Il sistema operativo NetBSD

una breve guida all’uso

Federico Lupi

Il sistema operativo NetBSD: una breve guida all’uso

Federico Lupi

Sommario

Lista delle Figure

Lista degli Esempi

Scopo di questa guida

Questa guida descrive l'installazione e la configurazione del sistema operativo NetBSD e si rivolge principalmente a persone provenienti da altri sistemi operativi nella speranza di essere di aiuto per la risoluzione di alcuni di quei piccoli problemi che si incontrano quando si è alle prese con un sistema con il quale si ha poca familiarità. Le istruzioni contenute nel seguito sono state verificate con la versione per i386 di NetBSD.

Nel seguito vengono date per scontate alcune conoscenze minime da parte del lettore: cosa sono un file, una directory, un programma, come si fa a editare un file e così via. Nelle librerie si possono trovare molti testi che trattano questi argomenti (in particolare per i sistemi Unix): questi libri hanno solitamente un carattere generale e sono un prerequisito essenziale per l'uso di un sistema come NetBSD. In particolare, chi installa un sistema BSD si dovrà incontrare (scontrare?), almeno nelle prime fasi, con l'editor vi: senza un po' di documentazione questo potrebbe essere un ostacolo insormontabile. In un secondo tempo, naturalmente, ognuno potrà installare sul sistema l'editor (e i programmi) che preferisce.

La guida non è ancora completa: alcuni argomenti non sono terminati o vanno rivisti e altri mancano del tutto. Cercherò di portarla avanti compatibilmente con la mia disponibilità di tempo. Se volete essere avvisati quando viene aggiornata o se siete interessati a argomenti specifici inviatemi un e-mail. Contattatemi anche se siete interessati a collaborare alla guida o alla sua traduzione in inglese o in altre lingue.

`flupi@mclink.it`

Capitolo 1. Licenza

Copyright (c) 1999, 2000, 2001 Federico Lupi. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:
   This product includes software developed by Federico Lupi for
   the NetBSD Project.
4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Capitolo 2. Che cos’é NetBSD

NetBSD è un sistema operativo UNIX-like altamente portabile, *free*, disponibile per molte piattaforme, dai server alpha a 64 bit fino ai dispositivi palmari. Il suo *design* pulito e le sue caratteristiche avanzate lo rendono idoneo a ambiti sia produttivi sia di ricerca. È completamente disponibile in forma sorgente e ha una ricca dotazione di applicazioni.

2.1. Le origini di NetBSD

La prima versione di NetBSD (0.8) risale al 1993 e deriva dal sistema operativo 4.3BSD Lite, una versione di Unix sviluppata all’Università della California di Berkeley (BSD = Berkeley Software Distribution) e dal sistema 386BSD, il primo *port* per sistemi Intel 386. In seguito sono state integrate le modifiche provenienti dal sistema 4.4BSD Lite, l’ultima release ufficiale del gruppo di sviluppatori di Berkeley prima dello scioglimento. Il ramo BSD di Unix ha avuto una grande importanza nella storia di questo sistema operativo, al quale ha contribuito con svariate innovazioni che oggi fanno parte di tutti i sitemi Unix (l’editor vi, la C shell, il job control, il Berkeley fast file system, l’integrazione del TCP/IP, tanto per menzionarne alcune). Questa tradizione di sviluppo e di ricerca sopravvive oggi nei sistemi BSD (sia freeware sia commerciali) e, in particolare, in NetBSD.

2.2. Caratteristiche di NetBSD

NetBSD funziona su una vasta gamma di piattaforme hardware ed è molto portabile. Con NetBSD viene fornito il codice sorgente dell’intero sistema operativo per tutte le piattaforme supportate. Senza dilungarmi in eccessivi dettagli, per i quali rimando senz’altro al sito ufficiale del progetto NetBSD (http://www.netbsd.org/), le caratteristiche fondamentali di questo sistema operativo sono le seguenti

- Estrema portabilità (oltre 20 piattaforme supportate)
- Qualità e correttezza del codice
- Aderenza agli standard
- Ricerca e innovazione

Le caratteristiche appena menzionate portano dei vantaggi anche indiretti. Per esempio, chi lavora su una piattaforma sola potrebbe non essere interessato alla portabilità; in realtà, però, la portabilità è strettamente legata alla qualità del codice: non sarebbe possibile supportare tutte queste piattaforme se il codice non fosse ben scritto e ben organizzato. L’attenzione verso l’aspetto architetturale e qualitativo del sistema viene ripagata con le grandi potenzialità del suo codice e la qualità dei suoi driver e quindi interessa tutti gli utenti.

Una delle caratteristiche del sistema è quella di non accontentarsi di implementazioni parziali: “se deve essere fatto, deve essere fatto bene”; il panorama informatico già tristemente abbonda di esempi di programmi e sistemi operativi iperviluppati e pieni di errori che collassano sotto il proprio stesso peso.

Un elenco dettagliato di caratteristiche di NetBSD si può trovare all’indirizzo http://www.netbsd.org/Misc/features.html.

2.3. Piattaforme supportate

La release 1.4.1 supporta le seguenti piattaforme

- 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 and 400
- i386 family IBM PCs and clones
- Apple Macintosh
- Apple Power Macintosh
- Motorola MVME 68k SBCs
- NeXT 68k ’black’ hardware
- The PC532
- Digital MIPS-based DECstations and DECsystems
- Sun SPARC
- Sun 3 and Sun3x
- Digital VAX
- Sharp X680x0

Sul sito di NetBSD si possono trovare i dettagli tecnici per tutte le piattaforme precedenti.

2.4. A chi si rivolge NetBSD

Secondo quanto compare sul sito di NetBSD, i suoi destinatari sono i professionisti, gli appassionati e i ricercatori che vogliono un sistema stabile che privilegi la qualità. Ma anche chi vuole imparare a usare Unix troverà in NetBSD la piattaforma ideale, soprattutto per la sua aderenza agli standard (uno degli scopi del progetto) e, infine, chi ha bisogno di una piattaforma Unix disponibile su una grande varietà di macchine, non può trovare migliore alleato di NetBSD.

Un’altra caratteristica interessante è che con NetBSD si possono riutilizzare sistemi hardware considerati obsoleti per la maggior parte dei sistemi operativi, il che rende NetBSD un’ottima piattaforma per apprendere Unix. Come dire “non c’è bisogno di comprare nuovo hardware per avere la vostra versione di Unix in funzione: potete tranquillamente riutilizzare il vecchio MacIIcx che avete in soffitta”.

2.5. Applicazioni per NetBSD

Chi installa NetBSD non si trova certo a corto di applicazioni tra cui scegliere. A parte il corredo di applicazioni standard di produttività personale e di sviluppo (C/C++) di ogni sistema Unix, è disponibile un gran numero di pacchetti esterni (oltre un migliaio e il numero è in costante aumento), installabili attraverso il comodo sistema dei package. In generale, tutti i principali pacchetti per Unix sono disponibili per NetBSD. Inoltre NetBSD è in grado di emulare Linux e di eseguire molti programmi Linux nativi (Netscape, Acrobat Reader, ... e persino Doom e Quake!). L'emulazione Linux è molto curata e viene considerata molto importante: quasi tutti gli utenti di NetBSD la usano per alcuni programmi.

Oltre a eseguire programmi Linux, NetBSD è in grado di emulare anche altri sistemi, in particolare FreeBSD, BSDI e altri ancora.

2.6. La filosofia della distribuzione NetBSD

A differenza di molti sistemi odierni, la distribuzione NetBSD è tutt'altro che mastodontica poiché si basa sul concetto di produrre un sistema di base stabile, completo ed efficiente ma privo di ridondanze. Al termine dell'installazione ci si trova con un sistema funzionante ma ancora privo di una serie programmi di utilità (per esempio, un browser per Internet): tutte queste applicazioni non fanno parte del nucleo di base; deve essere l'utente a decidere se vuole Netscape o Lynx, quale release di Perl gli serve e così via. Un altro vantaggio è che il sistema di base funziona in modo autosufficiente senza aver bisogno di questi pacchetti "aggiuntivi". Per esempio, il sistema dei package non richiede Perl: se si decide di installare una nuova versione di Perl non si rischia di "rompere" nessun componente standard del sistema. Quando si installa NetBSD, dunque, non si trovano selezioni preconfigurate di centinaia di pacchetti che vanno a ingombrare il disco fisso e sono destinati per la maggior parte a rimanere inutilizzati. L'utente BSD deve sempre essere consapevole di cosa installa e perché: il processo di installazione dei programmi, tra l'altro, è reso semplicissimo dal sistema dei package, di cui parleremo più avanti.

Anche quando si comincia a conoscere un po' il sistema e a orizzontarsi bene al suo interno, una delle cose che continua a stupire è l'estrema coerenza e logicità dell'insieme; niente è casuale o gratuito e tutto appare pensato con grande attenzione ai dettagli. A pensarci bene ritengo che, tutto sommato, questa sia la caratteristica più interessante di NetBSD; si possono passare giorni a discutere sui relativi meriti dei vari sistemi operativi (e qualcuno trova anche il tempo per farlo) ma solo quando si prova e si arriva a conoscere un sistema si può fare una scelta motivata. Sono convinto, per averlo osservato tante volte nelle mailing list, che chi prova NetBSD non può che rimanere conquistato dal perfetto bilanciamento tra complessità ed efficacia realizzato da questo sistema: NetBSD riesce sempre a realizzare le cose più complesse nella maniera più semplice. È un'attitudine mentale contagiosa: dopo un po' riesce difficile tollerare le inutili complicazioni (solitamente travestite da "miglioramenti" o da "comode interfacce grafiche") di altri sistemi.

2.7. Come procurarsi NetBSD

A tutt'oggi NetBSD non ha un rivenditore ufficiale, anche se esistono vari punti di distribuzione. Per un elenco aggiornato conviene consultare la relativa pagina (http://www.netbsd.org/Sites/cdroms.html) sul sito del progetto. Naturalmente è sempre possibile scaricare il sistema da Internet da uno dei vari mirror.

Capitolo 3. Installazione

3.1. Documentazione

La documentazione di NetBSD è per la maggior parte nel formato delle pagine di manuale (man) e comprende soprattutto documentazione tecnica di riferimento, molto valida e completa ma non particolarmente adatta a scopo introduttivo. A parte ciò è impossibile leggere una pagina man prima di aver installato il sistema ...

> **Nota:** dopo l'installazione si possono anche trovare varie guide di Unix/BSD nella directory `/usr/share/doc`; alcune hanno carattere storico, mentre altre vengono tuttora aggiornate. Sono divise in tre gruppi, *psd* (UNIX Programmer's Supplementary Documents), *smm* (UNIX System Manager's Manual) e *usd* (UNIX User's Supplementary Documents). Purtroppo alcune directory sono vuote, probabilmente per motivi di licenza dei contenuti. Il testo può essere letto sullo schermo con nroff. Per esempio:
>
> ```
> $ cd /usr/share/doc/smm/09.sendmail
> $ nroff -me 09.sendmail/intro.me | more
> ```
>
> Usando i makefile si può anche generare output in formato Postscript, pronto per essere stampato.

È tuttavia innegabile una certa carenza di guide utente e dei cosiddetti HOWTO e, pertanto, occorre sfruttare al massimo quelle esistenti. Una serie di documenti si trova nella release di NetBSD, in formato testo, mentre ulteriori informazioni si possono trovare sul sito web di NetBSD (www.netbsd.org (http://www.netbsd.org) e mirror), in particolare le cosiddette Frequently Asked Questions (FAQ).

> **Documentazione originale:** il sito di NetBSD contiene varie pagine di documentazione e di HOWTO, sia generici sia specifici per piattaforma. Si tratta di informazioni utili ed esposte in maniera semplice (per esempio come vedere una partizione Windows da NetBSD oppure come avviare NetBSD dal boot loader di Windows NT, ecc.). In questa guida per ora non vengono duplicate le informazioni già presenti sul sito di NetBSD.

Più o meno in tutte le versioni di NetBSD si trovano i seguenti file:

INSTALL

Note per l'installazione di NetBSD/i386. Da leggere (e rileggere) assolutamente: contiene una descrizione del sistema, l'elenco dei dispositivi hardware supportati e le istruzioni per l'installazione.

README.first

Non proprio essenziale per l'installazione, ma conviene leggerlo.

release.man

Descrive la struttura della release di NetBSD in formato man (preformattato: si può leggere con un normale editor di testo).

Sul sito web di NetBSD si trovano varie guide online, molto utili e molto sintetiche

NetBSD FAQ

Contiene alcune informazioni di carattere generale (per esempio come ricompilare il kernel) e i puntatori alle altre FAQ.

NetBSD/i386 FAQ

Argomenti specifici di NetBSD/i386.

Basic NetBSD Networking

Guida alla configurazione di rete e all'uso di PPP.

3.2. La struttura di una distribuzione NetBSD

Il *layout* dei file in una distribuzione NetBSD è descritto nel file `INSTALL`. Giusto per promemoria, i file binari da installare si trovano (per la piattaforma *i386*) nella subdirectory `i386/binary/sets`. I relativi sorgenti si trovano nella directory `source/sets`. Nella directory `source/patches` vengono messi i *patch* ai sorgenti per correggere eventuali problemi (solitamente relativi a problemi di sicurezza scoperti dopo l'uscita della release).

3.3. Installazione

La prima cosa da fare per installare NetBSD è leggere le note di installazione della release desiderata, che si trovano nel file INSTALL. La procedura di installazione è descritta in dettaglio. A questo punto si può decidere il mezzo da utilizzare per l'installazione tra quelli supportati.

ftp
nfs
CDROM
floppy
filesystem
directory locale

3.3.1. Tastiera

sysinst, il programma di installazione, non permette di cambiare il layout della tastiera e, pertanto, l'installazione deve essere effettuata utilizzando la disposizione della tastiera americana. In realtà non si tratta di un grosso problema, dal momento che per installare NetBSD da CDROM bisogna semplicemente scegliere tra una serie di opzioni di tipo alfanumerico, e quindi i tasti corrispondono. Se si installa via ftp, il problema può essere più noioso. Penso che le prossime versioni del programma di installazione consentiranno di modificare il layout della tastiera ma, per ora, ci si può orizzontare con la seguente tabella:

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

Al termine dell’installazione, una delle prime cose da fare sarà cambiare la disposizione della tastiera.

3.3.2. Geometrie

In alcuni punti del programma di installazione si fa riferimento alla geometria del disco fisso, e in particolare a

geometria reale
geometria del BIOS

La geometria reale è quella rilevata dal sistema e si riferisce alla geometria effettiva del disco. La geometria del BIOS è la geometria con la quale il BIOS vede il disco e potrebbe essere diversa da quella reale (per esempio, potrebbe essere rimappata LBA). Il disco fisso che verrà usato nell’esempio di installazione ha la seguente geometria

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

In questo caso il BIOS usa la modalità LBA per rimappare la geometria del disco diminuendo il numero di cilindri e aumentando quello di tracce (ma il prodotto resta uguale: 6232*16 = 779*128 = 99712). Ricordando che un settore è composto da 512 byte, la dimensione del disco è: 6232*16*63*512 = 3 GB. NetBSD non ha alcun bisogno di rimappare la geometria del disco, ed è anche possibile cambiare la geometria manualmente se quella rilevata dal sistema non fosse corretta.

3.3.3. Partizioni e avviamento

In questa sezione cercheremo di chiarire che cosa sono le partizioni (dal punto di vista del BIOS e dal punto di vista di NetBSD) spiegando nel contempo come si avvia un PC e come NetBSD arriva a prendere il controllo.

Quando un PC viene avviato, il controllo viene preso dal BIOS che effettua una serie di verifiche (dette POST, Power On Self Test, per es. il test della RAM) e determina qual'è il dispositivo di avviamento da utilizzare (solitamente configurabile nelle impostazioni del BIOS); in questo esempio supporremo che si tratti del primo disco fisso IDE (che NetBSD identifica con wd0). Il BIOS carica in memoria il "loader" (programma di caricamento) il cui codice si trova nel Master Boot Record (MBR) del disco fisso e gli cede il controllo (ossia salta alla prima istruzione del programma).

Il Master Boot Record si trova al cilindro 0, testina 0 e settore 1 del disco fisso e contiene il programma di caricamento già citato e la tabella delle partizioni BIOS che indica in che modo il disco fisso è stato partizionato.

Partizionare un disco fisso significa suddividerne la superficie fisica in unità logiche (le partizioni BIOS); tale suddivisione viene memorizzata nella già citata tabella delle partizioni all'interno del MBR. Partizionando un disco si hanno vari benefici, per esempio si possono installare più sistemi operativi, ognuno nella sua partizione. Assegnando più partizioni a uno stesso sistema operativo si diminuiscono i rischi di perdita di dati in caso di malfunzionamenti.

Supponendo di aver partizionato il disco per poterci installare più sistemi operativi, come fa il loader a decidere quale sistema avviare? La risposta è semplice: nella tabella delle partizioni è presente un flag in corrispondenza della partizione attiva; il loader cerca la partizione marcata e le cede il controllo, ossia carica un secondo programma di avviamento che si trova all'inizio (cioè nel primo settore) della partizione stessa (NetBSD chiama questo programma "bootblock" ed è quello che carica in memoria ed esegue il kernel).

Secondo questo schema di avviamento, sembrerebbe che sia possibile installare più sistemi operativi, ognuno nella sua partizione, ma che all'accensione del PC parta sempre lo stesso (quello della partizione attiva); in effetti, con un MBR standard è così. È a questo punto che interviene il cosiddetto "selettore di boot" (boot selector), un programmino che sostituisce il loader standard del MBR e che consente di *scegliere* a quale partizione cedere il controllo e quindi quale sistema operativo avviare.

NetBSD dispone di un proprio boot selector che il programma di installazione può scrivere nel MBR su richiesta dell'utente. A questo punto dovrebbe essere chiaro che il boot selector è necessario soltanto se si è in presenza di più sistemi operativi...

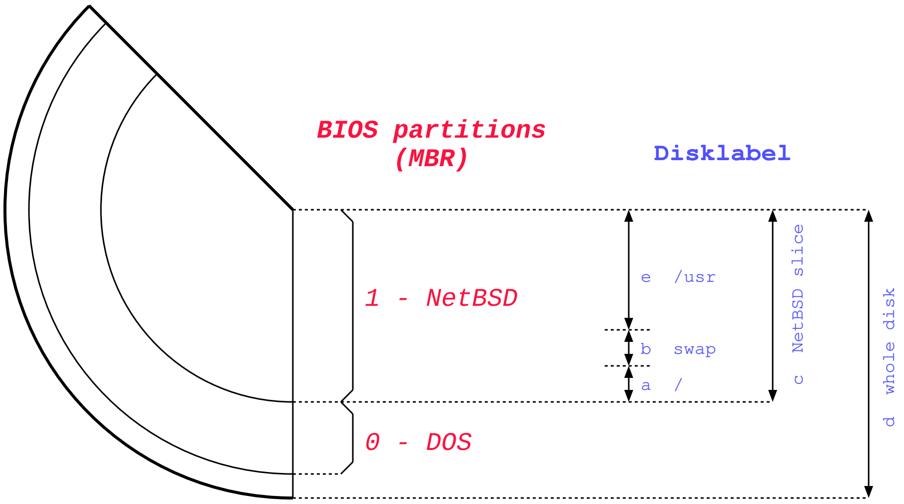
NetBSD si installa in una delle quattro partizioni primarie del BIOS.

Fin qui abbiamo parlato del BIOS e delle sue partizioni; NetBSD è in grado di leggere le partizioni BIOS del Master Boot Record ma non le utilizza per il proprio funzionamento. Invece, utilizza una propria tabella delle partizioni, chiamata "disklabel", che si trova all'inizio della partizione BIOS riservata a NetBSD (che deve essere una partizione primaria). Si noti che si tratta di una struttura propria di NetBSD, della quale il BIOS è del tutto ignaro.

Le partizioni NetBSDBSD (dette anche "slice") sono visibili solo da NetBSD e sono identificate mediante lettere (da "a" in avanti). Per esempio wd0a indica la partizione 'a' del primo disco IDE (wd0). Nell'esempio di Figura 3-1 vi sono due partizioni BIOS primarie; la prima è occupata dal DOS e la seconda da NetBSD, che la suddivide a sua volta in partizioni BSD scrivendo una disklabel.

> **Nota:** in realtà non è del tutto esatto dire che la disklabel suddivide la partizione BIOS di NetBSD in sotto-partizioni. Come si può vedere dalla figura le partizioni "c" e "d" non sono sotto-partizioni. La disklabel è, a tutti gli effetti, la tabella delle partizioni personale di NetBSD: il kernel di NetBSD non consulta la tabella BIOS ma la disklabel.

Figura 3-1. Partizioni



Nota: la disposizione descritta nella Figura 3-1 è tipica dei sistemi i386 per quanto riguarda il significato delle partizioni “c” e “d”.

Nota: come già spiegato, se NetBSD non è l’unico sistema operativo presente sul disco fisso (come nell’esempio precedente) è necessario installare un *boot manager*, cioè un programma che in fase di boot del computer consente di scegliere il sistema operativo da avviare. Il programma di installazione di NetBSD può installare automaticamente un boot manager se l’utente lo richiede.

Se l’altro sistema operativo sul disco fisso è Windows NT, è possibile utilizzare il boot manager di NT, aggiungendo NetBSD al menu; la procedura (molto semplice) è descritta sul sito di NetBSD.

3.3.4. Spazio necessario

Lo spazio da dedicare a NetBSD dipende chiaramente dall’uso che se ne vuole fare (per esempio server o workstation), dalla versione di NetBSD e dai programmi che si vogliono installare. A titolo di esempio si consideri un sistema usato come desktop con un disco fisso da 420 MB sul quale sono stati installati X, i sorgenti del kernel e alcuni programmi. Alla partizione di swap sono stati dedicati 32 MB. **df** mostra quanto segue:

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

Come si vede sul sistema sono rimasti circa 180 MB liberi.

3.3.5. Ritentare

Le prime installazioni, in generale, non vanno quasi mai bene; questo vale per tutti i sistemi operativi e NetBSD non fa eccezione. L'importante è non scoraggiarsi e riprovare: ciò che all'inizio sembra incomprensibile diventa presto chiaro in virtù delle prove fatte e dell'esperienza acquisita (nonché delle numerose riletture del file INSTALL).

Durante le prime installazioni conviene accettare tutti i valori di default che vengono proposti dal programma di installazione, evitando di avventurarsi in modifiche alla disklabel o simili. L'unica reale decisione da prendere è lo spazio da dedicare alla partizione NetBSD con fdisk.

3.4. Esempio di installazione

Nella restante parte di questo capitolo vediamo in dettaglio un esempio reale di installazione per uno dei casi più comuni: l'installazione da CD-ROM. Gli altri tipi di installazione non differiscono nei concetti ma solo nel modo in cui vengono reperiti gli archivi da installare. Si noti che alcuni dettagli dell'installazione possono presentare delle differenze a seconda della versione di NetBSD installata: l'esempio si riferisce alla versione 1.5.

Nell'esempio seguente vengono sempre prese le strade più “difficili”:

- Tabella delle partizioni senza spazio disponibile: bisognerà eliminare una o più partizioni per fare spazio a NetBSD.
- Partizionamento usando i settori anziché i megabyte come unità di misura.
- Modifica manuale della disklabel creata dal programma di installazione (anche qui usando i settori come unità di misura).
- Installazione di tipo "custom".

Queste scelte fanno apparire l'installazione molto più laboriosa di quanto non sia nella realtà ma, d'altronde, non avrebbe molto senso scrivere una guida che illustri solo le situazioni più facili. Si tenga comunque presente che scegliendo di utilizzare i default dell'installazione le operazioni sono *molto* più semplici.

3.4.1. Preparare l'installazione

Prima di procedere all'installazione è bene pianificare accuratamente le operazioni da compiere. Per prima cosa bisogna leggere attentamente il già citato INSTALL, che descrive i passi necessari per l'installazione, controllando in particolare l'elenco delle compatibilità hardware. Poi è necessario liberare uno spazio sufficiente a accogliere NetBSD sul disco fisso. Se NetBSD deve condividere il disco con altri sistemi operativi, bisognerà probabilmente creare una nuova partizione (a questo ci pensa il programma di installazione), eventualmente riducendone una esistente (a questo ci deve pensare l'utente). Per questo scopo si può usare un prodotto commerciale (per esempio Partition Magic) oppure un prodotto freeware come FIPS o pfdisk (presente nella distribuzione di NetBSD).

L'installazione è logicamente divisa in due parti. Nella prima parte si crea una partizione per NetBSD e si crea la disklabel per la partizione. Nella seconda parte si decide che cosa si vuole installare e si estraggono i file nella partizione appena creata. La prima parte è indipendente dal metodo di installazione (CD-ROM, ftp, NFS, ecc.). Al termine della prima parte viene richiesto di confermare le operazioni effettuate; fino a quel momento non è stato scritto ancora *nulla* sul disco fisso: annullando l'installazione tutto rimane come era prima di incominciare e il disco fisso non viene modificato in alcun modo.

3.4.2. Creare il floppy di installazione

> **Nota:** se si dispone di un CD-ROM avviabile di NetBSD non è necessario creare il dischetto di installazione; basterà riavviare il computer dopo averlo predisposto per fare il boot dal lettore di CD (questa opzione non è disponibile su tutti i PC). Chi fa il boot da CD può passare direttamente alla sezione seguente.

La prima operazione da compiere è la creazione del floppy di installazione. Si tratta di copiare l'immagine del dischetto da CD a floppy. Se questa operazione viene effettuata in ambiente DOS si userà il programma rawrite.exe che si trova nella directory `i386/installation/misc`. Il file da copiare su dischetto è `i386/installation/floppy/boot.fs`.

> Accertarsi sempre che i floppy usati per i dischi di installazione non presentino difetti. Questo semplice accorgimento può evitare parecchi fastidi in fase di installazione.

1. Formattare il floppy in ambiente DOS.
2. Portarsi nella directory \NETBSD-1.3.2\I386\INSTALLATION\FLOPPY del CD-ROM.
3. Eseguire **..\MISC\RAWRITE** indicando `BOOT.FS` come *Source file name* e A: come *Destination drive*.

Se la creazione del dischetto viene effettuata in ambiente Unix, si userà il comando **dd**; per esempio:

```
# cd i386/installation/floppy
# dd if=boot.fs of=/dev/fd0a bs=36b
```

dd copia blocchi di 512 byte: l'opzione `bs=36b` indica di copiare 36 blocchi per volta, per velocizzare l'operazione.

> **Nota:** un dischetto da 1440K contiene 1474560 byte ed è composto da 80 cilindri, 2 tracce, 18 settori da 512 byte, cioè da 80*2*18=2880 blocchi da 512 byte. Quindi, con l'opzione `bs=36b` si copia un cilindro per volta.

3.4.3. Ultime operazioni preparatorie

Ora tutto è pronto per l'installazione vera e propria ma, prima di cominciare, è bene acquisire alcune informazioni sull'hardware del PC.

La cosa più importante da scoprire è il tipo di disco fisso montato sul sistema (IDE, SCSI, ...) e la sua geometria. Queste informazioni si possono ricavare dal manuale del disco, da uno dei tanti programmi diagnostici per DOS oppure dal BIOS stesso all'avviamento del computer. Su alcuni dischi la geometria è stampata su un'etichetta attaccata al disco. Un'altra valida possibilità è di collegarsi al sito Internet del produttore e cercare le specifiche tecniche del disco fisso. Se si intende installare via ftp o via NFS è bene annotarsi anche le impostazioni della scheda di rete: una delle cause di problemi più frequente è che l'IRQ della scheda di rete non corrisponde a quello impostato nel kernel. Vediamo, a titolo di esempio, le impostazioni per una categoria di schede di rete a loro tempo molto diffuse, le NE2000 compatibili sul bus ISA. Il kernel è preconfigurato per individuare schede con queste due impostazioni:

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

Pertanto, se le impostazioni di una NE2000 installata sul sistema non coincidono con *ne0* o con *ne1* il kernel di installazione non la rileverà (dopo l'installazione è naturalmente possibile compilare un kernel personalizzato in grado di riconoscere anche altre configurazioni).

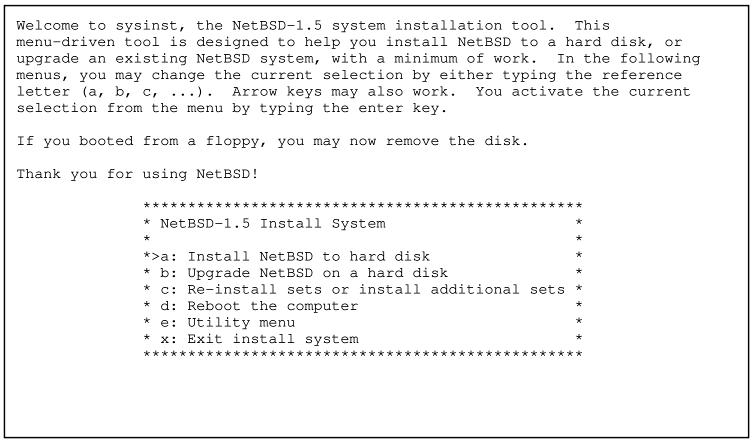
Già che ci siamo, conviene scoprire anche qualche informazione sul resto del sistema, per esempio il tipo di scheda audio, il numero di porte seriali e parallele, ecc. Tutte queste informazioni possono essere utili in un secondo tempo, anche se solitamente non servono in fase di installazione. È molto importante individuare eventuali impostazioni non standard delle periferiche; per far ciò, confrontare le impostazioni del proprio sistema (IRQ, porte di I/O, ecc.) con quelle di default riportate nel file INSTALL.

> **Nota:** si può procedere all'installazione anche senza sapere la geometria del disco. L'informazione sulla geometria serve unicamente per controllo: il programma di installazione è in grado di determinarla automaticamente.

3.4.4. Iniziamo l'installazione

Inserire il floppy di installazione appena creato nel lettore A: e riavviare il computer (oppure fare il boot da CD-ROM). Dal dischetto viene fatto il boot del kernel compresso `netbsd.gz`. Durante il boot del dischetto di installazione appena creato (per certe versioni di NetBSD i dischetti potrebbero essere due) all'avvio del kernel vengono visualizzate le informazioni sui dispositivi trovati e configurati. È probabile che vi siano molte segnalazioni di dispositivi non trovati: ciò è dovuto al fatto che il kernel sul dischetto comprende il supporto per una grande quantità di periferiche; è naturale che non tutte siano presenti sul sistema.

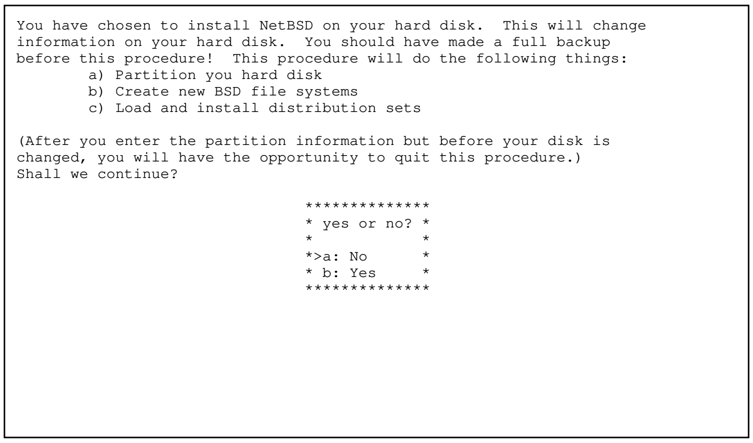
Figura 3-2. Inizio dell'installazione



```
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                       *
              ************************************************
```

Terminata la procedura di avvio ci si ritrova nel menu principale del programma di installazione, mostrato nella Figura 3-2. Non bisogna lasciarsi ingannare dall'aspetto spartano di sysinst: si tratta di un programma molto potente e flessibile. Da qui in avanti si seguono le istruzioni mostrate a video, tenendo presenti le spiegazioni del file INSTALL. Le schermate di sysinst sono tutte strutturate nello stesso modo: nella parte superiore viene visualizzata una breve spiegazione, nella parte centrale viene mostrata la situazione attuale riscontrata da NetBSD (non presente in questa schermata) e nella parte inferiore compare il menu con le scelte possibili. Scegliendo l'opzione di installazione ("a") si arriva alla schermata di conferma, mostrata nella Figura 3-3.

Figura 3-3. Conferma dell'installazione

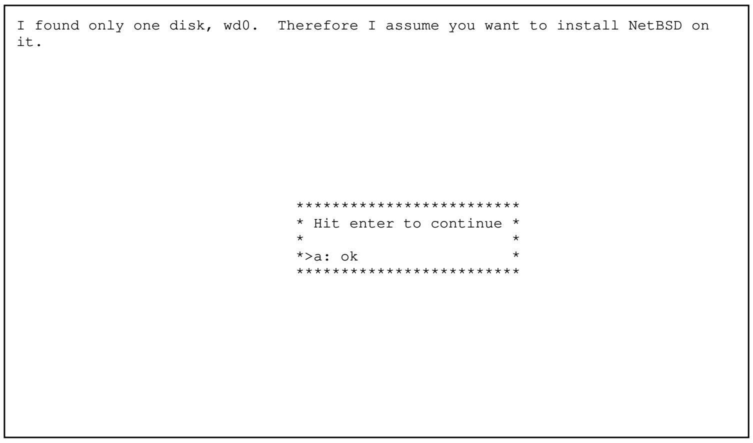


```
You have chosen to install NetBSD on your hard disk.  This will change
information on your hard disk.  You should have made a full backup
before this procedure!  This procedure will do the following things:
        a) Partition you hard disk
        b) Create new BSD file systems
        c) Load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

Dopo aver scelto di proseguire con l'opzione "b", viene il momento di indicare su quale disco fisso si vuole installare NetBSD. Se sul sistema sono presenti più dischi fissi viene mostrato un elenco da cui scegliere; in questo caso sul sistema c'è un solo disco fisso IDE e quindi il programma di installazione non propone scelte ma si limita a visualizzare una schermata informativa, mostrata in Figura 3-4.

> **Nota:** le informazioni mostrate in questa schermata cambieranno a seconda del tipo e del numero di dischi fissi installati sul sistema.

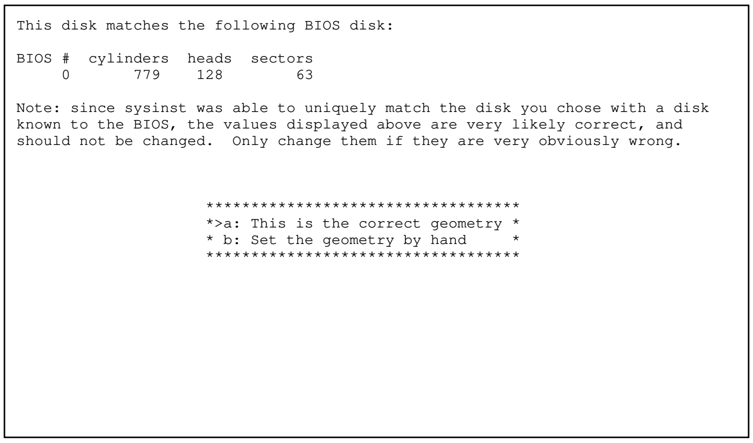
Figura 3-4. Scelta del disco fisso



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










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

Successivamente (Figura 3-5) viene visualizzata la geometria vista dal BIOS per il disco selezionato nella schermata precedente. Si può confermare la geometria mostrata oppure modificarla a mano. Questa seconda opzione serve nel caso in cui il sistema non abbia riconosciuto la geometria correttamente.

Figura 3-5. Conferma della 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     *
                     ***********************************
```

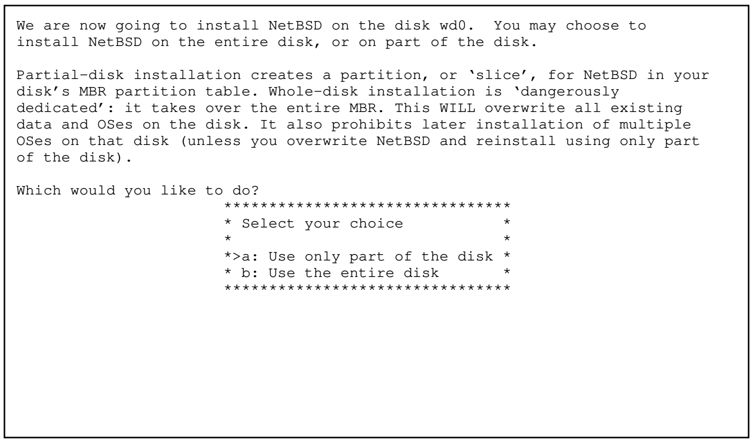
3.4.5. Partizioni

È arrivato il primo momento importante dell'installazione: il partizionamento del disco fisso. La prima cosa da fare è indicare se NetBSD occuperà una partizione (scelta consigliata) o l'intero disco (scelta temeraria). Anche scegliendo la prima opzione, comunque, si può creare una partizione che copre l'intero disco (Figura 3-6): la differenza è che nella prima modalità di installazione la tabella delle partizioni viene mantenuta in una modalità compatibile con altri sistemi operativi.

In questo esempio useremo un disco con la seguente geometria reale, corrispondente alla geometria BIOS mostrata nella Figura 3-5.

```
6232 cyl, 16 heads, 63 sec  (6232 x 16 x 63 = 6281856 settori totali)
1 settore = 512 byte
1 traccia = 63 settori = 63 * 512 byte = 32 K
1 cilindro = 16 * 63 * 512 byte = 504 K
```

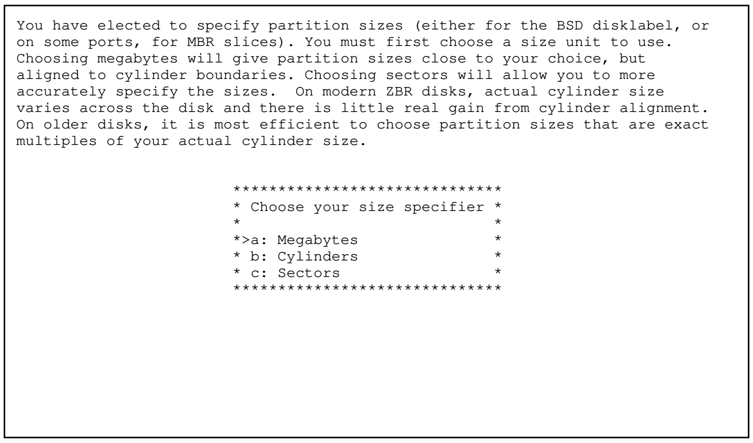
Figura 3-6. Scegliere lo schema di partizionamento



```
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       *
                       ********************************
```

Il passo seguente, illustrato nella Figura 3-7 è l’indicazione dell’unità di misura da utilizzare per partizionare il disco: i settori sono quelli che consentono la maggiore precisione, ricordando che sui dischi più vecchi conviene allineare le partizioni a multipli della dimensione del cilindro. I megabyte sono più semplici da utilizzare perché non richiedono calcoli manuali e sono più “intuitivi”.

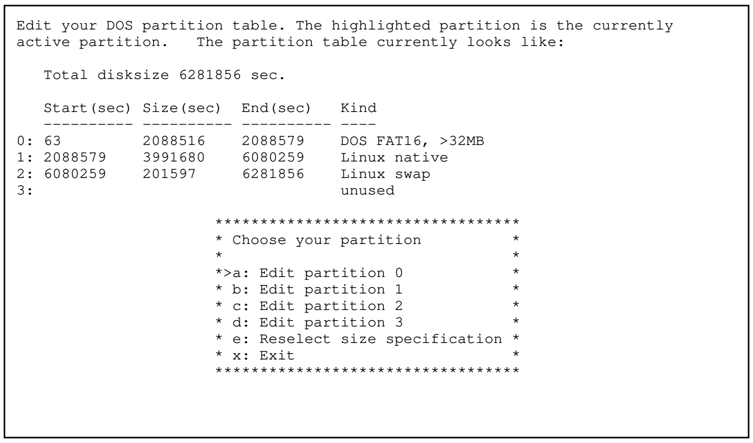
Figura 3-7. Scelta dell’unità di misura



```
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                 *
                        ******************************
```

Nel seguito utilizzeremo i settori perché sono più utili a fine dimostrativo. Scegliendo l’opzione “c” si arriva alla schermata di interfaccia di fdisk.

Figura 3-8. fdisk



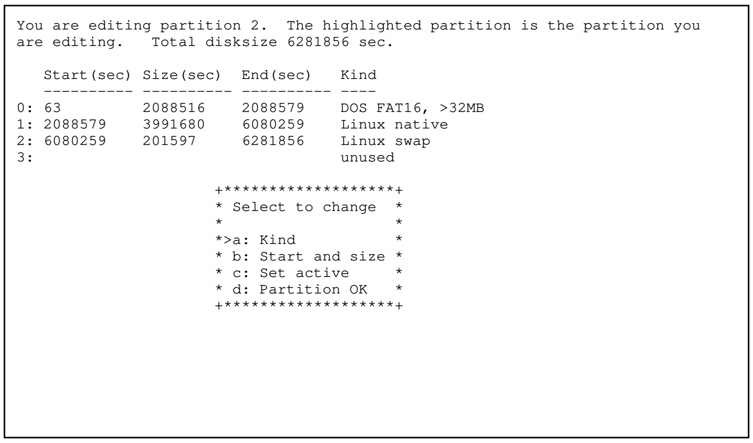
```
Edit your DOS partition table. The highlighted partition is the currently
active partition.   The partition table currently looks like:

   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                        *
                      **********************************
```

La Figura 3-8 mostra la situazione del disco fisso prima dell'installazione di NetBSD: vi sono quattro partizioni primarie, una per DOS/Windows, due per Linux e una non utilizzata. Anche se c'è una partizione libera non c'è spazio libero sul disco: la colonna End(sec) della partizione 2 mostra che sono occupati tutti i 6281856 settori totali del disco.

> **Nota:** nella schermata di fdisk vale sempre la formula
>
> ```
> Start(sec) + Size(sec) = End(sec)
> ```
>
> inoltre la colonna End(sec) di una partizione corrisponde alla colonna Start(sec) della partizione seguente. Questo non è molto intuitivo perché il settore visualizzato nella colonna End(sec) in realtà non appartiene alla partizione bensì è il primo settore della partizione seguente. Nella disklabel viene invece utilizzata una convenzione diversa (e più logica).

Per fare spazio bisogna sacrificare (a malincuore) le due partizioni di Debian GNU/Linux, incominciando dall'ultima (opzione "c"). Viene visualizzata la schermata di modifica dei dati di una partizione di Figura 3-9 che mostra i dati correnti della partizione 2.

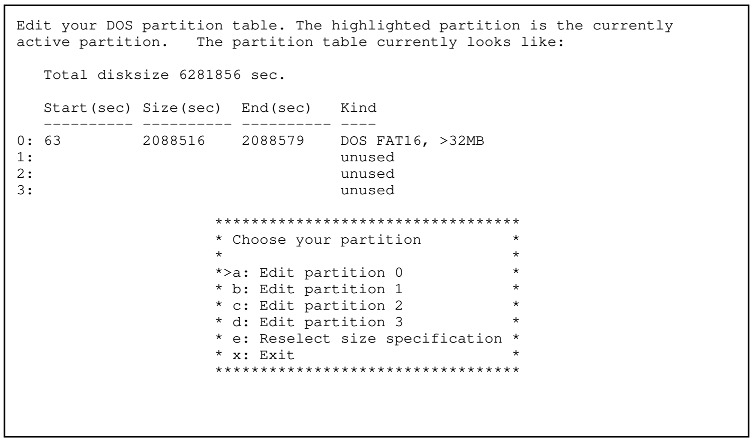
Figura 3-9. Cancellazione di una partizione



```
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   *
                       +******************+
```

Per cancellare la partizione si seleziona dapprima il tipo *unused* con l'opzione "a: Kind" e poi si sceglie l'ozpione "b" e si lasciano vuoti i campi "Start" e "Size" (premendo invio senza inserire nulla). Infine i conferma il tutto con l'opzione "d". Quando si ritorna alla schermata principale di fdisk, la partizione risulta libera. Allo stesso modo si cancellano le partizioni 2 e 1, fino a lasciare solo la partizione 0 (Figura 3-10).

Figura 3-10. Partizione cancellata



```
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                        *
                       **********************************
```

Rimane solo la partizione DOS/Windows di 2088516 settori, cioè 1029 MB (circa 1 GB). Lo spazio libero è dato dalla differenza tra il numero totale di settori, già calcolato sopra, e il punto finale della partizione DOS (colonna "End")

```
6281856 - 2088579 = 4193277 settori = 2047 MB liberi sul disco
```

> **Nota:** si noti che la partizione DOS inizia al settore 63 e non al settore 0 come ci si potrebbe aspettare. Infatti la prima traccia (cioè i primi 63 settori) è riservata. Al cilindro 0, traccia 0, settore 1 del disco fisso si trova il *Master Boot Record* (MBR). All'avvio del computer il BIOS carica in memoria il MBR dal disco fisso, determina qual'è la partizione attiva e carica in memoria il boot

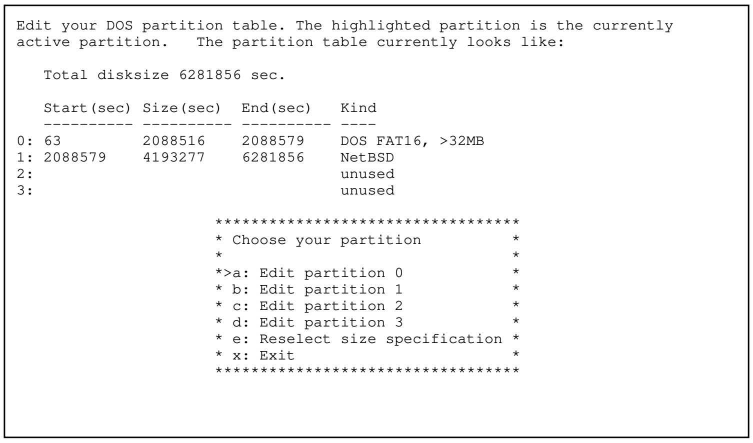
sector di quella partizione, al quale cede il controllo. Il boot sector della partizione provvede a avviare il relativo sistema operativo.

Ora si procede a creare una nuova partizione per NetBSD, che deve iniziare alla fine della partizione DOS, scegliendo l'opzione “b”. Per creare una nuova partizione occorre specificare

- il tipo della nuova partizione
- il settore iniziale della nuova partizione
- la dimensione in settori della nuova partizione

Per prima cosa si indica che la nuova partizione deve essere di tipo “NetBSD” (opzione "a: Kind") e poi si inseriscono i dati appena calcolati: inizio = 2088579 e dimensione = 4193277 (utilizzando l'opzione “b”). Alla fine si controlla che tutto sia corretto e si conferma la creazione con l'opzione “d” che riporta al menu principale di fdisk. Il risultato finale è visualizzato nella Figura 3-11 che mostra la tabella delle partizioni completata. A questo punto si può selezionare l'opzione “x” per passare al menu seguente.

Figura 3-11. Partizione creata



```
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                        *
                      **********************************
```

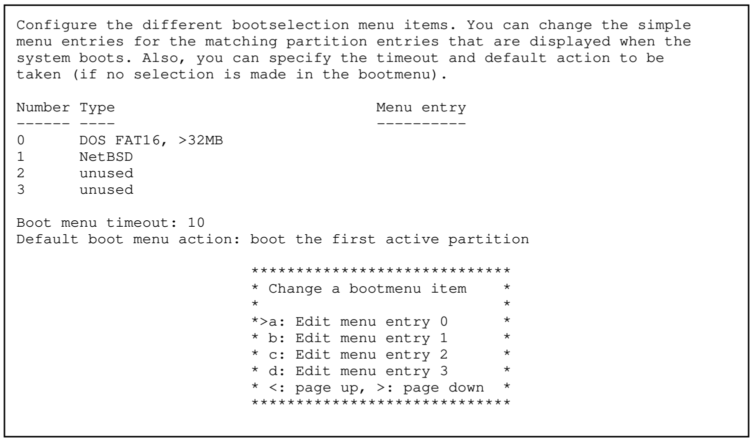
Nota: la versione di sysinst di NetBSD 1.5 verifica anche l'inizio e la fine delle partizioni non utilizzate (quelle marcate *unused*, anche se questi dati non vengono mostrati a video. Pertanto potrebbe accadere che il programma indichi che vi sono delle sovrapposizioni nelle partizioni mentre a video sembra tutto corretto. Consiglio pertanto di sistemare anche i dati di inizio e fine delle partizioni non utilizzate.

La schermata che viene visualizzata adesso dipende da vari fattori. Se è stato commesso qualche errore nel partizionamento (per esempio è stata creata una partizione che si sovrappone a un'altra già esistente) sysinst visualizzerà un messaggio e proporrà di ritornare alla schermata di partizionamento. Se i dati inseriti sono corretti, a seconda della versione di NetBSD e dei dati inseriti ora potrebbe comparire una schermata che avvisa che la partizione NetBSD si trova in una posizione nella quale il BIOS potrebbe non riuscire ad avviarla e chiede se si vuole proseguire lo stesso. Questo fatto potrebbe essere un problema sui PC meno recenti; il PC utilizzato per l'esempio si avvia perfettamente. Non è possibile dare una regola generale (dipende dal BIOS); per il momento, se compare questo avviso consiglio di ignorarlo e proseguire.

> **Nota:** questa non è una limitazione di NetBSD bensì di alcuni BIOS più vecchi che non sono in grado di fare il boot di una partizione che si trovi oltre il limite di 1024 cilindri. Per approfondire questo argomento bisogna studiare i diversi tipi di BIOS e le diverse modalità di indirizzamento utilizzate (CHS fisici, CHS logici, LBA, ecc.), argomenti che esulano dalle finalità di questa guida e comunque non essenziali per l'installazione.
>
> Con i BIOS più recenti, che supportano le estensioni int13, è possibile installare e avviare NetBSD su partizioni > 8GB purché il codice di boot di NetBSD sia installato nel MBR, cosa che si può ottenere installando il boot manager di NetBSD.

Anche la schermata successiva dipende dalla versione di NetBSD: la 1.5 se si accorge della presenza di più sistemi operativi chiede se si vuole installare un *boot manager*. Oltre a installare il boot manager, sysinst procede anche a configurarlo; si può specificare quali scritte appariranno all'avvio del computer, quale sistema sarà avviato di default e con quale timeout. La schermata di configurazione del boot manager è mostrata nella Figura 3-12.

Figura 3-12. Configurazione del boot selector

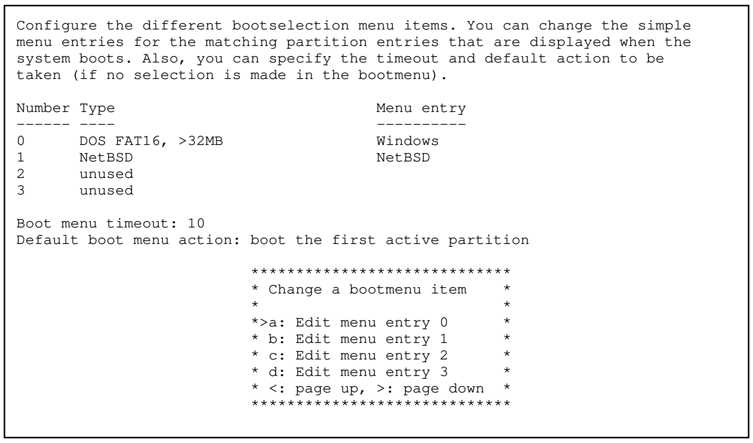


```
Configure the different bootselection menu items. You can change the simple
menu entries for the matching partition entries that are displayed when the
system boots. Also, you can specify the timeout and default action to be
taken (if no selection is made in the bootmenu).

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:** in questa schermata si possono usare i tasti < e > per scorrere la lista di opzioni se i tasti freccia per qualche motivo non funzionano.

Indicare le partizioni che devono comparire nel menu del boot manager all'avvio del PC selezionando una delle opzioni da “a” a “d”. Nella colonna "Menu entry" devono comparire i nomi di tutte le partizioni che si vogliono avviare dal boot manager come mostrato nella Figura 3-13.

Figura 3-13. Boot selector configurato



```
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  *
                          *****************************
```

Scegliendo l'opzione "e" si può specificare un timeout per il menu di avvio: trascorsi i secondi specificati viene automaticamente avviato il sistema di default, definito mediante l'opzione “f”. Come default si può indicare

- una partizione specifica
- un altro disco fisso
- la prima partizione attiva

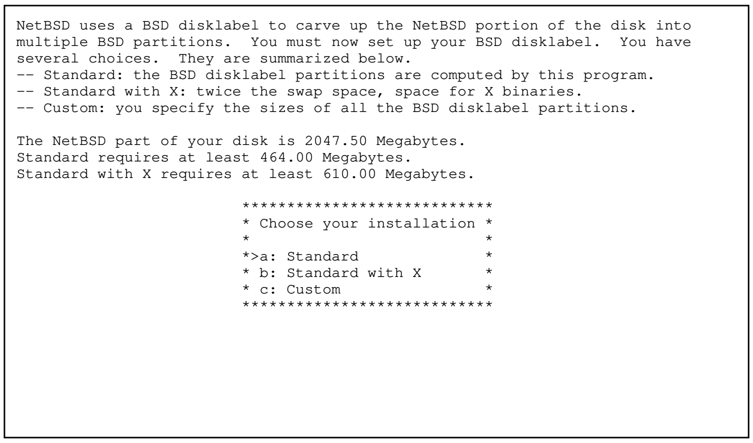
All'uscita di questa schermata si può dire terminata la parte dell'installazione relativa al partizionamento BIOS.

Fino ad ora abbiamo parlato di partizioni riferendoci a quelle che NetBSD chiama partizioni BIOS o *slice*: nell'esempio precedente abbiamo due slice (DOS e NetBSD). Adesso è venuto il momento di definire le partizioni BSD.

3.4.6. Disklabel

Il programma di installazione presenta tre alternative per la creazione delle partizioni, come illustrato nella Figura 3-14.

Figura 3-14. Tipo di 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                *
                         ****************************
```

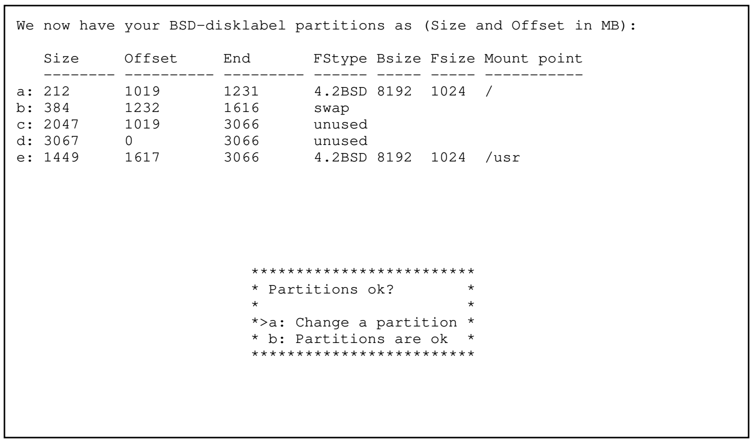
Per una prima installazione conviene selezionare "a:" oppure "b:", lasciando al sistema il compito di decidere le partizioni e la loro dimensione. Naturalmente in questo esempio non ci accontenteremo di una soluzione così semplice e cercheremo di complicarci la vita (a fini puramente didattici, si intende) modificando a mano le partizioni.

> **Nota:** quando si lascia decidere al sistema occorre comunque una certa cautela se lo spazio su disco è poco: controllare le indicazioni date dalla schermata sullo spazio necessario per le installazioni standard. Se non è sufficiente, bisognerà modificare la disklabel o crearne direttamente una personalizzata. Il programma di installazione della versione 1.5 si accorge automaticamente di queste incongruenze e visualizza una schermata di avvertimento.

3.4.7. Creazione di una disklabel

La strategia che seguiremo sarà di far creare al sistema una disklabel in automatico e poi di modificarla a mano. Scegliendo l’opzione “b” della Figura 3-14 viene mostrata la schermata di Figura 3-15.

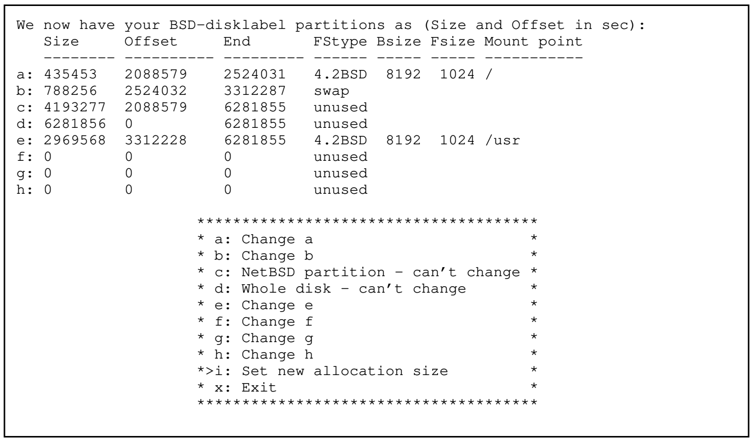
Figura 3-15. Disklabel standard



```
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  *
                          *************************
```

A questo punto ci si potrebbe limitare a confermare (scegliendo l’opzione “b”) e il lavoro sarebbe finito: supponiamo invece di decidere che la partizione di swap è troppo grossa e di volerla ridurre per aumentare `usr`. Scegliendo l’opzione “a” cambiamo la dimensione della partizione di swap: nella nuova schermata cambiamo l’unità di misura, passando ai settori, con il risultato mostrato in Figura 3-16.

Figura 3-16. Modifica della 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                            *
                    **************************************
```

La sequenza delle partizioni (cioè le lettere assegnate alle partizioni) è standard; alcune lettere hanno un significato ben preciso.

- *a* è la partizione *root*.
- *b* è la partizione di *swap*.
- *c* copre l’intera slice NetBSD (e quindi non copre la parte DOS).
- *d* copre l’intero disco. Si noti che l’area coperta da questa partizione “esce” dalla slice di NetBSD. Con un meccanismo analogo a questo (cioè creando una partizione che “esce” dalla slice di NetBSD) sarà possibile consentire a NetBSD di “vedere” partizioni DOS, Linux, ecc.

- *e* è la prima partizione libera, su cui viene solitamente montato `/usr`.

> **Nota:** il significato delle lettere attribuite alle partizioni non è lo stesso su tutte le piattaforme supportate da NetBSD. L'elenco precedente è specifico del port-i386.

Le partizione *b* e *c* dovrebbero essere lasciate inalterate (vanno modificate solo in caso di errata impostazione da parte di sysinst). Per la partizione *a* si può cambiare tranquillamente la dimensione ma è meglio mantenere il punto di *mount*. Si possono creare fino a 8 partizioni, per cui sono disponibili le lettere da *e* a *h*.

Per modificare la partizione di swap bisogna modificare la partizione *b*. Inoltre sarà necessario cambiare anche la partizione *e*, per farla iniziare dopo la fine della partizione *b*. Le partizioni *c* e *d* non si possono modificare (cosa abbastanza ovvia, dato il loro significato).

Supponiamo di voler assegnare circa 150 MB per lo swap (corrispondenti a 307200 settori): la partizione *b*, dunque, inizierà al settore 2524032 e finirà al settore 2831231 (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
 ...
```

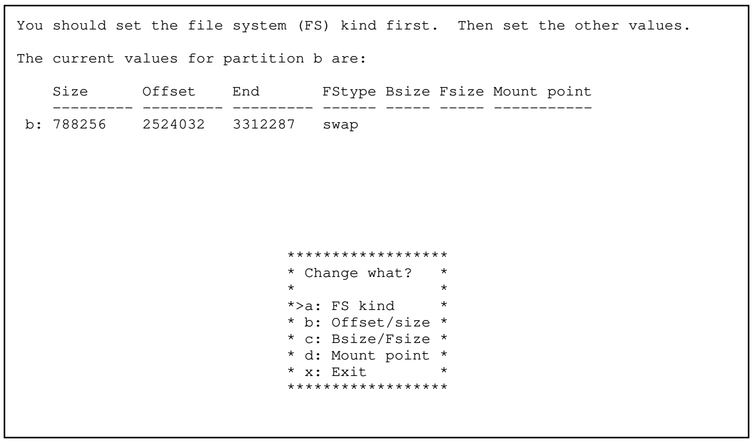
Lo spazio rimanente lo assegnamo alla partizione *e*: inizio 2831232, dimensione 3450624 e fine 6281855. I valori sono stati ricavati in questo modo: l'inizio è il settore che immediatamente segue il settore finale della partizione *b*, la fine è fissa (corrisponde all'ultimo settore della partizione NetBSD) e la dimensione è data da *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
```

L'esempio precedente mostra la situazione che si vuole ottenere sulla disklabel. Per modificare le due partizioni si usano le opzioni “b” ed “e” e si inseriscono i dati calcolati.

La schermata di modifica è mostrata in Figura 3-17.

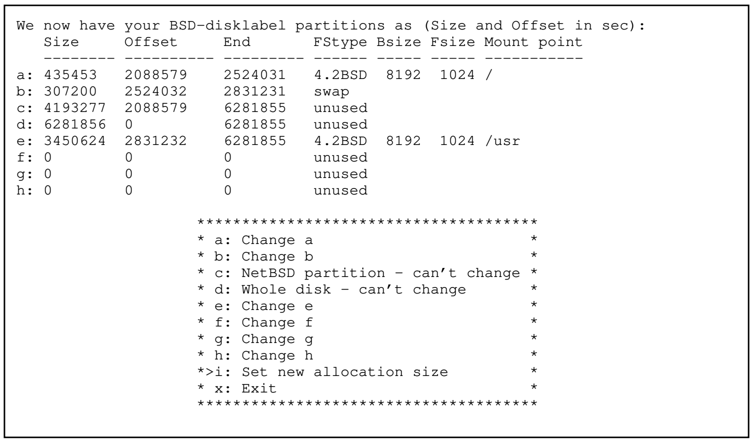
Figura 3-17. Modifica di una partizione BSD



```
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        *
                              ******************
```

La disklabel modificata è visualizzata nella Figura 3-18.

Figura 3-18. Disklabel modificata



```
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                            *
                    **************************************
```

Dimensioni delle partizioni: è molto difficile fornire dei criteri generali per decidere quante partizioni creare e come dimensionarle: dipende molto dall’uso che si farà del computer (server, workstation, server di posta, ecc.). Per una workstation possono bastare le partizioni *a* (per `/`) di almeno 50 MB, *b* (per lo swap) grande poco più della RAM disponibile ed *e* (per `/usr`) più grande possibile. Per un server si utilizzerà una suddivisione più soficitata. Le strategie di suddivisione del disco sono comunque moltissime ed esulano dallo scopo di questa guida.

Quando il risultato è soddisfacente si può selezionare "x: Exit" per salvare e uscire e si ritorna alla schermata della Figura 3-15 dove, questa volta, si sceglie l’opzione “b”.

3.4.8. Operazioni finali

La creazione delle partizioni (BIOS e BSD) era senz'altro lo scoglio più grosso dell'installazione; adesso la strada è tutta in discesa. Per completare la procedura bisogna dare un nome (a piacere) al disco (di default viene proposto *mydisk*) e confermare le operazioni eseguite.

> **Nota:** tutto quello che è stato fatto sino a ora non è ancora stato scritto su disco: è ancora possibile cambiare idea e abbandonare l'installazione.

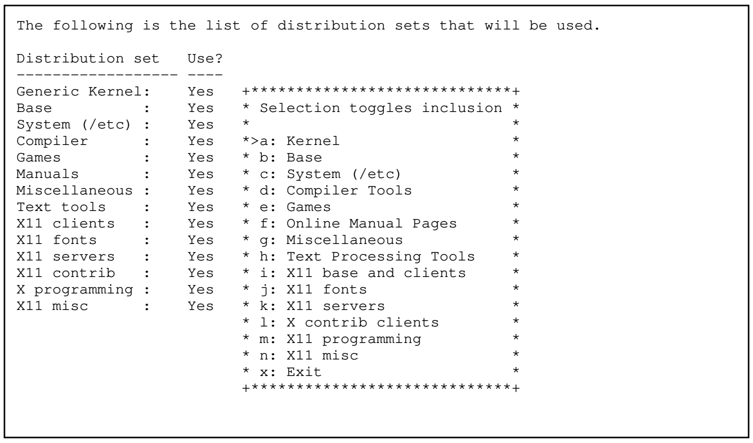
Il sistema provvede adesso a creare le partizioni e i filesystem eseguendo **fdisk**, **newfs**, **fsck** e **installboot** e poi, se tutto va bene, si passa all'installazione delle distribuzioni e del kernel.

> **Nota:** nella versione 1.5.3 di NetBSD è presente un'ulteriore schermata che chiede di selezionare il tipo di blocchi di boot da installare: normali o seriali. Selezionare il tipo normale, a meno di non sapere esattamente di che cosa si tratta...

3.4.9. Selezionare il tipo di supporto

La prima parte dell'installazione è terminata. Il prossimo passo è la scelta del tipo di installazione, che può essere *full* oppure *custom*. L'opzione full installa l'intera distribuzione mentre l'opzione custom installa il set di base (obbligatorio) e consente di selezionare i set restanti. È consigliabile scegliere l'opzione full a meno che non vi siano particolari problemi di spazio su disco. Nel seguito sceglieremo l'ozpione custom, che porta alla schermata di Figura 3-19.

Figura 3-19. Selezione dei set da installare



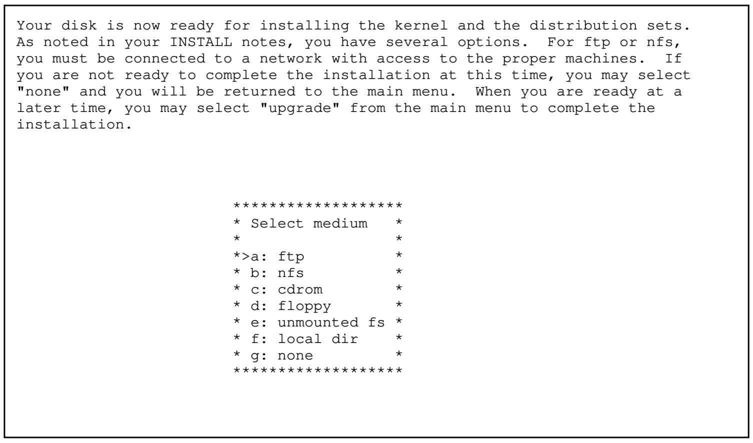
```
The following is the list of distribution sets that will be used.

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                     *
                         +*****************************+
```

I primi tre set sono obbligatori perché senza di essi il sistema non può funzionare. Gli altri set si possono attivare/disattivare con l'apposita opzione nel menu. Inizialmente tutti i set sono selezionati: in questo caso l'installazione sarebbe del tutto equivalente a quella dell'opzione "full" del menu precedente. Scegliamo di installare tutto e proseguiamo con l'opzione "x: Exit".

Successivamente sysinst chiede se si desidera vedere i nomi dei file estratti durante il processo di estrazione

A questo punto il programma di installazione ha bisogno di trovare gli archivi della distribuzione (i file `.tgz`): occorre indicargli dove andarli a prendere. Il menu offre varie possibilità:

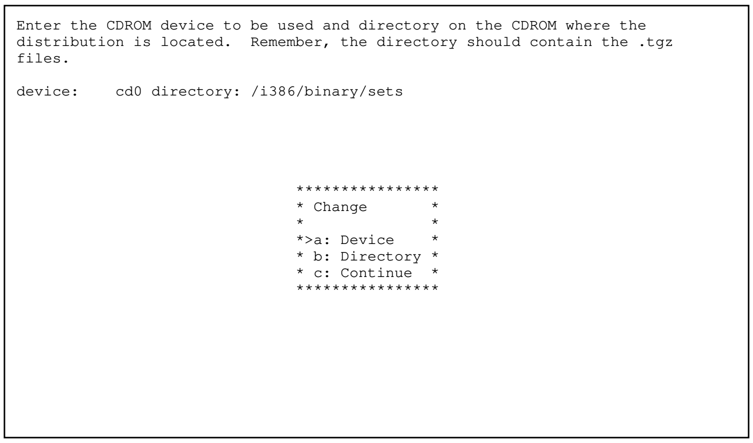
Figura 3-20. Media per l'installazione



```
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         *
                        *******************
```

Le varie opzioni sono spiegate in dettaglio nel solito file `INSTALL`. Si noti che è possibile installare anche da un file system non ancora montato come, per esempio, una partizione MSDOS. Ciò significa che è possibile copiare tutti i file della distribuzione in una partizione MS-DOS e utilizzarli poi per l'installazione.

Figura 3-21. Installazione da 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  *
                               ****************
```

Selezionando "cdrom" il sistema chiede il nome del dispositivo (per esempio cd0) e provvede a montarlo automaticamente. Inoltre bisogna specificare il percorso dei file della distribuzione binaria: sysinst presenta il percorso di default. Nel caso del CD-ROM creato da me, NetBSD si trova nella directory `NetBSD-1.5` e quindi bisogna modificare il percorso utilizzando l'opzione "b". Il nuovo percorso sarà:

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

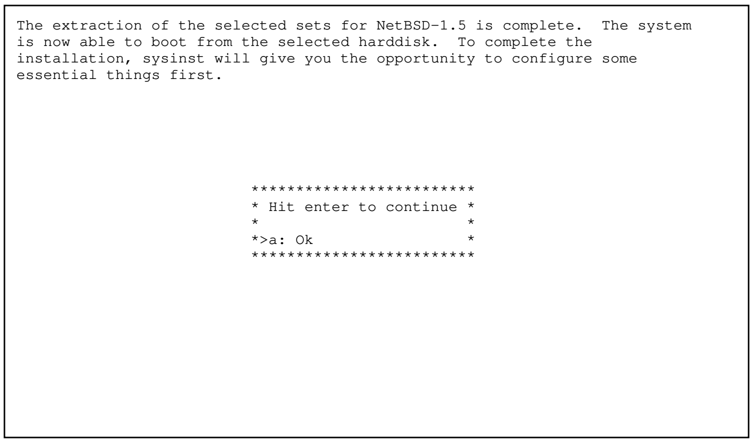
Nota: in quest'occasione si presenta il problema della tastiera americana a cui ho già accennato. Vedere la Sezione 3.3.1.

Il nome del dispositivo: è anche possibile controllare il nome del dispositivo corrispondente al CD-ROM.

1. Premere Ctrl-Z per sospendere sysinst: ci si trova al prompt della shell.
2. Dare il comando **cat /kern/msgbuf**; in questo modo vengono visualizzati i messaggi d'avvio del kernel, nei quali dovrebbe comparire anche il CD-ROM con il nome del dispositivo (per esempio *cd0*).
3. Se le scritte scorrono troppo rapidamente si può anche utilizzare **ed /kern/msgbuf** (a patto di conoscere ed).
4. Tornare al programma di installazione con il comando **fg**.

Al termine dell'installazione viene visualizzato un messaggio che conferma che le operazioni sono andate a buon fine. Infine, selezionando "a: ok" vengono creati i device file (dispositivi).

Figura 3-22. Installazione terminata!



```
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                 *
                          *************************
```

L'installazione è terminata. Sysinst offre la possibilità di effettuare alcune configurazioni prima di riavviare il sistema. La prima cosa che si può configurare è la timezone (cioè il fuso orario di appartenenza). Per l'Italia selezionare Europe/Rome. Nella schermata successiva si può definire la password di root. A questo punto si ritorna al menu principale e il sistema si può riavviare.

Capitolo 4. Il primo boot

Al termine dell'installazione si procede al primo avvio del sistema da disco fisso: se tutto è andato bene ci si troverà di fronte al prompt di login. Chi è abituato a trovarsi un sistema preconfigurato al termine dell'installazione del sistema operativo, con NetBSD può sentirsi un po' spaesato: seguendo le istruzioni di questa guida si potrà configurare rapidamente il sistema imparando nel contempo come funziona.

4.1. Se qualcosa è andato storto

Se qualcosa è andato storto nell'installazione può darsi che il sistema non si riavvii. Uno dei problemi più comuni è l'errata installazione del boot manager. In questo caso, inserire il floppy di installazione nel lettore e riavviare la macchina; quando compare la scritta

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

premere la barra spazio durante il conto alla rovescia di 5 secondi; il boot si interrompe e compare un prompt. Con il punto di domanda si visualizza l'aiuto.

```
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
```

Con il comando precedente si indica al sistema di avviarsi dal disco fisso anziché dal floppy. Se il sistema si avvia regolarmente dal disco fisso si tratta solo di un problema di boot manager, in caso contrario bisognerà probabilmente ripetere l'installazione. Per installare nuovamente il boot manager o modificarne la configurazione si usa il comando **fdisk** con l'opzione `-B` (vedere la Sezione 20.4 per una spiegazione dettagliata).

4.2. Login

Per il primo login bisogna utilizzare l'utente *root* (l'account dell'amministratore di sistema), l'unico definito sul sistema al termine dell'installazione. Se durante l'installazione è stata definita la password di root, bisogna inserirla, altrimenti è sufficiente dare invio alla richiesta della password.

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

4.3. Cambiare la disposizione della tastiera

A questo punto la disposizione della tastiera è rimasta ancora quella dell'installazione, cioè quella della tastiera americana. Prima di cominciare a configurare il sistema conviene cambiare la disposizione per utilizzare la tastiera italiana dando il comando

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

La lista completa delle configurazioni di tastiera supportate si trova in `/sys/dev/wscons/wsksymdef.h` (cercare KB_ENCTAB), ma le più comuni sono:

de
dk
fr
it
jp
sv
uk
us
no
es

Questo settaggio durerà fino al prossimo boot, il che non è comodo. Per ripristinarlo automaticamente si deve modificare il file `/etc/wscons.conf`: un'occhiata ai commenti contenuti nel file dovrebbe essere sufficiente a capirne il meccanismo di funzionamento. Per approfondire l'argomento: wscons.conf(5). Per esempio, per utilizzare la tastiera italiana basta aggiungere la riga seguente al file `/etc/wscons.conf`:

```
encoding it
```

Possiamo anche personalizzare alcuni tasti, per aggiungere le parentesi graffe, il carattere tilde, ecc. creando il file `/etc/itmap.it` (il nome è a piacere) con il seguente contenuto:

```
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
```

Infine aggiungere la riga per caricare la mappa nel solito file `/etc/wscons.conf`:

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

Nel seguito vedremo come fare in modo che NetBSD usi di default la tastiera italiana compilando un nuovo kernel e anche come personalizzare il layout della tastiera.

4.4. Il comando più importante

Anche se può apparire strano, il comando più importante in questo momento è **man**, che visualizza una pagina di manuale.

man comando per avere informazioni su un comando e **man -k argomento** per avere un elenco di comandi che hanno a che fare con 'argomento' (si può anche usare **apropos argomento**).

Naturalmente il primo comando da dare è

```
# man man
```

Il manuale è suddiviso in nove sezioni che non contengono solo informazioni sui comandi, ma anche descrizioni di alcune caratteristiche del sistema. Per esempio si provi a consultare la pagina di hier(7), che descrive dettagliatamente il layout del filesystem di NetBSD

```
# man hier
```

Per chi fosse un po' perplesso, *hier* è l'abbreviazione di hierarchy. Altre pagine informative di questo tipo sono, per esempio, release(7) e packages(7); ogni sezione del manuale ha poi una pagina *intro* che ne descrive il contenuto. Provare, per esempio

```
# man 8 intro
```

Esempio 4-1. Le sezioni del manuale

1. comandi utente dell'ambiente BSD
2. chiamate di sistema
3. funzioni della libreria C
4. dispositivi (device driver) e relative funzioni
5. formati dei file
6. giochi
7. informazioni varie (per esempio i pacchetti di macro di groff)
8. manutenzione di sistema
9. struttura del kernel

Talvolta un argomento è presente in più di una sezione del manuale; per visualizzare l'argomento della sezione che interessa è necessario indicare il numero di sezione nel comando man. Per esempio, *time* è presente nella sezione 1 (il comando utente time), nella sezione 3 (la funzione time della libreria C) e nella sezione 9 (la variabile di sistema time). Se ci interessa la funzione, dobbiamo specificare

```
# man 3 time
```

Per visualizzare tutte le sezioni

```
# man -a time
```

4.5. Cambiare la password di root

Al termine dell'installazione l'unico utente definito sul sistema è root, l'amministratore del sistema. Se nel corso dell'installazione non è stata definita una password per root (il che non era possibile con le versioni precedenti la 1.5) è arrivato il momento di farlo, usando il comando **passwd**.

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

Le password non vengono visualizzate sullo schermo mentre vengono scritte. Nel seguito vedremo come aggiungere altri utenti.

4.6. Cambiare la shell

La shell di default di root è csh; se questo non vi dice niente vi conviene mantenerla e cominciare a studiarne il funzionamento (**man csh**): si tratta di un'ottima shell per l'uso interattivo anche se manca della *history editing* (mancanza rimediata da tcsh, bash, ksh e dalla stessa `/bin/sh` di NetBSD). Se invece csh non vi piace potete cambiare la shell dell'utente root con il comando **chsh**.

> **Nota:** si noti che **chsh** per effettuare le modifiche lancia l'editor specificato dalla variabile di ambiente EDITOR (oppure vi se la variabile non è definita).

Le shell disponibili sul sistema al termine dell'installazione sono

csh
sh
ksh

La nuova shell verrà utilizzata al prossimo login.

Naturalmente è possibile installare molte altre shell (tcsh, bash, zsh, ecc.) utilizzando il sistema dei package, che vedremo più avanti.

Questo è un buon momento per creare/copiare i file di configurazione della shell (.cshrc, .login, ecc... a seconda della shell scelta).

4.7. Cambiare l'ora di sistema

NetBSD, come tutti i sistemi Unix, utilizza un orologio di sistema impostato sull'ora di Greenwich (UTC). Pertanto, per il funzionamento corretto del sistema è necessario impostare l'orologio CMOS sull'ora di Greenwich e non sull'ora locale. Purtroppo ciò non è possibile se si usa un sistema dual-boot con, per esempio, Windows, che richiede che l'orologio di sistema sia impostato sull'ora locale. Per risolvere questo problema bisogna informare NetBSD, agendo sulla variabile rtc_offset del kernel. Si può ricompilare il kernel dopo aver modificato il file di configurazione oppure modificare direttamente il kernel esistente (la modifica diventerà effettiva al prossimo riavvio), operazione illustrata dal prossimo esempio.

```
# 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
```

Si noti che il valore indicato è in minuti; se è in vigore l'ora legale utilizzare -120 al posto di -60.

Per visualizzare il corrente valore della variabile:

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

A questo punto abbiamo semplicemente indicato al kernel come fare a convertire in UTC l'ora dell'orologio di sistema. Se durante l'installazione non è stato impostato il fuso orario, lo si deve fare adesso (o comunque conviene controllare che l'impostazione sia corretta). Per l'Italia l'impostazione è la seguente:

```
# rm -f /etc/localtime
# ln -s /usr/share/zoneinfo/Europe/Rome /etc/localtime
```

Una volta che il sistema è impostato correttamente si possono cambiare la data e l'ora con il comando seguente:

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

4.8. Configurare `/etc/rc.conf`

Nei sistemi BSD il file `/etc/rc.conf` è quello che controlla cosa viene eseguito in automatico all'avvio del sistema: in un certo senso si può dire che è il cuore della configurazione del sistema.

Il meccanismo di funzionamento di questo file è cambiato con la versione 1.5 di NetBSD. Nelle versioni precedenti la 1.5 tutti i valori da definire si trovavano nel file `/etc/rc.conf`: inizialmente contenevano le impostazioni di default che l'utente doveva poi personalizzare modificando direttamente il file stesso. A partire dalla versione 1.5 le impostazioni di default sono definite nel file `/etc/defaults/rc.conf`: `/etc/rc.conf` contiene solamente le opzioni modificate dall'utente rispetto ai default. Le opzioni modificate vanno pertanto definite in `/etc/rc.conf`, usando `/etc/defaults/rc.conf` (che non deve essere modificato) come riferimento.

Un esame di questo file è altamente istruttivo e la lettura della relativa pagina di manuale, che descrive tutte le opzioni, è caldamente consigliata.

```
# man rc.conf
```

Al termine dell'installazione sysinst dovrebbe avere già effettuato le modifiche fondamentali al file. Controllare che siano presenti le seguenti linee:

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

Se “rc_configured” non è YES il sistema entra in modalità mono utente alla fine del boot. La direttiva “wscons” configura il driver di console. Già che ci siamo, se si desidera stampare, conviene inserire la direttiva:

```
lpd=YES
```

per attivare il demone di stampa.

4.9. Impostare il nome dell’host

È venuto il momento di indicare al sistema il nome del nostro host; vi sono due semplici modi per farlo. Se la macchina fa parte di una rete, il nome della rete sarà obbligato e il nome dell’host ci verrà probabilmente indicato dall’amministratore. Se invece si tratta di una postazione isolata (per esempio il computer di casa) si può scegliere un nome di fantasia; per esempio: ape.insetti.net. Per ulteriori approfondimenti vedere il Capitolo 9.

Il primo modo per indicare il nome dell’host è quello di inserire la seguente direttiva nel file `/etc/rc.conf`:

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

Anziché indicare il nome dell’host direttamente nel file di configurazione, lo si può scrivere nel file `/etc/myname`. I due metodi sono del tutto equivalenti.

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

4.10. Riavvio del sistema

In questa prima sessione abbiamo:

- Configurato la tastiera.
- Cambiato la password di root.
- Cambiato la shell di root (opzionale).
- Cambiato l’ora di sistema.
- Definito il fuso orario di appartenenza.
- Verificato il file `/etc/rc.conf`.
- Definito il nome dell’host.

Al termine di queste prime operazioni bisogna riavviare il sistema:

```
# reboot
```

Capitolo 5. Operazioni successive

Il primo boot è ormai storia passata e la configurazione di base del sistema è completa. In questa sezione vengono descritte, più o meno in ordine sparso, alcune operazioni comuni e alcuni comandi utili.

5.1. dmesg

Durante l'avvio del sistema il kernel scrive a video una lunga serie di messaggi relativi alle operazioni eseguite e alle periferiche riconosciute. Questi messaggi sono fondamentali sia per diagnosticare eventuali problemi sia per determinare i nomi dei dispositivi associati alle periferiche. Il comando **dmesg** ripropone i messaggi visualizzati dal kernel durante l'avvio del sistema, consentendo di controllarli con tutta calma. In particolare, quando si chiede assistenza su una delle mailing list di NetBSD perché qualche periferica non funziona a dovere conviene sempre accludere al messaggio l'output di dmesg (o almeno la parte relativa alla periferica in questione).

```
# dmesg | more
```

Se qualcosa sul sistema non funziona come dovrebbe e decidete di ricorrere all'aiuto di una delle mailing list di NetBSD, ricordatevi di accludere al vostro messaggio l'output di dmesg rilevante per il vostro problema. Queste informazioni saranno d'aiuto agli altri per diagnosticare il vostro problema.

> Dalla release 1.4.2 di NetBSD il processo di boot scrive una copia dell'output del comando dmesg nel file `/var/run/dmesg.boot`. Questa caratteristica è particolarmente utile per quei sistemi (per esempio server) che vengono resettati di rado. Nel corso del funzionamento, infatti, i nuovi messaggi aggiunti causano l'eliminazione di quelli più vecchi.

5.2. Montare il cdrom

Una delle domande che ricorre più frequentemente sulle mailing list è "come si fa a montare il CD-ROM?": i nuovi utenti sono di solito sorpresi dal fatto che il programma di installazione riconosce e utilizza il lettore di CD ma dopo il riavvio il sistema sembra essersene "dimenticato". Vediamo ora la procedura da adottare. Il device corrispondente al CDROM si può ricavare con il già citato comando **dmesg**. Supponiamo, per esempio, che dmesg visualizzi queste informazioni:

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

In questo caso si potrà montare il cdrom con

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

Per evitare di ripetere questi comandi ogni volta, conviene inserire una riga in `/etc/fstab`.

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

A questo punto, senza bisogno di riavviare, per montare il cdrom basta dare il comando

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

Se il cdrom è montato non è possibile espellerlo manualmente (cioè premendo il pulsante sul lettore): bisogna prima dare il comando

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

In alternativa si può espellerlo via software (anche se è montato) con il comando

```
# eject cd0
```

5.3. Montare il floppy

Per usare il floppy si può utilizzare un meccanismo simile a quello del CD-ROM. Per prima cosa si crea il punto di mount, per esempio:

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

Per leggere e scrivere un floppy in formato MSDOS si può ora dare il comando:

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

Se si usano molto i floppy in formato MSDOS, il mio suggerimento è di installare il package “mtools” che accede direttamente al floppy (ma anche a una partizione DOS) in modo molto comodo.

5.4. Accedere a una partizione MS-DOS/Windows

Se NetBSD condivide il disco fisso con MS-DOS o con Windows è possibile rendere visibili le relative partizioni modificando la disklabel. La prima cosa da fare è determinare la geometria della partizione DOS, per esempio con il 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:** in questo esempio è stato usato il disco *wd0*. Ognuno deve sostituire l'identificativo corretto per il proprio disco (per esempio, *sd0*, ecc.)

L'esame dell'output di **fdisk** mostra che la partizione DOS inizia al settore 63 e ha dimensione 2088516. La partizione NetBSD inizia al settore 2088579 (si noti che 2088579 = 2088516 + 63). Questi dati servono per modificare la disklabel BSD, usando l'editor specificato dalla variabile di ambiente $EDITOR. È necessario aggiungere una riga alla disklabel specificando i dati della partizione MS-DOS appena ricavati e usando la prima lettera libera come identificativo della partizione.

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

Le partizioni da "a" a "e" erano occupate e, pertanto, è l'identificativo scelto per la riga è stato "f". I campi "size" e "offset" sono stati riempiti con i dati trovati prima. Ora si crea il punto di mount, per esempio:

```
# mkdir /msdos
```

e infine si aggiunge la riga seguente a `/etc/fstab`

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

Per accedere alla partizione MS-DOS basta dare il comando:

```
# mount /msdos
```

Con il metodo appena descritto si possono leggere partizioni FAT e FAT32. Se si vuole che la partizione MS-DOS venga montata automaticamente all'avvio, basta eliminare l'opzione "noauto", in questo modo:

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

5.5. Aggiungere un nuovo utente

La gestione degli utenti e dei gruppi si effettua tramite i programmi **useradd(8)** e **groupadd(8)**. Per esempio, per aggiungere un nuovo utente carlo, assegnandolo a un gruppo con il suo stesso nome, creiamo dapprima il gruppo con:

```
# groupadd carlo
```

poi creiamo il nuovo utente:

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

Con un solo comando abbiamo fatto parecchie operazioni in un colpo solo:

-G wheel

indica di aggiungere l’utente anche al gruppo “wheel” (quello di root), il che gli consentirà di eseguire il comando **su**.

-g carlo

specifica il gruppo di appartenenza dell’utente, il questo caso il gruppo precedentemente creato.

-m

richiede la creazione della home directory per l’utente.

-s /bin/ksh

sceglie la shell ksh per l’utente.

carlo

il nome dell’utente da aggiungere.

In questo esempio abbiamo manualmente scelto la shell ksh, però i valori di default utilizzati da useradd possono essere configurati: vedere la pagina di manuale useradd(8). Il comando

```
# useradd -D
```

mostra i valori di default correnti.

Il comando:

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

imposta ksh come shell di default del comando useradd.

Nota: lo strumento da utilizzare per aggiungere o modificare gli utenti manualmente è **vipw**, che consente di modificare in modo sicuro il file delle password.

5.6. Shadow password

Le shadow password sono il default per NetBSD, e non possono essere disabilitate: tutte le password di `/etc/passwd` contengono un ’*’; le password crittografate sono contenute in un file separato, `/etc/master.passwd`, che può essere letto solo da root. Quando si lancia **vipw** per editare il file delle password, il programma apre una copia di `/etc/master.passwd`; quando si esce, vipw controlla la validità della copia, genera `/etc/passwd` e installa il nuovo `/etc/master.passwd`. Infine esegue **pwd_mkdb** che crea i file `/etc/pwd.db` e `/etc/spwd.db`, che sono dei database equivalenti a `/etc/passwd` e `/etc/master.passwd` e consentono un accesso più veloce.

La conseguenza di tutto ciò è che le shadow password sono gestite in modo automatico (trasparente, per usare un termine che va di moda oggi) da NetBSD: finchè si usa vipw per editare il file delle password non è necessario alcun cambiamento alle procedure standard di amministrazione di sistema.

È importante ricordare che l’amministrazione degli account deve *sempre* essere effettuata usando **vipw** (oppure chfn, chsh, chpass, passwd): non bisogna *mai* modificare direttamente `/etc/master.passwd`.

5.7. Localizzazione

La localizzazione è un settore in cui NetBSD è purtroppo ancora piuttosto debole; il supporto è ancora carente anche se sarà colmato nelle prossime release, visto che il codice necessario è già presente nella versione current. Nel frattempo ci si deve un po' arrangiare. Per gli utenti europei, la prima cosa da fare è procurarsi il tarball `locale.tgz` dal sito di NetBSD e installarlo con i comandi:

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

Successivamente aggiungere le istruzioni seguenti ai file globali di configurazione della shell (per esempio `/etc/profile`):

```
LANG=it
export LANG
LC_CTYPE=it_IT.ISO8859-1
# Per avere il simbolo dell'euro
# LC_CTYPE=it_IT.ISO8859-15
export LC_CTYPE
```

Queste impostazioni consentono di utilizzare le lettere accentate e di avere i menu in italiano in alcuni programmi (per esempio mutt).

5.8. Arresto/riavvio del sistema

Il comando da usare è **shutdown**. Per esempio:

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

In alternativa si possono usare i comandi:

```
# halt
# reboot
```

halt/reboot e **shutdown** non sono sinonimi: quest'ultimo è molto più sofisticato, consente di programmare la chiusura del sistema, di notificare gli utenti, ecc. Per i dettagli consultare le rispettive pagine di manuale.

Capitolo 6. Stampa

In questo capitolo viene descritta una semplice configurazione per una stampante utilizzando, a titolo di esempio, una HP Deskjet 690C collegata alla prima porta parallela. In un primo momento viene configurata la stampa di documenti di testo e poi la stampa di documenti PostScript con il programma Ghostscript.

6.1. Attivare il demone di stampa

Al termine dell'installazione non è ancora possibile stampare perchè il demone di stampa non è abilitato. Per attivarlo è sufficiente aggiungere la seguente linea al file `/etc/rc.conf`

```
lpd=YES
```

Per controllare se **lpd** è attivo, dare il seguente comando:

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

Se il comando precedente non dà luogo a output, vuol dire che **lpd** non è attivo, nel qual caso lo si può avviare manualmente con il seguente comando:

```
# lpd -s
```

Passiamo ora a effettuare una prova di stampa, usando il programma **lptest**. Per esempio:

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

L'output dovrebbe presentarsi sotto forma di righe incolonnate. Se invece si ottiene un effetto "a scalini", probabilmente la stampante è configurata per utilizzare un carattere <CR> (carriage return, ASCII 13) per tornare a inizio linea e un successivo carattere <LF> (line feed, ASCII 10) per avanzare di una linea mentre NetBSD (e UNIX in generale) usa solo il carattere <LF>. Questo problema si può sistemare:

- cambiando la configurazione della stampante
- utilizzando un semplice filtro di output (vedi sotto)

> **Nota:** nell'esempio precedente in realtà lo spooler lpd non viene utilizzato, perché l'output di lptest viene inviato direttamente al driver della stampante (`/dev/lpt0`).

6.2. Configurare `/etc/printcap`

In questa sezione viene spiegato come configurare la stampante di esempio per la stampa di documenti di testo.

Per prima cosa bisogna modificare `/etc/printcap`, creando una descrizione per la stampante usando l'identificativo standard *lp*.

Esempio 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:
```

Il formato del file e le opzioni sono descritte in dettaglio in printcap(5). L'unica cosa da segnalare è che, per risolvere il problema degli "scalini" è stato inserito un puntatore a un filtro con l'istruzione:

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

> **Driver di stampa:** nell'Esempio 6-1 viene usato il dispositivo *lpa#* che è il driver "polled" della stampante al posto di *lpd#*, che è il driver standard a interrupt. Alcune stampanti non gestiscono correttamente gli interrupt, con la conseguenza che impiegano ore per stampare una pagina; con questo tipo di stampanti (e la HP Deskjet 690C è una di quelle) basta usare il driver polled (si noti che è anche possibile compilare un kernel indicando che lpd deve funzionare in modo polled).

Poi bisogna creare le directory di spool specificate con l'opzione `sd`, che verranno usate da **lpd** per accumulare i dati.

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

A questo punto bisogna solo creare il filtro di stampa `lpfilter`. Il suo unico compito è configurare la stampante, eliminando il già citato problema degli scalini, prima di inviare il testo. Per la Deskjet 690C la sequenza da inviare è "ESC &k2G". Dopo aver creato il filtro bisogna rendelo eseguibile.

Esempio 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:** un altro filtro che si può utilizzare è
>
> ```
> :if=/usr/libexec/lpr/lpf:
> ```
>
> i cui sorgenti si trovano in `/usr/src/usr.sbin/lpr/filters`. Rispetto all'esempio presentato sopra, si tratta di un filtro molto più elaborato, concepito per elaborare l'output di **nroff** che oltre a convertire LF in CR + LF gestisce la sottolineatura e la sovrascrittura e espande i caratteri di tabulazione a colonne multiple di 8.

Adesso si può provare a eseguire nuovamente il programma **lptest**, questa volta sfruttando lo spooler **lpd**.

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

lpr è un programma di stampa che utilizza lo spooler per inviare i dati alla stampante (l'opzione `-h` serve a eliminare la stampa della pagina di intestazione).

> **Nota:** l'opzione `-h` non è necessaria se in `/etc/printcap` è stata specificata l'opzione `sh`.

Se tutto è andato bene gli scalini sono stati eliminati. Questo non è l'unico metodo per eliminare il problema degli scalini: si possono utilizzare molti tipi di filtri, anche programmi scritti in C, ma questa soluzione ha il pregio di essere molto semplice.

6.3. Configurazione per Ghostscript

Adesso che la stampa di base funziona, si può pensare a configurare la stampa di file in formato PostScript. Poiché la stampante utilizzata nell'esempio non supporta la stampa di file PostScript in nativo, bisogna usare un programma che converta il PostScript in un formato comprensibile per la stampante: questo programma è Ghostscript, che si può installare dalla collezione dei package (vedere Capitolo 8). In questa sezione verrà mostrato un esempio di configurazione per l'uso di Ghostscript con la stampante HP Deskjet 690C. Nell'esempio viene creato un secondo identificativo per la stampante; questo nuovo nome utilizza un filtro diverso, in grado di chiamare Ghostscript per la stampa dei file PostScript. Pertanto, i documenti di testo andranno stampati sulla stampante *lp* mentre quelli PostScript andranno sulla *lps*.

Come in molte altre situazioni, la soluzione proposta non è l'unica possibile. Per esempio, si potrebbe usare un unico filtro di stampa in grado di riconoscere il tipo di file che viene stampato e di agire di conseguenza. Con questo approccio basta definire un'unica stampante, un'unica directory di spool e un unico filtro, a prezzo di una maggiore complessità del filtro stesso. Infine, esistono programmi in grado di installare filtri "intelligenti" e di semplificare molto la fase di configurazione; si veda, per esempio, magicfilter nella collezione dei package.

Il file `/etc/printcap` si trasforma così:

Esempio 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:
```

Si noti, in particolare, l'opzione `mx#0`, che elimina le limitazioni alla massima dimensione del file di output: questa opzione è necessaria perché i file Postscript sono solitamente di notevoli dimensioni. Inoltre l'opzione `if` fa riferimento al nuovo filtro, che si chiama `lpfilter-ps`.

Adesso sul sistema possono essere utilizzati due nomi di stampante, lp e ps, che in realtà fanno riferimento a una stessa stampante fisica ma utilizzano filtri e directory diverse. Per stampare un file Postscript, naturalmente, non basta definire una voce in `/etc/printcap`: è necessario anche installare il programma Ghostscript.

Rimangono da creare la directory di spool e il filtro.

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

Il filtro per la stampa in formato postscript è più elaborato; il file da stampare viene dato in pasto all'interprete che lo trasforma in una sequenza di comandi adatta alla stampante. In questo modo si è trasformata una stampante economica a getto di inchiostro in una potente stampante Postscript. Per una descrizione dei comandi del filtro si veda la documentazione di Ghostscript. La chiave di volta del processo di stampa è la scelta del driver di stampante usato da Ghostscript, che in questo esempio è `cdj550`.

Esempio 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. Comandi utili per la stampa

In questa sezione vengono presentati alcuni comandi utili per la gestione delle stampe tipici degli ambienti BSD. Oltre ai già citati **lpr** e **lpd**, abbiamo:

lpq	esamina la coda dei lavori di stampa.
lprm	elimina lavori di stampa dalla coda.
lpc	controlla il sistema di stampa, consentendo di abilitare/disabilitare le stampanti e le loro proprietà.

6.5. Stampa in remoto

È possibile configurare il sistema di stampa in modo tale da poter stampare su una stampante collegata a un host remoto. Supponiamo, per esempio, di lavorare sull'host *wotan* e di voler stampare sulla stampante collegata all'host *loge*. Il file `/etc/printcap` di loge sia quello dell'Esempio 6-3. Da wotan quindi si potrà stampare anche file Postscript sfruttando Ghostscript su loge.

La prima operazione da compiere è di abilitare su loge le stampe provenienti dall'host wotan. Per far questo bisogna inserire il nome dell'host wotan nel file `/etc/hosts.lpd` di loge. Il formato del file è molto semplice: su ogni riga compare il nome di un host da abilitare.

Poi bisogna configurare `/etc/printcap` su wotan in modo da inviare i lavori di stampa a loge. Per esempio:

```
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
```

Vi sono quattro differenze tra questa configurazione e quella dell'Esempio 6-3.

1. La definizione di "lp" è vuota.
2. La voce "rm" definisce il nome dell'host a cui è collegata la stampante
3. La voce "rp" specifica il nome della stampante remota sull'host remoto.
4. Non è necessario specificare filtri di input perché vengono utilizzati quelli definiti su loge.

Una volta effettuate le operazioni precedenti i lavori di stampa di wotan vengono inviati automaticamente all'host loge.

Capitolo 7. Compilare il kernel

La maggior parte degli utenti di NetBSD finisce, prima o poi, per compilare un kernel personalizzato. In questo modo si ottengono vari risultati apprezzabili:

- si riducono considerevolmente le dimensioni del kernel (per es. da 2.5 MB a 1 MB) e quindi l'occupazione di memoria. Sulla versione 1.5 di NetBSD creando un kernel personalizzato si può passare da 4.7 MB a meno di due MB.
- si migliorano le prestazioni.
- si ottimizza il sistema.
- si risolvono problemi di conflitti/riconoscimenti di periferiche.
- si approfondisce la conoscenza del sistema.
- si personalizzano alcune opzioni (per esempio il tipo di tastiera, l'offset dell'orologio BIOS, ecc.).

7.1. Installare i sorgenti

> Se non avete l'archivio dei sorgenti del sistema operativo, lo potete scaricare dalla stessa fonte da cui avete ottenuto il sistema installato, per esempio: ftp://ftp.netbsd.org. È importante scaricare i sorgenti corrispondenti alla versione del sistema installato!

Il programma di installazione (sysinst) non installa i sorgenti del kernel, che devono quindi essere estratti a mano. L'archivio che li contiene è `syssrc.tgz` che si trova nella directory `source/sets`. Il file si scompatta con

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

Bisogna avere un po' di pazienza perché i file da estrarre sono molti e l'operazione dura parecchi minuti. I sorgenti vengono estratti nella directory `/usr/src/sys`; per accedervi si può anche usare il *link simbolico* `/sys`. Pertanto i comandi

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

sono equivalenti.

Terminata l'installazione è possibile rimuovere i sorgenti relativi alle architetture che non ci interessano e risparmiare un po' di spazio su disco (se questo è un problema). Per fare questo bisogna portarsi nella directory `/sys/arch` ed eliminare le sottodirectory non necessarie. Per esempio, per il port-i386 è sufficiente conservare la directory `i386` Per altri port può essere necessario conservare più di una directory sorgente in quanto vi possono essere delle dipendenze.

Tutto è pronto per creare un kernel personalizzato. Compilare un nuovo kernel è tutt'altro che difficile; le operazioni da eseguire (quattro o cinque in tutto) sono descritte sul sito di NetBSD.

7.2. Modifiche al layout della tastiera

Una volta installati i sorgenti del kernel, prima di ricompilare conviene effettuare una piccola modifica al layout della tastiera italiana contenuto nel file `/sys/dev/pckbc/wskbdmap_mfii.c`. Nella disposizione di default mancano alcuni caratteri utili per i programmatori, in particolare parentesi graffe e tilde. Io utilizzo queste impostazioni

```
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,
...
```

Con il codice precedente si ottengono queste corrispondenze

Combinazione	Carattere
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 di console:** dalla versione 1.4 NetBSD utilizza il driver multipiattaforma wscons per la gestione dello schermo e della tastiera. Le versioni precedenti utilizzavano pccons oppure pcvt. Per maggiori dettagli, vedere il Capitolo 13.

7.3. Ricompilare il kernel

La ricompilazione del kernel è descritta in modo semplice e preciso sul sito di NetBSD e, quindi, consiglio di fare riferimento a quelle istruzioni. Tuttavia, nel seguito riporto qualche indicazione, giusto per dare un’idea del procedimento.

Per poter ricompilare il kernel è necessario aver installato il set del compilatore (`comp.tgz`). La ricompilazione del kernel consiste dei seguenti passi

Compilazione del kernel

1. Modificare il file di configurazione del kernel
2. Configurare il kernel
3. Generare le dipendenze
4. Ricompilare
5. Installare il nuovo kernel

7.4. Modificare il file di configurazione del kernel

Il file di configurazione del kernel definisce il tipo, il numero e le caratteristiche dei dispositivi che devono essere supportati dal kernel. Il file si trova nella directory `/sys/arch/i386/conf`; per creare un file personalizzato conviene utilizzare come base una delle configurazioni esistenti. Per la maggior parte delle piattaforme la configurazione GENERIC è un buon punto di partenza; fare riferimento agli esempi presenti nella directory `/sys/arch/<arch>/conf`. Molte opzioni sono descritte da commenti nel file di configurazione; molte altre sono descritte in options(4).

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

Nomi: tradizionalmente i nomi utilizzati per i file di configurazione sono in maiuscolo.

Quando si modifica un file di configurazione del kernel si compiono, in sostanza, due tipi di operazioni:

1. si disabilita il supporto per le periferiche non presenti sul sistema e si abilita il supporto per quelle presenti (per esempio, chi non ha dispositivi SCSI può disabilitare il supporto per l'interfaccia SCSI nel kernel).
2. si abilita/disabilita il supporto delle caratteristiche supportate dal kernel (per esempio, il supporto per la compatibilità Linux, ecc.).
3. si ottimizzano i parametri del kernel.

Per disabilitare una linea del file di configurazione bisogna aggingere un carattere "#" all'inizio della linea stessa. Un buon punto di partenza per determinare cosa si può eliminare è lo studio dell'output del comando **dmesg**: per ogni linea del tipo

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

è necessario che nel file di configurazione siano presenti sia XXX sia YYY. Prima di arrivare a una configurazione minima è necessaria un po' di sperimentazione; per un sistema desktop, senza periferiche SCSI e schede PCMCIA si può arrivare a produrre un kernel di dimensioni meno che dimezzate rispetto a quello generico.

Conviene anche dare un'occhiata alle opzioni presenti nel file di configurazione, disabilitando quelle che non interessano. Ogni opzione è accompagnata da una breve descrizione; molte opzioni sono descritte più approfonditamente su options(4). Prima di ricompilare conviene sistemare le opzioni per il supporto della tastiera italiana e per il fuso orario:

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

Esiste anche un script Perl che genera automaticamente un file di configurazione ridotto analizzando l'output di **dmesg**. Questo script si può installare con la collezione dei package (pkgsrc/sysutils/adjustkernel). Per poterlo eseguire, naturalmente, è necessario avere installato Perl sul sistema, il che è tutt'altro che difficile: vedere il Capitolo 8. A titolo di anticipazione, un modo semplicissimo di installare Perl consiste nel procurarsi il package precompilato `perl-5.00404.tgz` e poi dare il comando:

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

A questo punto Perl è installato, configurato e pronto per lavorare: più facile di così...

Lo script si esegue con:

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

Personalmente, ho usato una sola volta questo script e ha dato ottimi risultati. Si faccia attenzione però al fatto che lo script configura le righe relative ai dispositivi, non quelle delle opzioni.

7.5. Configurare il kernel

Dopo aver modificato a piacere il file di configurazione del kernel (che supporremo si chiami `MYKERNEL`), dare il comando

```
# config MYKERNEL
```

Se `MYKERNEL` non contiene errori il programma creerà i file necessari per poter compilare il kernel. In caso contrario si deve tornare al punto precedente e sistemare gli errori. Il programma **config** analizza il file di configurazione indicato come parametro e crea i file necessari per la compilazione, abilitando i dispositivi richiesti.

7.6. Generare le dipendenze e ricompilare

Portarsi nella directory creata da config nel passo precedente, dare il comando **make depend** e, infine, il comando **make**.

```
# cd ../compile/MYKERNEL
# make depend
# make
```

In compilazione si possono verificare degli errori, nel qual caso la compilazione stessa si arresta. Se ciò avviene significa che il file di configurazione non è stato modificato correttamente e bisogna controllarlo e ripetere il procedimento. Un esempio tipico è il seguente: l'opzione B, che richiede l'opzione A, è attiva; l'opzione A, però, è commentata.

Una compilazione completa può durare da pochi minuti a molte ore, a seconda dell'hardware utilizzato. La tabella seguente contiene alcuni esempi:

CPU	**RAM (MB)**	**Tempo appross.**
486 DX2 50	20	1 ora
P166	96	15 minuti
PIII	128	5 minuti
68030/25	8	4 ore

Al termine della compilazione nella directory corrente sarà stato creato il file `netbsd`. L'ultimo passo da compiere è la copia del nuovo kernel in `/` (previo salvataggio del kernel precedente, mi raccomando).

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

Come già detto, personalizzando il kernel se ne possono ridurre le dimensioni in modo considerevole. Per esempio, al termine delle precedenti operazioni si ha:

```
-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
```

A questo punto non rimane che riavviare il sistema con un bel **reboot**.

7.7. Se qualcosa è andato male

È sempre possibile che al riavvio qualcosa non vada come sperato; al limite il nuovo kernel potrebbe non riuscire nemmeno a avviarsi. In questo caso non bisogna disperare; basta ripristinare il precedente kernel, che avevamo provvidenzialmente salvato, in modalità mono utente in questo modo:

- Riavviare la macchina
- Premere la barra spazio quando compare il prompt del boot durante il conto alla rovescia di 5 secondi

  ```
  boot:
  ```

- Dare il comando

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

- Dare i seguenti comandi

  ```
  fsck /
  mount /
  ```

```
mv netbsd.old netbsd
exit
```

Capitolo 8. I package

Il sistema dei package di NetBSD è una raccolta di strumenti che consentono di compilare e installare in maniera semplice una grande quantità di software di pubblico dominio per sistemi Unix. In generale sono sufficienti uno o due comandi per installare qualsiasi pacchetto perfettamente configurato e funzionante.

Il primo contatto con i package genera un po' di confusione perché apparentemente vi sono vari comandi che fanno le stesse cose. In realtà la questione è molto semplice: vi sono *due modi* per installare un programma

- Compilare i sorgenti del programma sul nostro sistema. Questo è il compito della collezione dei package, che è in grado di scaricare in modo automatico i sorgenti, compilarli, installare e configurare il programma e la documentazione con uno o due comandi. Come se non bastasse è in grado di determinare automaticamente le dipendenze e installare i package necessari. Il sistema dei package non viene installato con il sistema base, anche perché subisce frequenti aggiornamenti, per stare al passo con le nuove versioni dei programmi da installare. Il sistema dei package è descritto in modo molto completo ed esauriente sul sito di NetBSD.
- Installare una versione precompilata e preconfigurata del programma in un apposito formato. È questo il compito dei "pkgtools" una serie di programmi installati con il sistema di base. Questo sistema è più veloce ma meno flessibile del precedente. I pkgtools sono anche in grado di riconoscere e gestire (elencare, disinstallare, ecc.) i programmi installati con il sistema dei package; si tratta, per così dire, di strumenti complementari. Se si decide di installare esclusivamente programmi in forma precompilata, non è necessario installare la collezione dei package.

I due metodi appena illustrati sono quelli "comodi", che sfruttano il lavoro di configurazione fatto da qualcun altro. Se il programma che ci interessa non è compreso nella collezione dei package, nessuno ci impedisce di compilarlo a mano e di installarlo sul sistema. In questo caso è buona norma fare un piccolo sforzo in più e creare noi stessi un package, in modo che altri possano beneficiare a loro volta del nostro lavoro.

8.1. Installazione della collezione dei package

Prima di poter installare un pacchetto è necessario installare la collezione dei package, come descritto in dettaglio sul sito di NetBSD. In sostanza si tratta di:

1. Scaricare l'ultima versione dei sorgenti della collezione, che comprendono tutti i file di configurazione, da `ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/tar_files/`. Il nome del file da scaricare è `pkgsrc.tar.gz`.
2. Rimuovere la precedente collezione, se presente, con

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

3. Installare la nuova collezione con

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

L’esecuzione del comando precedente dura parecchio tempo perché il numero dei file da estrarre è notevole. Al termine dell’operazione il *framework* per l’installazione dei package è al suo posto: rimane solo da decidere quali package installare sul sistema.

Nota: immagino che a questo punto sia chiaro che con l’operazione precedente è stato installato sul sistema l’insieme dei file di configurazione necessari per l’installazione dei singoli programmi ma non i programmi stessi né i loro sorgenti. In sostanza adesso il sistema dispone della lista dei programmi installabili automaticamente e delle istruzioni per installarli.

Una volta installata la collezione dei package con un browser HTML come Lynx o Netscape si possono leggere i dettagli e le descrizioni di tutti i pacchetti disponibili. Per esempio:

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

Nota a marginePer non perdere i distfiles

Quando si installa una nuova collezione di package, occorre rimuovere la collezione precedente, che generalmente si trova in `/usr/pkgsrc`, il che comporta la perdita della sottodirectory `distfiles` se ci si dimentica di salvarne il contenuto. La directory `distfiles` è quella che contiene tutti i package scaricati da Internet e, pertanto, può essere seccante perderla. Il problema si può risolvere mettendo la directory `distfiles` da un’altra parte (non sotto la directory `pkgsrc`). Per esempio, creare la nuova directory per i distfiles:

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

e poi aggiungere al file `/etc/mk.conf` la riga seguente

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

Ovviamente si può usare una directory diversa da quella proposta nell’esempio, `/usr/pkgsrc_distfiles`.

Nel file `/etc/mk.conf` si possono configurare molti aspetti del comportamento del sistema dei package. Per una descrizione di tutte le opzioni si veda il file `/usr/pkgsrc/mk/mk.conf.example`.

8.2. Aggiornamento della collezione dei package

La collezione dei package viene aggiornata molto spesso; sul sito ftp si trova una nuova copia quasi ogni settimana. Per aggiornare la collezione sul proprio computer occorre seguire le istruzioni date per l’installazione.

A volte, oltre alla collezione dei package è necessario aggiornare anche i programmi di installazione (pkgtools). È facile capire se bisogna eseguire l’aggiornamento: quando si tenta di installare un pacchetto il sistema dei package avvisa che i pkgtools devono essere aggiornati. Per esempio:

```
# 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
```

Il metodo di aggiornamento più semplice è:

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

Fatto ciò, si può riprendere l'installazione del pacchetto originale, quello che aveva dato luogo al messaggio d'errore.

> **Nota:** Tecnicamente, il messaggio d'errore indica che la versione dei programmi di installazione è quella contenuta in `pkg_install-20000202.tar.gz`, che si può trovare sul sito ftp in `packages/distfiles/LOCAL_PORTS`. Pkg_install si installa come un qualsiasi altro pacchetto (`pkgsrc/pkgtools/pkg_install`).
>
> Per determinare esattamente la versione richiesta, esaminare il file `pkgsrc/mk/bsd.pkg.mk` e cercare la riga con
>
> ```
> PKGTOOLS_REQD = 20000202
> ```
>
> (la data 20000202 è solo un esempio).

8.3. Installazione di un programma

A titolo di esempio, proviamo a installare il programma addnerd , che serve a aggiungere utenti al sistema in maniera automatica (un po' come fa il comando adduser su altri sistemi). Per prima cosa ci si porta nella directory `/usr/pkgsrc/sysutils/addnerd`.

8.3.1. Scaricare i sorgenti

Se si dispone di un collegamento a Internet il Makefile è in grado di scaricare automaticamente il pacchetto e si può saltare alla sezione seguente.

Nota a margineScaricare su un'altra macchina

Uno scenario molto comune è quello in cui non si dispone di un collegamento veloce a Internet, per esempio sulla macchina di casa. In questo caso può essere conveniente scaricare i sorgenti su un'altra macchina che dispone di un collegamento migliore (per esempio al lavoro).

In caso contrario bisogna procurarsi il package sorgente e copiarlo nella directory `/usr/pkgsrc/distfiles`. Esaminando il Makefile, si trova il nome del pacchetto da scaricare (DISTNAME = addnerd-1.6). Il nome completo del pacchetto è `addnerd-1.6.tar.gz`.

Lo stesso risultato si può ottenere in modo più semplice con i comandi:

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

che in più mostrano anche gli indirizzi dei siti dai quali scaricare il programma.

8.3.2. Compilare i sorgenti

Per compilare il pacchetto scrivere

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

Il comando precedente scarica il package da Internet (se non è già presente nella directory dei distfile), estrae i sorgenti, applica i patch necessari per poterli compilare sotto NetBSD e infine compila il pacchetto.

Per installarlo:

```
# make install
```

L'installazione del pacchetto viene registrata dal sistema: si può verificare scrivendo **pkg_info -a**. A installazione completata si può fare un po' di pulizia con

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

Il secondo comando è mostrato solo a titolo di esempio; in questo caso non erano stati installati package dipendenti e quindi non era necessario fare pulizia.

8.4. Installazione di un pacchetto in formato binario

Come ho spiegato nell'introduzione, il sistema dei package è concepito per installare programmi in forma sorgente ma è in grado anche di installare pacchetti precompilati, generati da qualcun altro utilizzando la collezione stessa. In questo caso l'installazione è più veloce (non è necessario compilare il programma da installare).

I programmi precompilati si distinguono solitamente da quelli sorgenti perchè hanno estensione `.tgz` anziché `.tar.gz`.

> **Nota:** in realtà l'estensione dei tarball dei sorgenti non è sempre `.tar.gz`; il sistema dei package, tuttavia, è in grado di gestire anche le estensioni diverse (`.zip`, `.bz2`, ecc.).

Non è strettamente necessario scaricare manualmente un pacchetto binario per poterlo installare. Pkg_add è in grado anche di scaricarlo autonomamente quando vengono utilizzati URL FTP (l'uso di HTTP è possibile ma sconsigliato). Per esempio, si può specificare il nome del pacchetto con:

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

Se non si conosce il numero di versione del pacchetto disponibile sul sito FTP, è anche possibile omettere questa informazione dalla riga comando e **pkg_add** utilizzerà automaticamente l'ultima versione disponibile sul server FTP. Per esempio:

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

È anche possibile impostare la variabile di ambiente PKG_PATH con una lista di percorsi e URL separati da *;* e omettere il percorso nel comando pgg_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
```

Il comando precedente installa il primo pacchetto binario di tcsh trovato nei percorsi specificati.

A titolo di esempio, vediamo come procede l'installazione del programma texinfo in forma precompilata. L'installazione del pacchetto binario è molto semplice:

1. Copiare `gtexinfo-3.12.tgz` in una directory temporanea.
2. Dare il comando

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

3. Controllare l'avvenuta installazione con il comando

   ```
   # pkg_info
   ```

4. Rimuovere il file `gtexinfo-3.12.tgz` dalla directory temporanea.

I pacchetti precompilati sono molto comodi da usare perché comportano un tempo e uno sforzo minimo per l'installazione; tuttavia i pacchetti in forma sorgente sono più flessibili ancora perché consentono di personalizzare eventuali opzioni di configurazione e di installazione. L'installazione è più lunga ma, tutto sommato, non si installano pacchetti in continuazione...

Quando si installa un pacchetto binario con il comando **pkg_add**, conviene per prima cosa esaminarlo con il comando **pkg_info**. Per esempio:

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

In particolare conviene osservare il primo comando CWD per vedere dove viene installato il pacchetto. Se la directory base non è quella desiderata, bisognerà usare l'opzione `-p` di **pkg_add**. Per esempio, il pacchetto `jpeg-6b.tgz` si va a installare in `/usr/pkg`, mentre secondo me dovrebbe finire in `/usr/X11R6`. (In generale conviene estrarre in `/usr/X11R6` tutti i pacchetti della gerarchia X). Pertanto è preferibile estrarlo con:

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

8.5. Comandi per la gestione dei pacchetti

I comandi principali per l'amministrazione del sistema dei package sono

pkg_add

Programma di utilità per l'installazione di package precompilati.

pkg_delete

Programma di utilità per la disinstallazione di package precompilati. I nomi dei package possono essere specificati con o senza il numero di versione; pkg_delete determina automaticamente il numero di versione del pacchetto installato. Si possono usare caratteri jolly (facendo attenzione a prevenire l'espansione effetuata dalla shell); per esempio:

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

L'opzione `-r` consente la rimozione ricorsiva dei pacchetti; prima rimuove tutti i pacchetti che richiedono quello indicato e poi rimuove il pacchetto stesso. Per esempio:

```
# pkg_delete -r jpeg
```

rimuove il pacchetto jpeg e tutti quelli che lo utilizzano. Può essere utile per effettuare l'aggiornamento di un package.

pkg_info

Programma di utilità per la visualizzazione di informazioni sui pacchetti software installati e da installare.

pkg_create

Programma di utilità per la creazione di pacchetti software. Questo è il programma che viene usato per creare i pacchetti precompilati. Viene chiamato automaticamente dal sistema dei package e non è mai necessario lanciarlo a mano.

pkg_admin

Programma di utilità per l'esecuzione di varie funzioni amministrative sul sistema dei pacchetti.

8.6. Guida rapida alla creazione di package

L'autore di questa sezione (Guida rapida alla creazione di package) è Jason R. Fink

Questa sezione descrive un metodo semplice e veloce per la creazione di semplici package per la collezione di package di NetBSD. Per ulteriori approfondimenti si rimanda alla documentazione ufficiale contenuta nel documento `Packages.txt`.

8.6.1. Tools

Gli strumenti principali da utilizzare per la costruzione di un semplice package da aggiungere alla collezione dei package di NetBSD sono tre:

- url2pkg
- un prototipo di package
- pkglint

8.6.1.1. url2pkg

Il programma di utilità url2pkg si può installare dalla collezione dei package. Si tratta di uno strumento che consente al costruttore di package di preparare e verificare velocemente le strutture di base della costruzione di un package.

8.6.1.2. Prototipo di package

La scelta del prototipo è lasciata allo sviluppatore; `Packages.txt` mostra un esempio di costruzione di package nella sezione 11 ’A simple example of a package: bison’. Si noti che molti dei package della collezione sono tutt’altro che complicati.

8.6.1.3. pkglint

Il programma di utilità pkglint può essere installato dalla collezione dei package. Si tratta di uno strumento di verifica della correttezza dei package.

8.6.2. Primi passi

I passi iniziali sono molto semplici a patto di avere l’accortezza di verificare che il programma si compili correttamente dai sorgenti sotto NetBSD. In caso contrario la costruzione del package potrebbe complicarsi notevolmente. Si noti che è generalmente possibile generare dei “patch” da applicare ai sorgenti e da includere nel package per risolvere i problemi di compilazione (o, più in generale, di compatibilità), ma questo argomento esula dagli scopi di questa semplice introduzione.

8.6.2.1. Uso di url2pkg

Per prima cosa bisogna utilizzare url2pkg.

La procedura seguente dettaglia i passi da seguire per creare i file di base di un nuovo package.

1. Creare una nuova directory per il nuovo package, scegliendo una posizione appropriata in `pkgsrc`. Lasciare la nuova directory inizialmente vuota.
2. Portarsi nella nuova directory.
3. Dare il comando

   ```
   $ url2pkg
   ```

4. Il programma a questo punto chiede di inserire un URL: inserire l'URL richiesto e premere **Invio**.
5. Viene avviato l'editor vi per creare il Makefile per il nuovo package. Bisogna inserire il nome e la categoria di appartenenza del package e il nome e l'indirizzo e-mail della persona che curerà la manutenzione del package stesso.
6. Salvare il file e uscire.
7. url2pkg scarica automaticamente il package e lo mette nella sottodirectory `work`.
8. Infine, url2pkg genera i file MD5.

Con ciò si conclude la sessione di url2pkg. Si noti che url2pkg non riempie alcun file a eccezione del Makefile. Inoltre genera un file PLIST vuoto.

8.6.3. Operazioni successive

Ora che il Makefile è stato generato, occorre creare i restanti file sfruttando un package da usare come prototipo. Scegliere un package esistente e copiare i seguenti file dalla sottodirectory `pkg`, modificandoli opportunamente.

DESCR

Una descrizione del nuovo package, lunga alcune righe.

COMMENTS

Una descrizione del pacchetto lunga una riga. Non è necessario citare il nome del pacchetto stesso: sarà aggiunto automaticamente dai vari strumenti pkg_* quando verranno chiamati.

PLIST

Questo file descrive la posizione dei file da installare sul sistema. Per un package non molto complesso (per es. un eseguibile e una o due pagine di manuale) dovrebbe essere sufficiente dare un'occhiata al Makefile e agli script di installazione per capire dove mettere i file da installare.

8.6.4. Verifica con pkglint

Ora che tutti i file richiesti sono pronti, è venuto il momento di verificare la correttezza del package con il programma pkglint. Capita spesso che una sezione del Makefile debba essere spostata, modificata o integrata con ulteriori informazioni; è bene lanciare subito pkglint per avere un immediato riscontro di ciò che andrà cambiato. Vediamo ora un esempio di output di pkglint preso da 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.
```

In caso di errore è generalmente semplice risalire alle cause. Ecco un esempio di messaggio di errore ottenuto durante la costruzione di un package:

```
extract suffix not required
```

Il messaggio indica semplicemente che non era necessario definire nel Makefile il suffisso per l'estrazione.

8.6.5. Test finali

A questo punto, superata la verifica di pkglint, è consigliabile effettuare un "giro" completo di fetch (scaricamento), compilazione e installazione. Per uno svolgimento corretto bisogna prima cancellare la sottodirectory `work` e i distfiles già scaricati (che solitamente di trovano in `/usr/pkgsrc/distfiles`. In questo modo si garantisce che il test effettuato sarà completo.

8.6.6. Sottomettere un package con send-pr

Per prima cosa bisogna creare un "tarball" (file archivio tar compresso) dell'albero di directory del package stesso (compresa la sottodirectory `work`). Per esempio:

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

Successivamente il file archivio deve essere copiato in una locazione accessibile ai gestori della collezione dei package di NetBSD. Se questo non è possibile, conviene contattare i responsabili chiedendo se vi sono altri metodi utilizzabili per rendere disponibile l'archivio.

Il metodo preferito per informare i responsabili della collezione di package di NetBSD è usare il programma di utilità send-pr specificando la categoria "pkg", includendo il nome, la versione e l'URL del package e una breve descrizione dello stesso.

Si può utilizzare sia la versione di send-pr inclusa in NetBSD sia quella disponibile online all'indirizzo http://www.NetBSD.org/cgi-bin/sendpr.cgi?gndb=netbsd.

8.6.7. Note conclusive

Come già detto, questa breve guida si limita a considerare semplici package che installano un numero limitato di file sul sistema NetBSD, con l'ulteriore assunzione che i package stessi non abbiano dipendenze da altri package.

Per ulteriori approfondimenti si consiglia la lettura del già citato `Packages.txt`.

Capitolo 9. Introduzione alle reti TCP/IP

9.1. Reti TCP/IP

L'autore di questo capitolo (Introduzione alle reti TCP/IP) è Hubert Feyrer `<hubert@feyrer.de>`.

9.1.1. Introduzione

Il presente capitolo della guida è dedicato alle reti delle quali spiega gli elementi fondamentali per i lettori che non hanno grande familiarità con l'argomento. Il capitolo seguente, invece, contiene un'introduzione pratica alla configurazione di rete di NetBSD; che desidera configurare rapidamente la rete può iniziare con il prossimo capitolo e poi tornare con calma alla lettura del capitolo presente.

Da parte del lettore è richiesta la conoscenza di elementi di base dell'amministrazione del sistema: come diventare "root", come editare i file, come cambiare i permessi ai file, come fermare un processo, ecc. Vedere per esempio [AeleenFrisch] per approfondire questi argomenti. Altre a ciò, è necessario saper usare alcuni programmi di utilità quali telnet, ftp, ... Tali programmi non verranno qui descritti e, pertanto, per ulteriori informazioni si rimanda alle relative pagine di manuale, alla bibliografia e agli altri capitoli di questa guida.

Questa introduzione alle reti TCP/IP non ha pretesa di completezza: chi desidera approfondire gli argomenti trattati per avere un quadro più completo può fare riferimento a [CraigHunt]. Questo libro, oltre a descrivere gli elementi di base, spiega tutti i concetti e i dettagli necessari alla configurazione di tutti i servizi: un ottimo libro.

9.1.2. Protocolli di rete supportati

NetBSD supporta vari protocolli di rete, la maggior parte dei quali ereditati dal suo predecessore, 4.4BSD, e successivamente estesi e migliorati. Il primo protocollo implementato è quello oggi più importante, il Transmission Control Protocol/Internet Protocol (TCP/IP) del DARPA. Altri protocolli disponibili su NetBSD sono Xerox Network System (XNS), implementato all'UCB per collegare in rete macchine isolate, AppleTalk di Apple, ISO, CCITT X.25 e ARGO TP. Questi ultimi protocolli sono oggi usati solo da alcune speciali applicazioni.

Oggi TCP/IP è il protocollo più diffuso tra quelli menzionati al paragrafo precedente. Oltre a essere implementato da ogni sistema operativo e su ogni tipo di hardware, è anche il protocollo più usato in ambienti eterogenei. Pertanto, TCP/IP è la scelta più giusta per collegare il proprio computer a altre macchine a casa o alla rete aziendale o universitaria.

La versione corrente del protocollo TCP/IP è la 4 (*IPv4*); la versione 6 (*IPv6*) è in corso di sviluppo e il codice IPv6 del progetto KAME è stato integrato nelle release di NetBSD già a partire dalla 1.5.

Vi sono anche dei protocolli che non sono supportati da NetBSD, quali DECNET, IPX/SPX di Novell e NetBIOS di Microsoft. Questi protocolli, a differenza dei precedenti, sono proprietari, e non sono quindi definiti da alcuna RFC pubblica come gli altri standard aperti.

9.1.3. Interfacce supportate

Il protocollo TCP/IP può essere utilizzato su un'ampia gamma di interfacce. Tra quelle supportate da NetBSD ricordiamo Ethernet (10/100/1000MBd), linea seriale Arcnet, ATM, Fiber Channel, USB, HIPPI, FireWire (IEEE 1394), Token Ring, linee seriali e altre ancora.

9.1.3.1. Linee seriali

Vi sono varie ragioni che consigliano l'uso di TCP/IP su linea seriale.

- se un host remoto è raggiungibile solo per telefono, lo si può contattare via modem.
- praticamente tutti i computer dispongono di un'uscita seriale, e il cavo necessario è molto economico.

Lo svantaggio dei collegamenti via seriale è che sono più lenti degli altri metodi. NetBSD può utilizzare al massimo 115300 bit/s, il che rende il collegamento molto più lento rispetto ai 10 Mbit/s di Ethernet o ai 4 Mbit/s di Arcnet.

Per collegare un host con NetBSD a un altro host via seriale, eventualmente anche su linea telefonica, si possono usare due metodi:

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

La scelta dipende dal tipo di collegamento che si vuole effettuare. Per un collegamento telefonico utilizzando il modem conviene sfruttare PPP, che offre l'opzione di auto negoziare gli indirizzi IP e l'instradamento, cose complicate da impostare a mano. Invece, per collegarsi direttamente a un'altra macchina, per esempio con un cavo null-modem o una linea dedicata, il protocollo SLIP è più semplice da configurare con indirizzi e instradamento fissi ed è supportato praticamente da tutti i sistemi operativi.

PPP su una linea diretta è più difficile da configurare rispetto a SLIP a causa dell'handshake iniziale, che su SLIP non è presente. Ciò significa che con SLIP si inizia il collegamento da un lato e l'altro lato può inviare il primo pacchetto non appena è pronto.

[RFC1331] e [RFC1332] descrivono i protocolli PPP e TCP/IP su PPP. SLIP è descritto in [RFC1055].

9.1.3.2. Ethernet

Ethernet è l'interfaccia più utilizzata nell'ambito delle reti locali (LAN, Local Area Network) di macchine interconnesse in un'area limitata, quale può essere un ufficio, un'azienda o un'università. Ethernet si basa su un bus al quale possono connettersi più macchine e la comunicazione avviene sempre tra due nodi alla volta. Quando più coppie di nodi cercano di comunicare contemporaneamente si verifica un conflitto e le coppie cercheranno di riprovare la comunicazione al termine di un timeout. Il termine tecnico utilizzato per questa procedura è CSMA/CD (Carrier Sense w/ Multiple Access and Collision Detection.)

Inizialmente, l'hardware Ethernet consisteva in uno spesso cavo (giallo) che veniva collegato alle macchine usando speciali connettori che attraversavano la schermatura esterna del cavo stesso. Successivamente è arrivato il tipo 10base5, che usava connettori di tipo BNC da inserire in speciali connettori a T sulle macchine, con terminatori alle estremità della linea. Oggi il bus ethernet è contenuto all'interno degli switch e degli hub, e i cavi sono del tipo twisted pair e danno il nome a questo tipo di interfaccia: 10baseT per le reti a 10 Mbit/s e 100baseT per quelle a 100 Mbit/s. Nelle reti configurate con switch, si distinguono i due casi in cui la comunicazione tra nodo e switch avviene in modo full-duplex o half-duplex.

9.1.4. Il formato degli indirizzi TCP/IP

La corrente implementazione del protocollo TCP/IP (IPv4) usa identificativi di 4 byte (32 bit) (detti anche indirizzi IP) per indirizzare gli host.

Questi indirizzi IP sono unici a livello mondiale. Per raggiungere questo scopo, la loro amministrazione è delegata a un organismo centrale, InterNIC, che assegna gruppi di indirizzi direttamente ai siti che vogliono collegarsia Internet o ai provider, che li ridistribuiscono ai propri clienti.

Tutte le università e le ditte collegate a Internet hanno almeno uno di questi indirizzi, generalmente non assegnato direttamente da InterNIC ma ottenuto da un Internet Service Provider (ISP.)

Per configurare la rete privata di casa è sufficiente “costruirsi” i propri indirizzi IP privati come spiegato nel seguito. Per collegare le proprie macchine a Internet, però, è necessario ottenere un “vero” indirizzo IP dal proprio amministratore di rete o dal provider.

Per la scrittura degli indirizzi IP viene usata prevalentemente una notazione detta *dotted quad* (quadrupla puntata), nella quale i quattro byte dell’indirizzo vengono scritti uno alla volta (a partire dal più significativo) separandoli con un punto. Per esempio 132.199.15.99 è un indirizzo valido. Un indirizzo può anche essere scritto utilizzando una parola da 32 bit in esadecimale, per esempio: 0x84c70f63. Questo formato non è così intuitivo come il precedente, ma a volte può rivelarsi utile come si vedrà nelle sezioni seguenti.

Assegnare una rete a un host significa semplicemente impostare a determinati valori alcuni dei 32 bit dell’indirizzo dell’host. Questi bit, usati per identificare la rete, vengono detti “bit di rete”. I restanti bit servono per identificare l’host all’interno della rete, e vengono pertanto detti “bit dell’host”.

Nell’esempio precedente, l’indirizzo di rete è 132.199.0.0 (in un indirizzo di rete i bit dell’host sono messi a zero) mentre l’indirizzo dell’host nella rete è 15.99.

Come si fa a sapere che l’indirizzo dell’host occupa 16 bit? Semplice, viene indicato dal provider da cui si ottiene l’indirizzo stesso. Nell’instradamento senza classi tra domini (Classless Inter-Domain Routing, CIDR) utilizzato oggi, il numero di bit riservati all’host va da 2 a 16 e il numero di bit riservato alla rete viene scritto dopo l’indirizzo IP, separato da un “/”. Per esempio, 132.199.0.0/16 indica che la rete in questione ha 16 bit. Quando ci si riferisce alla “dimensione” di una rete lo si fa solitamente semplicemente con “/16”, “/24”, ecc.

Prima dell’avvento del CIDR le reti venivano divise in quattro classi. Ogni classe iniziava con una ben precisa sequenza di bit che la identificava. In dettaglio:

- La *Classe A* inizia con “0” come bit più significativo. I successivi 7 bit di un indirizzo di classe A identificano la rete e i restanti 24 bit vengono usati per indirizzare l’host. Pertanto, in una rete di classe A ci possono essere 2^{24} host. Non è facile, tuttavia, per una ditta o per un’università ottenere un intero indirizzo di classe A.

  La notazione CIDR per una rete di classe A con i suoi 8 bit di rete è “/8”.

- La *Classe B* inizia con “10” come bit più significativi. I successivi 14 bit identificano la rete e i restanti 16 bit servono per indirizzare gli host (oltre 65000). Gli indirizzi di classe B, una volta molto comuni per ditte e università, sono oggi difficili da ottenere a causa dell’attuale scarsità di indirizzi IPv4 disponibili.

  La notazione CIDR per una rete di classe A con i suoi 16 bit di rete è “/16”.

  Ritornando all’esempio precedente, si può vedere che l’host 132.199.15.99 (o l’equivalente esadecimale 0x84c70f63, che qui è più comodo) si trova in una rete di classe B, poiché 0x84 = **10**00... (in binario).

  Pertanto, l’indirizzo 132.199.15.99 può essere diviso nella parte di rete che è 132.199.0.0 e nella parte host che è 15.99.

- La *Classe C* viene identificata dai bit più significativi (detti anche MSB, Most Significant Bits) uguali a “110”, consentendo solo 256 host su ognuna delle 2^{21} possibili reti di classe C (in realtà, come si verdrà dopo, il numero di host possibili è 254). Gli indirizzi di classe C si trovano solitamente in piccole ditte.

  La notazione CIDR per una rete di classe C con i suoi 24 bit di rete è “/24”.

- Esiste anche un’ulteriore serie di indirizzi, che iniziano con “111”, che vengono usati per altri scopi (per esempio multicast) e non che interessano in questa sede.

Si noti che i bit utilizzati per identificare la rete fanno comunque parte dell’indirizzo di rete stesso.

Quando si separa la parte di rete da quella di host è comodo utilizzare la cosidetta *netmask* (maschera di rete). Si tratta di una valore da usare come “maschera” in cui tutti i bit di rete sono messi a “1” e tutti i bit di host sono a “0”. Mettendo insieme con un AND logico l’indirizzo con la netmask si ottiene l’indirizzo della rete.

Rifacendoci all’esempio precedente, 255.255.0.0 è una possibile netmask di 132.199.15.99. Quando si applica la netmask all’indirizzo, rimane la parte di rete 132.199.0.0.

Per gli indirizzi in notazione CIDR, il numero indicato di bit di rete specifica anche quali dei bit più significativi devono essere messi a “1” per ottenere la netmask della rete corrispondente. Per gli indirizzi di rete di classe A/B/C esiste una netmask di default:

- Class A (/8): netmask di default: 255.0.0.0, primo byte dell’indirizzo: 1-127
- Class B (/16): netmask di default: 255.255.0.0, primo byte dell’indirizzo: 128-191
- Class C (/24): netmask di default: 255.255.255.0, primo byte dell’indirizzo: 192-223

Un tipo particolare di indirizzo che è bene conoscere è l’“indirizzo di broadcast”. I messaggi inviati a questo indirizzo vengono ricevuti da *tutti* gli host della rete corrispondente. L’indirizzo di broadcast è caratterizzato dal fatto di avere tutti i bit di host messi a “1”.

Per esempio, dato l’indirizzo 132.199.15.99 con la maschera di rete (netmask) 255.255.0.0, l’indirizzo di broadcast è 132.199.255.255.

A questo punto ci si può chiedere: si può utilizzare un indirizzo di host con tutti i bit a “0” o a “1”? La risposta è no, perché il primo è l’indirizzo della rete e il secondo l’indirizzo di broadcast, e questi due indirizzi devono sempre essere presenti. Adesso si può capire perché una rete di classe B può contenere al massimo 2^{16}-2 host e una rete di classe C ne può contenere 2^{8}-2 = 254.

Oltre ai vari tipi di indirizzi già citati, ce n'è un altro che è speciale; è l'indirizzo IP 127.0.0.1 che si riferisce sempre all'host locale (*localhost*.) Ciò significa che quando si "parla" con 127.0.0.1 si comunica in realtà con se stessi, senza dare luogo a alcuna attività di rete, il che può essere utile per utilizzare servizi installati sulla propria macchina o per fare simulazioni e test quando non ci sono altri host sulla rete.

Per riassumere quanto discusso finora:

Indirizzo IP

indirizzo di 32 bit, comprendente la parte di rete e la parte di host.

Indirizzo di rete

indirizzo IP con tutti i bit di host messi a "0".

Netmask (maschera di rete)

maschera di 32 bit in cui i bit che si riferiscono alla rete sono messi a "1" mentre quelli che si riferiscono all'host sono a "0".

Indirizzo di broadcast

indirizzo IP con tutti i bit di host messi a "1".

Localhost

l'indirizzo IP dell'host locale è sempre 127.0.0.1.

9.1.5. Subnetting e routing

La precedente sezione ha descritto in dettaglio le maschere di rete e gli indirizzi di host e di rete. Il discorso sulle reti, però, è tutt'altro che concluso.

Consideriamo la situazione di una tipica università, con un indirizzo di classe B (/16), che le consente di avere fino a 2^{16} ~= 65534 host sulla rete. Anche se sarebbe comodo pensare di avere tutti questi host sulla stessa rete, ciò non si può realizzare a causa delle limitazioni fisiche dell'hardware in uso oggi.

Per esempio, la massima lunghezza di un cavo Ethernet thinwire è di 185 metri. Anche mettendo dei ripetitori che amplificano il segnale tra le postazioni, questo non basta per raggiungere tutte le postazioni in cui si trovano le macchine. Inoltre, il massimo numero di host su un cavo Ethernet thinwire è di 1024 e, avvicinandosi a questo limite, si ha una perdita di prestazioni.

Pertanto ci troviamo in una situazione in cui si abbiamo un indirizzo che ci consentirebbe di avere 60000 host ma siamo limitati dai collegamenti, che ci permettono di raggiungere un numero molto inferiore di postazioni.

Naturalmente la soluzione c'è, e consiste nel suddividere la "rete" di classe B in reti più piccole, solitamente dette *sottoreti*. Queste sottoreti possono ospitare, per esempio, fino a 254 host l'una (cioè abbiamo suddiviso la grande rete di classe B in più sottoreti di classe C).

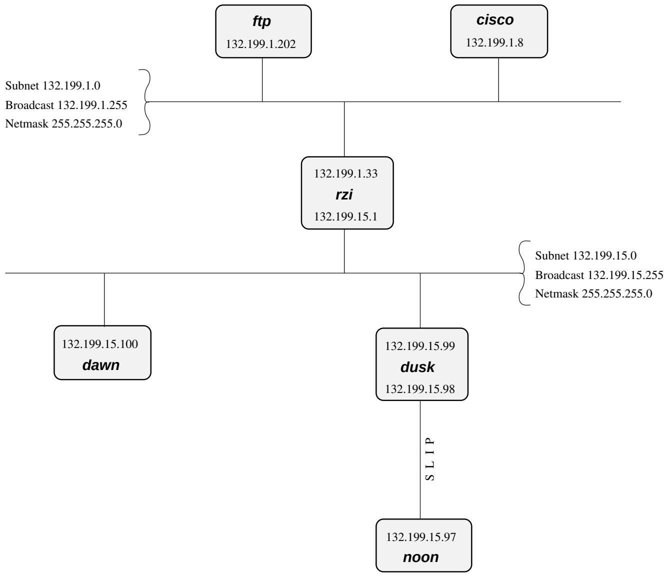
Per ottenere questo risultato, bisogna modificare la maschera di rete per avere più bit di rete e meno bit di host. Questo generalmente si fa lavorando sui byte della maschera (anche se è possibile scendere a livello dei singoli bit). Quindi la cosa più semplice è trasformare la maschera di rete da 255.255.0.0 (classe B) a 255.255.255.0 (classe C).

Usando la notazione CIDR, si scrive ora "/24" anziché il precedente "/16" per indicare che vengono usati 24 bit dell'indirizzo per identificare la rete e la sottorete, invece dei 16 usati prima.

Questa modifica ci consente di avere un byte di rete in più da assegnare a ognuna delle reti fisiche. Tutti i 254 host di ogni rete possono comunicare tra loro direttamente e si possono creare 256 reti di questo tipo (classe C). questa nuova configurazione dovrebbe essere adeguata alle esigenze della nostra ipotetica università.

Per chiarire meglio il concetto, continuiamo con l'esempio precedente. Supponiamo che il nostro host sia 132.199.15.99, che si chiami dusk e che abbia la netmask 255.255.255.0, il che significa che si trova sulla sottorete 132.199.15.0/24. Naturalmente vi sono anche altri host, come si vede dalla Figura 9-1.

Figura 9-1. Rete di esempio



Nella rete dell'esempio, dusk può parlare direttamente con dawn, poiché entrambi si trovano sulla stessa sottorete. Ci sono anche altri host sulla sottorete 132.199.15.0/24, ma per ora non li consideriamo.

Cosa succede se dusk vuole parlare con un host che si trova su un'altra sottorete? In questo caso il traffico uno o più gateway (router), attaccati alle due sottoreti. Per questo motivo un gateway ha sempre due indirizzi diversi, uno per ognuna delle sottoreti a cui è collegato. Dal punto di vista funzionale il router è del tutto trasparente per gli host, cioè non è necessario conoscerne l'indirizzo per raggiungere gli host che stanno dall'"altra parte". È sufficiente indirizzare direttamente gli host e i pacchetti giungeranno a destinazione correttamente.

Per esempio, supponiamo che dusk voglia scaricare dei file dal server FTP locale. Poiché dusk non può raggiungere ftp direttamente (perché si trova su una sottorete diversa) tutti i suoi pacchetti saranno inoltrati al suo "router di default" rzi (132.199.15.1), che sa dove deve inviarli perché giungano a destinazione.

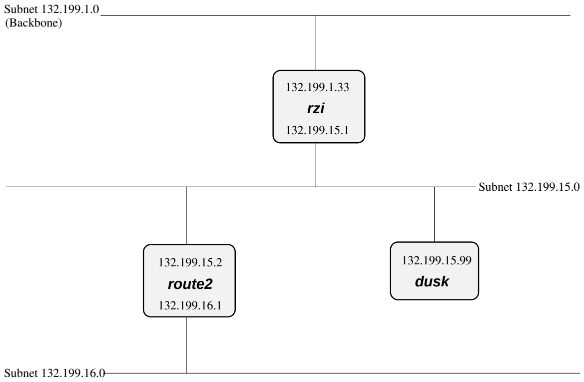
Dusk conosce l’indirizzo del router di default della sua rete (rzi, 132.199.15.1), e inoltra a lui tutti i pacchetti che non sono destinati alla stessa sottorete, cioè, in questo caso, tutti i pacchetti IP che hanno il terzo byte dell’indirizzo diverso da 15.

Il router di default invia i pacchetti all’host appropriato, visto che si trova anche sulla sottorete del server FTP.

In questo esempio *tutti* i pacchetti vengono inviati alla rete 132.199.1.0/24, semplicemente perché si tratta della *backbone* della rete, la parte più importante della rete, che trasporta tutto il traffico di passaggio tra varie sottoreti. Quasi tutte le sottoreti oltre a 132.199.15.0/24 sono collegate alla backbone in modo simile.

Che cosa succede se colleghiamo un’altra sottorete a 132.199.15.0/24 invece di 132.199.1.0/24? Probabailmente qualcosa di simile alla situazione descritta nella Figura 9-2.

Figura 9-2. Collegare una sottorete a un’altra sottorete



Adesso, per raggiungere un host che si trova sulla sottorete 132.199.16.0/24 da dusk, non si possono inoltrare i pacchetti a rzi, ma bisogna inviarli diretamemente a route2 (132.199.15.2). Dusk deve sapere che deve inoltrare questi pacchetti a route2 e inviare tutti gli altri a rzi.

Quando si configura dusk, gli si indica di inviare tutti i pacchetti per la sottorete 132.199.16.0/24 a route2, e tutti gli altri a rzi. Invece di esplicitare questo default con 132.199.1.0/24, 132.199.2.0/24, ecc., si può usare il valore 0.0.0.0 per impostare la route di default.

Ritornando alla Figura 9-1, si verifica un problema simile quando dawn vuole inviare a noon, che è collegato a dusk via linea seriale. Dall’esame del suo indirizzo IP, noon sembrerebbe essere attaccato alla rete 132.199.15.0, ma non lo è realmente. Invece, dusk viene usato come gateway e dawn deve inviare i suoi pacchetti a dusk, che li inoltrerà a sua volta a noon. Il sistema grazie al quale dusk viene forzato a accettare pacchetti che non sono destinati a lui ma a un altro host (noon) viene definito “proxy arp”.

Lo stesso discorso vale quando un host di un’altra sottorete vuole inviare a noon. Dovrà inviare i suoi pacchetti a dusk (eventualmente inoltrati da rzi).

9.1.6. Concetti di Name Service

Nelle sezioni precedenti, nel fare riferimento agli host abbiamo spesso utilizzato il loro indirizzo IP. questo era necessario per spiegare le differenze fra i vari tipi di indirizzi. In generale, tuttavia, quando si parla di host è più comodo utilizzare il loro “nome”, come abbiamo fatto nella sezione sul routing.

Alla maggior parte delle applicazioni non importa se gli host sono specificati usando il nome o l’indirizzo IP. Tuttavia, internamente esse usano gli indirizzi IP, ed esistono vari metodi che possono utilizzare per convertire i nomi degli host in indirizzi IP. In questa sezione verranno introdotti i concetti che stanno alla base di ognuno di questi metodi. Successivamente verrà spiegato come effettuarne la configurazione.

La conversione dei nomi degli host (e dei domini) in indirizzi IP, viene effettuata da un modulo software detto *resolver*. Non si tratta di un servizio aggiuntivo ma di una serie di funzioni di libreria che vengono collegate a ogni applicazione che usa funzioni di rete. Il resolver cerca di “risolvere” (da cui il nome) i nomi degli host in indirizzi IP. Si veda [RFC1034] e [RFC1035] per ulteriori dettagli sul resolver.

I nomi degli host hanno generlamente una lunghezza massima di 10 caratteri e possono contenere lettere, numeri e trattini (“-”). Maiuscole e minuscole sono considerate equivalenti.

Proprio come per le reti e le sottoreti, è possibile (anzi desiderabile) raggruppare gli host in domini e sottodomini. Quando si ottiene un indirizzo di rete dal provider, generalmente si ottiene anche un nome di dominio. Come per le sottoreti, è lasciato all’utente il compito di creare sottodomini. I sottodomini non sono in relazione diretta con le sottoreti; per esempio, un dominio può contenere host collegati a diverse sottoreti.

Figura 9-1 mostra le sottoreti 132.199.1.0/24 e 132.199.15.0/24 (e le altre) che fanno parte del sottodominio rz.uni-regensburg.de. Il dominio che l’università di Regensburg ha ottenuto dal suo provider IP è “uni-regensburg.de” (“.de” indica la Germania, Deutschland), il sottodominio “rz” indica il Rechenzentrum (Centro di Calcolo).

I nomi degli host, dei sottodomini e dei domini sono separati da punti (“.”). È anche possibile, sebbene non comune, usare più di un livello di sottodominio; per esempio: fox_in.socs.uts.edu.au.

Un nome di host che comprende il (sotto)dominio viene detto nome di dominio completamente qualificato (FQDN, Fully Qualified Domain Name). Per esempio, l’indirizzo IP 132.199.15.99 appartiene all’host il cui FQDN è dusk.rz.uni-regensburg.de.

In precedenza è stato detto che l’indirizzo IP 127.0.0.1 appartiene sempre all’host locale, qualunque sia il “vero” indirizzo IP dell’host stesso. Pertanto, 127.0.0.1 è sempre mappato al nome localhost.

I tre metodi diversi di tradurre un nome di host in un indirizzo IP sono: `/etc/hosts`, il Domain Name Service (DNS) e il Network Information Service (NIS).

9.1.6.1. `/etc/hosts`

Il primo e il più semplice dei tre metodi per tradurre da nome host a indirizzo IP consiste nell’usare una tabella che mette in corrispondenza i nomi e i relativi indirizzi. Questa tabella si trova nel file `/etc/hosts` e ha il formato seguente:

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

Le linee che iniziano con un cancelletto ("`#`") sono considerate commenti e vengono ignorate. Le altre linee contengono un indirizzo IP e il/i corrispondenti nomi di host.

Non è possibile che un nome di host corrisponda a più indirizzi IP. rzi, per esempio, ha due nomi diversi per i suoi due indirizzi: rzi e rzia.

Può essere comodo dare più nomi (alias) a un host; ciò consente di specificare un nome alternativo per l'host in relazione a un particolare servizio, come viene comunemente fatto, per esempio, per i server FTP. Il primo nome (quello più a sinistra) è solitamente il nome reale (canonico) dell'host.

Può anche essere utile usare il nome completo (FQDN) dell'host come nome canonico, e il suo hostname (senza dominio) come alias.

Importante: ci deve *sempre* essere una voce per il mapping di localhost su 127.0.0.1!

9.1.6.2. Il Domain Name Service (DNS)

L'uso di `/etc/hosts` comporta un problema, specialmente nelle reti di grandi dimensioni: quando si aggiunge un host o si cambia l'indirizzo di un host, tutti i file `/etc/hosts` di tutte le macchine devono essere aggiornati. Non solo questo comporta una perdita di tempo, ma è anche una possibile causa di errori e inconsistenze che inevitabilmente porteranno a dei problemi.

Un'altro sistema è di mantenere una sola tabella di nomi di host (database) per l'intera rete e di fare in modo che tutti i client interroghino quel "nameserver". Gli aggiornamenti saranno effettuati solo sul name server.

Questa è l'idea di base su cui si fonda il *Domain Name Service* (DNS).

Solitamente c'è un name server per ogni dominio (da cui il nome Domain Name Service) e ogni host nel dominio sa a quale dominio appartiene e quale name server deve interrogare per il proprio dominio.

Quando il DNS riceve un'interrogazione per un host che non si trova nel suo dominio, la inoltra al DNS del dominio corrispondente oppure a un DNS che sa a quale DNS chiedere l'informazione. Quest'ultimo inoltra l'interrogazione a un DNS di livello superiore, e così via. Questo processo non va avanti all'infinito: vi sono vari server "root" che posseggono le informazioni di tutti i domini.

Vedere Capitolo 11 per ulteriori dettagli sul DNS.

9.1.6.3. Il Network Information Service (NIS/YP)

Le Yellow Pages (YP) sono state inventate dalla Sun Microsystems. Il nome è stato poi cambiato in Network Information Service (NIS) perchè YP era già un marchio della British Telecom.

Nei sistemi Unix sono solitamente presenti un gran numero di file di configurazione ed è pertanto spesso desiderabile di limitare la manutenzione di questi file a un paio di host. Gli host vengono quindi raggruppati in domini NIS (che non hanno niente a che vedere con i domini DNS) e fanno solitamente parte di un cluster di workstation.

Alcuni dei file di configurazione condivisi tra questi host sono `/etc/passwd`, `/etc/group` e `/etc/hosts`.

È quindi possibile "abusare" del NIS per ottenere un'unica traduzione da nome a indirizzo su tutti gli host di un dominio NIS.

C'è solo uno svantaggio che impedisce a NIS di essere effettivamente utilizzato per questo tipo di traduzione: a differenza del DNS, NIS non fornisce alcun metodo per risolvere i nomi degli host che non fanno parte della tabella degli host. Non esistono host di livello superiore che il server NIS può interrogare e, pertanto, la traduzione è destinata a fallire. NIS+, sempre della Sun, affronta questo problema, ma poiché NIS+ è disponibile solo sui sistemi Solaris non ci è di grande aiuto.

Quanto detto non significa che NIS non sia un metodo eccellente per gestire, per esempio, le informazioni sugli utenti (`/etc/passwd`, ...) in cluster di workstation; semplicemente nonè molto utile per risolvere i nomi degli host.

9.1.6.4. Altri metodi

I metodi fin qui descritti per la risoluzione dei nomi degli host in indirizzi IP, sono quelli oggi di uso più comune, ma non sono gli unici. In linea di massima si può utilizzare un database qualsiasi, ma non vi sono implementazioni di questo tipo in NetBSD.

NIS+, come già detto, cerca di rimediare alla mancanza di gerarchia nelle strutture dati NIS. Le tabelle possono essere impostate in modo che se un'interrogazione non può essere soddisfatta dal server di un dominio, ci può essere un altro dominio di livello superiore in grado di dare una risposta. Per esempio, si può definire un dominio che elenca tutti gli host (utenti, gruppi, ecc.) appartenenti a tutta una ditta, un altro che definisce gli stessi dati per ogni divisione, e così via. NIS+ non è correntemente molto usato, persino la stessa Sun ha ripreso a fornire NIS di default.

In passato, lo standard X.500 era stato progettato per gestire sia database semplici, come `/etc/hosts`, sia strutture gerarchiche complesse come, per esempio, l'attuale DNS. X.500 non ha mai avuto molto successo, in parte perché cercava di fare troppe cose contemporaneamente. Una versione ridotta, chiamata LDAP (Lightweight Directory Access Protocol) è oggi disponibile e la sua popolarità sta crescendo negli ultimi anni per gestire i dati degli utenti (ma anche degli host) in organizzazioni di piccole e medie dimensioni.

9.1.7. IPv6, il protocollo Internet di nuova generazione

9.1.7.1. Il futuro di Internet

A giudizio degli esperti, Internet nella sua forma attuale si troverà a affrontare un problema serio nel giro di pochi anni. A causa della sua rapida crescita e delle limitazioni nella sua progettazione, si arriverà a un punto in cui non saranno più disponibili indirizzi per connettere nuovi host. In quel momento, non si potranno più aggiungere nuovi server web, gli utenti non potranno ottenere account dagli ISP, non si potranno collegare nuove macchine in rete per accedere al web o partecipare ai giochi online - in breve, un problema molto serio.

Sono stati pensati vari approcci per risolvere questo problema. Un metodo molto diffuso consiste nel non assegnare un indirizzo valido univoco a ogni macchina collegata, ma di assegnare indirizzi "privati" e di "nascondere" più macchine dietro a un unico indirizzo ufficiale globale. Questo metodo, che si chiama Network Address Translation (NAT, noto anche come IP Masquerading) presenta dei problemi, perché le macchine nascoste dietro l'unico indirizzo globale non possono essere indirizzate direttamente e, pertanto, non è possibile aprire connessioni con loro, come richiesto, per esempio, dai giochi online, dalle reti peer-to-peer, ecc. Per una discussione sugli svantaggi di NAT, vedere [RFC3027].

Un approccio diverso al problema della scarsità di indirizzi internet è di abbandonare il vecchio protocollo Internet e le sue limitate capacità di indirizzamento in favore di un nuovo protocollo che non soffre di queste limitazioni. Il protocollo, o meglio l'insieme di protocolli, usato dalle macchine connesse a formare l'attuale Internet è noto come suite TCP/IP (Transmission Control Protocol, Internet Protocol), e la versione 4, quella in uso adesso, ha tutti i problemi appena descritti. Per poter passare a una nuova versione del protocollo che risolve questi problemi bisogna ovviamente che tale versione “migliore” esista. E infatti, la versione 6 dell’Internet Protocol (IPv6) esiste, e risponde a tutte le possibili future esigenze di spazio di indirizzamento, affrontando anche altri problemi quali la segretezza, la crittografia e un miglior supporto al mobile computing.

Questa sezione introduce il protocollo IPv6; per una buona comprensione si richiede una conoscenza di base del funzionamento del protocollo IPv4. Vengono descritti i cambiamenti nel formato dell’indirizzo e nella risoluzione dei nomi. Sfruttando queste informazioni, la Sezione 10.2.4 mostrerà come utilizzare IPv6 anche se non viene offerto dall’ISP utilizzando un meccanismo di transizione semplice ma efficace denominame *6to4*. Lo scopo finale è di ritrovarsi online con IPv6, mostrando una configurazione di esempio per NetBSD.

9.1.7.2. Perché passare a IPv6?

Quando si consiglia a qualcuno di passare da IPv4 a IPv6, la prima domanda che ci si sente rivolgere è “perché”? Vi sono varie buone ragioni:

- Spazio di indirizzamento più grande.
- Supporto al “mobile computing”.
- Gestione integrata della sicurezza.

9.1.7.2.1. Maggiore spazio di indirizzamento

L’ampliamento dello spazio di indirizzamento offerto è il primo dei vantaggi offerti da IPv6 rispetto a IPv4. L’attuale architettura di Internet è basata su indirizzi di 32 bit, mentre la nuova versione dispone di indirizzi di 128 bit. Grazie a questo ampliamento, non sarà più necessario utilizzare soluzioni quali NAT e ciò porterà a una completa e illimitata connettività per tutte le attuali macchine basatae su IP nonché per i nuovi dispositivi mobili quali PDA e telefoni cellulari, che potranno disporre del completo accesso IP grazie a GPRS e UMTS.

9.1.7.2.2. Mobilità

Quando si parla di dispositivi mobili e IP, un punto importante da tenere presente è che è necessario un protocollo speciale per supportare la mobilità; l’implementazione di questo protocollo, detto *Mobile IP*, è uno dei requisiti di ogni stack IPv6. Pertanto, quando si attiva IPv6 si ottiene il supporto per il roaming tra reti diverse, con l’aggiornamento completo quando si lascia una rete e ci si trasferisce su un’altra. Il supporto al *roaming* è possibile anche con IPv4, ma bisogna risolvere alcuni problemi prima di ottenere un sistema funzionante. Con IPv6 non è necessario alcun accorgimento particolare, poiché il supporto alla mobilità è uno dei requisiti di progetto di IPv6. Vedere [RFC3024] per ulteriori informazioni sui problemi da risolvere per supportare il “mobile IP” su IPv4.

9.1.7.2.3. Sicurezza

Oltre al supporto per la mobilità, la sicurezza era uno dei requisiti del successore all’attuale versione dell’Internet Protocol. Come conseguenza, gli stack del protocollo IPv6 devono includere *IPsec*. IPsec consente di autenticare, crittografare e comprimere qualsiasi tipo di traffico IP. A differenza dei protocolli a livello di applicazione, quali SSL o SSH, si può gestire tutto il traffico IP tra due nodi senza dover apportare modifiche alle applicazioni stesse. Il vantaggio di qeusto approccio è che tutte le applicazioni su una macchina possono beneficiare della crittografia e dell'autenticazione, e che le relative politiche possono essere impostate in modo generale a livello di host o anche di rete, e non semplicemente a livello di applicazione/servizio. Un'introduzione a IPsec comprendente anche i collegamenti alla documentazione si può trovare in [RFC2411], mentre il protocollo di base è descritto in [RFC2401].

9.1.7.3. Modifiche a IPv4

Ora che abbiamo visto le più importanti caratteristiche di IPv6 possiamo passare a esaminare le basi del protocollo. Per una migliore comprensione è preferibile una conoscenza elementare di IPv4, rispetto al quale saranno descritte le differenze. Inizieremo con gli indirizzi IPv6, spiegando come sono strutturati, quali sono i tipi di indirizzo e cosa ne è stato dei broadcast, e poi, dopo aver discusso lo “strato” IP, passerememo ai cambiamenti relativi alla risoluzione dei nomi e alle novità di IPv6 per il DNS.

9.1.7.3.1. Indirizzamento

Un indirizzo IPv4 è un valore a 32 bit, scritto solitamente nel formato denominato “dotted quad”, dove ognuno dei valori separati dai punti è un valore compreso tra 0 e 255. Per esempio:

```
127.0.0.1
```

Questa struttura consente a un numero massimo teorico di 2^{32} o ~4 miliardi di host di essere collegati all'attuale Internet. A causa dei raggruppamenti, però, non tutti gli indirizzi sono realmente disponibili.

Gli indirizzi IPv6 usano 128 bit, il che ci porta a 2^{128} host teoricamente indirizzabili. Questo consente di indirizzare un numero davvero enorme di macchine, coprendo completamente senza problemi tutti i requisiti attuali e futuri anche di PDA e telefoni cellulari dotati di indirizzo IP. Per scrivere gli indirizzi IP, li si divide in gruppi di 16 bit scritti con quattro cifre esadecimali, separando i gruppi con il carattere “:” (due punti). Per esempio:

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

Questo esempio mostra anche che una serie di zeri consecutivi può essere abbreviato con un unico “::” nel'indirizzo IPv6. L'indirizzo precedente è quindi equivalente a fe80:0:00:000:2a0:d2ff:fea5:e9f5. Gli zeri iniziali all'interno dei gruppi possono essere omessi.

Per semplificarne la gestione, gli indirizzi vengono divisi in due parti: i bit che identificano la rete di appartenenza della macchina e i bit che identificano la macchina stessa sulla (sotto)rete. I primi vengono definiti “bit di rete” (*netbits*) e i secondi “bit di host” (*hostbits*) e, sia in IPv4 sia in IPv6, i bit di rete sono quelli più a sinistra (bit più significativi) mentre i bit di host sono quelli più a destra (bit meno significativi), come mostrato nella Figura 9-3.

Figura 9-3. Indirizzi: bit di rete e bit di host.

n netbits	128−n hostbits

In IPv4, i bit di rete e i bit di host vengono separati grazie alla maschera di rete. Esempi tipici sono 255.255.0.0, che usa 16 bit per l'indirizzo di rete, e 255.255.255.0, che usa altri 8 bit, consentendo, per esempio, di indirizzare 256 sottoreti su una rete di classe B.

Quando si è passati dall'indirizzamento per classi al formato CIDR, la separazione tra bit di rete e bit di host ha smesso di cadere necessariamente sul limite degli 8 bit e, in conseguenza, le maschere di rete hanno incominciato a essere più difficili da gestire. Al posto delle maschere si è quindi cominciato a indicare il limite usarndo il numero di bit di rete per un dato indirizzo; per esempio:

```
10.0.0.0/24
```

corrisponde a una maschera di rete di 255.255.255.0 (24 bit a 1). Lo stesso schema è usato in IPv6. Il valore:

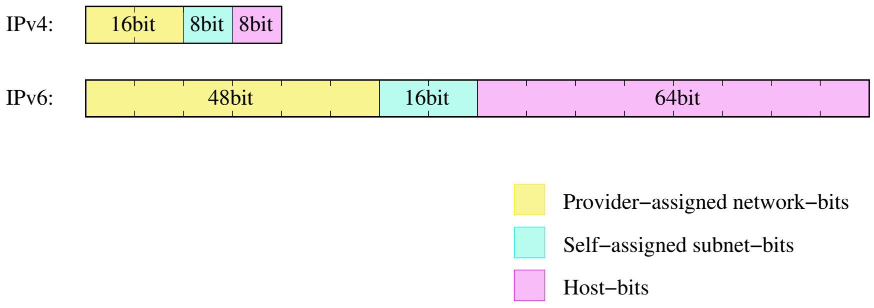
```
2001:638:a01:2::/64
```

indica che nell'indirizzo specificato i primi 64 bit (quelli più a sinistra) riguardano la rete mentre gli ultimi 64 bit (quelli più a destra) identificano la macchina sulla rete. I bit di rete vengono spesso chiamati "prefisso" (di rete) e la lunghezza del prefisso è qui di 64 bit.

Tra i più comuni schemi di indirizzamento di IPv4, c'erano le reti di classe B e di classe C. Con una rete di classe C (/24), il provider assegna 24 bit, lasciandone 8 disponibili. Se si aggiungono delle sottoreti si finisce con l'avere una maschera di rete "irregolare" non molto pratica da gestire. In questo caso una rete di classe B (/16) sarebbe più facile da gestire perché dal provider vengono fissati solo 16 bit, il che consente di creare sottoreti con maggiore facilità dividendo in due parti i restanti 16 bit: la prima parte indirizza la sottorete locale e la seconda indirizza l'host all'interno della sottorete stessa. Solitamente si "spezza" sul confine degli otto bit (un byte); usando una maschera di rete di 255.255.255.0 (/24) si riescono a gestire in modo flessibile anche reti di grandi dimensioni, fermo restando il limite di 256 sottoreti di 254 macchine l'una.

Con i 128 bit disponibili per l'indirizzamento in IPv6, lo schema utilizzato è lo stesso, solamente con i campi più grossi. I provider solitamente assegnano reti /48, il che lascia 16 bit per le sottoreti e 64 bit per gli host.

Figura 9-4. Gli indirizzi IPv6 hanno una struttura simile agli indirizzi di classe B



Mentre lo spazio per le reti è le sottoreti sembra adeguato, l'uso di 64 bit per gli host appare uno spreco di spazio. Visto che è improbabile utilizzare svariati miliardi di host sulla stessa sottorete, perché isare questa struttura?

L'idea alla base di un indentificativo di host di lunghezza fissa di 64 bit, è che gli identificativi di host non vengono più assegnati manualmente come viene oggi fatto con IPv4. Invece, si raccomanda (è solo una raccomandazione, non un obbligo) di costruirli partendo dai cosiddetti indirizzi EUI64. Gli indirizzi EUI64 occupano, come dice il nome, 64 bit e derivano dall'indirizzo MAC della scheda di rete. Per esempio, per Ethernet, all'indirizzo di rete di 6 byte (48 bit) vengono solitamente solitamente aggiunti i bit “fffe” nel mezzo e un bit viene impostato per indicare che l’indirizzo è unico (il che è vero per Ethernet). Per esempio, l’indirizzo MAC

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

diventa l’indirizzo EUI64

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

il che porta i bit di host dell’indirizzo IPv6 a avere la forma

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

Questi bit di host possono essere usati per assegnare l’indirizzo IPv6 a un host, supportando l’autoconfigurazione degli host IPv6: tutto ciò che serve per avere un indirizzo IPv6 completo sono i primi bit (rete/sottorete), ma IPv6 offre anche il sistema per assegnarli automaticamente.

Sulle reti di macchine IP c’è solitamente un router che agisce da gateway verso le reti esterne. Nelle reti IPv6, il router invia una serie di informazioni di “router advertisement”, che i client dovrebbero ricevere mentre sono in funzione oppure richiedere esplicitamente mentre sono in fase di avvio. Queste informazioni comprendono l’indirizzo del router e i prefissi di rete gestiti dal router stesso. Con queste informazioni e l’indirizzo EUI64 generato automaticamente, un host IPv6 può determinare il proprio indirizzo IP senza bisogno di assegnazione manuale. Naturalmente i router devono ancora essere configurati esplicitamente.

Le informazioni inviate dal router fanno parte del protocollo NDPNeighbor Discovery Protocol (vedere [RFC2461]), che è il successore del protocollo ARP dell’IPv4. A differenza di ARP, NDP non solo ricerca gli indirizzi IP corrispondenti agli indirizzi MAC, ma effettua anche un servizio simile per i router e i relativi prefissi di rete gestiti, il che viene utilizato per l’autoconfigurazione degli host IPv6 appena descritta.

9.1.7.3.2. Indirizzi multipli

Con IPv4, un host ha generalmente un indirizzo IP per ogni intrfaccia di rete o anche per macchina, se lo stack IP lo supporta. Solo alcune specifiche applicazioni, come i server web, richiedono macchine che hanno più di un indirizzo IP. Con IPv6 tutto ciò è diverso. Per ogni interfaccia, non solo c’è un indirizzo IP globale, ma ci sono anche altri due indirizzi interessanti: l’indirizzo “link local” e l’indirizzo “site local”. L’indirizzo link local ha prefisso fe80::/64 e i bit di host derivano dall’indirizzo EUI64; esso viene usato solo per contattare gli host e i router sulla stessa rete e non è visibile o raggiungibile da altre sottoreti. Se lo si desidera, si può scegliere tra usare indirizzi globali (assegnati dal provider) oppure indirizzi site local; questi ultimi hanno l’indirizzo di rete fec0::/10 e le sottoreti e gli host possono essere indirizzati esattamente come nel caso delle reti assegnate dal provider. L’unica differenza è che l’indirizzo non sarà visibile dalle macchine esterne perché si trovano su una rete diversa e i loro indirizzi site local, se esistono, sono su una rete fisica diversa. Per IPv6 è pratica comune assegnare agli host l’indirizzo globale e quello link local. Gli indirizzi site local stanno diventando sempre più rari anche perché per la connettività globale è richiesto un indirizzo globale unico.

9.1.7.3.3. Multicasting

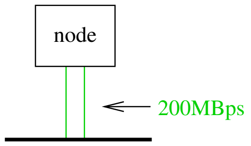
Nel dominio IP vi sono tre modi per raggiungere un host: *unicast*, *broadcast* e *multicast*. La forma più comune di colloquio sfrutta l'indirizzo unicast che, in IPv4, è il "normale" indirizzo IP assegnato a un host, comprendente i bit di rete. L'indirizzo di broadcast viene utilizzato per raggiungere tutti gli host sulla stessa sottorete IP e ha i bit di rete impostati all'indirizzo della rete e i bit di host tutti a "1". Gli indirizzi di tipo multicast servono per raggiungere gli host che fanno parte dello gruppo multicast, che possono trovarsi in un punto qualunque di Internet. Le macchine devono esplicitamente unirsi a un gruppo multicast pre poter partecipare, e vi sono indirizzi speciali IPv4 usati per il multicast, allocati sulla sottorete 224/8. Il multicast non viene molto utilizzato sotto IPv4; solo alcune applicazioni quali le utility MBone audio e video broadcast lo usano.

Gli indirizzi unicast IPv6 si usano allo stesso modo di quelli IPv4; nessun cambiamento, quindi - tutti i bit di rete e di host vengono assegnati per indentificare la rete e la macchina. Il broadcast non è più disponibile in IPv6 nel modo in cui lo era in IPv4: è qui che entra in gioco il multicasting. Gli indirizzi della rete ff::/8 sono riservati alle applicazioni multicast, e vi sono due speciali indirizzi mulicast che prendono il posto dell'indirizzo di broadcast di IPv4. Uno è l'indirizzo multicast "all routers" (tutti i router) e l'altro è "all hosts" (tutti gli host). Questi indirizzi sono specifici della sottorete, cioè un router collegato a due diverse sottoreti può indirizzare tutti gli host/router di una qualsiasi delle due sottoreti. In questo caso gli indirizzi sono:

- ff0*X*::1 per tutti gli host
- ff0*X*::2 per tutti i router

dove "*X*" è l'ID del collegamento, che identifica la rete e che, solitamente, inizia da "1" per l'ambito locale, "2" per il primo link, ecc. Si noti che è possibile che due interfacce di rete siano attaccate allo stesso link, risultando in un'ampiezza di rete doppia.

Figura 9-5. Più interfacce collegate a un link: un unico ID per il link



Un uso dell'indirizzo di multicast "all hosts" è nel codice NDP, quando una macchina che vuole comunicare con un'altra macchina invia una richiesta al gruppo all hosts e la macchina in questione deve rispondere.

9.1.7.3.4. La risoluzione dei nomi in IPv6

Dopo questa lunga introduzione all'indirizzamento sotto IPv6, è lecito sperare che vi sia un modo per continuare a usare dei nomi per riferirsi agli host, come si fa con IPv4, anziché snocciolare quei lunghi indirizzi IPv6. Naturalemente è possibile.

La traduzione da nome di host a indirizzo IP in IPv4 solitamente si fa in uno di questi tre modi: usando una semplice tabella in `/etc/hosts`, usando il Network Information Service (NIS, già YP) o con il Domain Name System (DNS).

Al momento, NIS/NIS+ su IPv6 è disponibile solo sotto Solaris 8, sia per il contenuto del database sia per il trasporto, usando una estensione RPC.

Una semplice mappa del tipo indirizzo<->nome, del tipo di `/etc/hosts` è supportata da tutti gli stack IPv6. Con l'implementazione KAME usata da NetBSD, il file `/etc/hosts` contiene sia gli indirizzi IPv4 sia quelli IPv6. A titolo di semplice esempio, vediamo la voce "localhost" nella versione di default per un'installazione di NetBSD.

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

Per il DNS non vengono introdotti fondamentalmente nuovi concetti. La risoluzione dei nomi IPv6 viene fatta con record di tipo AAAA che, come indica il nome, puntano a un'entità che è quattro volte più grossa di un record di tipo A. Un record AAAA ha un nome di host nella parte sinistra, proprio com per A, e l'indirizzo IPv6 nella parte destra. Per esempio:

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

Per la risoluzione inversa IPv4 usa la zona `in-addr.arpa`, alla quale aggiunge i byte (decimali) in ordine inverso, cioè con i byte più significativi più a destra. Com IPv6 il metodo è lo stesso, solo che vengono usate le cifre esadecimali (una ogni 4 bit) anziché i numeri decimali e i record delle risorse si trovano in un dominio differente, ip6.int.

Pertanto, per la risoluzione inversa dell'host precedente si deve mettere nel file `/etc/named.conf` una voce simile alla seguente:

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

e, nel file di zona `db.reverse`, oltre ai soliti record 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.
```

Qui l'indirizzo viene scritto una cifra esadecimale alla volta in ordine inverso, iniziando da quella meno significativa (quella più a destra) e separando le cifre con punti, come nei soliti file di zona.

Quando si imposta il DNS per IPv6 bisogna fare attenzione alla versione del software DNS che si utilizza. BIND 8.x capisce i record AAAA ma non offre la risoluzione dei nomi via IPv6; per questo è necessario utilizzare BIND 9.x che, tra l'altro, supporta alcuni tipi di record di risorsa ancora in fase di discussione e quindi non ufficialmente introdotti. Il più interessante è il record A6, che facilita il cambiamento di provider/prefisso.

Per riassumere, in questa sezione sono state discusse le differenze tecniche tra IPv4 e IPv6 per l'indirizzamento e la risoluzione dei nomi. Alcuni aspetti, quali le opzioni delle testate IP, QoS, ecc. non sono volutamente stati trattati per non appesantire troppo la materia, visto anche il carattere introduttivo di questa guida.

Bibliography

[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*.

Capitolo 10. Configurazione di rete

10.1. Pratica

10.1.1. Opzioni di configurazione del kernel

L'autore di questa sezione (Opzioni di configurazione del kernel) è Hubert Feyrer `<hubert@feyrer.de>`.

Prima di addentrarci nei meandri delle configurazioni di rete, è bene dare un'occhiata ai relativi parametri presenti nella configurazione del kernel. Per ulteriori dettagli sull'argomento si veda Capitolo 7; qui ci concentreremo solo sugli aspetti relativi al networking, prendendo il file di configurazione i386/GENERIC a titolo di esempio. I file di configurazione per le altre piattaforme contengono opzioni simili, generlmente accompagnate da commenti che ne chiariscono l'uso all'interno dei file stessi. Inoltre, le opzioni di compilazione del kernel sono documentate nella pagina di manuale options(4) e solitamente c'è una pagina di manuale per ogni driver (per esempio tlp(4)).

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

La prima linea del file di configurazione mostra il numero di versione che, in questo caso, è 1.354.2.15 e che può essere utile per confrontare con altre versioni via CVS o per riportare eventuali bug.

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

Se si vuole usare il Network Time Protocol (NTP), si può attivare questa opzione per ottenere la massima precisione. Se quest'opzione non è attiva, NTP funziona ugualmente. Vedere ntpd(8) per ulteriori dettagli.

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

Per utilizzare il disco fisso di un'altra macchina con il Network File System (NFS), è necessario attivare quest'opzione. Vedere la Sezione 10.2.2 per ulteriori informazioni.

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

Quest'opzione comprende il lato server del protocollo NFS e deve essere abilitata per consentire ad altre macchine di usare il disco fisso locale. Vedere la Sezione 10.2.2 per ulteriori informazioni.

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

Per impostare un router che inoltra i pacchetti tra più reti o più interfacce di rete, bisogna abilitare l'opzione GATEWAY. Non solo attiva l'inoltro dei pacchetti ma aumenta anche alcuni buffer. Vedere options(4) per ulteriori dettagli.

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

Attiva il supporto al TCP/IP nel kernel. Anche se non si intende usare il networking, quest'opzione è comunque necessaria per le comunicazioni interne alla macchina di sottosistemi quali X Window. Vedere inet(4) per i dettagli.

```
options         INET6           # IPV6
```

Chi vuole usare IPv6 deve attivare quest'opzione, che è stata introdotta già con la release 1.5 di NetBSD. Chi non vuole IPv6 può commentarla. Vedere la pagina di manuale inet6(4) e la Sezione 9.1.7 per ulteriori informazioni sul protocollo Internet di nuova generazione.

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

Include il supporto per il protocollo IPsec: autenticazione, compressione, chiavi, ecc. Quest'opzione si può utilizzare anche senza attivare IPv6 se si desidera usare IPsec solo con IPv4, il che è possibile. Vedere ipsec(4).

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

Quest'opzione si utilizza in aggiunta a IPSEC per attivare la crittografia IPsec.

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

Includere quest'opzione per attivare il routing di servizi multicast quali MBone. Si noti che il routing è poi controllato dal demone mrouted(8).

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

Queste opzioni abilitano la famiglia di protocolli Xerox Network Systems(tm). Non sono in relazione con lo stack del protocollo TCP/IP e non sono molto utilizzate oggi. La pagina di manuale ns(4) contiene alcuni dettagli.

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

Queste opzioni attivano lo stack del protocollo OSI che, dopo essere stato a lungo reputato il futuro del networking, oggi ha un interesse prevalentemente storico. Vedere la pagina di manuale iso(4).

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

Queste opzioni attivano il set di protocolli X.25 pre la trasmissione di dati su linea seriale. È, o meglio era, usata insieme ai protocolli OSI o nel networking WAN.

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

Include il supporto per lo stack del protocollo AppleTalk. Per poterlo usare sono necessari i programmi server che si possono trovare, per esempio, in `pkgsrc/net/netatalk` e `pkgsrc/net/netatalk-asun`. Ulteriori informazioni sul protocollo AppleTalk e sul relativo stack si possono trvare nella pagina di manuale 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)
```

Queste opzioni controllano vari aspetti del protocollo PPP (Point-to-Point Protocol). Le prime due determinano gli algoritmi di compressione usati/disponibili, mentre la terza abilita il codice per il filtraggio di alcuni pacchetti.

```
options         PFIL_HOOKS      # pfil(9) packet filter hooks
options         IPFILTER_LOG    # ipmon(8) log support
```

Queste opzioni abilitano il firewalling usando IPfilter. Vedere le pagine di manuale ipf(4) e ipf (8) per ulteriori informazioni sull'uso di IPfilter, nonché la Sezione 10.2.1 per un esempio di configurazione.

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

Quest'opzione è necessaria solo se sulla rete sono ancora presenti macchine che usano 4.2BSD o uno stack di rete derivato da questa versione. In presenza di macchine 4.2BSD bisogna fare attenzione alla corretta impostazione dell'indirizzo di broadcast, poiché 4.2BSD ha un bug nel codice di networking relativamente all'indirizzo di broadcast. Questo bug costringe a mettere tutti i bit di host a "0" nell'indirizzo di broadcast. L'opzione TCP_COMPAT_42 serve a questo.

```
options         NFS_BOOT_DHCP,NFS_BOOT_BOOTPARAM
```

Queste opzioni abilitano la ricerca di informazioni con DHCP o con il protocollo BOOTPARAM quando il kernel usa un filesystem "root" di tipo NFS. Vedere la pagina di manuale diskless(8) per ulteriori informazioni.

```
# 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
```

Queste linee indicano al kernel dove cercare il proprio filesystem "root" e di quale tipo di filesystem si tratti. Se, per esempio, si vuole creare un kernel che usa un filesystem root di tipo NFS tramite l'interfaccia tlp0, lo si può fare con la direttiva "`root on tlp0 type nfs`". Se si usa `?` al posto del tipo di dispositivo, il kernel cerca di determinare il 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
```

Per usare PPP o SLIP è necessario avere un'interfaccia seriale (com). Va bene anche una che si attacchi a USB, PCMCIA o PUC.

```
# Network Interfaces
```

La lista delle interfacce di rete è piuttosto lunga e contiene dispositivi di ogni tipo; bisogna semplicemente selezionare quello che corrisponde al proprio hardware, basandosi sui commenti presenti nel file. La maggior parte dei driver ha una propria pagina di manuale; per esempio tlp(4), ne(4), ecc.

```
# MII/PHY support
```

Questa sezione elenca le interfacce di rete indipendenti dal dispositivo. In caso di dubbio, abilitarle tutte e lasciare la scelta al kernel. Vedere la pagina di manuale mii(4) per ulteriori informazioni.

```
# 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
```

Gli adattatori ethernet USD hanno una banda limitata a soli 2 MBit/s ma sono molto pratici da usare. Naturalmente, per funzionare, bisogna configurare anche le opzioni per USB, che però non sono descritte in questa sede, oltre, naturalmente, a disporre dell'hardware necessario. Si vedano le varie pagine di manuale.

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

Questo *pseudo-dispositivo* consente lo "sniffing" di pacchetti di tutti i tipi. È necessario per tcpdump ma anche per rarpd e per altre applicazioni che devono esaminare il traffico di rete. Vedere bpf(4) per ulteriori informazioni.

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

Questa riga attiva l'interfaccia IPfilter del kernel per il filtraggio dei pacchetti usata per il firewalling, NAT (detto anche IP masquerading), ecc. Vedere ipf(4) e la Sezione 10.2.1 per ulteriori informazioni.

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

Questa è l'interfaccia "loopback" di rete, usata sia da alcuni programmi sia per effettuare alcuni tipi di routing. Deve essere abilitata. Vedere la pagina di manuale lo(4) per i dettagli.

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

Quest'opzione è necessaria per poter usare il protocollo PPP, sia via interfaccia seriale sia via ethernet (PPPoE). Vedere ppp(4).

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

Il "Serial Line IP" è semplicemente il protocollo IP incapsulato per viaggiare su linea seriale. Poiché non include opzioni quali la negoziazione degli indirizzi IP, oggi non è più molto usato. Si veda sl(4).

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

Vedere la pagina di manuale strip(4) per il supporto ai vecchi dispositivi di rete Metricon Ricochet. Se ne avete uno, abilitate quest'opzione.

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

Questo dispositivo di rete serve per predisporre un *tunnel* per i pacchetti di rete verso un dispositivo `/dev/tun*`. I pacchetti inoltrati all'interfaccia tun0 si possono leggere in `/dev/tun0`, e i dati scritti su `/dev/tun0` vengono inviati all'interfaccia di rete tun0. Con questo sistema si può implementare, per esempio, l'instradamento QoS in modalità utente. Vedere tun(4) per ulteriori dettagli.

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

L'incapsulamento GRE si può usare per effettuare un tunnel di pacchetti arbitari di livello 3 su IP, per esempio per l'implementazione di VPN. Vedere gre(4) per ulteriori dettagli.

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

Un altro dispositivo per icapsulare IP su IP, con un diverso formato di incapsulamento. Vedere la pagina di manuale ipip(4) per i dettagli.

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

L'interfaccia GIF serve per attivare un tunnel, per esempio di IPv6 su IPv4, che può essere usato per ottenere connettività IPv6 in assenza di una connessione IPv6 con l'ISP. Sono possibili anche altre combinazioni di operazioni: vedere gli esempi contenuti nella pagina di manuale gif(4).

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

L'interfaccia faith cattura il traffico TCP IPv6, per implementare relay da IPv6 IPv4, per esempio per transizioni di protocollo. Vedere la pagina di manuale faith(4).

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

Questa direttiva aggiunge un dispositivo di rete che si può usare per creare un tunnel IPv6 su IPv4 senza bisogno di configurare il tunnel stesso. Il mittente dei pacchetti in uscita contiene l'indirizzo IPv4, il che consente di inviare le risposte via IPv4. Vedere la pagina di manuale stf(4) e la Sezione 10.2.4 per ulteriori dettagli.

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

Questa interfaccia fornisce il supporto per le LAN virtuali IEEE 802.1Q, consentendo di marcare i pacchetti ethernet con un ID "vlan". Configurando opportunamente degli switch che supportano le VLAN, si possono creare delle VLAN nelle quali un gruppo di macchine non vede il traffico degli altri gruppi. La pagina di manuale vlan(4) contiene ulteriori informazioni in proposito.

10.1.2. I file di configurazione di rete

La lista seguente contiene un elenco dei file di configurazione della rete, alcuni dei quali già visti nei capitoli precedenti. La descrizione di questi file sarà ò l'oggetto delle prossime sezioni.

`/etc/hosts`

Il database degli host locali. Ogni linea descrive un host e contiene l'indirizzo IP, il nome dell'host e gli eventuali alias. Le reti di modeste dimensioni possono essere configurate usando solo il file degli host, senza bisogno di un *name server*.

`/etc/resolv.conf`

Questo file specifica le modalità operative per le funzioni di libreria dell'Internet Domain Name System. Generalmente contiene gli indirizzi dei "name server".

`/etc/ifconfig.xxx`

Questo file viene usato per la configurazione automatica della scheda di rete all'avvio.

`/etc/mygate`

Contiene l'indirizzo IP del gateway.

`/etc/nsswitch.conf`

File di configurazione del *Name Service Switch*; controlla le modalità di accesso ai vari database contenenti le informazioni su host, utenti, gruppi, ecc. In pratica, questo file specifica l'ordine in base al quale vengono consultati i database. Per esempio, la linea:

```
hosts:    files dns
```

specifica che le informazioni sugli host vengono da due fonti, il file locale `/etc/hosts`) e il *DNS* (Internet Domain Name System), e che i file locali vengono consultati *prima* del DNS.

Solitamente non è necessario modificare questo file.

10.1.3. Collegarsi a Internet

Vi sono molti tipi di collegamento a Internet: questa sezione spiega come collegarsi a un provider usando un modem via linea telefonica con il protocollo PPP, uno dei setup più comuni. Le operazioni da compiere sono, nell'ordine:

1. Richiedere le informazioni necessarie al provider.
2. Editare il `/etc/resolv.conf` e controllare il file `/etc/nsswitch.conf`.
3. Creare le directory `/etc/ppp` e `/etc/ppp/peers` se non esistono.
4. Creare lo script di connessione, il chat file e il file di opzioni pppd.
5. Creare il file di autenticazione utente-password.

A giudicare dalla lista precedente sembra una procedura complessa che richiede molto lavoro. In realtà, ognuno dei punti elencati è molto semplice: si tratta soltanto di creare, modificare o semplicemente controllare alcuni piccoli file di testo. Nell'esempio seguente supporremo che il modem sia collegato alla seconda porta seriale `/dev/tty01` (COM2 in DOS.)

> Oltre ai modem esterni (o interni) connessi a una porta seriale (usando `/dev/tty0[012]` su i386, `/dev/tty[ab]` on sparc, ...) si possono usare anche modem su USB (`/dev/ttyU*`) e pcmcia/cardbus (`/dev/tty0[012]`).

10.1.3.1. Procurarsi le informazioni di collegamento

La prima cosa da fare è chiedere al provider, che supponiamo si chiami BigNet, di fornirci i dati che servono per potersi collegare; in particolare:

- Il numero di telefono del POP (Point Of Presence) più vicino.
- Il tipo di autenticazione da utilizzare.
- Il nome utente e la nostra password da utilizzare per il collegamento.
- Gli indirizzi IP dei nameserver.

10.1.3.2. resolv.conf e nsswitch.conf

Il file `/etc/resolv.conf` deve essere configurato utilizzando alcune informazioni fornite dal provider, in particolare i DNS. Supponiamo che i due DNS siano "194.109.123.2" e "191.200.4.52".

Esempio 10-1. resolv.conf

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

> **Nota:** la riga "lookup file bind" indica che i name server vengono utilizzati solo per le voci non presenti in `/etc/hosts`. La riga è commentata perché, a partire dalla versione 1.4 di NetBSD, non viene più utilizzata: le sue funzioni sono ora svolte dal file `/etc/nsswitch.conf`. Il nuovo Name Service Switch consente di cambiare i metodi di accesso ai database usati dai programmi per accedere alle informazioni base del sistema.

Esempio 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:** l'unica riga modificata in questo file è quella che inizia con "hosts:". Si noti nelle versioni successive alla 1.4.1 non dovrebbe più essere necessario effettuare questa modifica a mano.

10.1.3.3. Creare le directory per ppp

Le directory `/etc/ppp` e `/etc/ppp/peers` dovranno contenere i file di configurazione per il collegamento; al termine di una nuova installazione di NetBSD non esistono, e quindi bisogna crearle (chmod 700).

10.1.3.4. Script di collegamento e chat file

Lo script di collegamento è quello che viene specificato sulla riga di comando di pppd; si trova in `/etc/ppp/peers` e ha generalmente il nome del provider. Per esempio, supponendo che il provider si chiami BigNet e che il nostro nome utente presso il provider sia alan, questo potrebbe essere lo script per il collegamento:

Esempio 10-3. Script di collegamento al provider

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

Nel precedente esempio, lo script indica il *chat file* da utilizzare per il collegamento. Per una spiegazione delle opzioni: pppd(8).

> **Nota:** in caso di problemi di connessione, è possibile aggiungere al precedente file due righe contenenti le opzioni
>
> ```
> debug
> kdebug 4
> ```
>
> Per ulteriori informazioni: pppd(8), syslog.conf(5) .

Lo script di collegamento chiama il programma chat per effettuare il collegamento fisico vero e proprio (inizializzazione del modem, composizione del numero, ecc.). I parametri da passare a chat sono abbastanza numerosi e conviene metterli in un file, anche se è possibile specificarli inline nello script di collegamento. Per esempio, supponendo che il numero di telefono del POP più vicino sia 02 99999999

Esempio 10-4. Chat file

```
# /etc/ppp/peers/bignet.chat
ABORT BUSY
ABORT "NO CARRIER"
ABORT "NO DIALTONE"
REPORT CONNECT
SAY "Chiamo BigNet"
'' ATDT0299999999
CONNECT ''
```

> **Nota:** se si verificano dei problemi con il chatfile, può essere comodo collegarsi manualmente al provider con cu: in questo modo si possono vedere le stringhe esatte che vengono ricevute. Per ulteriori informazioni: cu(1).

10.1.3.5. Autenticazione

Durante la fase di autenticazione ognuno dei due sistemi verifica l'identità dell'altro. In realtà il provider solitamente non si aspetta di essere autenticato ma pretende cha a farlo sia l'utente che si collega.

Generalmente i provider per provvedere alla nostra autenticazione utilizzano uno di questi due metodi:

- login

- PAP/CHAP

La maggior parte dei provider utilizza un'autenticazione di tipo PAP/CHAP. Questa informazione viene generalmente fornita dal provider stesso.

10.1.3.5.1. Autenticazione con PAP/CHAP

È sufficiente creare il file `/etc/ppp/pap-secrets` (o `/etc/ppp/chap-secrets`) le cui righe hanno il seguente formato:

```
utente * password
```

Per esempio:

```
alan * pZY9o
```

> **Nota:** per motivi di sicurezza è bene che il file `pap-secrets` sia di proprietà di *root* e abbia permessi '600'.

10.1.3.5.2. Autenticazione con login

Si tratta di una forma di autenticazione sempre meno usata; se il provider usa l'autenticazione login, il nome utente e la password non devono essere specificati in `pap-secrets` o in `chap-secrets`, bensì nel chatfile (naturalmente in questo caso è bene adottare opportuni permessi per il chatfile). In questo caso il chatfile simula un login interattivo.

Esempio 10-5. Chat file con 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. Opzioni di pppd

Rimane solo da configurare il file delle opzioni di **pppd**, che è `/etc/ppp/options` (chmod 644.)

Esempio 10-6. /etc/ppp/options

```
/dev/tty01
lock
crtscts
```

```
57600
modem
defaultroute
noipdefault
```

10.1.3.7. Test del modem

Prima di attivare il collegamento può essere conveniente effettuare un test del modem, per verificare che tutto funzioni correttamente. Per il test si può usare il programma cu.

1. Creare il file `/etc/uucp/port` contenente le seguenti righe:

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

   (naturalmente, al posto di `/dev/tty01` bisogna sostituire il device appropriato).

2. Dare il comando **cu -p modem** per avviare il test interattivo. A questo punto si possono inviare comandi al modem. Per esempio:

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

   Disconnected.
   #
   ```

   Nell'esempio precedente viene inviato al modem il comando di reset (ATZ) e il modem risponde con OK. Come si vede, per uscire da cu bisogna dare il comando **~.** (carattere tilde seguito dal carattere punto).

Se il modem sembra non funzionare, la prima cosa da controllare è che il modem sia realmente collegato al port usato da **cu**. Un'altra frequente causa di problemi sono i cavi.

> **Nota:** se quando si avvia **cu** compare un messaggio che dice "Permission denied", verificare chi è il proprietario dei file `/dev/tty##`: deve essere uucp. Per esempio:
>
> ```
> $ ls -l /dev/tty00
> crw-------  1 uucp  wheel  8, 0 Mar 22 20:39 /dev/tty00
> ```
>
> Se invece il proprietario è 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
> ```

Nota a margineCOM, com e tty

Non bisogna fare confusione tra il *com*, *COM* e *tty*. Per NetBSD, “com” è il nome del driver (quello che viene visualizzato da **dmesg**) e “tty” è il nome del port. Poiché la numerazione parte da 0, com0 sarà il driver di tty00. Negli ambienti DOS, invece, COM1 indica il primo port (quello che solitamente ha indirizzo 0x3f8), COM2 il secondo e così via. Pertanto COM1 (DOS) corrisponde a tty00 (NetBSD).

10.1.3.8. Attivare il collegamento

Finalmente tutto è pronto per attivare il collegamento con il provider dando il comando

```
# pppd call bignet
```

dove `bignet` è il nome dello script di collegamento già descritto. Per vedere i messaggi di collegamento di **pppd** e **chat**, dare il comando

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

Per scollegarsi bisogna fare un **kill -HUP** di **pppd**.

10.1.3.9. Automatizzare il collegamento e lo scollegamento

Quando il collegamento è a posto e funziona correttamente è il momento di creare un paio di comodi script per connettersi e per disconnettersi automaticamente. Per esempio, per connettersi:

Esempio 10-7. ppp-up

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

e per sconnettersi:

Esempio 10-8. ppp-down

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

```
else
  echo ppp e\' gia\' inattivo 1>&2
fi
```

Questi due script sfruttano il fatto che **pppd**, quando è attivo, crea il file `/var/spool/lock/LCK..tty01` che contiene il *pid* (identificativo di processo) di **pppd**.

I due script devono essere eseguibili:

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

> **Nota:** per lanciare pppd bisogna essere *root* oppure appartenere al gruppo *dialer* o al gruppo *operator*. Quindi per l'uso degli script da parte di un utente normale si può utilizzare sudo (`/usr/pkgsrc/security/sudo`) oppure assegnare l'utente a uno dei gruppi appena citati.

10.1.4. Creare una piccola rete domestica

Il networking è sicuramente uno dei punti di forza di Unix in generale e di NetBSD in particolare, e la creazione di una rete domestica è un'operazione semplicissima (ed economica). Tra le altre cose, ne la Sezione 10.2.1 si vedrà come configurare una macchina NetBSD per fare da gateway e da firewall Internet per il resto della rete, consentendo a tutte le macchine della rete l'accesso a Internet con un solo collegamento al provider; anche qui non c'è alcuna necessità di acquistare software aggiuntivo: NetBSD dispone già di tutto il necessario. L'unica accortezza che bisogna avere è quella di acquistare schede di rete supportate da NetBSD (consultare il file `INSTALL` per l'elenco dei dispositivi supportati).

La prima cosa da fare è installare le schede di rete Ethernet sulle macchine e collegarle a un hub, a uno switch oppure collegarle direttamente tra loro con l'apposito cavo coassiale e i terminatori. Per questo esempio si faccia riferimento alla figura Figura 10-1.

Una volta installate le schede di rete bisogna controllare che vengano riconosciute correttamente dal kernel studiando l'output di **dmesg**. Per esempio:

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

naturalmente è necessario che la scheda sia abilitata nel kernel, in caso contrario abilitarla, ricompilare il kernel e riavviare. Per esempio:

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

Se la scheda non viene riconosciuta dal kernel, bisogna esaminare il file di configurazione del kernel per controllare l'IRQ della scheda di rete. L'IRQ specificato nel file deve essere quello realmente utilizzato dalla scheda, altrimenti nella maggior parte dei casi il kernel non potrà riconoscerla all'avvio. Se l'IRQ è diverso si può cambiare la linea del file di configurazione e ricompilare il kernel oppure riconfigurare la scheda (solitamente con un dischetto di setup o, per le schede più datate, con un cavallotto).

Si provi ora a dare il comando seguente:

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

L'output del comando precedente mostra l'attuale configurazione della scheda di rete.

A questo punto si può passare alla configurazione software, che è molto semplice. Per prima cosa conviene provare la scheda, iniziando con il configurarla:

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

L'effetto del comando si può verificare con:

```
# 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 con l'output del precedente comando ifconfig si può notare che è comparso l'indirizzo di rete assegnato e sono comparsi i flag "UP" e "RUNNING". Se un'interfaccia non è "UP", il sistema non la utilizzerà per trasmettere pacchetti.

All'host è stato attribuito l'indirizzo IP 192.168.1.1, che appartiene agli indirizzi di rete cosiddetti "interni", ossia riservati alle reti non visibili da Internet. A questo punto si può dire che la configurazione è terminata e non resta che provarla; se vi sono altri host già in rete si può provare a fare un *ping* verso un altro host. Per esempio, se l'indirizzo IP di un host attivo della rete è 192.168.1.2, provare:

```
# 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
```

Al prossimo riavvio la configurazione della scheda andrà effettuata nuovamente; per automatizzare l'operazione conviene creare il file `/etc/ifconfig.ne0` contenente la linea

```
inet 192.168.1.1 netmask 0xffffff00
```

Poi, nel file `/etc/rc.conf` bisogna impostare l'opzione

```
auto_ifconfig=YES
```

infine si inseriscono gli indirizzi degli host in /etc/hosts. Per esempio:

Esempio 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
```

Il file `/etc/nsswitch.conf` deve essere modificato come già spiegato in Esempio 10-2.

Al prossimo riavvio tutto sarà configurato automaticamente.

> **Nota:** in questo esempio è stato creato il file `/etc/ifconfig.ne0` perché la scheda di rete veniva riconosciuta come *ne0*. Per altre schede bisogna sostituire il nome opportuno.

Per configurare la rete bisogna dunque semplicemente collegare fisicamente i computer, assegnare a ciascuno un indirizzo IP e configurare il file `/etc/hosts` con gli indirizzi di tutti su ogni host e modificare il file `/etc/nsswitch.conf`. Si tratta di una politica di gestione elementare, adatta soltanto a reti di piccole dimensioni.

10.1.5. Collegare due PC via seriale

Capita talvolta (almeno a me) di trovarsi nella necessità di trasferire dei file da un portatile a un PC. Se il portatile non si può collegare in rete e non ha il floppy (o magari il file è troppo grosso per un dischetto) una soluzione semplice ed efficace è quella di collegare le due macchine con un cavo *null modem*. Nelle sezioni seguenti vediamo alcuni esempi di possibili configurazioni.

10.1.5.1. Collegare due macchine BSD

Se entrambe le macchine da collegare utilizzano NetBSD, ci si trova nella situazione più favorevole: la realizzazione di un collegamento SLIP è semplicissima. Sulla prima macchina

```
# slattach /dev/tty00
# ifconfig sl0 inet 192.168.1.1 192.168.1.2
```

sulla seconda macchina

```
# slattach /dev/tty00
# ifconfig sl0 inet 192.168.1.2 192.168.1.1
```

A questo punto si può provare il collegamento con ping; per esempio, dal secondo PC eseguire

```
# ping 192.168.1.1
```

Se tutto è andato bene le due macchine sono *in rete* e si possono usare ftp, telnet, ecc. Può anche essere conveniente inserire i nomi e gli indirizzi degli host nel file `/etc/hosts`.

- Nell'esempio precedente entrambe le macchine usavano la prima porta seriale (/dev/tty00).
- Gli indirizzi 192.168.x.x sono tra quelli riservati per le reti private. La prima macchina ha indirizzo 192.168.1.1 e la seconda ha indirizzo 192.168.1.2.
- Per ottenere un collegamento più veloce bisogna specificare il parametro `-s` *`velocità`* a slattach.
- Per poter eseguire ftp deve essere stato avviato inetd e deve essere abilitato il server ftpd.

> **Nota:** se uno dei due PC utilizza Linux, i comandi su quest'ultimo sono leggermente diversi. Supponendo di dare al PC Linux l'indirizzo 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
> ```
>
> Non bisogna dimenticare "&" nel primo comando.

10.1.5.2. Collegare Windows NT a NetBSD

Windows NT e NetBSD possono essere messi in rete in modo semplice con un cavo seriale null modem. Il procedimento consiste nel creare una connessione di "Accesso remoto" sotto NT e nell'avviare pppd su NetBSD. Vediamo praticamente come fare.

Supponendo di avviare pppd come root, bisogna creare un file `.ppprc` in `/root`. Il contenuto del file deve essere simile al seguente:

```
connect '/usr/sbin/chat -v CLIENT CLIENTSERVER'
local
tty00
115200
crtscts
lock
noauth
nodefaultroute
:192.168.1.2
```

Il significato della prima riga verrà spiegato dopo; 192.168.1.2 è l'indirizzo IP che verrà assegnato da NetBSD all'host NT; tty00 è la porta seriale usata per la connessione.

Dal lato NT bisogna prima installare un modem di tipo *null modem* (connessione diretta) e poi creare una connessione di Accesso remoto che usi questo modem. Il driver null modem è standard sotto NT, ma non si tratta di un vero e proprio null modem, infatti all'attivazione del collegamento NT invia la stringa CLIENT e si aspetta di ricevere la risposta CLIENTSERVER (ecco il significato della prima riga di `.ppprc`: chat deve rispondere a NT, altrimenti la connessione non va a buon fine). Per la creazione della connessione utilizzare il null modem, indicare un numero telefonico qualsiasi (tanto non viene usato), scegliere un server ppp, abilitare solo il protocollo TCP/IP, usare indirizzo IP e nameserver indicati dal server (NetBSD, in questo caso). Inoltre impostare il flusso di controllo hardware e la porta a 115200 8N1.

A questo punto si può effettuare la connessione:

- Collegare fisicamente le seriali delle due macchine.
- Lanciare pppd su NetBSD. (Per vedere i messaggi inviati da pppd: **tail -f /var/log/messages**).
- Attivare la connessione di Accesso remoto su NT.

10.1.5.3. Collegare Windows 95 a NetBSD

Come per il caso di Windows NT, conviene usare la connessione di Accesso remoto di Windows e il server ppp di BSD. Purtroppo Windows 95 non dispone di un driver *null modem*, il che complica un po' le cose. La cosa più semplice è procurarsene uno su Internet (si tratta di un banale file `.INF`) e seguire le stesse istruzioni di Windows NT con l'unica avvertenza di rimuovere la linea che richiama chat dal file `.ppprc`.

In mancanza di un driver null modem si può usare un piccolo trucco:

- Creare una connessione di Accesso remoto come quella descritta ne la Sezione 10.1.5.2 ma utilizzando il "Modem Standard".
- In `.ppprc` sostituire la linea che chiama chat con la seguente

  ```
  connect '/usr/sbin/chat -v ATH OK AT OK ATE0V1 OK AT OK ATDT CONNECT'
  ```

- Per la connessione procedere come nel caso di Windows NT.

In questo modo il programma chat, richiamato alla connessione, emula il comportamento di un modem standard e restituisce a Windows le stringhe di risposta che riceverebbe da un modem: a ogni comando AT inviato, chat risponde con un OK.

10.2. Argomenti avanzati

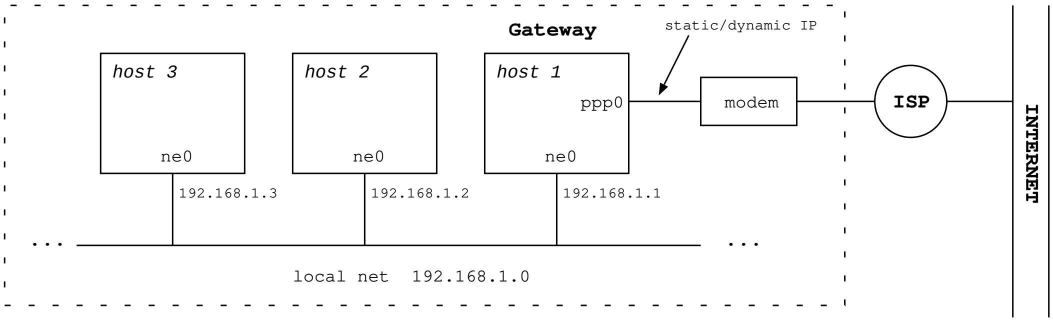
10.2.1. IPNAT

Dietro alla misteriosa sigla IPNAT si nasconde l'Internet Protocol Network Address Translation, che consente l'instradamento di una rete IP fittizia (locale) su una vera (Internet). Ciò significa che con un solo IP fornito dal provider si possono collegare a Internet più computer creando una finta rete IP interna e instradando i dati tramite un computer collegato a Internet sui cui è in funzione IPNAT.

Per alcuni esempi d'uso di IPNAT, vedere nella subdirectory `/usr/share/examples/ipf`, in particolare i file BASIC.NAT e nat-setup.

L'esempio presentato in questa sezione è illustrato dalla Figura 10-1: l'host 1 può collegarsi a Internet con il modem chiamando un provider, che assegna un indirizzo IP dinamico. Gli host 2 e 3 non possono comunicare con Internet. Per rendere possibile il collegamento anche a loro si usano IPF e IPNAT.

Figura 10-1. Rete con gateway



10.2.1.1. Configurazione del gateway/firewall

Se il kernel è stato compilato con l'opzione GATEWAY attiva, questa è abilitata di default senza eseguire ulteriori operazioni di configurazione. Per controllare se è abilitata:

```
# sysctl net.inet.ip.forwarding
net.inet.ip.forwarding = 1
```

Se il risultato è "1", come nell'esempio precedente, l'opzione è abilitata. Inoltre nel file di configurazione del kernel deve essere abilitato "pseudo-device ipfilter". Se il risultato è invece "0", allora l'opzione non è abilitata e vi sono due possibilità:

1. Compilare un nuovo kernel con l'opzione GATEWAY attiva.
2. Attivare l'opzione nel kernel corrente con il seguente comando:

   ```
   # sysctl -w net.inet.ip.forwarding=1
   ```

   Le impostazioni di sysctl vanno aggiunte al file `/etc/sysctl.conf` in modo che siano ripristinate automaticamente al riavvio del sistema. In questo caso bisogna aggiungere

   ```
   net.inet.ip.forwarding=1
   ```

Le istruzioni seguenti spiegano come creare una configurazione di IPNAT che entri in funzione automaticamente ogni volta che ci si collega con il provider e si attiva il link PPP. Con questa configurazione tutti i PC della rete domestica potranno condividere il collegamento a Internet, anche se non utilizzano NetBSD.

Creare il file `/etc/ipnat.conf` contenente le seguenti regole:

```
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 rappresenta il set di indirizzi interni da mappare. La prima linea consente il funzionamento del tcp attivo attraverso NAT ed è opzionale. La seconda riga assicura che i pacchetti tcp e udp vengano gestiti correttamente da NAT (il portmapping è necessario per via della relazione molti a uno). La terza riga fa funzionare tutto il resto (ICMP, ping, ecc.).

Creare il file `/etc/ppp/ip-up` che verrà chiamato automaticamente all'attivazione del link PPP.

```
#!/bin/sh
# /etc/ppp/ip-up
/etc/rc.d/ipnat forcestart
```

Creare il file `/etc/ppp/ip-down` che verrà chiamato automaticamente alla disattivazione del link PPP.

```
#!/bin/sh
# /etc/ppp/ip-down
/etc/rc.d/ipnat forcestop
```

`ip-up` e `ip-down` devono essere eseguibili:

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

10.2.1.2. Configurazione dei client

Creare il file `/etc/resolv.conf` (come quello del gateway).

Dare il comando **route add default 192.168.1.1** (192.168.1.1 è l'indirizzo IP del gateway precedentemente configurato).

Naturalmente è molto scomodo ripetere questo comando a ogni avvio. Per rendere automatica la configurazione del gateway si possono usare due metodi equivalenti:

- specificare l'indirizzo del gateway utilizzando l'opzione "defaultroute" del file `/etc/rc.conf`
- scrivere l'indirizzo del gateway nel file `/etc/mygate`

A ogni boot, il contenuto di `/etc/mygate` (o dell'opzione defaultroute) vengono usati come argomento per il comando **route add default**.

Se il client non è una macchina NetBSD, la sua configurazione sarà naturalmente diversa. Su una macchina Windows, per esempio, bisogna impostare la proprietà "gateway" del protocollo TCP/IP.

10.2.1.3. Alcuni comandi utili

Per effettuare verifiche possono essere utili i seguenti comandi:

ping

netstat -r

Visualizza le tabelle di instradamento (simile a **route show**).

traceroute

Sul client; mostra la strada seguita dai pacchetti per arrivare a destinazione.

tcpdump

Da eseguire sul gateway.

10.2.2. NFS

Una volta messa in funzione la rete è possibile condividere file e directory tra computer grazie a NFS. Dal punto di vista della condivisione dei file, il computer che offre accesso alle sue gerarchie di directory è il *server*, quello che accede è il *client*. Un computer può essere contemporaneamente client e server. Vediamo quali sono le operazioni da compiere

- Sia per il client sia per il server deve essere compilato un kernel con abilitata l'apposita opzione (non è difficile da trovare...).
- Il server deve abilitare i servizi RPC in `/etc/inetd.conf`.
- In `/etc/rc.conf` il server deve abilitare i daemon inetd e portmap e l'opzione nfs_server.
- In `/etc/rc.conf` il client deve abilitare inetd e nfs_client.
- Il server deve elencare le esportazioni in `/etc/exports` e poi eseguire il comando **kill -HUP 'cat /var/run/mountd.pid**.

L'accesso a una directory remota si basa su due presupposti:

- Il computer remoto esporta la directory.
- Il computer locale monta la directory remota con il comando **mount server:/xx/yy /mnt**

Il comando mount è dotato di un ricco assortimento di opzioni, non proprio semplicissime.

10.2.2.1. Esempio di impostazione di NFS

Questo esempio, abbastanza elaborato, è stato preso da una situazione reale presentata su una mailing list. Supponiamo di avere il seguente scenario: cinque macchine client (cli1, ..., cli5) devono accedere a alcune directory su un server (buzz.toy.org). Alcune delle directory esportate dal server sono specifiche dei singoli client, altre sono condivise da tutti i client. I client fanno tutti il boot dal server e devono montare le directory. Le directory da esportare dal server sono:

`/export/cli?/root`

deve servire da directory root per ognuna delle macchine cli?.

`/export/cli?/swap`

area di swap per ognuna delle macchine cli?.

`/export/common/usr`

directory `/usr` comune a tutte le macchine cli?.

`/usr/src`

da montare come `/usr/src` su ognuna delle macchine cli? (comune).

Sul server sono presenti i seguenti file system

```
/dev/ra0a su /
/dev/ra0f su /usr
/dev/ra1a su /usr/src
/dev/ra2a su /export
```

Sui client si vogliono ottenere i seguenti file system

```
buzz:/export/cli?/root su /
buzz:/export/common/usr su /usr
buzz:/usr/src su /usr/src
```

La configurazione del server è la seguente:

```
# /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
```

Ovviamente 123.45.67.0 deve essere sostituito con l'indirizzo reale della rete, come pure la netmask. Inoltre può essere necessario aggiungere: **-maproot=root** alla riga con "/usr/src".

Sui client `/etc/fstab` deve contenere

```
buzz:/export/cli?/root / nfs rw
buzz:/export/common/usr /usr nfs rw,nodev,nosuid
buzz:/usr/src /usr/src rw,nodev,nosuid
```

Naturalmente per ogni client bisogna sostituire il relativo numero al posto del carattere "?" in `/etc/fstab`.

10.2.3. Impostare `/net` con amd

L'autore di questa sezione (Impostare ...) è Hubert Feyrer <`hubert@feyrer.de`>.

10.2.3.1. Introduzione

Il problema con i mount NFS (e con gli altri tipi di mount) è che solitamente possono essere effettuati dall'utente "root", il che può risultare molto scomodo per gli altri utenti. Grazie a **amd** si può impostare una directory (nel nostro esempio verrà usata la directory `/net`) sotto la quale anche gli utenti normali possono effettuare mount NFS, a condizione che il filesystem a cui si deve accedere sia effettivamente esportato dal server NFS.

Per verificare quali sono i file system esportati da un server NFS, usare il comando **showmount** con l'opzione `-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
```

Per montare una directory e accedere a tutto il suo contenuto, subdirectory comprese, (per esempio `/archive/systems/unix/NetBSD`), basta portarsi nella directory con **cd**:

```
# cd /net/wuarchive.wustl.edu/archive/systems/unix/NetBSD
```

Il file system viene montato automaticamente da **amd** ed è possibile accedere a tutti i file proprio come se la directory fosse stata montata dall'amministratore di sistema.

10.2.3.2. Impostazioni pratiche

La directory `/net` del nostro esempio si può impostare con le seguenti operazioni (che includono il setup di **amd**):

1. in `/etc/rc.conf`, impostare la variabile seguente:

   ```
   amd=yes
   ```

2. **mkdir /amd**
3. **mkdir /net**
4. Usando `/usr/share/examples/amd/master` come esempio, aggiungere la direttiva seguente a `/etc/amd/master`:

   ```
   /net            /etc/amd/net
   ```

5. Usando `/usr/share/examples/amd/net` come esempio, aggiungere le seguenti righe 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. Riavviare la macchina, oppure riavviare **amd** manualmente:

   ```
   # sh /etc/rc.d/amd restart
   ```

10.2.4. Connettività IPv6 e passaggio a IPv6 con 6to4

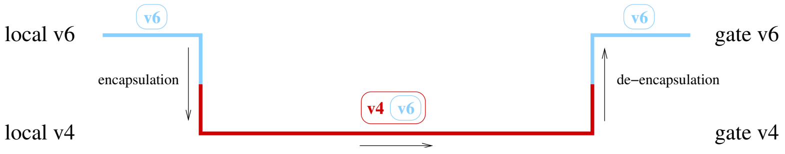
L'autore di questa sezione (Connettività ...) è Hubert Feyrer `<hubert@feyrer.de>`.

Questa sezione spiega come arrivare praticamente alla connettività IPv6 e, poiché quest'ultima non è ancora facile da ottenere in modo nativo, descrive diffusamente le alternative alla connettività nativa IPv6 e un metodo per effettuare la transizione finché le connessioni IPv6 diventeranno più comuni.

Trovare un ISP che offra la connettività IPv6 nativa richiede una buona dose di fortuna. Il passo successivo è trovare un router in grado di gestire il traffico; al momento non tutti i produttori di router supportano IPv6 sulle loro macchine e, anche auqndo lo fanno, è improbabile che offrano routing o switching IPv6 con accelerazione hardware. Un'alternativa economica e di uso comune al router hardware consiste nell'usare un PC standard configurandolo come router, per esempio utilizzando un sistema operativo quale NetBSD e un sofware del tipo di Zebra per gestire i protocolli di routing. Questa soluzione è oggi abbastanza diffusa per quei siti che vogliono subito la connettività IPv6; l'unico svantaggio è che serve un ISP che supporta IPv6 e un colegamento dedicato solo per IPv6.

In mancanza di connettività nativa, si può ancora utilizzare IPv6 grazie ai *tunnel*. Invece di inviarli direttamente, i pacchetti IPv6 vengono incapsulati dentro a pacchetti IPv4, come mostrato in Figura 10-2. Usando l'infrastruttura IPv4, i pacchetti incapsulati vengono inviati a un collegamento reale IPv6 che rimuove l'incapsulamento e li invia in modalità nativa.

Figura 10-2. Un tunnel di pacchetti IPv6 dentro a IPv4



Per usare i tunnel ci sono due possibilità. La prima consiste nell'uso di un tunnel cosiddetto "configurato", la seconda nell'uso di un tunnel "automatico". Un tunnel configurato è caratterizzato dal fatto di richiedere una preparazione da entrambi i lati del tunnel stesso, generalmente soggetta a una qualche forma di registrazione al fine di scambiare i dati di setup. Un esempio di questo tipo di tunnel è l'incapsulamento IPv6-su-IPv4 descritto in [RFC1933] e implementato, per esempio, dall'interfaccia gif(4) di NetBSD.

Un tunnel automatico richiede un server pubblico dotato di una qualche forma di connettività IPv6, per esempio via 6Bone. Questo server rende pubblici i propri dati di connettività e attiva un protocollo di "tunnelling" che non richiede una registrazione esplicita dei siti che lo usano per collegarsi. Un popolare esempio di questo protocollo è il meccanismo 6to4 descritto in [RFC3056], e implementato dal dispositivo stf(4) di NetBSD. Un altro meccanismo che non richiede registrazione di informazioni IPv6 è il cosiddetto 6over4, che implementa il trasporto di IPv6 su una rete IPv4 abilitata al multicast, invece di, per esempio, ethernet o FDDI. 6over4 è documentato in [RFC2529]. Il suo principale svantaggio è che è necessaria la presenza di un'infrastruttura multicast esistente. Se non c'eè una struttura di questo tipo, la sua creazione e impostazione richiede uno sforzo paragonabile alla configurazione di un tunnel IPv6 diretto, e pertanto, in questo caso, si tratta di un'ipotesi che non vale la pena di considerare.

10.2.4.1. Utilizzare IPv6 con 6to4

6to4 offre una soluzione semplice per ottenere la connettività IPv6 è per quegli host che sono dotati solo di un collegamento IPv4 (vedere la Sezione 9.1.7. Si può utilizzare con indirizzi IPv4 sia statici sia dinamici (il tipo più comune nel caso di connessioni "dial-up" via modem). Quando si usa un indirizzo IPv4 dinamico, i cambiamenti di indirizzo sono un problema per il traffico in entrata, il che comporta che non si può usare questo tipo di impostazione per un server web.

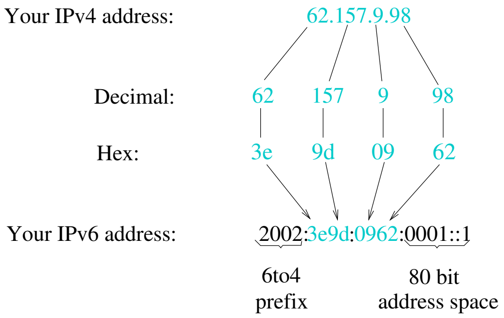
La configurazione di esempio descritta in questa sezione si riferisce a NetBSD 1.5.2.

10.2.4.2. Ottenere lo spazio di indirizzi IPv6 per 6to4

Il setup IPv6 di 6to4 non consiste di un solo indirizzo dal lato utente: infatti si ottiene un'intera rete /48! Gli indirizzi IPv6 vengono ricavati a partire dal singolo indirizzo IPv4 originale. Il prefisso "2002:" è riservato agli indirizzi di tipo 6to4 (cioè agli indirizzi IPv6 derivati da IPv4). I 32 bit seguenti contengono l'indirizzo IPv4. Questo corrisponde a avere la disponibilità di un'intera rete /48, lasciando 16 bit di spazio per 2^{16} sottoreti IPv6, ognuna con un massimo di 2^{64} nodi. Figura 10-3 mostra la costruzione dell'indirizzo (o meglio dell'insieme di indirizzi) IPv6 a partire da quello IPv4.

Grazie al prefisso 6to4 e all'unicità dell'indirizzo IPv4, il blocco di indirizzi IPv6 ottenuti è anch'esso unico ed è assegnato alla macchina che aveva l'indirizzo IPv4 di partenza.

Figura 10-3. 6to4 derives a IPv6 from an IPv4 address

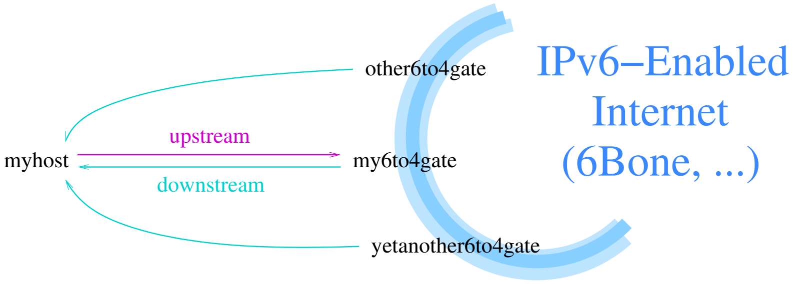


10.2.4.3. Come collegarsi

A differenza del setup configurato del tunnel "IPv6-over-IPv4", non è necessario registrarsi presso un gateway 6bone che inoltri il traffico IPv6. Invece, poiché l'indirizzo IPv6 deriva da quello IPv4, le risposte vengono inviate all'utente dal più vicino gateway 6to4. L'incapsulamento dei pacchetti viene elimnato da un'interfaccia di rete abilitata 6to4, che poi provvede a inoltrarli a seconda dell’instradamento impostato localmente (nel caso vi sia più di una macchina collegata alla rete 6to4 che è stata assegnata.

Per quanto riguarda l’invio di pacchetti IPv6 (traffico in uscita), l’interfaccia di rete abilitata 6to4 prende il pacchetto IPv6 e lo incapsula in un pacchetto IPv4. È necessario un collegamento al gateway 6to4, a sua volta collegato alla 6bone, che elimini l’incapsulamento dai pacchetti e li inoltri sulla 6Bone, come illustrato in Figura 10-4.

Figura 10-4. Richiesta e risposta possono passare da due gateway diversi nel setup 6to4



10.2.4.4. Sicurezza

A differenza del setup con un “tunnel preconfigurato”, in questo caso non si riesce solitamente a impostare filtri sui pacchetti per bloccare pacchetti 6to4 provenienti da fonti non autorizzate, proprio a causa del metodo di funzionamento del meccanismo 6to4. I pacchetti con le seguenti caratteristiche non dovrebbero essere considerati pacchetti 6to4 validi, e quindi è giustificato filtrarli:

- indirizzo sorgente/destinazione IPv4 non specificato: 0.0.0.0/8
- indirizzo di loopback (v4) come sorgente/destinazione: 127.0.0.0/8
- multicast IPv4 in sorgente/destinaziomne: 224.0.0.0/4
- broadcast limitato: 255.0.0.0/8
- indirizzo di broadcast di sottorete in sorgente/destinazione: dipende dal setup IPv4

La pagina di manuale stf(4) di NetBSD riporta alcuni comuni errori di configurazione che vengono intercettati dallo stack KAME e contiene anche alcuni suggerimenti per il filtraggio dei pacchetti, anche se, a causa dei requisiti di questi filtri, il 6to4 non è completamente sicuro. Tuttavia, se i pacchetti 6to4 fittizi diventano un problema, si può utilizzare l’autenticazione IPsec per assicurarsi che i pacchetti IPv6 non vengano modificati.

10.2.4.5. Dati necessari per il setup di 6to4

Per configurare IPv6 via 6to4, sono necessarie alcune informazioni, e cioè:

- L’indirizzo IPv4 locale, che si può determinare con il comando **ifconfig -a** oppure con **netstat -i** sulla maggior parte dei sistemi Unix. Se si usa un gateway con NAT o simili, bisogna utilizzare l’indirizzo ufficiale, visibile dall’esterno, e non quello privato (10/8 or 192.168/16).

In questo esempio verrà usato l'indirizzo IP locale 62.224.57.114.

- L'indirizzo IPv6 locale derivato da quello IPv4. Vedere Figura 10-3.

  In questo esempio l'indirizzo è 2002:3ee0:3972:0001::1 (62.224.57.114 == 0x3ee03972, 0001::1 è stato scelto arbitrariamente).

- L'indirizzo IPv6 del gateway 6to4 da utilizzare. L'indirizzo IPv6 va bene perché contiene quello IPv4, nella solita traduzione 6to4.

  In questo esempio usermo 2002:c25f:6cbf::1 (== 194.95.108.191 == 6to4.ipv6.fh-regensburg.de).

10.2.4.6. Preparazione del kernel

Pre elaborare pacchetti di tipo 6to4, il kernel del sistema operativo deve essere messo in grado di riconoscerli e gestirli, ed è pertanto necessario abilitare un apposito driver.

Per preparare il kernel di NetBSD all'uso di IPv6 e 6to4, bisogna attivare le linee seguenti nel file di configurazione del kernel:

```
options INET6                # IPv6
pseudo-device stf            # 6to4 IPv6 over IPv4 encapsulation
```

> **Nota:** il dispositivo stf(4) non è attivo di default.

Ricompilare il kernel e riavviare la macchina per attivare il nuovo kernel. Vedere Capitolo 7 per ulteriori informazioni sulla configurazione, compilazione e installazione di un nuovo kernel.

10.2.4.7. Setup di 6to4

La presente sezione descrive i comandi necessari per il setup di 6to4. In breve, le operazioni da compiere sono:

1. configurare l'interfaccia
2. impostare la route di default
3. impostare il Router Advertisement, se lo si desidera

Il primo passo da compiere per impostare 6to4 è di assegnare un indirizzo IPv6 all'interfaccia 6to4, usando il comando ifconfig(8). Data la configurazione appena descritta, il comando è:

```
# ifconfig stf0 inet6 2002:3ee0:3972:1::1 prefixlen 16 alias  (local)
```

Dopo aver configurato il dispositivo 6to4 con questo comando, bisogna impostare il routing, per inoltrare il traffico IPv6 al gateway 6to4. Il metodo migliore consiste nell'impostare una route di

```
# route add -inet6 default 2002:cdb2:5ac2::1 (remote)
```

Si noti che il dispositivo stf(4) di NetBSD determina l'indirizzo del gateway 6to4 dalla tabella di routing. Sfruttando questa caratteristica è semplice impostare un proprio gateway 6to4 se si dispone di un vero collegamento IPv6, per esempio via 6Bone.

Dopo aver dato quetsi comandi il colegamento al mondo IPv6 è attivo: congratulazioni! Assumendo che la risoluzione dei nomi venga sempre effettuata via IPv4, si può ora effettuare un "ping" di un sito IPv6, quale www.kame.net oppure www6.netbsd.org:

```
# /sbin/ping6 www.kame.net
```

L'ultimo passo nell'impostazione di IPv6 via 6to4 consiste nel setup del Router Advertisement, per il caso i cui vi siano più host sulla rete. Benché sia possibile ripetere le impostazioni 6to4 su ognuno dei nodi, quest'operazioni comporta dei costi di routing onerosi da un nodo all'altro - i pacchetti vengono inviati al gateway 6to4 che li inoltrera al nodo vicino. Invece è preferibile impostare 6to4 su un'unica macchina e impostare IPv6 nativo sulla rete locale.

Si inizia con l'assegnare un indirizzo IPv6 all'interfaccia di rete. Nell'esempio seguente assumiamo che la sottorete "2" della rete IPv6 sia la rete ethernet locale e che l'indirizzo MAC dell'interfaccia ethernet sia 12:34:56:78:9a:bc, cioè l'indirizzo IP dell'interfaccia ethernet dela gateway locale è 2002:3ee0:3972:2:1234:56ff:fe78:9abc. Assegnamo questo indirizzo all'interfaccia ethernet:

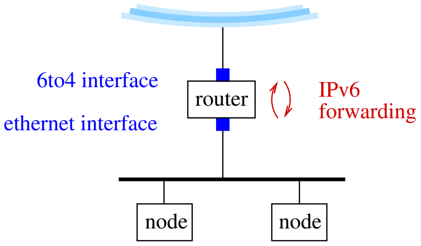
```
# ifconfig ne0 inet6 alias 2002:3ee0:3972:2:1234:56ff:fe78:9abc
```

Nel comando precedente, "ne0" è l'interfaccia che si riferisce alla scheda di rete ethernet; naturalmente sarà diversa a seconda della scheda montata sul sistema.

Per il setup del router bisogna poi verificare che inoltri effettivamente i pacchetti dal dispositivo 6to4 locale all'intrefaccia ethernet e viceversa. Per abilitare il forwarding dei pacchetti IPv6, impostare "ip6mode=router" nel file `/etc/rc.conf`, che mette a "1" il "sysctl" di net.inet6.ip6.forwarding.

```
# sysctl -w net.inet6.ip6.forwarding=1
```

Figura 10-5. Per un router 6to4 bisogna abilitare il forwarding dei pacchetti



Per impostare il Router Advertisement bisogna controllare il file `/etc/rtadvd.conf`. Questo file consente di configurare parecchie cose ma, solitamente, la configurazione di default di non contenere dati va bene. Con questa impostazione, gli indirizzi IPv6 di tutte le schede di rete del router saranno resi pubblici.

Dopo aver verificato che la configurazione del Router Advertisement è corretta e che il forwarding IPv6 è abilitato, si può avviare il relativo demone che, sotto NetBSD, si chiama **rtadvd**. La prima volta lo si può avviare manualmente a scopo di test, ma poi conviene usare gli script di avviamento del sistema: in ogni caso si vedranno tutti i nodi locali magicamente configurare la sottorete resa pubblica in aggiunta ai loro indirizzi link-local preesistenti.

```
# rtadvd
```

10.2.4.8. Breve introduzione all'uso di pkgsrc/net/6to4

Fino a ora abbiamo descritto il funzionamento e l'impostazione di 6to4. Per automatizzare queste operazioni, per esempio quando ci si collega "online", conviene utilizzare il package "6to4", che determina l'indirizzo IPv6 a partire da quello IPv4 assegnato dal provider e poi provvede alle necessarie impostazioni.

Il setup del pacchetto pkgsrc/net/6to4 richiede le seguenti operazioni:

1. Installare il package compilandolo da pkgsrc oppure usando il pacchetto precompilato 6to4-1.nb1 con il comando **pkg_add**.

   ```
   # cd /usr/pkgsrc/net/6to4
   # make install
   ```

2. Verificare che il pseudo-dispositivo stf(4) sia incluso nel kernel (vedere sopr).
3. Configurare il pacchetto "6to4". Per prima cosa, copiare `/usr/pkg/etc/6to4.conf-example` su `/usr/pkg/etc/6to4.conf`, e poi sistemare le variabili. Si noti che il file utilizza la sintassi perl.

   ```
   # cd /usr/pkg/etc
   # cp 6to4.conf-example 6to4.conf
   # vi 6to4.conf
   ```

   La pagina di manuale 6to4(8) descrive tutte le variabili che si possono impostare in `6to4.conf`. Se si ha una connessione IP dialup via PPP e non si è interessati al Router Advertising per le altre macchine IPv6 della rete locale domestica o dell'ufficio, non è necessario effettuare alcuna configurazione. Se, invece, si vuole utilizzare il Router Advertising, bisogna impostare la variabile `in_if` al valore dell'interfaccia ethernet interna. Per esempio:to the internal (ethernet)

   ```
   in_if="rtk0";            # Inside (ethernet) interface
   ```

4. Ra ci si può collegare e poi avviare il comando 6to4 manualmente:

   ```
   # /usr/pkg/sbin/6to4.pl start
   ```

   A collegamento effettuato, usare il comando ping6(8) per verificare che tutto funzioni correttamente. In caso afermativo, inserire le linee seguenti nello script `/etc/ppp/ip-up` che viene lanciato ogni volta che ci si collega:

   ```
   logger -p user.info -t ip-up Configuring 6to4 IPv6
   /usr/pkg/sbin/6to4.pl stop
   /usr/pkg/sbin/6to4.pl start
   ```

5. Per avere il routing IPv6 anche sulla rete interna, si può indicare a **6to4.pl** di impostare automaticamente il Router Advertising:

   ```
   # /usr/pkg/sbin/6to4 rtadvd-start
   ```

   Anche questo comando si può mettere nello script `/etc/ppp/ip-up` per renderlo permanente.

6. Se si è combiato `/etc/ppp/ip-up` per impostare automaticamente 6to4, conviene cambiare anche `/etc/ppp/ip-down` per fermarlo quando si chiude il collegamento e si passa offline. I comandi da inserire in `/etc/ppp/ip-down` sono:

   ```
   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

Al momento non sono ancora disponibili molti gateway 6to4. Una lista di quelli utilizzabili si trova in http://www.kfu.com/~nsayer/6to4/: naturalmente conviene scegliere il più vicino. Nei test sembra che solo 6to4.kfu.com e 6to4.ipv6.microsoft.com funzionino. Cisco ha un gateway presso il quale occorre registrarsi prima di poterlo usare; vedere http://www.cisco.com/ipv6/.

C'è anche un server sperimentale 6to4 in Germania, 6to4.ipv6.fh-regensburg.de. Questo server usa NetBSD 1.5 ed è stato impostato usando il metodo descritto in questo capitolo. La configurazione della macchina si può vedere all'indirizzo http://www.feyrer.de/IPv6/netstart.local.

10.2.4.10. Conclusione e approfondimenti

Se lo si confronta con il livello raggiunto oggi da IPv4, IPv6 sta ancora muovendo i suoi primi passi: funziona, molti servizi e client sono già disponibili; manca ancora una base di utenti. Le informazioni contenute in questo capitolo possono costituire un punto di partenza per consentire di iniziare a sperimentare e utilizzare IPv6.

Alcuni link utili:

- Uno script di esempio per impostare 6to4 su macchine BSD si può trovare su http://www.netbsd.org/packages/net/6to4/. Lo script determina l'indirizzo IPv6 della macchina e imposta, se lo si desidera, il Router Advertising. Lo script è stato progettato pensando agli ambienti con conessione dial-up, con indirizzo locale che cambia di volta in volta.
- Le implementazioni di BSD Unix hanno la propria documentazione; i link più interessanti si possono trvare su http://www.netbsd.org/Documentation/network/ipv6/ per NetBSD, http://www.freebsd.org/doc/en_US.ISO_8859-1/books/handbook/ipv6-implementation.html per FreeBSD e le pagine 61 e 62 della Administrator's Guide di BSD/OS su http://www.bsdi.com/products/internet/release-notes/rn42.pdf.
- I progetti che lavorano per implementare gli stack del protocollo IPv6 sono KAME per BSD e USAGI per Linux. I relativi siti sono http://www.kame.net/ e http://www.linux-ipv6.org/. Una lista di host e router si trova su http://playground.sun.com/pub/ipng/html/ipng-implementations.html.

- Oltre all’archivio ufficiale RFC all’indirizzo ftp://ftp.isi.edu/in-notes, si possono trovare informazioni su IPv6 su molti siti web. Il primo e più importante è la pagina web di 6Bone su http://www.6bone.net/ 6Bone è stato creato per servire come banco di prova per IPv6, ed è ora una parte importante del mondo degli utenti IPv6. Altre pagine web contenenti informazioni su IPv6 sono http://www.ipv6.org/, http://playground.sun.com/pub/ipng/html/ e http://www.ipv6forum.com/. La maggior parte di questi siti, che val la pena di visitare, contengono ulteriori utili link.

Capitolo 11. Il Domain Name System

In questo capitolo ci occuperemo del DNS (Domain Name System). Utilizzando come esempio la rete descritta nel Capitolo 10 verrà configurato un name server locale.

Nel Capitolo 10 abbiamo visto come impostare i file di configurazione di una semplice rete locale. Uno dei problemi da affrontare era la traduzione dei nomi degli host della rete in indirizzi IP e viceversa. Nel contesto del capitolo in questione il problema veniva risolto utilizzando il file `/etc/hosts`, nel quale erano indicate le corrispondenze tra host e indirizzi IP e che doveva essere presente (e aggiornato) su ognuno degli host della rete stessa. Per esempio:

```
192.168.1.1 ape.insetti.net ape
192.168.1.2 vespa.insetti.net vespa
```

Un ulteriore file, `/etc/nsswitch.conf`, indicava al sistema l'ordine in cui consultare le risorse per risolvere i nomi degli host.

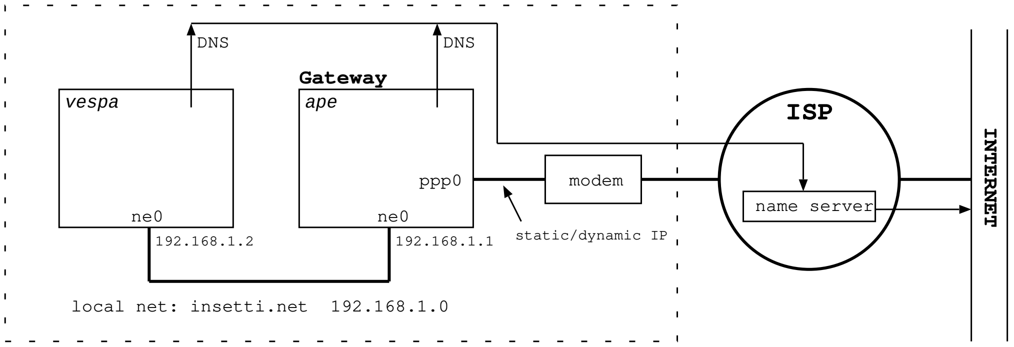
```
hosts:    files dns
```

Con questo sistema il file `/etc/hosts` veniva consultato per primo e solo in caso di fallimento veniva fatto ricorso al DNS. Poiché la rete non aveva un proprio name server, venivano utilizzati quelli forniti dal provider, specificandoli nel file `/etc/resolv.conf` con la direttiva "nameserver". In pratica il file `/etc/hosts` veniva usato per la risoluzione dei nomi della rete locale (non visibili all'esterno) e i server DNS del provider consentivano di risolvere i nomi appartenenti a Internet.

> **Nota:** col il termine *risolvere* si indica la conversione da nome a indirizzo IP e viceversa.

La situazione è illustrata nella Figura 11-1: ape è il gateway della rete e la direttiva nameserver dei file `/etc/resolv.conf` di entrambi gli host punta al name server del provider. Tutte le interrogazioni DNS vengono indirizzate a quest'ultimo. I nomi degli host locali sono risolti esaminando il file `/etc/hosts`.

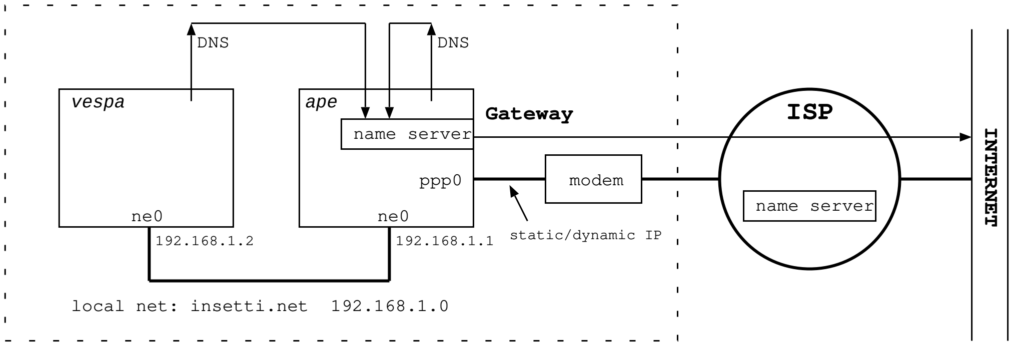
Figura 11-1. Uso del DNS del provider



In questo capitolo si vedrà come impostare un server DNS in locale, senza bisogno di ricorrere ai server del provider, che consenta di risolvere ogni tipo di nome, sia interno sia esterno e di evitare il ricorso al file hosts. Verranno descritti due tipi di approcci, il cui significato sarà chiarito nel seguito: un *caching* *server* e un server vero e proprio. L'ambiente in cui verrà configurato il name server sarà lo stesso di quello descritto nel Capitolo 10.

La situazione che si vuole ottenere è illustrata dalla Figura 11-2. Il name server si trova sull'host ape e viene utilizzato per risolvere sia i nomi interni sia quelli esterni. Il file `/etc/resolv.conf` di ape punta a se stesso mentre quello di vespa punta a ape. Il name server del provider non viene utilizzato (a meno di non usare la direttiva *forwarders* illustrata nel seguito).

Figura 11-2. Uso del DNS interno



> **Nota:** ci si può chiedere quando è opportuno seguire l'approccio descritto nel Capitolo 10 e quando conviene utilizzare un proprio server DNS. Come in molti altri casi, non esiste una risposta univoca. In generale, per computer isolati o per casi di reti locali molto semplici, come quella descritta in precedenza, che si appoggiano sui server DNS di un provider conviene seguire l'approccio già descritto, riservando l'uso di un proprio server DNS per situazioni più complesse. Al termine della lettura del capitolo dovrebbe essere abbastanza chiaro quand'è che l'uso di un server locale può essere più comodo.

11.1. Che cos'è il DNS?

Quando si fa riferimento a un host è decisamente più comodo utilizzare un nome del tipo www.netbsd.org che un indirizzo IP numerico quale 110.111.112.113. Al livello di comunicazione più basso, però, non vengono usati i nomi bensì gli indirizzi in forma numerica; appare quindi evidente l'esigenza di un servizio di traduzione dei nomi in indirizzi numerici e viceversa. Il DNS (Domain Name System) è precisamente questo (e anche di più). Si tratta di un database gerarchico e distribuito; *gerarchico* significa che lo spazio dei nomi degli host non è "piatto" ma che esiste una gerarchia di nomi. Ciò rende possibile l'esistenza di due host con lo stesso nome purché appartengano a domini diversi; se così non fosse, potrebbe esserci un solo host di nome www collegato a Internet mentre, con una struttura gerarchica, www.netbsd.org e www.mydomain.org sono due host diversi. *Distribuito* significa che il database DNS non è concentrato tutto in un punto (ossia non esiste un host che conosce i nomi e gli indirizzi di tutti gli altri host) ma è "spezzettato": ogni server possiede le informazioni relative al proprio dominio. Pertanto, per poter svolgere il loro compito è necessario che i server possano mettersi in contatto tra loro per scambiarsi informazioni.

L'applicativo di riferimento per quando riguarda la gestione del DNS è BIND (Berkeley Internet Name Domain System), un software liberamente distribuito dall'ISC (Internet Software Consortium) che implementa il protocollo DNS. BIND fa parte della distribuzione base di NetBSD e pertanto non è necessario installare alcun software aggiuntivo. La distribuzione di BIND comprende vari componenti, tra i quali il server DNS, rappresentato dal demone named, e alcuni programmi di utilità quali host, dig, nslookup e dnsquery. Inoltre fanno parte della distribuzione di BIND anche le funzioni di libreria per la risoluzione delle query DNS.

> **Nota:** nella collezione dei package è generalmente compresa una versione di BIND più recente di quella installata sul sistema. L'uso di questa versione non è solitamente necessario a meno di esigenze specifiche.

11.1.1. Come funziona un'interrogazione DNS?

Per effettuare un'interrogazione DNS un applicativo contatta un server DNS il quale provvede a sua volta a contattare i server responsabili per il dominio al quale si riferisce il nome da risolvere. In realtà l'interrogazione segue un percorso di tipo gerarchico, partendo dai server di livello più alto, i cosiddetti *root servers*, fino a arrivare al livello di dettaglio desiderato. Per esempio, supponiamo di voler trovare l'indirizzo IP di www.netbsd.org. Per prima cosa il nostro name server contatta un root server, il quale gli fornisce gli indirizzi dei server per il dominio "org" (org è uno dei cosiddetti TLD, ossia Top Level Domain, che sono i domini di livello più alto, quali "edu", "org", "com", ecc.). Il nostro server quindi contatta uno dei server per il dominio org, il quale a sua volta gli fornisce gli indirizzi dei server per il dominio netbsd.org. Infine il nostro server contatta uno dei server di netbsd.org per sapere l'indirizzo di www, e qui si conclude il giro di domande e risposte.

Questo esempio dovrebbe aver chiarito cosa si intende con "gerarchico e distribuito": il nome dell'host è stato analizzato secondo una precisa gerarchia e le interrogazioni sono state effettuate rivolgendosi di volta in volta ai server che erano a conoscenza delle varie parti.

> **Nota:** si noti che per il DNS l'ordine gerarchico dei nomi procede da destra a sinistra, ossia in www.netbsd.org, org è il livello più alto (il TLD, quello immediatamente sotto ai root server), netbsd appartiene al dominio org e www appartiene al dominio netbsd.org.

11.2. Impostare un caching server

Con il termine *caching server* si intende un server che non gestisce in proprio la conversione di nomi di host in indirizzi IP, ma fa sempre riferimento a altri server. La particolarità di questo tipo di server è che memorizza le risposte che gli giungono ed è quindi in grado di effettuare le interrogazioni successive più velocemente. Poiché un server di questo tipo non gestisce i nomi interni della nostra rete, per ricavare gli indirizzi di quei nomi sarà ancora necessario fare riferimento al solito file `/etc/hosts`.

Poiché NetBSD fornisce già una serie di file di configurazione predefiniti, l'impostazione di un server di questo tipo è a dir poco elementare, come si vedrà tra breve.

> **Nota:** il contenuto e il numero dei file di configurazione può variare a seconda della versione di NetBSD.

Il server vero e proprio è costituito dal demone named, il quale fa riferimento a un file di configurazione che, di default è `/etc/named.conf`. Il file di configurazione d'esempio di NetBSD si trova nella directory `/etc/namedb` e, pertanto, come prima cosa creiamo un link al file:

```
# ln -s /etc/namedb/named.conf /etc/named.conf
```

Il nostro name server è pronto per l'uso; il passo seguente sarà di effettuare dei test per verificarne il corretto funzionamento, ma prima dobbiamo indicare al sistema di usare il nostro server, inserendo nel file `/etc/resolv.conf` una riga del tipo:

```
nameserver 127.0.0.1
```

A questo punto tutto è pronto per avviare named.

```
# named
```

> **Nota:** il name server è stato qui avviato manualmente. Una volta verificato il corretto funzionamento è possibile riavviarlo automaticamente al boot agendo sull'apposita opzione del file `/etc/rc.conf`.

11.2.1. Test del server

Avviamo **nslookup** per esaminare la situazione e verificare il buon funzionamento del server.

```
# nslookup
Default server: localhost
Address: 127.0.0.1

>
```

Adesso effettuiamo un'interrogazione su (a titolo di esempio) www.mclink.it.

```
> www.mclink.it
Server:  localhost
Address:  127.0.0.1

Name:    www.mclink.it
Address:  195.110.128.8
```

Ripetendo l'interrogazione una seconda volta si ottiene un risultato leggermente diverso:

```
> www.mclink.it
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
Name:    www.mclink.it
Address:  195.110.128.8
```

Come si vede i dati sono gli stessi ma ora è comparsa la scritta “Non-authoritative answer”, che indica che il server non è uscito sulla rete per reperire l’indirizzo ma ha utilizzato i dati che aveva nella cache dopo la prima interrogazione. La risposta non proviene quindi dal server master del dominio mclink.it, come nel caso precedente, bensì dalla cache del nostro server.

Questi primi risultati ci confortano del buon funzionamento del server.

Può essere una buona occasione per provare anche il comando **host**.

```
# host www.mclink.it
www.mclink.it has address 195.110.128.8
```

11.3. Un name server per il dominio locale

Ne la Sezione 11.2 si è visto come impostare un caching server, il che è risultato essere piuttosto semplice. Proviamo ora a chiedere al server qualche informazione su un host della nostra rete locale.

```
# 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
```

Il server non sa nulla del nostro host locale. Infatti i dati relativi sono ancora confinati nel file `/etc/hosts`, che non viene consultato dal server. Pertanto il name server effettua il solito giro di interrogazioni partendo dai root server ma naturalmente non viene a capo di nulla. Ciò non significa che l’impostazione del server è sbagliata ma indica soltanto che la risoluzione dei nomi locali potrà essere effettuata solo tramite il file `/etc/hosts`.

Nel seguito del capitolo si vedrà come far gestire al name server il nostro dominio locale; per far ciò occorrerà sia modificare `named.conf` sia aggiungere i *file di zona*, cioè i file che contengono i record DNS che descrivono gli host (e altre caratteristiche) della rete locale. In realtà nella configurazione attuale sono già presenti dei file di zona, che si trovano nella directory `/etc/namedb`, ma su questo torneremo in un secondo tempo.

11.4. I file di configurazione di named

Il demone named legge due tipi di file di configurazione; il primo è il già citato `named.conf`, contenente le opzioni di configurazione del programma. Il secondo tipo di file è costituito dai cosidetti file di zona, che contengono i record DNS per gli host della rete locale. I record DNS consentono di trasformare i nomi degli host in indirizzi IP e viceversa, di definire i server di posta e di specificare molte altre informazioni. In questo capitolo non veranno descritti in dettaglio né il file di configurazione né i file di zona, ma ci si limiterà a presentare e commentare degli esempi che dovrebbero essere facili da modificare e da adattare.

> **Nota:** è possibile che nel file `named.conf` fornito con NetBSD siano presenti anche altre sezioni, per esempio per IPv6.

11.4.1. Named.conf

L’esempio seguente mostra una configurazione minima per il file `named.conf`. Gran parte del contenuto era già presente nel file di default di NetBSD; solo le righe dalla 17 alla 25 sono state aggiunte per gestire il dominio locale.

> **Nota:** i numeri di riga e il carattere “|” all’inizio della riga non fanno parte del file e sono presenti solo come riferimento.

Esempio 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| };
```

Già a una prima occhiata si nota che il file è diviso in sezioni (o più precisamente statement) distinte, con i corpi delle varie sezioni racchiusi tra parentesi graffe. Vediamo ora in dettaglio il significato delle varie parti del file (per una descrizione completa si rimanda alla pagina di manuale named.conf(5)).

Linea 1

Le linee che iniziano con il carattere “#” sono considerate dei commenti e vengono ignorate, al pari delle linee vuote. Vengono riconosciuti anche i comenti in stile C e C++.

Linee 3-5

La sezione *options* controlla le opzioni di configurazione globale del server e i default che verranno usati nelle altre sezioni del file. La direttiva *directory* definisce la directory di lavoro del server; tutti i percorsi non assoluti presenti nelle altre sezioni del file di configurazione verranno interpretati come relativi a questa directory. Per esempio, il file `root.chache` (linea 7) verrà cercato in `/etc/namedb/root.cache`.

Linee 7-10

Questa sezione definisce una zona. Si tratta di una zona speciale, predefinita, contenente i già citati root server, quelli dalle quali partono le ricerche. La direttiva *hint* indica che questa zona è proprio quella dei root server e la linea 7 specifica il nome del file in cui si trovano i dati della zona (il file è già cortesemente fornito da NetBSD).

Linee 12-15

Questa sezione definisce la zona inversa locale per l'indirizzo di loopback (127.0.0.1). Vedere la spiegazione delle righe 22-25.

Linee 17-20

Definizione della zona relativa al dominio locale. L'opzione "type master" indica che il server è il *master server* per questa zona, cioè è quello che ha la copia master del file di zona (nel file indicato alla linea 19). Il file di zona contiene i record DNS per effettuare la conversione da nome host a indirizzo IP nonché altre informazioni che saranno esaminate nel seguito.

Si noti che il nome della zona deve corrispondere al nome del dominio mentre il nome del file contenente i record DNS è libero anche se, normalmente, conviene attenersi a qualche convenzione.

Linee 22-25

Queste linee definiscono la *zona inversa* per il dominio locale, cioè il file che contiene i record DNS per effettuare la conversione da indirizzo IP a nome host. Anche in questo caso il server è di tipo master. La linea 24 indica il nome del file contenente i record DNS.

Il nome della zona merita una spiegazione. La prima parte è l'indirizzo IP della rete scritto al contrario (192.168.1 diventa 1.168.192), mentre la seconda si riferisce al dominio predefinito "in-addr.arpa" che serve per la *risoluzione inversa*, cioè la trasformazione degli indirizzi IP in nomi.

Server master e server slave: nell'esempio precedente il nostro server era di tipo master per tutte le zone, cioè era il server che possedeva la copia master dei file di zona. Un server di tipo slave, invece, possiede una copia (replica) dei file di zona del relativo master (che non viene scritta a mano dall'amministratore di sistema come la copia master). Tra server master e slave esistono dei metodi di sincronizzazione automatica la cui descrizione esula dagli scopi di questa breve introduzione.

La configurazione di `named.conf` è terminata; nelle sezioni seguenti esamineremo il contenuto dei file di zona, descrivendo i record DNS. I file per le zone "." e "127" sono predefiniti e non richiedono modifiche e non verranno pertanto esaminati. La loro struttura è comunque la stessa dei file per il dominio locale.

Prima di passare all’esame dei file di zona vediamo un’ulteriore direttiva che può essere inserita nella sezione options, e cioè *forwarders*. Per esempio:

```
options {
        directory "/etc/namedb";
        forwarders {
         1.2.3.4;
  1.2.3.5;
        };
};
```

Nell’esempio precedente si supponga che 1.2.3.4 e 1.2.3.5 siano i name server del provider. L’effetto di questa impostazione è che il nostro server effettuerà le interrogazioni servendosi dei name server del provider; in questo modo saranno loro a farsi carico del caching delle informazioni anziché il server locale. Nel caso in cui i forwarders non riescano a rispondere a un’interrogazione, allora il server cercherà di trovare la risposta per conto proprio. Questa impostazione riduce il carico di lavoro del server locale, che si limita a gestire in proprio solo le zone per cui è master (in definitiva il dominio locale).

11.4.2. Il file per la zona insetti.net

Per come è stato definito in `named.conf`, il file per la zona insetti.net si chiama `/etc/namedb/insetti.net`. Il suo contenuto è il seguente:

```
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
```

Il file precedente è il vero e proprio database DNS, contenente i *record DNS*. Nel seguito di questa sezione esaminiamo i record presenti nel file.

Linea 1

Questa linea è un commento.

Linee 3-8

Questo gruppo di linee definisce il record SOA (Start Of Authority) che delimita l’inizio di una zona di autorità. Ogni zona ha un solo record SOA.

Il carattere @ all’inizio della riga è un sinonimo per il nome del dominio specificato nella direttiva “zone” del file `named.conf` e quindi in questo caso equivale a “insetti.net”.

Il successivo “IN” indica sta per Internet e definisce il tipo di rete.

Dopo SOA vengono, nell’ordine, il name server e l’indirizzo e-mail del responsabile per la zona (nell’indirizzo e-mail il carattere @, che per il DNS ha un significato particolare, deve essere sostituito con un punto).

> **Nota:** si noti che entrambi gli indirizzi precedenti sono terminati da un punto. Tutti i nomi non terminati da un punto sono considerati *relativi* al dominio corrente e il server li completa aggiungendo il dominio (@) alla fine. Pertanto, nei file di zona bisogna fare attenzione a come si scrivono i nomi: se, per esempio, si scrive “ape.insetti.net” (anziché “ape.insetti.net.”) il server internamente tramuta il nome in “ape.insetti.net.insetti.net”, che non è certo quel che si vuole ottenere.
>
> Il record SOA continua, racchiuso tra parentesi tonde, fino alla linea 8. La riga 4 definisce il numero di serie del file. Il numero può essere scelto a piacere a condizione che venga incrementato ogni volta che si modifica il file di zona. In questo caso è stato scelta una codifica che si basa sulla data di ultima modifica del file espressa nel formato anno-mese-giorno seguita a un numero seriale. Per esempio, se il file verrà nuovamente modificato il 03/12/2000 il numero diventerà 2000120302, mentre se verrà modificato due giorni dopo diventerà 2000120501. È molto importante ricordarsi di incrementare questo numero di serie quando si modifica il file.
>
> Le linee dalla 5 alla 8 contengono dei tempi in secondi che si riferiscono all’interazione tra il master e i suoi slave. Ai fini di questa introduzione non interessa esaminarle in dettaglio e prenderemo per buoni i valori di default dell’esempio.

Linea 9

La riga 9 è un record di tipo *NS*, che indica qual’è il name server autoritario per la zona. Può esserci più di una riga di tipo NS. Si noti che il nome è terminato da un punto.

Linea 10

Questa linea contiene un record di tipo *MX*, che contribuisce a velocizzare il trasferimento della posta indicando qual’è (o quali sono) il server di posta della zona. Il numero “10” che compare dopo NS esprime una preferenza: minore è il numero, maggiore è la preferenza data al server; in caso di presenza di più record MX viene utilizzato per primo quello con il numero più basso. Non importa il valore del numero: conta solo il rapporto maggiore-minore con i numeri degli altri record MX. Nell’esempio seguente il mail hub ape verrà provato prima di vespa.

```
IN MX  10 ape
 IN MX  20 vespa
```

Si noti che poiché dopo il nome dell’host non compare il punto, al nome stesso sarà concatenato il dominio.

Linee 11-12

Finalmente siamo arrivati ai record che vengono usati per trasformare i nomi degli host in indirizzi IP (ma non viceversa: quello è il compito del file per la zona 1.168.192). I due record di tipo *A* associano gli indirizzi ai nomi dei due host della nostra rete. Se fossero presenti altri host andrebbero aggiunti altri record A. Si noti che i nomi degli host non sono seguiti dal punto.

11.4.3. Il file per la zona 1.168.192.in-addr.arpa

Per come è stato definito in `named.conf`, il file per la zona 1.168.192.in-addr.arpa si chiama `/etc/namedb/1.168.192`. Il suo contenuto è il seguente:

```
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.
```

La prima parte di questo file è del tutto equivalente a quella del file per la zona insetti.net e quindi non la esamineremo di nuovo. Le uniche novità sono le righe 8 e 9, che hanno la funzione inversa ai record A. Si tratta dei record PTR, che permettono di trasformare gli indirizzi IP in nomi di host. Alla luce di quanto detto fin qui il loro significato dovrebbe essere abbastanza chiaro. Si noti che i nomi degli host sono terminati dal punto.

All'inizio della riga 8 compare il numero “1”. Poiché non è seguito da un punto, il DNS gli concatena il nome della zona (1.168.192.IN-ADDR.ARPA), ottenendo 1.1.168.192.IN-ADDR.ARPA, che è l'indirizzo completo da risolvere. Stesso discorso vale per la riga 9, dove si ottiene 2.1.168.192.IN-ADDR.ARPA.

In-addr.arpa: Il dominio predefinito (si tratta di un TLD) in-addr.arpa serve per trasformare gli indirizzi IP in nomi di host. All'interno di questo dominio gli indirizzi IP devono essere scritti con le stesse convenzioni dei nomi degli host, e cioè con la parte più significativa a destra. Per questo motivo 192.168.1.2 diventa 2.1.168.192.IN-ADDR.ARPA. In tal modo BIND può analizzare questo nome esattamente come fa per i nomi degli host, procedendo da destra verso sinistra.

11.4.4. Altri tipi di record per i file di zona

Negli esempi precedenti nei file di zona sono stati inseriti solo i record strettamente necessari per il funzionamento della configurazione che ci eravamo proposti. In realtà esistono molti altri tipi di record che permettono di specificare in maggiore dettaglio e con più flessibilità la struttura della nostra rete. Nel seguito di questa sezione vediamo alcuni altri tipi di record ma, per una descrizione più completa e approfondita, si rimanda alla letteratura specializzata.

I record *CNAME* servono per creare alias di nomi esistenti, ossia per assegnare ulteriori nomi a un host. Per esempio:

```
www      IN       CNAME   ape
ftp      IN       CNAME   ape
```

In questo modo abbiamo indicato a BIND che i nomi www e ftp fanno entrambi riferimento all'host ape.

I record *TXT* servono per aggiungere testo descrittivo ai record DNS. Per esempio:

```
IN TXT "La mia rete locale"
```

11.5. Avviare il name server

Il server si avvia come già spiegato per il caching server. Se il server fosse già attivo, ora che i dati di configurazione sono stati modificati occorre riavviarlo, cosa che si effettua con il comando **ndc** (name daemon control program).

```
# ndc reload
```

Ora che il name server è attivo, non è più necessario inserire i nomi degli host nel file `/etc/hosts` perché ci pensa named a risolvere i nomi. Naturalmente tutti gli host della rete devono "puntare" al name server usando il file `/etc/resolv.conf`. In particolare su ape avremo:

```
nameserver 127.0.0.1
```

mentre su vespa avremo:

```
nameserver 192.168.1.1
```

Capitolo 12. Posta e news

In questo capitolo viene spiegato come impostare il sistema per l'uso della posta e delle news, un argomento che, a causa della sua complessità, se viene trattato in forma generale rischia di diventare incomprensibile per chi vi si accosta per la prima volta. Questo è dovuto al fatto che esiste una grande molteplicità di situazioni possibili (sia per quanto riguarda le tipologie di collegamento sia per quanto riguarda le modalità di lavoro) e che non si tratta di configurare un singolo programma ma di mettere insieme un sofisticato gruppo di programmi che cooperano al funzionamento del sistema. Per questo motivo l'argomento viene trattato illustrando un esempio che descrive una possibile configurazione per un PC di uso domestico collegato a Internet tramite un provider, una situazione molto comune che, pur non presentando particolari difficoltà, può rivelarsi abbastanza spinosa da gestire se non si sa dove mettere le mani.

Oltre alle impostazioni di sistema, viene descritta la configurazione necessaria per due tra i più diffusi e apprezzati programmi per la posta e le news: mutt e tin. I programmi in se stessi non vengono descritti, anche perché sono semplici da usare e ben documentati. Naturalmente, sia mutt sia tin non sono scelte obbligate e l'utente è libero di installare gli applicativi che preferisce; in particolare, chi non ama i programmi in modo testo potrà trovare valide alternative sotto X.

In breve, i programmi richiesti per la configurazione descritta in questo capitolo sono:

- sendmail
- fetchmail
- m4
- mutt
- tin

Di questi, solo sendmail e m4 sono già presenti nel sistema base; gli altri possono essere installati dalla collezione dei package.

Prima di proseguire, vorrei ricordare ancora una volta che nessuno dei programmi presentati in questo capitolo è una scelta obbligata; esistono numerose altre alternative a ognuno di essi ma, comunque, costituiscono un buon punto di partenza. Anche la strategia utilizzata per l'invio e la ricezione della posta non è l'unica. Ognuno poi la può modificare a seconda della propria configurazione e delle proprie esigenze.

All'estremo opposto dei programmi e della strategia illustrati in questo capitolo c'è l'utilizzo di un programma tuttofare come Netscape Communicator: con questo tipo di programma si può navigare su Internet, ricevere e inviare posta e leggere le news; inoltre la sua configurazione è molto semplice. Lo svantaggio di questo tipo di approccio è che Netscape è un sistema chiuso, che non coopera con gli altri programmi di NetBSD.

Un altro approccio molto seguito è quello di leggere la posta e le news utilizzando le apposite modalità di emacs. Emacs non ha certo bisogno di presentazioni per gli appassionati di Unix ma, per chi non lo sapesse, si tratta di un editor avanzato (veramente chiamarlo editor è un po' riduttivo) che si trasforma in un vero e proprio ambiente di lavoro all'interno del quale è possibile leggere la posta e le news ed effettuare molte altre operazioni. La configurazione di emacs per la posta e le news è descritta nel manuale del programma.

Nel seguito di questo capitolo si farà riferimento a un host collegato a Internet tramite un collegamento seriale PPP via modem con un provider. I dati per la configurazione sono:

- l'host locale si chiama "ape" e la rete è "insetti.net", per cui il FQDN (Fully Qualified Domain Name) è "ape.insetti.net".
- Il nome dell'utente id esempio su ape è "carlo".
- Il provider utilizzato si chiama BigNet.
- Il server di posta del provider è "mail.bignet.it"
- Il nome di login presso il provider dell'utente "carlo" è "alan" e la sua password è "pZY9o".

Prima di incominciare, un po' di terminologia:

MUA

(mail user agent): programma che permette di scrivere e leggere la posta. Per esempio: mutt, elm e pine ma anche il semplice mail preinstallato.

MTA

(mail transfer agent): programma che provvede a inoltrare la posta, trasferendola da un computer a un altro ma anche in locale. Il MTA è il programma che decide il percorso che la posta deve seguire per arrivare a destinazione. Sui sistemi BSD (ma non solo) lo standard è sendmail.

MDA

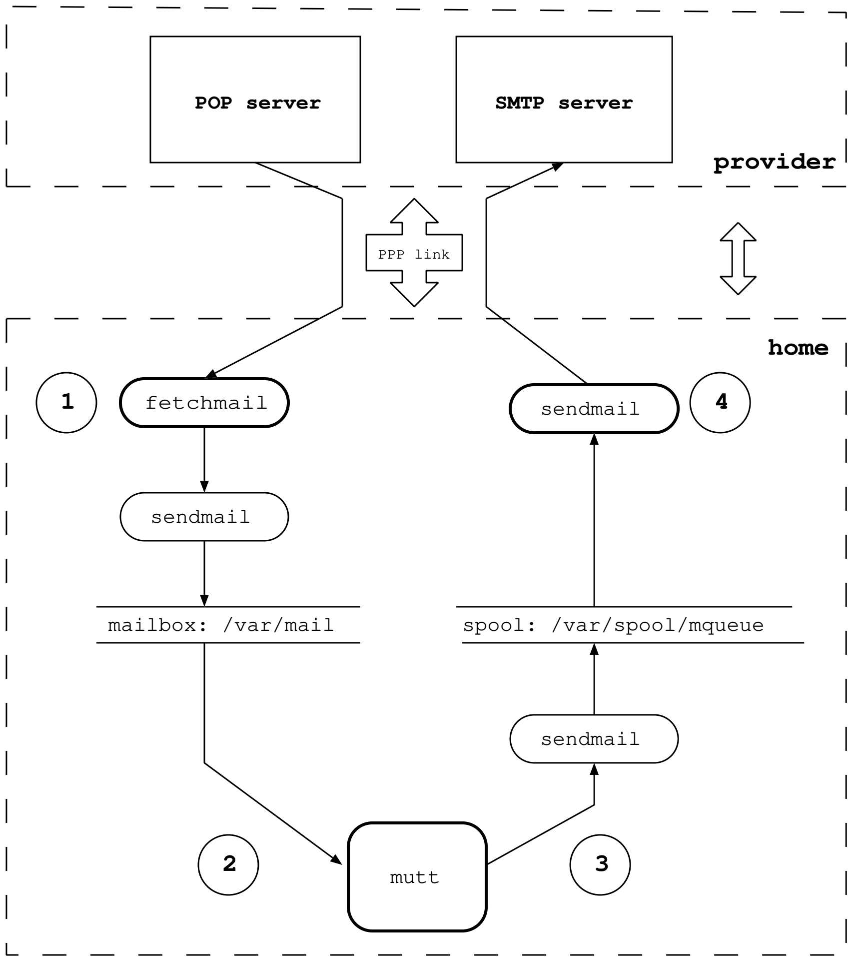
(mail delivery agent): programma, solitamente usato dal MTA, che serve per la consegna della posta: per esempio per mettere fisicamente i messaggi nella casella postale dell'utente destinatario. Per esempio, sendmail utilizza uno o più MDA per consegnare la posta.

La figura Figura 12-1 illustra il sistema di posta che si vuole organizzare. Tra la rete domestica (o anche l'host singolo) e il provider vi è una connessione PPP via modem. Le "bolle" con il bordo spesso si riferiscono a comandi lanciati esplicitamente dall'utente; le restanti "bolle" sono i programmi lanciati in automatico. I numerini cerchiati si riferiscono alle varie fasi logiche del processo di gestione del ciclo della posta:

1. Nella fase (1) la posta viene recuperata (scaricata) dal server POP del provider mediante fetchmail, che usa sendmail per metterla nella casella postale dell'utente.
2. Nella fase (2) l'utente esegue mutt (o un altro client di sua scelta) per leggere la posta, rispondere ai messaggi ricevuti e scriverne di nuovi.
3. Nella fase (3) l'utente invia la posta da mutt. I messaggi vengono accumulati nell'area di spool.
4. Nella fase (4) l'utente chiama sendmail per trasferire i messaggi al server SMTP del provider da dove verranno spediti ai destinatari.

Il collegamento con il provider deve essere attivo solamente durante le fasi (1) e (4); per le restanti operazioni può tranquillamente essere chiuso.

Figura 12-1. Struttura del sistema di posta



12.1. sendmail

Quando un MTA, sendmail in questo caso, riceve un messaggio di posta da spedire, se si tratta di un messaggio locale lo spedisce direttamente. Altrimenti, se il messaggio è destinato a un altro dominio deve procurarsi l'indirizzo del server di posta del dominio in questione. Per fare ciò il MTA si appoggia al servizio DNS (descritto nel Capitolo 11; il server DNS del dominio di destinazione conosce l'indirizzo del server di posta (record MX) e lo fornisce al MTA che, a questo punto, può spedire il messaggio al server destinatario.

Il *MTA* più diffuso, almeno in ambiente BSD, è sendmail. Il comportamento di sendmail è controllato dal file di configurazione, `/etc/mail/sendmail.cf` e da una serie di file accessori. In generale, a meno di non essere degli esperti, non è consigliabile scrivere o modificare questo file direttamente; il file di configurazione si può generare automaticamente utilizzando un insieme di macro M4 che facilita molto il compito.

> **Nota:** nelle versioni di NetBSD precedenti la 1.5 i file di configurazione di sendmail si trovavano in `/etc` anziché in `/etc/mail`.

Anche utilizzando le macro, la configurazione di sendmail è un argomento piuttosto ostico (a dir poco) e nel seguito mi limito a mostrare un esempio base, che può essere usato come punto di partenza e modificato per gestire la propria posta in presenza di configurazioni differenti. Per un collegamento a Internet via modem, il file presentato può essere usato tale e quale senza bisogno di alcuna modifica (salvo naturalmente inserire i propri dati al posto di quelli del'esempio).

Il primo problema da risolvere è che la rete domestica è del tutto fittizia, ossia non esiste realmente su Internet, e quindi i nomi e gli indirizzi interni non hanno senso per il mondo esterno. In breve, l'host "ape.insetti.net" non è raggiungibile da nessun altro host di Internet e se si manda un'e-mail con il nome locale dell'host nel mittente nessuno potrà rispondere (non solo, ma alcuni destinatari rifiuteranno il messaggio perché il mittente è inesistente). Il vero indirizzo, quello visibile da tutti, è quello assegnato dal provider. Quindi bisognerà trasformare l'indirizzo locale "carlo@ape.insetti.net" nell'indirizzo assegnato dal provider: "alan@bignet.it". Di questo si occuperà sendmail, opportunamente configurato, al momento di trasferire la posta.

Il secondo aspetto da considerare, è che conviene istruire sendmail per fargli inviare la posta al mail server del provider. Nella configurazione descritta in questa sezione, infatti, sendmail non si mette direttamente in contatto con i server di posta dei destinatari dei nostri messaggi (come descritto all'inizio di questa sezione), ma si appoggia solo sul server del provider, utilizzandolo come *relay*. Questo rende molto più veloce l'invio della posta, demandando tutto il lavoro di ricerca e di collegamento al server di posta del provider.

> **Nota:** si dice che un server di posta agisce da *relay* quando si occupa di inoltrare posta che non è di sua competenza diretta. In questo caso il server di posta relay agisce da intermediario, provvedendo a inoltrare la posta al server destinatario (o a un altro server relay).

Poiché la connessione con il provider non è sempre attiva si può disabilitare l'avvio di sendmail come demone in `/etc/rc.conf` con la riga 'sendmail=NO'. La posta locale verrà comunque spedita correttamente, mentre per trasferire la posta al provider sarà necessario lanciare sendmail manualmente (quando il collegamento con il provider è attivo, naturalmente).

Incominciamo quindi il lavoro di configurazione di sendmail.

12.1.1. Configurazione con genericstable

Questo tipo di configurazione utilizza il file `/etc/mail/genericstable` contenente le regole di riscrittura esplicite per gli indirizzi interni alla rete locale.

Per prima cosa, quindi, bisogna creare il file `genericstable`. Per esempio:

```
carlo:          alan@bignet.it
root:           alan@bignet.it
news:           alan@bignet.it
```

Per motivi di efficienza genericstable deve essere trasformato con il comando:

```
# /usr/sbin/sendmail -bi -oA/etc/mail/genericstable
```

Ora è il momento di creare il prototipo che verrà utilizzato per generare il nuovo file di configurazione di sendmail. La prima cosa da fare è portarsi nella directory dove verrà creato il nuovo file di configurazione:

```
# cd /usr/share/sendmail/m4
```

Supponiamo di chiamare il prototipo `mycf.mc`. Il contenuto del file è il seguente:

```
divert(-1)dnl
include(`../m4/cf.m4')dnl
VERSIONID(`mycf.mc creato da carlo@ape.insetti.net May 18 2001')dnl
OSTYPE(bsd4.4)dnl

dnl # Impostazioni per masquerading.  Vengono riscritti gli
dnl # indirizzi del 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 # La feature seguente serve soprattutto se sendmail viene usato
dnl # insieme a fetchmail.  Fetchmail chiama sendmail per spedire la
dnl # posta.  Se sendmail non riesce a risolvere il nome del mittente
dnl # la posta non viene consegnata.  Per esempio:
dnl # MAIL FROM:<www-owner@netbsd.org> SIZE=2718
dnl # 501 <www-owner@netbsd.org>... Sender domain must exist
FEATURE(`accept_unresolvable_domains')dnl

dnl # accept_unqualified_senders accontenta alcuni MUA, che inviano
dnl # la posta come
dnl # MAIL FROM:<carlo>
FEATURE(`accept_unqualified_senders')dnl

dnl # La posta per il mailer `smtp' viene messa in coda: sendmail non
dnl # tenta di inviarla direttamente (il flag `e' indica `expensive').
dnl # Sendmail inizia a lamentarsi dopo 16 ore di mancata consegna.
define(`SMTP_MAILER_FLAGS',`e')dnl
define(`confCON_EXPENSIVE',`True')dnl
define(`confTO_QUEUEWARN', `16h')dnl

dnl # Per gli utenti europei
define(`confDEF_CHAR_SET',`ISO-8859-1')dnl

dnl # Abilitare le due linee seguenti per utilizzare procmail per
dnl # la consegna della posta locale.  Vedere anche la riga
dnl # con MAILER(procmail)
dnl # define(`PROCMAIL_MAILER_PATH', /usr/pkg/bin/procmail)dnl
```

```
dnl # FEATURE(local_procmail)dnl

dnl # I due mailer seguenti devono sempre essere definiti
MAILER(local)dnl
MAILER(smtp)dnl

dnl # Abilitare opzionalmente la riga seguente (non e' necessaria
dnl # per usare procmail)
dnl # MAILER(procmail)dnl
```

> **Nota:** nell'esempio precedente, le linee che iniziano con "dnl" sono dei commenti e non sono quindi necessarie per la generazione del file di configurazione (il simbolo "dnl" indica a m4 di ignorare il resto della linea, e compare anche al termine delle varie direttive (allo scopo di evitare eventuali effetti indesiderati).
>
> Le linee relative all'uso di procmail sono lasciate commentate (ossia l'uso di procmail è disabilitato).

Questa configurazione indica di trasformare gli indirizzi del tipo "ape.insetti.net" andando a trovare i nomi reali nel file `/etc/mail/genericstable`. Inoltre viene specificato che la posta deve essere inviata all'host "mail.bignet.it". Il significato delle opzioni è descritto in dettaglio nel file `/usr/share/sendmail/README`.

Per creare il proprio file personalizzato è sufficiente inserire i propri dati modificando solo due righe dell'esempio precedente, e cioè:

```
GENERICS_DOMAIN(ape.insetti.net ape)dnl
define(`SMART_HOST',`mail.bignet.it')dnl
```

Generiamo il nuovo file di configurazione, salvando prima quello precedente:

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.bak
# m4 mycf.mc > /etc/mail/sendmail.cf
```

> **Nota:** in `/usr/share/sendmail/cf` c'è il file `netbsd-proto.mc`, che viene utilizzato per creare la versione `/etc/mail/sendmail.cf` che viene distribuita di default con NetBSD. Non dovrebbe essere mai necessario, ma con il comando **make** lo si può costruire.

Un altro file importante è `/etc/mail/aliases`, che però può essere mantenuto con la configurazione di default. Bisogna comunque dare il comando:

```
# newaliases
```

Ora tutto è pronto per spedire la posta.

12.1.2. Test della configurazione

Adesso che la configurazione di sendmail è completa si può procedere a alcune semplici verifiche, per confermare che tutto funzioni correttamente. Come primo test si può spedire un messaggio in locale:

```
$ sendmail -v carlo
Subject: test

Prova
.
carlo... Connecting to local...
carlo... Sent
```

Seguire lo schema dell'esempio, lasciando una linea vuota dopo Subject: e concludendo il messaggio con una linea contenente un solo punto. Il messaggio inviato dovrebbe essere leggibile utilizzando il client di posta; in particolare si verifichi il contenuto del campo

```
From: alan@bignet.it
```

nel quale ci deve essere stata la riscrittura dell'indirizzo.

Una seconda verifica importante è il test delle regole di riscrittura impostate nel file di configurazione di sendmail. Per fare ciò si esegue sendmail in modalità test: sendmail legge un indirizzo e mostra tutti i passaggi effettuati per trasformarlo. In questa modalità si possono anche effettuare altri tipi di test nonché visualizzare varie informazioni.

Avviare sendmail in modalità test usando l'opzione `-bt`

```
$ /usr/sbin/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Per visualizzare l'aiuto, utilizzare il comando **'?'**.

Per prima cosa verifichiamo il corretto funzionamento del file genericstable dando questo comando:

```
/map generics carlo
map_lookup: generics (carlo) returns alan@bignet.it
```

Fin qui tutto bene: sendmail ha trovato il nome locale e il suo corrispondente reale nella mappa generics.

Ora verifichiamo la riscrittura del mittente nell'envelope dando i comandi:

```
/tryflags ES
/try smtp carlo@ape.insetti.net
```

Il risultato dovrebbe essere simile al seguente:

```
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
>
```

Come si può vedere l’indirizzo locale è stato riscritto in quello reale, che comparirà nei messaggi quando lasciano il sistema.

Un risultato equivalente si può ottenere con il comando:

```
/try smtp carlo
```

Per verificare la riscrittura del mittente nell’intestazione, ripetiamo i comandi precedenti cambiando i flag; il risultato è equivalente.

```
/tryflags HS
/try smtp carlo@ape.insetti.net
```

12.1.3. Usare un altro MTA

A partire dalla versione 1.4 di NetBSD sendmail non viene richiamato direttamente:

```
$ ls -l /usr/sbin/sendmail
lrwxr-xr-x  1 root  wheel  21 Nov  1 01:14 /usr/sbin/sendmail@ -> /usr/sbin/mailwrapper
```

Lo scopo di mailwrapper è di semplificare l’uso di MTA alternativi a sendmail (per esempio postfix) ma, per farsi un’idea precisa di cosa si tratta, consiglio la lettura delle pagine di manuale mailwrapper(8) e mailer.conf(5), che sono molto chiare.

12.2. fetchmail

La posta viene ricevuta all'indirizzo presso il provider; a questo punto è necessario trasferirla (fisicamente) in locale. fetchmail è un programma che recupera la posta da un server di posta remota e la inoltra al sistema locale per la consegna (solitamente mediante sendmail). fetchmail è semplice da usare e da configurare; dopo averlo installato è sufficiente creare il file `.fetchmailrc` nella propria home directory (c'è dentro la password, attenzione ai permessi...). Ecco un esempio:

```
poll mail.bignet.it
protocol POP3
username alan there with password pZY9o is carlo here
flush
mda "/usr/sbin/sendmail -oem %T"
```

> **Nota:** la riga con "mda ..." serve solo se sendmail non è attivo come demone sul sistema. Inoltre si noti che il server di posta specificato in questo file potrebbe essere diverso dal relay usato da sendmail.

Per trasferire la posta basta dare il comando:

```
$ fetchmail
```

e poi la si può leggere con mutt.

12.3. Posta con mutt

mutt è un programma di posta che riceve molti consensi, sia perché è *leggero* sia perché è semplice da usare e potente al tempo stesso. mutt ha una pagina di manuale molto scarna; la vera documentazione si trova in `/usr/pkg/share/doc/mutt/`, in particolare in `manual.txt`.

La configurazione di mutt avviene tramite il file `.muttrc`, che deve essere creato nella propria home directory (di solito copiando il file di esempio fornito con mutt e poi modificandolo). L'esempio seguente mostra come modificare alcune righe del file per ottenere i seguenti effetti:

- Salvare i messaggi inviati.
- Definire una directory per la posta in arrivo e in uscita salvata da mutt (verrà messa nella subdirectory `~/Mail` nei file `incoming` e `outgoing`).
- Definire alcuni colori.
- Definire un 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>
```

Per lanciare mutt è sufficiente dare il comando

```
$ mutt
```

> **Nota:** a seconda del terminale mutt supporta l'uso di colori diversi per i diversi elementi delle schermate. Sotto X si può usare xterm-color. Per esempio:
>
> ```
> TERM=xterm-color mutt
> ```

12.4. Come ricevere la posta

Questa sezione descrive una semplice strategia per ricevere e leggere la posta. La connessione con il provider viene attivata solo per il tempo necessario al trasferimento della posta, che viene poi letta *offline*.

1. Attivare la connessione con il provider.
2. Dare il comando **fetchmail**.
3. Chiudere la connessione con il provider.
4. Leggere la posta con mutt.

12.5. Come inviare la posta

Una volta scritta e inviata la posta con mutt, la si deve inviare al server del provider con sendmail. L'invio della posta da mutt, che si effettua con il comando **y**, si limita a accodare le missive nello spool; a questo punto, se sendmail non è attivo come demone bisogna provvedere a chiamarlo esplicitamente. I passi da seguire sono:

1. Scrivere la posta con mutt, dare il comando di invio e uscire da mutt.
2. Attivare la connessione con il provider.
3. Dare il comando **/usr/sbin/sendmail -q -v** per trasferire la posta al provider.
4. Chiudere la connessione con il provider.

12.6. Strumenti avanzati per la posta

Quando si incomincia a usare la posta, generalmente non si hanno esigenze molto sofisticate e, pertanto, la configurazione standard descritta sopra è più che sufficiente. Con il passare del tempo, però, per molti utenti il volume di posta giornaliera cresce e si differenzia. A questo punto diventa necessario organizzare razionalmente i messaggi, separandoli in "caselle" diverse e ci si accorge di dover compiere un gran numero di operazioni manuali ripetitive ogni giorno per smistare la posta. Per esempio, chi si iscrive a *mailing list* può trovarsi a dover smistare quotidianamente un notevole numero di messaggi, per traferirli dalla mailbox di default all'apposita mailbox dedicata alla mailing list.

Perché effettuare a mano operazioni che possono essere agevolmente svolte da un programma? Esistono vari strumenti aggiuntivi che si inseriscono nel flusso normale della posta e consentono di effettuare varie elaborazioni sulla posta in base a configurazioni prefissate dall'utente. Tra i tool più conosciuti e usati troviamo:

- procmail, un mail delivery agent e filtro avanzato per la posta locale, che consente di elaborare automaticamente la posta in base a regole predisposte dall'utente ridefinendone, per esempio, la destinazione. Quel che è meglio, però, è che si integra completamente con sendmail.
- metamail, uno strumento per elaborare gli allegati (attachment) contenuti nei messaggi di posta elettronica.
- formail, un programma per riformattare i messaggi di posta.

Nel seguito di questa sezione verrà descritto un esempio d'uso di procmail, senza scendere però nei dettagli della configurazione e delle caratteristiche del programma, poiché si tratta di uno strumento che ha una grandissima varietà di usi. Ci si limiterà, quindi, a descrivere la configurazione di sendmail e procmail per un caso molto comune: lo smistamento della posta proveniente da mailing list. Per questo tipo di configurazione si farà in modo che sendmail utilizzi procmail come mailer locale; procmail, dal suo canto, leggerà un apposito file di configurazione contenente le regole di smistamento della posta.

Per prima cosa bisogna installare procmail, ricorrendo al sistema dei package (`mail/procmail`).

Poi bisogna configurare sendmail perché usi procmail. Per far ciò è sufficiente togliere il commento dalle linee relative a procmail dal file di configurazione di sendmail già descritto:

```
define(‘PROCMAIL_MAILER_PATH’, /usr/pkg/bin/procmail)dnl
FEATURE(local_procmail)dnl
```

```
MAILER(procmail)dnl
```

La prima riga specifica il percorso del programma procmail e deve essere cambiata a seconda di dove è stato installato (il che si può determinare con il comando **which procmail**). La seconda riga indica a sendmail di usare procmail come *mailer* locale e la terza aggiunge procmail all'elenco dei mailer di sendmail (questa riga è opzionale).

Infine si crea il file di configurazione di procmail, `.procmailrc`, nelle propria home directory. Questo file contiene le regole per la consegna della posta in locale ed è utilizzato dal programma per decidere la destinazione dei messaggi.

A titolo di esempio si immagini di essere iscritti a una mailing list sulle rose, il cui indirizzo è `<roses@flowers.org>`. Supponiamo, inoltre, che tutti i messaggi che arrivano da questa mailing list contengano, nell'intestazione, la stringa

```
Delivered-To: roses@flowers.org
```

Il file di configurazione di procmail ha un aspetto di questo tipo:

```
PATH=/bin:/usr/bin:/usr/pkg/bin
MAILDIR=$HOME/Mail
LOGFILE=$MAILDIR/from

:0
* ^Delivered-To: roses@flowers.org
roses_list
```

Il file precedente contiene una sola regola di smistamento, che inizia con la riga ":0". La configurazione precedente ridirige tutta la posta proveniente dalla mailing list nella mailbox di nome "roses_list" che l'utente avrà opportunamente creato nella directory $MAILDIR. Si noti che $MAILDIR è la stessa directory indicata nella direttiva del file di configurazione di Mutt

```
set folder="~/Mail"
```

I messaggi provenienti dalla mailing list vengono identificati dalla riga:

```
* ^Delivered-To: roses@flowers.org
```

contenuta nell'intestazione. I rimanenti messaggi andranno nella mailbox di default.

La mailing list è, naturalmente, solo un esempio, e procmail può essere usato per smistare qualunque tipo di messaggio in base, per esempio, alla provenienza, al contenuto, ecc.

Per controllare che procmail venga usato come mailer locale da sendmail si può eseguire quest'ultimo in modalità test nel modo seguente:

```
$ /usr/sbin/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Ora si richiede l'elenco dei mailer noti a sendmail con il comando

```
> =M
```

Nell'output dovrebbe comparire una riga di questo tipo:

```
mailer 3 (local): P=/usr/pkg/bin/procmail S=EnvFromL/HdrFromL ...
```

Come sempre, per ulteriori dettagli e per farsi un'idea più precisa delle potenzialità di procmail, consultare la pagina di manuale di procmail(1), procmailrc(5) e procmailex(5), quest'ultima contenente esempi di configurazione.

12.7. Come verificare un indirizzo di posta

Prima di passare ai *newsgroup* ho deciso di inserire questo piccolo "trucco" con la posta perché può essere utile: si tratta di verificare interattivamente la veridicità di un indirizzo di posta.

> **Nota:** non tutti i server accettano il comando VRFY, che può essere disabilitato.

Supponiamo di voler verificare l'utente carlo@dominio.com: il metodo è il seguente:

```
$ 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 con tin

Con il termine *news* si indica l'insieme degli articoli dei newsgroup di USENET, un servizio disponibile su Internet. Ogni newsgroup raccoglie articoli relativi a uno specifico argomento. Il meccanismo di lettura degli articoli è diverso rispetto alle mailing list. Quando ci si iscrive a una mailing list, gli articoli ci vengono inviati tramite posta e per leggerli (e rispondere) si usa un programma di posta (come, per esempio, mutt). La consultazione delle news avviene invece in modo diretto, utilizzando un programma detto *newsreader* come, per esempio, tin, che consente di effettuare la sottoscrizione ai gruppi che ci interessano e di seguirne i *thread*.

> **thread:** un thread è una sequenza di messaggi (risposte e controrisposte) derivanti da un messaggio che potremmo definire "iniziale". In pratica, qualcuno invia un messaggio al gruppo, altri inviano una risposta al messaggio originale, altri ancora (o magari gli stessi) rispondono a quelli che hanno risposto e così via, fino a creare una struttura ramificata di mesaggi e risposte. Senza un newsreader è impossibile districarsi nel groviglio (leggi "sequenza logica") dei thread.

Una volta installato il programma l'unica cosa da fare è impostare il nome del server NNTP, scrivendolo nel file `/usr/pkg/etc/nntp/server`. Per esempio:

```
news.bignet.it
```

Fatto questo si può lanciare il programma con il comando **rtin**. Sullo schermo comparirà una serie di scritte di questo 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...
```

Bisogna avere pazienza quando ci si collega per la prima volta, perché tin scarica una lista infinita di newsgroup alla quale ci si può iscrivere: l'operazione dura parecchi minuti. Terminato lo scaricamento compare la schermata del programma, solitamente vuota. Per vedere l'elenco dei gruppi premere **y**. Per sottoscriversi a un gruppo, portarsi sul nome del gruppo e premere **s**.

Per velocizzare gli avvii successivi si può usare il comando **rtin -Q**. In questo modo si evita la ricerca di nuovi newsgroup, si caricano solo i gruppi attivi (`-n`) e non si caricano le descrizioni dei gruppi(`-d`): naturalmente non sarà possibile usare il comando **y** (yank) durante la sessione con tin. Quando si avvia tin in questo modo il programma non è in grado di sapere quali newsgroup sono moderati.

> **Nota:** se ci si collega a Internet tramite un provider con un computer che non appartiene quindi a un "vero" dominio, quando si invia un messaggio l'indirizzo in testa al messaggio sarà sbagliato (perché inesistente). Per risolvere il problema si può usare l'opzione "mail_address" nel file di configurazione di tin (`~/.tin/tinrc`) oppure si può impostare la variabile di ambiente REPLYTO.

Capitolo 13. Driver di console

Nelle versioni prima della 1.4 per gestire lo schermo e la tastiera si poteve scegliere tra pccons (specifico per i386) e pcvt. Nella versione 1.4 ha fatto la sua comparsa il nuovo driver wscons che, col tempo, è destinato a sostituire completamente i primi due.

13.1. wscons

wscons è il nuovo driver di console di NetBSD, che dovrebbe sostituire pccons e pcvt. Offre terminali virtuali, supporto per tastiere internazionali, gestione del mouse, ecc. Le capacità di wscons cambiano a seconda del port: quello per i386 è il più ricco di funzionalità.

wscons è attivo di default al termine dell'installazione e, pertanto, non è necessario fare nulla per poterlo utilizzare. Per abilitare le console virtuali vedere Capitolo 4. Nel resto della sezione verranno descritte le opzioni del file di configurazione del kernel che riguardano wscons.

Per abilitare wscons nel file di configurazione del kernel bisogna attivare l'apposita opzione e disabilitare quelle di pcvt e pccons. (Attenzione, solo uno dei tre driver deve essere abilitato).

```
#pc0    at isa? port 0x60 irq 1   # pccons generic PC console driver
#vt0    at isa? port 0x60 irq 1   # PCVT console driver
```

Per abilitare la tastiera italiana

```
options     PCKBD_LAYOUT="KB_IT"
```

La disposizione della tastiera italiana non è l'ideale, specie per chi programma. Per una modifica vedere Capitolo 4.

Il numero di console virtuali autoallocate è controllato da

```
options     WSDISPLAY_DEFAULTSCREENS=4
```

Le console aggiuntive, cioè quelle in più rispetto alle console autoallocate, vanno abilitate in `/etc/wscons.conf`, togliendo il segno di commento (#) alle righe del tipo "screen x". Nell'esempio seguente viene riservata un'ulteriore console rispetto a quelle autoallocate:

```
# 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
```

Lo script `rc.wscons` trasforma ogni riga non commentata di questa tabella in una chiamata al comando **wsconscfg**: le colonne diventano parametri del comando stesso. La colonna *idx* diventa il parametro `index`, la colonna *screen* diventa il parametro `-t type` (che definisce il tipo di schermo, cioè il formato, il numero di colori, ecc.) e la colonna *emul* diventa il parametro `-e emul` (che definisce l'emulazione). Per esempio, se "screen 3" fosse attivo diventerebbe una chiamata a:

```
wsconscfg -e vt100 3
```

> **Nota:** si noti che si può verificare un conflitto (innocuo) tra le console autoallocate dal kernel e quelle allocate al momento del boot attraverso `/etc/wscons.conf`. Se all'avviamento si tenta di allocare uno schermo già autoallocato dal kernel verrà visualizzato un messaggio del tipo:
>
> ```
> wsconscfg: WSDISPLAYIO_ADDSCREEN: Device busy
> ```
>
> La soluzione consiste nel commentare le righe inutili di `/etc/wscons.conf`.

Le console create vanno naturalmente anche attivate in `/etc/ttys`. Per esempio:

```
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
...
```

La linea

```
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
```

di `/etc/ttys` consente al server X di trovare un terminale libero. Per usare uno schermo diverso dal numero 4 bisogna fare in modo di passare un parametro nella forma vt*n* al server X (*n* è il numero del tasto funzione associato allo schermo da riservare per la grafica).

Per esempio si può abilitare lo "screen 7" in `/etc/wscons.conf` e poi avviare X con “vt8”. Nel caso di xdm bisogna editare `/usr/X11R6/lib/X11/xdm/Xserver`, per esempio:

```
:0 local /usr/X11R6/bin/X +kb dpms -bpp 16 dpms vt8
```

Se si usa xdm3d, invece, il percorso è diverso: `/usr/X11R6/share/xdm3d/Xservers`.

> **Nota:** in wscons la combinazione Ctrl-Alt-Backspace resetta l'emulatore di terminale. Può essere utile in caso di problemi.

13.1.1. Modo testo a 50 linee con wscons

La modalità di testo a 50 linee si può utilizzare a partire dalla versione 1.4.1 di NetBSD. Per attivare questa modalità, oltre alle operazioni precedentemente descritte bisogna editare `/etc/wscons.conf` togliendo il commento alla linea:

```
font ibm  -  8  ibm  /usr/share/pcvt/fonts/vt220l.808
```

Alla fine del file modificare le righe nel modo seguente:

```
#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
```

In questo modo si ottengono otto schermi, ai quali si accede con la combinazione Ctrl-Alt-Fn (con n che va da 1 a 8); i dispositivi corrispondenti sono quelli da ttyE0 a ttyE7. Per abilitarli e avere un prompt di login, modificare `/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
```

A quanto sembra, non è possibile modificare l'impostazione di 80x25 dello schermo 0, probabilmente per evitare, in caso di problemi, di perdere l'unico schermo funzionante di console.

13.1.2. wsmouse

13.2. pccons

Pccons è il driver di console utilizzato dai dischetti di installazione del port i386 (e forse anche di altri port). Si tratta di un driver molto compatto (richiede poco spazio) ma che non dispone di console virtuali o di altre caratteristiche avanzate.

13.3. pcvt

pcvt è un emulatore di terminale VT220 ed è dotato di funzionalità molto più avanzate rispetto al semplice pccons. Tra le altre cose supporta più configurazioni di tastiera e console virtuali multiple (con Ctrl+Alt+F1...F8 oppure con i tasti funzione F9-F12). Per attivare pcvt è necessario ricompilare il kernel; occorre avere le seguenti linee nel proprio file di configurazione:

```
# Attivare solo una delle due linee seguenti
#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
```

Inoltre bisogna attivare la tastiera italiana all'avvio e scegliere il tipo giusto di terminale:

```
/usr/local/bin/kcon -m i2
TERM=pcvt25;    export TERM
```

Corrispondentemente bisogna modificare `/etc/ttys`. Per esempio:

```
#console "/usr/libexec/getty Pc"   pcvt25  on secure
ttyv0    "/usr/libexec/getty Pc"   pcvt25  on secure
```

> **Nota:** Nella definizione della tastiera i2 c'è un errore. Occorre modificare il file `/sys/arch/i386/isa/pcvt/Util/keycap/keycap.src`; la versione corretta è:
>
> ```
> i2|italy142|Italian 142 mapping:\
>      :A8={:A9=[:A10=]:A11=}:\
>      :A12=`:A13=~:\
>      :A17=@:A18=#:\
>      :tc=italy141:
> ```

All'avvio occorre caricare le impostazioni per la tastiera italiana, per esempio modificando `/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 '.'
```

Inoltre occorre modificare `/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
```

Occorre anche compilare e installare i programmi di utilità di pcvt.

```
cd /sys/arch/i386/isa/pcvt/Util
make
make install
```

> **Nota:** In caso di errori di compilazione, controllare che in `/usr/include/machine` sia presente il file `pcvt_ioctl.h`. Se non c'è bisogna copiarlo.

13.3.1. Cambiare le dimensioni dello schermo

Con pcvt è possibile cambiare il numero di righe e di colonne sullo schermo. Il seguente programma, per esempio, esegue l'operazione automaticamente, consentendo di scegliere tra varie configurazioni:

```
#!/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
```

Capitolo 14. Editing

Questo capitolo descrive l'editor standard di NetBSD, e cioè vi. La prima parte del capitolo introduce le basi dell'uso di questo editor, mentre le sezioni successive ne descrivono alcune caratteristiche avanzate, con particolare riferimento alla versione di vi installata con NetBSD. Chi desidera approfondire l'argomento può fare riferimento alla bibliografia alla fine di questo capitolo e ai link presentati nelle sezioni seguenti.

14.1. Introduzione a vi

Non c'è bisogno di presentazioni per l'editor vi: da quando è stato sviluppato da Bill Joy, all'Università della California di Berkeley è diventato l'editor standard di quasi tutte le versioni di Unix, il compagno e l'amico fedele degli amministratori di sistema e dei programmatori nonché la bestia nera di quasi tutti i principianti.

Vale dunque la pena di spendere qualche parola sull'uso di questo editor, anche perché, al termine dell'installazione di NetBSD, è l'unico disponibile sul sistema (fatta eccezione per ed) ed è quindi lo strumento che dovrà essere utilizzato per la configurazione iniziale del sistema stesso. In un secondo tempo, naturalmente, ciascuno è libero di installare l'editor che preferisce e conosce meglio, scegliendo tra la nutrita serie di quelli contenuti nella collezione dei package (vedere il Capitolo 8).

Questa sezione contiene un'introduzione all'uso di vi che dovrebbe essere sufficiente a consentire a un principiante di creare e modificare file di testo. Solo le caratteristiche di base e alcuni dei comandi di uso più comune vengono descritti, senza alcuna pretesa di completezza: per una trattazione più esauriente si rimanda ai documenti descritti ne la Sezione 14.2.2.

I principianti non devono commettere l'errore di giudicare vi a partire da queste poche istruzioni: usato in modo così "minimalista", vi è indubbiamente molto scomodo. Le sue virtù si scoprono quando si imparano tutti (!) i suoi comandi e si incominciano a utilizzare le sue funzioni avanzate; a questo punto vi non è più l'editor ostico degli inizi ma diventa uno strumento comodo, potente e affidabile. Questa guida è stata scritta, inutile dirlo, utilizzando l'editor vi.

14.1.1. Primi passi

Per lanciare vi, dare il comando:

```
$ vi nomefile
```

dove *`nomefile`* è il nome del file da editare (che può anche non esistere ancora). Se *`nomefile`* viene omesso, vi parte con un file inizialmente vuoto (e ancora senza nome: sarà necessario dargliene uno al momento di salvarlo). In questo esempio supponiamo di creare un file nuovo:

```
$ vi miofile
```

dove *`miofile`* è il nome di un file che non esiste nella directory corrente.

Una volta avviato vi, lo schermo mostra una colonna di caratteri tilde sulla sinistra e, sull'ultima riga, il nome del file, insieme a alcune informazioni di significato evidente; una cosa di questo tipo (ma con più righe):

```
~
~
~
~
~
~
miofile: new file: line 1
```

I caratteri tilde a inizio linea indicano che la linea in questione è vuota, cioè che la linea non contiene testo. Poiché il file caricato (miofile) non esisteva, tutte le righe sullo schermo di vi sono vuote.

A questo punto la pressione della maggior parte dei tasti provoca un segnale acustico o la visualizzazione di un messaggio d'errore (si provi, per esempio, a premere **j** oppure **k**). Ciò è dovuto al fatto che vi è un editor *modale* e pertanto i tasti premuti hanno un effetto diverso al seconda del "modo" in cui si trova l'editor. All'avvio vi si trova sempre in *modo comandi* e tutti i tasti premuti vengono interpretati come comandi per l'editor, anziché come testo da inserire nel file; pertanto, premendo **j**, vi non inserisce una "j" nel file ma cerca di eseguire il comando **j** che, come si vedrà tra breve, significa "sposta il cursore alla linea seguente". Poiché il file è ancora vuoto, non c'è alcuna linea seguente su cui portare il cursore e viene segnalato un errore.

Alcuni dei comandi di vi consentono di passare dal modo comandi al *modo inserimento*; una volta entrati in questo modo tutto ciò che si scrive viene interpretato come testo da inserire nel file. Vediamo ora un esempio.

Premere **i** per passare in modo inserimento e digitare le seguenti linee di testo (terminare ogni riga con **Invio**):

```
Prima riga di testo inserito,
seconda riga di testo inserito,
terza e ultima riga di testo inserito.
```

al termine dell'inserimento premere **ESC** per tornare in modo comandi.

In questo esempio abbiamo visto come passare dal modo comandi al modo inserimento premendo **i**, come inserire alcune righe di testo, terminando ciascuna riga con il tasto **Invio**, e come tornare infine al modo comandi premendo **ESC**. Adesso che il file contiene un po' di testo si possono sperimentare alcuni comandi; i primi che vedremo sono quelli che consentono di spostarsi all'interno del file.

> **Nota:** il testo inserito appartiene logicamente al file `miofile` (il nome passato come parametro all'editor) ma, per il momento, risiede solamente nella memoria dell'editor perché non è ancora stato salvato su file. Tutte le operazioni effettuate sul testo (aggiunte, modifiche, ecc.) non vengono salvate su disco finché l'utente non lo richiede esplicitamente con un comando di salvataggio del file. Pertanto è sempre possibile abbandonare le modifiche uscendo dall'editor senza salvare e la copia del file su disco rimarrà invariata. I comandi per salvare/abbandonare/uscire sono descritti ne la Sezione 14.1.9.

14.1.2. Comandi per spostarsi nel file

Quando ci si trova in modo comandi, alcuni comandi consentono di spostarsi all'interno di un file (cioè di modificare la posizione del cursore). L'elenco seguente riassume i principali comandi di spostamento.

```
h       cursore a sinistra      (freccia a sinistra)
j       cursore in basso        (freccia verso il basso)
k       cursore in alto         (freccia verso l'alto)
l       cursore a destra        (freccia a destra)
^F      pagina seguente         (PgDn)
^B      pagina precedente       (PgUp)
0       cursore a inizio riga
$       cursore a fine riga
b       parola precedente
w       parola seguente
Invio   primo carattere della riga seguente   (spazi esclusi)
+       primo carattere della riga seguente   (spazi esclusi)
-       primo carattere della riga precedente (spazi esclusi)
```

(il simbolo ^ indica la pressione del tasto Control insieme al tasto che rappresenta la lettera che segue; pertanto **^F** significa Control-F). A seconda della configurazione del terminale è anche possibile che l'editor riesca a utilizzare i tasti freccia e i tasti PgUp e PgDn.

Si noti che, come già detto, questi tasti servono per spostarsi quando ci si trova in modo comandi: se premuti in modo inserimento, invece, causano l'inserimento del testo relativo all'interno del file (cioè premendo **h** il carattere 'h' viene inserito nel file). Osserviamo quindi un primo fatto: in modo inserimento si può solo inserire testo sequenzialmente; per potersi spostare all'interno del file occorre uscire dal modo inserimento (con **ESC**) tornando al modo comandi. Questa è una delle conseguenze del fatto che vi è un editor modale.

In modo comandi è possibile spostarsi nel file anche specificando il numero di riga sulla quale ci si vuole portare. Per esempio:

```
12G     vai alla riga 12
```

Il comando **G** senza prefisso numerico porta il cursore a fine file.

> **Prefissi numerici:** molti comandi di vi accettano un *prefisso numerico* che ne modifica l'interpretazione da parte dell'editor. Anche se l'uso del prefisso numerico non viene qui descritto, riportiamo alcuni esempi tanto per dare un'idea dei possibili utilizzi.

```
9k      cursore 9 righe verso l'alto
12dd    cancella dodici righe
3w      cursore avanti di tre parole
5x      cancella 5 caratteri
```

In questa sezione sono stati descritti alcuni dei principali comandi di spostamento, che possono essere provati sulle linee di testo appena inserite. L'editor vi dispone di molte altre funzioni sofisticate per spostarsi all'interno del file, che però non vengono descritte in questo breve tutorial.

14.1.3. Comandi per cambiare modo

Nell'esempio iniziale si è già visto che per passare da modo inserimento a modo comandi bisogna premere il tasto **ESC**. Se si è preme ESC mentre vi è già in modo comandi, l'editor emette una innocua segnalazione acustica e, pertanto, se non ci si ricorda qual'è il modo corrente si può sempre premere ESC, con la garanzia di ritrovarsi in modo comandi.

Se ESC è l'unico metodo per tornare al modo comandi, i comandi che permettono di entrare in modo inserimento sono molti. A seconda del comando scelto l'effetto risulta diverso, come si può osservare dalla seguente tabella.

`i`	`inserisci testo prima del cursore`
`a`	`inserisci testo dopo il cursore`
`A`	`inserisci testo a fine riga`
`I`	`inserisci testo a inizio riga`
`o`	`apri una riga vuota dopo la riga corrente`
`O`	`apri una riga vuota prima della riga corrente`

La presenza di numerosi comandi che fanno cose simili è una costante dell'editor vi; l'effetto è quello di avere molti modi per fare la stessa cosa, e quindi molte possibili soluzioni. L'esperto di vi riesce solitamente a trovare la soluzione che gli consente di portare a termine la funzione desiderata con il minor numero possibile di comandi. Per esempio, per inserire la parola "fine" a fine riga si possono utilizzare queste due sequenze equivalenti:

```
$afineESC
AfineESC
```

nella prima ci si porta a fine riga con il comando **$**, poi si passa in modo inserimento con **a**, si aggiunge la stringa "fine" e si ritorna in modo comandi premendo **ESC**. Nella seconda sequenza, il comando **A** combina le funzioni di **$** e di **a** in un tasto solo.

14.1.4. Comandi per cancellare testo

La tabella seguente riassume alcuni comandi per la cancellazione di testo.

`dd`	`cancella la riga corrente`
`x`	`cancella il carattere sotto al cursore`
`X`	`cancella il carattere prima del cursore`
`dw`	`cancella fino a fine parola`
`D`	`cancella dal cursore a fine riga`

anche questi comandi si possono dare in modo comandi, e non in modo inserimento. Da qui in avanti non verrà più indicato il fatto che i comandi operano in modo comandi poiché dovrebbe essere oramai evidente. Quando verranno descritte alcune funzioni attivabili dal modo inserimento, questo fatto verrà indicato esplicitamente.

14.1.5. Esempio

Vediamo ora un esempio pratico che riassume l'uso di alcuni dei comandi presentati nelle sezioni precedenti. Supponiamo che il file in corso di editazione sia composto dalle tre righe inserite nell'esempio iniziale e che si voglia trasformare la terza riga in questo modo:

```
terza riga di testo inserito nel file.
```

Come prima operazione cancelliamo dall'ultima riga le parole "e ultima " e poi inseriamo il nuovo testo " inserito nel file" alla fine della riga, prima del punto.

Supponiamo che il cursore si trovi sulla prima riga (se così non fosse, portarsi sulla prima riga con il comando **1G**). Per andare dalla prima alla terza riga si può usare una delle sequenze seguenti (ma ve ne sono anche altre):

```
<Invio><Invio>
jj
3G
/terza/<Invio>
```

Una volta arrivati all'inizio della terza riga ci si porta sulla prima delle due parole da cancellare con

```
w
```

Infine si cancellano le due parole con:

```
dwdw
```

> **Nota:** ricordando quanto accennato sui prefissi numerici, si può intuire che i due comandi **dw** appena dati possono essere sostituiti dall'unico comando **2dw**, che cancella due parole in un colpo solo, o anche, il che può apparire sorprendente, da **d2w**.

Per aggiungere il nuovo testo ci si porta a fine riga con il comando **$**, si entra in modo inserimento con **i** e poi si scrive:

```
inserito nel file
```

Premendo **ESC** si torna in modo comandi.

14.1.6. Ricerca di testo

Per cercare una stringa all'interno del file corrente si usano i seguenti comandi:

```
/stringa/
?stringa?
```

il primo comando (seguito da Invio) cerca la stringa nel file andando avanti, cioè verso fine file, a partire dalla posizione del cursore. Il secondo comando (anch'esso seguito da Invio) cerca la stringa andando all'indietro, cioè verso l'inizio del file.

Per esempio, ritornando alle tre linee inserite inizialmente si provi a dare i seguenti comandi:

```
1G/ultima/
```

e a premere invio. Il comando **1G** fa spostare il cursore sulla prima riga del file (si noti che non viene fatta l'eco del comando). La sequenza **/ultima/** (che viene visualizzata sull'ultima riga dello schermo) seguita da **Invio** causa la ricerca della stringa "ultima" all'interno del testo e lo spostamento del cursore sulla riga relativa.

> **Nota:** i comandi di ricerca costituiscono quindi un ulteriore metodo di spostamento all'interno del file, mirato al contenuto del file stesso.

Per ripetere l'ultima ricerca effettuata senza doverla riscrivere si possono usare i comandi

```
n       ripeti la ricerca nella stessa direzione
N       ripeti la ricerca nella direzione opposta
```

14.1.7. Comandi che funzionano in modo inserimento

Anche quando si sta inserendo del testo sono disponibili alcuni comandi, generalmente ottenuti con il tasto Control:

```
^H      cancella il carattere a sinistra del cursore
BS      cancella il carattere a sinistra del cursore
^W      cancella la parola precedente
^U      cancella fino a inizio riga
ESC     torna in modo comandi
```

14.1.8. Annullare l'effetto di un'azione

Se, una volta tornati in modo comandi, ci si accorge di aver commesso un errore è possibile utilizzare il comando di *undo* (il tasto **u**) per annullare l'ultimo comando eseguito.

14.1.9. Comandi per uscire da vi

I principali comandi per terminare il lavoro e uscire da vi oppure semplicemente per salvare e continuare a lavorare sono:

```
:w      salva il file (senza uscire)
:q      esci
```

```
:wq     salva il file e esci
:q!     esci senza salvare (le modifiche vanno perse)
:w abc  salva il file dandogli il nome ’abc’
```

Per cambiare il nome al file corrente (o per dargliene uno se non ce l’ha):

```
:file nomefile
```

I comandi descritti in questa sezione sono esempi di un’ulteriore modalità di funzionamento di vi, il *modo ex*, che in questa breve introduzione non viene descritto. L’editor vi nasce infatti dalla combinazione di due distinti editor dei quali uno (vi, per l’appunto) serve da interfaccia *full screen* per l’altro (ex), che fornisce le funzioni base per l’editing. Ex si può ritenere una versione evoluta dell’editor ed.

14.1.10. Conclusione

Come già accennato nell’introduzione, in questa sezione sono stati descritti solo i comandi più elementari dell’editor vi, per consentire l’editing dei file di configurazione di NetBSD al termine dell’installazione. Per poter usare questo editor in modo produttivo (e comodo) è necessario approfondire la conoscenza dei comandi leggendo, per esempio, i documenti citati ne la Sezione 14.2.2.

14.2. Configurare vi

L’editor standard in dotazione a NetBSD è, come già detto, vi. Se non utilizzate vi questa sezione non vi interessa, in caso contrario leggetela prima di correre a installare vim o elvis. Il vi di NetBSD è, infatti, nvi, una versione storica di vi con molte potenti estensioni scritta da Keith Bostic dell’Università della California di Berkeley verso i primi anni 90 per avere una versione liberamente ridistribuibile di vi e diventata in seguito la versione ufficiale di BSD.

Tra le estensioni più interessanti troviamo

- Espressioni regolari estese (abilitate con l’opzione `extended`).
- Tag stacks.
- Undo infinito (attivato in modo un po’ curioso: per il primo *undo* si preme **u**; gli undo successivi si ottengono premendo il carattere punto (**.**)).
- Ricerca incrementale (abilitata con l’opzione `searchincr`).
- Scrolling delle linee (abilitato dall’opzione `leftright`; il numero delle colonne è definito da `sidescroll`).
- History della linea di comando (controllata dall’opzione `cedit`).
- Completamento della linea di comando (controllata dall’opzione `filec`).
- Schermi che si possono mettere in foreground e in background.
- Split screen.

14.2.1. Estensioni a `.exrc`

Nell'esempio seguente vediamo qualche comando di configurazione che ci semplifica un po' la vita.

```
set showmode ruler
set filec=^[
set cedit=^[
```

La prima riga abilita la visualizzazione della posizione del cursore (riga e colonna) e della modalità corrente (Command, Insert, Append) sulla riga di stato. La seconda riga (^[è il carattere ESC) abilita il completamento dei nomi dei file e la terza la *history* della linea di comando: entrambe utilizzeranno il carattere ESC. Per esempio, scrivendo ':' e poi premendo ESC si apre una finestra con l'elenco dei precedenti comandi; l'elenco si può modificare come se fosse un file. Quando si preme invio su una riga il corrispondente comando viene eseguito.

14.2.2. Documentazione

Nel *tarball* dei sorgenti (`src.tgz`) è contenuta una notevole quantità di documentazione originale di vi/nvi: si trova in `/usr/src/usr.bin/vi/docs`. In particolare si possono trovare:

- Edit: A tutorial
- Ex Reference Manual
- Pagina di manuale di vi
- An Introduction to Display Editing with Vi di William Joy e Mark Horton
- Ex/Vi Reference Manual di Keith Bostic, il manuale di riferimento di nvi, che descrive in dettaglio tutti i comandi dell'editor.
- Vi Command & Function Reference
- Vi tutorial (beginnner e advanced)

Chi desidera imparare a usare vi deve iniziare a leggere *An Introduction to Display Editing with Vi* di William Joy e Mark Horton, per poi passare a *Ex/Vi Reference Manual* di Keith Bostic.

14.3. Uso dei tag con vi

Questo argomento non è strettamente legato a NetBSD ma, già che ci siamo, merita un piccolo approfondimento perché può essere comodo per esaminare, per esempio, i sorgenti del kernel o di un programma complesso, composto da più subdirectory.

I *tag* sono una di quelle caratteristiche che rendono vi un potente strumento di editing per programmatori: si tratta in sostanza di un file contenente un elenco di nomi di procedure, macro, ecc. presenti nei sorgenti di un programma. Quando si editano i sorgenti, grazie ai tag è possibile saltare dal punto in cui viene chiamata una funzione al punto in cui è definita la funzione stessa. Per esempio, supponiamo di esaminare un sorgente con le seguenti righe:

```
for (i = 0; i < len; i++)
        funcXY(i);
```

portandosi con il cursore sulla prima lettera di “funcXY” e premendo ^] (Control- [), vi automaticamente apre il sorgente in cui è presente la definizione di funcXY e si posiziona sulla funzione stessa.

```
void
 funcXY(i)
         int i;
 {
  funcZ();
         ...
```

Una volta esaminata/modificata funcXY, premendo ^T (Control-T) si ritorna al file e alla posizione iniziale (quella del ciclo for). Si noti che questi “salti” possono essere annidati; se con ^] si salta alla funzione funcXY, e poi con un successivo ^] si salta alla funzione funcZ, con due ^T si ritorna al file e alla posizione iniziale.

Per ottenere il risultato descritto nel paragrafo precedente è necessario che vi trovi un file (solitamente) chiamato `tags` nella directory corrente o in una posizione da noi specificata con il comando **:set tags=/percorso/nomefile** (questo comando si può anche inserire nel file `.exrc`). Tale file, che poi è un file di testo, viene generato dal programma **ctags**. Per esempio il comando:

```
$ ctags *.c *.h
```

genera il file `tags` nella directory corrente. Quando parte vi, il file `./tags` viene trovato automaticamente.

Quando si esaminano dei sorgenti che si trovano in una gerarchia di directory e sottodirectory si possono ugualmente generare e utilizzare i *tag* di vi. La procedura è la seguente.

1. Ci si porta nella directory base dei sorgenti, con il comando

   ```
   $ cd /percorso
   ```

2. Si costruisce il file dei tag in due passaggi; prima si crea una lista dei file sorgenti da eseminare e poi la si sottomette al programma ctags per creare il file `tags`.

   ```
   $ find . -name "*.[ch]" > filelist
   $ ctags -L filelist
   ```

3. In `./.exrc` si inserisce la linea

   ```
   set tags=/percorso/tags
   ```

   sostituendo il percorso appropriato al posto di “percorso”.

14.4. Alternative a nvi

Nvi è la versione di vi nativa di NetBSD, ma non è certamente l’unica esistente. Senz’altro è una delle più vicine allo spirito della versione originale del programma, e quindi, se siete dei puristi, non vorrete usare altro, come del resto fanno molti degli stessi sviluppatori di NetBSD. Tuttavia esistono molti “cloni” di questo programma, alcuni dei quali molto diffusi e apprezzati, e pertanto chi si trova bene a lavorare con vi ma cerca “qualcosa in più” ha un’ampia rosa di scelte. Per una panoramica completa i siti “The VI pages” (http://www.math.fu-berlin.de/~guckes/vi) e “Vi Lovers Home Page” (http://www.thomer.com/thomer/vi/vi.html) sono il miglior punto di partenza. Oltre alla descrizione e ai link per tutti i cloni esistenti di vi, vi sono collegamenti a tutorial e informazioni varie.

Per iniziare a districarsi nella giungla dei programmi consiglio di dare un’occhiata a vim e vile. Si tratta di due versioni “potenziate” di vi, a cui sono stati aggiunti molti comandi, supporto per il mouse (nelle versioni X), evidenziazione della sintassi dei file sorgenti con colori diversi, possibilità di spostarsi all’interno di un file anche quando si è in modalità inserimento e tante, tante altre funzionalità. La filosofia dei due programmi, ottimi entrambi e meritevoli di essere provati, è molto diversa. Vile, in particolare, è programmabile, essendo dotato di un vero macrolinguaggio che consente di definire procedure e assegnarle ai tasti e quindi piacerà agli amanti della personalizzazione estrema. Vim, dal canto suo, è forse il clone più diffuso e amato, dotato di un ricchissimo (anche troppo) assortimento di nuovi comandi, molti dei quali studiati per i programmatori.

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*.

Capitolo 15. X

15.1. Che cos'è X?

Il sistema X Window è un ambiente di lavoro grafico disponibile per NetBSD e per molti altri sistemi Unix (e non Unix). In effetti X è più di un semplice ambiente grafico: grazie all'uso del protocollo X, il sistema X Window è "network transparent" (trasparente alla rete) ed è in grado di eseguire applicazioni distribuite (client-server). Ciò significa, in prima approssimazione, che è possibile lanciare un programma su un host "client" e visualizzarne l'output grafico sul un altro host "server" in modo del tutto trasparente; trasparente significa, in questo caso, che non è necessario modificare l'applicazione per ottenere questo risultato. Il sistema X Window e prodotto dall' "X Consortium" e la release corrente è X11R6. La versione di X usata da NetBSD è XFree86, un'implementazione open source liberamente ridistribuibile del sistema X Window.

quando si incomincia a usare X si incontrano parecchi termini il cui significato può apparire inizialmente oscuro. Gli elementi base di X sono:

- Hardware video supportato da XFree86.
- Un *server X* che si interfaccia con l'hardware. Il server X fornisce un modo standard per aprire finestre, effettuare operazioni grafiche (per esempio, utilizzare font per visualizzare del testo) e per leggere l'input del mouse, della tastiera o di altre periferiche. X è trasparente alla rete, e pertanto è possibile lanciare un client X su una macchina e il relativo server X (cioè il display con l'hardware video) su un'altra.
- Un *window manager* (gestore di finestre) che sfrutta il server X. Il window manager è, in sostanza, un tipo speciale di client X cui è consentito manipolare il posizionamento e l'aspetto delle finestre. Può "decorare" le finestre con "widget" standard, che solitamente consentono di spostare, ridimensionare, ridurre a icona, ecc. le finestre). Un window manager può essere dotato anche di altre funzionalità (chiudere le finestre, terminare i programmi, mostrare menu di programmi, ecc.) o di effetti speciali (ombreggiature, decorazioni, ecc.) e così via.
- Un *desktop manager* (opzionale). KDE e GNOME sono due esempi di desktop manager: si tratta di insiemi di programmi (suites) più o meno integrati e coordinati, progettati per fornire all'utente un insieme di applicazioni di base dotate di un'interfaccia comune. Il desktop manager può consentire l'accesso al file system utilizzando la metafora del desktop (piano di lavoro), può disporre di un sistema integrato di navigazione dell'help, di terminali che sostituiscono il classico xterm, di applicazioni per la gestione dell'audio, di ambienti di sviluppo, editor, e così via.
- Tutte le altre applicazioni (client X di terze parti) installate sul sistema. Queste "parlano" con il server X e con il window manager (generalmente il desktop manager non è coinvolto in modo particolare in ciò che viene eseguito dalle applicazioni.

Ricapitolando, per poter utilizzare l'ambiente grafico occorre:

- il sisyem XFree86
- un window manager (XFree86 dispone già di un semplice ma relativamente completo window manager di nome twm. Non è molto sofisticato ma molti lo trovano più che sufficiente) per le loro esigenze.

- Chi preferisce un ambiente più strutturato vorrà installare anche un desktop manager, benché ciò non sia assolutamente necessario. I desktop offrono caratteristiche che possono facilitare l'avvicinamento al sistema di utenti provenienti da altre ambienti, quali MacIntosh o una delle tante versioni di MS-Windows).

> **Nota:** a questo punto dovrebbe essere chiaro che i desktop come GNOME o KDE non forniscono un server X, bensì sfruttano un server X fornito da XFree86. GNOME e KDE hanno il proprio window manager ma possono anche sfruttarne uno esterno.

Normalmente si utilizza un solo window manager alla volta per ogni server X (ma è possibile attivare più di un server X su uno stesso computer).

15.2. Configurazione

A meno di non aver scelto una configurazione minima, al termine dell'installazione X, nella sua incarnazione XFree86, è già presente sul sistema e pronto per funzionare, a patto di creare il file di configurazione, il temuto `/etc/XF86Config`. Per farsi un'idea dell'aspetto di questo file si può dare un'occhiata a `/usr/X11R6/lib/X11/XF86Config.eg`. La struttura del file di configurazione è descritta formalmente in XF86Config(4/5) e si può esaminare con:

```
# man XF86Config
```

Prima di configurare il sistema è consigliabile un'attenta lettura della documentazione di XFree86 in `/usr/X11R6/lib/X11/doc`: vi sono dei file README specifici per le schede video e per il mouse e anche un `README.NetBSD`. Consiglio di cominciare con la lettura di `QuickStart.doc`. Il tempo dedicato alla lettura di queste informazioni non è affatto sprecato: la conoscenza della propria configurazione e di come utilizzarla con X è di grande importanza per sfruttare al meglio il proprio hardware.

Il file `/etc/XF86Config` può essere scritto manualmente con un qualsiasi editor oppure sfruttando un programma che lo crei in automatico, partendo dalla configurazione specificata interattivamente dall'utente. I programmi più noti sono xf86config e XF86Setup. Il primo funziona in modo testo e viene installato di default insieme a X, il secondo è un programma grafico che si trova nella collezione dei package.

Entrambi gli applicativi richiedono all'utente di specificare la configurazione del proprio sistema e, in particolare:

- il tipo di mouse e il dispositivo da utilizzare
- il tipo di tastiera e il suo layout
- il tipo di scheda video
- il tipo di monitor

Prima di scrivere il file di configurazione, a mano o utilizzando uno dei due programmi già menzionati, bisogna procurarsi queste informazioni.

15.3. Il mouse

Per quanto riguarda il mouse bisogna verificare se il mouse è di tipo seriale o PS/2 e se si sta usando il dispositivo *wsmouse*, che richiede un protocollo diverso. Per il mouse seriale bisogna scegliere il protocollo corretto e specificare la porta seriale. Per esempio, per un mouse sulla prima porta seriale:

```
Section "Pointer"
  Protocol    "Microsoft"
  Device      "/dev/tty00"
EndSection
```

Per un mouse sul dispositivo wsmouse:

```
Section "Pointer"
  Protocol    "wsmouse"
  Device      "/dev/wsmouse0"
EndSection
```

Nel campo "Device" si può anche specificare `/dev/mouse` a patto di creare l'appropriato link nel filesystem. Per esempio:

```
# ln -sf /dev/wsmouse0 /dev/mouse
```

15.4. La tastiera

Anche se avete già configurato la tastiera per wscons, per poter usare X è necessario effettuare una configurazione delle tastiera specifica per quest'ambiente, cosa, comunque, tutt'altro che complessa.

In mancanza di specifiche controindicazioni conviene utilizzare il protocollo XKB, selezionando il tipo di tastiera e il suo layout.

La configurazione della tastiera è uno dei punti deboli di tutti i programmi di configurazione automatica. Solitamente conviene lasciare impostata la configurazione originale inglese e poi ritoccare manualmente il file di configurazione generato dal programma. Per esempio:

```
# XkbDisable
# XkbKeymap   "xfree86(us)"

XkbModel        "pc102"
XkbLayout       "it"
XkbVariant      "nodeadkeys"
```

Sostituendo pc105 a pc102 in XkbModel si possono utilizzare anche i simboli delle tastiere Windows.

15.5. Il monitor

È fondamentale specificare correttamente i valori della frequenza orizzontale e verticale del monitor: una definizione corretta protegge il monitor da impostazioni incompatibili della scheda video, evitando che venga danneggiato. Questi dati sono presenti sui manuali di tutti i monitor. Nella directory con la documentazione è presente un file con le specifiche di un gran numero di monitor.

15.6. La scheda video e il server

La scheda video può essere scelta dal database dei programmi di configurazione. Una volta scelta la scheda, il programma si prenderà cura delle restanti impostazioni.

Una volta scelta la scheda bisogna definire il server da utilizzare. I programmi di configurazione di solito sono in grado di scegliere il server giusto in base alla scheda grafica selezionata. Alcune schede, però, sono supportate da più di un server e in questo caso occorre decidere in base alla documentazione; server diversi hanno generalmente caratteristiche e livelli di supporto diversi. Un esempio tipico è la scheda S3 Virge che è supportata sia dal server SVGA sia dal server S3V.

15.7. Avviare X

All'uscita dal programma di configurazione viene creato il file `/etc/XF86Config`, che può essere esaminato e modificato a mano.

Prima di lanciare X conviene:

- controllare che il link simbolico `/usr/X11R6/bin/X` punti al server giusto:

  ```
  # ls -l /usr/X11R6/bin/X
  ```

- Per controllare che tutto vada bene eseguire:

  ```
  # X -probeonly
  ```

  controllandone attentamente l'output.

A questo punto si può lanciare X con il comando:

```
# startx
```

Se X non parte bisognerà capire il motivo osservando i messaggi visualizzati sullo schermo e correggere il file di configurazione.

Se X si avvia ma qualcosa non funziona (per esempio il mouse) si può uscire rapidamente da X con la combinazione di tasti Ctrl-Alt-Backspace (non disponibile in tutti i port). Se invece tutto va bene ci si ritrova nell'ambiente X con il window manager di default che è twm; si tratta di un window manager semplice ma efficace, tanto che molti non sentono il bisogno di cercarne altri e continuano a usarlo. Naturalmente è possibile installare un altro window manager come pure personalizzare l'aspetto di twm. Per esempio, provare a dare il seguente comando in un xterm:

```
# xsetroot -solid DarkSeaGreen
```

15.8. Personalizzare X

Vi sono vari modi per personalizzare la configurazione (cioè l'aspetto e le finestre) di X all'avvio. Uno dei più semplici è creare un file `.xinitrc` nella propria home directory, copiando quello di default e modificandolo.

```
# cp /usr/X11R6/lib/X11/xinit/xinitrc ~/.xinitrc
# vi .xinitrc
```

Ecco un esempio di modifica che avvia il window manager, mostra l'orologio in basso a destra e apre due xterm. Inoltre imposta un colore per lo sfondo.

```
la prima parte è uguale
...
# 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
```

Con questo tipo di configurazione, per uscire da X bisogna chiudere l'ultimo xterm (quello con il nome "login").

Al di là dei gusti personali in fatto di disposizione delle finestre e di colori, mi sembra evidente che solo con queste piccole modifiche l'ambiente di lavoro è diventato più piacevole.

Per migliorare ancora di più l'aspetto del desktop si possono installare alcuni programmi di utilità dalla collezione dei package, tra cui:

xcolorsel

visualizza tutti i colori definiti in `rgb.txt`.

xpmroot

consente di utilizzare una pixmap come sfondo di X.

xscreensaver

screen saver per X.

xdaemon

nessun desktop può dirsi completo senza questo programma che visualizza la bitmap del demone simbolo di *BSD, in due grandezze selezionabili. L'immagine può essere spostata a piacimento.

15.9. Altri window manager

Chi non si accontenta di twm può installare un altro window manager tra i tanti disponibili nella collezione dei package. Tra i più popolari: fvwm2, olwm/olvwm (Open Look Window Manager), WindowMaker, Enlightenment, AfterStep.

Nel seguito vediamo, a titolo di esempio, l'installazione di WindowMaker. Per aggiungere il programma utilizziamo il package precompilato `windowmaker-0.60.tgz`, che dipende da una serie di package che devono essere installati. Come sempre sia **pkg_add** sia **make install** gestiranno automaticamente tutte le dipendenze, e pertanto in realtà non è necessario individuarle e installarle 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:** le dipendenze si potevano anche vedere con:
>
> ```
> # pkg_info -f windowmaker-0.61.0.tgz | grep depends
> ```

Dopo aver aggiunto i package richiesti, aggiungiamo anche WindowMaker e, già che ci siamo, un po' di temi:

```
# pkg_add windowmaker-0.61.0.tgz wmthemes-0.6x.tgz
```

WindowMaker è ora installato e per avviarlo basta modificare `.xinitrc` e/o `.xsession`, sostituendo la chiamata al precedente window manager (twm) con la chiamata a `wmaker`. Per esempio:

```
# start some nice programs
# start WindowMaker
wmaker &
xclock -geometry 50x50-1-1 &
xdaemon2 -geometry +0-70 &
...
```

In questo esempio viene lanciato automaticamente anche il programma xdaemon che "personalizza" il desktop.

Prima di avviare WindowMaker bisogna eseguire il programma di configurazione:

```
$ wmaker.inst
$ startx
```

15.10. Esempio di uso delle risorse di X

A titolo di esempio su come si personalizzano le risorse dei programmi X, vediamo come cambiare il font di default di un XTerm.

Per personalizzare le risorse utilizzate da un programma, nel nostro caso XTerm, si può agire a livello globale, modificando il file `/usr/X11R6/lib/app-defaults/Xterm`, ma è preferibile effettuare la modifica nel file ${HOME}/.Xresources, anché perchè, così facendo, non viene persa in caso di aggiornamento di X o del sistema operativo.

Una volta presa questa decisione bisogna scegliere un nuovo font usando i programmi xfontsel e xlsfonts. Conviene anche consultare il file che definisce gli alias dei font, che è `/usr/X11R6/lib/X11/fonts/misc/font.alias`. Per esempio, si scopre che il font di default “fixed” è in realtà un alias di “-misc-fixed-medium-r-semicondensed-13-120-75-c-60-iso8859-1”. Anche i font come, per esempio, “9x15” sono in realtà degli alias.

Una volta scelto il font, prima di installarlo in `.Xresources`, lo si può provare con un comando simile al seguente:

```
$ xterm -fn '-bitstream-courier-medium-r-normal-*-15-*-*-*-*-*-*-*'
```

Una volta trovato il font desiderato, che supponiamo sia “9x15”, lo si inserisce nel file `.Xdefaults` con una riga di questo tipo:

```
XTerm.VT100.font: 9x15
```

Per rendere immediatamente effettivo il cambiamento eseguire il comando:

```
$ xrdb -merge ~/.Xresources
```

Al successivo riavvio di NetBSD non è più necessario dare il comando precedente perché il file `.xinitrc` solitamente effettua automaticamente il merge di .Xresources.

> **Nota:** si noti che
>
> ```
> XTerm.VT100.font: 9x15
> ```
>
> è diverso da
>
> ```
> XTerm*font: 9x15
> ```
>
> Nel secondo caso il carattere “*” agisce da carattere jolly (wildcard) e quindi la direttiva ha effetto su tutti i tipi di font (per esempio anche su quelli dei menu). La scelta di una delle due forme dipende quindi dall’effetto che si vuole ottenere.

Esaminando la pagina di manuale di XTerm (come pure quella della maggior parte dei programmi X) si trovano le indicazioni sulle risorse che è possibile personalizzare.

Quasi tutti i programmi X (quelli scritti con i cosiddetti “X Toolkit Intrinsics”), accettano alcune opzioni comuni da linea di comando, quali:

- -display display
- -geometry geometry
- -bg color, -fg color
- -fn font
- -xrm resourcestring

Si veda la pagina di manuale di X per ulteriori dettagli.

15.11. Login grafico con xdm

Se si utilizza sempre X per lavorare, può essere piacevole avviarlo direttamente all'accensione del computer e avere un login grafico. Niente di più semplice:

1. Creare il file `.xsession` nella propria home directory. Questo file fa le veci di `~/.xinitrc` e può anche essere semplicemente un link a quest'ultimo.
2. Modificare `/etc/rc.conf` in questo modo:

   ```
   xdm=YES       xdm_flags=""       # x11 display manager
   ```

   In alternativa si può aggiungere la seguente riga in fondo a `/etc/rc.local`:

   ```
   /usr/X11R6/bin/xdm
   ```

   Questo metodo può essere utilizzato per avviare, per esempio, kdm oppure gdm al posto di xdm.

I file di configurazione di xdm si trovano in `/usr/X11R6/lib/X11/xdm`. In Xservers viene avviato X sul terminale virtuale vt05: se lo si vuole avviare su un altro terminale questo è il posto giusto per intervenire. Per un buon funzionamento di xdm, al fine di evitare che due processi entrino in conflitto per la tastiera, conviene avviare xdm su un terminale virtuale in cui getty non è attivo. In pratica, se in `Xservers` abbiamo:

```
:0 local /usr/X11R6/bin/X :0 vt04
```

in `/etc/ttys` è opportuno avere:

```
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
```

(si noti che vt04 corrisponde a ttyE3, perchè la numerazione dei vt incomincia da 1 mentre quella dei ttyE incomincia da 0).

Per rendere la schermata di login di xdm un po' meno "misera", basta intervenire sui file di configurazione di xdm. Per esempio, si può cambiare il colore dello sfondo aggiungendo al file `Xsetup_0` la riga seguente:

```
xsetroot -solid SeaGreen
```

Oppure, invece del colore, come sfondo si può mettere un'immagine, usando xpmroot. Per esempio:

```
xpmroot /path_to_xpm/netbsd.xpm
```

Giocando sui file di configurazione si possono ottenere molti effetti gradevoli.

Capitolo 16. Emulazione Linux

Il port per i386 di NetBSD è in grado di eseguire anche una gran parte di programmi Linux in modalità emulazione. Generalmente, quando si parla di emulazione si pensa a qualcosa di molto lento e inefficiente poiché, solitamente, le emulazioni comportano la riproduzione via software di architetture hardware spesso molto differenti da quella su cui si lavora. Nel caso dell'emulazione Linux, invece, si tratta semplicemente dell'aggiunta di un sottile strato di software che "adatti" una serie di chiamate di sistema già in partenza molto simili. Il codice dell'applicazione stessa viene elaborato nativamente dalla CPU e quindi, nessuna paura di performance scadenti, anzi c'è chi, con una buona dose di esagerazione, arriva a dire che i programmi per Linux girano addirittura meglio sotto NetBSD.

In questo capitolo viene spiegato come configurare l'emulazione Linux per installare un programma per Linux, che, nell'esempio, sarà il ben noto Acrobat Reader versione 4.

16.1. Setup dell'emulazione

La pagina di manuale compat_linux(8) descrive in dettaglio l'installazione dell'emulazione Linux su un sistema NetBSD. Si tratta di una delle pagine più user-friendly del manuale; per una volta tanto appare chiara l'intenzione di aiutare l'utente a compiere un'operazione anziché di fornire una elenco di specifiche e di dettagli tecnici. Per installare l'emulazione Linux sono necessari solo due passi:

1. Configurare il kernel.
2. Installare le librerie Linux.

16.1.1. Configurazione del kernel

La prima operazione da compiere è la compilazione di un kernel che supporti la compatibilità Linux. Chi usa il kernel generico non deve fare nulla perché GENERIC è già abilitato per la compatibilità con Linux. Chi usa un kernel personalizzato, invece, deve attivare le opzioni

```
option COMPAT_LINUX
option EXEC_ELF32
```

quando il kernel è pronto, si può passare all'installazione del software necessario.

16.1.2. Installazione delle librerie Linux

Le librerie Linux si possono ricavare da una distribuzione Linux, purché sufficientemente aggiornata. Per semplificare l'operazione conviene ricorrere al sistema dei package, che effettua l'installazione in modo automatico, utilizzando le librerie di Suse Linux. Le operazioni principali che verranno compiute sono:

- La creazione di una directory *root secondaria*, da utilizzare per i programmi Linux. Questa directory è `/emul/linux/`. I programmi Linux in emulazione su NetBSD faranno riferimento a questa directory come se fosse la directory radice del sistema.
- L'installazione delle librerie condivise per Linux. La maggior parte dei programmi sono infatti collegati dinamicamente e si aspettano quindi di trovare le librerie necessarie sul sistema. Per quanto

riguarda Acrobat Reader, portandosi nella directory `/usr/pkgsrc/print/acroread` e scrivendo **make depends** si ottiene il seguente messaggio:

```
===>  acroread-4.0 requires Linux glibc2 libraries - see compat_linux(8).
```

Entrambe le operazioni verranno compiute dal sistema dei package, senza bisogno di interventi manuali da parte dell'utente.

Prima di passare all'installazione delle librerie vere e proprie, bisogna installare il pacchetto necessario per la gestione dei file in formato *RPM*: si tratta di `rpm-2.5.4`, che consentirà l'estrazione dei file delle librerie Suse.

Successivamente bisogna installare il pacchetto `suse_base`. I file RPM Suse possono essere scaricati automaticamente dal sistema dei package oppure, chi ha un cdrom Suse li può copiare nella directory `/usr/pkgsrc/distfiles/suse` e poi eseguire **make** e **make install**.

Con lo stesso metodo si installano, `suse_compat`, `suse_libc5` e `suse_x11` ottenendo alla fine questa configurazione:

```
# 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. Installazione di Acrobat Reader

Ora tutto è pronto per l'installazione del programma Acrobat Reader (o di altri programmi Linux, naturalmente). Basta portarsi nella directory `/usr/pkgsrc/print/acroread` e dare i soliti comandi:

```
make
make install
```

Lo script di installazione di Acrobat chiede di accettare le condizioni della licenza; fatto ciò il programma è pronto per essere lanciato.

16.2. Struttura delle directory

Esaminando i risultati dell'installazione delle librerie e del programma, troviamo che `/emul/linux` è un link simbolico che punta a `/usr/pkg/emul/linux`, dove si trovano le seguenti directory:

```
bin/
boot/
cdrom/
dev/
etc/
floppy/
home/
lib/
```

```
mnt/
opt/
proc/
root/
sbin/
usr/
```

> **Nota:** si noti che è sempre bene fare riferimento a `/emul/linux`. Il fatto che si tratti di un link simbolico e `/usr/pkg/emul/linux` è un dettaglio di implementazione che potrebbe cambiare nelle future release.

Quanto spazio è necessario per installare l'emulazione Linux? A installazione avvenuta, la valutazione dello spazio da' i seguenti risultati:

```
# cd /usr/pkg/emul
# du -k linux
...
60525   linux/
```

Acrobat Reader, il programma vero e proprio, è stato installato nella consueta directory dei package, `/usr/pkg/bin/`.

Capitolo 17. Audio

Manolo De Santis

L'autore di questo capitolo (Audio) è Manolo De Santis

Questo capitolo vuole essere una breve guida per l'uso dell'audio su NetBSD, visto che nessuno amerebbe mai un computer muto...

17.1. Un po' di conoscenza dell'hardware

Innanzitutto bisogna sapere con che tipo di chipset audio abbiamo a che fare. Infatti non basta sapere chi ha prodotto la scheda audio per riuscire a farla funzionare con NetBSD, perché a volte i chipset di certe schede sono prodotti da terze parti. Ma niente paura! Nella maggioranza dei casi il kernel di NetBSD è in grado di rilevare il dispositivo, per cui basta dare un'occhiata all'output prodotto da dmesg.

Quindi dare un bel:

```
# dmesg | more
```

per vedere quale chipset audio sia stato individuato. In molti casi non si deve neppure intervenire sul kernel perché NetBSD individua e autoconfigura (quasi!) ogni scheda audio con molta facilità.

A volte non si riesce a far funzionare l'audio solo perché il chipset della scheda non è supportato da NetBSD (e qui molta gente ha ben pensato di acquistare una nuova scheda) o solo perché bisogna perderci un pò di tempo sopra.

17.2. Impostazione del BIOS

Ebbene, questa sezione sarà utile solo ai possessori di PC e compatibili, visto che su altre architetture (tipo Amiga) non serve a molto, se non a niente. La cosa più importante per utilizzare una scheda audio su un PC con NetBSD è la configurazione del BIOS. Infatti, bisogna sapere qual è il tipo di interfaccia della scheda audio in possesso.

La scelta è tra PCI e ISA.

Una scheda di tipo ISA è in genere più difficile da configurare, soprattutto perché il BIOS ci mette ogni tanto lo zampino...

Sui nuovi computer in commercio (quelli prodotti in genere dopo la fine del 1997) c'è un'opzione del BIOS che crea grossi problemi alle schede audio ISA su NetBSD. Questa è 'PNP OS Installed' (si trova in genere in 'PNP/PCI Configuration'). Ebbene se questa opzione è presente bisogna disabilitarla. Quindi, impostarla su 'NO'.

> **Nota:** può anche darsi che il tutto funzioni pur lasciando la suddetta opzione abilitata, non si sa mai!

17.3. Configurazione dei dispositivi

Durante l'installazione di NetBSD vengono creati tutti i device in `/dev`. A noi interessano principalmente:

```
/dev/audio
/dev/sound
/dev/mixer
```

Se essi non sono presenti li possiamo creare così:

```
# cd /dev
# ./MAKEDEV all
```

In tal modo vengono ricreati tutti i device, compresi quelli relativi all'audio.

Bene, a questo punto è molto probabile che la scheda audio sia già pronta per l'uso.

Si può subito testarla inviando un qualsiasi file (anche di testo o binario) direttamente a `/dev/audio` oppure `/dev/sound` in questo modo:

```
# cat NomeDelFile > /dev/audio
```

oppure anche

```
# cat NomeDelFile > /dev/sound
```

Se è possibile sentire qualcosa, allora non ci sono problemi, tutto funziona correttamente perché la scheda è direttamente supportata dal kernel del nostro sistema.

In caso contrario bisogna configurare il kernel impostando i device relativi alla nostra scheda audio.

17.4. Consigli sulla configurazione del kernel

Le schede audio supportate da NetBSD sono moltissime, infatti nel kernel GENERIC della versione 1.5 del sistema operativo sono già abilitate tutte le opzioni relative.

Alcuni PC non hanno la cosiddetta scheda sonora, bensì un chipset audio integrato nella mother board, e tale funzione non è abilitata nel kernel GENERIC. Bisogna individuare queste linee proprio nel file di configurazione del 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
```

A questo punto bisogna togliere i commenti per abilitare i chipset audio integrati e ricompilare il kernel.

> **Nota:** personalmente credo sia meglio fare una copia del file di configurazione GENERIC e lavorare su di essa, lasciando quindi invariato il file originale, come spiegato nel Capitolo 7.

Tornando alle schede audio, talvolta si devono impostare manualmente anche gli IRQ e i DMA.

Per una prova si possono lasciare tutte le opzioni abilitate perché molti chipset audio possono funzionare solamente emulando un altro tipo di dispositivo.

Generalmente molti chipset sfruttano la compatibilità con SoundBlaster, OPL ma nella maggioranza dei casi l'audio funziona sfruttando WSS.

OPL è un sintetizzatore MIDI prodotto da Yamaha e ne esistono diverse varianti in commercio (tra cui OPL2, OPL3SA, OPL3SA2, ecc.). Posso dire che moltissime schede audio in commercio possiedono questo microchip oppure uno totalmente compatibile. Ad esempio tutti i chip prodotti da Crystal (tra cui i comuni CS423x) possiedono questo chip, ed è proprio quest'ultimo a permetterne il funzionamento con NetBSD!

WSS non è proprio un microchip, anzi... è l'acronimo di Windows Sound System. Infatti wss è un driver del kernel di NetBSD che supporta il sistema audio di Microsoft Windows. Molte schede audio funzionano con Windows proprio perchè aderiscono a questo standard, WSS appunto. La stessa cosa fortunatament e vale anche per NetBSD.

Personalmente ho avuto modo di provare svariate schede con NetBSD e posso dire che molte di esse funzionano solo se opl* e wss* sono abilitati nel kernel!

Nessun problema con le Sound Blaster originali Creative: funzionano tutte, compresa la Sound Blaster Live 1024!

Una volta che tutto funziona correttamente si possono disabilitare dal kernel le opzioni relative ai devices audio che non ci interessano.

17.5. Comandi avanzati

NetBSD possiede delle utility progettate esclusivamente per il controllo e l'utilizzo dei device audio. Esse sono:

- audioctl
- mixerctl
- audioplay
- audiorecord

17.5.1. audioctl

Apparso dapprima in NetBSD 1.3, serve per impostare determinate variabili, tra cui le frequenze di campionamento per riproduzione e registrazione. Per vedere ciò che è possibile impostare dare un bel:

```
# audioctl -a | more
```

Ad esempio, per ascoltare musica in qualità CD, dare:

```
# audioctl -w play=44100,2,16,slinear_le
```

Questo imposta la frequenza a 44100Hz, 2 canali audio, 16 bit e cambio dell'encoding.

Per vedere quali sono gli encoding supportati:

```
# audioctl encodings
```

apparirà una lista di tutti gli encoding supportati dalla scheda audio in uso.

17.5.2. mixerctl

Serve chiaramente a regolare il mixing audio e funziona in modo analogo ad audioctl.

17.5.3. audioplay

Con questo è possibile ascoltare file audio, ma consiglio di procurarsi alcuni package che danno la possibilità di ascoltare svariati formati.

17.5.4. audiorecord

Utile per la registrazione dell'audio.

Capitolo 18. Ottenere i sorgenti con CVS

Reinoud Koornstra

L'autore di questo capitolo (Ottenere i sorgenti con CVS) è Reinoud Koornstra

Il CVS (Concurrent Versions System) è un sistema di controllo versione che può essere utilizzato per mantenere aggiornato l'albero dei sorgenti del sistema presente in locale con le modifiche apportate ai sorgenti "ufficiali" di NetBSD. Vi sono tre alberi (o, più precisamente, tre rami di uno stesso albero) che possono essere usati come riferimento per mantenere aggiornati i sorgenti locali: il ramo *current*, con il quale si può seguire la linea di sviluppo corrente (cioè quella più avanzata ma potenzialmente meno stabile), il ramo relativo alla *release 1.5*, al quale vengono applicate le correzioni relative alla release 1.5 per sistemare errori e problemi di sicurezza e il ramo della *release 1.4*, al quale vengono apportate le correzioni relative alla versione 1.4.

> **Nota:** i tre rami appena descritti sono quelli esistenti al momento della scrittura di questo capitolo. In futuro i numeri di versione delle release cambieranno.

18.1. Ottenere i sorgenti di sistema e di userland

> **Nota:** con il termine "userland" si intende l'insieme di programmi che fanno parte del sistema operativo NetBSD ma che non fanno parte del "kernel" vero e proprio.

Il sistema di controllo versione CVS non è compreso nelle versioni 1.4.x e 1.5.x di NetBSD (ma lo sarà per quanto riguarda la 1.6) e pertanto, se non lo avete già fatto, è necessario installarlo:

```
% pkg_add ftp://ftp.netbsd.org/pub/NetBSD/packages/<OS Ver>/<arch>/All/cvs-1.11nb2.tgz
```

I parametri <OS Ver>, e <arch> si ottengono con il seguente comando:

```
% sysctl kern.osrelease hw.machine_arch
```

Per scaricare i sorgenti partendo da zero, cioè senza avere niente in `/usr/src`:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs checkout -rnetbsd-1-5 -PA src
```

È possibile anche utilizzare ssh, trasferendo i dati crittografati:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
```

```
% cd /usr
% cvs checkout -rnetbsd-1-5 -PA src
```

I precedenti comandi recuperano i sorgenti del ramo relativo alla release 1.5. Per scaricare i sorgenti correnti basta omettere il parametro “-rnetbsd-1-5” dall’ultimo comando. Per scaricare i sorgenti del ramo relativo alla release 1.4, utilizzare il parametro “-rnetbsd-1-4” nell’ultima riga.

Per aggiornare un albero di sorgenti già esistente, per esempio per la release 1.5, dare i seguenti comandi:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

o, con ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

Per aggiornare un albero dei sorgenti con la versione corrente bisogna omettere il parametro “-rnetbsd-1-5” Per aggiornare un albero dei sorgenti con la release 1.4, utilizzare il seguente parametro: “-rnetbsd-1-4”

Per effettuare l’aggiornamento di un albero dei sorgenti non “pulito” (perché è stato fatto un “build” di una parte o di tutti i sorgenti oppure del kernel stesso) sul quale non è stato effettuato un **make cleandir**, è necessario effettuare un **make obj** in `/usr/src`:

```
% makedir /usr/obj
% make obj
```

> **Nota:** le directory object sono necessarie per effettuare un’operazione di update di un albero non pulito. Quando viene effettuata un’operazione di “build” su un albero o su una sua parte senza farla seguire da un “make clean” o da un “make cleandir”, si dice che l’albero rimane *non pulito*. In questo caso CVS potrebbe cercare di creare delle directory che hanno lo stesso nome di alcuni file binari già esistenti (per esempio dove c’era la directory “groff” viene ora costruito un binario che si chiama “groff”, ma CVS deve creare tutte le directory vuote prima di eliminarle).

Ora è possibile effettuare l’aggiornamento con “cvs update”. Oppure si può dare un **make cleandir** in `/usr/src` prima di usare il CVS. Questo metodo è più semplice e più veloce rispetto al make delle objectdir quando si deve aggiornare un albero non pulito.

L’esecuzione è notevolmente più veloce rispetto a sup e gli aggiornamenti sono più frequenti.

Non è possibile definire con esattezza il tempo necessario per lo scaricamento di tutti i sorgenti. A titolo di esempio, su una linea T1 l'operazione richiede poco più di un'ora, a seconda della qualità della connessione. Per utilizzare un modem conviene sfruttare la compressione dei dati. Per esempio:

```
% cvs -z5 checkout ...
```

oppure

```
% cvs -z5 -d $CVSROOT update ...
```

Negli esempi precedenti, il numero 5 indica il livello di compressione desiderato; si può specificare un numero compreso tra 1 e 9, dove 1 rappresenta la compressione più veloce e 9 quella più efficace ma più lenta. Si tenga presente che la compressione aumenta il carico di lavoro del server.

> **Nota:** è necessario portarsi in `/usr/src/sys/arch/$arch/compile/$kernel_conf_name`, dare il comando **make clean** e rimuovere la directory `$kernel_conf_name`, prima di effettuare un "cvs update", perché anche quella directory potrebbe non essere "pulita". In questo modo si garantisce che l'aggiornamento della parte relativa al kernel andrà a buon fine.

18.2. Recupereare pkgsrc

Pkgsrc (i sorgenti del sistema dei package) comprendono un insieme di utility e di file di configurazione che consentono di compilare e di utilizzare con NetBSD un nutrito gruppo di applicazioni, rendendo semplice l'installazione e la disinstallazione di software sul sistema.

Per scaricare tutto pkgsrc partendo da zero:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
(the password is: "anoncvs")
% cvs checkout -PA pkgsrc
```

o, con ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs checkout -PA pkgsrc
```

I precedenti comandi creano la directory `pkgsrc` in `/usr` e memorizzano i sorgenti dell'intero sistema dei package in `/usr/pkgsrc`

Per aggiornare pkgsrc dare i seguenti comandi:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
```

```
(the password is: "anoncvs")
% cvs -d $CVSROOT update -PAd pkgsrc
```

o, con ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -PAd pkgsrc
```

Prima di effettuare l’aggiornamento è bene assicurarsi che la directory `pkgsrc` sia “pulita”. Se non si è sicuri, dare il comando **make clean** in **`/usr/pkgsrc`**.

Capitolo 19. CCD: configurazione

Brian A Seklecki

L'autore di questo capitolo (CCD: configurazione) è Brian A. Seklecki <lavalamp@burghcom.com>

Il driver CCD consente all'utente di "concatenare" più dischi fisici tra loro formando un unico pseudo-volume. CCD consente inoltre di superare una limitazione di CMU RAIDFrame, che non permette di configurare RAID0 (file system su più dischi) su dischi di geometria differente. CCD consente inoltre di utilizzare un'"interleave" per migliorare le prestazioni diminuendo lo spreco di spazio (questa caratteristica non viene descritta nel presente esempio).

I passi necessari per la configurazione di CCD sono i seguenti:

1. Installazione dei dischi
2. Configurazione del supporto a CCD nel kernel
3. Creazione della disklabel per ogni volume membro del CCD
4. Creazione del file di configurazione del CCD
5. Inizializzazione del dispositivo CCD
6. Creazione di un file system 4.4BSD/UFS sul nuovo dispositivo CCD
7. Mount del file system CCD

Questo esempio fa riferimento all'impostazione di CCD su NetBSD/Sparc 1.5. Il CCD risiederà su quattro dischi SCSI in uno chassis esterno della Sun collegato al port esterno SCSI da 50 pin.

19.1. Installazione dei dischi

Questa fase dipende naturalmente dalla piattaforma e dall'hardware utilizzato.

Esempio di **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/di-
rect 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/di-
rect 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. Configurazione del kernel

Per il supporto ai dispositivi CCD è necessario attivare la seguente opzione nel file di configurazione del kernel (attivata di default in GENERIC):

```
pseudo-device  ccd  4    # concatenated disk devices
```

Nel mio file di configurazione del kernel, codifico direttamente l'associazione tra gli "id" SCSI e i corrispondenti dispositivi `/dev`, per avere il massimo controllo:

```
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. Scrittura delle disklabel

È necessario creare un file sistem speciale per ogni disco membro del CCD. Nel presente esempio, si dovrà creare la disklabel per:

```
/dev/rsd0c
/dev/rsd1c
/dev/rsd2c
/dev/rsd3c
```

si ricordi di creare sempre la disklabel per il dispositivo a caratteri, non per quello a blocchi. Per esempio: `/dev/r{s,w}d*`

> **Nota:** su tutte le piattaforme, la slice `c` è riservata e rappresenta l'intera partizione NetBSD.

Sulla piattaforma i386, la slice `d` è riservata e rappresenta l'intero disco.

Prima di creare le nuove disklabel si possono rimuovere quelle preesistenti con una serie di comandi simili ai seguenti:

Nota: sulla piattaforma i386, sostituire `rsd#c` a `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
```

La disklabel di default per i dischi sarà simile alla seguente:

```
# 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*)
```

È necessario creare una "slice" nella partizione NetBSD che occupi l'intera partizione. La slice deve iniziare almeno un cilindro dopo l'inizio del disco o della partizione per lasciare spazio alla speciale disklabel richiesta da CCD. L'offset deve essere 1 x settori/cilindro. Pertanto, il valore di "size" deve essere uguale al numero di settori totali meno il numero di settori presenti in un cilindro.

Nota: accertarsi che l'offset di una slice di tipo "ccd" sia un multiplo del valore "settori/cilindro".

Modificare la disklabel di conseguenza, specificando il percorso del dispositivo a caratteri, non di quello a blocchi.

Nota: prima di iniziare l'editing delle disklabel, verificare che sia esportata la variabile di ambiente EDITOR.

```
# disklabel -e /dev/rsd0c
```

Nota: il tipo ("fstype") della slice deve essere `ccd`.

Poichè la partizione conterrà una slice sola, si può riciclare la slice `c`, che normalmente è riservata. La disklabel viene quindi impostata così:

```
3 partitions:
#        size   offset    fstype   [fsize bsize   cpg]
  c:  4196361     1044       ccd                       # (Cyl. 1 - 4020*)
```

È anche possibile impostare una slice diversa da `c`, come nell'esempio seguente:

```
3 partitions:
#        size   offset    fstype   [fsize bsize   cpg]
  a:  4196361     1044       ccd                       # (Cyl. 1 - 4020*)
  c:  4197405        0    unused     1024  8192        # (Cyl. 0 - 4020*)
```

Al termine dell'operazione ci si assicuri di scrivere la disklabel, che deve superare i controlli di validità, pena il rifiuto da parte del comando **disklabel**.

19.4. Configurazione del CCD

Una volta terminata la creazione delle disklabel per tutti i dischi, si può passare alla scrittura del file di configurazione, che risiede nella directory `/etc` di default. Il formato è il seguente:

```
#ccd    ileave    flags   component    devices
```

Nota: per quanto riguarda il valore di "ileave", se viene messo zero allora i dischi vengono concatenati, ma se si usa un valore uguale al numero di settori/cilindro allora i dischi sono interlacciati.

Nell'esempio corrente, il contenuto del file è:

```
# more /etc/ccd.conf
ccd0  0  none /dev/sd0c /dev/sd1c /dev/sd2c /dev/sd3c
```

Nota: il file di configurazione di CCD fa riferimento ai file system CCD appena creati. Bisogna utilizzare i dispositivi a caratteri, e non quelli a blocchi.

19.5. Inizializzazione del CCD

Dopo aver verificato la correttezza delle impostazioni appena effettuate, si può finalmente inizializzare il dispositivo usando il comando **ccdconfig**: Configure:

```
# ccdconfig -c -f /etc/ccd.conf
```

La configurazione si cancella con il flag `-u`:

```
# ccdconfig -u -f /etc/ccd.conf
```

L’inizializzazione del CCD attiverà i dispositivi `/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. Creazione del file system sul dispositivo CCD

Ora si può creare la disklabel del nuovo disco virtuale associato con il CCD (usare il dispositivo a caratteri):

```
# disklabel -e /dev/rccd0c
```

Come in precedenza, visto che viene usata una slice sola è possibile riciclare la slice `c` oppure creare una slice separata.

```
# 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 bisogna anche creare il file system:

```
# 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. Montare il file system

Il file system del dispositivo CCD appena creato può essere montato su un punto di mount del sistema. Deve essere montata la slice etichettata `ffs` oppure `4.4BSD`:

```
# mount /dev/ccd0c /mnt
```

Poi:

```
# 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
```

Al termine di questo esempio dovreste essere in grado di configurare un dispositivo CCD. Consultate le pagine di manuale per ulteriori dettagli sui comandi descritti in questo capitolo e per attivare automaticamente il dispositivo CCD all’avvio mediante i file RC.

Capitolo 20. Operazioni varie

Questo capitolo raccoglie in ordine sparso vari argomenti, che non hanno trovato collocazione negli altri capitoli.

20.1. Creazione di floppy di installazione

L'autore di questa sezione (Creazione di floppy di installazione) è Eric Delcamp

Per poter creare dei floppy di installazione è necessario che nel kernel sia abilitato lo "pseudo-device" *vnd*.

1. Per prima cosa bisogna creare il kernel da mettere sui floppy, che supponiamo di chiamare FLOPPY, che deve derivare da uno dei kernel INSTALL.
2. Una volta compilato, il kernel si troverà nel file `/sys/arch/i386/compile/FLOPPY/netbsd`.
3. Portarsi nella directory `/usr/src/distrib/i386/floppies/ramdisk` e dare il comando

   ```
   # make
   ```

   Questo crea il file `ramdisk.fs` in quella directory.
4. Portarsi nella directory `/usr/src/distrib/i386/floppies/fdset` e dare il comando:

   ```
   # make KERN=/sys/arch/i386/compile/FLOPPY/netbsd
   ```

5. Il comando precedente ha creato uno o due file (dipende dalla dimensione del kernel) di nome `boot1.fs` e `boot2.fs`. A questo punto bisogna trasferire questi file su floppy con il comando (o meglio, i comandi):

   ```
   # dd if=boot1.fs of=/dev/fd0a bs=36b
   # dd if=boot2.fs of=/dev/fd0a bs=36b
   ```

   Naturalmente tra un comando e l'altro bisogna mettere un nuovo floppy nel drive.

20.2. Creazione di CD-ROM

Per creare un CD-ROM di dati si utilizzano i programmi mkisofs e cdrecord. NetBSD supporta l'utilizzo sia di masterizzatori SCSI sia di masterizzatori IDE. Questi ultimi si possono utilizzare in maniera del tutto trasparente, senza bisogno di installare emulazioni SCSI o altre diavolerie perché il driver è in grado di ricevere direttamente comandi ATAPI (come sempre NetBSD riesce a trovare la soluzione più semplice e più elegante).

La creazione di un CD procede in due fasi: dapprima si crea un'immagine ISO del CD sul disco fisso utilizzando il programma mkisofs e poi si scrive l'immagine sul CD con cdrecord. Negli esempi seguenti si suppone di utilizzare un masterizzatore IDE/ATAPI tra quelli supportati da cdrecord: il masterizzatore viene riconosciuto dal sistema come master secondario

> **Nota:** la masterizzazione di un CD-ROM è un'operazione nella quale la velocità di esecuzione è critica; in particolare il flusso dei dati verso il masterizzatore deve essere costante: cdrecord non deve mai trovarsi con il buffer dei dati vuoto o quasi vuoto. Pertanto è consigliabile eseguire questo tipo di operazione quando il sistema è scarico (meglio non ricompilare il kernel mentre si masterizza, tento per intenderci).

```
cd1 at atapibus1 drive 0: <HP CD-Writer Plus 8100> type 5 cdrom removable
```

20.2.1. Creazione dell'immagine ISO

> **Nota a margineFile system su CD-ROM**
>
> I CD-ROM possono essere creati utilizzando diversi file system incompatibili (o talvolta parzialmente compatibili) tra loro. Pertanto un CD prodotto su un sistema non sempre è leggibile su un altro sistema e questa situazione appare evidente quando si cerca di leggere da NetBSD un CD prodotto da un programma di masterizzazione che opera sotto Windows.
>
> Il formato ISO9660 (del 1988) è il primo che è stato proposto e rappresenta una sorta di denominatore comune tra i vari sistemi operativi (Apple, MS-DOS, Unix e VMS). Un CD prodotto in ambiente Windows con questo formato sarà soggetto a varie limitazioni (nomi i file in formato 8.3 e al massimo 8 livelli di sottodirectory). Per questo motivo, quando si crea un CD di installazione di NetBSD sotto Windows 95/98/NT, i nomi dei file, che apparivano normali nella "Gestione risorse" di Windows, appariranno troncati sul CD.
>
> Per risolvere questo tipo di limitazioni sono state introdotte delle estensioni al formato ISO9660 originale. Lo svantaggio di queste estensioni è che, in generale, non sono supportate da tutti i sistemi operativi.
>
> Il formato Joliet è stato introdotto dalla Microsoft per i propri sistemi, ed espande l'ISO9660 consentendo, per esempio, nomi di file lunghi. In generale un CD in questo formato non è leggibile sui sistemi Unix (NetBSD-current supporta anche il formato Joliet).
>
> Per i sistemi Unix, invece, sono state introdotte le estensioni Rock Ridge, che consentono il supporto delle convenzioni Unix pur mantenendo la compatibilità con il formato ISO9660. È questo il formato da usare per creare un CD per sistemi Unix sotto NetBSD (come si vedrà, è anche possibile creare CD compatibili RockRidge/Joliet).
>
> In aggiunta a questi standard, ne esistono altri (per esempio "El Torito"), che consentono di produrre dei CD “avviabili”, ormai supportati da tutti i PC abbastanza recenti.

Per creare l'immagine del CD su disco fisso bisogna innanzi tutto che ci sia spazio sufficiente sul disco: a seconda della quantità di dati può servire fino a 700MB. Supponendo che i dati che ci interessano siano nella directory `mieidati`, bisogna scrivere

```
# mkisofs -aflrTv -o cdimage mieidati/
```

Una volta creato il file immagine `cdimage`, lo si può esaminare montandolo come se fosse un filesystem per controllare che non vi siano errori e non sprecare un prezioso 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
```

Il valore 556 è stato ottenuto dividendo la dimensione del file, 284672, per 512.

> **Creare un CD ibrido:** mkisofs è in grado anche di creare CD in formato Joliet (leggibili dai sistemi Microsoft) e, persino, CD ibridi, che usano sia il formato RockRidge sia il formato Joliet e risultano quindi leggibili sia da NetBSD sia da Windows. Per esempio, per creare un CD ibrido:
>
> ```
> $ mkisofs -l -J -R -o cd.iso mieidati/
> ```
>
> Fare riferimento alla pagina di manuale di mkisofs per i dettagli sulle opzioni accettate dal programma.

20.2.2. Scrittura dell'immagine su disco

Nella seconda fase l'immagine appena creata viene scritta sul CD con il comando

```
# cdrecord -v speed=2 dev=/dev/rcd1d cdimage
```

> **Nota:** nel caso di CDR ATAPI è necessario usare il dispositivo *rcd#d*, perché il dispositivo 'a' non supporta l'invio diretto di comandi ATAPI.

è anche possibile effettuare un test di scrittura, disabilitando il laser, prima di procedere alla masterizzazione vera e propria; basta aggiungere al comando precedente le opzioni `-dummy` e `-nofix`. Per esempio

```
# cdrecord -v -dummy -nofix speed=2 dev=/dev/rcd1d cdimage
```

La fase di creazione dell'immagine e quella di scrittura su disco possono essere combinate in un'unico *job*, evitando la generazione del file immagine temporaneo sul disco fisso. Il comando è il seguente

```
# (nice -18 mkisofs -aflrT mieidati/) | cdrecord -v fs=16m speed=2 dev=/dev/rcd1d -
```

L'opzione `fs=16m` serve per aumentare la dimensione della *fifo*, evitando errori di underflow del buffer dei dati usato da cdrecord.

20.2.3. Copia di un CD

Per effettuare la copia diretta di un CD si usa l'opzione `-isosize` di cdrecord, dando il seguente comando

```
# cdrecord -v fs=16m -isosize speed=2 dev=/dev/rcd1d /dev/rcd0d
```

Nota: se si utilizzano due CD(-RW) IDE/ATAPI, è consigliabile montarli su canali diversi per avere un migliore flusso di dati. Per esempio:

```
wd0: hard disk master ide primario
cd0: lettore CD slave ide primario
cd1: masterizzatore CD master ide secondario
```

20.2.4. Creazione di un CD-ROM avviabile

Non ci sono particolari differenze tra la creazione di un cdrom normale e di uno avviabile. Per quest'ultimo occorre avere una copia del file di boot da mettere sul cd e informare mkisofs. Per esempio:

```
# mkisofs -avr -b boot.fs mieidati/ > cdimage
```

`boot.fs` è il file di boot del CD-ROM, il cui percorso deve essere indicato in modo relativo alla directory `mieidati`.

20.3. Sincronizzare l'orologio di sistema

Per qualche strano motivo sembra che gli orologi di sistema non funzionino mai granché bene, e non è raro trovarsi con sfasamenti di parecchi minuti. Il problema si aggrava quando si ha a che fare con più host collegati in rete: con buona probabilità non si riuscirà mai a mantenerli sincronizzati. Per risolvere il problema viene il nostro aiuto il protocollo NTP (Network Time Protocol) versione 3, che consente di mantenere sincronizzati gli orologi di più workstation collegate in rete utilizzando dei server appositi.

Grazie al protocollo NTP è possibile sia semplicemente mantenere corretta l'ora di sistema di un singolo host, sia sincronizzare un'intera rete. Si tratta di un protocollo piuttosto complesso, che definisce la struttura gerarchica master-slave di una serie di server suddividendoli in strati: in cima alla gerarchia vi sono i server di strato 1, connessi a una sorgente hardware di riferimento per la data e l'ora. I server di strato 2 si sincronizzano via NTP con quelli di strato 1 e, a loro volta, fanno da server secondari per gli host degli strati inferiori. Questa struttura gerarchica evita la congestione che si avrebbe se tutti gli host client facessero riferimento ai server di strato 1. Se, per esempio, si vuole sincronizzare un'intera rete, non si devono collegare tutti gli host allo stesso server pubblico di strato 1; invece, si crea un server locale che si sincronizza con quello pubblico e i restanti host si sincronizzano con il server locale.

Fortunatamente, per sfruttare gli strumenti NTP non è necessario comprendere i dettagli del protocollo e della sua implementazione (chi fosse interessato può far riferimento al documento RFC 1305) ma è sufficiente sapere come avviare alcuni programmi. Su NetBSD sono già presenti i programmi necessari per sfruttare questo protocollo, derivati dall'implementazione xntp. In questa sezione viene descritto un semplice metodo per mantenere aggiornata l'ora di sistema.

Per prima cosa bisogna procurarsi l'indirizzo dei server NTP pubblici da utilizzare come riferimento; un elenco completo si può trovare su http://www.eecis.udel.edu/~mills/ntp/servers.html, ma per l'Italia possono essere usati tempo.cstv.to.cnr.it e time.ien.it, che sono due server di strato 1.

A questo punto, per sincronizzare l'orologio di sistema basta dare il comando (è necessario essere root):

```
# ntpdate -b tempo.cstv.to.cnr.it time.ien.it
```

L'opzione `-b` serve per indicare a **ntpdate** di impostare direttamente il nuovo valore dell'ora di sistema anziché cercare di correggere quello attuale, ed è consigliata quando la differenza tra l'ora corretta (quella del server) e quella locale può essere notevole.

Abbiamo visto come funziona ntpdate; adesso bisogna decidere il modo migliore per avviarlo in automatico. Se l'host da sincronizzare è sempre collegato a Internet, la cosa migliore è lanciare il programma a ogni riavvio specificando

```
ntpdate=YES      ntpdate_hosts="time.ien.it"
```

in `/etc/rc.conf`. Il nome del server può essere specificato in ntpdate_hosts, altrimenti lo script andrà a estrarlo dal file `/etc/ntp.conf`.

Se l'host viene collegato a Internet tramite un provider e la connessione non è sempre attiva, si può far lanciare il comando ntpdate dallo script `ip-up`, come spiegato nel Capitolo 10. In questo caso la riga da inserire nel file è:

```
/usr/sbin/ntpdate -s -b time.ien.it
```

(il path è necessario). L'opzione `-s` causa il logging dei messaggi, che si possono poi ritrovare in `/var/log/messages` (salvo diversa configurazione di `syslog.conf`).

Oltre a ntpdate sono presenti vari altri comandi ed è anche possibile far diventare un host della rete il server NTP degli altri host client. Il server interno si sincronizzerà con i server di riferimento esterni alla rete e i client si sincronizzeranno con il nostro server. Per questo tipo di configurazione si deve utilizzare il demone **xntpd**, creando l'apposito file di configurazione `/etc/ntp.conf`. Ad esempio:

```
server time.ien.it
server tempo.cstv.to.cnr.it
```

Xntpd si può anch'esso avviare da `rc.conf`, abilitando l'apposita opzione.

```
xntpd=YES
```

Per la sincronizzazione interna alla rete, in alternativa alla configurazione descritta è anche possibile usare il demone timed, nato specificamente con 4.3BSD. Anche timed si basa su uno schema master-slave: quando viene avviato su una macchina, timed chiede al master l'ora di sistema della rete e corregge l'ora della macchina stessa appropriatamente. Si può quindi utilizzare una struttura mista, nella quale uno degli host si sincronizza con un server NTP pubblico e fa da master per la rete locale, i cui client si sincronizzano con timed. Sull'host di riferimento devono essere attivi sia NTP sia timed, e bisogna fare attenzione a non farli interferire (timed deve essere avviato con l'opzione `-F hostname`; in questo modo non tenta di aggiornare l'orologio locale).

20.4. Installare il boot manager

Il programma di installazione di NetBSD può installare il proprio boot manager, se l'utente lo richiede. In caso di errata o mancata installazione è possibile installarlo in un secondo tempo o modificarne le proprietà con il comando **fdisk**. Per esempio:

```
# fdisk -B wd0
```

Se il sistema non si avvia dal disco fisso, si può fare il boot con il dischetto di avviamento usato per l'installazione e far partire il kernel sul disco fisso con un comando di questo tipo:

```
> boot wd0a:netbsd
```

Nota: in alcuni casi **fdisk -B** non sembra dare il risultato voluto, per esempio se installano e rimuovono spesso altri sistemi operativi come Windows 95. In questo caso può servire fare un **fdisk /mbr** da DOS (per esempio con un dischetto) e poi riprovare. Lo stesso risultato si può ottenere da NetBSD con il comando **fdisk -i wd0**.

20.5. Cancellare la disklabel

Benché la cancellazione della disklabel non sia un'operazione frequente può essere utile sapere come fare in caso di necessità. Naturalmente si tratta di un'operazione da eseguire solo se si sa esattamente quello che si sta facendo. Per esempio:

```
# dd if=/dev/zero of=/dev/rwd0c bs=8k count=1
```

In questo modo viene cancellata la disklabel ma non la tabella delle partizioni del MBR. Per cancellare completamente il disco usare `wd0d`. Per esempio:

```
# dd if=/dev/zero of=/dev/rwd0d bs=8k
```

20.6. Speaker

Per produrre un suono sullo speaker (per esempio un avvertimento alla fine di uno script) si può utilizzare il device *spkr* del kernel che viene mappato sul dispositivo `/dev/speaker`. Per esempio, sulle mailing list ho trovato un post che suggeriva:

```
echo 'BPBPBPBPBP' > /dev/speaker
```

Nota: il dispositivo *spkr* non è abilitato nel kernel generico e, pertanto, per usarlo bisogna compilare un kernel personalizzato.

20.7. Memory file system

Per migliorare le prestazioni in presenza di determinate configurazioni si può utilizzare un *memory filesystem*. A titolo di esempio, questa strategia può essere utile quando si compila molto e si dispone di sufficiente memoria. Per creare un file system in memoria, aggiungere una riga di questo tipo a `/etc/fstab`:

```
/dev/wd0b /tmp  mfs     rw,async,noatime,nosuid,-s=20000
```

l'opzione `-s=20000` imposta le dimensioni a circa 10 MB (20000 settori), che dovrebbero essere sufficienti per la compilazione del kernel.

20.8. Dimenticata la password di root?

Quando si dimentica la password di root non tutto è perduto, e si può ancora "recuperare" il sistema. I passi da compiere sono i seguenti: fare il boot in modalità monoutente, montare / e cambiare la password di root. In dettaglio:

1. Fare il boot in modalità monoutente: quando compare il prompt di boot e inizia il conto alla rovescia di cinque secondi dare il comando:

   ```
   > boot -s
   ```

2. Alla richiesta

   ```
   Enter pathname of shell or RETURN for sh:
   ```

   premere Invio.

3. Dare i seguenti comandi:

   ```
   # fsck -y /
   # mount -u /
   # fsck -y /usr
   # mount /usr
   ```

4. Cambiare la password di root con **passwd**.
5. Dare il comando **exit** per passare alla modalità multiutente.

20.9. Aggiungere un nuovo disco fisso

Questa sezione spiega come aggiungere un nuovo disco fisso a un sistema NetBSD. Nell'esempio proposto verranno aggiunti un controller SCSI e un disco fisso collegato al nuovo controller. Chi non deve installare il controller può semplicemente saltare la lettura della parte relativa e andare direttamente alla configurazione dell'hard disk. L'installazione di un disco fisso IDE è identica a quella presentata nell'esempio; solo il nome del dispositico sarà diverso (`wd#` anziché `sd#`).

Come sempre, prima di acquistare nuovo hardware è bene consultare la lista delle compatibilità hardware di NetBSD, per accertarsi che il dispositivo scelto sia supportato dal sistema.

Completata l'installazione fisica del controller e del disco fisso, nonché il relativo collegamento si avvia il sistema e ci si accerta che il dispositivo sia riconosciuto con il solito **dmesg**. Per esempio, per un controller NCR-875 l'output è il seguente:

```
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 il dispositivo non compare nell'output bisogna accertarsi che il kernel in uso lo supporti ed eventualmente compilare un kernel personalizzato (si veda il Capitolo 7).

A questo punto si possono creare le partizioni con il comando **fdisk**. Per prima cosa visualizziamo lo stato attuale del 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>
```

In questo esempio la partizione DOS verrà eliminata e sostituita con una partizione NetBSD che occuperà l'intero disco. Il comando **fdisk -u sd0** consente di modificare interattivamente le partizioni. Le modifiche verranno scritte su disco soltanto alla fine, dopo aver chiesto conferma, il che consente di procedere con una certa tranquillità.

Nota a margineNota sulla geometria del disco

La geometria di questo disco può apparire abbastanza confusa. Dmesg riporta 4226725 settori con C/H/S uguale a 8188/3/172. Ma 8188*3*172=4225008 e non 4226725. In realtà, nella maggior parte dei dischi moderni la geometria non è fissa e il numero di settori per traccia cambia a seconda del cilindro. Il disco riporta ugualmente una geometria C/H/S fissa, ma si tratta di una geometria fittizia, e il valore 172 è ottenuto dividendo il numero totale di settori (che è effettivamente 4226725) per 8188 e poi per 3); il risultato della divisione intera è 172.

Il BIOS, poi, può utilizzare a sua volta una geometria "fittizia" diversa (e infatti usa C/H/S 524/128/63) che porta a 4225536 settori, che sono sempre un po' meno di quelli reali. Nel seguito, anche per mantenere la compatibilità con altri sistemi operativi, useremo la geometria vista dal BIOS, anche se questo porta allo spreco di alcuni settori (4226725-4225536=1189 settori= 594 KB.)

Dopo aver eseguito **fdisk -u**, confermiamo le modifiche e otteniamo le seguenti partizioni:

```
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>
```

A questo punto bisogna scrivere la disklabel sul disco. La sequenza corretta di operazioni è:

```
# disklabel sd0 > tempfile
# vi tempfile
# disklabel -R -r sd0 tempfile
```

Si noti che non è possibile effettuare direttamente

```
# disklabel -e sd0
```

in quanto si ottiene il seguente messaggio:

```
disklabel: ioctl DIOCWDINFO: No disk label on disk;
use "disklabel -r" to install initial label
```

dovuto al fatto che la disklabel non è ancora presente sul disco.

In questo esempio creiamo le seguenti partizioni con disklabel, editando il file `tempfile` come spiegato sopra:

```
#       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:** una volta creata la disklabel, la si può ottimizzare studiando l'output del comando **newfs -N /dev/sd0a** che segnala se alla fine di una partizione rimangono settori non allocati. Questo valore si può utilizzare per modificare la dimensione della partizione stessa.

In ultimo creiamo i nuovi file system per le partizioni *a* ed *e*.

```
# newfs /dev/sd0a
# newfs /dev/sd0e
```

Il disco è ora pronto per essere usato, con due partizioni da montare. Per esempio:

```
# mount /dev/sd0a /mnt
```

20.10. Password file is busy?

A alcuni utenti è capitato di ricevere il misterioso messaggio "Password file is busy" quando cercavano di modificare una password. La soluzione al problema è semplice: eliminare il file `/etc/ptmp`, dopo aver controllato che non contenga informazioni importanti (si tratta di una copia del file `/etc/master.passwd` che, per motivi imperscrutabili, è rimasta in circolazione anziché essere eliminata).

> **Nota:** l'esistenza del file `/etc/ptmp` può causare anche la visualizzazione di messaggi di avvertimento in fase di avvio. Per esempio:
>
> ```
> root: password file may be incorrect - /etc/ptmp exists
> ```

20.11. Come ricreare in dispositivi nella directory /dev

L'autore di questa sezione è Reinoud Koornstra

Per prima cosa è necessario passare alla modalità mono utente (single user) con le partizioni ancora montate "rw" (read-write). Per far ciò conviene semplicemente dare il comando **shutdown now** dalla modalità multi utente oppure riavviare il sistema con l'opzione `-s` e rendere `/` e `/dev` leggibili e scrivibili con i seguenti comandi:

```
# mount -u /
# mount -u /dev
```

Poi:

```
# mkdir /nudev
# cd /nudev
# cp /dev/MAKEDEV* .
# sh ./MAKEDEV all
# cd /
# mv dev odev
# mv nudev dev
# rm -r odev
```

In alternativa, se sul sistema sono installati i sorgenti in `/usr/src`, si può fare così:

```
# 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
```

Il valore da sostituire a $arch può essere determinato con il comando seguente:

```
# uname -m
```

oppure con:

```
# sysctl hw.machine_arch
```

> **Nota:** Il secondo metodo per ricreare i dispositivi (quello che sfrutta i sorgenti installati sul sistema) può aggiungere alcuni dispositivi a quelli già esistenti (almeno sull'architettura i386). Per esempio, attualmente è possibile avere 16 partizioni anziché 8. Utilizzando la versione originale di MAKEDEV (quella della directory `/dev`), i nuovi dispositivi non saranno creati.

Appendice A. Informazioni

A.1. Storia di questa guida

Questa guida è nata da una serie di annotazioni che dovevano più che altro servirmi da promemoria ed erano originariamente scritte in formato troff con le macro *ms*. Con il passare del tempo mi sono reso conto che potevano essere di aiuto anche a altri che volessero avvicinarsi a NetBSD e ho deciso di metterle sul mio sito dopo aver trovato un formato opportuno. Alla fine ho scelto di adottare *SGML/DocBook*.

Per la stesura di questa guida sono stati utilizzati i seguenti strumenti:

- l'editor vi fornito con NetBSD (nvi).
- il programma Jade che implementa le trasformazioni DSSSL. Jade genera direttamente i formati HTML e RTF.
- il sistema TeX dalla collezione dei package di NetBSD. TeX viene utilizzato come back-end per la generazione dei formati PS e PDF.
- il programma tgif per la generazione delle figure.
- i programmi gimp e xv per l'elaborazione e la conversione tra diversi formati di immagine.

Molti ringraziamenti a tutti gli sviluppatori di questi eccezionali strumenti di lavoro.

Appendice B. Installazione senza sysinst

L'autore di questa appendice è Wojciech Puchar

Questa appendice descrive il procedimento da utilizzare per installare NetBSD senza fare uso di sysinst. Ci si può chiedere perché non usarlo; ecco alcune ragioni:

- possibilità di usare opzioni non standard nella creazione dei file system.
- perché è facile!
- per avere il completo controllo del sistema e capire tutte le operazioni effettuate.
- per imparare a preparare i propri dischetti di installazione, mettendo altri utili strumenti accessori al posto di sysinst.

Questa appendice descrive solo la configurazione di sistemi i386 in cui NetBSD sarà l'unico sistema operativo sul disco fisso. Ho cercato tuttavia di evidenziare le parti specifiche di questa architettura sperando di ricevere qualche contributo per quanto riguarda gli altri port (da inviare, in inglese, a wojtek@3miasto.net).

Per mettere in pratica quanto descritto in quest'appendice è richiesta una conoscenza dei comandi di base di Unix (cd, ls, cp, tar, ecc.).

B.1. Partizionamento e installazione: teoria

Il processo di installazione consiste dei seguenti passi:

1. partizionamento del disco fisso
2. creazione del/dei filesystem
3. installazione del boot loader (dipendente dalla piattaforma e non presente in alcuni port, quali sparc, ecc.)
4. decompressione dei tarball nelle partizioni create
5. configurazione finale (creazione dei dispositivi `/dev`, `/etc/fstab`, impostazione del nome dell'host, ecc.)

B.1.1. Partizionamento

Gli schemi di partizionamento possono essere semplici o anche molto complessi; ai fini della presente appendice non interessa discutere quale sia il migliore schema possibile: ci si limiterà a descrivere uno schema classico per Unix, comprendente una piccola partizione root, una partizione di swap e una partizione `/usr` per tutto il resto.

In questa appendice, come in molti manuali di NetBSD, si parlerà spesso di "label" e di "disklabel". Le disklabel sono le strutture usate da NetBSD (e dai sistemi BSD in generale) per memorizzare le informazioni relative alle partizioni. Pertanto la creazione della disklabel equivale al partizionamento.

B.1.2. Dimensione della partizione di swap

Se il vostro di sistema ha memoria in sovrabbondanza e siete assolutamente certi che non avrete mai bisogno di swap, potete saltare questa sezione. Tuttavia, sui moderni dischi fissi di grandi dimensioni non è solitamente un problema lasciare una certa quantità di spazio per la partizione di swap, con il beneficio di poter avviare senza problemi un gran numero di programmi contemporaneamente.

Non esistono semplici formule per determinare la dimensione della partizione di swap. L'unica vera regola è che è meglio averne troppa che troppo poca, perché restare senza swap significa dover terminare non solo l'ultimo processo avviato ma anche tutti quelli che hanno bisogno di allocare memoria. Si può anche trattare di /sbin/init, il che porterà al collasso del sistema. Se avete uno di quegli enormi dischi fissi >10GB, laciate almeno 500MB per l'area di swap (ma anche di piu'). L'unica controindicazione è che l'uso dello swap comporta l'allocazione statica di 16 byte per ogni pagina i386 (4KB), il che significa usare 1MB di memoria per ogni 256MB di swap. Non è una cosa catastrofica, ma su una macchina con 16MB di memoria può diventare un problema.

Riassumendo: se avete un disco sufficientemente grande, allocate una generosa area di swap, altrimenti cercate di regolarvi in base alla memoria necessaria alle applicazioni che intendete utilizzare contemporaneamente.

B.1.3. Posizionamento della partizione di swap

Si può pensare che il posizionamento della partizione di swap sia ininfluente, il che è vero per quanto riguarda i dischi di vecchio tipo (per es. MFM). Sui dischi più recenti (intendendo dischi di oltre 80MB), però, sulle tracce esterne vengono scritti più dati rispetto a quelle interne a causa delle meggiori dimensioni. Ciò porta a un I/O più veloce sulle tracce esterne, che si trovano all'inizio del disco e quindi l'area di swap sarà più veloce se si trova nella zona iniziale.

B.1.4. Filesystems

Il filesystem standard usato da NetBSD è quello detto "ffs" o "ufs" (Fast File System o Unix File System). Anche se "fast" (veloce) non significa "il più veloce al mondo", si tratta di un file system dotato di buone prestazioni per uso generale e di meccanismi di protezione in caso di crash. Per creare il filesystem ci si può limitare a usare il comando **newfs** senza parametri, ma vale la pena di cercare di ottimizzarne l'uso. Vediamo ora una breve descrizione dei principali parametri su cui intervenire:

-m (percentuale)

> imposta la percentuale di spazio riservato. Il default di 10% è piuttosto abbondante e si può scendere al 5% senza particolari problemi ma è bene non scendere al di sotto di questo valore, perché riempiendo molto un filesystem ffs lo si rallenta.

-b (dimensione del blocco)

> imposta la dimensione in byte del blocco. Per i dischi di grosse dimensioni usare 8192 (8KB), mentre per quelli molto piccoli (<500MB) conviene usare 4096 per risparmiare spazio. Usando blocchi ancora più grandi non si velocizzano le operazioni e si può sprecare spazio, a meno di non memorizzare principalmente file di grosse dimensioni quali MP3 o filmati.

-f (dimensione del frammento)

> imposta la dimensione del frammento. Un blocco può essere suddiviso fino a un massimo di 8 frammenti e pertanto conviene impostare una dimensione pari a 1/8 della dimensione del blocco: 512 byte per blocchi da 4096 byte e 1024 byte per blocchi da 8192 byte. Il frammento è l'unità di allocazione minima per i file (come il cluster per Windows).

-i (byte di dati per inode)

> gli *inode* sono strutture usate per memorizzare informazioni sui file. Per ogni file ci deve essere almeno un inode (ma anche più di uno, per consentire al ffs di memorizzare i dati del file vicino al file stesso). La dimensione di default di 4096 è inadatta ai dischi di grosse dimensioni, perché consente di avere fino a 5 milioni di file su un dispositivo da 20GB. Una valutazione di massima consiglia l'uso di `-i 16384` per dischi >2GB e di `-i 32768` per dischi da 8GB. Usando valori più grandi non si risparmia più spazio perché un inode occupa 128 byte

Per esempio, le opzioni che uso per una partizione >10GB sono

```
# newfs -m 5 -b 8192 -f 1024 -i 32768 /dev/{nome_partizione}
```

La lettura della pagina di manuale di **newfs** è caldamente consigliata.

B.1.5. Installazione del bootloader

Durante il processo di boot il kernel (che di solito si chiama `/netbsd`) deve essere caricato in memoria dal disco. Il BIOS del PC (e il firmware delle altre architetture) non è di gestire astrazioni quali i filesystem, e quindi si limita a caricare in memoria il primo settore del disco e a eseguirne il codice. Questo codice determina la posizione del bootloader sul disco e lo carica. Il bootloader stesso, infine, può gestire il file system e carica il kernel. Poiché nel settore di boot devono venire memorizzate le coordinate fisiche del bootloader (`/boot`) non ci si può limitare a copiare il bootloader /boot nella partizione di destinazione, ma occorre dare il seguente comando:

```
# /usr/mdec/installboot -v /usr/mdec/biosboot.sym /dev/rwd0a
```

Per quanto riguarda la decompressione dei tarball e la configurazione finale del sistema non è necessaria una particolare introduzione teorica e quindi possiamo passare direttamente alla parte pratica.

B.2. L'installazione in pratica

Negli esempi seguenti supporremo che il disco fisso sia `wd0`, il CD `cd0` (se presente) e la scheda di rete `ep0` (se presente). Se i nomi reali sul vostro sistema sono diversi, basta sostituirli a quelli degli esempi.

B.2.1. Il boot

Per prima cosa bisogna fare il boot da floppy o da CD. Quando compare il menu di sysinst selezionare "x - exit to shell". Dare il comando **dmesg | more** per verificare i nomi dei dispositivi dell'hardware presente sul sistema. Per esempio, l'output seguente

```
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 che il disco “ST320423A” (ST di solito indica un disco Seagate) è collegato come master sul primo canale del controller IDE e ha 40011300 settori di 512 byte, il che porta a una dimensione di circa 19GB. Frequentemente la dimensione mostrata è minore (5-8%) rispetto a quella dichiarata dal produttore, perché i produttori considerano che 1GB equivalga a 1.000.000.000 byte anziché ai consueti 2^30 byte usati in ambito informatico. Poiché 2^30=1.073.741.824, si tratta di una differenza del 7,4%.

L’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 che la scheda ethernet viene identificata con `ep0`.

La maggior parte dei messaggi non è difficile da capire anche per i meno esperti, e la loro lettura è molto utile.

B.2.2. Inizializzazione del disco

Poiché vogliamo ottenere un sistema con solo NetBSD, possiamo eliminare tutte le partizioni preesistenti sul disco con il seguente comando:

```
# dd if=/dev/zero of=/dev/rwd0d bs=1m count=1
```

che “ripulisce” il primo megabyte del disco riempiendolo di zeri.

`/dev/rwd0d` significa: la partizione “raw” d del disco wd0 (su i386 d è una pseudo partizione che copre l’intero disco). “Raw” indica che non viene effettuato buffering dal sistema (qui non è necessario).

B.2.3. Creare le partizioni con disklabel

```
# disklabel -I -i wd0
```

Il comando precedente serve per iniziare a partizionare un disco vuoto. Non specificare l’opzione `-I` se si vuole modificare una disklabel esistente.

Verrà mostrato il prompt:

```
partition>
```

Il sistema è in attesa di comandi. I comandi più importanti sono **?** (help), **I** (imposta varie informazioni sul disco) e le lettere minuscole da **a** fino a **h** (impostano le informazioni per le rispettive partizioni).

Incominciamo con il comando **I**. Il sistema chiede il tipo di disco (accettare il default), il nome del disco (idem) e la label. La label è a piacere; per esempio si può scrivere il proprio nome. In caso di furto del disco fisso si può dimostrare di essere il proprietario perchè fdisk di DOS/Windows non cancella questa informazione.

Numero di partizioni: scegliere 8 (ma anche di più se ce n'è bisogno). Lasciare i valori di default per le altre opzioni, visto che le informazioni relative alla geometria dei dischi moderni sono in realtà fittizie.

Ora bisogna creare le partizioni, cominciando con le partizioni "d" e "c" che si riferiscono all'intero disco:

```
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:** sul port i386 la partizione "d" si riferisce all'intero disco e la "c" alla parte NetBSD del disco. Nel nostro esempio NetBSD copre l'intero disco e pertanto le partizioni "d" e "c" coincidono.

Poi si passa alla creazione della partizione da mettere all'inizio del disco. Nel partizionamento classico di Unix si tratta della partizione "a", di circa 50MB. Questa dimensione deve essere aumentata opportunamente se si intende tenere molta posta in spool o avere spazio in `/tmp` o `/home` senza creare partizioni separate per `/var` o `/home`.

```
partition> b
Filesystem type [?] [unused]: 4.2BSD
Start offset [0c, 0s, 0M]: 0s
Partition size ('$' for all remaining) [0c, 0s, 0M]: 100000s
```

100000 settori corrispondono a circa 50MB o poco meno.

Creiamo la seconda partizione (swap) di circa 585MB:

```
partition> a
Filesystem type [?] [unused]: swap
Start offset [0c, 0s, 0M]: 100000s
Partition size ('$' for all remaining) [0c, 0s, 0M]: 1200000s
```

e infine la partizione "e" per `/usr` fino alla fine del disco:

```
partition> e
Filesystem type [?] [unused]: 4.2BSD
Start offset [0c, 0s, 0M]: 1300000s
Partition size ('$' for all remaining) [0c, 0s, 0M]: $
```

Verificare che tutto sia a posto con il 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:** bisogna accertarsi che le partizioni non si sovrappongano (fatta eccezione per c e d). È molto importante ed è bene controllare due volte.

Infine dare i comandi:

```
partition> W
Label disk [n]? y
Label written
partition> Q
```

La disklabel è stata scritta. Congratulazioni, la parte più impegnativa dell'operazione è terminata

B.2.4. Creazione dei filesystem e installazione del bootloader

Per una piccola partizione root usare:

```
# newfs -m 5 -b 4096 -f 512 /dev/rwd0a
```

Per partizioni più grandi:

```
# newfs -m 5 -b 8192 -f 1024 -i 32768 /dev/rwd0<partition name>
```

È importante aver letto la teoria presentata nelle sezioni precedenti e non limitarsi a ricopiare i comandi.

Ora installiamo il bootloader:

```
# /usr/mdec/installboot -v /usr/mdec/biosboot.sym /dev/rwd0a
```

B.2.5. Decomprimere i set di installazione

Montare le partizioni su `/mnt`:

```
# mount -o async,noatime /dev/wd0a /mnt
# mkdir /mnt/usr
# mount -o async,noatime /dev/wd0e /mnt/usr
...
montare le altre partizioni create
...
```

Le opzioni `async` e `noatime` non sono obbligatorie: tendono a favorire la velocità delle operazioni a discapito dell'integrità del filesystem. La protezione dai crash di sistema in questo momento non è fondamentale, visto che si può ripulire tutto e riinstallare facilmente in caso di crash.

Le sezioni seguenti descrivono alcuni comuni metodi di installazione:

CDROM
FTP
NFS
nastro

B.2.5.1. Installazione da CDROM

```
# mount_cd9660 /dev/cd0a /mnt2
# cd /mnt2/dove_si_trovano_i_tarball
```

"dove_si_trovano_i_tarball" è solitamente `i386/binary/sets`. Usare i soliti comandi **cd** e **ls** per verificare, se non si è sicuri.

Copiare con **cp** i file desiderati su `/mnt/usr`. Come minimo bisogna installare `base.tgz`, `etc.tgz` e `kern.tgz`. Se il disco è grande copiare tutto (saltando i file `kern*.tgz` non utilizzati, visto che basta un solo kernel).

Ora eseguire **cd /** e **umount /mnt2** e, infine, rimuovere il CD.

B.2.5.2. Installazione da un server FTP

Attivare la rete. Per esempio, se l'indirizzo IP è 192.168.3.7 e la maschera di rete è 255.255.255.0 (24 bit), scrivere:

```
# ifconfig ex0 192.168.3.7/24 up
```

Se la scheda di rete non lo riconosce automaticamente può essere necessario indicare il tipo di "media" aggiungendo "media 10base2" o "media 10baseT" oppure "media 100baseT". Consultare la pagina di manuale di **ifconfig** per ulteriori informazioni.

Se il server FTP non si trova sulla rete locale, bisogna aggiungere una "route" di default ooportuna. Per esempio:

```
# route add default 192.168.3.1
```

poi raggiungere con **ftp** il server contenente i file di NetBSD. Per il momento si possono solo usare indirizzi IP, e non nomi.

```
# cd /mnt/usr
# ftp 192.168.3.13
```

> **Nota:** se proprio si vogliono usare i nomi, si può definire un name server (per esempio 192.168.3.2) in questo modo:
>
> ```
> # echo "nameserver 192.168.3.2" >
> /etc/resolv.conf
> ```
>
> e poi dare il comando:
>
> ```
> # ftp ftp.netbsd.org
> ```

Fare il login, portarsi nella directory dove si trovano i file della distribuzione e scaricarli tutti con il comando **get**. Infine uscire da ftp.

B.2.5.3. Installazione da NFS

Attivare la rete come già spiegato. Poi:

```
# mount IP-del-server-NFS:/directory /mnt2
```

per esempio:

```
# mount 192.168.3.13:/public/ /mnt2
```

Poi procedere come per l'installazione da CD. Per esempio:

```
# cd /mnt2/NetBSD-1.5.2/i386/binary/sets
# ls
```

e copiare tutti i file necessari. Poi:

```
# cd /
# umount /mnt2
```

B.2.5.4. Installazione da nastro

Anche se i desktop solitamente non hanno unità a nastri, i server spesso ne hanno una (e talvolta anche un lettore di CD, tanto per occupare un po' di spazio). L'installazione da nastro è molto semplice anche se, purtroppo, i PC non possono fare il boot da nastro e quindi bisogna utilizzare i floppy.

Per preparare un nastro per l'installazione bisogna avere a disposizione un'altra macchina con un'unità a nastri.

Riavvolgere il nastro:

```
# mt rewind
```

impostare una dimensione di blocco senza limitazioni per i nastri SCSI

```
# mt setblk 0
```

Questo cambia la piccola dimensione di default del blocco a una dimensione qualsiasi. Usando blocchi di grandi dimensioni si possono velocizzare le operazioni, soprattutto sui drive DAT/DDS.

È meglio disabilitare la compressione perché alcuni lettori non sono in grado di leggere un flusso compresso. D'altronde comprimendo dei file già compressi non si hanno benefici, anzi si potrebbe perfino ottenere un peggioramento (fino al 15% sui dischi HP DDS!):

```
# mt compression 0
```

Memorizzare i file della distribuzione:

```
# cd /dove_si_trovano_i_tarball
# tar -b64 -cvf /dev/nrst0 *.tgz
```

Quest'ultimo comando scrive tutti i file tgz su nastro con una dimensione di blocco di 64*512 byte (32KB, una dimensione di blocco ben testata per tutti i DAT).

Avendo usato il dispositivo `/dev/nrst0` (prima unità a nastri SCSI, senza riavvolgimento) si può ora registrare anche qualche altro archivio, per esempio qualche utile package di base (per es. editor).

```
# cd /dove_sono_i_package/All
# tar -b64 -cvf /dev/nrst0 joe*.tgz screen*.tgz
```

Con il comando precedente sono stati aggiunti l'editor joe e il programma screen. È meglio non aggiungere troppi package in questo file.

Ora si possono aggiungere altri package da installare dopo che il sistema è entrato in funzione in modalità multiutente.

```
# tar -b64 -cvf /dev/nrst0 package_vari
```

e anche, volendo,

```
# tar -b64 -cvf /dev/nrst0 *.tgz
```

Ora si può riavvolgere ed estrarre il nastro:

```
# mt rewoffl
```

Il nastro per l'installazione è pronto! Poiché i lettori DDS sono compatibili all'indietro, può essere conveniente acquistare un lettore DDS-1 o DDS-2 usato economico per il computer di casa. Così sarà possibile preparare nastri di installazione e, ovviamente, fare il backup del computer: un vantaggio non da poco, che a mio avviso vale sicuramente la spesa.

B.2.5.4.1. Copiare da nastro

Caricare il nastro preparato e dare i comandi:

```
# mt rewind
# mt setblk 0
# cd /mnt/usr
# tar -b64 -xvf /dev/nrst0
```

B.2.6. Decomprimere i set della distribuzione

Non è una cattiva idea copiare qualche altro package extra, in particolare un editor (come joe o jed) per chi non si trova a proprio agio con vi. L'importante è copiare i package su `/mnt` e non su `/mnt/usr`.

Naturalmente per trasferire i file si può scegliere uno qualunque tra i metodi esposti nelle sezioni precedenti oopure anche utilizzare dei dischetti.

Decomprimere i tarball tutti insieme:

```
# cd /mnt
# for x in usr/*.tgz;do tar xzpvf $x;done
# rm usr/*.tgz
```

o, se si preferisce, uno alla volta:

```
# tar xzpvf /usr/base.tgz
```

Se si è creata una sola grande partizione per `/usr` (e magari una piccola root) può essere una buona idea spostarci dentro `/tmp`, `/var` e `/home` creando dei link simbolici:

```
# cd /mnt
# tar cf - var home tmp|(cd usr; tar xpvf -)
# rm -rf var home tmp
# ln -s usr/var .
# ln -s usr/home .
# ln -s usr/tmp .
```

Questa sequenza di comandi sposta `/var`, `/home` e `/tmp` dentro a `/usr` e crea i link necessari.

Ancora non è venuto il momento di installare i package aggiuntivi!

B.2.7. Completare l'installazione

Il comando seguente crea i nodi dei dispositivi in `/dev` (su un 486 può richiedere un po' di tempo):

```
# cd /mnt/dev
# ./MAKEDEV all
```

Poi **cd /mnt/etc** e creare `fstab`. Chi ha dei problemi con vi può usare il comando:

```
# 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)
```

Se si vuole usare kernfs, creare la directory `/kern` e aggiungere una riga al file `/etc/fstab`:

```
# mkdir /mnt/kern
# echo "/kern /kern kernfs rw" >> fstab
```

Infine verificare che tutto è a posto:

```
# cat fstab
```

Rimuovere CD/dischetti/ecc., scrivere **reboot** e attendere che il sistema si riavvii in modalità monoutente.

Al prompt della shell, scrivere:

```
# /sbin/mount -a
```

per montare tutti i filesystem “read-write”. Chi ha copiato dei package aggiuntivi (editor, ecc.) li può ora installare con il comando **pkg_add**. Poi si possono rimuovere i tarball con **rm**.

Adesso bisogna editare `/etc/rc.conf`, `/etc/defaults/rc.conf` e gli altri file di configurazione. Leggere il documento di installazione INSTALL per i dettagli. Generalmente, per far funzionare il sistema è sufficiente mettere

```
rc_configured=YES
```

in `/etc/rc.conf` e impostare il nome dell’host in `/etc/defaults/rc.conf`.

Con ^D (Control-D) si passa alla modalità multiutente. Il sistema è installato e funzionante.

Appendice C. Come contribuire alla guida a NetBSD

Ultimamente si può notare un notevole aumento dell'interesse verso la documentazione, introduttiva e avanzata, di NetBSD: si tratta indubbiamente di un segno della crescente popolarità del sistema e dell'aumento della base di utenti. Pertanto, è più che mai importante continuare a aggiornare e a accrescere il materiale sia di questa guida sia dei vari HOWTO presenti sul sito di NetBSD.

Indipendentemente dal proprio livello di conoscenza e di esperienza con NetBSD, chiunque può contribuire allo sviluppo di questa guida. Questa appendice spiega in che modo si può contribuire alla guida e cosa bisogna sapere prima di incominciare.

I principianti e, più in generale, coloro che installano il sistema per la prima volta, possono inviare i propri commenti e suggerimenti direttamente a me indicando, per esempio, se qualcosa non ha funzionato, se un argomento non è spiegato in modo sufficientemente chiaro, ecc. Oppure si può anche inviare una richiesta perché sia trattato un determinato argomento. Questo tipo di riscontro è molto utile.

Gli utenti più avanzati possono scrivere una nuova sezione o anche un intero capitolo da aggiungere alla guida. Oppure possono modificare una parte già esistente, aggiungendo esempi, figure o riformulandola in maniera più chiara.

Chi ha un po' di tempo libero può prendere in considerazione l'idea di tradurre la guida in una nuova lingua.

Qualunque cosa scegliate di fare, per prima cosa contattatemi, onde evitare lavoro duplicato o inutile.

C.1. Tradurre la guida

Chi vuole tradurre la guida deve per prima cosa mettersi in contatto con me oppure scrivere alla mailing list netbsd-docs@netbsd.org.

Se altre persone stanno già lavorando a una traduzione della guida nella stessa lingua è possibile aggiungersi al gruppo, abbreviando i tempi di completamento del lavoro. Se nessun altro è al lavoro, è sempre possibile che alcuni capitoli siano già stati tradotti in passato, altrimenti bisogna iniziare una traduzione ex-novo. Qualunque situazione si presenti, non è necessario tradurre l'intera guida: anche la traduzione di un solo capitolo è un contributo utile e apprezzato.

Anche quando una traduzione è già stata completata, è sempre necessario mantenere aggiornata la guida con il nuovo materiale e con le correzioni a quello esistente, ed è quindi possibile diventare il *mantainer* di una versione della guida.

C.1.1. Cosa serve per iniziare una traduzione?

Per iniziare una traduzione della guida basta avere i sorgenti della stessa. Mettetevi in contatto con me e vi invierò la versione più recente.

Più in dettaglio, quel che serve è:

- sorgenti della guida (in formato SGML/DocBook)
- un editor di testo, quale vi o emacs.

Il lavoro di traduzione può essere svolto sotto qualsiasi sistema operativo.

> **Nota:** non iniziate a lavorare in HTML o altri formati: sarà molto oneroso convertire il lavoro in formato SGML/DocBook.

C.1.2. Come scrivere in formato SGML/DocBook

Per tradurre la guida non è necessario *imparare* a scrivere in SGML/DocBook: è sufficiente lavorare con un editor sui sorgenti sostituendo il testo originale e lasciando invariati i *tag*, cioè riutilizzando la struttura del documento. A titolo di esempio, per tradurre la precedente nota si potrebbe operare in questo modo:

1. caricare i sorgenti del capitolo corrente `ap-contrib.sgml`, nell'editor.
2. trovare il testo corrispondente alla nota precedente, che ha il questo aspetto:

```
<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. aggiungere la traduzione all'interno dei tag, dopo la versione italiana (o prima, come si preferisce). Ora il testo ha questo aspetto:

```
<note>
  <para>
    non iniziate a lavorare in HTML o altri formati:
    sar&agrave; molto oneroso convertire il lavoro in
    formato SGML/DocBook.
    your translation goes here
    your translation goes here
    your translation goes here
  </para>
</note>
```

4. eliminare le tre linee di testo originale italiano lasciando la traduzione.

```
<note>
  <para>
    your translation goes here
    your translation goes here
    your translation goes here
  </para>
</note>
```

Quando si scrive la traduzione è preferibile utilizzare la stessa indentazione e la stessa formattazione della versione originale. Vedere la Sezione C.3 per un esempio.

Un problema che si deve quasi sempre affrontare quando si scrive in SGML/DocBook è quello dei "caratteri nazionali" (per esempio, le lettere accentate come "è"). Anche se questo tipo di caratteri si può utilizzare in un documento, è preferibile sostituirli con le corrispondenti *entità* SGML. Per esempio, il carattere "è" si scrive "è". È evidente che ciò rende più difficile la scrittura e, soprattutto, la lettura del testo ma, se non altro, il problema della scrittura può essere alleviato dall'uso di un editor che supporti le macro. Vi ed emacs, due editor tra i più noti, entrambi offrono la possibilità di associare macro ai tasti, generando in automatico le entità richieste. Per esempio, per ottenere automaticamente la e accentata precedente si può inserire il seguente comando nel file `.exrc`:

```
map! è &egrave;
```

L'Appendice D spiega come installare gli strumenti software necessari per generare il formato HTML e gli altri formati a partire dai sorgenti DocBook. Il formato HTML è il più semplice da generare ed è utile per verificare la correttezza dei sorgenti (cioè che non vi sia qualche errore nei tag) nonché per una lettura più agevole del testo. Naturalmente, come già detto, la generazione del formato HTML non è assolutamente necessaria per effettuare la traduzione: quello che conta sono solo i sorgenti tradotti.

C.2. Inviare materiale per la guida

Chi vuole contribuire alla guida inviando materiale ha diverse possibilità, a seconda della quantità di testo che vuole scrivere. Se si tratta solo di una breve correzione o aggiunta, la cosa più semplice è inviare un'e-mail a me con il testo e provvederò a inserirlo nella guida.

Se si desidera scrivere una certa quantità di testo, come una sezione o un intero capitolo, si può scegliere tra più formati:

- SGML/DocBook; questo è il formato preferito. Se si sceglie questo formato bisogna procurarsi i sorgenti della guida e usarli come schema per l'indentazione e il formato del testo e dei tag.
- testo; se la formattazione del testo è semplice, la conversione al formato SGML non è particolarmente laboriosa.
- HTML; è preferibile scriverlo a mano (cioè con un normale editor di testo) anziché utilizzare un generatore di HTML perché la conversione a SGML risulta più semplice. L'uso di HTML rispetto al puro testo non porta particolari vantaggi, se non quello di rendere più evidente il formato che si desidera dare al contenuto. Un altro beneficio collaterale è che le entità HTML per i caratteri accentati, ecc., saranno probabilmente già presenti nel testo.
- vengono accettati anche altri formati, ma solo nei casi in cui non sia possibile utilizzare nessuno di quelli precedenti.

C.3. Schema di disposizione SGML/DocBook

Per i sorgenti della guida viene usato uno stile di formattazione simile a quello di un programma. Eccone uno schema:

```
<chapter id="chap-xxxxx">
  <title>Questo &egrave; il titolo di un capitolo</title>

  <para>
    Questo &egrave; il testo.  Questo &egrave; il testo.
    Questo &egrave; il testo.  Questo &egrave; il testo.
    Questo &egrave; il testo.
  </para>

  <!-- ============================================================ -->

  <sect1>
    <title>Titolo di una sect1</title>

    <para>
      Questo &egrave; il testo.  Questo &egrave; il testo.
      Questo &egrave; il testo.  Questo &egrave; il testo.
      Questo &egrave; il testo.
    </para>

    <!-- .......................................................... -->

    <sect2>
      <title>Titolo di una sect2</title>

      <para>
 Una sect2 &egrave; annidata dentro a una sect1.
      </para>
    </sect2>

  </sect1>

  <!-- ============================================================ -->

  <sect1>
    <title>Titolo di una sect1</title>

    <para>
      Questa &egrave; una lista:
      <itemizedlist>
 <listitem>
   <para>
     testo
   </para>
 </listitem>
 <listitem>
   <para>
     testo
```

```
    </para>
  </listitem>
      </itemizedlist>
    </para>

  </sect1>
</chapter>
```

I default sono:

- due spazi per ogni livello di indentazione
- linee non più lunghe di 72 caratteri
- usare linee di separazione (commenti) tra sect1/sect2

Appendice D. Introduzione a SGML/DocBook

Contrariamente a quello che si potrebbe pensare leggendone il titolo, questo capitolo descrive l'installazione e la configurazione degli strumenti necessari per produrre una versione formattata della presente Guida a NetBSD. Anche se SGML/DocBooks e i DSSSL non vengono qui descritti, alla fine del capitolo sono presenti alcuni link a materiale informativo essenziale per chi vuole saperne di più.

L'ambiente SGML/DocBook può essere installato utilizzando il "meta-pacchetto" *netbsd-docs*; questo è il metodo più semplice ed è quello che consiglio di usare. Questo capitolo, invece, descrive l'installazione e la configurazione dei componenti necessari uno a uno, al fine di eseminarli più in dettaglio e, eventualmente, di verificarne la corretta installazione.

> **Nota:** È possibile che il "meta-pacchetto" *netbsd-docs* installi qualche pacchetto in più rispetto a quelli qui descritti. Si tratta di pacchetti utili ma non strettamente necessari ai fini della produzione della guida.

I pacchetti necessari per la guida verranno installati usando package precompilati anche se, se si preferisce, è possibile compilarli partendo dai sorgenti. Per ulteriori dettagli si veda Capitolo 8.

> **Nota:** i numeri di versione dei pacchetti installati sono naturalmente soggetti a cambimento, con l'uscita di nuove versioni.

D.1. Che cos'è SGML/DocBook?

SGML (Standard Generalized Markup Language) è un linguaggio utilizzato per definire la struttura di altri linguaggi basati su "markup". Ciò significa che con SGML si può definire la *grammatica* dei linguaggi di markup. HTML, per esempio, può essere definito usando SGML. I programmatori possono pensare a SGML come all'equivalente della BNF (Backus-Naur Form) per i linguaggi di programmazione: una notazione per definire grammatiche.

DocBook è uno schema di markup definito usando SGML; DocBook specifica quali tag possono essere utilizzati in un documento e in che modo possono essere combinati tra loro. I programmatori possono pensare a DocBook come a un linguaggio di programmazione, con le sue parole chiave valide e le sue regole. Per esempio, per DocBook i tag

```
<para> ... </para>
```

definiscono l'inizio e la fine di un paragrafo. Inoltre DocBook specifica che il tag <para> può trovarsi all'interno di una sezione <sect1> ma che il tag <sect1> non può trovarsi all'interno di un <para>.

Alla luce di quanto appena detto, si può capire che i documenti prodotti con questi strumenti sono documenti DocBook, e non SGML; da questo punto di vista, DocBook è la controparte di HTML (benché i tag siano più numerosi e i concetti alla base dei due linguaggi diversi).

La specifica DocBook (cioè la lista dei tag validi e le regole per combinarli) viene chiamata DTD (Document Type Definition.)

Un DTD definisce quello che è l'aspetto di un documento *sorgente* valido ma non dà alcuna indicazione sul formato del documento finale generato a partire dai sorgenti. A titolo di esempio, il tag <title> marca l'inizio di un titolo ma non specifica che aspetto avrà il titolo nel documento finale (dimensione e corpo dei caratteri, allineamento, spaziatura, ecc.) Per arrivare alla "presentazione" del documento è necessario un ulteriore passo, effettuato utilizzando il programma Jade, che applica al documento sorgente le trasformazioni DSSSL. DSSSL (Document Style Semantics and Specification Language) è il formato utilizzato per definire gli *stylesheets* (fogli di stile) necessari per convertire il documento DocBook in altri formati.

La "vita" di un documento DocBook è pertanto la seguente:

- scrittura del documento sorgente
- il DTD di DocBook viene utilizzato da nsgmls per "validare" il documento
- Jade applica i fogli di stile DSSSL al documento sorgente e genera un nuovo documento.

  Non è ancora possibile stampare il documento generato, perché contiene le direttive di formattazione indicate sui fogli di stile (si può trattare di un documento HTML, RTF, TeX, ecc.)

- un programma di formattazione (un browser per HTML, un word processor per RTF, TeX, ...) crea la versione finale, visualizzabile o stampabile, del documento

In definitiva, ciò che serve per iniziare a lavorare è:

- un DTD per DocBook
- i fogli di stile DSSSL usati da Jade, per generare HTML/RTF, ecc.
- il programma Jade e il parser nsgmls

D.2. Jade

Jade è un parser SGML/XML che implementa il meccanismo dei DSSSL. Il pacchetto Jade include anche il parser nsgmls, che valida i documenti.

> **Nota:** OpenJade è un'evoluzione di Jade che, al momento, non si compila sotto NetBSD (ma che non è necessaria per la guida).

Per installare il package precompilato di Jade:

```
# pkg_add jade-1.2.1.tgz
```

Al termine dell'installazione, la documentazione di Jade si trova in `/usr/pkg/share/doc/jade/index.htm`. La directory principale dell'installazione è `/usr/pkg/share/sgml/jade/`: è qui che si trova il file `catalog` di Jade.

D.3. DocBook

Dopo aver installato Jade si può procedere all'installazione del DocBook DTD.

Questo pacchetto richiede l'installazione delle "entity" per il set di caratteri ISO 8879:1986. Aggiungiamo quindi il relativo package:

```
# pkg_add iso8879-1986.tgz
```

Le entity vengono installate nella directory `/usr/pkg/share/sgml/iso8879/` e il relativo catalogo è `/usr/pkg/share/sgml/iso8879/catalog`.

Finalmente si può installare il DTD.

```
# pkg_add docbook-4.1.tgz
```

A dispetto del nome questo package installa varie versioni di DTD (2.4.1, 3.0, 3.1, 4.0, 4.1). Ciò consente di elaborare documenti che usano differenti versioni di DocBook.

> **Nota:** la guida di NetBSD al momento usa la versione 4.1 del DTD di DocBook e, pertanto, le altre versioni non sono strettamente necessarie. Tuttavia, in considerazione del fatto che ogni DTD occupa meno di 250 KB può convenire mantenerle sul disco fisso poiché consentono di elaborare altri documenti.

La radice dell'installazione del sistema è `/usr/pkg/share/sgml/docbook/4.1/`. Ogni versione del DTD si trova in una directory separata e possiede un proprio file catalogo. Per esempio: `/usr/pkg/share/sgml/docbook/4.1/catalog`.

D.4. I fogli di stile DSSSL

L'ultimo passo consiste nell'installazione dei fogli di stile DSSSL.

```
# pkg_add dsssl-docbook-modular-1.57.tgz
```

Anche i fogli di stile installano un proprio catalogo, che si trova in `/usr/pkg/share/sgml/docbook/dsssl/modular/catalog`. Inoltre sono corredati da una dettagliata documentazione: `/usr/pkg/share/sgml/docbook/dsssl/modular/doc/index.html`.

D.5. Usare gli strumenti

Adesso che è stato installato tutto il necessario, si può incominciare a utilizzare gli strumenti per produrre la versione HTML della guida (per esempio la versione italiana).

Con un **cd** ci si porta nella directory radice della guida e si dà il comando:

```
$ cd it
$ make netbsd.html
```

Un lungo elenco di errori scorrerà sullo schermo; ciò è dovuto al fatto che il parser SGML nsgmls non riesce a trovare i file `catalog`. Pertanto occorre dare i seguenti comandi (che conviene inserire nel proprio `~/.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: modificare opportunamente i comandi se si usa la shell C o una shell derivata da questa.

Ora che la variabile d'ambiente SGML_CATALOG_FILES è attiva, si può ritentare il 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
```

Questa volta l'operazione va a buon fine e viene generata la versione HTML della guida. La creazione del formato RTF non presenta maggiori difficoltà:

```
 $ 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
```

Con il setup appena descritto si possono generare solo i formati HTML e RTF: per creare i formati PS e PDF occorre installare e configurare TeX e Jadetex (un formato per TeX).

D.6. Un approccio alternativo ai file di catalogo

Nelle mie installazioni generalmente creo un file di catalogo *master*, che contiene i riferimenti a tutti gli altri. Per utilizzare questo metodo, si può creare il file `/usr/pkg/share/sgml/catalog` con il seguente contenuto:

```
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"
```

Fatto ciò, le inizializzazioni nel ~/.profile si possono semplificare così:

```
SGML_CATALOG_FILES=/usr/pkg/share/sgml/catalog
export SGML_CATALOG_FILES
```

D.7. Generare la versione Postscript

Per creare la versione Postscript della guida sono necessarie alcune operazioni preliminari:

- installazione di TeX
- abilitazione della silabazione italiana di TeX
- creazione del formato hugelatex richiesto da jadetex
- installazione di jadetex

Le sezioni seguenti descrivono in dettaglio i punti precedenti.

D.7.1. Installazione di TeX

L'installazione di TeX è resa semplice, come di consueto, dal sistema dei package. Utilizzando i pacchetti precompilati è sufficiente dare i seguenti comandi (i numeri di versione dei pacchetti potrebbero essere differenti):

```
# pkg_add teTeX-share-1.0.2.tgz
# pkg_add teTeX-bin-1.0.7nb1.tgz
```

D.7.2. Abilitare la sillabazione italiana di TeX

La guida a NetBSD è correntemente disponibile in quattro diverse lingue: francese, inglese, italiano e portoghese. Al termine dell'installazione di TeX, la sillabazione per le lingue italiana portoghese non è abilitata; per attivarla sono necessarie alcune semplici operazioni:

Editare `/usr/pkg/share/texmf/tex/generic/config/language.dat` rimuovento il carattere di commento (%) all'inizio della riga relativa alle lingue italiana e portoghese. Cioè:

```
%italian ithyph.tex
%portuges pt8hyph.tex
```

diventa

```
italian ithyph.tex
portuges pt8hyph.tex
```

> **Nota:** quando saranno disponibili ulteriori traduzioni della guida, sarà probabilmente necessario attivare la sillabazione anche per le nuove lingue.

Ora bisogna ricreare i formati latex e pdflatex:

```
# cd /usr/pkg/share/texmf/web2c
# fmtutil --byfmt latex
# fmtutil --byfmt pdflatex
```

A scopo di verifica si può controllare, per esempio, il file `latex.log`: dovrebbe contenere (tra le altre informazioni) una riga di questo tipo:

```
Babel <v3.6Z> and hyphenation patterns for american, french, german,
ngerman, italian, nohyphenation, loaded.
```

> **Nota:** vi sono vari modi di eseguire queste operazioni, a seconda del proprio livello di conoscenza di TeX (il mio è piuttosto basso). Per esempio, è possibile usare il programma interattivo texconfig, oppure ricreare i formati usando direttamente **tex** (fmtutil, infatti, è semplicemente uno script che richiama tex).
>
> Se conoscete un metodo migliore per effettuare le operazioni descritte un questo capitolo, vi prego di segnalarmelo.

D.7.3. Creazione del formato hugelatex

Per poter utilizzare jadetex è necessario creare il formato hugelatex, che altro non è se non un formato latex con aumentati limiti di capacità per riuscire a gestire i grossi file creati da jadetex. Prima di effettuare le modifiche conviene fare una copia di backup del file `/usr/pkg/share/texmf/web2c/texmf.cnf`. Poi aggiungere alla fine del file le seguenti righe (le definizioni per jadetex e pdfjadetex serviranno in seguito ma tanto vale aggiungerle già adesso):

```
% 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
```

Aggiungere le righe seguenti, che descrivono il formato hugelatex, al file `fmtutil.cnf` (nella directory `/usr/pkg/share/texmf/web2c`)

```
# hugelatex format created for jadetex
hugelatex tex  language.dat latex.ini
```

salvare il file e lanciare il comando

```
fmtutil --byfmt hugelatex.
```

Il formato hugelatex è ora pronto per l'uso.

> **Nota:** come già segnalato, con TeX vi sono più modi di raggiungere lo stesso risultato (in questo caso la creazione del formato hugelatex).
>
> La guida all'installazione di jadetex riporta le seguenti istruzioni:
>
> ```
> # 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. Installazione di Jadetex

> **Nota:** la versione più recente di Jadetex può essere scaricata da http://sourceforge.net/projects/jadetex/

Scaricare il package Jadetex (la versione corrente è `jadetex-3.12.zip`), scompattarlo e poi dare i seguenti comandi:

```
# cd jadetex
```

```
# make install
# mktexlsr
# cd /usr/pkg/bin
# ln -s tex jadetex
```

L'installazione copia i file di formato di jadetex e pdfjadetex (nonché alcuni file di supporto) nell'albero delle directory di TeX.

La distribuzione di Jadetex contiene anche due pagine di manuale che non vengono installate in automatico. Se lo si desidera è sufficiente copiarle a mano nella directory opportuna. Per esempio:

```
# cp jadetex.1 pdfjadetex.1 /usr/local/man/man1
```

Ora tutto è pronto per creare la versione Postscript della guida a NetBSD (e naturalmente di qualunque altro documento). Per esempio, portarsi nella directory base della guida e dare i seguenti comandi:

```
$ cd it
# make netbsd.ps
```

D.8. Collegamenti utili

Una semplice e ben scritta introduzione a SGML/DocBook, nonché una descrizione degli strumenti necessari si trova in 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/) è la fonte ufficiale di informazioni su DocBook. Trs le altre informazioni si pò consultare in linea o scaricare il libro DocBook: The Definitive Guide (http://www.oasis-open.org/docbook/documentation/reference/) di Norman Walsh and Leonard Muellner, una guida fondamentale per chi vuole approfondire l'argomento.

Per informazioni sulle DSSSL conviene incominciare da nwalsh.com (http://nwalsh.com).

I sorgenti di Jade/OpenJade e la relativa documentazione si può trovare sulla OpenJade Home Page (http://openjade.sourceforge.net/).

A chi desidera produrre documenti in formato Postscript o PDF a partire dai sorgenti DocBook conviene dare un'occhiata alla home page di JadeTex (http://www.tug.org/applications/jadetex).

La home page of Markus Hoenicka (http://ourworld.compuserve.com/homepages/hoenicka_markus/) spiega tutto ciò che è necessario sapere per lavorare con SGML/DocBook su Windows NT.

Appendice E. Ringraziamenti

Varie persone hanno contribuito alla realizzazione di questa guida sottoponendo materiale, rivedendo i contenuti, dando suggerimenti, correggendo errori e anche aiutando a configurare e a installare i programmi utilizzati per la sua realizzazione.

In particolare desidero ringraziare:

- Paulo Aukar
- Manolo De Santis
- Eric Delcamp
- Hubert Feyrer, che ha scritto l'introduzione al networking del la Sezione 9.1 compresa la parte relativa al protocollo di nuova generazione IPv6 e le sezioni sulla connettività IPv6 la Sezione 10.2.4.
- Jason R. Fink
- Reinoud Koornstra (Mipam)
- David Magda
- Guillain Seuillot
- Wojciech Puchar
- Oliver Tonnhofer

Se per caso mi sono dimenticato di aggiungere il vostro nome alla lista precedente vi prego di segnalarmelo.