd0a
crw-r-----  1 root  operator  23, 1 Apr 28 21:35 /dev/rccd0b
crw-r-----  1 root  operator  23, 2 Jun 12 20:58 /dev/rccd0c
crw-r-----  1 root  operator  23, 3 Apr 28 21:35 /dev/rccd0d
crw-r-----  1 root  operator  23, 4 Apr 28 21:35 /dev/rccd0e
crw-r-----  1 root  operator  23, 5 Apr 28 21:35 /dev/rccd0f
crw-r-----  1 root  operator  23, 6 Apr 28 21:35 /dev/rccd0g
crw-r-----  1 root  operator  23, 7 Apr 28 21:35 /dev/rccd0h
```

19.6. Criação do sistema de arquivos no dispositivo CCD

Agora pode-se criar o disklabel do novo disco virtual associado ao CCD (usar o dispositivo de caracteres):

```
# disklabel -e /dev/rccd0c
```

Como anteriormente, visto que é usada uma só repartição, é possível reciclar as repartição (slice)`c` ou criar uma repartição separada.

```
# disklabel -r /dev/rccd0c
# /dev/rccd0c:
type: ccd
disk: ccd
label: default label
flags:
bytes/sector: 512
sectors/track: 2048
tracks/cylinder: 1
sectors/cylinder: 2048
cylinders: 6107
total sectors: 12508812
rpm: 3600
interleave: 1
trackskew: 0
cylinderskew: 0
headswitch: 0           # microseconds
```

```
track-to-track seek: 0  # microseconds
drivedata: 0
#        size   offset    fstype   [fsize bsize   cpg]
  c: 12508812        0    4.2BSD     1024  8192    16  # (Cyl. 0 - 6107*)
```

Naturalmente é necessário criar o sistema de arquivos:

```
# newfs /dev/rccd0c
Warning: 372 sector(s) in last cylinder unallocated
/dev/rccd0c:    12508812 sectors in 6108 cylinders of 1 tracks, 2048 sectors
        6107.8MB in 382 cyl groups (16 c/g, 16.00MB/g, 3968 i/g)

super-block backups (for fsck -b #) at:
[...]
```

19.7. Montando o sistema de arquivos

O sistema de arquivos do dispositivo CCD criado há pouco pode ser montado em um ponto de montagem do sistema. Deve ser montada a repartição etiquetada `ffs` ou `4.4BSD`:

```
# mount /dev/ccd0c /mnt
```

Depois:

```
# export BLOCKSIZE=1024; df
Filesystem  1K-blocks     Used   Avail Capacity  Mounted on
/dev/sd6a      376155   320290   37057    89%    /
/dev/ccd0c    6058800       1  5755859     0%    /mnt
```

Ao término deste exemplo você deveria estar em condição de configurar um dispositivo CCD. Consulte as páginas do manual para detalhes adicionais sobre os comandos descritos nesse capítulo e para ativar automaticamente o dispositivo CCD na inicialização, mediante os arquivos RC.

Capítulo 20. Operações variadas

Este capítulo reúne, sem uma ordem determinada, vários argumentos que não encontraram colocação nos outros capítulos.

20.1. A criação dos disquetes de instalação

O autor desta seção (A criação dos disquetes de instalação) é Eric Delcamp

Para criar os disquetes de instalação é necessário que o "pseudo-device" *vnd* esteja habilitado no kernel.

1. Em primeiro lugar é necessário criar o kernel para gravar no disquete, que suporemos chamar-se FLOPPY, que deve derivar de um dos kernels INSTALL.
2. Uma vez compilado, o kernel estará no arquivo `/sys/arch/i386/compile/FLOPPY/netbsd`.
3. Deslocar-se para o diretório `/usr/src/distrib/i386/floppies/ramdisk` e executar o comando

   ```
   # make
   ```

   Isto cria o arquivo `ramdisk.fs` naquele diretório.
4. Deslocar-se para o diretório `/usr/src/distrib/i386/floppies/fdset` e executar o comando

   ```
   # make KERN=/sys/arch/i386/compile/FLOPPY/netbsd
   ```

5. O comando precedente criou um ou dois arquivos (depende da dimensão do kernel) de nome `boot1.fs` e `boot2.fs`. Nesse ponto é necessário transferir estes arquivos para o disquete com o comando (ou melhor, os comandos):

   ```
   # dd if=boot1.fs of=/dev/fd0a bs=36b
   # dd if=boot2.fs of=/dev/fd0a bs=36b
   ```

   Naturalmente, entre um comando e outro é necessário por um novo disquete no drive.

20.2. Criação de um CD-ROM

Para criar um CD-ROM de dados utilizam-se os programas mkisofs e cdrecord. O NetBSD suporta a utilização tanto dos gravadores SCSI quanto dos gravadores IDE. Esses últimos podem ser utilizados de maneira integralmente transparente, sem a necessidade de utilizar emulações SCSI ou outros artifícios porque o driver é capaz de receber diretamente os comandos ATAPI (como sempre o NetBSD consegue encontrar a solução mais simples e mais elegante).

A criação de um CD transcorre em duas fases: primeiro criamos uma imagem ISO do CD no disco rígido utilizando o programa mkisofs e depois se grava a imagem no CD com o cdrecord. Nos exemplos seguintes supõe-se a utilização de um gravador IDE/ATAPI entre os suportados pelo cdrecord: o gravador é reconhecido pelo sistema como um disco master secundário.

> **Nota:** a gravação de um CD-ROM é uma operação na qual a velocidade de execução é crítica. Particularmente o fluxo de dados deve ser constante para o gravador. O cdrecord jamais deve encontrar-se com o buffer dos dados vazio ou quase vazio. Portanto é aconselhável executar esse tipo de operação quando o sistema não esteja sobrecarregado (vamos deixar combinado que é melhor não recompilar o kernel enquanto se grava...)

```
cd1 at atapibus1 drive 0: <HP CD-Writer Plus 8100> type 5 cdrom removable
```

20.2.1. Criação da imagem ISO

SidebarO sistema de arquivos do CD-ROM

Os CDs-ROM podem ser criados utilizando-se diversos sistemas de arquivos incompatíveis (ou eventualmente apenas em parte compatíveis) entre si. Portanto, um CD produzido em um sistema nem sempre é legível em outro sistema. Esta situação fica evidente quando se procura ler no NetBSD um CD produzido por um programa de masterização que opera em Windows.

O formato ISO9660 (de 1988) é o primeiro que foi proposto e representa uma espécie de denominador comum entre os vários sistemas operacionais (Apple, MS-DOS, Unix e VMS). Um CD produzido em ambiente Windows com este formato será sujeito a várias limitações (nomes e arquivos em formato 8.3 e, no máximo, 8 níveis de subdiretórios). Por esse motivo, quando se cria um CD de instalação do NetBSD no Windows 95/98/NT, os nomes dos arquivos, que apareciam normalmente no "gerenciamento de recursos" do Windows, aparecerão truncados no CD.

Para resolver esses tipos de limitações, foram introduzidas extensões ao formato ISO9660 original. A desvantagem destas extensões é que, em geral, não são suportadas por todos os sistemas operacionais.

O formato Joliet foi introduzido pela Microsoft para seus próprios sistemas e expande o ISO9660 permitindo, por exemplo, nomes longos para os arquivos. Em geral um CD neste formato não é legível nos sistemas Unix (o NetBSD-current também suporta o formato Joliet).

Para os sistemas Unix, entretanto, foram introduzidas extensões Rock Ridge, que permitem o suporte das convenções Unix mantendo embora a compatibilidade com o formato ISO9660. Este é o formato a ser usado para criar um CD para os sistemas Unix sob o NetBSD (como será visto, também é possível criar CDs compatíveis Rock Ridge/Joliet).

Além desses padrões ainda existem outros (por exemplo, "El Torito"), que permitem a produção de CDs inicializáveis, já suportados por todos os PCs mais recentes.

Para criar a imagem do CD no disco rígido é necessário antes de mais nada que haja espaço suficiente no disco. De acordo com a quantidade de dados, ela pode ter até 700MB. Supondo que os dados que nos interessam estejam no diretório `meusdados`, é preciso escrever

```
# mkisofs -aflrTv -o cdimage meusdados/
```

Uma vez criado o arquivo imagem `cdimage`, podemos examiná-lo montando-o como se fosse um filesystem para verificar se não existem erros e não desperdiçar um precioso CD.

```
# ls -l cdimage
-rw-rw-r--  1 auser     user  284672 Dec  1 11:58 cdimage
# vnconfig -v vnd0 cdimage 512/556/1/1
# mount -r -t cd9660 /dev/vnd0c /mnt
  ... browsing su /mnt ...
# umount /mnt
# vnconfig -u vnd0
```

O valor 556 foi obtido dividindo a dimensão do arquivo, 284672, por 512.

> **Criando um CD híbrido:** mkisofs é capaz também de criar CDs em formato Joliet (legíveis pelos sistemas Microsoft) e até mesmo CDs híbridos, que usam tanto o formato Rock Ridge quanto o formato Joliet e que se tornam legíveis seja pelo NetBSD, seja pelo Windows. Por exemplo, para criar um CD híbrido:
>
> ```
> $ mkisofs -l -J -R -o cd.iso mieidati/
> ```
>
> Fazer referência à página de manual do mkisofs para os detalhes sobre as opções aceitas pelo programa.

20.2.2. Gravação da imagem no disco

Na segunda fase a imagem recém criada é gravada no CD com o comando

```
# cdrecord -v speed=2 dev=/dev/rcd1d cdimage
```

> **Nota:** no caso do CDR ATAPI é necessário usar o dispositivo *rcd#d*, porque o dispositivo 'a' não suporta o envio direto de comandos ATAPI.

Também é possível efetuar um teste de gravação, desabilitando o laser antes de proceder à gravação propriamente dita. Basta adicionar ao comando precedente as opções `-dummy` e `-nofix`. Por exemplo

```
# cdrecord -v -dummy -nofix speed=2 dev=/dev/rcd1d cdimage
```

A fase de criação da imagem e a fase da gravação no disco podem ser combinadas em um só *job*, evitando-se a geração do arquivo-imagem temporário no disco rígido. O comando é o seguinte

```
# (nice -18 mkisofs -aflrT mieidati/) | cdrecord -v fs=16m speed=2 dev=/dev/rcd1d -
```

A opção `fs=16` serve para aumentar a dimensão do *fifo*, evitando erros de underflow do buffer dos dados usados pelo cdrecord.

20.2.3. Cópia de um CD

Para efetuar a cópia direta de um CD usa-se a opção `-isosize` do cdrecord, executando o seguinte comando

```
# cdrecord -v fs=16m -isosize speed=2 dev=/dev/rcd1d /dev/rcd0d
```

> **Nota:** se utilizamos dois CDs(-RW) IDE/ATAPI, é aconselhável montá-los em canais diferentes para termos um fluxo de dados melhor. Por exemplo:
>
> ```
> wd0: hard disk master ide primário
> cd0: leitor de CD slave ide primário
> cd1: gravador de CD master ide secundário
> ```

20.2.4. Criação de um CD inicializável

Não há diferenças especiais entre a criação de um CD-ROM normal e a de um inicializável. Para este último é necessário ter uma cópia do arquivo de boot para inserí-lo no CD, informando o mkisofs. Por exemplo:

```
# mkisofs -avr -b boot.fs mieidati/ > cdimage
```

`boot.fs` é o arquivo de inicialização do CD-ROM, cujo caminho (path) deve ser indicado de modo relativo ao diretório `meusdados`.

20.3. Sincronizando o relógio do sistema

Por qualquer motivo estranho parece que os relógios de sistema jamais funcionem direito e não é raro encontrarem-se com uma defasagem de muitos minutos. O problema se agrava quando se tem que mexer com vários hosts conectados em rede: com boa probabilidade jamais conseguiremos mantê-los sincronizados. Para resolver o problema vem nos socorrer o protocolo NTP (Network Time Protocol) versão 3, que permite manter sincronizados os relógios de muitas workstations conectadas em rede, utilizando o servidores adequados.

Graças ao protocolo NTP é possível tanto simplesmente manter a hora correta de um host único quanto da rede inteira. Trata-se de um protocolo um pouco complexo, que define a estrutura hierárquica master-slave de uma série de servidores, subdividindo-os em estratos. No topo da hierarquia estão os servidores do estrato 1, conectados a um hardware que é a fonte de referência para a data e para a hora. Os servidores de estrato 2 sincronizam-se via NTP com os do estrato 1 e, por sua vez, funcionam como servidores secundários para os hosts dos estratos inferiores. Esta estrutura hierárquica evita o congestionamento que se teria se todos os hosts clientes fizessem referência aos servidores do estrato 1. Se, por exemplo, se quer sincronizar uma rede inteira, não se devem conectar todos os hosts ao mesmo servidor público do estrato 1. Em vez disso, cria-se um servidor local que se sincroniza com o público e os hosts restantes sincronizam-se com o servidor local.

Afortunadamente, para desfrutar dos instrumentos NTP não é necessário compreender os detalhes do protocolo e da sua implementação (quem estiver interessado pode fazer referência ao documento RFC 1305) mas é suficiente saber como abrir alguns programas. No NetBSD já estão presentes os programas necessários para se aproveitar desse protocolo, derivados da implementação xntp. Nesta seção descreve-se um método simples para manter atualizada a hora do sistema.

Em primeiro lugar é necessário o endereço dos servidores públicos de NTP para utilizá-los como referência. Uma lista completa pode ser encontrada em http://www.eecis.udel.edu/~mills/ntp/servers.html, mas para a Itália podem ser usados tempo.cstv.to.cnr.it e time.ien.it, que são dois servidores de estrato 1.

Nesse ponto, para sincronizar o relógio do sistema basta executar o comando (é necessário ser root):

```
# ntpdate -b tempo.cstv.to.cnr.it time.ien.it
```

A opção `-b` serve para indicar ao **ntpdate** para assumir imediatamente o novo valor da hora do sistema, ao invés de procurar corrigir o atual, e é aconselhada quando a diferença entre a hora correta (a do servidor) e a local pode ser grande.

Já vimos como funciona o ntpdate. Agora é necessário decidir o melhor modo para inicializá-lo automaticamente. Se o host a sincronizar está sempre ligado à Internet, a melhor coisa é abrir o programa a cada reinicialização especificando

```
ntpdate=YES      ntpdate_hosts="time.ien.it"
```

no `/etc/rc.conf`. O nome do servidor pode ser especificado em ntpdate_hosts. Do contrário o script procurará obtê-lo do arquivo `/etc/ntp.conf`.

Se o host está conectado à Internet através de um provedor e a conexão não está sempre ativa, pode-se lançar o comando ntpdate a partir do script `ip-up`, como se explica no Capítulo 10. Nesse caso a linha a ser inserida no arquivo é:

```
/usr/sbin/ntpdate -s -b time.ien.it
```

(o caminho é necessário, senão o script não encontrará o programa). A opção `-s` produz o arquivamento das mensagens, que depois poderemos encontrar em `/var/log/messages` (salvo configuração diferente do `syslog.conf`).

Além do ntpdate estão presentes vários outros comandos e também é possível fazer um host da rede tornar-se o servidor NTP dos outros hosts clientes. O servidor interno sincronizar-se-á com os servidores de referência externos à rede e os clientes se vão sincronizar com o nosso servidor. Para esse tipo de configuração deve-se utilizar o dæmon **xntpd**, criando o arquivo de configuração adequado `/etc/ntp.conf`. Por exemplo:

```
server time.ien.it
server tempo.cstv.to.cnr.it
```

O xntpd pode ser ainda inicializado a partir do `rc.conf`, habilitando-se a opção adequada.

```
xntpd=YES
```

Para a sincronização interna à rede, como alternativa à configuração descrita, também é possível usar o dæmon timed, nascido especificamente com o 4.3BSD. Também o timed baseia-se em um esquema master-slave. Quando é inicializado em uma máquina o timed pede ao master a hora do sistema da rede e corrige apropriadamente a hora da máquina local. Pode-se, assim, utilizar uma estrutura mista, na qual um dos hosts se sincroniza com um servidor NTP público e faz o papel de master para a rede local, cujos clientes, por sua vez, sincronizam-se com o timed. No host de referência devem estar ativos seja o NTP, seja o timed, e é necessário prestar atenção para que não conflitem (o timed deve ser inicializado com a opção `-F hostname`. Desse modo não tentará atualizar o relógio local).

20.4. Instalando o gerenciador de inicialização (boot manager)

O programa de instalação do NetBSD pode instalar seu próprio gerenciador de inicialização, se o usuário assim o desejar. No caso de uma instalação errada ou falha é possível instalá-lo de novo ou modificar suas propriedades com o comando **fdisk**. Por exemplo:

```
# fdisk -B wd0
```

Se o sistema não se inicializa a partir do disco rígido, pode-se fazer o boot com o disquete de inicialização usado para a instalação e acionar o kernel do disco rígido com um comando como o seguinte:

```
> boot wd0a:netbsd
```

> **Nota:** em alguns casos o **fdisk -B** não parece dar o resultado desejado. Por exemplo, quando instalamos ou removemos com freqüência outros sistemas operacionais como o Windows 95. Nesse caso pode ser útil fazer um **fdisk /mbr** do DOS (por exemplo, com um disquete) e depois tentar de novo. O mesmo resultado pode-se obter no NetBSD com o comando **fdisk -i wd0**.

20.5. Apagando o disklabel

Ainda que a deleção do disklabel não seja uma operação freqüente, pode ser útil saber como fazer em caso de necessidade. Naturalmente trata-se de uma operação a ser executada apenas quando se sabe exatamente aquilo que se está fazendo. Por exemplo:

```
# dd if=/dev/zero of=/dev/rwd0c bs=8k count=1
```

Desse modo fica deletado o disklabel mas não a tabela das partições do MBR. Para deletar completamente o disco usar `wd0d`. Por exemplo:

```
# dd if=/dev/zero of=/dev/rwd0d bs=8k
```

20.6. Speaker

Para produzir um som no speaker (por exemplo, uma advertência ao fim de um script), pode-se utilizar o device *spkr* do kernel, que aparece através do dispositivo `/dev/speaker`. Por exemplo, nas mailing lists encontrei uma mensagem que sugeria:

```
echo 'BPBPBPBPBP' > /dev/speaker
```

> **Nota:** o dispositivo *spkr* não está habilitado no kernel genérico e, portanto, para usá-lo precisa compilar um kernel personalizado.

20.7. Memory file system

Para melhorar a performance dos serviços, dadas determinadas configurações, pode-se utilizar um *memory file system*. A título de exemplo, essa estratégia pode ser útil quando se compila muito e se dispõe de memória suficiente. Para criar um sistema de arquivos (file system) na memória, deve-se acrescentar uma linha do seguinte tipo no `/etc/fstab`:

```
/dev/wd0b /tmp  mfs     rw,async,noatime,nosuid,-s=20000
```

a opção `-s=20000` define as dimensões em cerca de 10 MB (20000 setores), que deveriam ser suficientes para a compilação do kernel.

20.8. A senha do root foi esquecida?

Quando se esquece a senha do root nem tudo está perdido e ainda se pode recuperar o sistema. Os passos a cumprir são os seguintes: inicializar em modo monousuário, montar / e mudar a senha do root. Em mais detalhes:

1. Inicializar em modo monousuário: quando aparece o prompt do boot e se inicia a contagem regressiva de cinco segundos, executar o comando:

   ```
   > boot -s
   ```

2. Diante da solicitação

   ```
   Enter pathname of shell or RETURN for sh:
   ```

   apertar Enter.

3. Executar os seguintes comandos

   ```
   # fsck -y /
   # mount -u /
   # fsck -y /usr
   # mount /usr
   ```

4. Mudar a senha do root com **passwd**.

5. Acionar o comando **exit** para passar ao modo multiusuário.

20.9. Adicionando um disco rígido novo

Esta seção explica como adicionar um novo disco rígido a um sistema NetBSD. No exemplo proposto serão adicionados um controlador SCSI e um disco rígido conectado ao novo controlador. Quem não precisa instalar o controlador pode simplesmente pular a leitura da parte correspondente e ir diretamente para a configuração do disco rígido. A instalação de um disco rígido IDE é idêntica à apresentada no exemplo. Somente o nome do dispositivo será diferente (`wd#`) no lugar de (`sd#`).

Como sempre, antes de adquirir hardware novo é bom consultar a lista das compatibilidades de hardware do NetBSD para certificar-se de que o dispositivo escolhido seja suportado pelo sistema.

Concluída a instalação física do controlador e do disco rígido, assim como o cabeamento correspondente, inicializa-se o sistema e nos certificamos de que o disco tenha sido reconhecido com o comando **dmesg**. Por exemplo, para um controlador NCR-875 o output é o seguinte:

```
ncr0 at pci0 dev 15 function 0: ncr 53c875 fast20 wide scsi
ncr0: interrupting at irq 10
ncr0: minsync=12, maxsync=137, maxoffs=16, 128 dwords burst, large dma fifo
ncr0: single-ended, open drain IRQ driver, using on-chip SRAM
ncr0: restart (scsi reset).
scsibus0 at ncr0: 16 targets, 8 luns per target
sd0(ncr0:2:0): 20.0 MB/s (50 ns, offset 15)
sd0: 2063MB, 8188 cyl, 3 head, 172 sec, 512 bytes/sect x 4226725 sectors
```

Se o dispositivo não aparece no output é necessário certificar-se de que o kernel em uso o suporte e, eventualmente, compilar um kernel personalizado (veja-se o Capítulo 7).

Nesse ponto podem ser criadas as partições com o comando **fdisk**. Em primeiro lugar visualizamos o estado atual do disco:

```
# fdisk sd0
NetBSD disklabel disk geometry:
cylinders: 8188 heads: 3 sectors/track: 172 (516 sectors/cylinder)

BIOS disk geometry:
cylinders: 524 heads: 128 sectors/track: 63 (8064 sectors/cylinder)

Partition table:
0: sysid 6 (Primary 'big' DOS, 16-bit FAT (> 32MB))
    start 63, size 4225473 (2063 MB), flag 0x0
        beg: cylinder    0, head   1, sector  1
        end: cylinder  523, head 127, sector 63
1: <UNUSED>
2: <UNUSED>
3: <UNUSED>
```

No exemplo a partição DOS será eliminada e substituída por uma partição NetBSD que ocupará o disco inteiro. O comando **fdisk -u sd0** permite a modificação interativa das partições. As modificações serão escritas no disco somente no final, depois de um pedido de confirmação, o que permite que procedamos com uma certa tranqüilidade.

SidebarNota sobre a geometria do disco

A geometria desse disco pode parecer bastante confusa. O dmesg reporta 4226725 setores com C/H/S igual a 8188/3/172. Mas 8188*3*172=4225008 e não 4226725. Na realidade, na maior parte dos discos modernos a geometria não é fixa e o número de setores por trilha muda de acordo com o cilíndro. O disco reporta igualmente uma geometria C/H/S fixa, mas trata-se de uma geometria fictícia e o valor 172 é obtido dividindo-se o número total de setores (que é efetivamente 4226725) por 8188 e depois por 3. O resultado das divisões é 172.

A BIOS, ademais, pode utilizar, por sua vez, uma geometria "fictícia" diferente (e de fato usa C/H/S 524/128/63) chegando a 4225536 setores, que são sempre um pouco menos que os reais. Na seqüência, também para manter a compatibilidade com outros sistemas operacionais, usaremos a geometria vista pela BIOS, mesmo que essa leve ao desperdício de alguns setores (4226725-4225536=1189 setores = 594 KB).

Depois de ter executado **fdisk -u**, confirmamos as modificações e obtemos as seguintes partições:

```
Partition table:
0: sysid 169 (NetBSD)
    start 63, size 4225473 (2063 MB), flag 0x0
        beg: cylinder    0, head   1, sector  1
        end: cylinder  523, head 127, sector 63
1: <UNUSED>
2: <UNUSED>
3: <UNUSED>
```

Nesse ponto é necessário escrever o disklabel no disco. A seqüência correta de operações é:

```
# disklabel sd0 > tempfile
# vi tempfile
# disklabel -R -r sd0 tempfile
```

Note-se que não é possível efetuar diretamente

```
# disklabel -e sd0
```

que se obtém a seguinte mensagem:

```
disklabel: ioctl DIOCWDINFO: No disk label on disk;
use "disklabel -r" to install initial label
```

devido ao fato de que o disklabel ainda não está presente no disco.

Nesse exemplo criemos as seguintes partições com o disklabel, editando o arquivo `tempfile` como foi explicado acima:

```
#        size   offset    fstype [fsize bsize   cpg]
a:  2048004       63    4.2BSD   1024  8192    16 # (Cyl.  0*- 3969*)
c:  4226662       63    unused      0     0       # (Cyl.  0*- 8191*)
d:  4226725        0    unused      0     0       # (Cyl.  0 - 8191*)
e:  2178658  2048067    4.2BSD   1024  8192    16 # (Cyl.  3969*- 8191*)
```

> **Nota:** uma vez criado o disklabel, podemos otimizá-lo estudando o output do comando **newfs -N /dev/sd0a** que assinala se no fim de uma partição permanecem setores não alocados. Este valor pode-se utilizar para modificar o tamanho da própria partição.

Por último, criemos os novos sistemas de arquivos para as partições *a* e *e*.

```
# newfs /dev/sd0a
# newfs /dev/sd0e
```

Agora o disco está pronto para ser usado, com duas partições para montar. Por exemplo:

```
# mount /dev/sd0a /mnt
```

20.10. Password file is busy?

Para alguns usuários acontece o recebimento de uma misteriosa mensagem “Password file is busy” quando tentam modificar uma senha. A solução ao problema é simples: eliminar o arquivo `/etc/ptmp`, depois de ter verificado se não contém informações importantes (trata-se de uma cópia do arquivo `/etc/master.passwd` que por motivos inescrutáveis ficou em circulação ao invés de ser eliminado).

> **Nota:** a existência do arquivo `/etc/ptmp` pode causar também a visualização de mensagens de advertência na fase de inicialização. Por exemplo:
>
> ```
> root: password file may be incorrect - /etc/ptmp exists
> ```

20.11. Como recriar os dispositivos no diretório /dev

O autor desta seção é Reinoud Koornstra

Em primeiro lugar é necessário passar ao modo monousuário (single user) com as partições ainda montadas “rw” (read-write). Para fazer isso convém simplesmente dar o comando **shutdown now** em modo multiusuário ou reinicializar o sistema com a opção `-s` e tornar `/` e `/dev` passíveis de escrita e leitura com os seguintes comandos:

```
# mount -u /
# mount -u /dev
```

Depois:

```
# mkdir /nudev
# cd /nudev
# cp /dev/MAKEDEV* .
# sh ./MAKEDEV all
# cd /
# mv dev odev
# mv nudev dev
# rm -r odev
```

Alternativamente, com as fontes instaladas no sistema em `/usr/src`, pode-se fazer assim:

```
# mkdir /nudev
# cd /nudev
# cp /usr/src/etc/MAKEDEV.local .
# cp /usr/src/etc/etc.$arch/MAKEDEV .
# sh ./MAKEDEV all
# cd /
# mv dev odev; mv nudev dev
# rm -r odev
```

O valor a substituir em $arch pode ser detrminado com o seguinte comando:

```
# uname -m
```

ou com:

```
# sysctl hw.machine_arch
```

> **Nota:** O segundo método para recriar os dispositivos (o que se aproveita das fontes instaladas no sistema) pode adicionar alguns dispositivos àqueles já existentes (pelo menos na arquitetura i386). Por exemplo, atualmente é possível ter 16 partições ao invés de 8. Utilizando a versão original do MAKEDEV (aquela do diretório `/dev`), os novos dispositivos não serão criados.

Apêndice A. Informações

A.1. História deste guia

Este guia nasceu de uma série de anotações que me deveriam servir apenas de apontamentos e que foram originalmente escritas em formato troff com as macros *ms*. Com o passar do tempo, dei-me conta de que podiam ser úteis também para outros que quisessem aproximar-se do NetBSD e decidí inserí-las em meu site depois de ter encontrado o formato adequado. Acabei adotando a *SGML/DocBook*.

Para a redação deste guia foram utilizados os seguintes instrumentos:

- o editor vi fornecido com o NetBSD (nvi).
- O programa Jade que implementa as transformações DSSSL. O Jade gera diretamente os formatos HTML e RTF.
- o sistema TeX da coleção de pacotes do NetBSD. TeX foi utilizado como suporte para a geração dos formatos PS e PDF.
- o programa tgif para a geração das figuras.
- os programas gimp e xv para a elaboração de imagens e sua conversão entre os diversos formatos.

Sinceros agradecimentos a todos os desenvolvedores desses excepcionais instrumentos de trabalho.

Apêndice B. Instalação sem o sysinst

O autor deste apêndice é Wojciech Puchar

Este apêndice descreve o procedimento que utilizamos para instalar o NetBSD sem fazer uso do sysinst. Pode-se perguntar por que não usá-lo. Eis algumas razões:

- possibilidade de usar opções não padronizadas na criação dos sistemas de arquivos.
- porque é fácil!
- para ter total domínio do sistema e compreender todas as operações efetuadas.
- para aprender a preparar os próprios disquetes de instalação, inserindo outros acessórios úteis no lugar do sysinst.

Este apêndice descreve apenas a configuração dos sistemas i386, em que o NetBSD será o único sistema operacional presente no disco rígido. Procurei entretanto evidenciar as seções específicas desta arquitetura, esperando receber alguma contribuição sobre as outras (a serem enviadas, em inglês, para wojtek@3miasto.net).

Para o entendimento deste apêndice é suposto certo conhecimento dos comandos de base do Unix (cd, ls, cp, tar, etc.).

B.1. Partição e instalação: teoria

O processo de instalação é dividido da seguinte maneira:

1. particionamento do disco rígido
2. criação do/dos sistemas de arquivos
3. instalação do carregador de boot (dependente da plataforma e não presente em algumas versões como a versão para sparc, etc.)
4. descompressão dos tarballs nas partições criadas
5. configuração final (criação dos dispositivos `/dev`, `/etc/fstab`, fixação do hostname, etc.)

B.1.1. Particionamento

Os esquemas de particionamento podem ser simples ou até mesmo muito complexos. Para os fins do presente apêndice não interessa discutir qual seja o melhor esquema possível. Limitar-nos-emos a descrever um esquema clássico para o Unix, que compreende uma pequena partição root, uma partição de swap e uma partição `/usr` para todo o resto.

Neste apêndice, como em muitos manuais do NetBSD, falar-se-á freqüentemente de "label" e de "disk label". O disk label corresponde às estruturas utilizadas pelo NetBSD (e pelos sistemas BSD em geral) para memorizar as informações relativas às partições. Portanto, a criação do disklabel equivale ao particionamento.

B.1.2. Dimensões da partição de swap

Se o seu sistema tem memória em abundância e você está absolutamente certo de que jamais terá necessidade do swap, você pode pular esta seção. Todavia, nos modernos discos rígidos de grandes dimensões, geralmente não é problema deixar uma certa quantidade de espaço para a partição de swap, com o benefício de se poder inicializar sem problemas um grande número de programas ao mesmo tempo.

Não existem fórmulas simples para se determinar a dimensão da partição de swap. A única verdadeira regra diz que é melhor ter muito que ter muito pouco. Isso porque ficar sem swap significa ter que encerrar não apenas o último processo inicializado, mas também todos os que têm necessidade de alocar memória. Pode tratar-se até mesmo do /sbin/init, o que levará ao colapso do sistema. Se você tem um daqueles discos rígidos enormes >10GB, deixe ao menos 500MB para a área de swap (mas pode ser bem mais). A única contra-indicação é que o uso do swap comporta a alocação estática de 16 bytes para cada página i386 (4KB), o que significa usar 1MB de memória para cada 256MB de swap. Não é uma coisa catastrófica, mas em uma máquina com 16MB pode tornar-se um problema.

Resumindo: se você tem um disco suficientemente grande, aloque uma área generosa para o swap. Do contrário, procure regular-se de acordo com a memória necessária às aplicações que pretende utilizar simultaneamente.

B.1.3. Posicionamento da partição de swap

Pode-se pensar que o posicionamento da partição de swap seja indiferente. Isso é verdadeiro no que diz respeito aos discos de tipo mais antigo (por exemplo, MFM). Nos discos mais recentes (entenda-se discos com mais de 80MB), contudo, sobre as trilhas externas são gravados mais dados que nas internas, por causa de suas maiores dimensões. Isto leva a um I/O mais veloz nas trilhas externas, que se encontram no início do disco. Portanto, a área de swap será mais veloz se se encontra na zona inicial.

B.1.4. Sistemas de arquivos

O sistema de arquivos padrão usado pelo NetBSD é aquele a que se denomina “ffs” ou “ufs” (Fast File System ou Unix File System). Ainda que “fast” (rápido) não signifique “o mais veloz do mundo”, trata-se de um sistema de arquivos que se presta bem para o uso geral e dotado de mecanismos de proteção em caso de queda do sistema (crash). Para a criação do sistema de arquivos pode-se limitar à execução do comando **newfs** sem qualquer parâmetro, mas vale a pena procurar otimizar-lhe o uso. Vejamos agora uma breve descrição dos principais parâmetros em que intervir:

-m (percentual)

> fixação do percentual de espaço reservado. O default de 10% tende ao despedício. Pode-se descer aos 5% sem muito problema, mas é bom não descer abaixo desse valor porque enchendo muito um sistema de arquivos ffs nós o tornamos mais lento.

-b (dimensão do bloco)

> fixação em bytes da dimensão do bloco. Para os discos de grandes dimensões usar 8192 (8KB), enquanto que para os muito pequenos (<500MB) convém usar 4096 para economizar espaço. Usando blocos ainda maiores não se aceleram as operações e se pode desperdiçar espaço, a menos

que não se memorize predominantemente arquivos de grandes dimensões como os MP3 e os arquivos de vídeo.

-f (tamanho do fragmento)

fixação da dimensão do fragmento. Um bloco pode ser subdividido até a um máximo de 8 fragmentos e, portanto, convém fixar uma dimensão equivalente a 1/8 da dimensão do bloco: 512 bytes para blocos de 4096 bytes e 1024 bytes para blocos de 8192 bytes. O fragmento é a unidade de alocação mínima para os arquivos (como o cluster para o Windows).

-i (bytes de dados por inode)

os *inodes* são estruturas usadas para memorizar informações sobre os arquivos. Para cada arquivo deve haver pelo menos um inode (até mesmo mais de um para permitir ao ffs memorizar os dados do arquivo vizinho do próprio arquivo). A dimensão por default de 4096 não se adapta aos discos de grandes dimensões porque permite que se tenha até 5 milhões de arquivos em um dispositivo de 20MB. Uma avaliação pelo máximo aconselha o uso de `-i 16384` para discos >2GB e de `-i 32768` para discos a partir de 8GB. Usando valores maiores não se economiza mais espaço porque um inode ocupa 128 bytes

Por exemplo, as opções que uso para uma partição >10GB são

```
newfs -m 5 -b 8192 -f 1024 -i 32768 /dev/{nome_da_partição}
```

A leitura da página de manual do **newfs** é enfaticamente aconselhada.

B.1.5. Instalação do carregador da inicialização

Durante o processo de inicialização o kernel (que usualmente se chama `/netbsd`) deve ser carregado na memória pelo disco. A BIOS do PC (e a firmware das outras arquiteturas) não consegue administrar abstrações como os sistemas de arquivos, limitando-se a carregar na memória o primeiro setor do disco e a executar seu código. Este código determina a posição do carregador da inicialização (bootloader) no disco e o carrega. O próprio carregador de inicialização, enfim, pode gerir o sistema de arquivos e carrega o kernel. Uma vez que no setor de inicialização devem ser memorizadas as coordenadas físicas do carregador da inicialização (`/boot`) não nos podemos limitar a copiar o carregador de inicialização /boot na partição designada, mas é necessário executar o seguinte comando:

```
# /usr/mdec/installboot -v /usr/mdec/biosboot.sym /dev/rwd0a
```

No que se refere à descompressão dos tarballs e à configuração final do sistema, não é necessária uma introdução teórica específica e, portanto, podemos passar diretamente à parte prática.

B.2. A instalação na prática

Nos exemplos seguintes suporemos que o disco rígido seja `wd0`, o CD `cd0` (se houver um) e a placa de rede `ep0` (se presente). Se os nomes reais no seu sistema são diferentes, basta substituir os nomes dos exemplos.

B.2.1. A inicialização

Em primeiro lugar é preciso fazer a inicialização a partir de um disquete ou de um CD. Quando aparece o menu do sysinst selecionar “x - exit to shell”. Executar o comando **dmesg | more** para verificar os nomes dos dispositivos de hardware presentes no sistema. Por exemplo, o output seguinte

```
wd0 at pciide0 channel 0 drive 0: <ST320423A>
wd0: drive supports 32-sector pio transfers, lba addressing
wd0: 19536 MB, 16383 cyl, 16 head, 63 sec, 512 bytes/sect x 40011300 sectors
wd0: 32-bit data port
wd0: drive supports PIO mode 4, DMA mode 2, Ultra-DMA mode 4
```

Indica que o disco “ST320423A” (ST usualmente indica um disco Seagate) está conectado como master no primeiro canal do controlador IDE e tem 40011300 setores de 512 bytes, o que perfaz uma dimensão de cerca de 19GB. Freqüentemente a dimensão mostrada é menor (5-8%) que a declarada pelo fabricante, uma vez que os fabricantes consideram que 1GB equivalha a 1.000.000.000 de bytes, no lugar dos 2^30 bytes usados no âmbito da informática. Como 2^30=1073741824, trata-se de uma diferença de 7,4%.

O output:

```
ep0 at pci0 dev 10 function 0: 3Com 3c590 Ethernet
ep0: interrupting at irq 9
ep0: address 00:20:af:f7:cd:71, 32KB byte-wide FIFO, 1:1 Rx:Tx split
ep0: 10baseT, 10baseT-FDX, 10base5, 10base2 (default 10baseT)
```

Indica que a placa ethernet foi identificada como `ep0`.

A maior parte das mensagens não são difíceis de compreender mesmo para os menos entendidos e a sua leitura é muito útil.

B.2.2. Inicialização do disco

Já que queremos obter um sistema apenas com o NetBSD, podemos eliminar todas as partições pré-existentes no disco com o seguinte comando:

```
# dd if=/dev/zero of=/dev/rwd0d bs=1m count=1
```

que preenche o primeiro megabyte do disco com zeros.

`/dev/rwd0d` significa: a partição “raw” d do disco wd0 (no i386 é uma pseudo partição que cobre o disco inteiro). “Raw” indica que não foi efetuado o buffering (utilização de memória intermediária) pelo sistema (que não é necessário).

B.2.3. Criar a partição com disklabel

```
# disklabel -I -i wd0
```

O comando precedente serve para dar início ao particionamento de um disco vazio. Não especificar a opção `-I` se se quer modificar um disklabel existente.

Será mostrado o prompt:

```
partition>
```

O sistema está no aguardo dos comandos. Os comandos mais importantes são **?** (ajuda), **I** (fixa várias informações no disco) e as letras minúsculas de **a** até **h** (fixam as informações para as respectivas partições).

Começemos com o comando **I**. O sistema pergunta o tipo de disco (aceitar o default), o nome do disco (idem) e o label. O label é definido à vontade. Por exemplo, pode-se escrever o próprio nome. Em caso de furto do disco rígido pode-se provar ser o proprietário já que o fdisk do DOS/Windows não cancela esta informação.

Número de partições: escolher 8 (e mesmo mais se houver necessidade). Deixar os valores por default para as outras opções visto que as informações relativas à geometria dos discos modernos são, na realidade, fictícias.

Agora é necessário criar as pertições, começando com as partições “d” e “c” que se referem ao disco inteiro:

```
partition> d
Filesystem type [?] [unused]:
Start offset [0c, 0s, 0M]: 0s
Partition size ('$' for all remaining) [39693.8c, 40011300s, 19536.8M]: $
partition> c
Filesystem type [?] [unused]:
Start offset [0c, 0s, 0M]: 0s
Partition size ('$' for all remaining) [39693.8c, 40011300s, 19536.8M]: $
```

> **Nota:** na versão i386 a partição d refere-se ao disco inteiro e a c à parte NetBSD do disco. No nosso exemplo o NetBSD cobre o disco inteiro e, portanto, a partição d e c coincidem.

Depois passa-se à criação da partição a ser posta no início do disco. No particionamento clássico do Unix trata-se da partição “a”, de cerca de 50MB. Esta dimensão deve ser aumentada oportunamente quando se pretende ter muita correspondência na spool ou ter espaço vazio em `/tmp` ou `/home` sem criar partições separadas para `/var` ou `/home`.

```
partition> a
Filesystem type [?] [unused]: 4.2BSD
Start offset [0c, 0s, 0M]: 0s
Partition size ('$' for all remaining) [0c, 0s, 0M]: 100000s
```

100000 setores correspondem a cerca de 50MB ou pouco menos.

Criemos a segunda partição (swap) de cerca de 585MB:

```
partition> a
Filesystem type [?] [unused]: swap
Start offset [0c, 0s, 0M]: 100000s
Partition size ('$' for all remaining) [0c, 0s, 0M]: 1200000s
```

e enfim a partição“e” para `/usr` até o fim do disco:

```
partition> e
```

```
Filesystem type [?] [unused]: 4.2BSD
Start offset [0c, 0s, 0M]: 1300000s
Partition size ('$' for all remaining) [0c, 0s, 0M]: $
```

Verificar se tudo está no lugar com o comando **P**:

```
partition> P
8 partitions:
#        size   offset     fstype   [fsize bsize cpg/sgs]
  a:   100000        0     4.2BSD        0     0     0   # (Cyl.    0 - 99*)
  b:  1200000   100000       swap                        # (Cyl.   99*- 1289*)
  c: 40011300        0     unused        0     0         # (Cyl.    0 - 39693*)
  d: 40011300        0     unused        0     0         # (Cyl.    0 - 39693*)
  e: 38711300  1300000     4.2BSD        0     0     0   # (Cyl. 1289*- 39693*)
```

> **Nota:** é necessário certificar-se de que as partições não se sobrepõem (exceção feita para c e d). É muito importante e é bom verificar duas vezes.

Enfim executar os comandos:

```
partition> W
Label disk [n]? y
Label written
partition> Q
```

O disklabel foi escrito. Congratulações, a parte mais exigente das operações terminou.

B.2.4. Criação dos sistemas de arquivos e instalação do carregador de inicialização

Para uma pequena partição root usar:

```
# newfs -m 5 -b 4096 -f 512 /dev/rwd0a
```

Para partições maiores:

```
# newfs -m 5 -b 8192 -f 1024 -i 32768 /dev/rwd0<partition name>
```

É importante ter lido a teoria apresentada nas seções precedentes e não se limitar a reproduzir os comandos.

Agora instalemos o carregador da inicialização (bootloader):

```
# /usr/mdec/installboot -v /usr/mdec/biosboot.sym /dev/rwd0a
```

B.2.5. Descomprimir os conjuntos (sets) de instalação

Montar as partições em `/mnt`:

```
# mount -o async,noatime /dev/wd0a /mnt
# mkdir /mnt/usr
# mount -o async,noatime /dev/wd0e /mnt/usr
...
talvez mais se você criou mais [partições]
...
```

As opções `async` e `noatime` não são obrigatórias: tendem a favorecer a velociade das operações em prejuízo da integridade do sistema de arquivos. A proteção frente às quedas do sistema não é fundamental nesse momento, visto que se pode reparar tudo e reinstalar facilmente em caso de queda.

As seções seguintes descrevem alguns métodos comuns de instalação:

CDROM
FTP
NFS

B.2.5.1. Instalando com CDROM

```
# mount_cd9660 /dev/cd0a /mnt2
# cd /mnt2/onde_se encontram_os_tarballs
```

“onde_se_encontram_os_tarballs” é usualmente `i386/binary/sets`. Usar os comandos habituais **cd** e **ls** para verificar se não temos certeza.

Copiar com o comando **cp** os arquivos desejados em `/mnt/usr`. No mínimo é necessário instalar o `base.tgz`, o `etc.tgz` e o `kern.tgz`. Se o disco é grande, copiar tudo (deixando de lado os arquivos `kern*.tgz`, já que basta um só kernel).

Agora executar **cd /** e **umount /mnt2** e, enfim, remover o CD.

B.2.5.2. Instalação a partir de um servidor de FTP

Ativar a rede. Por exemplo, se o endereço IP e a máscara de rede do host são 192.168.3.7 e 255.255.255.0 (24 bits), escrever:

```
# ifconfig ep0 192.168.3.7/24 up
```

Acrescentar "media 10base2" ou "media 10baseT" ou "media 100baseT" pode ser necessário se a sua placa não pode autodetectá-la ou faz isso erradamente. Remeta-se à página de manual do ifconfig para mais informação.

Se o seu servidor de FTP não se encontra em uma rede local (preferível), adicione uma rota por default (default route):

```
# route add default 192.168.3.1
```

execute então o comando **ftp** direcionado ao servidor que contém os arquivos do NetBSD. Nesse momento você *deve* usar IPs e não nomes.

```
# cd /mnt/usr
# ftp 192.168.3.13
```

Entre no servidor (login), vá para o diretório em que estão os arquivos da distribuição, execute o **ls** a pegue todos os arquivos necessários com o comando **get**. Então saia do servidor (logout).

B.2.5.3. Copiando por NFS

Ative sua rede como em 3.e.2. Então

```
# mount IP-do-seu-servidor-nfs:/diretório /mnt2
```

por exemplo:

```
# mount 192.168.3.13:/public/ /mnt2
```

então proceda como na instalação por CD. Por exemplo:

```
# cd /mnt2/NetBSD-1.5.2/i386/binary/sets
# ls
```

e copie todos os arquivos necessários. Então

```
# cd /
# umount /mnt2
```

B.2.5.4. Copiando de uma fita (tape)

Embora as máquinas que funcionam como desktop usualmente não têm drives para tapes, os servidores usualmente os têm, podendo não conter um drive de CDROM apenas para economizar espaço. Instalar usando um tape é muito fácil. Infelizmente os PCs não podem inicializar a partir dos tapes. Então você deve inicialziar com disquetes.

Para preparar a sua fita de instalação você necessita de uma outra máquina com um drive para rolos de fita.

Rebobine a fita:

```
# mt rewind
```

estabeleça tamanho de bloco ilimitado para fitas SCSI

```
# mt setblk 0
```

Isto vai mudar o tamanho de bloco por default muito pequeno para qualquer tamanho. Usaremos blocos grandes para tornar as coisas mais rápidas. Isso é útil para drives DAT/DDS.

!!!ADVERTÊNCIA!!! Por favor corrija-me se isso vier a não funcionar para outros tipos de fita.

Agora desabilite a compressão, já que diferentes drives a serem acessados podem ser incapazes de ler uma unidade de fita comprimida. Comprimir arquivos já comprimidos, de qualquer modo, não vai melhorar em nada ou até mesmo apresentar taxas negativas de compressão (chegando a -15% para os drives HP DDS!):

```
# mt compression 0
```

então grave os arquivos da distribuição:

```
# cd /onde_estão_os_arquivos_da_distribuição
# tar -b64 -cvf /dev/nrst0 *.tgz
```

O último comando vai gravar todos os arquivos tgz no tape com blocos do tamanho 64*512 bytes=32 kB. Este é um tamanho de bloco bem testado para todo DAT.

Como você usa `/dev/nrst0` (primeiro drive de tape SCSI não rebobinante) agora você pode gravar outro arquivo com, por exemplo, alguns pacotes básicos como os editores.

```
# cd /onde_estão_os_pacotes/All
# tar -b64 -cvf /dev/nrst0 joe*.tgz screen*.tgz
```

Isto vai gravar apenas o editor de texto joe e o programa screen. Não ponha muitas coisas neste arquivo.

Agora grave outros pacotes que você vai instalar depois de ter o sistema funcionando em modo multi-usuário.

```
# tar -b64 -cvf /dev/nrst0 todos_os_pacotes_que_você_precisa
```

ou talvez até mesmo

```
# tar -b64 -cvf /dev/nrst0 *.tgz
```

Agora rebobine e descarregue o tape:

```
# mt rewoffl
```

Seu tape de instalação está pronto! Como os drives DDS são compatíveis com regressão, pode ser conveniente comprar barato alguns drives DDS-1 ou DDS-2 usados para a sua máquina doméstica. É uma coisa bem legal - vale $30 para mim :)

B.2.5.4.1. Instalando a partir do rolo de fita

Carregue o tape no drive e faça:

```
# mt rewind
# mt setblk 0
# cd /mnt/usr
# tar -b64 -xvf /dev/nrst0
```

B.2.6. Descomprimindo os sets da distribuição

Não é má idéia copiar agora alguns pequenos pacotes extra, especialmente editores diferentes do vi (como joe, jed) se você não gosta do vi. Copie os pacotes em `/mnt` e não em `/mnt/usr`! É claro que você pode escolher qualquer método para transferir os arquivos: NFS, drives de fita e até mesmo disquetes.

Agora descomprima os tarballs:

```
# cd /mnt
# for x in usr/*.tgz;do tar xzpvf $x;done
# rm usr/*.tgz
```

ou, se você preferir um a um

```
# tar xzpvf /usr/base.tgz
```

Não é má idéia mover /tmp /var e /home para /usr, criando links simbólicos, se você criou uma grande partição `/usr`.

```
# cd /mnt
# tar cvf - var home tmp|(cd usr;tar xpvf -)
# rm -rf var home tmp
# ln -s usr/var .
# ln -s usr/home .
# ln -s usr/tmp .
```

Esta seqüência vai mover var e home para usr, criando ligações simbólicas.

Não descomprima os pacotes extra ainda!

B.2.7. Finalizando sua instalação

Isto vai criar todos os nodes `/dev` (leva algum tempo nos 486 !):

```
# cd /mnt/dev
# ./MAKEDEV all
```

Agora **cd /mnt/etc** e crie o fstab. Você pode usar o vi, mas se você não gosta dele ou não o conhece, simplesmente faça:

```
# cat > fstab
# /dev/wd0a / ffs rw,softdep,noatime 1 1
# /dev/wd0b none swap sw 0 0
# /dev/wd0e /usr ffs rw,softdep,noatime 1 1
# ^D (control-D)
```

Agora verifique se tudo está correto:

```
# cat fstab
```

Se sim, remova os CD/disquetes, digite **reboot** e aguarde até o sistema inicializar em modo mono-usuário!

Ao acessar uma shell digite

```
# /sbin/mount -a
```

para montar todos os sistemas de arquivos em modo de leitura-escrita (read-write). Se você tem alguns pacotes como os editores, por exemplo, instale-os com o **pkg_add**. Depois remova os tarballs com **rm**.

Agora você tem que editar o `/etc/rc.conf`,o `/etc/defaults/rc.conf` e outros arquivos. Leia o documento INSTALL padrão sobre como e o quê fazer. Geralmente definindo

```
rc_configured=YES
```

no `/etc/rc.conf` e definindo o hostname no `/etc/defaults/rc.conf` é suficiente para fazer o sistema funcionar.

Digite ^D (Control-D) para permitir a inicialização multi-usuário. Seu sistema foi instalado com sucesso.

Apêndice C. Como contribuir com o guia do NetBSD

Ultimamente pode-se notar um significativo aumento do interesse pela documentação, introdutória ou avançada, do NetBSD. Trata-se sem dúvida de um sinal da crescente popularidade do sistema e do aumento da massa de usuários. Portanto, é mais que importante continuar a atualizar e a acrescentar o material seja deste guia, seja dos vários HOWTO presentes no site do NetBSD.

Independentemente do próprio nível de conhecimento e de experiência com o NetBSD, qualquer um pode contribuir com o desenvolvimento deste guia. Este apêndice explica de que modo se pode contribuir com o guia e o que é necessário saber antes de começar.

Os principiantes e, mais em geral, aqueles que instalam o sistema pela primeira vez, podem enviar seus próprios comentários e sugestões diretamente para mim indicando, por exemplo, se alguma coisa não funcionou, se um tópico não está explicado de modo suficientemente claro, etc. Ou se pode enviar uma solicitação para que seja tratado de um determinado assunto. Esse tipo de intercâmbio é muito útil.

Os usuários mais avançados podem escrever uma nova seção ou até mesmo um capítulo inteiro para acrescentar ao guia. Ou podem modificar uma parte já existente acrescentando exemplos, figuras ou reformulando-a de maneira mais clara.

Quem tem um pouco de tempo livre pode levar em conta a idéia de traduzir o guia para uma nova língua.

O que quer que se escolha fazer, antes de tudo contate-me. Isso pode evitar trabalho duplicado e inútil.

C.1. Traduzindo o guia

Quem quiser traduzir o guia deve antes de mais nada pôr-se em contato comigo ou escrever à mailing list netbsd-doc@netbsd.org.

Se outras pessoas já estiverem trabalhando em uma tradução do guia na mesma língua, é possível juntar-se ao grupo, abreviando os prazos de término do trabalho. Se ninguém estiver traduzindo no momento, é sempre possível que alguns capítulos já tenham sido traduzidos no passado. Do contrário será necessário iniciar a tradução a partir do zero. Qualquer que seja a situação que se apresente, não é necessário traduzir o guia inteiro. Mesmo a tradução de um só capítulo será uma contribuição útil e apreciada.

Mesmo quando uma tradução já foi terminada, é necessário manter o guia sempre atualizado, com material novo e com correções no material existente. E é possível, portanto, tornar-se o *mantainer* de uma versão do guia.

C.1.1. O que é preciso para iniciar uma tradução

Para iniciar uma tradução do guia basta ter as fontes do mesmo. Ponha-se em contato comigo e lhe enviarei a versão mais recente.

Em maiores detalhes, o que se precisa é:

- fontes do guia (em formato SGML/DocBook)
- um editor de texto como o vi ou o emacs.

O trabalho de tradução pode ser desenvolvido em qualquer sistema operacional.

> **Nota:** não comece a trabalhar em HTML ou outros formatos: será muito trabalhoso converter o trabalho para o formato SGML/DocBook.

C.1.2. Como escrever em formato SGML/DocBook

Para traduzir o guia não é necessário *aprender* a escrever em SGML/DocBook. É suficiente trabalhar com um editor em cima das fontes, substituindo o texto original e deixando invariados os *tags*. Ou seja, reutilizando a estrutura do documento. A título de exemplo, para traduzir a nota precedente poder-se-ia operar do seguinte modo:

1. abrir as fontes do capítulo corrente `ap-contrib.sgml` no editor.
2. encontrar o texto correspondente à nota precedente, que tem o seguinte aspecto:

   ```
   <note>
     <para>
       non iniziate a lavorare in HTML o altri formati:
       sar&agrave; molto oneroso convertire il lavoro in
       formato SGML/DocBook.
     </para>
   </note>
   ```

3. acrescentar a tradução no interior dos tags, depois da versão italiana (ou antes, como preferir). Agora o texto terá este aspecto:

   ```
   <note>
     <para>
       non iniziate a lavorare in HTML o altri formati:
       sar&agrave; molto oneroso convertire il lavoro in
       formato SGML/DocBook.
       sua tradução vai aqui
       sua tradução vai aqui
       sua tradução vai aqui
     </para>
   </note>
   ```

4. eliminar as três linhas do texto original italiano deixando a tradução.

   ```
   <note>
     <para>
       sua tradução vai aqui
       sua tradução vai aqui
       sua tradução vai aqui
     </para>
   </note>
   ```

Quando se escreve a tradução é preferível utilizar a mesma indentação e a mesma formatação da versão original. Ver Secção C.3 para um exemplo.

Um problema que se deve quase sempre enfrentar quando se escreve em SGML/DocBook é o dos “caracteres nacionais” (por exemplo, as letras acentuadas como “è”). Ainda que possamos utilizar esse tipo de caracter em um documento, é preferível substituí-lo com a correspondente *entidade* SGML. Por exemplo, o caracter “é” escreve-se “è”. É evidente que isso torna mais difícil a escrita e, sobretudo, a leitura do texto mas, pelo menos o problema da escrita pode ser minorado com o uso de um editor que suporte macros. O vi e o emacs, dois editores dentre os mais conhecidos, oferecem a possibilidade de se associar macros às teclas, gerando automaticamente as entidades requeridas. Por exemplo, para se obter automaticamente a letra "e" acentuada precedente, pode-se inserir o seguinte comando no arquivo `.exrc`:

```
map! è &egrave;
```

Apêndice D explica como instalar as ferramentas de software necessárias para gerar o formato HTML e outros formatos a partir das fontes DocBook. O formato HTML é o mais simples de gerar e é útil para se verificar a correção das fontes (isto é, que não haja qualquer erro nos tags), assim como para uma leitura mais ágil do texto. Naturalmente, como já dito, a geração do formato HTML não é absolutamente necessária para se efetuar a tradução. O que conta são as fontes traduzidas.

C.2. Enviando material para o guia

Quem quiser contribuir com o guia enviando material, tem diversas possibilidades, de acordo com a quantidade de texto que quer escrever. Quando se trata apenas de uma breve correção ou adendo, a coisa mais simples é enviar um e-mail para mim com o texto. Providenciarei a inserção no guia.

Se o desejo for escrever uma certa quantidade de texto, como uma seção ou um capítulo inteiro, pode-se escolher entre vários formatos:

- SGML/DocBook. Este é o formato preferido. Quando se escolhe esse formato é preciso que se obtenha as fontes do guia e usá-las como esquema para a indentação e para o formato do texto e dos tags.
- texto. Se a formatação do texto é simples, a conversão ao formato SGML não é muito trabalhosa.
- HTML. É preferível escrevê-lo a mão (isto é, com um editor de texto normal) no lugar de utilizar um gerador de HTML para que a conversão à SGML fique mais simples. O uso do HTML em comparação com o texto puro não traz vantagens particulares, exceto o de tornar mais evidente o formato que se deseja dar ao conteúdo. Um outro benefício colateral é que as entidades HTML para os caracteres acentuados, etc., já estarão provavelmente presentes no texto.
- são aceitos também outros formatos, mas apenas nos casos em que não seja possível utilizar nenhum dos precedentes.

C.3. Esquema da disposição SGML/DocBook

Para as fontes do guia é uso um estilo de formatação parecido com o de um programa. Eis um esquema:

```
<chapter id="chap-xxxxx">
  <title>Este é o título de um capítulo</title>

  <para>
    Este é o texto. Este é o texto.
    Este é o texto. Este é o texto.
    Este é o texto. Este é o texto.
  </para>

  <!-- ============================================================ -->

  <sect1>
    <title>Título de uma sect1</title>

    <para>
      Este é o texto. Este é o texto.
      Este é o texto. Este é o texto. Este é o texto.
      Este é o texto.
    </para>

    <!-- .......................................................... -->

    <sect2>
      <title>Título de uma sect2</title>

      <para>
 Uma sect2 é aninhada dentro de uma sect1
      </para>
    </sect2>

  </sect1>

  <!-- ============================================================ -->

  <sect1>
    <title>Título de uma sect1</title>

    <para>
      Esta é uma lista:
      <itemizedlist>
 <listitem>
   <para>
     texto
   </para>
 </listitem>
 <listitem>
   <para>
     texto
   </para>
```

```
 </listitem>
      </itemizedlist>
    </para>

  </sect1>
</chapter>
```

Os defaults são:

- dois espaços para cada nível de indentação
- linhas não mais longas que 72 caracteres
- usar linhas de separação (comentários) entre sect1/sect2

Apêndice D. Introdução à SGML/DocBook

Contrariamente ao que se poderia pensar lendo o título, este capítulo descreve a instalação dos instrumentos necessários para a produção formatada do presente Guia ao NetBSD. Ainda que a SGML/DocBook e a DSSSL não sejam descritas aqui, ao fim do capítulo estão presentes alguns links em que pode ser encontrado o material informativo para aqueles que desejam saber mais.

O ambiente SGML/DocBook pode ser instalado com a utilização do "meta-pacote" *netbsd-docs*. Este é o método mais simples e é o que aconselho usar. O presente capítulo, entretanto, descreve a instalação e a configuração de cada um dos componentes necessários, a fim de que os examinemos com maior detalhamento e, eventualmente, de também verificar sua correta instalação.

> **Nota:** O "meta-pacote" *netbsd-docs* instala alguns pacotes não estritamente necessários para a produção do guia. Tais pacotes, que não são descritos neste documento, são:
>
> - iso12083-1993
> - unproven-pthreads-0.17nb2
> - opensp-1.4
> - html-4.0b

Os pacotes necessários para o guia serão instalados com a utilização de pacotes pré-compilados. Para quem preferir, é possível compilá-los a partir do código-fonte. Para mais detalhes veja-se o Capítulo 8.

> **Nota:** Os números das versões dos pacotes instalados são, naturalmente, sujeitos a mudança com a saída de novas versões.

D.1. O que é SGML/DocBook?

SGML (Standard Generalized Markup Language) é uma linguagem utilizada para definir a estrutura de outras linguagens baseadas em "markup". Isso significa que com a SGML pode-se definir a *gramática* das linguagens de markup. HTML, por exemplo, pode ser definida usando-se SGML. Os programadores podem pensar em SGML como o equivalente da BNF (Backus-Naur Form) para as linguagens de programação: uma notação para definir gramáticas.

DocBook é um esquema de markup definido para o uso da SGML. O DocBook especifica quais tags podem ser utilizados em um documento e de que modo podem ser combinados entre si. Os programadores podem pensar em DocBook como em uma linguagem de programação, com as suas palavras-chave válidas e as suas regras. Por exemplo, para o DocBook os tags

```
<para> ... </para>
```

definem o início e o fim de um parágrafo. Além disso o DocBook especifica que o tag <para> pode estar no interior de uma seção <sect1> mas que o tag <sect1> não pode estar no interior de um <para>.

À luz do que acaba de ser dito pode-se compreender que os documentos produzidos com estes instrumentos são documentos DocBook e não SGML. Desse ponto de vista, DocBook é a contraparte da HTML (ainda que os tags sejam mais numerosos e os conceitos à base das duas linguagens, diversos).

A especificação DocBook (isto é, a lista dos tags válidos e as regras para combiná-los é chamada DTD (Document Type Definition).

Uma DTD definirá qual será o aspecto de um documento-*fonte* válido mas não dá nenhuma indicação sobre o formato do documento final gerado a partir das fontes. A título de exemplo, o tag <title> marca o início de um título mas não especifica que aparência terá o título no documento final (tamanho e corpo dos caracteres, alinhamento, espaçamento, etc.) Para chegar à “apresentação” do documento é necessário um passo a mais, efetuado com a utilização do programa Jade, que aplica ao documento-fonte as transformações DSSSL. DSSSL (Document Style Semantics and Specification Language) é o formato utilizado para definir as *stylesheets* (folhas de estilo) necessárias para converter o documento DocBook em outros formatos.

A “vida” de um documento DocBook é portanto a seguinte:

- redação do documento fonte;
- a DTD do DocBook é utilizado pelo nsgmls para “validar” o documento;
- Jade aplica as folhas de estilo DSSSL ao documento fonte e gera um novo documento.

  ainda não é possível imprimir o documento gerado porque ele contém as diretrizes de formatação indicadas nas folhas de estilo (pode tratar-se de um documento HTML, RTF, TeX, etc.);

- um programa de formatação (um navegador para HTML, um processador de textos para RTF, TeX, ...) cria a versão final, visualizável ou imprimível do documento.

Em definitivo, o que precisa para iniciar o trabalho é:

- uma DTD para DocBook
- as folhas de estilo DSSSL usadas pelo Jade para gerar HTML/RTF, etc.
- o programa Jade e o parser (analisador gramatical) nsgmls

D.2. Jade

Jade é um parser SGML/XML que implementa o mecanismo da DSSSL. O pacote Jade inclui também o parser nsgmls, que valida os documentos.

> **Nota:** OpenJade é uma evolução do Jade que, no momento, não se compila sob NetBSD (mas que não é necessário para o guia).

Para instalar o pacote pré-compilado do Jade:

```
# pkg_add jade-1.2.1.tgz
```

Ao fim da instalação, a documentação do Jade encontra-se em `/usr/pkg/share/doc/jade/index.htm`. O diretório principal da instalação é `/usr/pkg/share/sgml/jade/`: é aí que se encontra o arquivo `catalog` do Jade.

D.3. DocBook

Depois de ter instalado o Jade pode-se proceder à instalação da DocBook DTD.

Este pacote exige a instalação das "entities" para o conjunto de caracteres ISO 8879:1986. Adicionemos pois o pacote correspondente:

```
# pkg_add iso8879-1986.tgz
```

As entities são instaladas no diretório `/usr/pkg/share/sgml/iso8879/` e o catálogo correspondente é: `/usr/pkg/share/sgml/iso8879/catalog`.

Finalmente pode-se instlar a DTD.

```
# pkg_add docbook-4.1.tgz
```

Apesar do nome este pacote instala várias versões de DTD (2.4.1, 3.0, 3.1, 4.0, 4.1). Isto é, permite a elaboração de documentos que usam diferentes versões de DocBook.

> **Nota:** o guia do NetBSD usa no momento a versão 4.1 do DTD do DocBook e, portanto, as outras versões não são estritamente necessárias. Todavia, em consideração ao fato de que cada DTD ocupa menos de 250 kb, pode ser conveniente mantê-las no disco rígido, já que permitem a elaboração de outros documentos.

A raiz da instalação do sistema é `/usr/pkg/share/sgml/docbook/4.1/`. Cada versão de DTD encontra-se em um diretório separado e possui um arquivo catálogo próprio. Por exemplo: `/usr/pkg/share/sgml/docbook/4.1/catalog`.

D.4. As folhas de estilo DSSSL

O último passo consiste na instalação das folhas de estilo DSSSL.

```
# pkg_add dsssl-docbook-modular-1.57.tgz
```

Também as folhas de estilo instalam um catálogo próprio que se encontra em `/usr/pkg/share/sgml/docbook/dsssl/modular/catalog`. Ademais são munidas de detalhada documentação acessada em `/usr/pkg/share/sgml/docbook/dsssl/modular/doc/index.html`.

D.5. Usando as ferramentas

Agora que foi instalado todo o necessário, pode-se começar a utilizar as ferramentas para produzir a versão HTML do guia (por exemplo, a versão italiana).

Com um **cd** entra-se no diretório raiz do guia e se executa o comando **make**:

```
$ cd it
$ make netbsd.html
```

Uma longa relação de erros escorrerá sobre a tela. Isso é devido ao fato de que o parser (analisador gramatical) SGML nsgmls não consegue encontrar os arquivos `catalog`. Portanto, é necessário digitar os seguintes comandos (que convém inserir no próprio `~/.profile`):

```
SGML_ROOT=/usr/pkg/share/sgml
SGML_CATALOG_FILES=${SGML_ROOT}/jade/catalog
SGML_CATALOG_FILES=${SGML_ROOT}/iso8879/catalog:$SGML_CATALOG_FILES
SGML_CATALOG_FILES=${SGML_ROOT}/docbook/3.0/catalog:$SGML_CATALOG_FILES
SGML_CATALOG_FILES=${SGML_ROOT}/docbook/3.1/catalog:$SGML_CATALOG_FILES
SGML_CATALOG_FILES=${SGML_ROOT}/docbook/4.0/catalog:$SGML_CATALOG_FILES
SGML_CATALOG_FILES=${SGML_ROOT}/docbook/4.1/catalog:$SGML_CATALOG_FILES
SGML_CATALOG_FILES=${SGML_ROOT}/docbook/dsssl/modular/catalog:$SGML_CATALOG_FILES
export SGML_CATALOG_FILES
```

> **Nota:** modificar oportunamente os comandos se for utilizada uma shell C (csh) ou uma shell derivada desta.

Agora que a variável do ambiente SGML_CATALOG_FILES está ativa, pode-se tentar de novo o comando precedente:

```
 $ make netbsd.html
nsgmls -sv netbsd.sgml
nsgmls:I: SP version "1.3.3"
jade -d ../dsl/myhtml.dsl -t sgml -o netbsd.html netbsd.sgml
```

Desta vez a operação chega a bom termo e será gerada a versão HTML do guia. A criação do formato RTF não apresenta maiores dificuldades:

```
 $ make netbsd.rtf
nsgmls -sv netbsd.sgml
nsgmls:I: SP version "1.3.3"
jade -d ../dsl/myrtf.dsl -t rtf -o netbsd.rtf netbsd.sgml
```

Com a instalação descrita acima podem ser gerados apenas os formatos HTML e RTF. Para se criar os formatos PS e PDF é necessário instalar e configurar o TeX e o Jadetex (um pacote de macros para o TeX).

D.6. Uma abordagem alternativa aos arquivos de catálogo

Nas minhas instalações geralmente crio um arquivo de catálogo *master*, que contém uma referência a todos os outros. Para usar esse método pode-se criar o arquivo `/usr/pkg/share/sgml/catalog` com o seguinte conteúdo:

```
CATALOG "/usr/pkg/share/sgml/docbook/3.0/catalog"
CATALOG "/usr/pkg/share/sgml/docbook/3.1/catalog"
CATALOG "/usr/pkg/share/sgml/docbook/4.0/catalog"
CATALOG "/usr/pkg/share/sgml/docbook/4.1/catalog"
CATALOG "/usr/pkg/share/sgml/docbook/dsssl/modular/catalog"
CATALOG "/usr/pkg/share/sgml/iso8879/catalog"
CATALOG "/usr/pkg/share/sgml/jade/catalog"
```

Feito isso, as inicializações no ~/.profile podem ser simplificadas assim:

```
SGML_CATALOG_FILES=/usr/pkg/share/sgml/catalog
export SGML_CATALOG_FILES
```

D.7. Gerando a versão Postscript

Para criar a versão Postscript do guia são necessárias algumas operações preliminares:

- instalação do TeX
- habilitar a silabação italiana do TeX
- criação do formato hugelatex exigido pelo Jadetex
- instalação do Jadetex

As seções seguintes descrevem em detalhe os pontos precedentes.

D.7.1. Instalação do TeX

A instalação do TeX foi bastante simplificada, como de costume, pelo sistema de gerenciamento pacotes. Utilizando-se os pacotes pré-compilados é suficiente executar os seguintes comandos (os números das versões dos pacotes poderiam ser diferentes):

```
# pkg_add teTeX-share-1.0.2.tgz
# pkg_add teTeX-bin-1.0.7nb1.tgz
```

D.7.2. Habilitando a silabação italiana do TeX

O guia do NetBSD é correntemente disponível em quatro línguas diferentes: francês, inglês, italiano e português brasileiro. Destas, apenas inglês e francês têm silabação automática feita pelo TeX. Para habilitar a silabação para a língua italiana (e também para o português. Nota do tradutor.), são necessárias algumas operações simples:

Editar a arquivo `/usr/pkg/share/texmf/tex/generic/config/language.dat` removendo o caracter indicador de comentário (%) ao início da linha relativa à língua italiana. Isto é:

```
%italian ithyph.tex
```

torna-se

```
italian ithyph.tex
```

> **Nota:** quando estiverem disponíveis ulteriores traduções do guia, será provavelmente necessário ativar a silabação também para as novas línguas.

Agora é necessário recriar os formatos latex e pdflatex:

```
# cd /usr/pkg/share/texmf/web2c
# fmtutil --byfmt latex
# fmtutil --byfmt pdflatex
```

Para se fazer uma verificação pode-se conferir, por exemplo, o arquivo `latex.log` que deveria conter, entre outras informações, uma linha do seguinte tipo:

```
Babel <v3.6Z> and hyphenation patterns for american, french, german,
ngerman, italian, nohyphenation, loaded.
```

> **Nota:** há vários modos de se executar estas operações, de acordo com o próprio nível de conhecimento do TeX (o meu é razoavelmente baixo). Por exemplo, é possível usar o programa interativo texconfig, ou recriar os formatos usando diretamente **tex** (fmtutil, de fato, é simplesmente um script que invoca o tex).
>
> Se você conhece um método melhor para efetuar as operações descritas neste capítulo, peço-lhe mo aponte.

D.7.3. Criação do formato hugelatex

Para poder utilizar o jadetex é necessário criar o formato hugelatex, que não é senão um formato latex com os limites de capacidade aumentados para poder gerir os grandes arquivos criados pelo jadetex. Antes de efetuar as modificações convém fazer uma cópia de segurança (back up) do arquivo `/usr/pkg/share/texmf/web2c/texmf.cnf`. Depois, acrescentar ao fim do arquivo as linhas seguintes (precisaremos das definições para o jadetex e para o pdfjadetex quando instalarmos o Jadetex mais tarde):

```
% hugelatex settings
main_memory.hugelatex = 1100000
param_size.hugelatex = 1500
stack_size.hugelatex = 1500
hash_extra.hugelatex = 15000
string_vacancies.hugelatex = 45000
pool_free.hugelatex = 47500
nest_size.hugelatex = 500
save_size.hugelatex 5000
pool_size.hugelatex = 500000
max_strings.hugelatex 55000
font_mem_size.hugelatex = 400000
```

```
% jadetex & pdfjadetex
main_memory.jadetex = 1500000
param_size.jadetex = 1500
stack_size.jadetex = 1500
hash_extra.jadetex = 15000
string_vacancies.jadetex = 45000
pool_free.jadetex = 47500
nest_size.jadetex = 500
save_size.jadetex 5000
pool_size.jadetex = 500000
max_strings.jadetex 55000

main_memory.pdfjadetex = 2500000
param_size.pdfjadetex = 1500
stack_size.pdfjadetex = 1500
hash_extra.pdfjadetex = 50000
string_vacancies.pdfjadetex = 45000
pool_free.pdfjadetex = 47500
nest_size.pdfjadetex = 500
save_size.pdfjadetex 5000
pool_size.pdfjadetex = 500000
max_strings.pdfjadetex 55000
```

Acrescentar as seguintes linhas, que descrevem o formato hugelatex, ao arquivo `fmtutil.cnf` (no diretório `/usr/pkg/share/texmf/web2c`)

```
# hugelatex format created for jadetex
hugelatex tex  language.dat latex.ini
```

salvar o arquivo e executar o comando

```
fmtutil --byfmt hugelatex.
```

Agora o formato hugelatex está pronto para o uso.

> **Nota:** como já se assinalou, com o TeX há muitos modos de se atingir o mesmo resultado (neste caso, a criação do formato hugelatex).
>
> O guia para a instalação do Jadetex nos dá as seguintes instruções:
>
> ```
> # cp -R /usr/pkg/share/texmf/tex/latex/config /tmp
> # cd /tmp/config
> # tex -ini -progname=hugelatex latex.ini
> # mv latex.fmt hugelatex.fmt
> # mv hugelatex.fmt /usr/pkg/share/texmf/web2c
> # ln -s /usr/pkg/bin/tex /usr/pkg/bin/hugelatex
> ```

D.7.4. Instalação do Jadetex

> **Nota:** a versão mais recente do Jadetex pode ser obtida em http://www.tug.org/applications/jadetex/

Fazer o download do pacote Jadetex (a versão corrente é `jadetex-3.6.zip`), descompactá-lo e depois executar os seguintes comandos:

```
# cd jadetex
# make install
# mktexlsr
```

A instalação copia os arquivos de formatação do jadetex e do pdfjadetex (assim como alguns arquivos de apoio) na árvore dos diretórios do TeX.

A distribuição do Jadetex contém também duas páginas de manual que não são instaladas automaticamente. Se as quisermos é suficiente copiá-las manualmente no diretório adequado. Por exemplo:

```
# cp jadetex.1 pdfjadetex.1 /usr/local/man/man1
```

Agora está tudo pronto para se criar a versão Postscript do guia do NetBSD (e naturalmente de qualquer outro documento). Por exemplo, deslocar-se para o diretório base do guia e dar os seguintes comandos:

```
$ cd it
# make netbsd.ps
```

D.8. Links úteis

Uma simples e bem escrita introdução à SGML/DocBook, assim como uma descrição das ferramentas necessárias encontram-se em SGML comme format de fichier universel (http://casteyde.christian.free.fr/SGML/index.html).

The official DocBook home page (http://www.oasis-open.org/docbook/) é a fonte oficial de informações sobre o DocBook. Entre outras informações pode-se fazer consultas on line ou o download do livro DocBook: The Definitive Guide (http://www.oasis-open.org/docbook/documentation/reference/) de Norman Walsh e Leonard Muellner, um guia fundamental para quem quer aprofundar-se no tema.

Para informações sobre as DSSSL convém começar com nwalsh.com (http://nwalsh.com).

As fontes do Jade/OpenJade e documentação correspondente podem-se encontrar em OpenJade Home Page (http://openjade.sourceforge.net/).

Para quem deseja produzir documentos em formato Postscript ou PDF a partir das fontes DocBook é bom dar uma olhada na home page do JadeTex (http://www.tug.org/applications/jadetex).

A home page de Markus Hoenicka (http://ourworld.compuserve.com/homepages/hoenicka_markus/) explica tudo o que é necessário saber para trabalhar com SGML/DocBook no Windows NT.

Apêndice E. Agradecimentos

Várias pessoas contribuiram para a confecção desse guia acrescentando material, revendo os conteúdos, dando sugestões, corrigindo erros e até ajudando a configurar e a instalar os programas utilizados para a sua elaboração.

Desejo agradecer particularmente:

- Paulo Aukar
- Manolo De Santis
- Eric Delcamp
- Hubert Feyrer, que escreveu a introdução ao manejo de redes de computadores com TCP/IP da Secção 9.1 incluída a parte relativa ao protocolo de última geração IPv6 e a seção sobre conectividade IPv6, à Secção 10.2.4.
- Jason R. Fink
- Reinoud Koornstra (Mipam)
- David Magda
- Guillain Seuillot
- Wojciech Puchar
- Oliver Tonnhofer

Se por acaso esqueci-me de acrescentar o seu nome à lista acima, peço que me comunique.