NetBSD

Inhalt:

Zu NetBSD Intro und Ausstieg

Chapter 1. Licence

Anmerkung des Übersetzers: Der Inhalt dieser Datei besagt nichts anderes, a
Inhalte des Guides keinerlei Verantwortung übernommen wird, egal, ob es sic
Handbuch handelt, um Resultate irgendwelcher Versuche, Fehler in irgendeine
Software oder Ähnliches.

Inhalt des Shortguides

Kapitel 2: Was ist NetBSD

Inhalt:

NetBSD ist ein freies, portierbares Unix- Betriebssystem, das es für eine ganze Reihe von Plattformen gibt. Das fängt beim 64Bit-Alpha-Server an und hört bei Handheld- Geräten auf. Durch das saubere Design eignet es sich hervorragend sowohl für Produktions- als auch für Forschungsumgebungen. Ausserdem unterstützt es die Benutzer mit den kompletten Sourcen. Es gibt viele Applikationen dafür, die einfach zu bekommen sind.

nach oben

2.1. Die Geschichte von NetBSD

Die erste Version von NetBSD lässt sich auf 1993 zurückdatieren. Es ist aus dem 4.3BSD-Lite-Betriebssystem entstanden; einer Unixversion, die an der University of California in Berkeley entwickelt worden ist (BSD= Berkeley Software Distribution) und aus dem 386BSD- System; das die erste BSD- Portierung für Intels 368er-CPU gewesen ist. In den folgenden Jahren wurden die Modifikatiionen der 4.4BSD-Lite- Ausgabe (Die letzte Version von der Berkeley-Gruppe) in das System integriert. Der BSD- Zweig von Unix war sehr wichtig und hatte grossen Einfluss auf die Geschichte dieses Betriebssystems, zu dem es viele Tools, Ideen und Erweiterungen geliefert hat (Der vi- Editor, die C- Shell, Job Control, das Berkeley-Fast-File-Dateisystem, verlässliche Signale, Unterstützung für virtuellen Speicher, die TCP/IP- Implementation, um hier nur mal einige wenige zu nennen), die heute in allen Unix- Systemen Standard sind. Diese Tradition der Forschung und Entwicklung überlebte bis heute in den BSD- Systemen (freie und kommerzielle) und, ganz besonders, in NetBSD.

nach oben

2.2. NetBSD- Features

NetBSD funktioniert auf sehr vielen Hardwareplattformen und ist sehr gut portierbar. Möglicherweise ist es das Bertriebssystem, das auf den meisten Plattformen läuft. Der gesamte Source des NetBSD-Kernels und der Userland- Programme für alle unterstützten Plattformen liegen offen; Bitte für weitere Informationen auf www.netbsd.org nachschauen.

Eine genaue Liste mit den Features von NetBSD findet sich unter http://www.netbsd.org/Misc/features .

Die grundlegenden NetBSD- Features sind:

- Portierbarkeit (Zur Zeit läuft NetBSD auf 40 Plattformen)

- Codequalität und -Korrektheit
- Einhaltung der Normen
- Forschung und Innovation

Die beschriebenen Charakteristika bringen auch indirekte Vorteile. Wenn Du beispielsweise nur auf einer Plattform arbeitest, kann es sein, dass Du an der Portierbarkeit nicht interessiert bist. Aber die Portierbarkeit ist an die Codequalität gebunden: Ohne eine sauber geschriebene und einer gut organisierte Codebasis wäre es unmöglich, diese vielen Plattformen zu unterstützen. Ausserdem ist die Codequalität die Basis jedes guten und stabilen Softwaresystems. Es ist manchmal sehr überraschend, dass nur wenige Leute das zu verstehen scheinen. Die Beachtung von Qualitäts- und Architekturregeln wird mit dem grossen Potential belohnt, das im Code von NetBSD und der Qualität seiner Treiber steckt.

Ein weiterer der verschiedenen Charakterzüge von NetBSD ist, dass wir mit teilweisen Implementationen nicht zufrieden sind. Manche System leben von der Philosophie »Es funktioniert, also ist es richtig«. In diesem Licht kann NetBSD als »Es funktioniert nicht, bis es richtig ist« beschrieben werden. Denke mal darüber nach, wie viele überladene Programme heutzutage traurigerweise unter ihrem Gewicht und ihren Features kollabieren, und Du wirst verstehen, warum NetBSD diese Situation unter allen Umständen vermeiden will.

nach oben

2.3. Unterstützte Plattformen:

NetBSD 1.5.2 unterstützt unter anderem folgende Plattformen (Die technischen Details aller Plattformen finden sich auf der NetBSD- Website):

- 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
- i386er IBM PCs und Nachbauten
- Apple Macintosh
- Apple Power Macintosh
- Motorola MVME 68k SBCs
- NeXT 68k 'black' hardware
- PC532
- Digital MIPS-basierte DECstations und DECsystems

- Sun SPARC
- Sun 3 und Sun3x
- Digital VAX

nach oben

2.4. NetBSD's Zielgruppe

Die NetBSD- Site sagt das: »Das NetBSD- Projekt bietet ein frei zu beschaffendes System an, das frei weitergegeben werden kann. Professionelle, Lobbyisten und Forscher für können es für alles verwenden, was ihnen einfällt. Ich möchte hinzufügen, dass NetBSD ein ideales System ist, um Unix zu lernen; hauptsächlich, weil es sich an den Normen orientiert (eines der Projektziele) und weil es auf Hardware läuft, die bei anderen Betriebsystemen längst als veraltet bezeichnet wird. Wir können sagen: » Lerne Unix, benutze Unix; Du brauchst keine neue, teure Hardware dazu: Du kannst Deinen alten Mac oder PC, der auf dem Dachboden verstaubt, dazu verwenden. Ausserdem: Wenn Du ein Unix brauchst, das auf den verschiedensten Plattformen stabil läuft, ist NetBSD möglicherweise allerste (und manchmal die einzige) Wahl.

nach oben

2.5. Anwendungen für NetBSD

Wenn Du NetBSD installierst, hast Du ein reichliches Sortiment von Programmen und Applikationen zur Auswahl, die Du auf Deinem System installieren kannst. Abgesehen von den Unix-Standardtools, wie Editoren, Formatieren, C/C++-Compiler und -Debugger und so weiter gibt es eine grosse Anzahl (Ich denke, es sind wohl weit über 1000) von Paketen, die sowohl dem Quellcode heraus als auch als in vorkompilierter Form installiert werden können. Alle Pakete, die Du auf einem gut konfigurierten System erwartest, sind für NetBSD kostenlos erhältlich. Und dann gibt es da noch eine Anzahl kommerzieller Applikationen. Du kannst Dein System ausserdem mittels Nutzung der vorhandenen Emulatoren Binaries anderer *nix- Betriebsysteme laufen lassen. Die Linux-Emulation ist hier das prominenteste Beispiel; das von nahezu allen NetBSD- Usern angewendet wird. Du kannst die Linux-Versionen von:

- Netscape
- Acrobat Reader
- Doom, Quake
- Adobe FrameMaker
- vielen anderen Programmen

auf dem System laufen lassen.

NetBSD kann auch FreeBSD, BSDI und andere Systeme emulieren.

nach oben

2.6. Die Philosophie von NetBSD

Im Gegensatz zu vielen anderen Betriebsystemen ist eine NetBSD- Installation mit Features gesegnet, ohne viel Platz zu verbrauchen; weil es danach strebt, ein komplettes Basissystem ohne Redundanzen zu sein. Nach der Installation hast Du ein voll funktionsfähiges System, bei dem manche Leute vielleicht eine Anzahl von Applikationen vermissen werden; zum Beispiel den Webbrowser (NetBSD betrachtet den Webbrowser nicht als Systembestandteil, im Gegensatz zu anderen Betriebsystemen): Du hast die Freiheit, zu entscheiden, welche Programme Du auf dem System installieren willst. Die Installation neuer Software ist mit der Package-Collection sehr leicht.

Ein anderer Vorteil dieses Herangehens ist, dass das Basissystem auch ohne diese zusätzlichen Pakete funktioniert. Wenn Du Deine Perl- Version updaten willst, kannst Du das tun, ohne befürchten zu müssen, dass Du dabei andere Teile des Systems zerstörst. Wenn Du NetBSD installierst, wirst Du nicht viele vorgefertigte Pakete mit Anwendungen finden: Du magst das am Anfang als einen Nachteil sehen; aber wenn Du anfängst, die Philosophie dahinter zu verstehen, wirst Du herausfinden, dass Dir das zu mehr Freiheit verhilft. Wenn Du diese Softwaresammlungen installierst (bei denen jemand anderes eine Entscheidung für Dich gefällt hat), füllst Du Deine Festplatte nur mit Tonnen von Software, die meistens unbenutzt (und unbekannt) bleibt und nur Platz vergeudet (und vielleicht auch die Systemstabilität beeinträchtigt): Das ist etwas, das der typische BSD- User nicht haben will.

Gerade, wenn Du anfängst, NetBSD kennenzulernen, ist da immer etwas das Dich immer wieder erstaunen wird: Die extreme Stabilität des System und die Beachtung der Details; nichts scheint das Ergebnis einer Chance und alles ist durchdacht. Genau das ist es, was Qualität ausmacht und aus meiner Sicht ist genau das das Feature, das den Unterschied zu anderen Systemen ausmacht.

Wir können Tage damit verbringen, über die relativen Verdienste der verschiedenen Betriebsysteme zu diskutieren (einige Leute tun das gerne); aber wenn Du nichts ernsthaftes damit versuchst, kannst Du eigentlich kein Urteil darüber fällen. Ich lebe bequem, weil ich in den Mailinglisten sehr häufig sah, dass, wenn Du NetBSD ausprobierst, dass Du von der perfekten Balance zwischen Komplexität und Effektivität begeistert sein wirst. Für alle Probleme gibt es mehr als eine Lösung: NetBSD ist nicht glücklich mit »der einen« Lösung; aber es versucht, immer den einfachsten und elegantesten Weg zu gehen. NetBSD ist ein Werkzeug, das Dich Deine Arbeit so tun lässt, wie es Dir passt, ohne Dir im Weg zu stehen. In diesem Licht ist es ein optimales Werkzeug: Es ist wie die Benutzung eines Bleistifts: Du musst hart arbeiten, um zu lernen, wie er benutzt wird, aber wenn Du es einmal gelernt hast, kannst Du schreiben oder zeichnen, ohne an den Stift zu denken.

nach oben

2.7. Wie man an NetBSD herankommt

Es gibt keine offiziellen Vertriebe von NetBSD- CD-Roms, aber es gibt verschiedene Wiederverkäufer. Du findest eine aktuelle Händlerliste auf der relevanten Seite auf der NetBSD-Site. URL ist: http://www.netbsd.org/Sites/cdroms.html . Natürlich kannst Du NetBSD auch aus dem Internet von einem der Spiegel downloaden.

nach oben

Inhaltsverzeichnis

Kapitel 3: Installation

Aus dem Inhalt:

3.1. Dokumentation

Der Grossteil der NetBSD- Dokumentation wurde im Manpage-Format erstellt und ergibt eine hervorragende technische Referenz zum System. Ich will nicht leugnen, dass das nicht als Tutorial gemacht und gedacht ist (Abgesehen davon, kannst Du bei der Installation ohnehin nicht auf diese Seiten zugreifen). Das sind dann auch schon die zwei wichtigsten Gründe, warum es diesen Guide überhaupt gibt.

Anmerkung: Natürlich kannst Du über das Web mit einem anderen Rechner die Homepage von NetBSD) zugreifen. Aber ich glaube nicht, dass das Lesen dieser Seiten ein geeigneter Weg ist, das System kennenzulernen.

Nach der Installation wirst Du ein paar NetBSD- Guides im »/usr/share/doc«- Verzeichnis finden. Diese sind in drei Bereiche unterteilt: »psd«(UNIX Programmer's Supplementary Documents), »smm« (UNIX System Manager's Manual) und »usd« (UNIX User's Supplementary Documents). Du kannst die Texte auf der Konsole lesen, hier ein Beispiel:

```
$ cd /usr/share/doc/smm/09.sendmail
$ nroff -me 09.sendmail/intro.me | more
```

...oder Du erzeugst einen Postscript- Output mit Hilfe der Makefiles. Es ist unbestreitbar, dass es einen gewissen Mangel an HOWTOs gibt. Aus diesem Grund solltest Du für die meisten ein Make durchführen. Die NetBSD- Release beinhaltet ein paar Dokumente im Textformat. Und auf der NetBSD- Website findest Du weitere Informationen und FAQs.

Originaldokumentation: Die NetBSD- Website verfügt über verschiedene Seiten mit Dokumentationen und HOWTOs sowie über plattformspezifische Informationen. Diese Sachen sind gut geschrieben und normalerweise leicht zu verstehen. Hier sin ein paar Beispiele:

- Wie man auf eine DOS/Windows- Partition zugreift
- Wie NetBSD mit dem Windows NT- Bootloader gestartet wird
- ...

Zu allen NetBSD- Versionen gehüren auch die folgenden Dateien:

INSTALL

> Installationshinweise: Das ist das wichtigste Dokument, das Du aufmerksam lesen solltest (Möglichst mehrmals). Darin steht eine Beschreibung des NetBSD- System, eine Liste mit der unterstützten Hardware und das allerwichtigste: Die Anleitungen zur Installation.

README.first

Das solltest Du auch lesen.

release.man

beschreibt die Struktur der NetBSD- Release, die Du installierst. Ist eine Textdatei im »man«- Layout: sie ist vorformatiert und kann mit jedem Editor gelesen werden.

Auf der NetBSD- Website findest Du, neben den anderen, folgende Guides:

NetBSD FAQ

Grundsätzliche Informationen und Links zu anderen FAQs.

NetBSD/i386 FAQ

NetBSD/i386 specifische FAQs.

Basic NetBSD Networking

Anleitung zu Netzwerk und PPP- Konfiguration.

nach oben

3.2. Das Layout einer NetBSD- Installation

Das Layout der Dateien einer NetBSD- Installation wird im dazugehöhrigen »INSTALL«- Text beschrieben. Die i386- Binaries befinden sich im »i386/binary/sets«- Verzeichnis und die Quellen im »source/sets«- Verzeichnis, um hier mal wieder ein Beispiel zu nennen. Im »source/patches«- Verzeichnis liegen die Patches für die Basisrelease, die für gewöhnlich Fehler oder Sicherheitsprobleme beseitigen, die nach dem Erscheinen der Release entdeckt worden sind.

nach oben

3.3. Installation

Das erste, was vor der Installation getan werden muss, ist es, die Release- Informationen und Installationshinweise zu lesen, die in »INSTALL« stehen: Das ist die offizielle Beschreibung der Installationsprozedur. Als nächstes solltest Du Dich jetzt entscheiden, von welchem Medium aus die Installation durchgeführt werden soll. Du hast die Wahl zwischen:

- ftp
- nfs
- CDROM
- floppy
- nicht gemountetem Dateisystem
- lokalen Verzeichnissen

3.3.1. Tastatur

In sysinst kann das Tastaturlayout während der Installation nicht geändert werden. Wenn Du eine US- Tastatur hast, ist alles ok; für den Rest der Welt ist das zwar ein kleines Ärgernis, aber kein

grosses Problem. Wenn Du von einer CD-ROM installierst, musst Du nur alphanumerische Tasten benutzen (die meistens dasselbe Layout haben). Nur an einigen Plätzen musst Du andere Tasten drücken. Ich hoffe, dass das Installationsprogramm der nächsten Release eine Änderung der Tastaturbelegung erlaubt; für den Moment kannst Du aber diese Tabelle verwenden, um Dich zu orientieren:

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

Wenn Du eine non-US- Tastatur hast; ist das erste, was Du tun wirst, das Ändern des Layouts. Bis dahin muss ich aber noch um Geduld bitten.

3.3.2. Geometrien

Das Installationsprogramm kennt zwei Arten von Festplattengeometrien. Du solltest wissen, was sie bedeuten:

- reale Geometrie
- BIOS- Geometrie

Die »reale Geometrie« der Platte ist die Geometrie, die das System aus der Hardware ermittelt. Die »BIOS«- Geometrie ist die, die das BIOS benutzt. Diese kann sich von der realen unterscheiden (Beispiel: Das BIOS kann die Platte mittels LBA remappen). Die Platte, die wir bei der Installation benutzen, ist ein IDE-Modell mit diesen Daten:

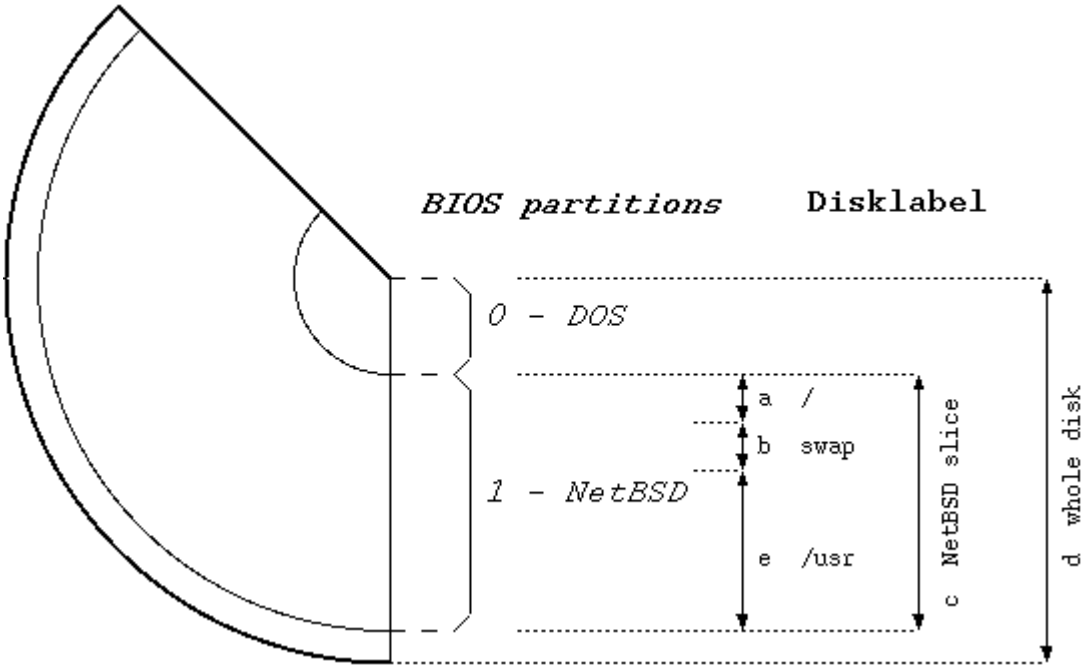
```
real:  6232 cyl,   16 heads,  63 sec
BIOS:   779 cyl,  128 heads,  63 sec   (LBA)
```

Wie Du sehen kannst, remappt das BIOS die Platte mittels LBA. Die Anzahl der Zylinder wird dabei reduziert und die Anzahl der Köpfe erhöht (Aber das Ergebnis ist das Gleiche: 6232*16=779*128=99712). Zu einem Sektor gehören 512 Byte, was bedeutet, dass die Platte 6232*16*63*512= 3GB gross ist. NetBSD muss die Geometrie nicht remappen (und wird es auch nicht tun). Während der Installation kann die Geometrie manuell geändert werden, wenn sysinst sich geirrt haben sollte.

3.3.3. Partitionen

Die Terminologie, die NetBSD für die Partitionierung benutzt, unterscheidet sich von der typischen DOS/Windows- Terminologie. Faktisch gibt es da zwei Schemata für die Partitionieruung. NetBSD installiert sich in einer der vier primären BIOS- Partitionen (die in der Festplatten-Prtitionstabelle definiert werden).
In seiner BIOS- Partition (auch »Slice« genannt) definiert NetBSD seine Partitionen mit Hilfe eines »Disklabes«. Diese Partitionen können nur von NetBSD gesehen werden und identifizieren sich durch Kleinbuchstaben (Beginnend mit »a«). wd0a ist beispielsweise die »a«- Partition auf der ersten IDE-Festplatte (wd0 und sd0a ist eben die erste »a«- Partition auf der ersten SCSI- Platte. In Bild 3-1 gibt es zwei ptimöre BIOS- Partitionen. Eine benutzt DOS und die andere NetBSD. NetBSD beschreibt das Layout der Platte durch das Disklabel.

Bild 3-1. Partitionen



Anmerkung: Die Bedeutung der Partitionen »c« und »d« ist typisch für den i386- Port. Andere Ports benutzen andere Konventionen (Bsp.: »c« repräsentiert die ganze Platte).

Anmerkung: Wenn NetBSD die Platte mit einem anderen OS teilen muss (wie im vorigen Beispiel), musst Du möglicherweise einen Bootmanager installieren; ein Programm also, dass Dir die Wahl des zu startenden Betriebsystems beim Bootvorgang erlaubt. Sysinst kann einen einfachen und effektiven Bootmanager installieren.

Wenn Windows NT auf der selben Platte installiert ist, kannst Du den NT- Bootmanager benutzen, um NetBSD zustarten. Ein einfacher Weg, um das einzurichten, ist auf der NetBSD- Website beschrieben.

3.3.4. Benötigter Plattenplatz

Der Platz, den NetBSD für seine Installation voraussetzt, hängt davon ab, was Du mit der Maschine tun willst (Bsp. Server oder Workstation). Beispiel: Wenn Du ein normales Desktop- System für den Hausgebrauch nebst X, den Kernelquellen und ein paar Applikationen wie Netscape auf einer 420MB- Platte installieren willst (was heutzutage schon sehr klein ist): Die Swap- Partition ist 32 MB gross. »df« gibt dann das hier aus:

```
Filesystem  1K-blocks     Used    Avail Capacity  Mounted on
```

```
/dev/wd1a       31887    16848    13444    56%    /
/dev/wd1e      363507   173202   172129    50%    /usr
```

Wie Du sehen kannst, hast Du noch 180 MB übrig. nach oben

3.3.5. Neuer Versuch

Wenn Du zum ersten Mal ein Betriebssystem installierst, wird Dir das kaum auf Anhieb gelingen. Das gilt auch für NetBSD. Gerade, wenn alles glattging, wirst Du kurz nach der Installation vielleicht feststellen, dass Du (wieder ein Beispiel) das Layout Deiner Platten vielleicht anders hättest anlegen sollen. Es ist wichtig, hier nicht aufzugeben; wenn Du es noch einmal versuchst, wirst Du begreifen, dass es eben nicht soo einfach ist, alle Abläufe im ersten Anlauf zu begreifen. Vieles wird eben erst später klar, wenn man die ersten Erfahrungen gesammelt und sich das INSTALL- Dokument mehrfach durchgelesen hat. Während der ersten Installation ist es eigentlich eine weise Entscheidung, die Defaultwerte zu akzeptieren, die sysinst einem so anbietet und aufÄndereungen, beispielsweise am Disklabel, zu verzichten. Das einzige, das Du entscheiden musst, ist die Grösse des Plattenplatzes, den NetBSD für sich in Anspruch nehmen darf.

nach oben

3.4. Installationsbeispiel

Der wesentliche Teil dieses Kapitels dreht sich um eine reale Installation für ein gängiges System in einer weit verbreiteten Vorgehensweise: Der Installation von einer CD-ROM. Die Konzepte sind in allen Arten der Installation die selben; der einzige Punkt, der einen Unterschied ausmacht, ist der Weg, auf dem die Binary- Sets über sysinst ihren Platz auf der Platte finden (Bsp.: ftp). Bitte denke daran, dass sich einige Details der Installation von Release zu Release unterscheiden können: Dieses Beispiel orientiert sich an Version 1.5. Um einen gewissen Lerneffekt zu erreichen, wird in dieser Prozedur immer der »schwierigste« Weg gewählt.

- BIOS- Partitionstabelle voll: Eine oder mehrere Partitionen werden gelöscht, um Platz für NetBSD zu schaffen.
- Partitionierung mittels fdisk; Benutzung von Sektoren anstelle von MB
- Manuelle Modifikation des Disklabels, das von sysinst erstellt wurde; Berechnung ebenfalls in Sektoren.
- »custom«- Installation (was bedeutet, dass Du jedes Binary- Set einzeln auswählen kannst, das Du installieren willst).

Dieses Sortiment von Wahlmöglichkeiten vermittelt den Eindruck, dass es sehr kompliziert ist und eine Menge Arbeit verlangt: Bedenke aber, dass mit dem Akzeptieren der Defaults die ganze Angelegenheit weitaus einfacher ist. Andererseits ist ein Tutorial, das nur die »einfachen« Bestandteile einer Installation erläutert, nicht der nützlichsten Eines (Die Marketingabteilungen einiger bekannter Firmen sehen das aber anders...).

nach oben

3.4.1. Vorbereitung der Installation

Vor der Installation ist es zweckmässig, sich einen detaillierten Plan über die Schritte, die Du gehen musst zurechtzulegen. Als Erstes: Lies die »INSTALL«-Datei (Es sei hiermit versprochen, dass ich das zu letzten Male sage ;-)); lies die Beschreibung der Installation und überprüfe die Hardwarekompatibilitäten. Der nächst Schritt: Wenn schon etwas auf der Platte herumliegt, überlege

Dir, wie Du Platz schaffen kannst. Wenn NetBSD den Platz mit einem anderen System teilen muss, wirst Du vielleicht eine neue Partition erstellen müssen (was Du mit sysinst ohnehin tun wirst); und vielleicht musst Du eine existierende Partition in der Grösse verändern. Man kann mit sysinst die Grösse einer existierenden Partition nicht ändern; aber es gibt da ein paar kommerzielle Produkte (wie Partition Magic), und einige freie Tools (fips, pfdisk), die das können. Die Installation gliedert sich in zwei logische Schritte. Im ersten Teil erstellst Du die Partition und schreibst das Disklabel für NetBSD. Im zweiten Teil legst Du fest, welche Binärpakete in den neuen Partitionen installiert und extrahiert werden sollen. Der erste Teil ist unabhängig von der Installationweise (CDROM; NFS;FTP...). Am Ende des ersten Teils wurde noch nichts auf die Platte geschrieben; Du wirst nun gebeten, weiterzumachen. Wenn Du weitermachen willst; geht es weiter; ansonsten geht das Programm in das Hauptmenü zurück und die Platte bleibt unverändert.

nach oben

3.4.2. Herstellen der Installationsdisketten

> **Notiz:** Wenn Du ein bootfähiges CD-ROM-Laufwerk hast, brauchst Du keine Installationsdiskette: Schalte die«Boot from CD-ROM« in Deinen BIOS- Einstellungen ein; lege die CD ein und starte die Maschine neu. Diese Optionen gibt es eventuell nicht auf älteren Maschinen.

Bevor Du installierst, musst Du die Installationsdiskette erstellen, indem Du das Floppy- Image von der CD-ROM auf eine Diskette kopierst. Um das in DOS durchzuführen, kannst Du das »rawrite«- Programm im »i386/installation/mic«- Verzeichnis verwenden. Die Imagedatei ist »i386/installation/floppy/boot.fs«.

Bevor Du damit anfängst, stelle sicher, dass die Installationsdisketten in Ordnung sind. Dieser Aspekt wird oft übersehen und kann Dir eine Menge Ärger ersparen.

1. Formatiere die Diskette
2. Gehe in das Verzeichnis »i386\INSTALLATION\FLOPPY« auf der CD-ROM.
3. Starte das »\misc\rawrite«- Programm. Das »source file« ist »BOOT.FS« ud der »Destination drive« ist A:

Wenn Du die Disketten in einer Unix- Umgebung erstellst, kannst Du das mit dem »dd«- Kommando machen; Beispiel:

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

»dd« kopiert 512-Byte- Blöcke: Die »bs=36b«- Option kopiert 35 Blöcke auf einmal; Der Effekt ist, dass die Operation dann etwas schneller geht.

> **Notiz:** Eine 1440k- Floppy besitzt 1474560 Bytes, die sich auf 80 Zylinder, 2 Spuren, 18 Sektoren und 512 Byte pro Sektor verteilen (80*2*18 = 2880 Blöcke). Das »bs=36b« kopiert einen Zylinder (18*2 Blöcke) gleichzeitig und wiederholt die Operation 80 mal anstatt 2880 mal...

nach oben

3.4.3. Die letzen Vorbereitungen

Jetzt ist alles für die Installation vorbereitet; aber bevor Du durchstartest, ist es besser, noch etwas Information über die Hardware des PCs zu sammeln. Am wichtigsten ist es, den Typen der Festplatte (IDE, SCSI) und die Geometrie zu kennen. Du findest die Information entweder im Hanbuch der Disk oder mit Hilfe eines Diagnoseprogramms. Manche Platten besitzen auch einen Aufkleber, auf dem diese Daten stehen. Ein anderer Weg ist es, auf der Website des Herstellers nach einigen Informationen zu suchen. Wenn Du via ftp oder NFS installierst, solltest Du auch die Hardwareeinstellungen Deiner Netzwerkkarte kennen: Wenn der Installationskernel eine NE2000-kompatible Karte auf einem Interrupt erkennt, aber die Einstellung nicht mit der Erkenntnis des Kernels übereinstimmt; wirst Du nicht installieren können. Beispiel: Der Kernel kann eine NE2000-kompatible Karte mit den folgenden zwei Einstellungen erkennen:

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

Wenn Deine Netzwerkkarte anders eingestellt ist, wird der Kernel sie nicht finden (nach der Installation kannst Du allerdings einen eigenen Kernel mit Deinen Einstellungen kompilieren). Wo Du gerade dabei bist, solltest Du auch nach den anderen Harwaredetails Ausschau halten, wie die Anzahl der seriellen und parallelen Ports etc.. Das wird für die Installation zwar nicht verlangt, aber es kann später von Nutzen ein. Vergleiche Deine Einstellungen mit denen, die in der »INSTALL«-Datei stehen.

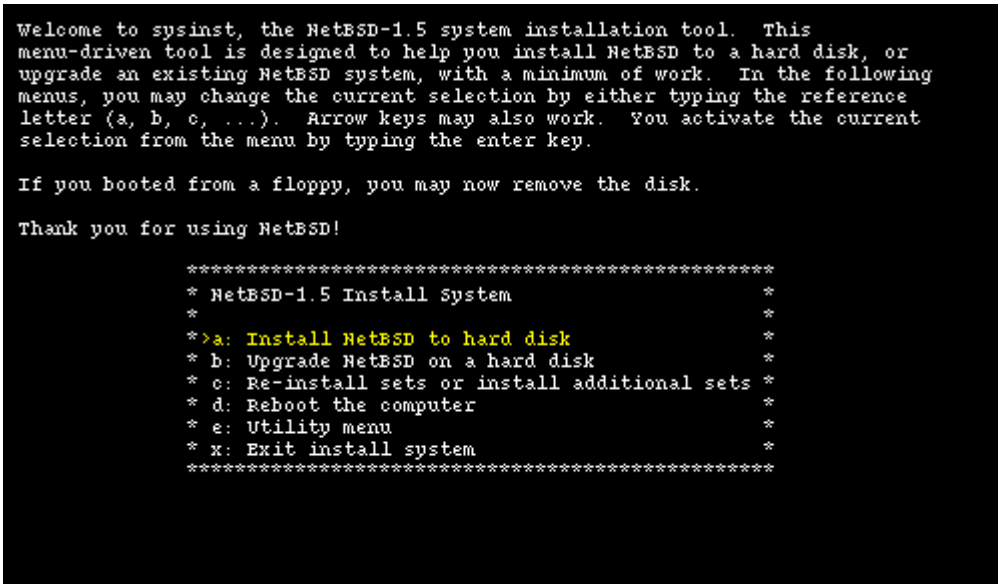
> **Hinweis:** Du kannst auch installieren, wenn Du die Geometrie Deiner Platte nicht kennst; genau, wie Du die anderen Details der Maschine nicht genauestens kennenmusst; Du musst Dich dann allerdings darauf verlassen, dass sysinst alles richtig macht.

nach oben

3.4.4. Der Anfang

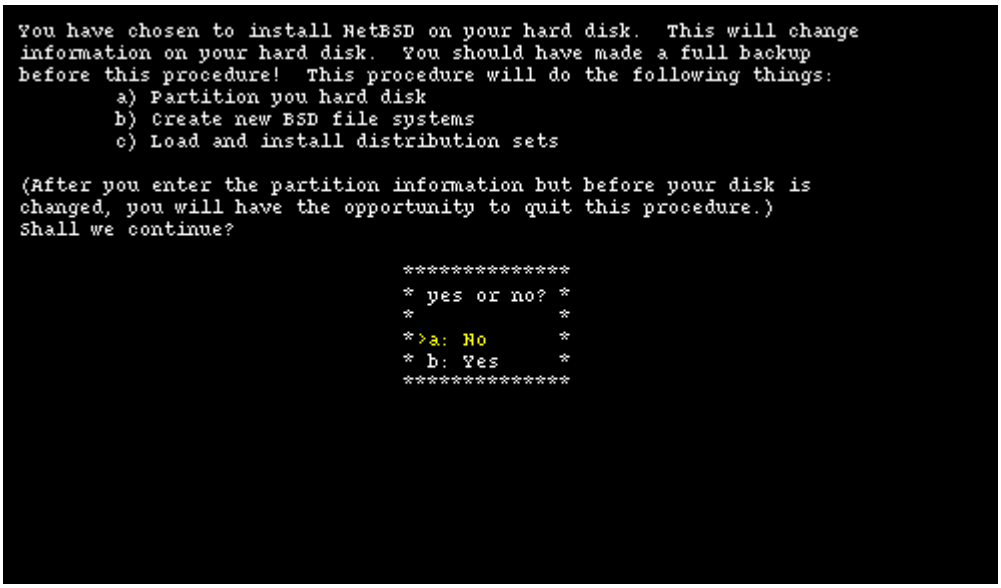
Schiebe die frischerzeugte Bootdiskette in in Laufwerk a: und starte das System neu (oder boote von der CD). Der Kernel wird gebootet und beginnt damit, eine Menge Meldungen auf dem Bildschirm anzuzeigen. Die meisten von ihnen sagen etwas darüber, ob Hardware nicht gefunden oder nicht konfiguriert wurd. Das ist normal: Der Kernel auf dem Bootmedium versucht die gesamte Hardware zu finden, die von NetBSD unterstützt wird. Du wirst seeehr wahrscheinlich nicht alle diese Geräte in Deiner Maschine haben.

Bild 3-2. Der Beginn der Installation



Wenn die Maschine mit dem Booten fertig ist, wirst Du Dich im Hauptmenü des Installationsprogramms wieder finden, wie in Bild 3-2 gezeigt. Wundere Dich nicht über das spartanische Aussehen von sysinst: es ist ein wirklich leistungsfähiges und flexibles Programm. Ab hier solltest Du den Anweisungen auf dem Bildcschirm folgen und das »INSTALL«- Dokument als Referenz benutzen. Die verschiedenen sysinst- Bildschirme besitzen alle mehr oder weniger dasselbe Layout: Im oberen Teil des Monitors wird eine kurze Beschreibung der aktuellen Operation oder eine kurze Hilfestellung gegeben; der mittlere Teil zeigt die aktuellen Einstellungen, die NetBSD gefunden hat und unten findet sich ein Menü, aus dem Du verschiedene Punkte wählen kannst. Die »Install«- Wahl bringt Dich auf den nächsten Bildschirm, (Bild 3-3) auf dem Du die Installation bestätigst.

Bild 3-3. Bestätigung der Installation

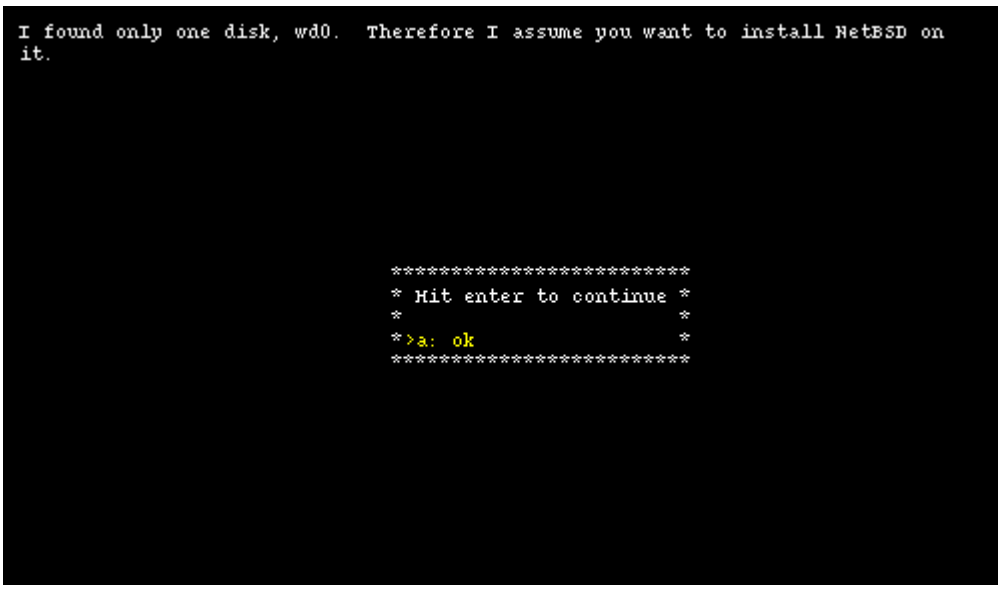


Nachdem Du Dich für »continue« mit Option b entschieden hast, is es Zeit, festzulegen, auf welche Festplatte NetBSD installiert werden soll. Wenn es mehr als eine Platte gibt, zeigt sysinst eine Liste mit allen vorhandenen Platten, von denen Du Dir eine aussuchen kannst In unserem Beispiel haben wir nur eine Festplatte und das Installationsprogramm zeigt uns nur eine informelle Nachricht, zu sehen in Bild 3-4.

Info: Die Informationen auf diesem Bildschirm sind abhängig von Typ und Anzahl der auf

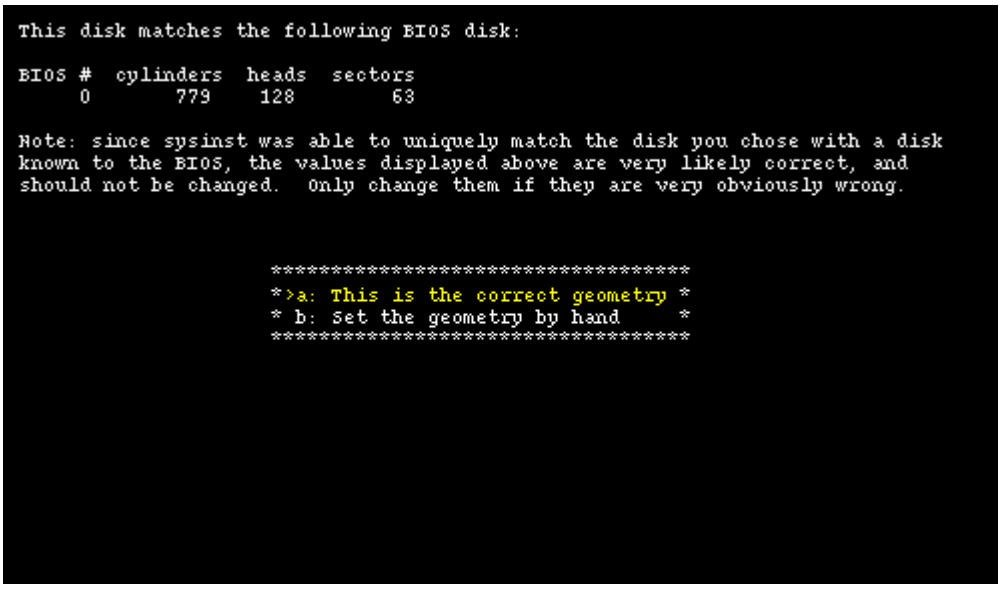
dem System installierten Platten.

Bild 3-4. Eine Platte auswählen



Als nächstes zeigt sysins die BIOS- Geometrie für die gewählte Platte. Du kannst das als korrekt bestätigen oder, falls das Installationsprogramm im Irrtum ist, neue Werte von Hand eingeben.

Bild 3-5. BIOS Geometrie:

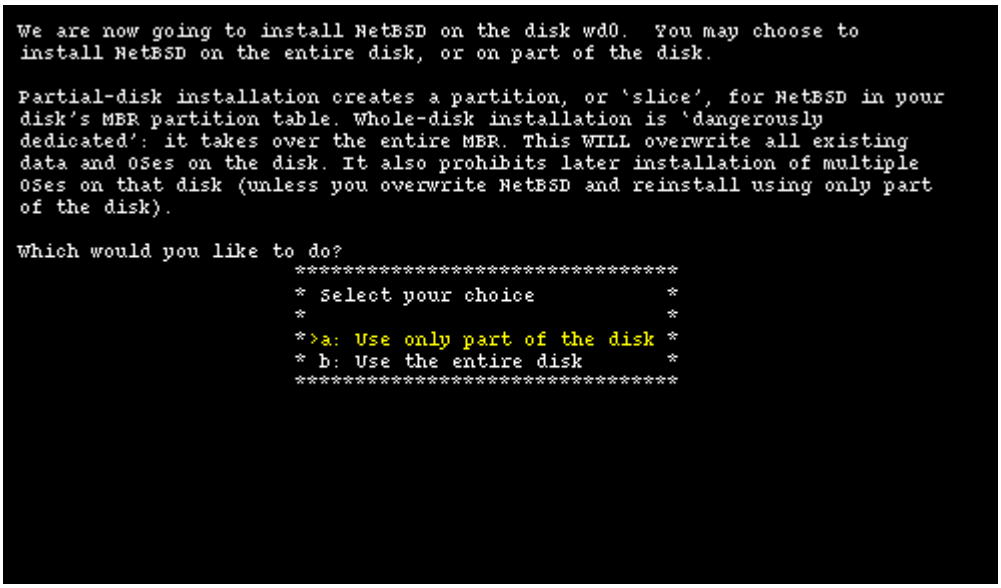


3.4.5. Partitionen

Der erste wichtige Schritt in der Installation muss nun getan werden: Die Partitionierung der Festplatte. Als erstes musst Du angeben, ob NetBSD einen Teil (empfohlen) oder die gesamte Platte benutzen soll (»gefährliche Wahl«). Im ersten Fall ist es immer noch möglich, eine Partition zu erstellen, die die gesamte Platte benutzt. Ergo empfehle ich, diese Option zu verwenden, die, wenn ich das richtig verstanden habe, die BIOS- Partitionstabelle in einem Format anlegt, das auch mit anderen Betriebssystemen kompatibel ist.
In diesem Beispiel benutzen wir eine Platte mit der folgenden »realen« Geometrie; korrespondierend mit der BIOS- Geometrie in Bild 3-5.

```
6232 cyl, 16 heads, 63 sec  (6232 x 16 x 63 = 6281856 total sectors)
1 sector = 512 bytes
1 track = 63 sectors = 63 * 512 bytes = 32 K
1 cylinder = 16 * 63 * 512 bytes = 504 K
```

Bild 3-6. Auswahl des Partitionsschemas



Der nächste Schritt, der in Bild 3-7 gezeigt wird, ist die Auswahl einer Masseinheit für die Partitionierung der Platte: Sektoren bieten einem hier die grösste Flexibiltät und Präzision (Es ist für gewöhnlich aus Performancegründen besser, die Partitionsgrössen an die Zylinder zu binden. Zumindest für ältere Platten gilt das.). Megabytes sind allerdings einfacher und »intuitiver« zu benutzen, weil man sich so ein bisschen Rechenarbeit erspart. **Bild 3-7. Masseinheit auswählen**

In diesem Tutorial werden wir allerdings Sektoren verwenden, weil ihr Gebrauch für Lehrzwecke geeigneter ist. Option »c« bringt Dich auf den fdisk- Bildschirm. **Bild 3-8. fdisk**

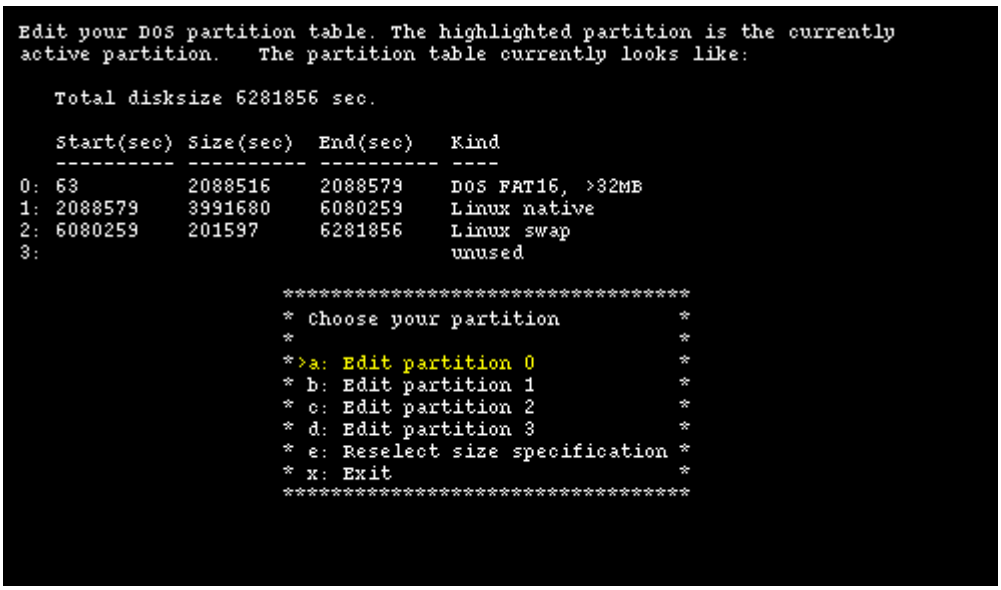
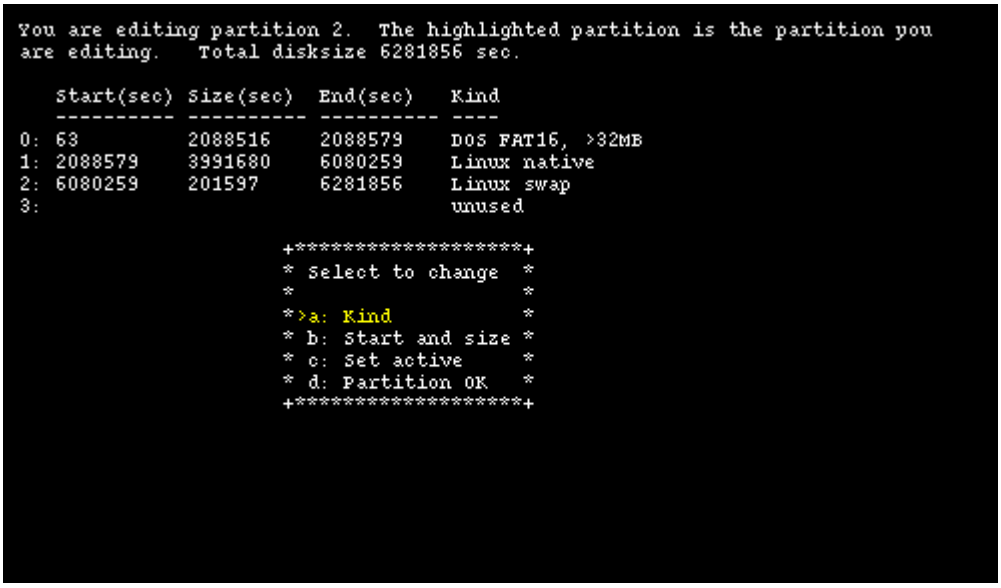


Bild 3-8 zeigt die Situation auf der Festplatte vor der Installation von NetBSD. Da sind schon vier primäre Partitionen: Auf einer liegt DOS/Windows und zwei sind von GNU/Linux belegt. Es gibt keinen freien Platz auf der Platte: Die »End«- Spalte von Partition 2 zeigt, dass die 6281856 Sektoren der Platte vollständig belegt sind.

> **Hinweis:** Im fdisk- Bildschirm wird folgende Formel angezeigt:
>
> `Start(sec) + Size(sec) = End(sec)`
>
> Das bedeutet, dass die »End(sec)«- Spalte einer Partition der »Start(sec)«- Spalte der nächsten entspricht, was nicht sehr intuitiv ist, weil der Sektor in der »End(sec)«- Spalte einer Parttion aktuell schon der nächsten gehört. Disklabel benutzt eine andere (und logischere) Konvention.

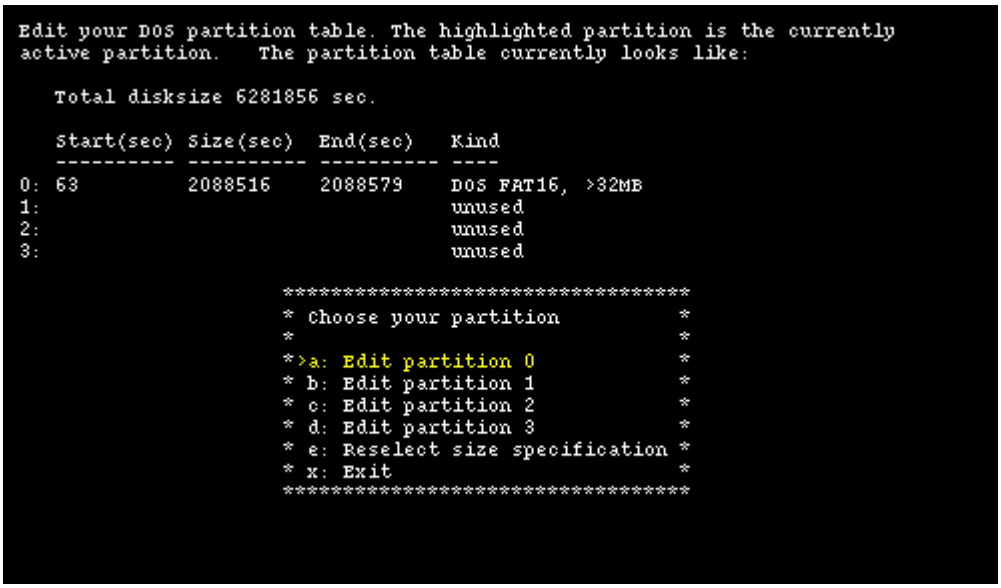
Um Platz zu schaffen, müssen die beiden Debian GNU/Linux- Partitionen dran glauben; die letzte zuerst. Sysinst zeigt einen Bildschirm, auf dem die vorhandenen Daten für eine Partition geändert werden können und Bild 3-9 zeigt die aktuellen Daten für Partition 2.

Bild 3-9. Löschen einer Partition



Um eine Partition zu löschen, wird »unused« mit der Option »a« gewählt; anschliessend verlässt Du mit der Option »b« die »Start« und »Size« - Felder im leeren Zustand (einfach Enter drücken). Zum Schluss bestätigst Du den Typ »unused« mit der Option »d«; un schon bist Du wieder im fdisk-Hauptbildschirm, in dem Pertition nun leer ist. Benutze dieselbe Methode, um Partition 2 und 1 zu löschen. Lasse nur Partition 0 auf der Platte (Bild 3-10).

Bild 3-10. Gelöschte Partition



Nur die DOS/Windows- Partition bleibt in der Grösse von 2088516 Sektoren erhalten, was 1029 MB entspricht (etwas mehr als 1 Gigabyte). Der freie Platz ergibt sich aus der Gesamtzahl der Sektoren abzüglich des von DOS/Windows genutzen und dem Endsektor der DOS- Partition (Die Zahl in der »End(sec)- Spalte):

```
6281856 - 2088579 = 4193277 sectors = 2047 MB frei auf der Platte
```

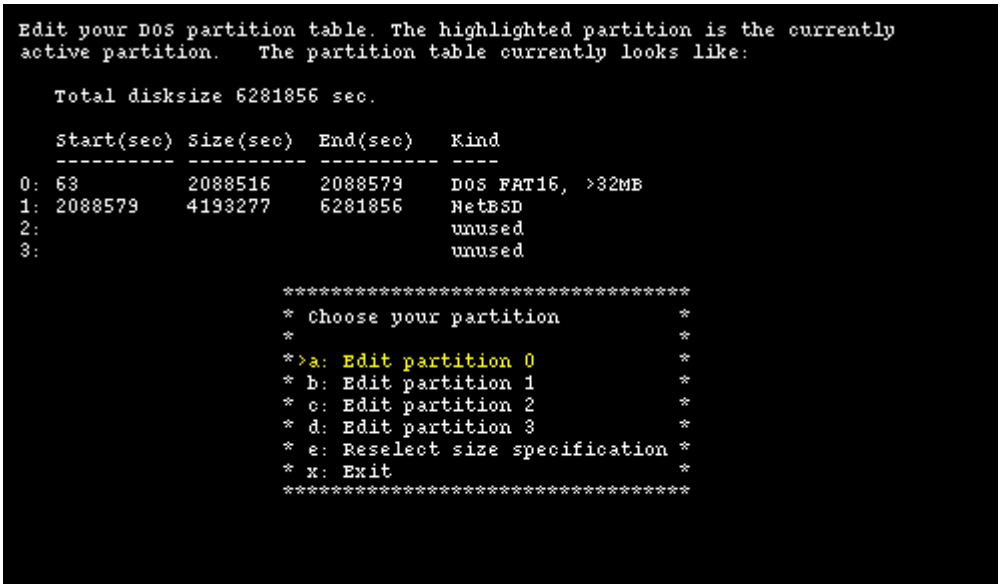
> **Hinweis:** Die DOS- Partition beginnt bei Sektor 64 und nicht bei Sektor 0, wie Du vielleicht glaubst. Ds ist nicht ungewöhnlich: Die erste Spur (63 Sektoren) ist reserviert. Auf Spur 0, Sektor 1 der Platte ist der Master-Boot-Record (MBR). Wenn das System das BIOS gebootet hat, lädt das BIOS den MBR von der Platte in den Speicher, ermittelt, welche Partition aktiv ist und lädt den Bootsektor dieser Partition in den Speicher, um um ihm dann das Heft zu übergeben. Dann startet der Bootsektor das Betriebssystem auf seiner Partition.

Wenn Du jetzt die Option »b« verwendest, wird eine neue Partition für NetBSD erstellt, die am Ende der DOS- Partition beginnt. Um diese neue Partition zu erstellen, müssen diese Informationen vorhanden sein:

- Der Typ der neuen Partition
- Der erste Sektor der neuen Partition
- Die Grösse der neuen Partition

Wähle den Typ »NetBSD« für die neue Partition (Option a: »kind«) und gib die berechneten Daten ein (Option »b«): Start: 2088579 und Grösse: 4193277. Prüfe diese Eingabe auf Korrektheit und stimme der Erstellung mit Option »d« zu; was Dich in das fdisk- Hauptmenü zurückbringt. Das Ergebnis wird in Bild 3-11 gezeigt, das das endgültige Layout der Partitionstabelle zeigt. Jetzt gehst Du mit der Option »x« zum nächsten Menü.

Bild 3-11. Die fertigen Partitionen



> **Hinweis:** sysinst für NetBSD 1.5 prüft auch die Start- und Endsektoren der unbenutzten Partitionen; nur die Information darüber wird nicht auf dem Bildschirm ausgegeben. Deshalb kann es passieren,dass das Programm eine Warnung über überlappende Partitionen ausgibt, besonders, wenn alles auf dem Schirm korrekt erscheint. Ich rate dazu, die Startsektoren und die Grössen der Partitionen genau anzugeben.

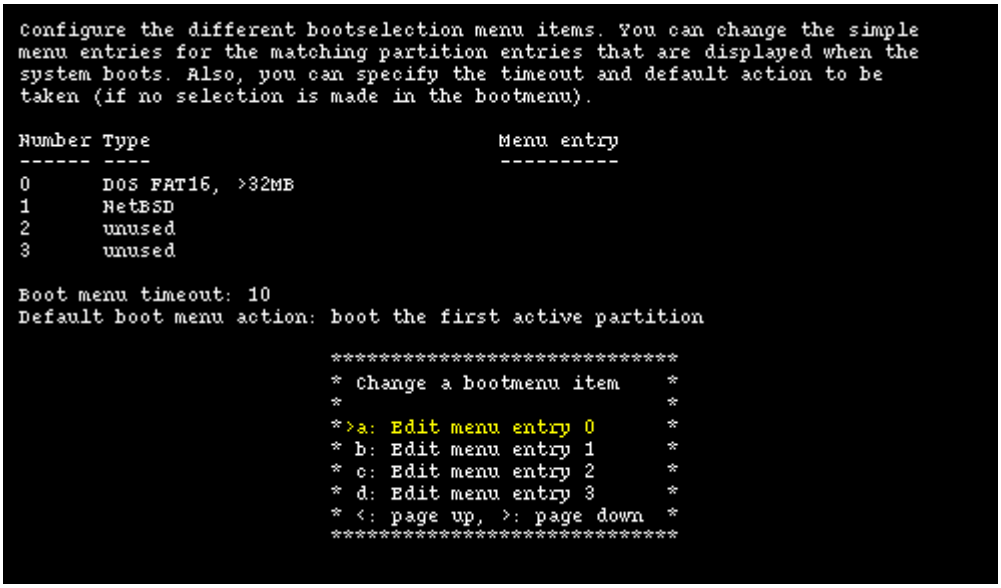
Wenn Du in der Partitionerung einen Fehler gemacht hast (vielleicht überlappen sich da ja zwei Partitionen); wird sysinst eine Nachricht ausgeben und empfehlen, zum fdisk- Menü zurückzugehen (es ist aber auch erlaubt, weiterzumachen). Wenn die Daten auf der NetBSD- Partition korrekt sind, aber die NetBSD- Prtition ausserhalb der Sektoren liegt, die das BIOS booten kann, warnt sysinst Dich und fragt, ob Du irgendwie weitermachen willst. Das kann bei älteren PCs zu Problemen führen: Der PC in unserem Beispiel bekommt diese Warnung, bootet aber perfekt. Es ist nicht möglich, hier eine generelle Regel zu setzen (hägt vomBIOS ab). Wenn der PC nicht sehr alt ist, kannst Du die Warnung meistens ignorieren und mit der Installation fortfahren.

> **Anmerkung:** Das ist kein von NetBSD gesetztes Limit. Einige alte BIOSe können nicht von einer Partition booten, wenn sie oberhalb des 1024. Zylinders liegt. Um das Problem voll und ganz zu verstehen, solltest Du die verschiedenen BIOSe und die vielen Adressschemata untersuchen, die sie verwenden (phylikalische CHS, logische CHS, LBA usw...). Diese Dinge werden in diesem Guide nicht beschrieben.
>
> Mit den meisten neueren BIOSen, die int13- Erweiterungen unterstützen, ist es möglich NetBSD auch von Partitionen zu booten, die ausserhalb der ersten 8 GB der Festplatte liegen, vorausgesetzt, der NetBSD- Bootselector ist installiert.

Wenn die Daten korrekt sind und sysinst herausfindet, dass da mehr als ein Betriebssystem auf der Festplatte installiert ist, bietet es an, ein Bootmenü auf der Platte zu installieren. Mit dem Installationsprogramm kannst Du beides tun: Den Bootmanager installieren und ihn konfigurieren. Du musst nur die Zeichen angeben, die im Bootmenü für jedes Betriebssystem beim Systemstart ausgegeben werden. Ausserdem musst Du angeben, weches System beim Start als Defaultwert hochgefahren werden soll, wenn der Benutzer keine Auswahl trifft. Dieser Bildschirm wird in Bild 3-12 gezeigt.

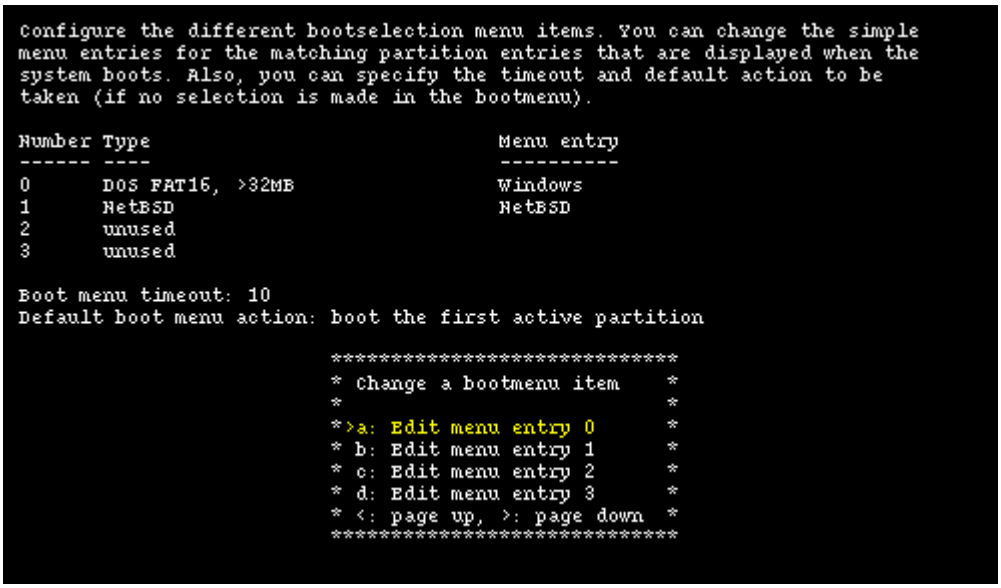
> **Hinweis:** Du kannst mit den »> und »<«- Tasten navigieren, wenn die Funktionstasten nicht funktionieren.

Bild 3-12: Konfiguration des Bootselektors



Wähle die Partitionen aus, auf die der Bootmanager zugreifen soll und definiere einen Namen für den Menünamen mit Hilfe der Optionen »a« bis »d«. In der Menü- Eintragsspalte solltest Du einen Eintrag für jede bootfähige Partition sehen, wie in Bild 3-13 zu sehen ist.

Bild 3-13. Konfiguration des Bootmanagers



Mit Option »e« kannst Du die Zeit festlegen, die der Bootmanager warten soll, bis er weitermacht: Wenn eine definierte Zeitspanne ohne Auswahl des Users erreicht ist, wird von der als Defaultwert eingestellten Partition gebootet. Du kannst eine der folgenden als Defaultwerd nutzen:

- Eine Partition
- Eine andere Platte
- Die erste aktive Partition

Wenn Du mit der Konfiguration des Bootmanagers fertig bist, ist der erste Teil der Installation, das Partitionieren der Festplatte zu Ende. Die BIOS- Partitionen, in NetBSD such slices genannt, sind erstellt worden; es gibt jetzt zwei Slices: DOS und NetBSD. Es ist jetzt Zeit, die NetBSD- Partitionen zu definieren.

nach oben

3.4.6. Das Disklabel

Es gibt drei Alternativen, die NetBSD- Pertitionen zu erstellen, wie in Bild 3-14 gezeigt wird.

Bild 3-14. Disklabel

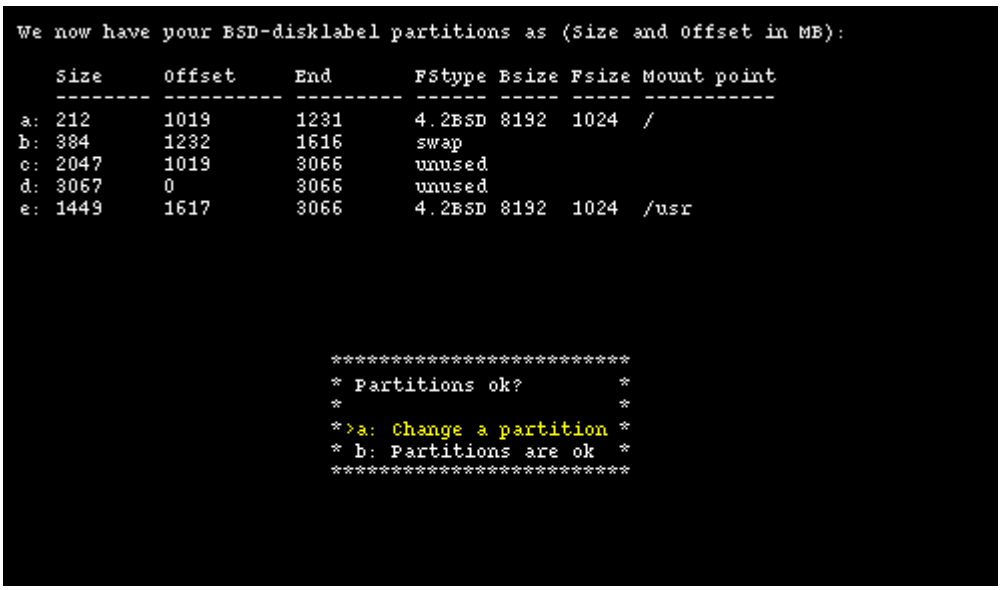
Wenn Du zum erstenmal eine Installation durchführst, empfehle ich »a« oder »b«; und sysinst damit die Entscheidung über die Partitionierung zu überlassen. In diesem Beispiel wird das Leben allerdings etwas komplizierter, weil wir das Disklabel manuell ändern wollen (Nur zu Lehrzwecken natürlich).

> **Hinweis:** Gerade wenn Du dem System die Entscheidung überlässt, ist es immer noch besser, das Disklabel sorgfältig zu überprüfen. Wenn der Platz auf der Platte unzureichend ist, ist das 2.5-sysinst intelligent genug, es herauszufinden und Dich zu warnen. Ältere Versionen taten das nicht und erstellten in aller Schweigsamkeit ungültige Partitionen.

3.4.7. Herstellen eines Disklabels

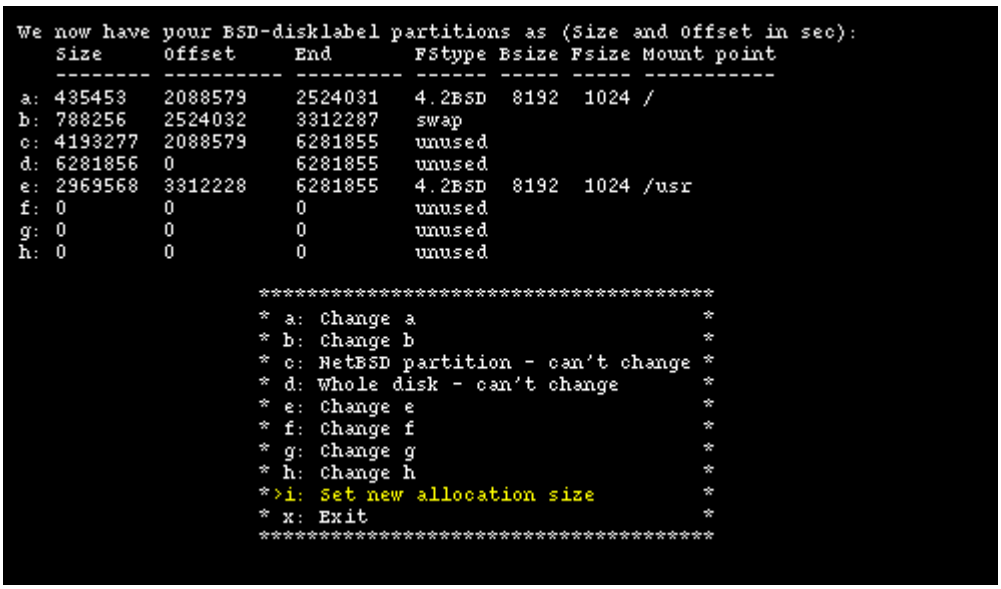
Als Erstes lassen wir das Installationsprogramm automatisch ein Disklabel erstellen. Greife zu Option »b« von Bild 3-14 und wir kommen zu Bild 3-15.

Bild 3-15. Standarddisklabel



Wenn das getan ist, brauchst Du nur alles für richtig zu befinden und zu bestätigen (mit Option »b«); damit ist Deine Arbeit erledigt. Stattdessen lass uns mal sehen, was zu tun ist, wenn die Swap- kleiner und die /usr«- Partition grösser werden sollen. Um die Grösse zu ändern wählst Du die »a«-Option: Im neuen Bildschirm wechseln wir die Masseinheit in Sektoren. Das Ergebnis siehst Du in Bild 3-16.

Bild 3-16. Modifikation des Disklabels (sec)



Die Identifikatoren der Partitionen sind standardisiert: Einige Buchstaben sind vordefiniert:

- *a* ist normalerweise die Root- Partition
- *b* ist die Swap- Partition.
- *c* beschreibt den gesamten NetBSD- Slice.
- *d* umfasst die gesamte Platte: also alles, auch das, was ausserhalb von NetBSD liegt. Mit einer ähnlichen Methode kannst Du eine DOS oder Linux- Partition für NetBSD lesbar machen, die sich ausserhalb des NetBSD- Slices befindet.
- *e* ist die erste freie Partition. Normalerweise wird sie nach »/usr« gemountet.

  Anmerkung: Was unter einer Partition zu verstehen ist, kann sich von Port zu Port unterscheiden. In dieser Beschreibung geht es speziell um den i386- Port.

Im Normalfall wirst Du die Partitionen »b« und »c« nicht ändern wollen. Es steht Dir aber frei, die Grössen und Mountpoints zu ändern und neue Partitionen zu ertellen (maximal 8, benutzt werden die Buchstaben e bis h). Wenn Du die Swap- Partition ändern willst, musst Du Partition »b« modifizieren. Du wirst auch Partition »e« ändern, so dass sie direkt hinter dem Ende der Partition »b« endet. Die Partitionen »c« und »d« bleiben unverändert. Du erstellst jetzt eine 150 MB (307200 Sektoren) grosse Swap- Partition; was bedeutet, dass »b« bei Sektor 2524032 startet und bei 2831231 (2524032+307200-1) endet:

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

Der neue freie Platz geht an Partition »e«, die so aussieht: Start: 2831232; Grösse: 3540624 und Ende:6281855. Diese Werte werden wie folgt berechnet: »Start« ist der Sektor, der unmittelbar auf den Endsektor von Partition »b« folgt; »Ende« ist gelich dem letzten Sektor der NetBSD- Partition; »Grösse« ergibt sich aus »Ende- 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
```

Das Beispiel zeigt das Disklabel, das Du willst. Mit Option »b« und »e« kannst Du die berechneten Daten eingeben. Das ist in Bild 3-17 dargestellt.

Bild 3-17. Modifizieren einer NetBSD- Partition

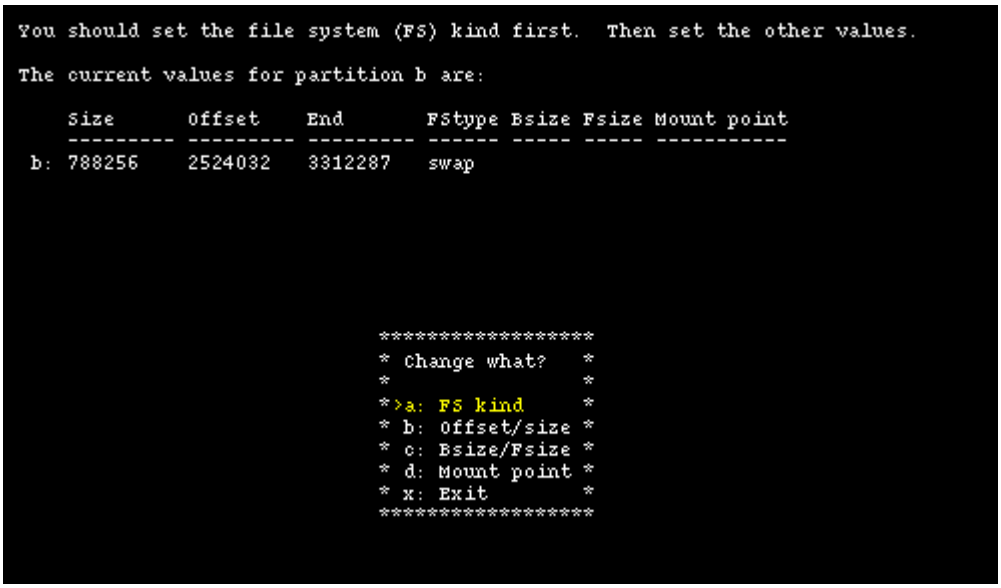
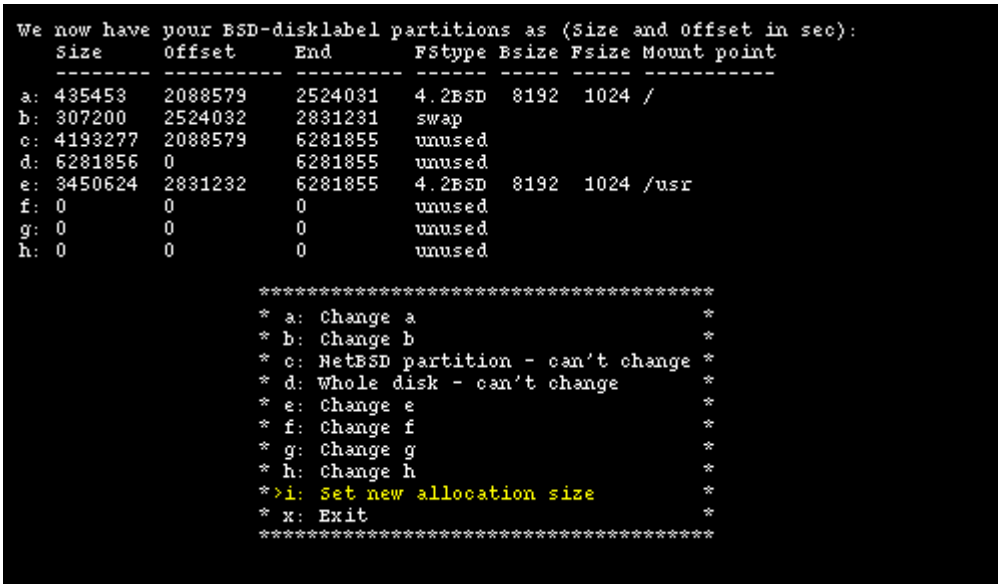


Bild 3-18 zeigt das geänderte Disklabel

Bild 3-18. Modifiziertes Disklabel



> **Partitionsgrössen:** Es ist sehr schwierig, eine generelle Regel für eine Entscheidung darüber zu geben, wie gross die einzelnen Partitionen und ihre günstigsten Grössen sein sollen: Das hängt davon ab, was der Computer tun soll (Server, Workstation, Mailserver...). Das ist der Grund für die Empfehlung, die von sysinst erzeugten Defaultwerte zu verwenden. Ein komplexer Server braucht wahrscheinlich eine weitaus ausgefeiltere Partitionierung. Darauf wissen die Leute Antwort, die öfter mit der Materie zu tun haben.

Wenn Du mit den Resultaten zufrieden bist, kannst Du Option »x« anwählen, um die Sachen zu speichrn und das Programm zu verlassen. Du bist jetzt zurück in Bild 3-15, wo Du Option »b« anwählen kannst.

nach oben

3.4.8. Abschliessende Operationen

Der schwierige Teil (Erstellen der BIOS- und BSD- Partitionen) ist jetzt vorbei. Der übrige Teil der Installation ist weitaus einfacher. Du kannst die Festplatte jetzt mit einem Namen versehen (Defaultwert ist »mydisk«) und die gemachten Operationen bestätigen.

> **Hinweis:** Alles, was Du bis hierher getan hast, wurde nicht auf der Platte abgelegt: Es ist Dir immer noch möglich, Deine Meinug zu ändern und zum sysinst- Huptmenü zurückzugehen, ohne irgendwelche Spuren auf der Platte zu hinterlassen.

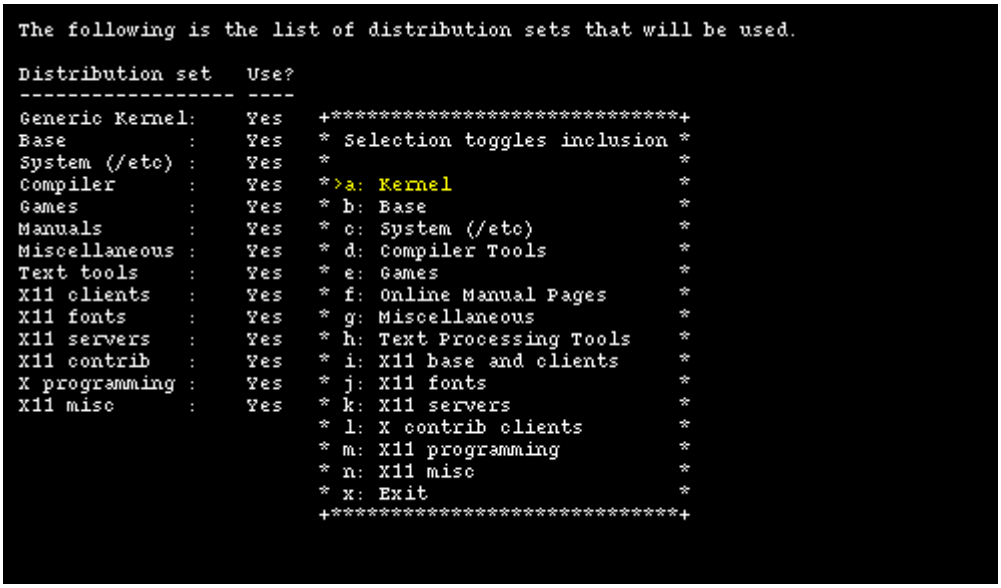
Sysinst wird jetzt die Partitionen und die Dateisysteme mit fdisk, newfs, fsck und installboot erstellen. Dann werden wir die NetBSD- Sets installieren.

nach oben

3.4.9. Auswahl des Installationsmediums

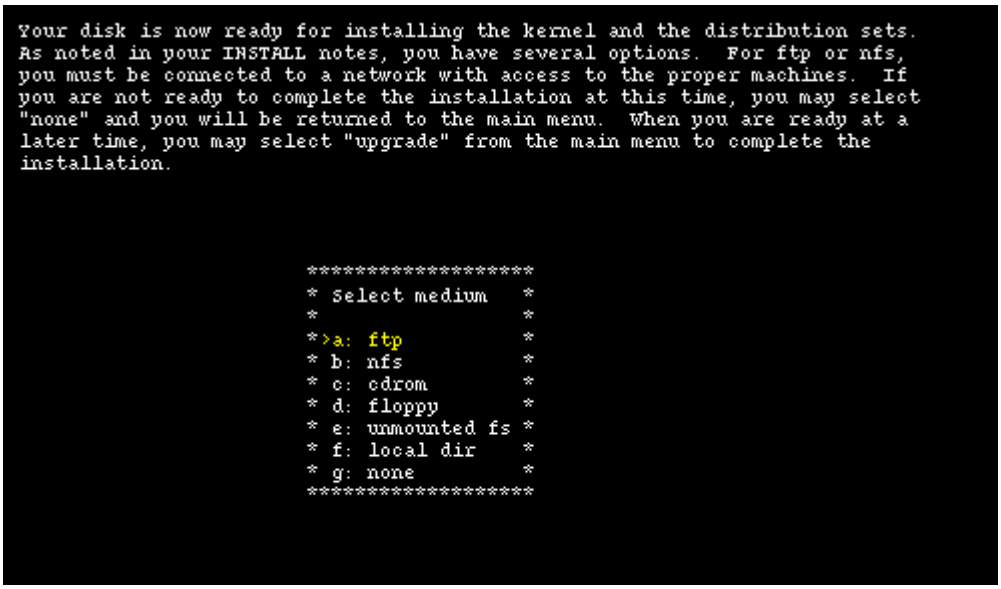
Mit dem ersten und schwierigstn Teil der installation bist Du nun fertig. Im nächsten Schritt wählst Du den Typ der Installation aus, der »full« sein kann und alle Sets installiert oder »custom«, wodurch Du die Wahl zwischen den angebotenen Sets hast. Wenn Du keinen Platzmangel auf der Platte hast, empfehle ich Ersteres. In diesem Beispiel wählen wir »custom«, nur um zu zeigen, wie das aussieht. Das bringt Dich nach Bild 3-19.

Bild 3-19: Auswahl der Sets



Die ersten drei Sets sind obligatorisch: Ohne sie funktioniert das System nicht. Du kannst Die Installation der einzelnen Sets mit Hilfe der einzelnen Optionen festlegen. Anfangs sind alle Pakete für die Installation ausgewählt, was dasselbe ist wie die vorhin beschriebene »full«-Option. Lass alles so stehen und gehe zur nächsten Option: »x:Exit«. sysinst fragt Dich jetzt, ob Du während der Extraktion der Sets die Dateinamen sehen willst. Jetzt muss sysinst die NetBSD- Sets finden (die ».tgz«- Dateien). Du musst sysinst diese Information zukommen lassen. Das Menü bietet verschieden Möglichkeiten:

Bild 3-20. Installationsmedien



Die einzelnen Optionen werden im »INSTALL«- Dokument detailliert beschrieben. Es ist auch möglich, von einem nicht gemoutetem Dateisystem zu installieren (vorausgesetzt, dass der Typ erkannt wird): Das heisst, dass es auch möglich ist, alle Sets auf eine MS-DOS- Partition zu kopieren und von da aus die Installation durchzuführen.

Bild 3-21. Installation von CD-ROM

Wenn Du »cdrom« anwählst, fragt sysinst nach dem Gerätenamen (z.B. »cd0«) und mountet es automatisch. Du solltest ausserdem den Pfadnamen ändern, wenn dieser vom Defaultwert abweicht. Beispiel: Wenn Die NetBSD- Distribution im »NetBSD-1.5«- Verzeichnis liegt, benutzt Du Option »b« wie hier:

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

Hinweis: Wenn Du keine US- Tastatur verwendest, solltest Du mit dem »/«- Zeichen vorsichtig sein... Siehe Sektion 3.3.1

Der CD-ROM- Gerätename: Wenn Du den Gerätenamen des Leufwerks nicht kennst, kannst Du ihn auf folgendem Weg herausfinden:

1. »CTRL-Z« drücken, um sysinst zu unterbrechen und zum Shellprompt zu kommen (nettes Feature!)
2. Gib dieses Kommando ein:

```
# cat /kern/msgbuf
```

Damit zeigst Du die Startup- Nachrichten des Kernels an, einschliesslich des CD-ROM- Gerätes (Beispiel: cd0)

3. Wenn das Display zu schnell scrollt, kannst Du den »ed«- Editor benutzen.

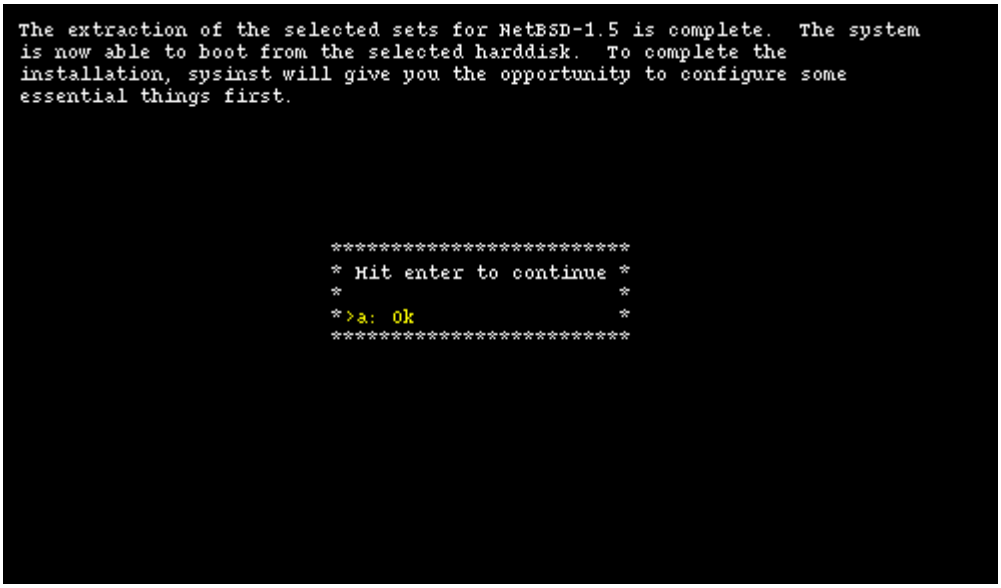
```
# ed /kern/msgbuf
```

4. Gehe mit diesem Kommando in das Installationsprogramm zurück:

```
# fg
```

Am Ende der Installation gibt sysinst eine Nachricht aus, dass alles gut gegangen ist. Mit Option »a:ok« werden die Device- Dateien erstellt.

Bild 3-22. Herzlichen Glühstrumpf!



Die Installation ist gelaufen. Mit sysinst kannst Du jetzt ein wenig Systemkonfiguration machen, bevor Du neu startest. Als erstes konfigurierst Du die Zeitzone und im nächsten Bildschirm erstellst Du das Passwort fü root. Jetzt ist es Zeit, das System neu zu starten. Wähle »a:ok« und gehe zurück in das Hauptmenü. Nimm die Bootdiskette aus dem Schlund Deines Laufwerks und Wähle Option »d: Reboot the computer«

nach oben

Inhalt

Kapitel 4: Der erste Start

Inhalt:

Nach der Installation führt der Computer einen Neustart von der Festplatte durch. Wenn alles gut gegangen ist, wirst Du innerhalb weniger Sekunden den Login- Prompt sehen (Das können je nach Hardware allerdings auch Minuten werden). Das System ist natürlich noch nicht konfiguriert, aber sei unbesorgt. Die Konfiguration von NetBSD ist eigentlich sehr einfach und keineswegs unangenehm. Stattdessen verhilft Dir diese manuelle Vorgehensweise zu einer vielleicht noch nie gekannten Flexibilität. Du wirst sehen, wie schnell alles geht; und im Laufe der Zeit wirst Du auch verstehen, wie das System funktioniert. Falls Probleme auftauchen sollten, wirst Du auch wissen, wo Du nach Fehlern suchen musst.

4.1. Falls etwas schiefging

Wenn das System nicht startet, kann es sein, dass der Bootmanager nicht korrekt installiert worden ist oder das es da einen Problem mit dem MBR (Master-Boot-Record) gibt. Starte die Maschine von der Bootdiskette, und wenn Du diese Meldung siehst...:

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

drücke die Leertaste während des fünf Sekunden dauernden Countdowns. Der Startvorgang wird unterbrochen und ein Prompt angezeigt. Du kannst hier auch eine rudimentäre Hilfe erhalten, wenn Du ein "?" oder "help" eingibst.

```
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
```

Das System sollte jetzt statt von der Diskette von der Festplatte booten. Wenn NetBSD sauber von der Platte hochgefahren werden kann, ist das möglicherweise ein MBR- Problem: Du kannst den Bootmanager installieren oder seine Konfiguration mit dem `fdisk -B` -Kommando ändern.

Seitenanfang

4.2. Der erste Login

Das erste Login wirst Du als Superuser **root** ausführen müssen. Er ist der einzige User, den es am Ende einer neuen Installation gibt. Gib das Passwort ein, das Du während der Installation definiert hast. Wenn Du noch kein Passwort hast, drücke einfach die Eingabetaste.

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

Seitenanfang

4.3. Ändern des Tastatulayouts

Die Tastaturbelegung ist immer noch die aus den USA. Wenn Du eine andere Tastatur hast, ist es besser, das Layout zu ändern, bevor Du anfängst, das System zu konfigurieren. Leute, die mit der deutschen Tastatubelegung schreiben, erledigen das so:

 # wsconsctl -k -w encoding=de

`encoding >de`sollte das System dann antworten.

Italiener machen das mit:

wsconsctl -k -w encoding=it

`encoding -> it` sollte dann die Antwort sein.

Eine vollständige Liste findet sich in `/sys/dev/wscons/wsksymdef.h`; aber die verbreitetsten sind diese hier:

- de
- dk
- fr
- it
- jp
- sv
- uk
- us

Diese Einstellung bleibt bis zum nächsten Reboot erhalten. Um das dauerhaft beizubehalten, füge das Kommando an das Ende der Datei `/etc/rc.local` ein. Beim nächsten Reboot wird die Tastatur automatisch konfiguriert.

```
# echo "wsconsctl -k -w encoding=it" >> /etc/rc.local
```

Achtung: Sei vorsichtig und gib wirklich zwei ">" - Zeichen ein. Wenn Du nur ein ">" eingibst, überschreibst Du die gesamte Datei, statt nur diese eine Zeile einzufügen !!!

Es gibt auch noch eine bessere Lösung, für dieses Problem: Du kannst einen neuen Kernel backen, der Deine Tastaturbelegung als Defaultwert verwendet. Wie das geht, steht im Kernel- Kapitel.

Seitenanfang

4.4. Das man- Kommando

Wenn Du noch nie ein Unixähnliches Betriebssystem benutzt hast, wird das **man**- Kommando Dein bester Freund werden. Damit werden die NetBSD- Manualseiten angezeigt, die als die besten und detailliertesten gelten, die es so gibt; sie sind allerdings sehr techniklastig.

man name zeigt die Manpage des **name**- Kommandos. **man -k name** zeigt eine Liste mit Manpages, die mit `name` im Zusammenhang stehen (Du kannst auch das **apropos**- Kommando nehmen).

Um die Grundlagen zu **man** zu lernen, gib mal folgendes ein:

`man man`

Das Manual ist in neun Sektionen unterteilt, die nicht nur grundsätzliche Informationen zu den Kommandos enthalten, sondern auch die Beschreibungen einiger NetBSD- Features und -Strukturen. Lege mal ein Auge auf die `hier(7)` -Seite, die das Layout des NetBSD- Dateisystems detailliert beschreibt.

man hier

Andere, ähnliche Seiten sind `release(7)` und `packages(7)`. Jede Sektion das Manuals besitzt eine sog. `intro`- Seite, die mit einem Kommando in dieser Art ausgeführt wird:

man 8 intro

Seitenanfang

4-1. Manual- Sektionen

1. grundlegende Kommandos, Tools und Utilities
2. Systemaufrufe und Fehlernummern
3. C- Bibliotheken
4. Spezielle Dateien und Hardwareunterstützung
5. Dateiformate
6. Spiele
7. Sonstige Informationen

8. Systempflege und Bedienungskommandos für root
9. Internes zum Kernel

Ein Aufruf kann auch in mehr als eine Sektion des Manuals passen. Um eine bestimmte Seite aufzurufen, kannst Du die Nummer der Sektion als Parameter im Kommando verwenden. Beispiel: `time` findet sich in Sektion 1 (als Benutzerkommando), in Sektion 3 (Die time- Funktion in der C-Bibliothek) und in Sektion 9 (die Systemvaraible). Um die Manpage für die C- Funktion `time` zu lesen, musst Du also das hier eingeben:

```
# man 3 time
```

Wenn Du alle vorhandenen Seiten lesen willst, kannst Du das mit:

```
# man -a time
```

Seitenanfang

4.5. Das root- Passwort ändern

Wenn Du während der Installation kein root- Passwort definiert hast (was in den pre-1.5- Systemen unmöglich war), ist es an der Zeit, das mit dem **passwd-** Kommando zu tun.

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

Das Passwort wird nicht auf dem Bildschirm angezeigt, während Du tippst. Später wirst Du lernen, wie man andere Benutzerkonten einrichtet.

Seitenanfang

4.6. Die Shell wechseln

Die Default- Shell für root ist die csh. Wenn Dir das nichts sagt, solltest Du die Manpage mit **man csh** durcharbeiten. Sie ist eine gute, interaktive Shell, allerdings besitzt sie keine History- Funktion (Schau Dir mal die tcsh, bash und gerade die /bin/sh für dieses Feature an). Wenn Du die Shell wechseln willst, mach das mit **chsh** . Die folgenden Shells stehen nach der Installation zur Auswahl:

- csh
- sh
- ksh

Die neue Shell wird beim nächsten Login gestartet. Im Moment kannst Du Dich mit diesem Kommando behelfen:

```
# set filec
```

Damit ist die automatische Vervollständigung von Dateinamen in der Kommandozeile eigeschaltet (mit der ESC- Taste; mit CTRL- D kommt eine Liste mit den möglichen Vervollständigungen).

Du kannst aber auch noch weitere Shells auf dem System installieren, wenn Dir danach ist: tcsh, bash, zsh und einige andere sind in der Package- Kollektion zu finden (was Du vielleicht später ausprobieren solltest).

Das ist eine günstige Zeit, die Initialisierungsdateien für die Shells zu erstellen (.cshrc, .login und die anderen).

Seitenanfang

4.7. Systemzeit

NetBSD benutzt, wie alle anderen Unix- Derivate auch eine Systemuhr, die auf der Greenwich- Zeit basiert. Das ist auch der Wert, auf den Du Deine Systemzeit einstellen solltest. Wenn Du, aus welchem Grund auch immer (beispielsweise, wenn Du noch ein Windows auf der Maschine installiert hast), musst Du das NetBSD mitteilen, indem Du die `rtc_offset`- Systemvariable änderst. Du kannst die Konfigurationsdatei des Kernels ändern und ihn neu kompilieren oder den vorhandenen Kernel direkt patchen (Die neue Zeit steht aber erst nach einem Reboot zur Verfügung). Das ist einfacher als es sich anhört. Hier ist ein Beispiel:

```
# 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
```

Der eingegebene Wert (-60) ist die Anzahl der Minuten, die sich Deine Maschine westlich von Greenwich befindet.

Um die derzeitige Einstellung der `rtc_offset`- Variable anzuzeigen gibst Du das hier ein:

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

Jetzt weiss der Kernel, wie er die Zeit auf der PC- Uhr übersetzen soll; aber Du musst das System immer noch für Deine lokale Zeit konfigurieren (Du findest sie unter `/usr/share/zoneinfo`.). Hier ist ein Beispiel für Leute, die in Italien leben, das aber in diesem Sinne auch In Deutschland oder Österreich anwendbar ist. Hier musst Du nur die Stadt an die Hauptstadt Deines Landes anpassen:

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

Wenn einmal alles richtig aufgesetzt ist, kannst Du die Zeit mit dem folgenden Komando verändern:

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

Seitenanfang

4.8. Basiskonfiguration in der /etc/rc.conf

NetBSD benutzt die `/etc/rc.conf` für die Systemkonfiguration beim Starten. Diese Datei legt fest, was gemacht wird, wenn das System hochfährt. Diese Datei zu kennen und zu verstehen, ist sehr wichtig!

Ab Version 1.5 von NetBSD wurde die Administration der rc.conf geändert. In früheren Versionen wurden alle Defaultwerte in `/etc/rc.conf` abgelegt und der User musste diese Datei direkt ändern. Ab Version 1.5 wurde die `/etc/defaults/rc.conf`- Datei eingeführt, in der die Defaultwerte stehen. Um einen Defaultwert zu ändern, muss der User den neuen Wert in `/etc/rc.conf` eintragen. Diese Definition wird höher gewichtet als der eingetragene Defaultwert in `/etc/defaults/rc.conf`, die unverändert bleibt. Zur Wiederholung: Das Verständnis dieser Datei ist wirklich sehr wichtig. In der Manpage gibt es eine genauere Beschreibung all dieser Optionen:

```
# man rc.conf
```

Die ersten Modifikationen sind diese hier:

- `rc_configured=YES` (Diese Modifikation wurde meistens schon vom Installationsprogramm durchgeführt)
- `wscons=YES`: Einschalten der virtuellen Konsolen
- `lpd=YES` aktiviert den Druckdienst
- Definiere einen Namen (`hostname`) für Deine Maschine (nimm einen qualifizierten Hostnamen). Wenn Deine Maschine ein Einzelgänger ist, kannst Du jeden Namen verwenden (Beispiel: `frufru.ruft.lifru`), der Dir da so einfällt. Wenn Deine Maschine an einem Netzwerk hängt, musst Du an dieser Stelle den korrekten Netzwerknamen eintragen.

  Statt einen Hostnamen einzutragen, kannst Du das auch in `/etc/myname` tun. Das Ergebnis ist dasselbe.

Seitenanfang

4.9. Die virtuellen Konsolen einschalten

Die hier beschriebenen Einstellungen werden unter Version 1.5 nicht mehr vorgenommen; ab dort sind sie Standard.

In der Datei `/etc/ttys` muss geprüft werden, ob ttyE0-ttyE3 eingeschaltet sind. ttyE4 kann für die Benutzung mit X ausgeschaltet bleiben. Beispiel:

```
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   on secure
```

```
ttyE4   "/usr/libexec/getty Pc"         vt220   off secure
```

Für die Mutigen: Wenn Du `/etc/ttys` erst einmal geändert hast, kannst Du eine schnelle Aktivierung mit diesen Kommandos ohne Reboot ausführen:

1. **`sh /etc/rc.wscons`**
2. **`kill -1 1`**

Einmal aktiviert, kannst Du mit der CTRL-ALT-Fn -Tastenkombination zwischen den Konsolen umschalten (Fn ist eine Funktionstase: F1 ist Konsole 1; F2 ist Konsole 2; ...).

Anmerkung: Man nehme zu diesem Zweck die Tasten auf der linken Seite der Tastatur.

Seitenanfang

4.10. Neustart des Systems

In dieser ersten Sitzung hast Du:

- die Tastatur konfiguriert
- das root- Passwort geändert
- die root- Shell gewechselt (optional)
- die Systemzeit und den RTC- Offset angepasst
- die Ortszeit definiert
- `/etc/rc.conf` konfiguriert
- die virtuellen Konsolen eingeschaltet

Jetzt ist es Zeit, mit dem folgenden Komando neu zu starten und ein Käffchen zu schlabbern:

```
# reboot
```

Seitenanfang

Inhaltsverzeichnis

Kapitel 5: Der zweite Start

...und das steht hier drin:

Während des ersten Startes hast Du eine Grundkonfiguration Deines System aufgesetzt. Dieses Kapitel beschreibt einige Kommandos und Operationen

5.1. dmesg

Beim Systemstart gibt der Kernel eine lange Reihe von Meldungen auf dem Monitor aus. Diese Meldungen informieren über den Kernelzustand (zum Beispiel vorhandener Speicher) und die Peripheriegeräte, die auf dem System gefunden wurden. Diese Informationen sind sehr wichtig, um die Hardware zu untersuchen, Konfigurationsprobleme zu finden und die Gerätenamen für die Peripheriegeräte herauszufinden (Du kannst so feststellen, ob Deine Netzwerkkarte als »ne0« oder »ne1« erkannt worden ist). Für gewöhnlich laufen diese Meldungen zu schnell auf, um sie mitlesen zu können. Um diese Meldungen nachzulesen, gibt es **dmesg**.

```
# dmesg | more
```

Wenn irgendwas in Deinem System nicht zu funktionieren scheint und Du in einer der NetBSD-Mailinglisten um Hilfe bittest, vergiss nicht, den relevanten System- Output in Deine Mail mit einzubauen. Ohne diese Informationen kann Dir niemand weiterhelfen.

> Seit NetBSD 1.4.2 legt der Bootprozess eine Kopie des dmesg- Outputs in »/var/run/dmesg.out« ab. Diese Funktion ist von grossem Nutzen, weil das System mit der Zeit alte dmesg- Meldungen »ausscrollt«

nach oben

5.2. CD-ROMs mounten

Neue User sind oft überrascht von dem Faktum, dass das Installationsprogramm die CD-Rom korrekt erkennt und mountet, aber das installierte System anscheinend »vergessen« hat, wie eine CD- Rom benutzt wird. Es ist kein besonderer Zauber mit der Benutzung einer CD verbunden: Du kannst sie wie jedes andere Dateisystem in Deinen Verzeichnisbaum einhängen. Alles, was Du kennen musst, ist der Name des Gerätes und einige Optionen zum »mount«- Kommando. Den Gerätenamen kannst Du mit dem »dmesg«. Kommando herausfinden. Als Beispiel; wenn dmesg das hier ausgibt:

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

ist der Gerätename cd0 und Du kannst die CD- Rom mit diesen Kommandos mount
```

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

Um die Dinge einfacher zu machen, kannst Du eine Zeile in »/etc/fstab« einfügen:

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

Ohne neu zu booten, kannst Du die CD-Rom so mounten:

```
# mount /cdrom
```

Wenn eine CD-Rom gemountet ist, kannst Du sie nicht von Hand auswerfen. Du musst sie immer unmounten, bevor Du das tust:

```
# umount /cdrom
```

Es gibt auch ein Kommando dafür:

```
# eject /dev/cd0a
```

nach oben

5.3. Die Floppy mounten

Um eine Diskette zu mounten, musst Du den Namen des Gerätes und das Dateisystem der Diskette kennen. Wenn Du auf eine Diskette im MS-DOS- Format zugreifen willst (lesend und schreibend), geht das so:

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

Anstelle von »/mnt« kannst Du auch jedes andere Verzeichnis verwenden; Du kannst auch ein »/floppy«- Verzeichnis erstellen, wie Du das für die Cd-Rom getan hast. Wenn Du oft mit MS-DOS-Disketten arbeitest, wirst Du das »mtools«- Paket installieren wollen, mit dem man auf eine Diskette oder Festplattenpartition dieser Art zugreifen kann, ohne dauernd mounten zu müssen. Es für den schnellen Kopiervorgang auf eine Diskette sehr praktisch.

nach oben

5.4. Zugriff auf eine DOS/Windows- Partition

Wenn sich NetBSD die Platte mit MS-SOS oder Windows teilt, kann das System so modifiziert werden, dass die DOS- Partitionen für NetBSD sichtbar sind. Zuerst musst Du die Geometrie für die M-DOS- Partition festlegen, eventuell mit **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>
```

Hinweis: Dieses Beispiel nutzt die wd0- Festplatte.

Der Output des fdisk- Kommandos sagt uns, dass die DOS- Partition bei Sektor 63 beginnt und eine Grösse von 2088516 Sektoren hat. Die NetBSD- Partition beginnt bei Sektor 2088579(2088579 = 2088516 + 63). Du benutzt diese Daten, um das BSD- Disklabel zu modifizieren: Alles, was Du tun musst, ist, eine Zeile hinzuzufügen, in der steht, wo sich diese Partition befindet und von welchem Typ diese MS-DOS- Partition ist, in dem Du einen der noch nicht genutzten Partitions- ID-Buchstaben dazu benutzt. Für diesen Zweck gibt es das Kommando **disklabel**. Wenn Du den -e Parameter an das Kommando dranhängst, benutzt das Programm Deinen Lieblingseditor ($EDITOR), um das Disklabel zu modifizieren. Beispiel:

```
# disklabel -e wd0
...
#        size   offset     fstype  [fsize bsize  cpg]
...
  e:  3450624  2831232    4.2BSD    1024  8192    16   # (Cyl.  2808* - 623
  f:  2088516       63    MSDOS
```

Die Partitionen von »a« bis »e« werden bereits von NetBSD benutzt. Die erste freie ID war »f«. Die Grössen- und die »offset«- Felder wurden mit den vorher berechneten Zahlen gefüllt. Als nächstes musst Du Dir einen Mountpoint verschaffen. Das hier bietet sich an:

```
# mkdir /msdos
```

Zu guter Letzt wird zu der »/etc/fstab« noch eine Zeile hinzugefügt.

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

Jetzt kann die Dos- Partition mit einem einfachen Kmmando gemountet werden:

```
# mount /msdos
```

Mit dieser Methode können FAT und FAT32- Partitionen gemountet werden. Wenn das automatisch passieren soll, musst Du nur die »noauto«- Otion aus der »/etc/fstab« entfernen.

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

nach oben

5.5. Neue Benutzer

Es ist an der Zeit, neue Benutzer in das System zu integrieren. Schon, weil Du sicher kein Risiko eingehen willst und Deine tägliche Arbeit schon deshalb als Root erledigen willst. NetBSD hat kein Programm, um neue User zu erschaffen; stattdessen solltest Du Dir mal die *adduser*- Manpage zu Gemüte führen.

```
# man adduser
```

Wenn Du Dich nach den Instruktionen in dieser Manpage richtest, wirst Du beginnen, **vipw** zu benutzen, das das Standard- Administrationstool für neue Benutzerkonten unter NetBSD ist.

Hinweis: Mit NetBSD kommt ein ganzer Satz dieser Werkzeuge; ein **useradd**- Kommando und auch andere. Um einen neuen User anzumelden, musst Du das hier eingeben:

```
# useradd -m joe
```

Die Standardeinstellungen für das **useradd**- Kommando können geändert werden. Wie das geht, steht in der useradd(8) Manpage.

Wenn Du eine frühere Version von NetBSD hast und Du neue Konten nicht manuell einrichten willst, kannst Du dafür ein Paket installieren; beispielsweise addnerd aus der Packages- Sammlung. Ich empfehle allerdings dringend, mal einen Blick auf die Manpage zu werfen und mindestens ein Benutzerkonto manuell einzurichten.

nach oben

5.6. Shadow- Passwörter

Shadow- Passwörter werden bei NetBSD als Standard benutzt und können nicht abgeschaltet werden. Alle Passwörter in »/etc/passwd« werden durch ein «#« ersetzt. Die verschlüsselten Passwörter liegen in einer anderen Datei, »/etc/master.passwd; die nur von root gelesen werden kann. Wenn Du **vipw** startest, um diese zu editieren, öffnet das Programm eine Kopie von /etc/master.passwd; wenn Du das Programm beendest, prüft vipw, ob die Kopie gültig ist, erstellt eine neue /etc/paswd und installiert eine neue /etc/master/passwd- Datei. Am Schluss ruft vipw pwd_mkdb auf, das eine neue /etc/pwd.db - und eine /etc/spwd.db- Datei erstellt. Das sind zwei Datenbanken, die /etc/passwd und /etc/master.password entsprechen, die aber schneller zu verarbeiten sind.

Wie Du siehst, werden Passwörter von NetBSD automatisch gehandhabt. Wenn Du vipw benutzt, um die passwd- Datei zu bearbeiten, brauchst Du keine spezielle Administrationsprozedur durchzuführen.

Es ist sehr wichtig, vipw und die andern Tools *immer* zu benutzen, um ein Benutzerkonto zu verwalten (chfn,shsh,chpass,passwd) und niemals /etc/master.passwd direkt zu ändern.

nach oben

5.7. System anhalten und neu starten

Das Kommando, das benutzt wird, um das System zu stoppen oder neu zu starten ist **shutdown**.

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

Zwei andere Kommandos bewirken dasselbe:

```
# halt
# reboot
```

halt/reboot und **shutdown**sind keine Synonyme. Letzteres ist etwas weiter entwickelt. Auf einem Mehrbenutzersystem solltest Du wirklich shutdown benutzen: Du kannst gleichzeitig das System anhalten, die Mitbenutzter informieren und noch einiges Andere. Mehr Informationen steht in den Manpages shutdown(8), halt(8) und reboot (8).

nach oben

Inhalt

Kapitel 6: Drucken

Inhalt:

Dieses Kapitel beschreibt eine einfache Konfiguration für das Drucken auf einen HP Deskjet 690c, der an den ersten Parallelport des Rechners angeschlossen ist als Beispiel. Als Erstes konfigurieren wir das System, um Textdokumente auszudrucken. Als Nächstes erweitern wir die Konfiguration, um PostScript- Dokumente mit dem Ghostscript- Programm ausdrucken zu können.

6.1. Einschalten des Druckdaemons

Nach einer frischen Installation ist es noch nicht möglich, sofort zu drucken, weil der »lpd«- Druck-Spooling Daemon nocht nicht eingeschaltet ist. Um das zu tun, muss eine Zeile in »/etc/rc.conf von...

```
lpd=NO
```

nach

```
lpd=YES
```

Geändert werden.

Diese Änderung wirkt sich erst nach dem nächsten Neustart des Systems aus; aber der Daemon kann für jetzt auch manuell gestartet werden:

```
# lpd -s
```

Um zu kontrollieren, ob »lpd« aktiv ist, gibt es dieses Kommando:

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

Wenn Du keine Eintragung für lpd im Output dieses Kommando findest, ist der Daemon nicht aktiv.

Bevor Du »/etc/printcap« konfigurierst, ist es besser, einen Testlauf zu fahren, um zu testen, ob es eine funktionierende Verbindung zu Drucker gibt. Beispiel:

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

Um zu sehen, wie der Output aussehen sollte, versuche mal dasselbe Kommando ohne Umleitung der Ausgabe auf den Drucker:

```
# lptest 20 10
```

Ein häufiger auftretendes Problem ist, dass die Linien im Output zum Drucker nicht korrekt in ihren Spalten stehen (Staircase- Problem), sondern abgestuft dargestellt werden. Das heisst gewöhnlich, dass der Drucker so konfiguriert ist, dass er mit einer neuen Zeile anfängt, wenn er ein <cr>(carriage return, ASCII 13) und ein <LF> (line feed, ASCII 10)- Zeichen empfangen hat, in einer neuen Zeile weiterdruckt. Das Problem lässt sich aber beheben:

- Änderung der Druckerkonfiguration
- Benutzung eines einfachen Druckfilters (kommen wir später zu)

  Wichtig: Am vorigen Beispiel ist der lpd- Spooler nicht beteiligt, weil wir den Programmoutput nicht zwischengespeichert, sondern direkt an das Druckdevice (/dev/lpt0) geschickt haben.

nach oben

6.2. Konfiguration von »/etc/printcap«

Diese Sektion erklärt, wie der Beispieldrucker konfiguriert wird, um Texte damit zu drucken.

Der Drucker muss in »/etc/printcap« eingetragen sein. Der Eintrag beinhaltet die ID der Druckers (seinen Namen) und seine Beschreibung. Die »lp«- ID wird von vielen Programmen als Defaultwert verwendet.

Beispiel 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:
```

Das Dateiformat und die Optionen werden im Detail in »printcap(5)« beschrieben. Du solltest wissen, dass wir gerade einen Input- Filter spezifiziert haben (mit der »if«- Option), der das Staircase-Problem nicht beseitigt:

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

Drucktreiber und HP- Drucker:
Beispiel 6-1:
anstelle des »lpd#«(Treiber via Interrupt) benutzt das »lpa#«- Device (Polled Printer) für den Drucker. Die Benutzung von Interrupts sorgt bei einigen Druckern für Kommunikationsprobleme. Der HP Deskjet 690c ist einer davon: Das Drucken geht extrem langsam und bei einer PostScript- Seite kann das schon mal ein paar Stündchen dauern, bis der Apparat seine Arbeit fertig hat. Gelöst wird das Problem mit dem »lpa«-Treiber. Du kannst aber auch einen Kernel kompilieren, in dem lpd immer »polled« läuft.

Der Eintrag in die printcap bescreibt ausserdem wo das Spoolverzeichnis ist, das erstellt werden muss. Dieses Verzeichnis benutzt lpd, um die Daten zu sammeln, die gedruckt werden sollen.

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

Der einzige Teil, der fehlt, ist der »lpfilter«, der geschrieben werden muss. Das einzige, was dieser Filter tut, ist, dass er den Drucker so konfiguriert, dass das Staircase- Problem beseitigt wird, bevor der Text zum Drucken abgeschickt wird. Der in unserem Beispiel benutzte Drucker benötigt diesen String, um initialisiert zu werden: »ESC &k2G«. **Beispiel 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*
```

> **Hinweis:** Es gibt da noch einen anderen Filter, der benutzt werden kann:
>
> ```
> :if=/usr/libexec/lpr/lpf:
> ```
>
> Dieser Filter ist weitaus komplexer, als der eben gezeigte. Er wurde geschrieben, um den Output von »nroff« zu verarbeiten und händelt Unterstreichen und Überschreiben, erweitert die Tabulatoren und konvertiert LF nach CR + LF. Die Quelle dieses Filterprogramms findet sich in /usr/src/usr.sbin/lpr/filters/lpf.c.

»lptest« kann jetzt noch einmal ausgeführt werden. Diesmal benutzt er den lpd- Spooler.

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

Das »lpr«- Programm druckt den Text in den Spooler und schickt die Daten an den Drucker. Mit der »-h«- Option wird der Druck einer Bannerseite ausgeschaltet (Wegen der »sh«- Option in der »/etc/printcap« aber nicht lebensnotwendig).

Du kannst das Staircase- Problem mit Hilfe verschiedener Tools und Methoden lösen, beispielsweise mit C- Programmen. Die präsentierte Lösung hat den Vorteil., dass sie sehr einfach ist.

nach oben

6.3. Die Konfiguration von Ghostscript

Jetzt funktioniert das einfache Drucken. Die Funktionalität des Druckens von Postscript- Dateien kann jetzt dazugefügt werden. Der einfache Drucker in diesem Beispiel unterstützt das Drucken von nativen Postscript- Dateien nicht. Ein Programm, das ein Postscript- Dokument in ein für den Drucker verständliches Format konvertiert, muss her. Das Ghostscript- Programm aus der Packages- Kollektion kann dafür verwendet werden (Siehe Kapitel 8). Diese Sektion erklärt, wie Ghostscript konfiguriert werden muss, um Postscript- Dateien auf dem HP Deskjet 690c auszudrucken.

Eine zweite ID für den Drucker wird in »/etc/printcap« erstellt: Diese neue ID benutzt einen anderen Inputfilter, der Ghostscript aufruft, um den Ausdruck eines gerade aktuellen Postscript- Dokumentes

durchzuführen. Danach werden Textdokumente aus dem »lp«- Drucker ausgegeen und Postscript-Dokumente auf dem »ps«- Drucker: Beise Einträge benutzen denselben physikalischen Drucker; aber zwei verschiedene Filter.

Das gleiche Ergebnis kann auch mit anderen Konfigurationen erreicht werden. Als Beispiel kann ein einzelner Eintrag mit einem Filter verwendet werden: Der Filter sollte automatisch das Format der Dokumente erkennen und das geeignete Druckprogramm verwenden. Dieser Weg ist einfacher, aber er führt zu einem komplexeren Filter. Wenn Du willst, solltest Du Dir überlegen, ob Du nicht das »magicfilter«- Programm aus der Packages. Kollektion installierst: Es tut dies und viele andere Dinge automatisch.

Die neue /etc/printcap sieht so aus:

Beispiel 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:
```

Option »mx#0« ist für den Ausdruck von Postscript- Dateien sehr wichtig, weil sie die Grössenrestriktionen der Input- Datei eliminiert. Postscript- Dokumente neigen dazu, sehr gross zu sein. Die »if«- Option zeigt auf den neuen Filter. Es gibt ausserdem ein neues Spool- Verzeichnis.

Der letzte Schritt ist das Erstellen des neuen Spoolverzeichnisses und des Filterprogramms.

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

Das Filterprogramm für den Postscript- Output ist komplexer als das für die Texte: Mit der Datei, die gedruckt werden soll, wird der Interpreter gefüttert, der in dieser Runde an der Drucker eine Reihe von Kommandos in der Sprache des Druckers schichkt. Wir haben es erreicht, mit Hilfe der Tugenden von NetBSD und ein paar leistungsfähigen Freeware- Paketen, einen billigen Farbdrucker in ein Gerät zu verwandeln, das für den Postscript- Output geeignet ist. Die Optionen, die wir benutzt haben, um Ghostscript zu konfigurieren werden in der Dokumentation beschrieben: »cdj550« ist das Device, mit dem wir den HP- Drucker steuern.

nach oben

Beispiel 6-4. »/usr/local/libexec/lpfilter-ps«

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

```
exit 2
```

Zusammenfassung: Wir haben zwei verschiedene Druckernamen auf dem System geschaffen, die beide auf denselben Drucker zeigen, aber verschiedene Optionen verwenden, verschiedene Filter und verschiedene Spoolverzeichnisse. Texte und Postscript- Dokumente können gedruckt werden. Um Postscript- Dokumente zu drucken, muss Ghostscript auf dem System installiert sein.

nach oben

6.4. Druckanagement- Kommandos

In dieser Sektion werden einige nützliche NetBSD- Kommandos für die Drucker- und Druckjobadministration gelistet. Bisher ging es nur um die »lpr«- und »lpd«- Kommandos. Es gibt aber auch noch ein paar andere:

lpq

: prüft die Schleife mit den Druckjobs und gibt ihren Inhalt aus

lprm

: löscht Druckjobs aus der Warteschleife

lpc

: prüft das Drucksystem, schaltet ganze Drucker oder bestimmte Features ein oder aus.

nach oben

6.5. Drucken im Netz

Es ist auch möglich, das Drucksystem so zu konfigurieren, dass ein Ausdruck auch auf einem anderen, entfernten System möglich ist. Sagen wir, dass Du auf dem Host »wotan« arbeitest und auf »loge«, also einem anderen Rechner, drucken willst. Die »/etc/printcap« aud »loge« ist die aus Beispiel 6-3. Es ist möglich, von »Wotan« aus mittels dem Ghostscript auf »loge« in Postscript zu drucken.

Der erste Schritt ist, loge für das Ausdrucken von Druckjobs, die von Wotan kommen, vorzubereiten. Das wird getan, indem eine Zeile mit dem Namen Wotans in der »/etc/hosts.lps«- Datei auf loge eingetragen wird. Das Dateiformat ist sehr einfach: In jeder Zeile steht ein Hostname, der drucken darf.

Als nächsts muss die »/etc/printcap« für das Senden von Druckjobs an loge konfigurert werden. Beispiel:

```
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
```

Es gibt vier Hauptunterschiede in dieser Konfiguration und der in Beispiel 6-3:

1. Die Definition von »lp« ist leer.
2. Der »rm«- Eintrag definiert den Namen des Rechners, an den der Drucker angeschlossen ist.
3. Der »rp«- Eintrag definiert den Namen des Druckers, der an diesen Rechner angeschlossen ist.
4. Es ist nicht notwendig, Input- Filter zu spezifizieren, weil die Definitionen auf loge verwendet werden.

Jetzt werden die Druckjobs für »lp« und »ps« von Wotan automatisch an den Drucker, der an loge angeschlossen ist, gesendet.

nach oben

Inhaltsverzeichnis

Kompilieren und Installieren eines neuen Kernels

Inhalt

Die meisten NetBSD- User werden früher oder später einen an die eigenen Bedürfnisse angepassten Kernel kompilieren wollen oder müssen. Das verschafft ihnen allerdings auch einige Vorteile:

- Du kannst dir Grösse Deines Kernels dramatisch reduzieren und dadurch natürlich auch den Speicherbedarf verkleinern. Das Erzeugen eines eigenen Kernels in NetBSD 1.5 reduziert die Grösse in den meisten Fällen auf weniger als die Hälfte gegenüber dem Original- Kernel, der immerhin 4.7 MB gross ist.
- Die Systemleistung kann dadurch gesteigert werden.
- Du kannst Dein System vernünftig einstellen.
- Du kannst eventuelle Probleme mit der Hardwareerkennung und/oder Konflikte bei den Peripheriegeräten lösen.
- Du kannst einige Optionen an Deine Bedürfnisse anpassen (Beispielsweise die Tastatur, Anpassen der Uhrzeit und anderes)
- Du kannst mit Deinem System vertrauter werden.

nach oben

1. Installation der Kernelquellen

Du bekommst die Kernelquellen bei der selben Stelle, von der Dein installiertes System stammt (beispielweise ftp://ftp.netbsd.org.). Denke aber daran, dass Du die zu Deiner Release passenden Quellen nimmst.

Sysinst, das Installationsprogramm, kopiert die Kernelquellen nicht mit auf die Festplatte; deshalb müssen sie von Hand extrahiert und nachinstalliert werden. Das Archiv mit den Kernelquellen liegt im »/source/sets«- Verzeichnis und heisst »syssrc.tgz«.

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


Übe Dich in Geduld: Diese Operation kann
ihre Zeit dauern, weil das Archiv einige Hundert Dateien enthält.
Die Quellen wohnen jetzt in »/usr/src/sys«; der symbolische Link »sys«
```

```
verweist in dieses Verzeichnis. Daher haben folgende Kommandos exakt
denselben Effekt:
 # cd /usr/src/sys
# cd /sys
```

Nach dem Auspacken der Kernelquellen kannst Du Plattenplatz sparen, indem Du die Verzeichnisse für die Architekturen entfernst, die Du nicht brauchst. Wechsle einfach in das »sys/arch«- Verzeichnis und lösche die jeweiligen, nicht benötigten Verzeichnisse. Für die i386er- Version brauchst Du auch nur das »i386«- Verzeichnis. Wenn die Quellen erst einmal installiert sind, kannst Du Deinen eigenen Kernel backen; Das ist nicht sooo schwierig, wie es vielleicht aussieht. Ehrlich: Ein Kernel kann in vier bis fünf Minuten fertig sein, wie es in den folgenden Kapiteln beschrieben wird.

nach oben

2. Italienisches Keyboard- Layout

Bevor der Kernel kompiliert wird, sollten italienische Benutzer die in der Quelldatei vordefinierte Tastaturbelegung ändern. /sys/dev/pckbc/wskbdmap_mfii.c heisst die Datei. Im Standardlayout fehlen einige für Programmierer nützliche Zeichen (beispielsweise die geschweiften Klammern und und die Tilde). Dieses hier ist ein alternatives Layout:

```
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,
...
```

Dieses Layout definiert die folgende Belegung:

Tasten	Zeichen
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 + +	}

nach oben

Konsolentreiber:

Seit Version 1.4 benutzt NetBSD den Wscons- Multiplattformtreiber, um den Bildschirm, die Tastatur und die Maus zu steuern. Ältere Versionen arbeiteten mit pccons oder pcvt.

3. Den neuen Kernel kompilieren

Um den neuen Kernel kompilieren zu können, muss das Compiler- Set installiert sein (comp.tgz).

Die grundsätzlichen Schritte für eine Kernel- Kompilierung:

1. Erstellen oder modifizieren der Konfigurationsdatei
2. Konfigurieren des Kernels
3. Abhängigkeiten erzeugen
4. den Kernel »backen«
5. ...Und diesen installieren.

nach oben

4. Die Kreation einer Kernel- Konfigurationsdatei

> **Anmerkung:** Die beschriebenen Verzeichnisse beziehen sich auf die i386- Ausgabe. Benutzer anderer Architekturen müssen diese Namen durch die passenden ersetzen (normalerweise Unterverzeichnisse von »arch«).

Die Kernel- Konfigurationsdatei definiert neben bestimmten Optionen besonders die Arten, die Anzahl und Eigenschaften der Gräte, die vom Kernel unterstützt werden sollen. Die Konfigurationsdateien liegen im »/sys/arch/i386/conf«- Verzeichnis. Der einfachste Weg, eine neue Datei zu erzeugen ist der, eine davon zu nehmen, sie zu kopieren und anschliessend zu modifizieren. Auf den meisten Plattformen sollte die GENERIC- Konfiguration die passende sein. In dieser Konfigurationsdatei sind Kommentare eingefügt, die die Optionen beschreiben. Eine genauere Beschreibung findet sich in der »options(4)«- Manpage.

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

> **Kernelnamen:** Die Namen der Konfigurationsdateien werden traditionell in Gossbuchstaben geschrieben.

Um eine solche Konfigurationsdatei anzupassen, sind grundsätzlich diese drei Schritte nötig:

1. Unterstützung für Hardware, die unterstützt werden soll oder eben nicht (Beispiel: Die SCSI-Unterstützung kann man entfernen, wenn man sie nicht braucht).
2. Unterstützung für bestimmte Features ein- oder abschalten (Beispiel: NFS- Client-

Unterstützung, Linux- Kompatibilität undsoweiter...)

3. Einstellen der Kernelparameter

Zeilen, die mit einem »#« beginnen, sind Kommentare. Zeilen bzw. Optionen werden eingeschaltet, wenn man den Hash entfernt und ausgeschaltet, wenn man den Hash vor die jeweilige Zeile setzt. Es ist besser, Zeilen nur auszukommentieren, anstatt sie zu löschen. Es ist dann immer möglich, sie wieder zu aktivieren. Wenn das nötig sein sollte.

Der Output des **dmesg**- Kommandos kann benutzt werden, um herauszufinden, welche Zeilen auskommentiert werden können. Für jede Zeile dieses Typs:

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

müssen XXX und YYY in der Konfigurationsdatei aktiviert sein. Du wirst möglicherweise ein klein wenig experimentieren müssen, bevor sich eine Minimalkonfiguration herauskristallisiert. Aber auf einem Desktop kannst Du die Kernelgrösse allein durch den Verzicht auf SCSI und PCMCIA schon halbieren.

Du solltest ausserdem sämtliche Optionen in der Konfigurationsdatei überprüfen und alles auskommentieren, was Du nicht brauchst. Jede Option ist mit einem kurzen Kommentar versehen, der normalerweise ausreichen sollte, um zu verstehen, was die Option bedeutet. Viele Optionen werden in der »options(4)«- Manpage genauer beschrieben. Wenn Du schon mal dabei bist, solltest Du die Optionen für »Dein« Tastaturlayout und die lokale Zeit der CMOS- Uhr anpassen. Hier ein Beispiel für Deutschland/Österreich:

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

Das »adjustkernel«- Perlscript, das man unter
http://www.feyrer.de/Misc/adjustkernel finden kann, untersucht den Output
von **dmesg** und generiert automatisch
eine minimale Konfigurationsdatei. Um diese nutzen zu können, muss Perl
auf Deinem System installiert sein. Die Installation neuer Software ist in
Kapitel 8 des NetBSD- Handbuches genauer beschrieben. Wenn Du das installieren
willst, sauge Dir das vorkompilierte »perl-5.00404.tgz«- Paket und
befiehl Deinem Rechner folgendes: # **pkg_add**
perl-5.00404.tgz

Jetzt ist Perl installiert, konfiguriert und fertig zum Gebrauch: War wohl mal wieder leichter als gedacht...

Du kannst das Script mit diesen Kommandos ausführen:

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

Ich habe das Script ausprobiert und es funktionierte recht gut;
schon weil es mir eine Menge manueller Tipparbeit erspart hat. Denke aber daran,
dass das Script nur die vorhandenen Geräte konfiguriert. Alle anderen
Optionen müssen immer noch manuell konfiguriert werden (z.B. Linux-
Emulation, ...).

nach oben

5. Kernelkonfiguration

Wenn Du mit der Modifikation Deiner Konfigurationsdatei fertig bist (die wir mal MYKERNEL nennen), gibst Du folgendes Kommando an der Konsole ein:

```
# config MYKERNEL
```

Wenn MYKERNEL fehlerfrei ist, erstellt das **config**- Programm die nötigen Dateien für die Kompilierung des Kernels. Ansonsten ist es notwendig, die Fehler zu korrigieren und das ganze noch einmal zu versuchen (Spiel das Lied noch einmal, Sam...).

6. Abhängigkeiten erzeugen und neu kompilieren

Das Erzeugen der Abhängigkeiten und das Kompilieren des Kernels wird mit diesen beiden Kommandos durchgeführt:

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

Es kann passieren, dass die Kompilierung mit einer Fehlermeldung abgebrochen wird. Das kann eine ganze Reihe von Gründen, die nicht immer von **config** verursacht werden. Manchmal wird dieser Fehler durch ein Hardwareproblem hervorgerufen (Oft sind defekte RAMChips die Übeltäter): Die Kompilation erzeugt eine höhere Systemlast, als es die meisten Anwendungen tun. Ein anderer Fehler liegt hierin: Option B ist aktiv; verlangt aber nach option A, die nicht aktiviert wurde.

Eine komplette Kompilierung braucht, abhängig von der Hardware, ein paar Minuten bis ein paar Stunden. Die Tabelle zeigt ein paar Beispiele, die Du Dir mal reintun solltest:

CPU	RAM (MB)	Durchschnittl. Zeit
486 DX2 50	20	1 Stunde
P166	96	15 Minuten
PIII	128	5 Minuten
68030/25	8	4 Stunden

Der Output des **make**- Kommandos ist die »netbsd«- Datei im »compile«- Verzeichnis: Diese Datei muss nach der Sicherung der alten Version in das Wurzelverzeichnis kopiert werden:

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

Die Anpassung des Kernels wird meistens seine Grösse reduzieren. In folgenden Beispiel ist »netbsd.old« der Standard- Kernel und »netbsd« die neue Version:

```
-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
```

Mit einem Neustart wird der neue Kernel aktiviert:

```
# reboot
```

nach oben

7. Wenn etwas schiefgelaufen ist

Wenn der PC neu gestartet wird, kann es passieren, dass der neue Kernel nicht läuft wie erwartet oder überhaupt nicht bootet. Kein Grund zu Sorge: Sollte das passieren, starte das System einfach mit dem alten Kernel und entferne den neuen (Es ist besser, den Neustart im »single-user«- Modus auszuführen):

- Starte die Maschine neu
- Drücke die Leertaste am Bootprompt während des 5-Sekunden- Countdowns. Folgende Meldung erscheint:

  ```
  boot:
  ```

- Gib das hier ein:

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

- Führe nun diese Kommandos aus, um den alten Zustand wieder herzustellen und den alten Kernel wieder zu aktivieren:

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

nach oben

Inhaltsverzeichnis

Kapitel 8: Die Package- Collection

Inhalt:

Die NetBSD- Package-Collection ist eine Werkzeugsammlung, die die Kompilierung und Installation der riesigen Mengen an freier Software für Unix- Systeme sehr vereinfacht. Man muss nur ein oder zwei Kommando kennen, um ein nahezu perfekt konfiguriertes und funktionierendes Paket auf der Maschine zu haben.

Der erste Kontakt mit dem Package-System von NetBSD kann für eine gewisse Verwirrung sorgen: Es gibt manchmal mehrere Kommandos, die dasselbe bewirken. Die Frage, die sich dabei stellt, ist eigentlich einfach zu beantworten: Es gibt *zwei Wege*, ein Programm zu installieren. Du kannst:

- Ein neues Paket aus den Quellen auf Deinem System kompilieren. Wenn man die Package-Collection benutzt, die automatisch die Quellen aus dem Internet herunterlädt, kompiliert, installiert und konfiguriert; geschieht das mit ganzen zwei Kommandos. Die Package-Collection beinhaltet ein Set mit Makefiles und Konfigurationsdateien, die die Standard- Unixtools verwenden, die mit dem Basissystem installiert werden. Ein weiteres Feature des Package-Systems ist die automatische Prüfung des Vorhandenseins bestimmter Pakete, die vielleicht für ein neues Paket nötig sind. Diese Pakete können ebenfalls gleich mit heruntergeladen und installiert werden. Die Package-Collection wird nicht automatisch mit dem Basissystem installiert, weil sie ständig überarbeitet und fast wöchentlich in einer neuen Version veröffentlicht wird. In der folgenden Sektion wird erklärt, wie man sie downloadet und auf dem System installiert. Auf der NetBSD- Site gibt es eine sehr genaue technische Beschreibung dieses Verfahrens.

- Die Installation einer vorkompilierten und -konfigurierten Version des Programms. Das wird mit den *pkgtools* gemacht, die ebenfalls mit dem Basissystem installiert werden. Diese Methode ist schneller, aber nicht so flexibel wie die vorherige (Beispiel: Du kannst die Compile- Time-Optionen nicht konfigurieren). Die *pkgtools* werden auch für das Management der installierten Programm (egal ob aus den Quellen oder vorkompiliert) verwendet, die in einer Datenbank eingetragen werden. Du kannst, um ein Beispiel zu nennen, die installierten Pakete auflisten, ein Paket löschen usw.

  Wenn Du nur vorkompilierte Pakete installieren willst, brauchst die Package-Collection nicht dafür.

Die beiden erwähnten Methoden setzen voraus, dass schon irgendwer ein »Paket erschaffen« hat; also ein Paket nach NetBSD portiert dafür konfiguriert hat. Momentan bietet die Package-Collection weit mehr als 1000 Programme. Es ist aber trotzdem möglich, dass das von Dir gesuchte Paket noch nicht dabei ist. In so einem Fall kannst Du das Paket ohne das Managementsystem kompilieren und benutzen; und, wenn es läuft, ein Paket herstellen, das in die Kollektion integriert werden kann. Andere User werden auch davon profitieren; genau so, wie Du es gerade tust und der Übersetzer es auch schon getan hat.

nach oben

8.1. Installation der Package-Collection

Bevor Du ein Paket aus den Quellen installierst, solltest Du die Package-Collection von der NetBSD-Site oder einer ihrer Mirrors Deiner Wahl herunterladen und installieren. Das wird in den folgenden Schritten beschrieben.

1. Ziehe Dir die neueste Version der Package-System- Quellen, in der alle nötigen Makefiles und konfigurationsdateien enthalten sind. Das gibt es unter der Adresse: ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/tar_files/. Die Datei, die gezogen werden muss ist `pkgsrc.tar.gz`

2. Wenn Du eine Collection auf dem System installiert hast, muss sie mit diesem Kommando entfernt werden:

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

3. Installiere die neue Collection, die Du gezogen hast:

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

   Die Ausführung dieses Kommandos kann eine ganze Zeit dauern, weil eine grosse Anzahl kleiner Pakete extrahiert wird. Am Ende steht das fertige Gerüst für die Installation neuer Programme. Du kannst jetzt damit anfangen, sie zu installieren.

   > **Wichtig:** Bis jetzt ist möglicherweise klar, dass Du mit den vorigen Kommandos die Konfigurationsdateien für die automatische Installation von Programmen auf Deinem System installiert hast. Du hast noch keine Programme installiert. Im Grunde genommen hast Du jetzt eine Liste mit den verfügbaren Paketen und die Anleitungen, sie zu besorgen, zu kompilieren und zu installieren. Nicht mehr; aber auch nicht weniger.

Wenn Du die Package-Collection installiert hast, kannst Du sie mit Lynx oder Netscape da durchsuchen und Dir die Details und Beschreibungen aller greifbaren Pakete und -Kategorien durchlesen. Beispiel für Lynx:

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

Pass auf, dass Du die distfiles nicht verlierst!

Wenn Du eine existierende Package-Collection löschst, um eine neuere Version zu installieren, vergiss nicht, Dein /usr/pkgsrc/distfiles-Unterverzeichnis zu sichern; ansonsten wirst Du alle Tarbälle, die Du heruntergeladen hast, verlieren. Wenn Du das nicht riskieren willst, kannst Du das System anweisen, ein anderes Verzeichnis für die Distfiles zu benutzen; eines, das kein Unterverzeichnis von /usr/pkgsrc ist. Beispiel für ein neues Verzeichnis:

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

Ergänze /etc/mk.conf mit dieser Zeile:

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

Du kannst natürlich ein Verzeichnis Deiner Wahl verwenden. /usr/pkgsrc_distfiles ist nur ein Beispiel.

Du kannst viele Aspekte des Package-Systems in /etc/mk.confkonfigurieren. Ein detailliertes Beispiel findest Du in

/usr/pkgsrc/mk/mk.conf.example.

nach oben

8.2. Update der Package-Collection

Die Package-Collection wird oft überarbeitet: Du kannst fast jede Woche eine neue Version davon auf der FTP-Site finden. Um ein Update durchzuführen, musst Du nach dem gleichen Schema vorgehen wie bei der beschriebenen Erstinstallation.

Manchmal, wenn Du ein Update der Package-Collection durchführst, wird es nötig sein, für das »pkgtools«- Utility ebenfalls ein Update einzuspielen. Du wirst schon herausbekommen, ob das nötig wird: Genau dann, wenn Du versuchst, ein Programm aus dem Package-System zu installieren und das Package-System Dir höflich mitteilt, dass Deine pkgtools veraltet sind.

```
# 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
```

Der einfachste Weg zu einem Update ist:

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

Danach kannst Du die Installation des Paketes, das sich beschwert hat, noch einmal durchführen.

> **Hinweis:** Die geforderte Version kannst Du herausfinden, wenn Du pkgsrc/mk/bsd.pkg.mk nach der geforderten Version untersuchst. Suche eine Zeile, die aussieht wie diese hier:
>
> ```
> PKGTOOLS_REQD = 20000202
> ```
>
> (Das 20000202-Datum ist nur ein Beispiel) Das bedeutet, dass die benötigte Version der Programme in pkg_install-20000202.tar.gz zu suchen ist, die Du auf der NetBSD-FTP-Site unter packages/distfiles/LOCAL_PORTS. pkg_install findest und wie jedes andere Paket installieren kannst.

nach oben

8.3. Beispiel: Installation eines Programms aus den Quellen

Diese Sektion beschreibt die installation eines Beispielsprogramms: die addnerd- Applikation, die das Erstellen von neuen Benutzerkonten auf Deinem System vereinfacht. Als Erstes gehst Du nach /usr/pkgsrc/sysutils/addnerd .

nach oben

8.3.1. Download der Quellen

Wenn du mit dem Internet verbunden bist, besorgt das Makefile die nötigen Quellen; Du musst dann diese Sektion nicht lesen.

Download von einer anderen Maschine

Ein weit verbreitetes Szenario ist der Download der Paketquellen von einer anderen Maschine mit einer schnellen Internetverbindung (Vielleicht auf der Arbeit) und sie dann auf dem NetBSD-System zu installieren (Vielleicht zu Hause). Hinweis des Übersetzers: Frage vorher Deinen Boss, um Ärger zu vermeiden.

Ansonsten musst Du sehen, wie Du selbst an die Tarbälle kommst. In diesem Fall brauchst Du die Namen der Tarbälle. Schau im Makefile nach der Zeile:

```
DISTNAME = addnerd-1.6
```

Der volle Name des Pakets ist addnerd-1.6.tar.gz.

Du kannst dasselbe Ergebnis auf einem einfacheren Weg mit diesen Kommandos erzielen:

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

...womit Du gleichzeitig eine Liste der Sites angezeigt bekommst, von denen Du das Paket downloaden kannst.

nach oben

8.3.2. Kompilieren und Installieren

Zum Kompilieren gib das hier ein:

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

Das vorige Kommando beschafft das Quellarchiv (Wenn es nicht im distfiles- Verzeichnis vorhanden ist), extrahiert die Quellen; fügt die Patches hinzu, um es auf NetBSD kompilieren zu können und baut dann das Paket.

Dann installierst Du das Ganze:

```
# make install
```

Die Installation des neuen Programms wird auf dem System aufgezeichnet: Du kannst das mit **pkg_info -a** nachprüfen.

Das *addnerd*- Paket ist gebrauchsfertig. Du kannst wieder etwas Platz schaffen, indem Du die vom Compiler erstellten »Vermittlungsdateien« entfernst:

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

Das zweite Kommando muss nur durchgeführt werden, wenn einige abhängige Pakete installiert wurden, was bei addnerd nicht der Fall ist. Das gleiche Ergebnis kann auch mit diesem Kommando erzielt werden:

```
# make clean CLEANDEPENDS=1
```

nach oben

8.4. Beispiel: Installation eines Binärpakets

Ich habe schon im ersten Teil dieses Kapitels erklärt, dass das Package-System die Programme sowohl aus den Quellen heraus als auch Binärpakete installieren kann, die jemand anderes für NetBSD vorbereitet hat. Die zweite Form der Installation ist schneller, weil das Kompilieren des Pakets nicht nötig ist und die Tarbälle der Binärpakete normalerweise kleiner und damit schneller zu laden sind. Für ein Binärpaket wird die Package-Collection nicht gebraucht: Die »pkgtools«- Utilities reichen.

Die Tarbälle der Binärprogramme besitzen meistens die Endung `.tgz,` während die Tarbälle mit den Quellen mit `.tar.gz` enden.

> **Hinweis:** Nicht alle Source-Tarbälle enden mit .tar.gz.Das Package-System kommt auch mit anderen Pakettypen zurecht, beispielsweise .zip, .bz2.

Man muss die Binärpakete auch nicht unbedingt vor der Installation herunterladen. Mit den ftp://-URLs geht das genauso. Beispiel:

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

Wenn du nicht genau weisst, welche Version des Paketes auf der FTP-Site vorhanden ist, kannst Du die Versionsinfo auch weglassen und **pkg_add** die Auswahl überlassen: Es wird immer die neueste Version auf dem Server benutzen. Beispiel:

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

Es ist auch möglich, den PKG_PATH zu setzen. Man erstellt dann eine Liste mit den Pfaden und den URLs, wobei die Listeneinträge mit einem Semikolon getrennt werden müssen und pkg_add die Arbeit ganz zu überlassen:

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

Das Kommando installiert das erste tcsh- Binärpaket, das es findet.

Lass uns mal zu Übung das texinfo- Programm in der vorkompilierten Form installieren:

1. Kopiere gtexinfo-3.12.tgz in ein temporäres Verzeichnis.
2. Setze folgendes Kommando ab:

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

3. Prüfe, dass die Installation auch durchgeführt wurde:

   ```
   # pkg_info
   ```

4. Entferne gtexinfo-3.12.tgz aus dem temporären Verzeichnis.

Vorkompilierte Pakete sind sehr bequem, weil sie mit weniger Aufwand an Zeit und Aufmerksamkeit installiert werden können. Source- Pakete erlauben Dir eine bessere Kontrolle, weil die Compileroptionen an die eigenen Bedürfnisse angepasst werden können. Die Installation dauert wegen der Kompilierung länger, was auf einigen (meistens älteren) Plattformen kritisch werden kann.

Vor der Installation eines Binärpaketes mit **pkg_add** ist es besser, das Paket mit dem **pkg_info**-Kommando zu überprüfen. Beispiel:

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

Es hat auch einen gewissen Wert, mit dem CWD- Kommando nachzuprüfen, wo das Paket installiert wurde (Basisverzeichnis). Meistens sind das /usr/pkg und /usr/X11R6. Wenn das Basisverzeichnis nicht das Gewünschte ist, kannst Du das mit der Option **-p** hinter dem **pkg_add**- Kommando ändern. Beispiel: das Paket jpeg-6b.tgz ist standardmässig in /usr/pkg installiert; aber Du kannst es auch in /usr/X11R6 installieren, wenn Du das Paket mit diesem Kommando entpackst:

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

nach oben

8.5. Paketmanagement- Kommandos

Die wichtigsten Kommandos für das Paketmanagement sind:

pkg_add

... fügt vorkompilierte Pakete dazu.

pkg_delete

Löscht installierte Pakete. Paketnamen können mit oder ohne Version angegeben werden. Wenn keine Version vorgegeben wird, findet pkg_delete heraus, welche Version installiert ist. Wildcard können benutzt werden, sofern die Shell das akzeptiert. Beispiel:

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

Die »-r«-Option kann eine Menge: damit werden alle die Pakete entfernt, die das eigentliche Paket benötigt und bei der Installation nachgefragt hat. Beispiel:

```
# pkg_delete -r jpeg
```

entfernt jpeg und alle Pakete, die es brauchte. Das erlaubt das Updating des jpeg- Pakets.

pkg_info

zeigt die Informationen über Pakete an, installierte und nicht installierte.

pkg_create

erstellt ein neues Paket für die Package-Collection. Dieses Programm wird benutzt, um neue vorkompilierte Pakete zu schnüren. Es wird automatisch vom Build- System aufgerufen; Handarbeit bzw. manueller Aufruf ist nicht nötig.

pkg_admin

führt diverse administrative Funktionen im Package- System aus.

nach oben

8.6. Quickstart Packaging-Guide

Diese Sektion (Quickstart Packaging Guide) ist ein Beitrag von Jason R. Fink.

Diese Sektion zeigt eine Methode zum Bauen von relativ kleinen Paketen für das Packaging-System von NetBSD. Für weitere Details über einige verwirrende Dinge in diesem System solltest Du die Datei Packages.txt im pkgsrc- Baum konsultieren.

nach oben

8.6.1. Tools

Es gibt da drei primäre Dinge, die gebraucht werden, um mit NetBSD ziemlich schnell ein Paket erzeugt werden kann:

url2pkg

Das eigentliche Paket

pkglint

nach oben

8.6.1.1. url2pkg

Url2pkg kann vom pkgsrc- Baum aus installiert werden. Das Tool unterstützt den Package-Builder darin, rudimentäre Aspekte des Paketbaus einzubinden und zu testen.

8.6.1.2. Das eigentliche Paket

Das wird das eigentliche Paket bzw. Verzeichnis. In Packages.txt, Sektion 11 'A simple example of a package: bison' wird ein Beispiel gezeigt. Viele Pakete in NetBSD sind weitaus weniger komplex.

nach oben

8.6.1.3. pkglint

Pkglint kann aus dem pkgsrc- Baum installiert werden. Dieses Tool überprüft im Wesentlichen die Pakete auf ihre Korrektheit.

nach oben

8.6.2. Ein Anfang

Der Start des Prozesse ist relativ einfach. Es ist wichtig, dass der Hersteller (z.B. Du) das Bauen der Pakete aus den Quellen schon auf einem NetBSD-System getestet hat. Ansonsten kann das Aufsetzen eines neuen Pakets problematisch werden, wenn der Build fehlschlägt. Meisten kann man aber für die Quellen einen Patch schreiben, mit dem diese Probleme beseitigt werden können. Das ist aber nicht das Thema dieser Anleitung (Details stehen in Packages.txt).

nach oben

8.6.2.1. url2pkg

Der nächste Schritt ist url2pkg.

Folgendes muss getan werden, um die am Anfang benötigten Dateien für das neue Paket zu erzeugen:

1. Erstelle das Verzeichnis unter Deinem pkgsrc- Verzeichnis, in dem das neue Paket liegen soll. Lege noch nichts darin ab.

2. cd in das neue Verzeichnis.

3. Schreibe:

   ```
   $ url2pkg
   ```

4. Du wirst an diesem Punkt aufgefordert, einen URL einzugeben. Tue das und drücke <return>

5. Eine vi- Sitzung nimmt ihren Anfang

   Hinweis: Das Programm benutzt den Standardplatz von vi für NetBSD. Wenn Du normalerweise einen anderen Klon verwendest (wie vim) können .exrc-Fehlermeldungen auflaufen.

   Die vi- Sitzung ist für das Makefile des neuen Paketes. Du musst den Paketnamen angeben, wer das Paket pflegt und die Kategorie, in die diese Paket gehört.

6. Speichere die Datei und verlasse vi.

7. url2pkg greift das Paket automatisch ab und legt es in das Arbeitsverzeichnis.

8. Dann generiert urlpkg die md5- Dateien.

Das beendet eine urlpkg- Sitzung. Du solltest wissen, dass urlpkg nichts in andere Dateien schreibt, abgesehen vom Makefile. Es generiert eine leere PLIST- Datei.

nach oben

8.6.3. Den Rest dazutun

Jetzt ist das Makefile generiert worden. Die übrigen Dateien müssen generiert werden. Wenn Du in Deinem eigentlichen Paket arbeitest, musst Du folgende Dateien aus Deinem Paket in Dein Unterverzeichnis des pkg-Baumes kopieren.

DESCR

Ein Mehrzeiler mit der Beschreibung der Software. Hier sollten auch die Hinweise zu den Urhebern drinstehen.

COMMENTS

Eine einzeilige Kurzbeschreibung der Software. Hier muss der Paketname nicht drinstehen. Das erledigen die pkg_*- Tools, wenn sie aufgerufen werden.

PLIST

In dieser Datei steht, wo die Dateien auf dem System installiert werden: Ein kleines Paket (z.B.: eine Binärdatei und ein, zwei Manpages) ; ein verstohlener Blick auf das Makefile, installscript etc. sollte mit wenigen Worten illustrieren, wo die Daten abgelegt werden sollen.

nach oben

8.6.4. Prüfen mit pkglint

Wenn alle diese Dateien fertig sind, ist es Zeit, das Paket mit dem pkglint- Tool zu kontrollieren. Oft muss im Makefile eine Sektion verschoben, geändert oder sonstwie bearbeitet werden. Für die erste Runde ist es hilfreich, pkglint einfach vorher mal ablaufen zu lassen, damit Du genau weisst, was Du vielleicht ändern musst. Das hier ist der Output, der in Packages.txt nach einer pkglint- Sitzung nachzulesen ist:

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

Wenn ein Fehler aufläuft, ist die Ausgabe im Normalfall in Klartext; hier ist so eine Beispielsfehlermeldung, die ich bekam, während ich ein Paket baute:

```
extract suffix not required
```

Es bestand kein Bedarf, einen extract- Suffix im Makefile zu definieren.

nach oben

8.6.5. Installationen und gebaute Pakete ablaufen lassen und testen

Wenn das Paket pkglint durchlaufen hat, mache ich normalerweise einen kompletten Check, der den Beschaffen des Pakets, das Bauen und die Installation beinhaltet. Um das mit Erfolg durchzuführen, muss ich das Arbeitsverzeichnis und die Distfiles in /usr/pkgsrc/distfiles löschen. Nur auf diesem Weg kann ich sicherstellen, dass ich einen vollen und kompletten Testlauf durchführen kann.

nach oben

8.6.6. Ein Paket mit send-pr veröffentlichen

Als Erstes machst Du ein Archiv aus dem Package-Baum selbst (einschließlich des pkg/Arbeitsverzeichnisses); etwa so:

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

Als nächstes legst Du das Paket an einer Stelle ab, auf die die NetBSD-Package-Maintainer zugreifen können. Wenn Du das Archiv nirgends uploaden kannst, kontaktiere NetBSD, um zu sehen, ob es eine andere Methode gibt, Dein Archiv den Package-Maintainern zugänglich zu machen.

Die bevorzugte Methode, die NetBSD-Package-Maintainer zu informieren, ist »send-pr« mit einer Kategorie vom »pkg«, eine Zusammenfassung aus Paketname, Versionsnummer, eine Kurzbeschreibung und dem URL des Tarballs.

Du kannst sowohl send-pr dazu verwenden als auch das Online-Formular unter http://www.netbsd.org/cgi-bin/sendpr.cgi?gndb=netbsd benutzen, wenn Du aus irgendeinem Grund send-pr nicht bekommen kannst, um damit lokal zu arbeiten.

nach oben

8.6.7. Schlußbemerkungen

Nochmals zu Abschluß: Dieser kurze Anleitung ist für kleine Pakete gedacht, die nur ein paar kleinere Dateien enthalten, die auf einem NetBSD- System installiert werden sollen. In dieser Anleitung wird davon ausgegangen, dass keine Patches gebraucht werden und das Paket ohne Abhängigkeiten von anderen Paketen gebaut werden kann.

Für weitere Informationen lies bitte auch Packages.txt.

nach oben

Inhaltsverzeichnis

Kapitel 9. Netzwerke

Inhalt:

Dieses Kapitel soll eine Einführung in die Netzwerkgrundlagen sein. Hier geht es darum, eine Maschine mit einem LAN und dem Internet zu verbinden. Die Verbindung zweier Computer mittels eines seriellen Kabels wird hier ebenfalls beschrieben.

Wenn zwei Computer Daten austauschen wollen, müssen sie auf irgendeinem Weg miteinander verbunden werden. Beispiel: Es können zwei Maschinen mit Netzwerkkarten versehen werden; möglicherweise benutzen sie auch noch einen Hub oder Switch. Diese Art der Verbindung wird gewöhnlich als LAN (Local Area Network) bezeichnet.

Im Gegensatz dazu ist das Internet ein WAN (Wide Area Network): Wenn Du mit dem Internet verbunden bist, kannst Du auf Computer zugreifen, die physikalisch sehr weit von Dir entfernt sind; und das, ohne die näheren Details über die Verbindung zwischen den beiden Maschinen zu kennen.

nach oben

9.1. Die Konfigurationsdateien -Übersicht

Das, was jetzt kommt, ist eine Liste mit den Dateien, die benutzt werden, um das Netzwerk zu konfigurieren. Einige dieser Dateien, die schon in ersten Kapiteln benutzt wurden, werden hier und in den folgenden Absätzen detaillierter beschrieben.

/etc/hosts

> Datenbank mit den lokalen Maschinen. In jeder Teile stehen Informationen, die zu einem Host gehören. Hier stehen die Internet- Adresse, der Name und die Aliase drin. Kleine lokale Netze können so ohne Nameserver betrieben werden.

/etc/resolv.conf

> In dieser Datei steht drin, wie die Routinen, die den Zugriff auf das Internet- Domain-Name-System anbieten, arbeiten sollen. Generell steht hier die Adresse eines Nameservers drin.

/etc/ifconfig.xxx

> Mit dieser Datei wird die automatische Konfiguration der Netzwerkkarte beim Booten eingerichtet.

/etc/mygate

> Hier steht die IP- Adresse des Gateways ins Internet drin.

/etc/nsswitch.conf

Mit dieser Datei wird der Name-Service-Switch konfiguriert. Diese Datei regelt, wie und in welcher Reihenfolge auf die verschiedenen Datenbanken mit Benutzern, Hosts, Gruppen usw. zugegriffen werden soll. Beispiel:

Die Zeile...

```
hosts:    files dns
```

...beschreibt, dass die Datenbank mit den Hosts aus zwei Dateien kommt: files (»/etc/hosts«) und dns(Das Internet Domain Name System) und dass die lokalen Dateien vor dem DNS durchsucht werden sollen.

Normalerweise muss diese Datei nicht nachgearbeitet werden.

specifies that the hosts database comes from two sources, *files* (the l

nach oben

9.2. Verbindung zum Internet

Es gibt viele Arten einer Internetverbindung: In dieser Sektion steht, wie eine Verbindung zu einem Provider mit einem Modem über die Telefonleitung aufgebaut wird. Dazu wird das »PPP«- Protokoll benutzt. Damit wir eine funktionierende Verbindung herstellen können, sind diese Schritte hier nötig:

1. Die nötigen Informationen vom Provider
2. »/etc/resolv.conf« und »/etc/nsswitch.conf« müssen überarbeitet oder geprüft werden
3. Die Verzeichnisse »/etc/ppp« und »/etc/ppp/peers« müssen erstellt werden, wenn es sie noch nicht gibt.
4. Das Verbindungsscript, die »chat«- und die »options«- Datei müssen erstellt werden.
5. Ausserdem brauchen wir noch eine Datei, in der der Username und das Passwort stehen.

Wenn man sich die Liste so ansieht, erscheint einem das als viel Arbeit, die vorausgesetzt wird. Zur Zeit sind die einzelnen Schritte aber recht einfach: Man muss nur ein paar kleinere Textdateien erstellen, modifizieren oder auch nur überprüfen. In unserem Beispiel wird davon ausgegangen, dass das Modem an den zweiten seriellen Anschluss (»/dev/tty01«) gestöpselt ist (COM2 in DOS).

> Nebenbei bemerkt: Es können diese Modems benutzt werden: »/dev/tty0[012]« auf i386, »/dev/tty[ab] auf Sparcs, USB- Modems (»/dev/ttyU*«) und pcmcia/cardbus- Modems (»/dev/tty0[12]«).

nach oben

9.2.1. Informationen über die Verbindung einsammeln

Als erstes brauchst Du vom Provider die nötigen Informationen für die Verbindung; was heisst:

- Die Telefonnummer für den nächsten POP.
- Die benutzte Authentisierungsmethode.
- Benutzername un Passwort für die Verbindung.
- Die IP- Adresse des Nameservers.

nach oben

9.2.2. resolv.conf und nsswitch.conf

Die »/etc/resolv.conf« muss mit den Daten vom Provider konfiguriert werden, besonders, was die IP-Adressen des DNS angeht. In diesem Beispiel sind die Adressen des DNS 194.109.123.2 und 191.200.4.52 .

Beispiel 9-1. resolv.conf

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

> **Anmerkung:** Die letzte Zeile (»lookup file bind«) besagt, dass die Nameserver nur für die Namen benutzt werden, die in »/etc/hosts« nicht eingetragen sind. Diese Zeile ist auskommentiert, weil sei seit NetBSD 1.4 nicht mehr benötigt wird. Diese Information wird seitdem in »/etc/nsswitch.conf« definiert. Der neue Name-Service-Switch ändert den Zugriff auf die Datenbanken, die von Programmen benutzt werden, um die Systemkonfiguration zu ermitteln.

Hier ist ein Beispiel für eine »/etc/nsswitch.conf«-Datei..

Beispiel 9-2.: 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
```

> **Notiz:** Nur eine Zeile wurde geändert; und zwar die mit dem Wort »hosts«: Wenn ein Hostname aufgelöst wird, wird die lokale »hosts«- Datei zuerst durchsucht, bevor das DNS gefragt wird.

nach oben

9.2.3. Erstellen der Verzeichnisse für PPP

In den Verzeichnissen »/etc/ppp« und »/etc/ppp/peers« wird die Konfiguration der PPP-Verbindung gespeichert. Nach einer NetBSD- Neuinstallation gibt es sie noch nicht. Sie müssen erst erstellt werden (chmod 700).

nach oben

9.2.4. Verbindungsscript und chat-Datei

Das Verbindungsscript wird als ein Parameter für die pppd- Kommandozeile benutzt; es liegt in »/etc/ppp/peers« und beinhaltet normalerweise den Namen des Providers. Beispiel: Wenn der Name des Providers »BigNet« und der Verbindungsname »alan« ist könnte ein Verbindungsscript so aussehen:

Beispiel 9-3: Verbindungsscript

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

In diesem Beispiel beschreibt das Script eine »chat«- Datei, die bei der Verbindung benutzt werden soll. Die Details im Script werden in der ppp(8)- Manpage beschrieben.

> **Hinweis:** Wenn Probleme mit der Verbindung auftreten sollten, hänge diese beiden Zeilen an das Verbindungsscript an:
>
> ```
> debug
> kdebug 4
> ```
>
> Du bekommst dann eine Aufzeichnung der ausgeführten Operationen, wenn das System versucht, eine Verbindung aufzubauen. Siehe auch pppd(8) und syslog.conf (5).

Das Verbindungsscript ruft die »chat«- Applikation auf, die an der physikalischen Verbindung arbeitet (Modemstart, Einwahl...). Die Parameter können im Verbindungsscript angegeben werden; es ist aber besser, das in einer separaten Datei zu tun. Wenn die Telefonnummer der POP 02 999999999 ist, kann ein Beispielscript so aussehen:

Beispiel 9-4. Chatdatei

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

> **Hinweis:** Wenn Problem mit dem Chatscript auftauchen sollten, kannst Du versuchen, mit »cu« manuell eine Verbindung zum POP aufzubauen und die exakten Strings, die Du bekommst überprüfen. Siehe auch cu(1)

nach oben

9.2.5. Authentisierung

Während der Authentisierung ermittelt jedes System die Identität des anderen; praktisch bedeutet das, dass Du den Provider nicht authentisieren muss. Du wirst nur von ihm überprüft, und zwar mit diesen Methoden:

- login
- PAP/CHAP

Üblich ist bei den meisten Providern die PAP/CHAP- Authentisierung.

nach oben

9.2.5.1. PAP/CHAP- Authentisierung

Die Informationen zur Authentisierung liegen in »/etc/ppp/pap-secrets« für PAP und in »/etc/ppp/chao-secret« für CHAP. Die Zeilen darin benutzen dieses Format:

```
user * password
```

Beispiel:

```
alan * pZY9o
```

> **Hinweis:** Aus Sicherheitsgründen sollten »pap-secrets uns »chap-secrets« immer zu root gehören und nur für diesen les- und schreibbar sein (chmod 600).

nach oben

9.2.5.2. Login- Authentisierung

Diese Art der Authentisierung ist heute nicht so weit verbreitet. Wenn der Provider die Login-Authentisierung verwendet, müssen Username und Passwort in der Chat- Datei abgelegt werden; nicht in den PAP/CHAP- Dateien. In diesem Fall setzte die Zugriffsrechte auf die Dateien entsprechend.

Dieses ist ein Beispiel für eine Chat- Datei mit einer Login- Authentisierung:

Beispiel 9-5. Chatdatei mit Login

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

```
CONNECT ''
TIMEOUT 50
ogin: alan
ssword: pZY9o
```

nach oben

9.2.6. pppd- options

Das einzige, was noch fehlt, ist die »options«- Datei, die »/etc/ppp/options« (chmod 644) genannt wird.

Beispiel 9-6. /etc/ppp/options

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

Was die Optionen bedeuten, steht in der »pppd(8)«- Manpage.

nach oben

9.2.7. Das Modem testen

Bevor der Link aktiviert wird, sollte ein schneller Modemtest gefahren werden, um die physikalische Verbindung und die Kommunikation, mit der das Modem arbeitet, zu testen. Für den Test kann das »cu«- Programm benutzt werden. Beispiel:

1. Erstelle die Datei »/etc/uucp/port« mit folgenden Zeilen:

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

   (Trage bitte das richtige Gerät an der Stelle von »/dev/tty01« ein).

2. Mit dem Kommando »cu -p modem« wird begonnen, einige Kommandos an das Modem zu schicken. Beispiel:

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

   Disconnected.
   ```

```
#
```

In diesem Beispiel wurde das ATZ- Kommando an das Modem geschickt, das mit einem OK antwortete: Die Kommunikation funktioniert. Um »cu« zu verlassen, gib ein »~«(Tilde), gefolgt von einem Punkt ein; wie im Beispiel.

Wenn das Modem nicht funktioniert, kontrolliere, ob das Modem am richtigen Anschluss hängt (Wenn Du mit »cu« den richtigen Anschluss benützt). Auch die Kabel können schon mal die Übeltäter sein.

Hinweis: Wenn Du »cu« startest, und die Ausgabe auf dem Schirm »Permission denied« ist, prüfe nach, wer der Eigentümer der »/dev/tty**«- Geräte ist: uucp wäre korrekt. Beispiel:

```
$ ls -l /dev/tty00
crw-------  1 uucp  wheel  8, 0 Mar 22 20:39 /dev/tty00
```

Wenn der Eigentümer root ist, passiert folgendes:

```
$ 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
```

COM, com und tty

Ein paar Worte zum Unterschied zwischen »com«, »COM« und »tty«: In NetBSD ist »com« der Name des Treibers für die seriellen Anschlüsse (der, der von »dmesg« angezeigt wird). Und »tty« ist der Name des Anschlusses. Seit die Numerierung bei 0 beginnt, ist com0 der Treiber für den ersten seriellen Anschluss; genannt wird er tty00. In DOS ist statt dessen COM1 der erste serielle Anschluss (Liegt meistens auf 0x3f8), COM2 der zweite usw. Daher entspricht COM1 tty00 in NetBSD.

nach oben

9.2.8. Aktivierung der Verbindung

Jetzt ist alles fertig, um sich mit diesem Kommando mit dem Provider zu verbinden:

```
# pppd call bignet
```

»bignet« ist der Name im schon beschriebenen Verbindungsscript. Um die Informationen, die »pppd« für Dich hat, zu lesen, tue das hier:

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

Um die Verbindung abzubrechen mache ein »kill -HUP« für »pppd«

nach oben

9.2.9. Aufnahme und Abbruch der Verbindung mit einem Script steuern

Wenn die Verbindung funktioniert, ist es Zeit ein paar Scripts zu schreiben, um das dauernde Wiederholen der Kommandos zu verhindern. Diese beiden Script können beliebig benannt werden; wir benutzen hier »ppp-up« und »ppp-down«.

Ppp-up ist für die Aufnahme der Verbindung zuständig.

Beispiel 9-7. ppp-up

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

ppp-down bricht die Verbindung ab:

Beispiel 9-8. ppp-down

```
#!/bin/sh
MODEM=tty01
if [ -f /var/spool/lock/LCK..$MODEM ]; then
  echo -f killing pppd...
  kill -HUP `cat /var/spool/lock/LCK..$MODEM`
  echo done
else
  echo ppp is not active
fi
```

Die beiden Scripte gehen davon aus, dass »pppd« aktiv ist; es erstellt die Datei »LCK..tty01« im »/var/spool/lock«- Verzeichnis. In dieser Datei steht die »pid« des Prozesses.

Beide Scripte müssen ausführbar sein:

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

nach oben

9.3. Eine kleines, privates Netzwerk

Das Networking ist eine der Hauptstärken von Unix und NetBSD macht da keine Ausnahme: Es ist billig zu haben, einfach zu installieren und leistungsfähig, weil man keine zusätzliche Software kaufen muss, um einen Server zu bauen oder mit diesem zu kommunizieren. In

Sektion 9.4 steht, wie eine NetBSD- Maschine als Gateway für eine Netzwerk funktioniert: Mit IPNAT können alle Hosts im Netz über eine einzige Verbindung auf dem Gateway- Rechner mit einem Provider das Internet erreichen. Das einzige, was Du tun musst, ist, ein paar Netzwerkkarten zu kaufen, die von NetBSD unterstützt werden (In der INSTALL- Datei soll es eine Liste mit den unterstützten Karten geben).

Als Erstes müssen die Karten in die Maschinen eingebaut werden und mit einem Hub oder direkt verbunden werden (Siehe Bild 9-1).

Dann schau im »dmesg«- Output nach, ob die Karten vom Kernel erkannt werden. Im Beispiel handelt es sich um einen korrekt erkannten NE2000- Klon:

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

Wenn die Karte nicht erkannt wird, musst Du prüfen, ob die Karte in der Konfiguration eingeschaltet ist und die IRQs da gesetzt sind, wo der Kernel sie erwartet. Das hier ist eine NE2000- Karte für den ISA- Bus, die der Kernel auf Interrupt 9 erwartet.

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

Wenn die Konfiguration der Karte anders ist, wird sie wahrscheinlich beim Booten nicht gefunden. In diesem Fall hat Du zwei Möglichkeiten: Die entsprechende Zeile in der Konfigurationsdatei zu ändern und einen neuen Kernel zu backen oder die Einstellungen der Karte zu ändern (meistens mit einer Setup- Diskette oder, bei alten Karten, ein Jumper auf der Karte).

Das folgende Kommando zeigt die aktuelle Konfiguration der Karte:

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

Dir Softwarekonfiguration der Netzwerkkarte ist sehr leicht. Die IP- Adresse »192.168.1.1« (reserviert für interne Netze) wird der Karte zugewiesen.

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

Wenn Du das Kommando wiederholst, gibt es ein anderes Ergebnis:

```
# ifconfig ne0
ne0: flags=8863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST> mtu
          media: Ethernet 10base2
          inet 192.168.1.1 netmask 0xffffff00 broadcast 192.168.1.255
```

Der Output von »ifconfig« hat sich geändert: Die IP- Adresse wird jetzt angezeigt; Ausserdem zwei neue Flags: »UP« und »RUNNIG«. Wenn das interface nicht »UP« ist, wird das System es nicht zu Verschicken von Paketen benutzen.

Der Host bekam die Adresse 192.168.1.1, die zu einem Satz von Adresse gehört, die für lokale Netze reserviert und nicht aus dem Internet erreichbar sind. Die Konfiguration ist nun fertig und muss getestet werden. Wenn ein anderer Host im Netz aktiv ist, kann ein »ping« versucht werden. Im Beispiel ist 192.168.1.2 der andere aktive Host:

```
# 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
```

Mit der aktuellen Installation muss die Konfiguration nach einem Reboot wiederholt werden. Um das zu verhindern, müssen zwei Dinge getan werden: Erstens: Die Datei »/etc/ifconfig.ne0« mit dieser Zeile als Inhalt:

```
inet 192.168.1.1 netmask 0xffffff00
```

In »/etc/rc.conf« muss die folgende Option gesetzt werden:

```
auto_ifconfig=YES
```

Ab dem nächsten Boot wird die Netzwerkkarte automatisch konfiguriert.

»/etc/hosts« ist eine Datenbank mit IP- Adressen und den dazugehörigen Aliases: Sie sollte die Adressen aller Hosts enthalten, die zum lokalen Netzwerk gehören. Beispiel:

```
#       $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
```

»/etc/nsswitch.conf« sollte modifiziert werden wie in Beispiel 9-2 beschrieben.

> **Hinweis:** In diesem Beispiel wurde die Datei »/etc/ifconfig.ne0« erstellt, weil der Kernel die Netzwerkkarte als »ne0« erkannt hat. Wenn du eine andere Karte benützt, ersetze den Namen durch den für Dich richtigen.

Summasummarum: Um ein Netzwerk einzurichten, muss das hier getan werden: Die Netzwerkkarten müssen eingebaut und physikalisch verbunden werden. Dann muss man sie konfigurieren (mit »ifconfig«). Und zu guter Letzt müssen »/etc/hosts« und »/etc/nsswitch.conf« angepaßt werden. Diese Art des Netzwerkmanagements stark vereinfacht und für Netze ohne besondere Bedürfnisse passend.

nach oben

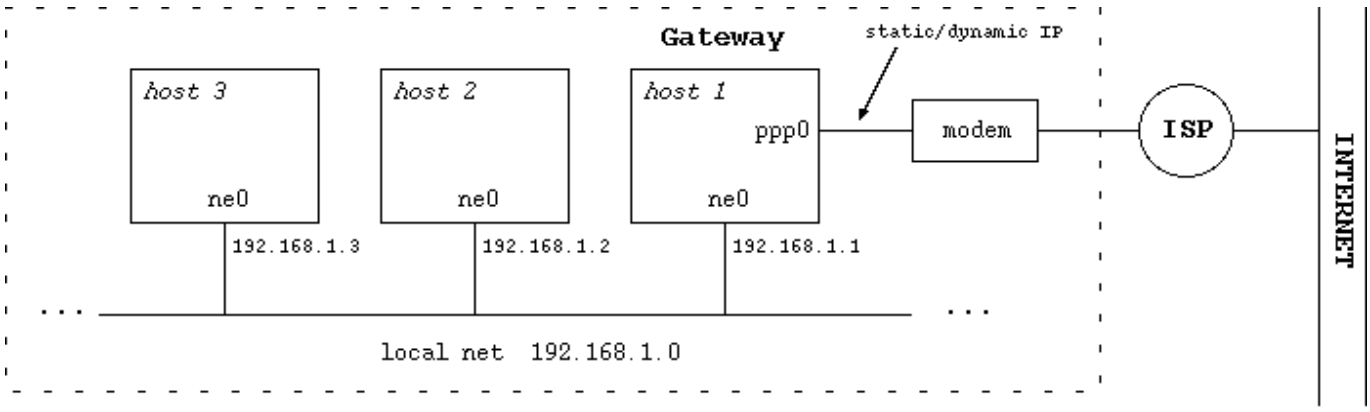
9.4. IPNAT

Die mysteriöse Abkürzung IPNAT steht für Internet Protocol Network Address Translation, womit man von einem internen Netzwerk in ein reales Netz routen kann (Gemeint ist das Internet). Das bedeutet: Mi nur einer »realen« IP- Adresse, egal ob statisch oder dynamisch, ist es möglich, für alle Hosts im internen Netzwerk simultane Verbindungen ins Internet aufzubauen.

Ein paar Beispiele, wie IPNAT benutzt wird, finden sich im Unterverzeichnis »/usr/share/examples/ipf: Schau Dir die Dateien »BASIC.NAT und »nat-setup« an.

Das Aufsetzen des in dieser Sektion beschriebenen Beispiels wird in Bild 9-1 detailliert gezeigt: Host 1 kann sich über ein Modem mit einem Provider verbinden und bezieht von diesem eine dynamische IP- Adresse. Host 2 und Host 3 können nicht direkt mit dem Internet kommunizieren: Host 1 dient in diesem Fall als Gateway für die anderen.

Bild 9-1. Netzwerk und Gateway



nach oben

9.4.1. Konfiguration von Gateway/Firewall

Um IPNAT benutzen zu können, muss das »Pseudo-Device Ipfilter« im Kernel vohanden sein. Stelle bitte fest, ob das der Fall ist:

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

Wenn das Ergebnis »1« ist, wie oben gezeigt, ist es im Kernel vorhanden. Wenn nicht ist die Ausgabe »0«. Dann kannst Du zwei Sachen machen:

1. Einen neuen Kernel kompilieren, in dem die GATEWAY- Option als Standard vorhanden ist

2. ...oder die Option mit diesem Kommando im aktuellen Kernel einschalten:

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

   Wenn Du diese Option in ein Bootscript einsetzst (Bsp: »/etc/rc.local«), passiert das künftig automatisch beim Booten.

Der Rest dieser Sektion beschreibt, wie eine IPNAT- Konfiguration aussieht, die immer dann automatisch gestartet wird, wenn mittels PPP eine Verbindung zum Provider aufgebaut wird. Mit dieser Konfiguration können sich alle Hosts im heimischen Netz durch einen Gateway- Rechner mit dem Internet verbinden, besonders, wenn sie mit NetBSD laufen.

Als Erstes brauchst Du eine leere »/etc/ipf.conf«:

```
# touch /etc/ipf.conf
```

Als nächstes brauchst Du die »/etc/ipnat.conf« mit diesen Regeln:

```
map ppp0 192.168.1.1/24 -> 0/32 proxy port ftp ftp/tcp
map ppp0 192.168.1.1/24 -> 0/32 portmap tcp/udp 40000:60000
map ppp0 192.168.1.1/24 -> 0/32
```

192.168.1.1/24 sind die Netzwerkadressen, die gemappt werden sollen (in diesem Fall geht das auch mit 192.168.1.0/24). Die erste Zeile in der Konfigurationsdatei ist optional: Sie erlaubt FTP- Verbindungen über den Gateway. Die zweite Zeile wird benutzt. Um icp und udp- Pakete korrekt zu händeln; Das Portmapping ist wegen des Verhältnisses Viele zu eins nötig. Die dritte Zeile erlaubt ICMP- Verbindungen für Ping usw.

In »/etc/rc.conf« muss das Portmapping eingeschaltet sein (Ipfilter ist nicht nötig).

Erstelle die »/etc/ppp/ip-up«-Datei, die automatisch aufgerufen wird, wenn der PPP- Link aktiviert wird.

```
#!/bin/sh
# /etc/ppp/ip-up
/usr/sbin/ipnat -F
/usr/sbin/ipnat -C
/sbin/ipf -E
/usr/sbin/ipnat -f /etc/ipnat.conf
```

Du brauchst dann noch »/etc/ppp/ip-down«, die aufgerufen wird, wen die Internetverbindung

abgebrochen wird.

```
#!/bin/sh
# /etc/ppp/ip-down
/sbin/ipf -D
/usr/sbin/ipnat -C
```

Beide Dateien müssen ausführbar sein.

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

Der Gateway ist jetzt fertig.

nach oben

9.4.2. Die Client- Konfiguration

Erstelle eine »/etc/resolv.conf« wie auf dem Gateway.

Dann schreibst Du das Kommando hier:

```
# route add default 192.168.1.1
```

192.168.1.1 ist die Adresse des Gateways, den wir vorhin konfiguriert haben.

Natürlich wirst Du dieses Kommando nicht nach jedem Systemstart neu eingeben wollen; es ist besser, die Defaultroute in »/etc/rc.conf« einzutragen oder, was denselben Effekt hat, eine »/etc/mygate«- Datei anzulegen: Die Defaultroute wird beim Systemstart automatisch gesetzt, indem die Inhalte von »/etc/mygate« (oder die »defaultroute«- Option) als Argumente des »route add default«- Kommandos benutzt werden.

Wenn die Clients nicht mit NetBSD laufen, wird die Konfiguration anders sein. Auf Windows-PCs setzt Du den Gateway des TCP/IP- Protokolls auf die IP- Adresse des NetBSD- Gateways.

Das ist alles, was bei den Clients zu tun ist.

nach oben

9.4.3. Ein paar nützliche Kommandos

Dieses Kommando ist nützlich, um Probleme zu diagnostizieren:

ping

netstat -r

Damit werden die Routingtabellen angezeigt:

traceroute

Auf dem Client kann man damit den Weg verfolgen, die die Pakete zu ihrem Ziel

benutzen: destination.

tcpdump

Benutze es auf dem Gateway, um TXP/IP- Traffic zu verfolgen.

nach oben

9.5. Verbindung zwischen zwei PCs über eine serielle Verbindung

Wenn Du Dateien zwischen zwei nicht vernetzten PCs verschieben willst, gibt es eine einfache Lösung, wenn das Kopieren auf Disketten nicht praktikabel ist: Die beiden Maschinen können mit einem seriellen Kabel verbunden werden (Ein sog. Null- Modem- Kabel). In den folgenden Sektionen werden einige Konfigurationen beschrieben.

nach oben

9.5.1. Eine Verbindung mit NetBSD oder Linux

Am einfachsten ist es, wenn auf beiden Maschinen NetBSD läuft: Eine Verbindung mit dem SLIP- Protokoll ist sehr einfach. Auf der ersten Maschine gibst Du diese Kommandos ein:

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

Auf Maschine zwo das hier:

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

Jetzt kannst Du die Verbindung mit einem »ping« vom zweiten PC aus testen:

```
# ping 192.168.1.1
```

Wenn alles funktioniert, gibt es jetzt eine aktive Verbindung zwischen den beiden Maschinen. FTP, telnet und Ähnliches können ausgeführt werden. Die Aliase und Adressen können in »/etc/hosts« abgelegt werden.

- Im Beispiel benutzten beide PCs den ersten seriellen Anschluss (/dev/tty00). Passe das bitte an, wenn Du einen anderen Anschluss benutzen willst.
- IP- Adressen wie 192.168.X.X sind für »interne« Netze reserviert. Der erste PC hat die Adresse 192.168.1.1 und der zweite 192.168.1.2
- Um eine schnellere Verbindung zu erreichen, kann die Geschwindigkeit mit »-s speed« als Option zu »slattach« festgelegt werden.
- **ftp** kann nur benutzt werden, wenn »inetd« aktiviert und der FTPD- Server eingeschaltet ist.

  Linux: Wenn auf einer Maschine Linux läuft, sind die Kommandos etwas anders

(nur auf der Linux- Maschine). Wenn die Linuxmaschine die Adresse 192.168.1.2 bekommt werden diese Kommandos gebraucht:

```
# 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
```

Vergiß das »&« im ersten Kommando nicht!

nach oben

9.5.2. NetBSD und Windows NT

NetBSD und Windows NT können (meistens) einfach mit einem Nullmodemkabel verbunden werden. Was wir dazu brauchen, ist eine »Remote Access«- Verbindung und ein pppd auf NetBSD.

Starte pppd, sobald Du eine ».ppprc«- Datei in »/root« erstellt hast. Hier ist ein Beispiel, das als Schablone dienen kann:

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

Was die erste Zeile bedeutet, wird später in dieser Sektion beschrieben. 192.168.1.2 ist die Adresse des NT- Rechners, die von NetBSD dem NT- Rechner zugeordnet wird. »tty00« ist der serielle Port, der für die Verbindung verwendet wird (der erste serielle Port).

Auf der NT- Seite muss ein Nullmodem- Gerät vom Modem- Kontrollfeld aus installiert werden. Ausserdem wir eine Remote-Access- Verbindung gebraucht. Der Nullmodemtreiber, der unter NT 4 Standard ist, ist kein 100%iger Nullmodemtreiber. Wenn die Verbindung aktiviert wird, sendet NT den String CLIENT und erwartet als Antwort CLIENTSERVER. Das besagt die erste Zeile von ».ppprc«: »chat« muss NT antworten, ansonsten klappt's nicht mit dem Nachbarn.

In der Konfiguration der Remote-Access-Verbindung muss folgendes festgelegt werden: das Nullmodem, Telefonnummer »1«(wird aber nicht benutzt) und die Nameserver des Servers (In diesem Fall NetBSD). Benutze Hardware-Flow und setze den Port auf 115200 8N1.

Jetzt kann die Verbindung aktiviert werden:

- Verbinde die beiden Maschinen mit dem Nullmodemkabel.
- Starte den pppd auf der NetBSD- Maschine
- Aktiviere die Remote-Access- Verbindung auf Windows NT.

nach oben

9.5.3. NetBSD und Windows 95

Das Setup für Windows 95 ähnelt dem für Windows NT: Remote-Access auf Windows 95 und der PPP- server auf NetBSD werden benutzt. Die meisten (nicht alle) Versionen von Windows 95 haben keinen Nullmodemtreiber, was diese Dinge ein bißchen komplizierter macht. Die einfachste Lösung ist es, sich einen Treiber aus dem Internet zu beschaffen (ist eine kleine ».inf«- Datei) und dieselben Schritte wie unter NT auszuführen. Der einzige unterschied ist, dass auf die erste Zeile in ».ppprc« (die, die »chat« aufruft) verzichtet werden kann.

Wenn Du so einen Treiber nicht im Internet finden kannst, geht es immer noch mit einem kleinen Trick:

- Erstelle eine Remote-Access- Verbindung über das »Standardmodem«
- In ».ppprc« muss die Zeile, mit der »chat« aufgerufen wird so aussehen:

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

- Aktiviere die Verbindung wie in Sektion 9.5.2 beschrieben.

Auf diesem Weg schickt das »chat«- Programm dieselben Antworten an Windows 95, wie es in Standardmodem tut, sobald die Verbindung aktiv ist. Wenn Windows 95 ein Kommandostring sendet, schickt »chat« ein »ok« zurück.

nach oben

9.6. NFS

Jetzt, wo das Netzwerk funktioniert, kann man mit NFS Dateien und Verzeichnisse über das Netzwerk teilen. Aus dem Gesichtspunkt des Filesharings ist der Computer, der Zugriff auf seine Verzeichnisse und Dateien erlaubt; der Server und der andere, der auf diese Sachen zugreift, der Client. Ein Computer kann gleichzeitig Server un Client sein.

- Ein Kernel mit den nötigen Optionen für Client und Server muss kompiliert werden (Die Optionen sind in der Kernel- Konfigurationsdatei leicht zu finden).
- Der Server muss RPC- Dienste über »/etc/inetd.conf« anbieten.
- In »/etc/rc.conf« müssen inetd, portmap und die nfs_server- Option aktiviert sein.
- In der »/etc/rc.conf« des Clients müssen inetd und nfs_client aktiviert sein.
- Im Server müssen die zu exportierenden Verzeichnisse gelistet sein und das Kommando **kill -HUP `cat /var/run/mountd.pid** muss anschliessend ausgeführt werden.

Ein Client kann auf ein entferntes Verzeichnis via NFS zugreifen, wenn...

- der Server das Verzeichnis exportiert;
- ...und der Client das Verzeichnis mit dem Kommando »mount server:/xx/yy /mnt« mountet.

Das »mount«- Kommando kommt mit einer grossen Anzahl von Optionen für entfernte Dateisysteme, die nicht sehr intuitiv sind (um das wenigste zu sagen).

9.6.1. NFS- Konfigurationsbeispiel

Warnung: Dieses ziemlich komplizierte Beispiel stammt von einer Mailingliste und ich habe es nicht getestet; aber es sollte Dir eine Ahnung verschaffen, wie NFS funktioniert.

Das Szenario ist das: Wir haben fünf Clients (cli1, ... ,cli5), die sich ein paar Verzeichnisse auf einem Server teilen sollen. Einige der exportierten Verzeichnisse sollen für einen bestimmten Client reserviert bleiben; auf die anderen Verzeichnisse dürfen alle Maschinen zugreifen. Alle Clients booten vom Server und müssen die Verzeichnisse mounten.

Die Verzeichnisse auf dem Server sehen so aus:

/export/cli?/root

Fünf Rootverzeichnissse für fünf Clients. Jeder Client besitzt ein eigenes Rootverzeichnis.

/export/cli?/swap

Fünf Swap- Verzeichnisse für die fünf Clients.

/export/common/usr

/usr - Verzeichnis: darauf sollen alle zugreifen.

/usr/src

Ein allegemeines /usr/src -Verzeichnis für alle Clients.

Dies ist das Dateisystem auf dem Server:

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

Jeder Client braucht diese Dateisysteme:

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

Der Server wird folgendermaßen konfiguriert:

```
# /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.
```

Die /etc/fstab auf den Clients sieht so aus:

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

Jeder Client hat seine Nummer, die mit den »?« in der ersten Zeile des Beispiels dargestellt wird.

nach oben

Inhaltsverzeichnis und Ausstieg

Kapitel 10. Nameserver

Inhalt:

Dieses Kapitel (DNS) ist ein Beitrag von Jason R. Fink

Das Domain Name System von NetBSD

In diesem Kapitel wird das Aufsetzen einer einfachen kleinen Domäne mit einem Domain-Nameserver auf einem NetBSD- System beschrieben. Es soll keinen detaillierten Überblick darüber bieten, was DNS ist und was es kann. Eine kurze Erläuterung ist aber im Angebot. Weitere Informationen gibt es beim »DNS Resources Directory«(DNSRD) unter www.dns.net/dnsrd.

10.1. Anmerkungen und Voraussetzungen

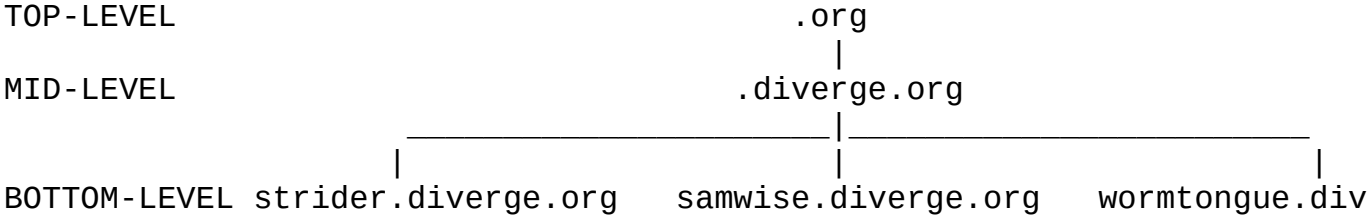
Die Beispiele in diesem Kapitel basieren auf der ersten Version 8; man sollte allerdings wissen, dass das Datenbankformat und die »named.conf« ziemlich 100%ig kompatibel untereinander sind. Der einzige Unterschied, den ich bemerkte, war, dass die »$TTL«- Information nicht vorausgesetzt wird.

Der Leser sollte aber wissen, wie ein Hostname auf eine IP- Adresse übertragen wird; und die Klassifizierungen von IP- Adressen kennen.

nach oben

10.2. Was ist DNS?

Das »Domain Name System« konvertiert Maschinennamen in IP- Adressen. Das Konvertieren wird vom Namen zur Adresse und von der Adresse zum Namen durchgeführt. Der Unterschied zwischen dem einfachen Übersetzen der Hostnamen und von DNS ist der, dass DNS für seine Zwecke einen hierarchischen Standard benutzt. Diese Hierarchie arbeitet von rechts nach links, wobei die höchste Stufe auf der rechten Seite zu finden ist. Als ein Beispiel findet sich hier eine einfache Skizze:



Es scheint einfach zu sein; aber das System kann auch noch weitere logische Aufteilung erfahren, wenn es jemand an verschiedenen Stellen wünscht. Das oben gezeigte Beispiel zeigt drei Nodes in der »diverge.org«- Dömäne. Wir können das aber auch noch weiter aufteilen, wie

»strider.nat1.diverge.org«, »samwise.net2.diverge.org« und wormtongue.net2.diverge.org. In diesem Fall residieren zwei Nodes auf net2.diverge.org und einer auf net1.diverge.org.

nach oben

10.3. Die DNS- Dateien

Lass uns mal einen Blick auf ein kleines DNS- fähiges Netzwerk werfen. Wir machen weiter, indem wir das Beispiel von oben verwenden. Bevor es losgeht, müssen wir allerdings ein paar Sachen sicherstellen.

- Unsere Host- IP- Übersetzung ist korrekt
- IPNT funktionert richtig
- Derzeit nutzen alle Hosts den ISP für die Namensauflösung

  Hinweis: Diese Art von Konfiguration wird in Kapitel 9 beschrieben.

Unser Nameserver wird der »strider«- Host werden. Auf ihm läuft IPNAT; und unsere beiden Clients nutzen ihn als Gateway ins Internet. Es ist nicht wirklich von Bedeutung, von welcher Art das Interface auf strider ist; aber um dem Kind einen Namen zu geben, nehmen wir eine typische Internet- Verbindung mit einem schlichten 56k- Modem.

Also, bevor es weitergeht, betrachten wir uns das »hosts«- File auf strider, bevor wir etwas für die Nutzung von DNS ändern.

nach oben

Beispiel 10-1. strider's /etc/hosts file

```
127.0.0.1       localhost
192.168.1.1     strider
192.168.1.2     samwise sam
192.168.1.3     wormtongue worm
```

...nicht wirklich ein grosses Netzwerk; aber es ist sinnlos, wenn wir nicht die selben Regeln wie in grösseren Netzen verwenden, also eben die, die wir im Kontext dieser Sektion diskutieren.

nach oben

10.3.1. /etc/namedb/named.conf

Das NetBSD- Betriebssystem bietet eine Reihe von Standardzeilen, von denen aus Du durchstarten kannst. Sie befinden sich in »/etc/namedb. Ich empfehle allerdings wärmstens, für Vergleichszwecke eine Sicherungskopie davon anzufertigen.

Das Standardverzeichnis enthält diese Dateien:

- 127
- localhost

- loopback.v6
- named.conf
- root.cache

Du wirst in meiner Konfiguration nichts anderes als geänderte Versionen dieser Dateien finden.

Das erste Teil, auf das wir einen Blick werfen ist »/etc/named/named.conf. Das ist die Konfigurationsdatei von Bind (man achte auf den unauffälligeren Namen). Ein System wie das, das wir herstellen wollen, ist ziemlich einfach. Meine Datei sieht so aus:

```
options {
        directory "/etc/namedb";
        allow-transfer { 192.168.1.0/24; };
        recursion yes;
        allow-query { 192.168.1.0/24; };
        listen-on port 53 { 192.168.1.1; };
};

zone "localhost" {
   type master;
   notify no;
   file "localhost";
};

zone "127.IN-ADDR.ARPA" {
   type master;
   notify no;
   file "127";
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6
   type master;
   file "loopback.v6";
};

zone "diverge.org" {
   type master;
   notify no;
   file "diverge.org";
};

zone "192.168.1.in-addr.arpa" {
   type master;
   notify no;
   file "192.168.1";
};

zone "." in {
   type hint;
   file "root.cache";
};
```

Man achte darauf, dass in meiner »named.conf« die Root- Abteilung die letzte ist, weil es keine andere Domain gibt, die diverge.org heisst (Ich besitze sie); also lasse ich den Resolver als letztes im Internet suchen. Auf den meisten Systemen ist das der Normalfall.

Eine andere wichtige Sache ist, daran zu denken, dass Du, falls Du eine interne Einstellung hast; keine Anfragen nach draussen schicken kannst. Mit anderen Worten: es gibt keine Live-Verbindung ins Internet oder keinen Bedarf nach Root- Server- Lookups, die Root- Zone auskommentierst. Es kann dann Probleme geben, wenn ein einzelner Client sich entscheidet, eine Anfrage über das Internet abzuklären.

Das klingt nach einem grösseren Problem. Bei genauerer Überprüfung scheint es so, dass viele Zeilen in den Sektionen redundant sind. Es ist aber nur erklärungsbedürftig.

Lass und mal die einzelen Sektionen von »named.conf« durchgehen.

nach oben

10.3.1.1. options

Diese Sektion definiert ein paar globale Parameter; der prominenteste ist der Ort, an dem sich die DNS- Tabellen auf diesem System finden. Sie wohnen suf diesem System in »/etc/named«.

Die anderen Parameter:

»allow-transfer«

> Für entfernte DNS- Server, die als Secondary arbeiten und Zoneninformationen von Dir brauchen.

»allow-query«

Das Netzwerk, das diesen Server benutzen darf

»listen-on port«

> Der Port auf dem dieser Server läuft.

Der Rest von »named.conf« ist in Zonen unterteilt; zu jeder Zone gehört eine Datei mit Tabellen darin, um diesen bestimmten Bereich (oder Zone) der Domain bearbeiten zu können. Deren Format in »named.conf« ist ziemlich ähnlich. Ich werde nur einen dieser Records hervorheben:

nach oben

10.3.1.2. zone "diverge.org"

type

> Der Zonentyp in allen Fällen, ausser ».«: Das sind die Master- Server

notify

> Willst Du Informationen verschicken, wenn sich Deine Zone ändert? In dieser Konfiguration nicht

file

Der Dateiname im »named«- Verzeichnis, wo Aufzeichnungen über diese spezielle Zone gefunden werden können.

nach oben

10.3.2. /etc/namedb/localhost

Zu einem grossen Teil sehen die Zonendateien ziemlich ähnlich aus, allerdings hat jede einzelne von ihnen eigene Besitzverhältnisse. So wie hier sieht die »localhost«- Datei aus:

Beispiel 10-2. localhost

```
 1|$TTL    3600
 2|@               IN SOA  strider.diverge.org. hostmaster.diverge.org.
 3|                         1       ; Serial
 4|                         8H      ; Refresh
 5|                         2H      ; Retry
 6|                         1W      ; Expire
 7|                         1D)     ; Minimum TTL
 8|                         IN NS   localhost.
 9|localhost.               IN      A       127.0.0.1
10|                         IN      AAAA    ::1
```

Zeile für Zeile: :

Zeile 1

Dieses ist die »Time to Live(Lebenszeit)« für Abfragen. Es ist immer dieselbe in allen Dateien.

Zeile 2

Diese Zeile ist generell dieselbe in allen Zonendateien ausser »root«. Unser besonderes Interesse in dieser Zeile sind strider.diverge.org und root.diverge.org. Einer dieser Namen ist der Name dieses Servers und der andere ist die Kontaktadresse für diesen Server, was meistens etwas zweideutig erscheint. Es sollte allerdings eine reguläre eMail- Adresse für die Kontaktinformationen verwendet werden. (Als Beispiel: Die meinige ist jrf.diverge.org).

Zeile 3

Diese Zeile ist die Seriennummer. Die meisten Leute tragen hier etwas in einem Format wie diesem hier ein: MMDDYYYY. Die Seriennummer sollte immer erhöht werden, wenn eine Änderung in der Datei vorgenommen wurde, nachdem sie installiert wurde. Deshalb starte ich lieber mit einer schlichten 1.

Zeile 4

Das ist die Refresh- Rate des Servers; in diesem Fall passiert das einmal alle 8 Stunden.

Zeile 5

Die Retry- Rate..

Zeile 6

Das Verfallsdatum eines Lookups.

Line 7

Die minimale Lebenszeit (TTL)

Line 8

Das ist die Namerserverzeile: Wie Du siehst, ist sie auf Localhost gesetzt.

Line 9

Der Eintrag für localhost.

Line 10

Der IPV6- Eintrag.

nach oben

10.3.3. /etc/named/zone.127.0.0

```
Diese Datei dient der Rückwärtsauflösung
für Localhost. Sie sieht aus wie das hier:

 1| $TTL    3600
 2| @               IN SOA  strider.diverge.org. root.diverge.org. (
 3|                         1       ; Serial
 4|                         8H      ; Refresh
 5|                         2H      ; Retry
 6|                         1W      ; Expire
 7|                         1D)     ; Minimum TTL
 8|                  IN NS   localhost.
 9| 1.0.0            IN PTR  localhost.
```

Von Zeile 9 einmal abgesehen, entspricht diese Datei der der Localhost- Zone. Dieses ist der Record für die Rückwärtsauflösung. Er wird in einer separaten Datei definiert, weil Localhost eine völlig andere Adresse hat, als die andere Zonen. Einige werden wir noch diskutieren, wenn wir uns alle Zonendateien angeschaut haben.

nach oben

10.3.4. /etc/namedb/diverge.org

Diese Zone wird von den Records von allen unseren Hosts bevölkert. So wie unten sieht diese Datei aus:

```
 1| $TTL    3600
```

```
 2| @              IN SOA  strider.diverge.org. root.diverge.org. (
 3|                         1       ; serial
 4|                         8H      ; refresh
 5|                         2H      ; retry
 6|                         1W      ; expire
 7|                         1D )    ; minimum seconds
 8|                  IN NS   strider.diverge.org.
 9|                  IN MX   10 maila.diverge.org.   ; primary mail serv
10|                  IN MX   20 mailb.diverge.org.   ; secondary mail se
11| strider          IN A     192.168.1.1
12| maila            IN CNAME strider.diverge.org.
13| samwise          IN A     192.168.1.2
14| www              IN CNAME samwise.diverge.org.
15| mailb
16| worm             IN A     192.168.1.3
```

Es gibt hier viele neue Sachen, also pflücken wir diese Datei Zeile für Zeile auseinander:

Zeile 12

Diese Zeile beschreibt unseren Mailhandler. In diesem Fall ist es strider. Aber um das besser skalieren zu können, nennen wir ihn maila. Wie Du unten sehen wirst, ist es vom Aufwand her keine Riesenaktion. Die Nummer die an »maila.diverge.org« vergeben worden ist, ist die »Priority Number«, was bedeutet, dass dieser Rechner (versehen mit der Nummer 10) der erste Mailserver im System ist. »mailb.diverge.org« hat die Nummer 20, ist also der zweite Mailer im Netz. Noch anders: Die kleinste Zahl hat die höchste Priorität. Der Sinn der Sache ist dieser: Wenn »maila(strider)« die Mail aus irgendwelchen Gründen nicht bearbeiten kann, erledigt »mailb(samwise)« den Job.

Zeile 12

> CNAME steht für »kanonischer Name (canonical name)«, oder prosaisch gesprochen, ist das ein Aliasname. Wir haben die folgenden Aliase vergeben:
>
> ```
> maila.diverge.org to strider.diverge.org
> mailb.diverge.org to samwise.diverge.org
> www.diverge.org   to samwise.diverge.org
> ```

Der Rest der Aufzeichnungen sind einfache Mappings von IP- Adressen in vollständige Hostnamen.

nach oben

10.3.5. /etc/namedb/192.168.1

Diese Zonendatei ist das Reverse-File der Host- Records. Das Format entspricht weitgehend der localhost- Version, abgesehen davon, dass die IP- Adressen anders sind:

```
 1|$TTL    3600
 2|@              IN SOA  strider.diverge.org. root.diverge.org. (
 3|                       1       ; serial
 4|                       8H      ; refresh
 5|                       2H      ; retry
```

```
 6|                      1W      ; expire
 7|                      1D )    ; minimum seconds
 8|                IN NS   strider.diverge.org.
 9|1               IN PTR  strider.diverge.org.
10|2               IN PTR  samwise.diverge.org.
11|3               IN PTR  worm.diverge.org.
```

nach oben

10.3.6. /etc/namedb/root.cache

Diese Datei ist eine Liste mit den Root- Servern, um nachzufragen, wenn Anfragen ausserhalb seiner eigenen Domain eingehen. Hier sind die ersten paar Zeilen einer Root- Zonendatei:

```
;       $NetBSD: root.cache,v 1.8 1997/08/24 15:50:47 perry Exp $
;
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  file"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC registration services
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.RS.INTERNIC.NET
;       -OR- under Gopher at    RS.INTERNIC.NET
;           under menu          InterNIC Registration Services (NSI)
;              submenu          InterNIC Registration Archives
;           file                named.root
;
;       last update:    Aug 22, 1997
;       related version of root zone:   1997082200
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
. . .
```

Diese Datei kann von ISC auf http://www.isc.org bezogen werden. Ausserdem kommt das gewöhnlich mit einer Distribution von BIND. Eine »root.cache«- Datei wird mit dem NetBSD-

Kernsystem geliefert.

nach oben

10.4. Nutzung von DNS

In dieser Sektion werden wie sehen, wie wir DNS zum Laufen kriegen und strider dazu bewegen, seine eigenen Dienste zu nutzen.

NetBSD bietet immer eine DNS- Cachingserver- Installation an (sieher nächste Sektion). Dazu gehören auch die Tools, um den Server während der Laufzeit administrieren zu können. Bevor wir starten, müssen wir herausfinden, wie der Server efrfolgreich gestartet werden kann.

Den named zu Starten zu bringen ist ziemlich einfach. In »etc/defaults/rc.conf« gehst Du einfach zur Zeile mit den Eintrag »named« und ersetzt No durch YES. Weitere Optionen können in der Zeile zwischen den Anführungszeichen definiert werden. Ich ziehe es (als Beispiel) vor, den named- Prozess als non- root- Prozess ablaufen zu lassen, damit ein non- Root- Accout den named- Prozess ausführen kann.

Zu named gehört auch eine Datei namens **ndc** , mit der der named- Prozess gesteuert werden kann. Kurz gesagt, kann **ndc** den named starten, stoppen und neu starten. Es kann aber auch noch eine Menge andere Dinge(Siehe auch die **ndc**- Manpage für weitere Details).

Das generelle Kommando ist **ndc**.
Als nächstes wollen wir, dass strider bei den Lookups zuerst bei sich selbst sucht. Wir tun das in zwei Schritten. Als erstes entscheiden wir die Reihenfolge der Namensauflösung. In einem kleinen Netzwerk hat jeder Host eine Kopie der »hosts«- Tabelle. Wir können damit weitermachen, dass wir zuerst »hosts« und dann DNS benutzen; wobei die Nutzung von DNS in grösseren Netzen weitaus einfacher ist. Gleich welcher Weg eingeschlagen wird; die Datei, in der das festgelegt wird, heisst »/etc/nsswitch.conf«:

```
. . .
group_compat:   nis
hosts:          files dns
netgroup:       files [notfound=return] nis
. . .
```

Die Zeile, um die wir ein wenig besorgt sind, ist hosts. Die Datei besagt letztlich, dass das System als erstes »/etc/hosts« durchsuchen soll, um die Namensübersetzung festzulegen. Der Eintrag auf der linken Seite, ist immer die erste Methode, nach der ein Name aufgelöst wird.

Die naächste Datei ist »/etc/resolv.conf«. Diese Datei ist die DNS- Auflösungsdatei. Das Format ist schön selbsterklärend, aber wir werden das trotzdem noch einmal irgendwie durchgehen.

```
domain diverge.org
search diverge.org
nameserver 192.168.1.1
```

Diese Datei erzählt dem Resolver, dass diese Maschine zur Domäne diverge.org gehört und dass sie, bevor sie anfängt, woanders zu suchen, erst einmal die eigene Domain nach Daten abgrasen soll. Der Nameserver ist 192.168.1.1. Um den Namerserver zu testen, setzen wir mal folgendes Kommando ab:

```
# host www.blah.net
```

Hier ist die Antwort von **host www.yahoo.com**:

```
www.yahoo.com is a nickname for www.yahoo.akadns.net
www.yahoo.akadns.net has address 216.32.74.50
www.yahoo.akadns.net has address 216.32.74.51
www.yahoo.akadns.net has address 216.32.74.52
www.yahoo.akadns.net has address 216.32.74.53
www.yahoo.akadns.net has address 216.32.74.55
```

Das Verfahren des Aufsetzen eines Clients ist das gleiche; einfach »/etc/resolv.conf« und »/etc/nsswitch.conf« einrichten.

nach oben

10.5. Installation eines Caching- Only- Nameservers

Ein »caching-only«- Nameserver besitzt keine lokalen Zonen. Alle Anfragen gehen zum root-Server und die Antworten werden im lokalen Cache abgelegt. Wenn die Anfrage noch einmal abgeschickt wird, kommt die Antwort schneller, weil die Daten schon im Servercache vorhanden sind. Weil so ein Server lokale Zonen nicht handeln kann, müssen die Namen der lokalen Rechner in der allseits bekannten »/etc/hosts« eingetragen werden.

Seit NetBSD mit allen Dateien kommt, die für einen Caching- Only- Server gebraucht werden, ist die Konfiguratiuon von DNS sehr einfach: Sie kann mit ein paar simplen Kommandos durchgeführt werden, ohne eine einzige Zeile in die Konfigurationsdateien zu schreiben.

> **Anmerkung:** Die Anzahl der Konfigurationsdateien variiert von zwischen den NetBSD- Versionen.

Das Programm, das den DNS- Server beinhaltet, ist der named- Daemon, der die »named.conf«- Datei benutzt, um zu starten. Die Standardatei liegt im »/etc/namedb«-Verzeichnis, aber der Daemon sucht im »/etc«- Verzeichnis danach; also fangen wir damit an, einen Link zu erstellen:

```
# ln -s /etc/namedb/named.conf /etc/named.conf
```

Der Nameserver ist damit gebrauchsfertig. Wir können dem System nun sagen, dass es das Programm benutzen soll, indem wir diese Zeile in der »/etc/resolv.conf« hinzufügen:

```
nameserver 127.0.0.1
```

Jetzt können wir den named starten:

```
# named
```

> **Wichtig:** Wir haben den Nameserver jetzt per Hand gestartet. Wenn er einmal getestet ist und wir sicher sind, dass er funktioniert, machen wir das in Zukunft mit

dem Eintrag in »/etc/rc.conf«

nach oben

10.5.1. Den Server testen

Der Server läuft. Wir sollten das mal mit einem **nslookup** testen:

```
# nslookup
Default server: localhost
Address: 127.0.0.1

>
```

Lass uns mal einen Hostnamen auflösen, Beispiel wäre www.lindloff.com

```
> www.lindloff.com
Server:  localhost
Address:  127.0.0.1

Name:    www.lindloff.com
Address:  162.67.198.5

Wenn Du eine Anfrage wiederholst,
ist das Ergebnis deutlich anders:

> www.lindloff.com
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
Name:    www.lindloff.com
Address:  162.67.198.5
```

Wie Du vielleicht bemerkt hast, ist die Adresse dieselbe, aber es kommt die Botschaft »Non-authoritatve answer« mit der Antwort. Das heisst, dass die Antwort nicht von einem für lindloff.com autorisierten Server kommt, sondern aus dem Cache Deines eigenen Servers.

Die Ergebnisse dieses ersten Tests besagen nichts anderes, als dass der Server korrekt arbeitet.

Wir können auch mal das »host«- Kommando versuchen. Die Antwort sollte dann so aussehen, wie diese hier:

```
# host www.lindloff.com
www.lindloff.com has address 192.67.198.5
```

Seitenanfang

Inhalt

Kapitel 11. Mail und news

Inhalt:

Dieses Kapitel beschreibt, wie NetBSD für den Gebrauch von E- Mail und News eingerichtet wird. Nur ein simples und sehr verbreitetes Verfahren wird an dieser Stelle beschrieben: Die Konfiguration eines Hosts, der über ein Modem mit einem Provider verbunden ist. Betrachte dieses Kapitel bitte als eine Fortsetzung von Kapitel 10, in dem eine einfachere Netzwerkkonfiguration beschrieben wird. Gerade diese eigentlich »einfache« Installation wird schwierig werden, wenn Du nicht weisst, wo Du anfangen sollst und wenn Du nur die einführende und/oder technische Dokumentation gelesen hast. In der Realität wirst Du herausfinden, dass einige Details eine echte Herausforderung darstellen können (Beispiel: Das Mapping von internen Hostnamen auf »echte«, also Internet- Hostnamen verlangt schon eine recht intime Kenntnis von sendmail). Eine grundsätzliche Beschreibung von Mail- und Newskonfiguration ist ausserhalb des Fokus' dieser Anleitung. Für eine grundsätzliche Beschreibung über die Funktionen von Mail und News solltest Du Dir ein richtig gutes Buch zum Thema Unix anschaffen. Eine Liste mit sehr guten Büchern findet sich auf der NetBSD-Website. Dieses Problem gibt es wegen der Riesenauswahl an möglichen Verbindungen und Konfigurationen und weil es nicht ausreicht, eine einzelne Datei zu bearbeiten die ein einzelnes Programm bedient. Du musst eine exakt passende Konfiguration mehrerer miteinander kooperierender Programme durchführen.

Dieses Kapitel beschreibt ausserdem die Konfiguration (nicht jedoch die Nutzung) zweier populärer Programme: Mutt als Mailer und tin für die News. Die Benutzung ist nicht beschrieben, weil sie einfach zu bedienen und gut dokumentiert sind. Auch hier gilt: mutt und tin sind nicht »das ultimative Tool«. Es gibt eine Menge Programme, die das Gleiche machen; aber ich denke, dass sie einen guten Ausgangspunkt bieten, weil sie sehr weit verbreitet, einfach und zuverlässig sind und nicht viel Platz auf Festplatte und im Arbeitsspeicher beanspruchen. Beides sind Konsolenprogramme. Wenn Du grafische Applikationen bevorzugst, gibt es für X ebenfalls eine grosse Auswahl.

In Kürze: Die Programme, die für die beschriebene Konfiguration gebraucht werden sind:

- sendmail
- fetchmail
- m4
- mutt
- tin

Von diesen Programmen sind nur sendmail und m4 auf dem Basissystem installiert; die anderen Programme können von der Package- Collection installiert werden.

Bevor Du weitermachst, solltest Du wissen, dass keines dieser Programme unersetzlich ist: Es gibt andere Applikationen, die das Gleiche tun; und es gibt viele User, die andere Programme bevorzugen. In den Mailinglisten wirst Du auf verschiedene Meinungen stossen. Du kannst auch verschiedene Strategien für das Senden und Empfangen von Mails verwenden: Die hier beschriebene ist nur ein Ausgangspunkt. Wenn einmal verstanden hast, wie das funktioniert, kann es sein, dass Du die Konfiguration ändern willst, um das System besser an Deine Bedürfnisse anzupassen oder eine andere Arbeitsmethode zu übernehmen.

Das andere Extrem des hier dargestellten Beispiels ist die Benutzung des Netscape-Communicators, der alles erledigt und Dich von der Notwendigkeit der Konfiguration vieler Komponenten befreit: Mit dem Communicator kannst Du im Web surfen, E-Mail lesen, empfangen und senden, und News lesen. Nebenbei ist die Konfiguration sehr einfach. Aber es ist ein Preis dafür zu bezahlen: Der Communicator ist ein in sich »geschlossenes« Programm, das nicht mit anderen Standard- Unixanwendungen zusammenarbeitet.

Eine andere Möglichkeit bietet sich durch Emacs, das man auch nutzen kann, um E-Mail und News zu lesen. Emacs muss Unix- Usern nicht weiter erklärt werden; falls Du ihn nicht kennst: Er ist ein erweiterbarer Editor(Obwohl das als Beschreibung unzureichend ist), der mit einer kompletten Arbeitsumgebung kommt, eben auch zum Lesen von Mails und News benutzt werden kann und viele andere Funktionen bietet. Für viele Leute ist Emacs das einzige Programm, das sie für ihre gesamte Arbeit benutzen. Die Konfiguration von Emacs für das Lesen von Mail und News wird im Emacs- Manual beschrieben.

Im Rest diese Kapitels werden wir mit einem Host arbeiten, der über PPP über ein serielles Modem über einen Provider mit dem Internet verbunden ist.

- Der Name des lokalen Hosts ist »ape« und das interne Netzwerk heisst »isetti,nett, was bedeutet, dass der FDQN (Full qualified Domain Name) »ape.isetti.net ist.
- Der Loginname des Users ist »carlo«.
- Der Name des Providers ist BigNet.
- Der Mailserver des Providers ist "mail.bignet.it".
- Der Newsserver des Providers ist "news.bignet.it".
- Der Useracoount des Benutzers (carlo) beim Provider ist »alan; sein Passwort »pZY9o«

Als Erstes Grundsätzliches zur Terminologie:

MUA (mail user agent)

> Das Programm, mit dem Mail geschrieben und gelesen wird; Beispiele: mutt, elm und pine, aber auch die einfache Mail- Applikation, die mit dem Basissystem installiert wird.

MTA (mail transfer agent)

> Das Programm, das die Mail zwischen zwei Hosts transferiert und auch lokal benutzt wird (auf dem selben Host). Der MTA entscheidet über den Pfad, den die Mail verfolgen soll, um ans Ziel zu gelangen. Auf BSD- Systemen ist das (nicht nur) sendmail.

MDA (mail delivery agent)

> Ein Programm, das gewöhnlich vom MTA benutzt wird, um die Mail zuzustellen;

beispielsweise legt es die Mail für die Empfänger physikalisch in deren Mailbox ab. Sendmail nutzt einen oder mehrere MDAs, um Mail auszuliefern.

Bild 11-1 zeigt das Mailsystem, das aufgesetzt werden soll. Zwischen dem lokalen Netz (oder dem einzelnen Host) und dem Provider gibt es eine PPP- Verbindung. Die Seifenblasen an der dicken Linie sind die Programme, die vom User manuell aufgerufen werden. Die anderen sind die Programme, die vom System gestartet werden. Die eingekreisten Nummern bezeichnen die logischen Schritte eines Mailzyklus'.

1. In Schritt 1 wurde die Mail vom POP- Server des Providers mit fetchmail abgeholt, das sendmail benutzt, um die Nachrichten in der Mailbox des Users abzulegen.

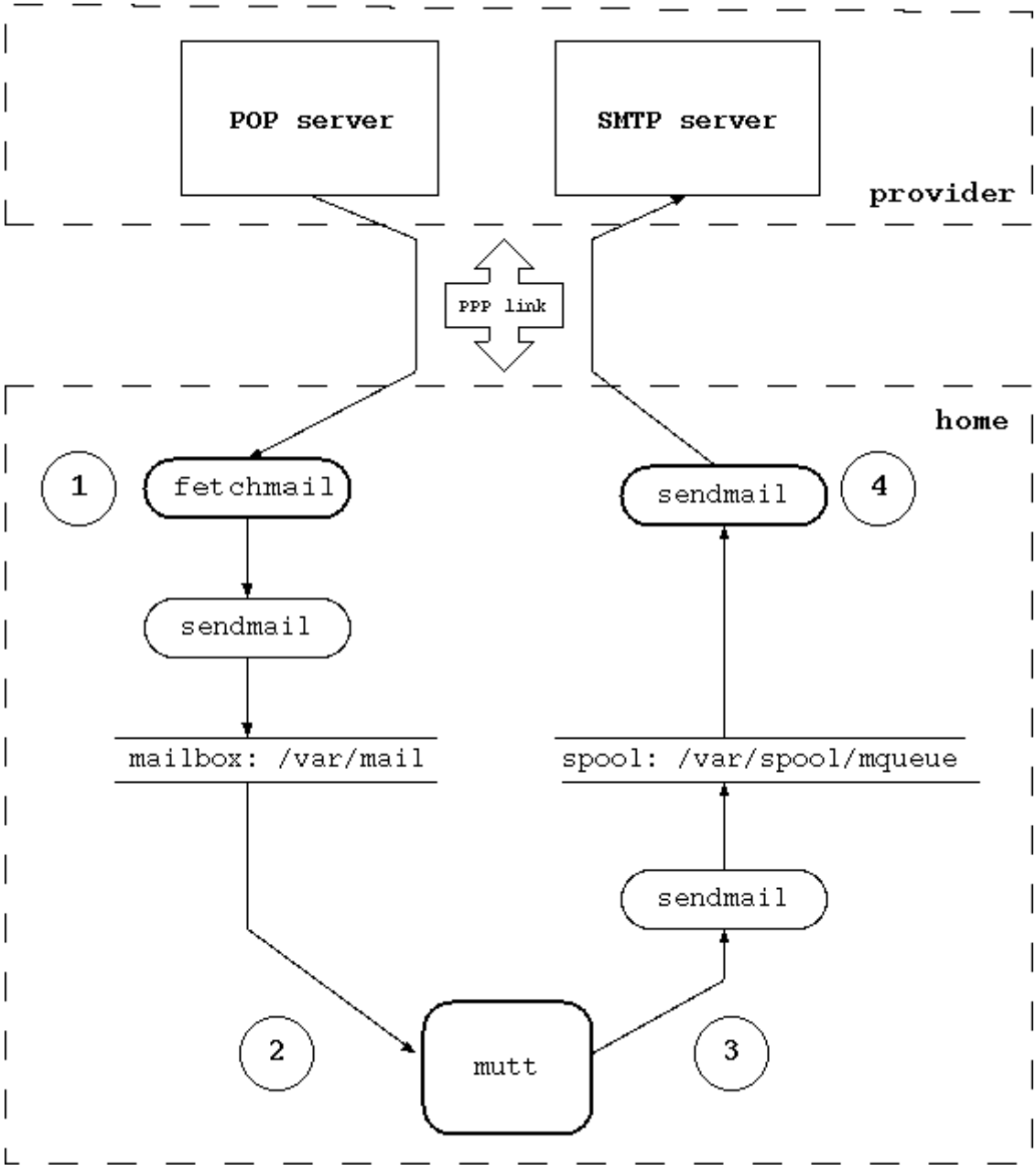
2. In Schritt 2 startet der User mutt (oder einen anderen MUA), um die Mail zu bearbeiten und neue Mail zu schreiben.

3. In Schritt 4 »versendet« der User die Mail von Mutt aus. Die Mail wird im Spool- Bereich zwischengelagert.

4. In Schritt 4 ruft der User sendmail auf, damit dieser die Nachrichten an den SMTP- Server des Providers weiterleitet, der diese an das eigentliche Ziel weiterleitet (möglicherweise auch durch weitere Mailserver). Der SMPT- Server des Providers agiert als Relaisstation (engl. Relay) für unsere Mail.

Die Verbindung mit dem Provider muss nur für die Schritte 1 und 4 bestehen; für die anderen Schritte ist das nicht nötig.

Bild 11.1: Struktur eines Mailsystems:



nach oben

11.1. sendmail

Wenn ein MTA wie sendmail eine Nachricht zustellen soll, macht er das direkt, wenn es sich um eine Nachricht im lokalen Natz handelt. Wenn die Botschaft für eine andere Domain bestimmt ist, muss der MTA die Adresse des Mailservers der anderen Domain harausfinden. Sendmail nutzt den DNS- Dienst für diesen Zweck, wie er in Kapitel 10 beschrieben ist (Wird als MX-Record vom DNS- Server gelagert) und liefert dann die Nachricht an den Zielserver.

Der meistverwendete MTA in BSD ist vermutlich sendmail. Sendmail wird mit einer Reihe von Konfigurationsdateien und Datenbanken gesteuert, von denen »/etc/sendmail.cf« die wichtigste ist. Wenn Du kein Sendmail- Experte bist (Wer ist das schon???), bist Du besser damit bedient, diese Datei nicht direkt zu bearbeiten. Stattdessen kannst Du das mit Hilfe einiger vordefinierter Makros und dem »m4«- Präprozessor machen, was die Konfiguration (meistens) stark vereinfacht.

> **Hinweis:** Vor Version 1.5 lagen die Konfigurationsdateien nicht in »/etc/mail«, sondern in »/etc«.

Auch wenn man die Makros benutzt, ist die Konfiguration von sendmail nichts für schwache Gemüter. Die nächsten Sektionen beschreiben ein Beispiel, das modifiziert werden kann, wenn man andere Bedürfnisse hat oder sie anderen Konfigurationen anpassen muss. Wenn Du nur über ein Modem mit dem Internet verbunden bist, könnte diese Beispielkonfiguration schon hinreichen, um Deinen Bedarf zu decken: Ersetze einfach die fiktiven Daten durch Deine realen.

Das erste Problem, das zu lösen ist, ist dass das lokale Netz, mit dem wir arbeiten, ein internes Netzwerk ist, das nicht direkt vom Internet aus erreichbar ist. Das bedeutet, dass die intern benutzten Namen keine Beziehung zum Internet haben; kurz gesagt: »ape.isetti.net« kann von einem externen Host nicht erreicht werden. Niemand wird in der Lage sein, eine Mail, die mit dieser Return- Adresse verschickt wird, zu beantworten(Viele Mailsysteme werden diese Nachricht zurückschicken, weil sie von einem unbekannten Host kommt). Die echte Adresse, also die, die für jeden sichtbar ist, wird prinzipiell vom Provider vergeben. Deshalb ist es nötig, die lokale Adresse » carlo@ape.isetti.net« in die reale Adresse alan@bignet.it zu konvertieren. Sendmail erledigt diesen Job; sofern es korrekt konfiguriert ist, wenn es die Nachrichten weiterleitet.

Es kann aber auch sein, dass Du sendmail so konfigurieren willst, dass es E- Mails an den Mailserver des Providers weiterleitet und diesen eben als Relay benutzt. In der Konfiguration, die hier beschrieben wird, kontaktiert sendmail den Empfänger nicht direkt (Wie vorhin beschrieben), sondern leitet die gesamte Mail an den Mailserver des Providers weiter.

> **Wichtig:** Des Providers Mailserver agiert hier als Relay, was bedeutet, dass alle Mail, die nicht für seine Domäne bestimmt ist, an einen anderen Mailserver weitergeleitet wird. Er arbeitet also als Vermittler zwischen zwei Servern.

Weil die Verbindung mit dem Provider nicht immer aktiv ist, ist es nicht notwendig, sendmail als Daemon in »/etc/rc.conf« zu starten: Du kannst sendmail mit der Zeile «sendmail=NO« abschalten; sieh aber vorher nach, ob da nicht schon ein Eintrag in der Datei vorhanden ist. Als Konsequenz daraus wird es nötig, sendmail manuell zu starten, wenn Du die Mail zum Provider schicken willst. Lokale Mail wird korrekt ausgeliefert, auch wenn sendmail nicht als Daemon aktiv ist.

So lasset und mit der Sendmail- Konfiguration beginnen... (gar biblisch, gar biblisch)

nach oben

11.1.1. Konfiguration mit genericstable

Diese Art der Konfiguration benutzt die Daei »/etc/mail/genericstable«, in der das Mapping der Hostnamen durch Sendmail konfiguriert wird (Die lokalen Hostnamen werden also überschrieben).

Der erste Schritt ist, die Datei »genericstable« zu schreiben. Beispiel:

```
carlo:          alan@bignet.it
root:           alan@bignet.it
news:           alan@bignet.it
```

Der Funktion zuliebe muss »genericstable« mit folgendem Kommando übersetzt werden:

```
# /usr/sbin/sendmail -bi -oA/etc/mail/genericstable
```

Jetzt ist die Zeit reif, den Prototypen einer Konfigurationsdatei zu erschaffen.

```
# cd /usr/share/sendmail/m4
```

Die neue Sendmail- Konfigurationsdatei, die wir »mycf.mc« nennen sieht so aus:

```
divert(-1)dnl
include(`../m4/cf.m4')dnl
VERSIONID(`mycf.mc created by carlo@ape.insetti.net May 18 2001')dnl
OSTYPE(bsd4.4)dnl

dnl # Settings for masquerading.  Addresses of the following types
dnl # are rewritten
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 # The following feature is useful if sendmail is called by
dnl # fetchmail (which is usually the case.)  If sendmail cannot
dnl # resolve the name of the sender, the mail will not be delivered.
dnl # For example:
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 is useful with some MUA, which send
dnl # mail as, for example:
dnl #   MAIL FROM:<carlo>
FEATURE(accept_unqualified_senders)dnl

dnl # Mail for `smtp' mailer is marked `expensive' (`e' flag):
dnl # instead of connecting with the relay, sendmail puts it in
dnl # a queue for delayed processing.
dnl # Sendmail starts complaining about undelivered messages after
dnl # 16 hours.
define(`SMTP_MAILER_FLAGS',`e')dnl
define(`confCON_EXPENSIVE',`True')dnl
define(`confTO_QUEUEWARN', `16h')dnl

dnl # For european users
define(`confDEF_CHAR_SET',`ISO-8859-1')dnl

dnl # Enable the following three lines to use procmail as a local
dnl # delivery agent.  The third line is optional, only the first
dnl # two are required.
dnl # define(`PROCMAIL_MAILER_PATH', /usr/pkg/bin/procmail)dnl
dnl # FEATURE(local_procmail)dnl
```

```
dnl # MAILER(procmail)dnl

dnl # The following two mailers must always be defined
MAILER(local)dnl
MAILER(smtp)dnl
```

> **Hinweis:** In dieser Datei ist alles, was hinter einem »dnl« erscheint, ein Kommentar, der von m4 ignoriert wird.

Diese Konfiguration erzählt Sendmail, dass es die Adressen des Typs »ape.isetti.net« mit den realen Namen in der »/etc/genericstable«- Datei überschreiben soll. Es besagt auch, dass die Mail an den Server »mail.bignet.it« weitergeleitet werden soll. Was die Optionen bedeuten sollen, ist in »/usr/share/sendmail/README« beschrieben.

Weil es für Dich angebracht ist, eine eigene Version der Beispielkonfiguration zu erstellen, musst Du zwei Zeilen ändern, um Deine realen Daten einzutragen:

```
GENERICS_DOMAIN(ape.insetti.net ape)dnl
define(`SMART_HOST',`mail.bignet.it')dnl
```

Zu guter Letzt muss die neue Konfigurationsdatei erzeugt werden, nachdem Du die alte Version gesichert hast:

```
# cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.bak
# m4 mycf.mc > /etc/mail/sendmail.cf
```

> **Anmerkung:** Im »usr/share/sendmail/cf«- Verzeichnis gibt es eine Datei namens »netbsd-proto.mc, die die Default- Konfiguration von »/etc/mail/sendmail.cf« erstellt, die mit der Distribution mitkommt. Mit dem »make«- Kommando kann sie neu gebaut werden, fall das nötig werden sollte.

Eine weitere wichtige Datei ist »/etc/mail/aliases«, die in der Default- Konfiguration erhalten bleiben kann. Allerdings ist es immer noch nötig, das folgende Kommando abzusetzen:

```
# newaliases
```

Jetzt sollte alles fertig sein, um mit dem Senden von Mails beginnen zu können.

11.1.2. Testlauf der neuen Konfiguration

Sendmail ist jetzt fertig konfiguriert und damit gebrauchsfertig; aber bevor Du damit anfängst, Mails durch die Gegend zu schicken, ist es besser, ein paar simple Testläufe durchzuführen. Als Erstes senden wir eine lokale Mail; wobei wir dieses Kommando benutzen:

```
$ sendmail -v carlo
Subject: test

Prova
.
carlo... Connecting to local...
```

```
carlo... Sent
```

Bitte folge exakt diesem Beispiel: Lass die Zeile hinter dem Subject (Betreff) leer und beende die Nachricht einfach mit einem Punkt. Du solltest jetzt in der Lage sein, die Nachricht mit Deinem Mail- Client lesen und das »From« identifizieren zu können: Das »From«- Feld wurde korrekt überschrieben.

```
From: alan@bignet.it
```

Als nächstes kannst Du die Überschreibungsregeln direkt ändern, indem Du sendmail mit der Option »-bt« startest. Dieser Modus zeigt, wie sendmail eine Adresse ignoriert und wie diese nach den Regeln der Konfigurationsdatei überschrieben wird. Es ist so auch möglich, andere Tests durchzuführen und einige Informationen zu bekommen.

```
$ sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Du kannst die Hilfe mit dem »?«- Kommando anzeigen lassen.

Zuerst einmal prüfen wir, ob die »genricstable«- Datei korrekt arbeitet:

```
/map generics carlo
map_lookup: generics (carlo) returns alan@bignet.it
```

Hier scheint alles in Ordnung zu sein: Sendmail hat den lokalen Namen und sein reales Gegenstück gefunden.

Jetzt können wir das Überschreiben der Absenderadresse mit den folgenden Kommandos prüfen:

```
/tryflags ES
/try smtp carlo@ape.insetti.net
```

Das Ergebnis sollte so aussehen:

```
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
>
```

Wie Du sehen kannst, wurde die lokale Adresse in die reale Adresse übersetzt, die in Deinen Mails erscheint, wenn sie das System verlassen.

Du bekommst mit folgendem Kommando ein ähnliches Ergebnis:

```
/try smtp carlo
```

Wir können das Überschreiben des Headers mit diesen Kommandos ebenfalls überprüfen:

```
/tryflags HS
/try smtp carlo@ape.insetti.net
```

nach oben

11.1.3. Einen anderen MTA verwenden:

Ab Version 1.4 von NetBSD wird sendmail nicht mehr direkt aufgerufen:

```
$ ls -l /usr/sbin/sendmail
lrwxr-xr-x  1 root  wheel  21 Nov  1 01:14 /usr/sbin/sendmail@ -> /usr/
```

Der Zweck von mailwrapper ist der, die Benutzung eines alternativen MTA zuzulassen (Beispiel: postfix oder Exim). Wenn Du einen anderen Mailer verwenden willst, solltest Du Dir mal die »mailwrapper (8)«- und die »mailer.conf(5)«- Manpages anschauen, die sehr leicht zu verstehen sind.

nach oben

11.2. fetchmail

Die Mail wird von Provider empfangen und nicht automatisch an den lokalen Host übertragen; deshalb muss man sie downloaden. Fetchmail ist ein sehr populäres Programm, das die Mail von einem Remote- Server lädt und diese an das lokale System zur Auslieferung weiterleitet (meistens passiert das mit sendmail). Es ist leistungsfähig und einfach zu bedienen: Nach der Installation muss die Datei »/.fetchmailrc« erstellt werden und schon ist das Programm gebrauchsfertig. »./fetchmailrc« beinhaltet ein Passwort; also sind entsprechende Zugriffsrechte für die Ausführung Voraussetzung.

Beispiel für .fetchmailrc:

```
poll mail.bignet.it
protocol POP3
username alan there with password pZY9o is carlo here
flush
mda "/usr/sbin/sendmail -oem %T"
```

> **Anmerkung:** Die letzte Zeile(»mda...«) wird nur verwendet, wenn sendmail nicht als Daemon auf dem System aktiv ist. Bitte denke daran, dass der Mailserver, der in dieser Datei verwendet wird, nicht notwendigerweise derselbe ist, der als Mail- Relay von sendmail benutzt wird.

Mit dem folgenden Kommando kann die Mail heruntergeladen und an das lokale Mailsystem ausgeliefert werden:

```
$ fetchmail
```

Die Nachrichten können jetzt mit mutt gelesen werden.

nach oben

11.3. Lesen und Schreiben von mails mit mutt

Mutt ist eines der populärsten Mailprogramm, die es so gibt: Es ist ein »Leichtgewicht«, einfach zu bedienen und bietet eine Mange Features. Die Manpage beschreibt nur das nackte Skelett; die wirkliche Dokumentation findet sich in »/usr/share/doc/mutt/«, ganugenommen in »manual.txt«.

Die Konfiguration von mutt wird in der »/.muttrc«- Datei definiert. Der einfachste Weg, sie zu erstellen ist der, einfach die »muttrc«- Datei (gewöhnlich in »/usr/pkg/etc/muttrc«) in das Home-Verzeichnis zu kopieren und anzupassen. Das folgende Beispiel zeigt, wie man an Ergebnisse kommt:

- Eine Kopie verschickter Mail speichern
- Ein Verzeichnis und zwei Dateien für eingehende und ausgehende Mail definieren, die von mutt gespeichert werden sollen (In diesem Baispiel ist das »/Mail« und die Dateien sind »/incoming« und »outgoing«).
- Einige Farben definieren.
- Ein Alias definieren

```
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>
```

mutt starten:

```
$ mutt
```

> **Notiz:** Mutt unterstützt farbige Drstellung; aber das hängt auch von den Einstellungen des Terminals ab. Unter X kannst Du das mit xterm-color machen:
>
> ```
> TERM=xterm-color mutt
> ```

nach oben

11.4. Eine Strategie, um Mail zu empfangen

In dieser Sektion beschreiben wir ein einfache Methode, nach der Mails empfangen und gelesen werden können. Die Verbindung zum Provider ist nur so lange aktiv, wie sie gebraucht wird, um die Mails herunterzuladen. Das Lesen der Mail findet Off- Line statt.

1. Aktiviere die Verbindung zum Provider
2. Starte fetchmail.
3. Deaktiviere die Verbindung.
4. Lese die Mail mit mutt.

nach oben

11.5. Strategie zum Verschicken von Mail

Wenn die Mail geschrieben und von mutt »gesendet« worden ist, müssen die Mails mit Hilfe von sendmail zum Provider übertragen werden. Mail wird von mutt aus mit dem »y«- Kommando abgeschickt. Aber damit passiert das Senden nicht wirklich; Die Nachrichten werden im Spool-

Bereich abgelegt. Wenn sendmail nicht als Daemon aktiviert ist, müssen die Nachrichten manuelle verschickt werden; sonst bleiben sie auf der Harddisk liegen, bis Du grün-lilagepunktet wirst. Die notwendigen Schritte sind diese hier:

1. Schreibe die Mail mit mutt.
2. Baue die Verbindung zum Provider auf.
3. Mit dem Kommando »sendmail -q -v« schaufelst Du die zwischengespeicherten Daten auf den Server des Providers.
4. Beende die Verbindung zum Provider.

nach oben

11.6. Fortgeschrittene Mail- Tools

Wenn du anfängst, Mail zu benutzen, wirst Du nicht allzugrosse Anforderungen an Dein System stellen und meistens mit der beschriebenen Standardkonfiguration auskommen. Aber für viele Nutzer wird die Anzahl der täglichen Botschaften mit der Zeit ansteigen und damit eine besser organisierte Konfiguration der Mail- Ablage nötig werden. Vielleicht willst Du die Mail noch weiter unterteilen, also in verschiedenden Mailboxen nach Themen oder Herkunft sortiert zwischenlagern. Wenn Du beispielsweise eine Mailingliste abonnierst, wirst Du täglich etliche Nachrichten empfangen und sie vielleicht von den anderen Mails getrennt lagern wollen. Du wirst es bald als lästig empfinden, Deine Zeit für zeitraubende manuelle Aktionen aufzuwenden, die sich ohnehin täglich wiederholen, nur um Deine Mailbox zu pflegen.

Also warum eigentlich dauernd die selben Tätigkeiten manuell wiederholen, wenn es ein Programm gibt, das diesen Job automatisch erledigt? Es gibt zahlreiche Tools, die Du Deinem Mailsystem hinzufügen kannst, um die Flexibilität zu erhöhen und die Nachrichten automatisch zu bearbeiten. Die bekanntesten und meistgenutzten Programme sind diese hier:

- procmail ist ein erweiterter MDA mit einem einfachen Filter für lokale Mail, der eingehende Mail automatisch nach benutzerdefinierten Regeln bearbeitet. Er arbeitet harmonisch mit Sendmail zusammen.
- Metamail: Ein Tool, das Attachments verarbeitet.
- Formail: Ein Mailformatierer

Im Hauptteil dieser Sektion wird eine grundsätzliche Konfiguration von procmail für einen sehr verbreiteten Fall gezeigt: Das Zustellen von Mails in einer benutzerdefinierten Mailbox, wenn es sich dabei um Mails aus einer Mailingliste handelt. Die Konfiguration von Sendmail wird so geändert, dass procmail direkt aufgerufen werden kann (procmail wird der »lokale Mailer«, der von sendmail benutzt wird.). Ausserdem erstellen wir eine eigene Konfigurationsdatei für procmail.

Zuerst einmal muss procmail mit Hilfe des Package- Systems installiert werden (»mail/procmail«).

Dann wird die Konfiguration von sendmail geändert, damit procmail als lokaler Mailer verwendet werden kann. Diese Zeilen in der »mycf.m4«- Prototypendatei müssen hinzugefügt werden (bzw.das Kommentarkommando muss entfernt werden). Anschliessend muss die sendmail-Konfigurationsdatei neu erstellt werden.

```
define(`PROCMAIL_MAILER_PATH', /usr/pkg/bin/procmail)dnl
FEATURE(local_procmail)dnl
MAILER(procmail)dnl
```

Die erste Zeile definiert den Pfad, in dem das procmail- Programm zu finden ist (Wo das genau ist, kannst Du mit »which procmail« abfragen). Die zweite Zeile sagt sendmail, dass es procmail für die lokale Auslieferung von Mail benutzen soll. Zeile drei fügt procmail zur Liste der lokalen Mailer hinzu, mit denen sendmail zusammenarbeitet(Diese Zeile ist optional).

Der letzte Schritt ist das Erstellen der procmail- Konfigurationsdatei, die die Regeln für die lokale Auslieferung enthält.

Als Beispiel sagen wir, dass Du eine Liste auf von »roses« abonniert, dessen Adresse »roses@flowers.org« ist und dass jede Nachricht von diesem Server folgende Zeile im Header enthält:

```
Delivered-To: roses@flowers.org
```

Die procmail- Konfiguration (».procmailrc«) sieht so aus:

```
PATH=/bin:/usr/bin:/usr/pkg/bin
MAILDIR=$HOME/mail
LOGFILE=$MAILDIR/from

:0
* ^Delivered-To: roses@flowers.org
roses_list
```

Die voherige Datei besitzt nur eine Regel, die mit der Zeile anfängt, die mit »0:« beginnt. Die folgende Zeile identifiziert alle Nachrichten, die den String » Delivered-To:roses@flowers.org« beinhalten. Die letzte Zeile besagt nichts anderes, als dass die ausgesucheten Nachrichten in der »roses_lists«- Mailbox abgelegt werden sollen (die Du in $MAILDIR erstellt haben solltest.). Der Rest geht in die Default- Mailbox. Denke daran, dass $MAILDIR dasselbe Verzeichnis ist, das Du mit mutt konfiguriert hast:

```
set folder="~/Mail"
```

Natürlich ist diese Mailingliste nur ein Beispiel; procmail ist ein sehr vielseitiges Tool, mit dem man die Mails nach vielen Kriterien filtern kann. Wenn Du mehr wissen willst, sieh mal in den Manpages procmail(1), procmailrc(5) und procmailex(5) nach (In der letzten finden sich etliche Beispiele für Konfigurationsdateien).

Du kannst überprüfen, ob procmail als lokaler Mailer verwendet wird, wenn Du Letzteres im Testmodus laufen lässt:

```
$ /usr/sbin/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
```

Diese Kommando zeigt die Liste der Mailer an, die sendmail bekannt sind:

```
> =M
```

Du solltest eine Zeile wie diese hier finden:

```
mailer 3 (local): P=/usr/pkg/bin/procmail S=EnvFromL/HdrFromL ...
```

nach oben

11.7. News mit tin

Der Terminus NEWS beschreibt ein Sortiment von Artikeln aus den USENET- Newsgruppen; ein Dienst, der im Internet angeboten wird. Jede dieser Newsgroups bietet einem diverse Artikel zu einem bestmmten Thema. Das Lesen eine Newsgroup unterscheidet sich vom Abonnieren einer Mailigliste: Wenn Du Dich in so eine Liste einträgst, empfängst Du die Artikel per Mail und Du liest sie mit einem Standard- Mailer wie mutt, mit dem Du auch Deine Antworten verschickst. Stattdessen werden News direkt von einem News- Server mit einem separaten Programm gelesen, das auch »NEWSREADER« genannt wird; beispielsweise tin. Mit tin kannst Du die für Dich interessanten Newsgroups abonnieren, an denen Du interessiert bist und deren Verlauf verfolgen.

> **thread:** Ein Thread ist eine Sequenz von Artikeln, die alle mit einem Artikel zu tun haben, den wir als »Original« bezeichnen können. Kurz: Eine Nachricht geht an die Gruppe. Jemand antwortet darauf und wieder ein anderer beantwortet die Antwort undsoweiter. Dabei entsteht eine Baumstruktur aus Nachrichten und Antworten: Ohne einen Newsreader ist es unmöglich, die geanue Reihenfolge der Nachrichten zu verfolgen und zu verstehen.

Nach der Installation von tin (gewöhnlich von der Packages- Kollektion) ist das einzige, was Du tun musst, den Namen des NNTP- Servers in die Datei »/usr/pkg/etc/nntp/server« einzutragen. Beispiel:

```
news.bignet.it
```

Wenn das einmal getan ist, kann das Programm mit »rtin« gestartet werden. Auf dem Bildschirm wird sowas ähnliches wie dieses hier angezeigt:

```
$ rtin
Connecting to news.bignet.it...
news.bignet.it InterNetNews NNRP server INN 1.7.2 08-Dec-1997 ready (po
Reading groups from active file...
Checking for new groups...
Reading attributes file...
Reading newsgroups file...
Creating newsrc file...
Autosubscribing groups...
Reading newsrc file...
```

Übe Dich in Geduld, wenn Du Dich zum ersten Mal mit einem Newsserver verbindest, weil tin eine immens grosse Liste mit Newsgroups herunterlädt, die Du abonnieren kannst, was einige Zeit in Anspruch nehmen kann. Wenn der Download getan ist, wird der Hauptbildschirm des Programms angezeigt; normalerweise werden hier keine Gruppen angezeigt. Um diese zu sehen drücke »y«. Um eine Gruppe zu abonnieren, gehe auf den Namen der Gruppe und

drücke »y«.

Wenn Du erst einmal verschiedene Newsgroups abonniert hast, kannst Du tin mit »rtin -Q« viel schneller starten. Die Such nach neuen Gruppen ist ausgeschaltet (»-Q«), nur aktive Gruppen werden durchsucht (»-n«) und die Beschreibung der Newsgroup wird nicht geladen (»-d«): es wird unmöglich sein,. Das »y«(yank)- Kommando in tin zu benutzen. Wenn tin mit dieser Option gestartet wird, kann es Dir nicht sagen, ob die Newsgroup moderiert wird.

> **Hinweis:** Wenn Du Dich von einem internen Nertwerk aus verbindest (wie in unserem Beispiel), wird die Adresse am Anfang der Nachricht falsch sein, weil es die interne Adresse ist. Um das Problem zu lösen, benutze die Option "mail_adress" in der tin- Konfiguratonsdatei (~/.tin/tinrc) oder setze die REPLYTO Umgebungsvariable.

nach oben

Inhalt

Kapitel 12. Konsolentreiber

Inhalt:

In den NetBSD- Versionen vor 1.5 hatte der Benutzer die Wahl zwischen zwei verschiedenen Treibern für Bildschirm und Tastatur: pccons (speziell für i386er- Maschinen) und pcvt. In Version 1.4 wurde der neue Wscons- Multiplattformtreiber eingeführt, der die beiden alten Implementierungen ersetzt(die aber noch unterstützt werden).

12.1. wscons

Wscons ist der neue NetBSD- Konsolentreiber. Er bietet virtuelle Terminals, Unterstützung für internationale Tastaturen, kümmert sich um die Maus und anderes. Die Talente von wscons variieren von Portierung zu Portierung (wscons gibt es nicht auf allen): Die 386er- Version ist allerdings ziemlich reichlich mit Features gesegnet.

Am Ende der Installation ist wscons standardmässig aktiv und es muss nichts getan werden, um es zu benutzen (Hey, Lionel, don't workin' too hard...). In Version vier sind die virtuellen Konsolen nach einer Neuinstallation nicht aktiviert. Um sie in Betrieb zu nehmen, lies bitte in Kapitel vier nach. Der Rest dieser Sektion beschreibt die wscons- Optionen in der Kernel-Konfugurationsdatei.

Wenn Du Deinen eigenen Kernel zu backen gedenkst, musst Du die für wscons relevanten Optionen aktivieren (»#« entfernen) und die Optionen von »pcvt« und »pccons« deaktivieren (Beides gleichzeitig geht nicht!). Beispiel:

```
#pc0    at isa? port 0x60 irq 1   # pccons generic PC console driver
#vt0    at isa? port 0x60 irq 1   # PCVT console driver
```

In der Konfigurationsdatei des Kernels kannst Du ausserdem aus USA- Sicht ausländische Tastaturen als Standard setzen. Hier das teutonische Beispiel:

```
options     PCKBD_LAYOUT="KB_DE"
```

Achtung, Italiener: Das Standard- Layout dieser Tastatur ist nur bedingt zum Programmieren geeignet, weil da ein paar Zeichen fehlen. Wie das zu ändern ist, steht in Kapitel 4.

Die Anzahl der vorinstallierten virtuellen Konsolen wird mit der folgenden Option geregelt:

```
options     WSDISPLAY_DEFAULTSCREENS=4
```

Andere Konsolen können hinzugefügt werden, indem man in den relevanten Zeilen in der »/etc/wscons.conf«- Datei die Kommentarmarke entfernt (»#«). Sie muss aus den Zeilen entfernt werden, die mit »screen x« anfangen. Wir legen mal eine fünfte Konsole zu unseren vorhandenen vier Exemplaren an:

```
# 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
```

Das »rc.wscons«- Script übersetzt jede der nicht auskommentierten Zeilen in einen Aufruf des wsconscfg- Kommandos: Die Spalten mutieren zu den Parametern des jeweiligen Aufrufs. Die »idx«- Spalte wird der »index«- Parameter; die »screen«- Spalte wird der »-t (type)«- parameter, der den Typ des Bildschirms beschreibt: Zeilen und Spalten, Anzahl der Farben usw. und die »emul«- Spalte wird zum »-e (emul)«- Parameter, der die Art der Emulation definiert. Beispiel:

```
screen 3        -       vt100
```

Wird zu diesem Aufruf:

```
wsconscfg -e vt100 3
```

> **Hinweis:** Es ist möglich, dass ein (allerdings harmloser) Konflikt zwischen den Konsolen, die vom Kernel angelegt werden und denen, die beim Booten durch »/etc/wscons.conf« hinzufügt werden, auftritt. Wenn das System während des Bootens versucht, einen bereits vorhandenen Bildschirm zu finden, wird diese Nachricht angezeigt:
>
> ```
> wsconscfg: WSDISPLAYIO_ADDSCREEN: Device busy
> ```
>
> Das Problem kann man lösen, indem die betreffenden Zeilen in »/etc/wscons.conf« auskommentiert werden.

Die virtuelle Konsole muss ausserdem in »/etc/ttys« aktiv sein; Beispiel:

```
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
...
```

Die Zeile

```
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
```

»/etc/ttys« wird vom X- Server benutzt, um ein freies Terminal zu finden. Um einen Schirm zu nutzen, der nicht die Nummer vier trägt, muss ein Parameter in der Form *vt*n vom X- Server ausgeführt werden (*n* ist die Zahl auf der Funktionstaste, die benutzt wird, um den Bildschirm für X zu aktivieren).

Als Beispiel benutzen wir hier mal die Konsole Nummer sieben, die in »etc/wscons.conf« aktiviert wurde. X kann also mit »vt8« gestartet werden. Wenn Du xdm in Gebrauch hast, musst Du »usr/X11R6/lib/xdm/Xserver/« bearbeiten. Beispiel:

```
:0 local /usr/X11R6/bin/X +kb dpms -bpp 16 dpms vt8
```

Für xdm3d ist das ein anderer Pfad: /usr/X11R6/share/xdm3d/Xservers. nach oben

12.1.1. 50- Zeilen- Textmodus mit wscons

Ein Textmodus mit 50 Zeilen kann seit Version 1.4.1 von NetBSD benutzt werden. Dieser Modus ist in »/etc/wscons.conf« aktiviert. Diese Zeile muss unkommentiert dastehen:

```
font ibm  -  8  ibm  /usr/share/pcvt/fonts/vt220l.808
```

Dann müssen folgende Zeilen angepasst werden:

```
#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
```

Diese Konfiguration erlaubt acht Bildschirme, auf die mittels C*trl-Alt-Fn* zugegriffen werden kann (n variiert von 1 bis acht). Die korrespondierenden Geräte sind »ttyE0« bis »ttyE7«. Um sie betriebsfähig zu machen und sich einloggen zu können, muss »/etc/ttys« geändert werden:

```
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
```

Es ist nicht möglich, die 80x25- Einstellung von Schirm 0 zu modifizieren; möglicherweise ist das gedacht, um im Problemfall immer einen funktionierenden Bildschirm zu haben.

nach oben

12.2. pccons

...ist der Konsolentreiber, der sich auf der i386- Installationsdiskette befindet. Er bietet weder virtuelle Konsolen noch Programme zur Konfiguration; hat aber den Vorteil, dass er sehr schlank ist.

nach oben

12.3. pcvt

Pcvt ist eine VT220- Terminal- Emulation, die über mehr Features verfügt als das simple pccons. Es unterstützt internationale Tastaturen und virtuelle Konsolen (mit Ctrl-Alt_Fn). Um »pcvt« zu aktivieren, muss das Kommentarzeichen aus folgenden Zeilen entfernt werden.

```
# Enable only one of the following lines
#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
```

Um eine nicht- US- Tastatur zu benutzen, muss man das beim Booten aktivieren. Ausserdem muss das korrekte Terminal ausgewählt werden. Beispiel:

```
/usr/local/bin/kcon -m i2
TERM=pcvt25;    export TERM
```

/etc/ttys muss ausserdem geändert werden. Beispiel:

```
#console "/usr/libexec/getty Pc"   pcvt25  on secure
ttyv0    "/usr/libexec/getty Pc"   pcvt25  on secure
```

> **PCVT, italienische Tastatur:** Die Definition der »i2«- tastatur ist nicht korrekt. Die Datei /sys/arch/i386/isa/pcvt/Util/keycap/keycap.src muss geändert werden. Das hier ist eine funktionierende Variante, getestet mit NetBSD 1.3.3
>
> ```
> i2|italy142|Italian 142 mapping:\
>           :A8={:A9=[:A10=]:A11=}:\
>           :A12=`:A13=~:\
>           :A17=@:A18=#:\
>           :tc=italy141:
> ```

Die Einstellungen für die internationale Tastatur (die italienische in diesem Beispiel) muss beim Booten geladen werden, beispielsweise in »/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 '.'
```

/etc/ttys muss ebenfalls geändert werden:

```
#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
```

Die pcvt- Programme müssen ebenfalls kompiliert und installiert werden:

```
cd /sys/arch/i386/isa/pcvt/Util
make
make install
```

nach oben

12.3.1. Die Bildschirmgrösse ändern

Mit »pcvt« kannst Du die Zeilen- und die Spaltenzahl auf dem Monitor ändern. Dieses Script kann benutzt werden, um automatisch zwischen zwei Konfigurationen hin- und herzuschalten:

```
#!/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
```

So, das war's. Viel Spass dabei.

Seitenanfang

Inhaltsverzeichnis Sitemap Home

Kapitel 13.:Texteditor

Inhalt:

13.1. Einführung in vi

Diese Sektion ist ein Beitrag von Jason R. Fink

Es ist nicht so, dass gelegentliche UNIX- User eine Einführung in die Benutzung des vi- Editors brauchen. Dieser Editor, der im Original ein Entwicklung von Bill Joy von Sun Microsystems ist, ist ein endlos erweiterbarer, eigentlich ziemlich leicht zu bedienender ASCII- Editor und lebenswichtig für Newbies. Diese Sektion wird den Newbie in die Bedienung einführen und ein vielleicht paar Einfälle des Gelegenheitsusers verwerfen.

In der ersten Hälfte dieser Sektion geht es um das Schreiben, Speichern, Entfernen/Einfügen und Navigieren durch eine Datei in einer vi- Sitzung. In der zweiten Hälfte findet eine Schritt-für-Schritt- vi- Sitzung statt, um eine Hilfestellung für den Anfang zu bieten.

Die Idee hinter diesem Kapitel ist es, eine Anleitung in der Benutzung des Editors zu bieten; ohne ein Handbuch ersetzen zu wollen. Sie ist gedacht, um dem Anfänger zu unterstützen, damit dieser mit diesem Editor arbeiten kann, also hinreichende Fertigkeiten besitzt, um Dateien zu erstellen und zu bearbeiten.

13.1.1. Das vi- Interface

Die Benutzung des vi- Editors ist nicht sehr viel anders als die anderer terminalbasierter Software; mit einem Ausschluss: Es benutzt kein TabType-Style- Interface (Menüs, wenn man das so will) , obwohl viele vi- Typen das benutzen, um den selben Look-and-Feel eines solchen Programms zu bekommen. Statt dessen arbeitet vi in zwei Modi, »command« und »edit«. Das erscheint einem zunächst etwas befremdlich, aber gibt keinen wirklich grossen Unterschied zwischen vi und dem Windows-basiertem Editieren, wenn Du mal darüber nachdenkst.

Nimm das als ein Beispiel: Wenn Du, sagen wir, gedit benutzt und die Maus nimmst, einen Text markierst, dann Ausschneiden und Einfügen anwählst; benutzst Du die ganze Zeit die Maus, Du bearbeitest keinen Text. In vi wird dieselbe Aktion durchgeführt, indem Du einfach die Zeile mit den »dd«- Kommando löschst, zu der gewünschten Zeile gehst, in der Du den Text einfügen willst und das dann mit »p« im Kommandomodus tust. Du könntest meistens sagen, dass das die Analogie »Mausmodus versus Kommandomodus« ist (natürlich sind sie nicht identisch, aber die Konzepte ähneln sich).

Um eine vi- Sitzung zu starten, ist es am einfachsten, dies hier in irgendeine Terminalsoftware einzugeben:

```
vi Dateiname
```

Wichtig ist es noch, zu wissen, dass eine editierte Datei in einem Speicherpuffer abgelegt wird. Der Rest des Textes liegt im Puffer und die Datei in einem Kontext mit ihm. Eine gespeicherte Datei wird nur geändert, wenn der User die Änderungen mit dem Schreibkommando speichert.

nach oben

13.1.2. In den Editiermodus schalten

vi bietet eine Reihe von Optionen, auf die Du nach dem Start zugreifen kannst. Im Moment ist aber nur der Default- Startup für uns von Interesse. Wenn Du wie oben beschrieben vorgehst, ist der Standardmodus der Kommandomodus; als Ergebnis daraus kannst Du nichts in den Puffer schreiben. Stattdessen musst Du den Kommandomodus verlassen, um Text einzugeben.

Der folgende Text beschreibt die angebotenen Modi nach dem Start:

```
a     Hinter dem Cursor etwas einfügen.
A     Am Zeilenende etwas dranhängen.
C     Den rest der Zeile ändern.
cw    Das aktuelle Wort ändern.
i     Vor dem Cursor etwas einfügen.
I     Vor der letzten Leerzeile etwas einfügen.
o     Eine neue Zeile unter den Cursor öffnen, um etwas einzufügen.
O     Eine neue Zeile über dem Cursor öffnen, um etwas einzufügen.
```

nach oben

13.1.3. Zwischen den Modi umschalten und den Puffer speichern

Natürlich reicht es nicht, die Editierkommandos zu kennen, wenn Du nicht in den Kommandomodus zurückkommst, um die Datei zu sichern: Drücke einfach die [ESC]- Taste. Um mehrere Kommandos einzugeben, muss der Doppelpunkt benutzt werden. Schreibkommandos sind so ein Satz von Kommandos. Um das zu tun, gib einfach mal »:« ein.

Der Doppelpunkt bringt den User in den sog. »colon«- Prompt (oder Kommandomodus, wenn man das so will) in der unteren linken Ecke des Bildschirms. Schauen wir uns jetzt einmal die Speicherkommandos an:

```
:w    Schreibt den Puffer in eine Datei.
:wq   wie oben, beendet ausserdem den Editor.
```

nach oben

13.1.4. Entfernen und einfügen

Wie gut ist ein Editor, wenn Du keine Textblöcke manipulieren kannst? Natürlich unterstützt vi dieses Feature und wie die meisten dieser Kommandos sind sie auf gewisse Weise auch intuitiv. Um eine Zeile zu markieren, ohne sie zu löschen, gib einfach »yy« ein und schon wird die markierte Zeile in einen Puffer kopiert werden. Um die Zeile irgendwo abzulegen, gehe zu der Zeile, die sich über der Stelle befindet, wo besagter Text eingefügt werden soll und drücke »p«, das »put«- Kommando. Um eine Teile zu verlegen, lösche also einfach die ganze Zeile mit dem »dd«- Kommando, navigiere und putte.

nach oben

13.1.4.1. uuuups, das wollte ich so aber nicht...

Auch Undo geht: »u« setzt die letzte Aktion zurück und »U« ist das Undo für die letzte geänderte oder gelöschte Zeile.

nach oben

13.1.5. Navigation im Puffer

Viele Einführungen oder Tutorials zum Thema vi fangen erst einmal mit der Navigation an. Wie dem auch sei; in denmeisten Editoren dreht sich dieses Thema meisten darum, wieman sich innerhalb einer Datei bewegt oder wie zwischen mehreren Dateien hin- und hergesprungen werden kann. Abhängig von Deiner vi- Sorte (Es gibt da ja nicht nur vi, sondern auch noch elvis, nvi oder vim) kannst Du in beiden Betriebsmodi navigieren (Editier- und Kommandomodus).

Für einen Anfänger ist das Umschalten zwichen Lommando- und Editiermodus ein wenig sicherer, denke ich; bis er ein bisschen geübt hat und sich Peaxis und Erfahrung einschleichen. Bei den Navigationstasten für Terminals, die entweder nicht erkannt werden oder keine Cursortasten besitzen oder bei denen sie nicht unterstützt werden, funktioniert das mit diesen Tasten:

```
k     Cursor eine Zeile höher
j     Cursor eine Zeile tiefer
l     Ein Zeichen weiter nach rechts
h     Ein Zeichen weiter nach links
```

Wenn das Terminal erkannt wird und die Cursortasten unterstützt, können sie im Kommandomodus zur Navigation durch den Puffer benutzt werden.

Für eine einfache »One-Spot-Navigation« unterstützt vi auch den Sprung zu einer Zeile, deren Nummer im Kommandoprompt (:) eingegeben wird. Wenn Du zu Zeil 223 springen willst, musst Du das hier eingeben:

```
ESC
:223
```

nach oben

13.1.6. Dateisuche; Alternative Navigationshilfe

Vi unterstützt die Suche mittels einer geregelten, erweiterten Syntax. Allerdings ist sie anders als das Erflehen einer Datei im Kommandomodus. Einfach die »/«-Taste drücken und eingeben, was gesucht wird; beispielsweise eine Datei mit dem Namen »foo«:

```
/foo
```

Das war's. Um eine andere Suche zu illustrieren, suchen wir mal nach »foo bar«:

```
/foo bar
```

nach oben

13.1.6.1. Weitere Navigationskommandos

Suchen un Scrollen sind nicht die einzigen Wege, auf denen man sich durch einen vi- Puffer bewegen kann. Es folgt kurze eine Liste mit den Navigationskommandos von vi:

```
0     Gehe zu Zeilenanfang
$     Gehe zum Zeilenende
b     Ein Wort zurück
w     Ein Wort weiter
G     Zum Ende des Textes
H     Zu ersten Zeile auf dem Bildschirm
L     Zur letzten Zeile auf dem Bildschirm
M     Zur Bildschirmmitte
N     Gehe rüchwärts zum nächsten Ergebnis einer Suche.
n     Zum nächsten Ergebnis eine Suche, aber vorwärts
```

nach oben

13.1.7. Ein Beispiel für eine Sitzung

So, jetzt haben wir die grundlegenden Dinge besprochen. Lass uns mal eine Sitzung abhalten, in der viele der bis hierher diskutierten Punkte vorkommen. Als erstes öffnen wir einen leeren Dateipuffer in der Kommandozeile:

```
# vi foo.txt
```

Wir schalten in den Editiermodus und geben zwei Zeilen ein, die von einer leeren Zeile getrennt werden. Denke daran: Der Puffer ist leer; wir müssen zuerst »i« drücken, um vor dem Cursor etwas einzufügen und tippseln mal etwas Text:

```
Dies ist ein Text

und über uns ist eine leere Zeile.
~
~
~
~
```

Drücke jetzt die [ESC]- Taste, um in den Kommandomudus zurückzuschalten.

Jetzt sind wir im Kommandomodus; lass uns die Datei mal speichern. Zuerst drückst Du die »:«-RTaste. Der Cursor sollte sich nun unten links rechts neben dem Prompt befinden. Dort gibs Du »w« ein und drückst »ENTER« oder »RETURN«. Die Datei wurde eben gerade gespeichert. Jetzt sollte eine Nachricht darüber kommen; einige Varianten sagen uns den Namen, wieviele Zeilen und die Dateigrösse.

Es ist Zeit, geneigter Leser; Zeit zum Navigieren. Der Cursor sollte sitzen, wo er schone gesessen hat, als die Datei gespeicert wirde. Versuche mal, mit den Cursortasten ein wenig in der Datei herumzuspazieren. Wenn das nicht funktioniert, versuche es mit den besagten »hjkl«-Taste, um zu sehen, wie sie arbeiten.

Lass uns mal zum guten Ende zwei weitere Dinge tun. Als Erstes gehen wir in die erste Zeile und dann auf das erste Zeichen. Probiere hier mal ein paar der anderen Navigationstasten aus dem Kommandomodus, indem Du folgende Tasten nacheinander drückst:

```
$
0
$
0
```

Der Cursor sollte zum Zeilenende springen, dann zurück zum Anfang und dann wieder zu Ende.

Als nächstes suchen wir nach einem Ausdruck, indem wir die »/«- Taste benutzen:

```
/Dies
```

Der Cursor sollte nun zu der ersten Stelle springen, an der diese Zeichenfolge auftaucht.

Speichere sie Datei und beende vi:

```
:wq
```

nach oben

13.2. Die Konfiguration von vi

Der Standardeditor, der mit NetBSD kommt, ist vi; eigentlich muss man das nicht mehr erwähnen. Vi ist der meistgeliebte und -gehasste Editor der Welt. Wenn Du vi nicht benutzst, überspringe diese Sektion; ansonsten lies weiter, bevor Du andere Versionen von vi installierst. Der NetBSD- vi wurde von Ken Bostic (University of California in Berkeley; kurz UCB) geschrieben, um eine freie Version dieses Editors zu bekommen. Er hat viele nützliche Erweiterungen, wobei er trotzdem sehr kompatibel mit dem Original ist. Nvi ist die vi-Standardversion für BSD.

Die interessantesten Erweitrungen sind diese hier:

- Erweiterte reguläre Ausdrücke
- Tag- Stapel
- Unendliches Undo (um eine Aktion zurückzusetzen, drücke »u«; um damit weiterzumachen, drücke ».«)
- Inkrementelle Suche, die mit der Option »searchincr« eingeschaltet wird
- Links-rechts- Scrollen von Zeilen mit der Option »leftright«; Die Anzahl der Spalten, durch die gescrollt werden kann, wird mit der »sidescroll«- option definiert.
- Kommandozeilen- History, wird mit »cedit« eingeschaltet
- Dateinamenvervollständigung, einschalten mit »filec«

- Hintergrundbildschirme und -Anzeigen
- Teilung des Bildschirms

nach oben

13.2.1. Erweiterungen in .exrc

Das folgende Beispiel zeigt eine ».exrc«- Datei mit ein paar eingeschalteten Erweiterungen.

```
set showmode ruler
set filec=^[
set cedit=^[
```

Die erste Zeile schaltet die Anzeige der Cursorposition (in Reihe und Spalte) ein und informiert über den aktuellen Betriebsmodus (Kommando, Einfügen, Anhaängen). Die zweite Zeile (in der »^[« für die [ESC]- Taste steht, schaltet die Dateinamenvervollständigung mit der ESC- Taste ein.. Kommando Nummer drei gestattet das »History-Editing« mit der ESC- Taste. Beispiel: wenn Du einen »:« schreibst und dann ESC drückst, kommt ein Fenster, das eine Liste mit den vorgehenden Komandos öffnet, die dann bearbeitet und ausgeführt werden können (Mit »Enter« kannst Du so ein Kommando ausführen).

nach oben

13.2.2. Dokumentation

Der Tarball mit den (src.tgz) beinhaltet eine Menge nützlicher Dokumentationen zu (n)vi und ex; das liegt im »/usr/src/usr.bin/vi/docs«-Verzeichnis. Beispiele:

- Edit: A tutorial
- Ex Reference Manual
- Vi man page
- An Introduction to Display Editing with Vi by William Joy and Mark Horton
- Ex/Vi Reference Manual by Keith Bostic
- Vi Command & Function Reference
- Vi tutorial (beginnner and advanced)

Wenn Du vi noch nie benutzt hast, ist das vi- Tutorial ein guter Ausgangspunkt. Es ist gemacht, um mit vi gelesen zu werden und führt den Leser schrittweise in alle vi- Kommandos ein, die während des Lesens ausprobiert werden können. *An Introduction to Display Editing with Vi* vonWilliam Joy ud Mark Horton ist ebenfalls ein sehr guter Ausgangspunkt.

Wenn Du mehr über vi und seine Erweiterungen lesen willst, solltest Du Dir mal *Ex/Vi Reference Manual* von Keith Bostic durchlesen, worin alle Editorkommandos und -Optionen dokumentiert sind.

nach oben

13.3. Tags mit vi

Das ist nicht direkt NetBSD- spezifisch, aber es kann von Nutzen sein um beispielsweise die Kernelquellen zu überprüfen.

Wenn Du einen ganzen Satz von Quellen in einem Baum aus Verzeichnissen und Unterverzeichnissen überprüfen willst, kannst Du Dir die Arbeit mit dem »tag«- Feature erleichtern. Die Methode sieht so aus:

1. **cd** in das Basisverzeichnis, in dem Du arbeiten willst:

   ```
   $ cd /path
   ```

2. **Gib mal diese Kommandos ein:**

```
$ find . -name "*.[ch]" > filelist
$ ctags -L filelist
```

1. Diese Zeile fügst Du .exrc hinzu

   ```
   set tags=/path/tags
   ```

   (Gib aber den richtigen Pfad für »path« ein...)

nach oben

Inhalt

Kapitel 14. X

Inhalt:

14.1. Was ist X?

X- Windows ist eine grafische Betriebssystemumgebung, die es für NetBSD und viele Unices (und Nicht- Unices) gibt. In der Realität ist es viel mehr als das: Dank der Nutzung des X-Protokolls ist das X-Window-System »netzwerktransparent«: Es können verteilte Anwendungen darauf laufen (Client-Server). Das heisst, dass Du eine Applikation auf dem einen Host laufen lassen kannst(Client) und die grafische Ausgabe auf einem anderen Host (dem Server). Transparent bedeutet, dass Du die Applikation als solches nicht modifizieren musst, um diesen Effekt zu erreichen. Das X-Window-System wird vom X-Konsortium hergestellt und gewartet. Die aktuelle Ausgabe ist X11R6. Das Aroma von NetBSD heisst Xfree86 und ist eine frei weiterverteilbare Open-Source-Implementation des X-Window-Systems.

Wenn Du anfängst, X zu benutzen, wirst Du viele neue Dinge finden, die Dir anfangs vielleicht etwas komisch erscheinen könnten. Die Grundelemente von X sind:

- Grafikhardware, die von XFree86 unterstützt wird.
- Ein X-Server, der an der Spitze der Hardware läuft. Der X-Server bietet einen standardtisierten Weg, wie Fensterchen geöffnet werden, macht die Grafik (einschliesslich der Schiften für den Text) und verarbeitet den Input von Maus/Tastatur/anderem. X ist netzwertransparent. Damit kannst Du X-Clients auf einer Maschine laufen lassen und den Server(Beispielsweise das Display mit der Grafikhardware) auf einer anderen Maschine.
- Ein Windowmanager läuft auf dem X-Server. Der Windowmanager ist genaugenommen ein spezieller Client, der die Platzierung der einzelnen kontrollieren darf. Er dekoriert die Fenster auch mit den Standard- Widgets (Gewöhnlich bieten diese die Bewegung der Fenster, Änderung der Grösse, Ablegen als Symbol auf der Arbeitsfläche und einige andere Aktionen an). Ein Windowmanager kann ausserdem inaktiver Fenster nach hinten verschieben, etc. Sie erlauben es Dir ausserdem, Fenster oder Programme zu killen indem man auf diese Fenster klickt undsoweiter.
- Ein optionaler Desktopmanager: Kde und Gnome als solche Beispiele sind Desktops: Sie sind komplette Suiten mit mehr oder weniger integrierter Software, um Dir einen wohldefinierten Bereich von Programmen und mit einem mehr oder weniger verbreiteten Interface für jedes dieser Programme. Dazu gehört ein Help- Browser in irgendeiner Art; ein Dateimanager, angepasste Teminals, um xterm zu ersetzen, Entwicklungumgebungen für Software, Audio, Bild/-Animationsbetrachter etcetc.

- Irgendwelche anderen Applikationen (X- Clients von Dritten), die Du hast. Diese kommunizieren mit dem X-Server und dem Windowmanager. Solange der Windowmanager Bestandteil des Desktops ist, wird der Desktop nicht darüber informiert, was diese Programme tun (Allerdings: GNOME beispielsweise kann in der Lage sein, herauszufinden, ob Du GIMP installiert hast und bietet dann ein Menü an, mit dem Du auf Gimp zugreifen kannst).

Zusammenfassung: Wenn Du eine grafische Umgebung benutzen willst, bbrauchst Du:

- Das XFree86-System
- Einen Windowmanager (XFree86 kommt immer mit einem sehr rudimentären Windowmanager, TWM heisst der).
- Wenn Du es gerne komfortabler hättest, kann es sein, dass Du auch einen Desktop installieren willst; es ist aber nicht notwendig. Die Desktops besitzen ein paar nette Features, die für User, die aus der MAC-OS- oder Windows-Ecke kommen, neu sind (Der KDE- Desktop ist MS-WINDOWS in der Anmutung sehr ähnlich).

  Achtung: Es sollte Dir klar sein, dass Desktops wie Gnome und KDE keine X-Server anbieten. Sie laufen auf der obersten Schicht eines vorhandenen X-Servers, der von XFree86 kommt. KDE und GNOME können so gemacht werden, dass sie entweder ihren eigenen oder einen anderen, separat instalierten Windowmanager benutzen.

Im Normalfall kannst Du meistens nur einen Windowmanager zu einer definierten Zeit auf einem definierten X-Server laufen lassen (Aber Du kannst mehrere X-Server auf einem einzelnen Rechner laufen lassen). Wenn bei Dir kein Windowmanager Deiner Wahl läuft und Du KDE oder GNOME startest, wird die jeweilige Schreibtischumgebung einen geeigneten Windowmanager für Dich starten.

nach oben

14.2. Konfiguration:

Wenn Du Dich bei der Installation nicht für die Minimalkonfiguration entschieden hast, ist X bereits auf Deinem Rechner installiert. Du musst nur die Datei »/etc/XF86Config erstellen. Um eine Ahnung davon zu bekommen, wie so eine Datei aussieht, schau Dir mal »/usr/X11R6/lib/X11/XF86Config.eg« an. Die Struktur der Konfigurationsdatei ist in XF86Config (4/5) beschrieben, die mit diesem Kommando gelesen werden kann:

```
# man XF86Config
```

Bevor Du mit der Konfiguration des Systems beginnst, ist es zu empfehlen, die Dokumentation in »/usr/X11R6/lib/X11/doc« gründlich durchzulesen: dort finden sich verschiedene READMEs für die Grafikkarten, die Maus und besonders ein NetBSD- spezifisches README (README.NetBSD). Meine Empfehlung ist es, mit »QuickStart.doc« anzufangen. Es könnte Dir so vorkommen, dass das mit anderen Systemen schneller geht, aber die Zeit, die Du aufbringst, dieses Dokument zu lesen, ist nicht vergeudet: X und seine Konfiguration zu kennen, ist immer nützlich, besonders für spätere Gelegenheiten. Ausserdem wirst Du dann den grössten Teil Deiner Hardware kennen (Die Software natütlich auch).

Du kannst die »/etc/XF86Config« manuell mit einem Editor erstellen oder sie automatisch mit Hilfe eines interaktiven Konfigurationsprogramms generieren. Die besten bekannten Programme sind »xf86config« und XF86Setup«. Das Erste ist ein Programm im Texmodus und

kommt immer mit, wenn X installiert wird; das Letztere ist ein grafisches Programm und kann aus der Packages-Collection installiert werden.

Es kann ja auch sein, dass Du eine Mischung aus beiden besser findest: Zuerst erstellst Du »XF86Config« mit einem der Programme und dann nimmst Du die Feineinstellungen mit einem Editor vor.

Das Interface der beiden Programm ist unterschiedlich, aber sie brauchen beide dieselben Informationen:

- Den Maustyp
- Tastaturtyp und -Layout
- Typ der Grafikkarte
- Den Monitortyp

Fazit: Bevor Du anfängst, das System zu konfigurieren, solltest Du diese Informationen griffbereit haben.

nach oben

14.3. Die Maus

Das Erste, was Du herausfinden solltest, ist der Maustyp, den Du verwendest (Beispiel: seriell oder PS/2) und das Maus-Device (Beispiel: »wsmouse« setzt ein anderes Protokoll voraus). Wenn Du eine serielle Maus benutzst, erwähle das korrekte Protokoll und lege den Anschluss fest, an dem sie angeschlossen ist. Hier ein Beispiel für ein Tier, das am ersten seriellen Port knabbert:

```
Section "Pointer"
  Protocol    "Microsoft"
  Device      "/dev/tty00"
EndSection
```

Für eine Maus, die das wsmouse- Gerät verwendet, musst Du das hier eintragen:

```
Section "Pointer"
  Protocol    "wsmouse"
  Device      "/dev/wsmouse0"
EndSection
```

Im "Device-Field" kannst Du auch »/dev/mouse« festlegen, wenn Du einen korrekten Link im Dateisystem hast. Beispiel:

```
# ln -sf /dev/wsmouse0 /dev/mouse
```

nach oben

14.4. Die Tastatur

Besonders, wenn Du Deine Tastatur für wscons konfiguriert hast, musst Du sie auch für X konfigurieren, wenn Du ein non-US-Layout willst.

Eine einfache Lösung ist die Nutzung des XKB- Protokolls, mit dem der Tastaturtyp und das -Layout beschrieben werden.

Dies ist ein Gebiet, auf dem Konfigurationsprogramme schwach sind: Du solltest das Standard-Layout wählen und die erzeugte Konfigurationsdatei manuell korrigieren.

```
# XkbDisable
# XkbKeymap   "xfree86(us)"

XkbModel        "pc102"
XkbLayout       "de"
XkbVariant      "nodeadkeys"
```

Wenn du »Windows-Tasten« auf Deiner Tastatur benutzen willst, benutze »pc105« statt »pc102« für das KB- Modell.

nach oben

14.5. Der Monitor

Es ist sehr wichtig, die vertikale und horizontale Frequenz des Monitors korrekt anzugeben: Eine korrekte Konfiguration schützt Deinen Monitor vor Defekten, die eine inkompatible Einstellung Deiner Grafikkarte verursachen kann. Diese Information findest Du im Handbuch Deines Monitors. Im X-Dokumentationsverzeichnis gibt es eine Datei, in der die richtigen Einstellungen für viele Monitore zu finden sind. Diese kannst Du nutzen, um Deine eigenen Einstellungen herauszufinden.

nach oben

14.6. Grafikkarten und X-Server

Die Grafikkarte kann aus der Datenbank der automatischen Konfigurationsprogramme gewählt werden. Das Programm erledigt dann die nötigen Einstellungen für Dich.

Wenn Du die richtige Grafikkarte ausgesucht hast, musst Du Dich für den passenden X-Server entscheiden. Im Normalfall finden die Programme automatisch den richtigen Server, aber einige Karten können von mehreren Servern gesteuert werden (Beispiel: Die S3Virge geht sowohl mit dem SVGA- als auch mit dem S3V- Server). In diesem Fall lies bitte die Dokumentation der Server, um zu entscheiden, welchen Du brauchst: Unterschiedliche Server besitzen unterschiedliche Fähigkeiten und einen unterschiedlichen Grad der Unterszützung für die Grafikkarten.

nach oben

14.7. X starten

Wenn Du das Konfigurationsprogramm beendest, erstellt es die Datei »/etc/XF86Config, die hiterher überprüft und manuell geändert werden kann.

Bevor Du X startest, solltest Du:

- sicherstellen, dass der Link »/usr/X11R6/bin/X« auf den richtigen Server zeigt:

  ```
  # ls -l /usr/X11R6/bin/X
  ```

- Überprüfe, ob die Konfiguration korrekt ist:

  ```
  # X -probeonly
  ```

  ... und prüfe die Ausgabe gründlich.

Jetzt kannst Du X mit diesem Kommando starten:

```
# startx
```

Wenn X nicht hochfährt, kann das ein Fehler in der Konfigurationsdatei sein.

Wenn X startet, aber nicht so funktioniert wie gedacht (Beispiel: Du kannst den Mauszeiger nicht bewegen), kannst Du X mit [ctrl-Alt-Backspace] beenden (Achtung: Gibt es nicht auf allen Protierungen!). Wenn alles richtig funktioniert, befindest Du Dich in der X- Umgebung mit dem Default- Windowmanager (twm). Das ist ein simpler Windowmanager, von dem viele 'Leute glauben, dass er für ihre Zwecke hinreicht. Wenn Du einen stärker konfigurierbaren Windowmanager mit mehr Features willst, hast Du in der Packages-Collection bereits eine grosse Auswahl.

Um X anzupassen, versuche mal folgendes Kommando in einem xterm einzugeben, um die Hintergrundfarbe zu wechseln:

```
# xsetroot -solid DarkSeaGreen
```

nach oben

14.8. X anpassen:

Das Aussehen von X kann auf verschiedenen Wegen angepasst werden. Der einfachste ist es, die Standard ».xinitrc« in Dein Home- Verzeichnis zu kopieren und sie zu modifizieren. Als Beispiel:

```
# cp /usr/X11R6/lib/X11/xinit/xinitrc ~/.xinitrc
# vi .xinitrc
```

Das folgende Beispiel zeigt, wie der Windomanager (twm) zu starten ist, der eine Instanz des »xclock«- Programms im unteren rechten Teil des Bildschirms und zwei »xterm«- Fenster starten soll. Die Farbe »Bisque4« wird im Hintergrund verwendet.

```
Der erste Teil der Datei ist derselbe...
...
# 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
```

Mit dieser Art der Konfiguration musst Du das letzte »xterm«(das mit dem »login«-Titel) schliessen, um X zu verlassen.

Gerade mit dieser simplen Konfiguration sieht X schon viel netter aus. Um die Umgebung noch weiter aufzuhübschen, kannst Du ein paar Utilities aus der Package-Kollektion installieren. Beispiele:

xcolorsel

> zeigt alle Farben, die in »rgb.txt« definiert sind. Benutze es, um die Hintergrundfarbe des Rootfensters oder der xterms zu ändern.

xpmroot

> lässt Dich ein pixmap für den Hintergrund benutzen.

xscreensaver

> X- Dildschirmschoner.

xdaemon

> Ohne dieses Paket kann kein Desktop komplett sein, der ein bewegliches Bitmap des BSD- Teufelchens in zwei wählbaren Grössen darstellt.

nach oben

14.9. Andere Windowmanager

Wenn Du den sehr einfachen twm nicht magst, der nicht sehr konfigurierbarer Windowmanager ohne allzuviele Features ist,kannst Du einen anderen Windomanager aus der Paketkollektion verwenden. Die populärsten sind wohl: fvwm2, olwm/olvwm (Open Look Windowmanager), WindowMaker, Enlightenment, Afterstep.

Im Rest dieser Sektion wird die Installation von WindowMaker als Beispiel beschrieben. WindowMaker ist ein sehr nett aussehender und stark konfigurierbarer Windowmanager. Um das Programm zu installieren, wird das vorkompilierte das »windowmaker-0.60.tgz«- Paket verwendet, das von einigen anderen Pakteten abhängig ist, die ebenfalls installiert werden müssen. Für gewöhnlich installieren »pkg_add« und »make install« diese abhängigen Pakete automatisch mit. Es besteht also keine Notwendigkeit, diese Abhängigkeiten manuell durchzugehen.

```
# 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
```

Hinweis: Du kannst die Abhängigkeiten auch mit diesem Kommando sehen:

```
# pkg_info -f windowmaker-0.61.0.tgz | grep depends
```

Nachdem die geforderten Pakete hinzugefügt wurden, können der WindowMaker und ein paar vorkonfigurierte Themes dazugefügt werden:

```
# pkg_add windowmaker-0.61.0.tgz wmthemes-0.6x.tgz
```

WindowMaker ist jetzt installiert. Um ihn zu starten, musst Du Deine ».xinitrx« und/oder ».xsession« ändern: Ersetze die Zeile, die twm aufruft durch eine mit dem Namen »wmaker«. Beispiel:

```
# start some nice programs
# start WindowMaker
wmaker &
xclock -geometry 50x50-1-1 &
xdaemon2 -geometry +0-70 &
...
```

In diesem Beispiel hüpft das BSD- Teufelchen automatisch über den Bildschirm...

Bevor WindowMaker gestartet werden kann, muss das Konfiguratinsprogramm laufen:

```
$ wmaker.inst
$ startx
```

nach oben

14.10. Grafischer Login mit XDM

Wenn Du X immer für Deine Arbeit verwendest und das Erste, was du tust, wenn Du Dich eingeloggt hast »startx« ist, dann solltest Du Dich gleich grafisch auf Deiner Maschine einlogggen. Das geht sehr leicht:

1. Erstelle die ».xsession«-Datei in Deinem Home-Verzeichnis. Die Datei ähnelt der »$home/.xinitrc« und kann auch ein Link zu letztere sein.

2. Modifiziere »/etc/rc.conf«:

   ```
   xdm=YES       xdm_flags=""       # x11 display manager
   ```

   Wenn Du es vorziehst, kannst Du auch die folgende Zeile am Ende Deiner »/etc/rc.local« anhängen, anstatt »rc.conf« zu modifizieren (Warum eigentlich?):

   ```
   /usr/X11R6/bin/xdm
   ```

Mit dieser Methode kann man beispielsweise auch kdm oder gdm anstelle von xdm starten.

Die Konfigurationsdateien für xdm liegen im »/usr/X11R6/lib/xdm«- Verzeichnis. In der »Xservers«- Datei wird X standardmässig auf dem virtuellen Terminal »vt05« gestartet. Wenn Du stattdessen ein anderes Terminal verwenden willst, ist das der richtige Platz, das zu ändern. Um Tastaturkonflikte zwischen getty und xdm zu verhindern, ist anzuraten, xdm auf einem virtuellen Terminal zu starten, auf dem getty abgeschaltet ist. Zum Beispiel hat Du in »Xservers«:

```
:0 local /usr/X11R6/bin/X :0 vt04
```

In »/etc/ttys« sollte das hier stehen:

```
ttyE3   "/usr/libexec/getty Pc"         vt220   off secure
```

(Denke bitte daran, dass vt04 mit ttyE3 korrespondiert, weil vt mit 1 startet und ttyE mit 0)

Wenn Du willst, dass der xdm- Loginbildschirm nett aussieht, kannst Du die xdm- Konfigurationsdatei ändern. Beispiel: Du kannst die Hintergrundfarbe ändern, indem Du folgende Zeile der »Xsetup_0«- Datei hinzufügst:

```
xsetroot -solid SeaGreen
```

Statt eine Farbe zu setzen, kannst Du mit dem »xpmroot«- Programm auch ein Bild auf den Hintergrund legen. Beispiel:

```
xpmroot /pfad_nach_xpm/netbsd.xpm
```

Wenn Du ein wenig mit der Konfigurationsdatei experimentiert hast, kannst Du viele nett aussehende Effekte erzeugen und einen witzigen Login- Bildshirm erstellen.

nach oben

Inhalt

Kapitel 15: Linux- Emulation

Inhalt:

Der NetBSD- Port für i386- Computer kann eine grosse Anzahl von nativen Linux- Programmen ausführen, indem er einen Layer für die Linux- Emulation benutzt. Grundsätzlich gilt, wenn Du über eine Emulation nachdenkst, dass Du Dir vorstellen wirst, dass solche Dinge langsam und ineffizient sind. Eulatoren müssen Hardwarebefehle und besonders Architekturen softwaremässig nachbilden (besonders die von alten Maschinen). Das gilt jedoch nicht für die Linux- Emulation: Das ist nur ein kleiner Softwarelayer, meistens für Systemaufrufe, die in beiden Systemen sehr ähnlich sind. Der Code der Applikation selbst wird mit der maximalen Leistung Deiner CPU ausgeführt. Die Performance des Systems wird sich mit der Linux-Emulation also nicht verringern und das Gefühl bei der Benutzung ist das Gleiche wie bei der Anwendung von nativen NetBSD- Anwendungen.

Dieses Kapitel beschreibt anhand eines Beispiels, wie die Linux- Emulation zu konfigurieren ist: Die Installation des allseits bekannten Adobe Acrobat Reader 4 soll durchgeführt werden.

15.1. Aufsetzen der Emulation

Die Installation ist in der »compat_linux(8)«- Manpage beschrieben. Die Benutzung des Package- Systems braucht nur zwei Schritte.

1. Konfiguration des Kernels
2. Installation der Linux- Bibliotheken

Seitenanfang

15.1.1. Konfiguration des Kernels

Wenn Du einen GENERIC- Kernel benutzt, brauchst Du an dieser Stelle nichts zu tun.

Wenn De einen eigenen Kernel benutzt, schau nach, ob die folgenden Optionen aktiviert sind:

```
option COMPAT_LINUX
option EXEC_ELF32
```

Wenn Du einen Kernel mit diesen Optionen kompiliert hast, kannst Du damit anfangen, die nötige Software zu installieren.

Seitenanfang

15.1.2. Installation der Linux- Bibliotheken

Du kanst die Linux- Bibliotheken von einer beliebigen Linux- Distrbution verwenden, vorausgesetzt, dass sie nicht zu alt sind. Aber die empfohlene Methode ist, das Package-

System zu verwenden und die Bibliotheken automatisch zu installeieren (Suse- Bibliotheken werden mitgeliefert). Wenn Du die Bibliotheken installierst, passiert folgendes:

- Ein zweites Root- Verzeichnis wird angelegt, das für die Linux- Programme benutzt wird. Das Verzeichnis ist »/emul/linux«. Die Linux- Programme im Emulationmodus nutzen diese Verzeichnis als ihr Root- Verzeichnis.

- Die Shared Libraries für Linux werden installiert. Die meisten Applikationen sind dynamisch gelinkt und gehen davon aus, dass sich die nötigen Bibliotheken auf dem System befinden. Beispielsweise der Acrobat Reader. Wenn Du in das »/usr/pkgsrc/print/acroread«- Verzeichnis gehst und das **make depends**- Kommando ausführst, bekommst Du folgenden Systemmeldung:

  ```
  ===>  acroread-4.0 requires Linux glibc2 libraries - see compat_lin
  ```

Beide Aktionen werden automatisch vom Package- System gehandhabt, ohne dass der User manuell intervenieren muss (Ich unterstütze das; für jetzt; ich habe gerade begonnen, das Package- system zu lieben...)

Um die Bibliotheken zu installieren, muss ein Programm installiert sein, dass das RPM- Format handhaben kann: es ist rpr-2.5.4, das von den Suse- Bibliotheken verwendet wird, um selbige auszupacken.

Als nächstes muss das »suse_base«- Paket installiert werden. Die Suse- RPM- Files können mit dem Package- System heruntergeladen werden oder, falls Du eine Suse- CD hast, kannst Du die Sachen auch nach »/usr/pkgsrc/distfiles/suse« kopieren und sie mit **make** und **make install** installieren.

Nach derselben Methode werden auch »suse_compat«, »suse_libc5« und »suse_x11« installiert. Die Konfiguration sieht dann so aus:

```
# 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 librari
suse_libc5-6.1p1    Linux compatibility package for libc5 binaries
```

Seitenanfang

15.1.3. Installation des Acrobat Readers

Jetzt ist alles fertig, um den Acrobat Reader oder irgendein anderes Programm zu installieren. Wechsle nach »usr/package/print/acroread« und gib die folgenden Kommandos ein:

```
make
make install
```

Die Installationsscripts des Acrobat Reader fragen Dich, ob Du die Lizenzbedingungen akzeptierst. Wenn Du das getan hat, kannst Du das Programm ausführen.

Seitenanfang

15.1.4: Verzeichnisstruktur

Wenn wir das Ergebnis der Installation der Linux- Bibliotheken überprüfen, finden wir heraus, dass »/emul/linux« ein symbolischer Link ist, der auf«usr/pkg/emul/linux« zeigt, wo diese Verzeichnisse erstellt wurden:

```
bin/
boot/
cdrom/
dev/
etc/
floppy/
home/
lib/
mnt/
opt/
proc/
root/
sbin/
usr/
```

> **Hinweis:** Bitte immer unter »/emul/linux« und nicht unter »usr/pkg/emul/linux«, wenn Du in diesen Verzeichnissen arbeiten musst. Der Grund ist ein Detail in der Implementation und kann sich künftig ändern.

Wieviel Platz braucht man eigentlich für die Linux- Emulation? Auf meinem System bekam ich diese Antwort:

```
# cd /usr/pkg/emul
# du -k linux
...
60525   linux/
```

Der Acrobat Reader, um den es hier ging, wurde im üblichen Verzeichnis für Binärpakete installiert: »/usr/pkg/bin«.

Seitenanfang

Inhaltsverzeichnis

Kapitel 16: Musike mitten Compi

Inhalt:

Original von Manolo De Santis

Dieses Kapitel soll eine kleine Einführung in die Benutzung von Soundgeräten mit NetBSD sein (Wer will schon einen stummen Computer?)

16.1. Hardwarevoraussetzungen

Wenn Du Deine Maschine zum Reden, Musikmachen oder anderem überreden willst, das Krach macht, musst Du wissen, welche Soundkarte in Deiner Maschine eingebaut ist. Schade ist nur, dass es nicht ausreicht, den Hersteller und das Modell der Karte zu kennen, weil viele Hersteller ihre Chipsätze von Dritthersteller beziehen. Daher ist es manchmal sinnvoll, zu wissen, welcher Chipsatz auf Deiner Soundkarte seinen Dienst versieht. Der NetBSD- Kernel erkennt die meisten Chipsätze automatisch; ein schneller Blick auf den **dmesg** -Output sollte in den meisten Fällen reichen. Dazu gib bitte folgendes Kommando ein:

```
# dmesg | more
```

und suche nach der Soundkarte und ihrem Chipsatz. Wenn Du Glück hast, brauchst Du nichts weiter zu tun, weil der Kernel Deine Karte erkannt hat.

Manchmal geht Audio aber nicht, weil die Karte nicht unterstützt wird oder Du ein bisschen arbeiten musst, damit die vom Kernel bereits erkannte Karte funktioniert. Viele Soundkarten sind heutzutage sehr billig, so dass es sich lohnen kann, eine andere Karte zu beschaffen. Aber bevor Du das tust, kannst Du mit ein paar kleinen Schritten Deine Karte vielleicht zum Laufen bringen.

nach oben

16.2. BIOS- Einstellungen

Diese Sektion ist nur für i386- PC- User interessant; auf anderen Architekturen (Amiga...) gibt es diese Features nicht. Das wichtigste, was zuerst festgestellt werden muss, ist, für welchen Bus die Karte gebaut worden ist. Am weitesten verbreitet sin ISA und PCI.

ISA- Karten zu konfigurieren ist wegen der Interaktion mit dem BIOS des Computers deutlich schwieriger

Auf den neueren Maschinen (ab Baujahr 1997) gibt es eine BIOS- Option, die bei der Konfiguration von ISA- Karten einige Kopfschmerzen bereiten kann. Diese Option heisst meistens `PNP OS Installed` und findet sich gewöhnlich in der `PNP/PCI Configuration` (Das kann in Deinem BIOS auch anders heissen.). Generell gilt: Wenn Du NetBSD benutzt, ist es besser, diese Option ausgeschaltet zu lassen oder auszuschalten (Also auf `no` zu setzen).

Anmerkung: Auf vielen System läuft trotzdem alles gut, wenn das eingeschaltet ist. Das hängt eben vom System ab.

nach oben

16.3. Die Soundkarte konfigurieren

Während der Installation von NetBSD werden die erstellten Geräte im `/dev`- Verzeichnis angelegt. Uns interessiert primär:

```
/dev/audio
/dev/sound
/dev/mixer
```

Wenn sie nicht da sind, kannst Du sie mit diesem Kommando herstellen:

```
# cd /dev
# ./MAKEDEV all
```

Dieses Kommando erstellt alle Geräte, inklusive der Audiogeräte.

Die Soundkarte könnte jetzt ohne weiteres Zutun gebrauchsfertig sein.

Das Beste ist, wenn Du jetzt erst einmal einen schnellen Test machst und eine Audiodatei an die Karte schickst (Im Normallfall ist die Dateinamenendung `.au`); wenn Du keine Audiodatei zur Hand hast, nimmst Du einfach irgendeine Binär- oder Textdatei. Probiere es mit /dev/audio oder /dev/sound:

```
# cat filename > /dev/audio
```

oder

```
#cat filename > /dev/sound
```

Wenn Du etwas hörst, heisst das, dass die Karte von NetBSD unterstützt wird und sie vom Kernel bei Booten erkannt und konfiguriert wurde. Ansonsten musst Du die Kerneleinstellungen für die installierte Karte konfigurieren (mal davon ausgehend, dass die Karte vom Kernel unterstützt wird).

nach oben

16.4. Konfiguration des Kernels

NetBSD unterstützt eine ganze Reihe von Soundkarten. Der GENERIC- Kernel erkennt und konfiguriert die meisten von ihnen.
Viele PCs besitzen keine Soundkarten, sondern auf dem Motherboard integrierte Chipsätze. Diese Kartensorte wird vom GENERIC- Kernel nicht eingeschaltet. Du bist darauf angewiesen, Deinen eigenen Kernel zu backen, wenn Du sie nutzen willst. Schaue Dir die folgenden (oder ähnlichen) Zeilen in der GENERIC- Datei an:

```
# 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
```

Lösche das Doppelkreuz in der `pnpbios`- Zeile und in der Zeile, die das Gerät auf dem Motherboard beschreibt.

Anmerkung: Wenn Du einen eigenen Kernel backen willst, ist es besser, mit einer Kopie des GENERIC- Files zu arbeiten, wie im Kernelback- Kapitel beschrieben.

Manchmal kann es nötig sein, den IRQ und den DMA von Hand zu setzen.

Wenn Du immer noch Probleme hast, kannst Du es damit versuchen, einfach alle Geräte freizuschalten, weil einige Soundkarten so gemacht sind, dass sie nur funktionieren, wenn sie eine andere Karte emulieren.

Viele Chipsätze nutzen die Soundblaster- und OPL- Kompatibilität; aber eine grössere Anzahl arbeitet mit der WSS- Emulation.

OPL ist ein MIDI- Synthesizer, der von Yamaha hergestellt wird; es gibt ziemlich viele davon (OPL2, OPL3SA, OPL3SA usw...). Viele Audiokarten basieren auf dieser Komponente oder einer, die kompatibel damit ist. Ein Beispiel sind die Chips von Crystal (unter anderen der weit verbreitete CS423x), die alle wie dieser Chipsatz arbeiten. Das ist auch der Grund, warum sie mit NetBSD funktionieren.

WSS ist kein Mikrochip, sondern ein Akronym für **`Windows Sound System`**. WSS ist der Name des NetBSD- Kerneltreibers, der das Soundsystem von Microsoft Windows unterstützt. Viele Soundkarten funktionieren mit Windows, weil sie diesem Standard entsprechen und dasselbe für NetBSD bereithalten.

Von den vielen Soundkarten, die ich getestet habe, funktioniert eine grosse Anzahl nur, wenn *opl** und *wss** aktiviert sind.

Du solltest keine Probleme damit haben, Soundblaster- Karten unter NetBSD zum Laufen zu bringen: Die meisten von ihnen werden unterstützt, einschliesslich der (Hifi- Freaks aufgemerkt!) SoundBlaster Live 1024!

Wenn alles funktioniert, kannst Du die nicht benötigten Geräte aus der Kernel-Konfurationsdatei entfernen.

nach oben

16.5. Erweiterte Kommandos

NetBSD kommt mit einer Anzahl von Kommandos, die das Audio- Device ansprechen. Das sind:

- audioctl
- mixerctl

- audioplay
- audiorecord

nach oben

16.5.1. audioctl

Audioctl erschien erstmals in NetBSD 1.3. Es wird benutzt, um einige Variablen, die die Audio-Ein- und Ausgabe betreffen; die Aufnahme- und Wiedergabefrequenzen gehören in diese Kategorie. Vorhandene Parameter können mit diesem Kommando angezeigt werden:

```
# audioctl -a | more
```

Als Beispiel: Wenn Du Musik in CD- Qualität hören willst, kannst Du das mit diesem Kommando tun:

```
# audioctl -w play=44100,2,16,slinear_le
```

Dieses Kommando setzt die Frequenz auf 44100 Hz, 2 Audiokanäle, 16 bit; slinear_le-Codierung

Du kannst die unterstützten Codierungen so abfragen:

```
# audioctl encodings
```

Mit diesem kommando werden alle Codierungen, die von der Soundkarte unterstützt werden, angezeigt.

nach oben

16.5.2. mixerctl

Damit kann man das Mixen von Audiodaten konfigurieren. Die Bedienung ist der von »audioctl« ähnlich.

nach oben

16.5.3. audioplay

Mit diesem Kommando kannst Du Audiodateien abspielen. Für besondere Anforderungen kannst Du eines der vielen Pakete installieren, das Dir erlaubt, auch Audiodaten in anderen Formaten (MP3 beispielweise) abzuspielen

16.5.4. audiorecord

Was vielleicht niemanden mehr überrascht: Dieses Kommando dient dazu, Audiodaten aufzunehmen.

nach oben

Inhalt

Kapitel 17: Quellcode per CVS organisieren

Inhalt dieses Dokuments:

Dieses Kapitel stammt von Reinoud Koornstra.

Mit CVS (Concurrent Versions System) ist der geneigte Benutzer in der Lage, seinen NetBSD-Quellbaum, unter Beachtung von Änderungen in den Quelldateien, aktuell zu halten. Es gibt drei Bäume, die regelmässig gewartet werden und die Du für CVS verwenden kannst, um das System »up to date« zu halten: Den Current- Quellbaum, mit dem man den scharfen, blutigen Schnitt in den Operationen an den Innereien von NetBSD verfolgen und testen kann. Dann gibt es den 1.5.1- Release- Quellbaum in dem Patches für Fehler, die im Laufe der Zeit gefunden wurden (unabhängig davon, ob es dabei um Sicherheit wie den Fragmentierungsfehler in ipf oder nur um Leistung ging). Bisweilen werden auch neue Dinge hinzugefügt; oder es wird ein Programm durch eine neue Version ersetzt, die für stabiler und sicherer in der Handhabung befunden wurde. Beispielsweise ist es jetzt möglich, die gesamte ARP- Tabelle mit dem arp-Kommando zu löschen: Das ist eine neue Funktion und kein Bugfix. Dann gibt es noch den 1.4-Baum, auf dem die Patches für bekannte Fehler zu finden sind. Dieser Baum wird allerdings nicht mehr weiterentwickelt.

nach oben

17.1. Beschaffung von System- Userlandcode

In NetBSD 1.4.x und 1.5.x ist CVS kein Bestandteil des eigentlichen Systems. In Version 1.6 wird sich das ändern. In der Current- Version gehört das auch schon dazu. Um CVS zu installieren, tue das hier:

```
% pkg_add ftp://ftp.netbsd.org/pub/NetBSD/packages/<OS Ver>/<arch>/All/
```

Wo sich <OS VER> und <arch> befinden kannst Du so herausfinden:

```
% sysctl kern.osrelease hw.machine_arch
```

Wenn Du die Quellen in einem Rutsch und ohne dass irgendwas in /usr/src zurückbleibt, updaten willst, kannst Du das so machen:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs checkout -rnetbsd-1-5 -PA src

Oder Du machst das mit ssh, um die Daten veschlüsselt zu halten:

% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs checkout -rnetbsd-1-5 -PA src
```

Damit ziehst Du Dir den Code von der 1.5er Ausgabe. Um den 1.5-Current- Code zu laden, gib einfach »-rnetbsd-1-5« in der letzten Zeile ein. Für 1.4 machst Du das mit »-rnetbsd-1-4« in der letzten Zeile.

Wenn Du den 1.5 Release- Quellbaum updaten willst, weil Du schon einen hast:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
password: anoncvs
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

Oder mit ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -rnetbsd-1-5 -PAd src
```

Um den 1.5-Current- Code zu laden, gib einfach »-rnetbsd-1-5« in der letzten Zeile ein. Für 1.4 machst Du das mit »-rnetbsd-1-4« in der letzten Zeile.

Wenn Du ein Update von einem unsauberen Baum ausführen willst, z.B. weil Du einen Teil, den gesamten Quellbaum oder den Kernel neu gebaut hast, ohne ein »make cleandir« zu machen, musst Du neue Objetktdateien im Quellbaum erstellen.

Die Objektverzeichnisse werden gebraucht, um ein CVS- Update zu machen. Ein unsauberer Baum ist ein Quellbaum, in dem Du Teile Deines Baumes gebaut hast; beispielsweise kompilierte Teile oder den gesamten Quellbaum, ohne ein »make clean« in diesen Teilen oder ein »make« cleandir im gesamten Quellbaum durchzuführen. Ansonsten wird CVS versuchen, Verzeichnisse zu erstellen, die dieselben Namen tragen, wie einige der Binaries. Das wird natürlich fehlschlagen. Wo Du ein Verzeichnis mit dem Namen »groff« hast, baust Du jetzt auch eine Datei mit dem Namen »groff«, CVS muss alle diesen leeren Verzeichnisse erstellen, nur um sie nach Beendigung der Arbeit automatisch entfernen zu können).

So in /usr/src:

```
% makedir /usr/obj
% make obj
```

Jetzt bist Du bereit für Dein CVS- Update. Oder mach ein **make cleandir** in /usr/src bevor Du CVS anwendest. Es ist einfacher und macht weniger Arbeit als das Erstellen der Objektverzeichnissen, als wenn Du von einem unsauberen Baum aus arbeitest. CVS ist viel schneller, als wenn man alles einzeln updatet. Ich weiss nicht genau, wie lange es dauert, die ganzen Quellen zu holen. Ich habe nur Erfahrungen mit T1 und schnelleren Leitungen, mit denen es nur eine Stunde oder etwas länger braucht, um den kompletten Quellcode zu laden; was natürlich auch von der aktuellen Verbindungqualität abhängt. Ich weiss nicht genau, wie lange das mit einem Modem dauert. Falls Du ein Modem für diese Arbeit verwendest, wirst Du die Daten komprimieren und dekomprimieren, wenn sie einmal übertragen sind. In diesem Fall nutze dieses Kommando:

```
% cvs -z5 checkout .........
```

oder

```
% cvs -z5 -d $CVSROOT update ......
```

Die »5« ist das Komprimierungslevel; Du kannst eine Ziffer zwischen 1 und 9 eingeben, wobei 1 die schellste Kompressionsmethode ist und 9 die beste, aber auch die langsamste. Denke aber auch daran, dass das zu einer häheren Belastung des CVS- Servers führt.

> **Anmerkung:** Du musst nach«/usr/src/sys/arch/$arch/compile/$kernel_conf_name« gehen, dort ein make clean ausführen und dieses Verzeichnis entfernen, wenn Du ein CVS- Update ausführen willst, ansonsten bleibt dieses Verzeichnis unsauber wie gehabt. Das sorgt dafür, dass der Quellcode des Kernels während des CVS-Updates ebenfalls in Ordung gebracht wird.

nach oben

17.2. pgksrc download

Pkgsrc (package source) ist ein Set, bestehend aus Programmen und Bibliotheken, die nach NetBSD portiert worden sind. Es ist sehr einfach, auf diesem Weg Software zu installieren oder sie wieder vom System zu putzen. Es holt sich die benötigten Dateien, patcht den Quellcode, wenn nötig, und kompiliert, konfiguriert und installiert die Binärdaten und man- Pages. Es enthält eine Datenbank mit allen Paketen, die installiert sind und mit Informationen, welche Dateien zu welchem Paket gehören, und wo sie bgelegt sind.

Um alle pkgsrc's in einem Rutsch zu nuckeln, machst Du das hier:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
(the password is: "anoncvs")
% cvs checkout -PA pkgsrc
```

Oder mit ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs checkout -PA pkgsrc
```

Damit wird ein Verzeichnis namens »pkgsc« in Deinem »/usr« erschaffen und alle Paketquellen werden unter »/usr/pkgsrc« abgelegt.

Um die pkgsrc zu updaten, mach mal das hier:

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs login
```

```
(the password is: "anoncvs")
% cvs -d $CVSROOT update -PAd pkgsrc
```

Oder mit ssh:

```
% setenv CVS_RSH ssh
% setenv CVSROOT anoncvs@anoncvs.netbsd.org:/cvsroot
% cd /usr
% cvs -d $CVSROOT update -PAd pkgsrc
```

Wie auch immer; stelle sicher, dass das pkgsrc- Verzeichnis sauber ist, wenn Du mit dem Updaten anfängst. Mache also lieber ein »make clean« in »/usr/pkgsrc«, wenn Du Dir nicht 100%ig sicher bist.

Seitenanfang

Inhalt

Kapitel 18: Weitere wichtige Dinge

Inhalt:

Diese Kapitel ist eine Sammlung von verschiedenen Themen in loser Reihenfolge, die in den anderen Kapiteln keinen Platz fanden

18.1. Boot- und Installationsdiskettendisketten für i386er PCs

Diese Sektion (Boot- und Installationsdisketten) ist ein Beitrag von Eric Delcamp

Zu allererst brauchen wir einen Kernel mit aktiviertem »vnd« Pseudo- Device (Im GENERIC-Kernel ist das integriert).

1. Als erstes musst Du einen gültigen Kernel backen, um ihn auf die Disketten zu packen; nennen wir ihn FLOPPY. Dieser Kernel muss ein Derivat von einem INSTALL- Modell sein. Dann hast Du eine gültige »/sys/arch/i386/FLOPPY/netbsd«- Datei.

2. Gehe nach » /usr/src/distrib/i386/floppies/ramdisk« und gib das hier ein:

   ```
   # make
   ```

   Damit erzugst Du die »ramdisk.fs«- Datei im Verzeichnis.

3. Gehe nach«/usr/src/distrib/i386/floppies/fdset« und mache das hier:

   ```
   # make KERN=/sys/arch/i386/compile/FLOPPY/netbsd
   ```

   Damit werden je nach Kernelgrösse eine oder zwei Dateien mit den Namen »boot1.fs und »boot2.fs« erzeugt.

4. Diese Dateien werden mit den folgenden Kommandos auf Disketten transferiert:

   ```
   # dd if=boot1.fs of=/dev/fd0a bs=36b
   # dd if=boot2.fs of=/dev/fd0a bs=36b
   ```

5. Lege die erste Diskette in das Laufwerk und schalte den Rechner ein.

nach oben

18.2. Eine CD-ROM

Um eine Daten-CD-ROM herzustellen, können mkisofs und cdrecord benutzt werden: Es werden sowohl IDE- als auch SCSI- Brenner unterstützt. IDE/ATAPI- Laufwerke werden von NetBSD ohne Emulation unterstützt, weil der Treiber die ATAPI- Kommandos direkt entgegennehmen kann, was eine einfache und elegante Lösung ist.

Zwei Schritte sind nötig, um eine CD herzustellen: Als Erstes muss das ISO- Image der CD mittels mkisofs hergestellt werden. Der nächste Schritt ist das Schreiben des Images auf eine CD mit cdrecord. Im folgenden Beispiel wird eine IDE/ATAPI- Brenner benutzt. dmesg gibt uns das hier aus:

```
cd1 at atapibus1 drive 0: <HP CD-Writer Plus 8100> type 5 cdrom removab
```

> **Hinweis:** Beim Brennen die Ausführungsgeschwindigkeit kritisch: Der Datenstrom zum Brenner muss konstant und pausenlos sein. Der Datenpuffer des Laufwerks darf nie leer werden. Das heisst, dass es besser ist, eine CD zu brennen, wenn das System gerade wenig zu tun hat (Lasse also das Kompilieren eines Kernel oder das Erzeugen eines MP3 bleiben, solange cdrecord läuft...)

nach oben

18.2.1. Erstellen eines ISO-Images ISO

CD-ROM-Dateisysteme

CD-ROM's können in verschiedenen (und manchmal inkompatiblen) Dateisystemen hergestellt werden. Daher ist es möglich, dass eine CD, die auf einem System hergestellt wurde, auf einem anderen nicht lesbar ist, oder dass Informationen verloren gegangen sind. Diese Situation ist manifestiert, wenn Du beispielsweise eine CD unter NetBSD lesen willst, di unter Windows entstanden ist. Die folgenden Absätze geben einen groben Überblick über die am weitesten verbreiteten Dateisysteme für CD-Roms.

»ISO9660« war das erste Format, das 1998 festgelegt wurde und repräsentiert eine Art verbreiteten Standard zwischen verschiedenen Betriebssystemen: Apple, MS-DOS, UNIX und VMS. Eine CD-Rom, die unter Windows geschrieben wurde gibt es weitere Grenzen: Das 8.3- Format bei den Dateibezeichnern und nur 8 Levels in der Verzeichnishierarchie. Aus diesem Grund werden beispielsweise bei einer unter Windows erstellten NetBSD-Installations-CD die Dateinamen abgeschnitten, wenn man sich den Inhalt der CD im Explorer anschaut.

Um sich um diese Grenze herumzumogeln, wurden einige Erweiterungen des originalen ISO9660-Formates eingeführt. Der Nachteil dieser Erweiterungen ist es, dass nicht alle dieser Erweiterungen auf allen Plattformen unterstützt werden.

Das Joliet-Format, eingeführt von Microsoft, erweitert ISO9660 um die Unterstützung langer Dateinamen und tiefere Verzeichnishierarchien. Dieses Format kann gewöhnlich nicht von Unix- Systemen gelesen werden (wird aber von NetBSD-current unterstützt).

Die Rock-Ridge-Erweiterungen wurden für Unixsysteme eingeführt, um Unixkonventionen für diese Dateisysteme zu unterstützen, ohne die Kompatibilität mit der originalen ISO9660

einzubüssen. Das ist das Format, das verwendet wird, wenn eine CD unter NetBSD oder einem anderen Unix-System hergestellt wird (Im Beispiel erklären wir, wie man in beiden Formaten (Joliet und RockRidge) eine CD herstellen kann.

Zusätzlich zu diesen Normen gibt es da noch ein andere Standards (»El Torito« zum Herstellen bootfähiger CDs ist so einer, der von allen neueren PCs unterstützt wird).

Seit ISO-Images dazu tendieren, ziemlich gross zu sein, ist es besser, vorher zu überprüfen, ob der Platz auf der Festplatte hinreichend ist, um die zu schreibenden Daten unterbringen zu können. Um ein Image von Daten zu erstellen, die im Verzeichnis »mydata« und seinen nterverzeichnissen liegen, schreibe folgrndes Kommando:

```
# mkisofs -aflrTv -o cdimage mydata/
```

Wenn die »cdimage«-Datei erstellt wurde, kann sie wie ein reguläres Dateisystem gelesen und überprüft werden, um sicherzustellen, dass dort keine Fehler sind, bevor die CD geschrieben wird. Beispiel:

```
# 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
```

Der Wert 556 ist das Ergebnis der Grösse der »cdimage«-Datei geteilt durch 512.

> **Hybrid-CDs:** mkisofs kann auch CDs im Joliet-Format erstellen; solche CDs sind auf Microsoft- Plattformen lesbar. Es ist ausserdem möglich, Hybrid-CDs zu erstellen, die sowohl RockRidge- als auch Joliet-Erweiterungen enthalten und damit sowohl auf Unix- als auch auf Windows- Plattformen lesbar sind. Beispiel:
>
> ```
> $ mkisofs -l -J -R -o cd.iso mydata/
> ```
>
> Lies bitte auch die »mkisofs«- Manpage, um weitere Details zu erfahren.

18.2.2. Das Schreiben des Images auf eine CD

Im zweiten Schritt wird das Image mit diesem Kommando auf die CD geschrieben:

```
# cdrecord -v speed=2 dev=/dev/rcd1d cdimage
```

> **Hinweis:** Bei ATAPI- Geräten muss »rc#d« benutzt werden, weil die »a«-Geräte keine ATAPI- Kommandos verstehen.

Bevor die Cd geschrieben wird, kann ein Testlauf durchgeführt werden, indem man den Laser ausgeschaltet lässt: Einfach die »-dummy« und »-nofix«-Optionen an das Kommando anhängen. Beispiel:

```
# cdrecord -v -dummy -nofix speed=2 dev=/dev/rcd1d cdimage
```

Die zwei Schritte, das Erstellen und das Brennen eines Images, können zu einem eizigen Kommando zusammengefasst werden, ohne eine (riesige) Teporäre Datei auf der Platte erstellen zu müssen. Das Kommando dafür sieht so aus:

```
# (nice -18 mkisofs -aflrT mydata/) | cdrecord -v fs=16m speed=2 dev=/d
```

Die Option »fs=16m« wird benutzt, um einen grösseref »fifo« (first in-first out) einzurichten, um einen Buffer-Underflow zu vermeiden (was beteuten würde, dass cdrecord keine Daten zum Schreiben mehr hätte).

nach oben

18.2.3. CDs kopieren

Um eine CD direkt zu kopieren, kann die »-isosize«- Option von cdrecord benutzt werden. Beispiel:

```
# cdrecord -v fs=16m -isosize speed=2 dev=/dev/rcd1d /dev/rcd0d
```

> **Hinweis:** Wenn Du zwei IDE/ATAPI CD(-RW)- Geräte hast, ist es besser, wenn sie an zwei verschiedene Controller angeschlossen sind (eins an den primären und eins an den sekundären Controller), weil der Datenfluss besser ist. Das hier ist eine Beispielkonfiguration:
>
> ```
> wd0: hard disk, IDE primary master
> cd0: CD reader, IDE primary slave
> cd1: CD writer, IDE secondary master
> ```

nach oben

18.2.4. Herstellung einer bootfähigen CD

Das Erstellen einer bootfähigen CD ist nur eine Frage des Besitzens einer bootfähigen Binärdatei: Diese Bootdatei emuliert eine Diskette. Dann kann mkisofs mit der »b«-Option benutzt werden. Beispiel:

```
# mkisofs -avr -b boot.fs -o cdimage mydata/
```

»boot.fs« ist das Boot- Binary für die CD. Denke daran, dass der Pfad von »boot.fs« relativ zum »mydata«-Verzeichnis liegen muss.

nach oben

18.3. Synchronisation der Systemuhr

Es ist nichts ungewöhnliches, herauszufinden, dass die Systemuhr falsch geht, oft um ein paar Minuten: Aus einigen befremdlichen Gründen scheint es, dass die Uhren in den Computern nicht sehr genau sind. Das Problem wird aktuell, wenn Du viele vernetzte Hosts zu administrieren hast: Alle Uhren synchron zu halten kann leicht zu einem Albtraum werden. Als Problemlösung bietet sich die Hilfe des NTP- Protokolls in der Version 3 an: Damit können die Uhren in den Rechnern synchronisiert werden, indem einer oder mehrere NTP- Server benutzt werden.

Dank des NTP- Protokolls ist es möglich, sowohl die Uhr einer einzigen Workstation als auch ein gesamtes Netzwerk zu synchronisieren. Das NTP- Protokoll ist ziemlich komplex, weil eine hierarchische Master-Slave-Struktur von Servern definiert wird, die in Strata unterteilt wird: Die Spitze der Hierarchie besetzen die Stratum-1- Server, die mit einer externen Uhr verbunden sind (vielleicht mit einer Funkuhr). Um eine hohe Genaugkeit zu sichern. Darunter befinden sich die Stratum-2-Server, die ihre Uhren mit Stratum-1 synchronisieren und so weiter. Die Genauigkeit sinkt, je weiter wir in der Hierarchie absteigen. Diese hierarchische Struktur verhindert, dass eine Überlastung des Stratum-1-Servers entsteht, weil alle Hosts ihre Anfragen an den selben öffentlichen Stratum-1-Server schicken. Wenn Du ein Netzerk synchronisieren willst, verbindest Du nicht alle Hosts mit demselben öffentlichen Stratum-1-Server. Stattdessen baust Du einen lokalen Server auf, der mit dem Hauptserver verbunden ist; Die anderen Host-Uhren synchronisieren sich dann mit dem lokalen Server.

Zum Glück musst Du die Datails des NTP- Protokolls und seiner Implementation nicht verstehen (Bei Interesse: RFC 1305 lesen). Du musst nur wissen, wie das konfiguriert wird und ein paar Programme starten. Das Basis-System von NetBSD bringt die nötigen Tools für dieses Protokoll mit (und für andere zeitrelevante Protokolle, wie wir sehen werden). Es sind Derivate der »xntp«- Implementation. Diese Sektion beschreibt eine einfache Methode, die uns immer zu einer korrekten Systemzeit verhilft.

Als erstes solltest Du wissen, wo Du einen öffentlichen NTP-Server als Referenz finden kannst: Eine Liste findet sich unter http://www.eecis.udel.edu/~mills/ntp/servers.html . In Italien können die beiden Stratum-1-Server tempo.cstv.to.cnr.it und time.ien.it benutzt werden:

Als nächstes stellst Du die Systemzeit mit diesem Kommando ein:

```
# ntpdate -b tempo.cstv.to.cnr.it time.ien.it
```

In Deutschland sind die hier zwei Beispiele für öffentliche Server:

ntps1-1.tu-berlin.de (IP: 130.149.17.8); Kontaktadresse Gerard Gschwind (gg@cs.tu-berlin.de)

ntps-1-1.uni-osnabrueck.de (IP: 131.173.17.7); Konakt über Gernot Skalla (timeadm@uni-osnabrueck.de)

Ersetze die Namen der Server durch die, die Du verwendest. Option »-b« gibt an »ntpupdate« die Anweisung, die Systemzeit mit dem »settimeofday«- Systemaufruf zu setzen, anstatt das mit »adjtime« zu erledigen (Standard). Diese Option ist zu empfehlen, wenn der Unterschied zwischen der Ortszeit und der genauen Zeit vernachlässigt werden kann.

Wie Du gesehen hast, ist die Benutzung von ntpdate nicht schwierig. Der näschste Schritt ist, das Programm automatisch zu starten, um die Systemzeit immer aktuell zu halten. Wenn Du eine permanente Vernbindung ins Internet hast, kannst Du das Programm beim Booten mit diesem Kommando in der »/etc/rc.conf«-Datei starten:

```
ntpdate=YES       ntpdate_hosts="time.ien.it"
```

Der Name des NTP-Servers wird in der »ntpdate_hosts«- Variablen definiert. Wenn Du das Feld leer lässt, wird das Bootscript versuchen, den Namen aus der Datei »/etc/ntp.conf« zu extrahieren.

Wenn Du keine permanente Internetverbindung hast (Dialup- Verbindung über einen ISP) kannst Du ntpdate aus dem »ip-up«- Script heraus starten, wie das in Kapitel 9 beschrieben wird. In diesem Fall muss diese Zeile an »ip-up« drangehängt werden:

```
/usr/sbin/ntpdate -s -b time.ien.it
```

... oder eben ein einheimischer Server. Der Pfad ist für das Script nötig; es wird sonst möglicherweise das Programm nicht finden. Mit Option »-s« wird der Logging- Output vom Standard-Output (Defaultwert) auf das Syslog- System der Maschine umgeleitet. Das bedeutet, dass die Nachrichten von »ntpdate« für gewöhnlich in »/var/log/messages« abgelegt werden.

Neben »ntpdate« gibt es noch weitere nützliche NTP- Kommandos. Es ist auch möglich, einen lokalen Rechner in einen NTP- Server für die Rechner im lokalen Netz umzuwandeln. Der lokale Server synchronisiert seine Uhr mit einem öffentlichen Server. Für diese Konfiguration muss der »xntp«- Daemon benutzt werden. Ausserdem ist die &raqo;/etc/ntp.conf«-Datei zu erstellen:

```
server time.ien.it
server tempo.cstv.to.cnr.it
```

Xntpd kann aus »rc.conf« heraus gestartet werden:

```
xntpd=YES
```

NTP ist nicht die einzige Möglichkeit, ein Netzwerk zu synchronisieren: Du kannst das auch mit dem »timed«- Daemon machen, der mal für 4.3BSD entwickelt wirden ist. Timed benutzt ebenfalls eine Master-Slave-Hierarchie: Wenn das auf einem Host gestartet wird, fragt timed das Netzwerk nach einem Master und stellt die lokale Zeit sofort ein. Auch eine gemischte Struktur aus timed und xntpd ist machbar. Einer der lokalen Rechner bezieht die Zeit von einem öffentlichen NTP-Server für die Maschinen im lokalen Netz, die seine Clients werden und sich mittels timed mit dem Server synchronisieren. Das heisst auch, dass auf dem lokalen Server timed und NTP laufen müssen. Man muss dabei Vorsicht walten lassen, damit beide nicht miteinander konkurrieren (timed muss mit der »-F Rechnername«-Option gestartet werden, um die lokale Uhr einzustellen.

nach oben

18.4. Den Bootmanager installieren

Sysinst, das NetBSD- Installationsprogramm, kann den NetBSD- Bootmanager auf der Festplatte instalieren. Der Bootmanager kann auch später noch installiert oder umkonfiguriert werden, wenn es sein muss; mit dem »fdisk«- Kommando:

```
# fdisk -B wd0
```

Wenn NetBSD nicht von der Harddisk bootet, kannst Du auch von der Installationsdiskette aus booten und den Kernel auf der Platte starten. Lege die Installationsdiskette ein und, gib am Bootprompt, dieses Kommando hier ein:

```
> boot wd0a:netbsd
```

Damit bootet der Kernel auf der Festplatte (Benutze das korrekte Gerät, beispielsweise sd0a auf einer SCSI-Platte).

> **Hinweis:** Manchmal ist das Resultat von »fdisk -B« nicht wie erwartet (Zumindest ist mir das schon passiert), möglicherweise, wenn Du andere Systeme wie Windows 9X installierst oder deinstallierst. In so einem Fall versuche es mal mit »fdisk/mbr« von DOS aus und anschliessend nochmal mit »fdisk« von NetBSD

nach oben

18.5. Löschen des Disklabels

Dieses ist keine Operation, die Du allzuoft vornehmen musst. Es kann aber nützlich sein zu wissen, wie das im Bedarfsfall geht. Bitte sei Dir sicher, dass Du genau weisst, was Du da tust, bevor Du sowas machst. Beispiel:

```
# dd if=/dev/zero of=/dev/rwd0c bs=8k count=1
```

Das vorige Kommando löscht das Disklabel (allerdings nicht die MBR- Partitionstabelle). Um Die Platte komplett zu löschen, muss das wd0d- Gerät benutzt werden:

```
# dd if=/dev/zero of=/dev/rwd0d bs=8k
```

nach oben

18.6. Lautsprecher

Ich fand diesen Tip in einer Mailingliste (kann mich aber an den Autor nicht mehr erinnern). Um ein Geräusch über einen Lautsprecher auszugeben (Beispielsweise am Ende eines langen Scriptes) kann das »spkr«- Gerät im Kernel benutzt werden, das auf »/dev/speaker« gemappt wird:

```
echo 'BPBPBPBPBP' > /dev/speaker
```

> **Hinweis:** das »spkr«- Gerät ist nicht im Standardkernel enthalten. Du brauchst einen eigenen Kernel.

nach oben

18.7. Uuups... root- Passwort vergessen

Wenn Du das Root-Passwort vergessen hast, ist nicht alles verloren. Du kannst das System immer noch mit diesen Schritten »recovern«: Boote im Sigle-User-Modus; mounte »/« und ändere das Passwort von root. Im Detail:

1. Single- User: Wenn der Bootprompt erscheint und der Boot- Countdown startet, gib das hier ein:

   ```
   > boot -s
   ```

2. An diesem Prompt:
   ```
   Enter pathname of shell or RETURN for sh:
   ```

   ...drücke Enter.

3. Setze diese Kommandos ab:

   ```
   # fsck -y /
   # mount -u /
   # fsck -y /usr
   # mount /usr
   ```

4. Ändere das Root- Passwort mit »passwd«

5. Mit »exit« gehst Du in den Multiuser- Modus.

nach oben

18.8. Eine neue Festplatte einbinden

Diese Sektion beschreibt, wie eine neue Festplatte in ein bereits laufendes NetBSD- System integriert wird. In unserem Beispiel geht es um einen neuen Controller und eine neue Platte, die integriert werden sollen. Wenn Du keinen neuen Controller integrieren musst, kannst Du den relevanten Teil übergehen und direkt an die Konfiguration der Festplatte gehen. Die Installation einer IDE- Platte ist identisch, nur die Gerätenamen sind anders (wd# statt sd#).

Wie immer, wenn Du neue Hardware kaufst, solltest Du die Kompatibilitätsliste durchgehen und sicherstellen, dass das neue Gerät vom System unterstützt wird.

Wenn der SCSI- Controller in das System eingebaut wurde und die neue Festplatte angeschlossen ist, ist es Zeit, den Rechner zu starten und mittels »dmesg« zu prüfen, ob das Gerät richtig erkannt wurde. Das hier ist der Output füe einen NCR-875- Controller:

```
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
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 sector
```

Wenn das Gerät nicht im Output erscheint, prüfe, ob das Gerät vom verwendeten Kernel unterstützt wird; wenn nötig, kompiliere Deinen eigenen Kernel wie in Kapitel 7 beschrieben.

Jetzt können die Partitionen mittels »fdisk« erstellt werden. Als erstes muss der aktuelle Zustand der Platte festgestellt werden:

```
# 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 diesem Beispiel liegt auf der Platte schon eine DOS- Partition, die gelöscht und durch eine NetBSD- Partition ersetzt werden soll. Mit dem Kommando »fdisk -u sd0« kannst Du die Partitionen interaktiv ändern. Die modifizierten Daten werden erst auf die Platte geschrieben, wenn Du fdisk verlässt und Du wirst gefragt werden, bevor geschrieben wird. Du kannst also unverkrampft an die Arbeit gehen.

Festplattengeometrien:

Die Geometrie der Platte, die von fdisk ausgegeben wird, kann einem schon mal etwas konfus erscheinen. Dmesg berichtet von 4226725 Sektoren mit 8188/2/172 für C/H/S, aber 8188*3*172 ergibt 4225008 und nicht 4226725. Was ist passiert? Die meisten modernen Festplatten besitzen keine statische Geometrie; und die Anzahl der Sektoren hängt ab von der Anzahl der Zylinder: Der einzige interessante Parameter ist die Anzahl der Sektoren. Die Platte liefert die C/H/S- Werte; aber die Geometrie ist fiktiv: Der Wert 172 ist das ergebnis aus der Gesamtzahl der Sektoren (4226725) geteilt durch 8188 und dann durch 3.

Um die Dinge noch weiter zu erschweren, benutzt das BIOS noch eine andere »Fake-Geometrie«(C/H/S 524/128/63) woraus ein Zahl von 4225536 resultiert. Ein Wert, der sich weiter an den realen 425008 orientiert. Um die Platte zu partitionieren, benutzen wir die BIOS- Geometrie. Um die Kompatibilität zu erhalten, werden wir einige Sektoren verlieren (4226725-4225536= 1189 Sektoren = 594 KB).

Um die BIOS- Partitionen zu erstellen muss das Kommando »fdisk -u« benutzt werden. Hier ist das Ergebnis:

```
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>
```

Es ist jetzt Zeit, das Disklabel der NetBSD- Partition zu erstellen. Das hier sind die korrekten Schritte:

```
# disklabel sd0 > tempfile
# vi tempfile
# disklabel -R -r sd0 tempfile
```

Wenn Du versuchst, ein Disklabel direkt zu erstellen:

```
# disklabel -e sd0
```

...bekommst Du diese Antwort:

```
disklabel: ioctl DIOCWDINFO: No disk label on disk;
use "disklabel -r" to install initial label
```

...weil das Disklabel nicht auf der Platte existiert.

Jetzt erstellen wir ein paar Disklabel- Partitionen, indem wir das bereits erklärte »tempfile« bearbeiten. Das Ergebnis:

```
#      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*)
```

> **Hinweis:** Wenn das Disklabel erstellt worden ist, kann man das noch optimieren, wenn man den Output des Kommandos »newfs -n /dev/sd0a« durcharbeitet, der uns vor nicht lokalisierten Sektoren am Ende einer Disklabel- Partition warnt. Die Werte, die von newfs ausgegeben werden, können zur Anpassung der Partitionsgrössen in einem iterativen Prozess verwendet werden.

Am Schluss der Operation steht das Erstellen der Dateisysteme auf den neu definierten Partitionen (a und e):

```
# newfs /dev/sd0a
# newfs /dev/sd0e
```

Jetzt kann die Platte gemountet und benutzt werden. Beispiel:

```
# mount /dev/sd0a /mnt
```

nach oben

18.9. Password- Datei »busy«?

Wenn Du versuchst, ein Passwort zu ändern und die mysteriöse Nachricht »Password file busy« bekommst, kann das heissen, dass »/etc/ptmp« nicht vom System gelöscht wurde. Diese Datei ist eine temporäre Kopie der »/etc/master.passwd«- Datei. Stelle sicher, dass Du keine Informationen verlierst und lösche sie dann **(»ptmp«, nicht etwa »master.passwd«!!!!!).**

> **Hinweis:** Wenn die Datei »etc/ptmp« existiert, kann beim Systemstart eine Warnung auflaufen. Beispiel:
>
> ```
> root: password file may be incorrect - /etc/ptmp exists
> ```

nach oben

18.10. Gerätedateien in /dev neu erstellen

Beitrag von Reinoud Koornstra

Fahre das System zuerst in den Single- User- Modus herunter. Die Partitionen bleiben als »rw«(read-write) gemountet. Du kannst das tun, indem Du einfach »shutdown now« eingibst, wenn Du Dich im Multiuser- Modus befindest oder mit der Option »-s« einen Reboot ausführen und »/« und »/dev« lese- und beschreibbar machen indem Du das hier tust:

```
# mount -u /
# mount -u /dev
```

Danach:

```
# mkdir /nudev
# cd /nudev
# cp /dev/MAKEDEV* .
# sh ./MAKEDEV all
# cd /
# mv dev odev
# mv nudev dev
# rm -r odev
```

Oder wenn Du alle Sourcen in »/usr/src« hast:

```
# 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
```

Du kannst $arch mit

```
# uname -m
```

bestimmen oder mit

```
# sysctl hw.machine_arch
```

Auf dem zweiten Weg (Das Kopieren der MAKEDEVs aus dem Quellbaum) fügt zumindest in der i386- Architektur zusätzliche Geräte hinzu. Es ist beispielsweise jetzt möglich, 16 Partitionen zu haben anstatt nur 8. Wenn Du die »alten« MAKEDEVs aus »/dev« benutzst, werden die zusätzlichen Gerätedateien nicht hergestellt.

nach oben

Inhalt

Anhang A: Information und Historisches

A.1. Geschichte

Dieser Guide wurde als eine Sammlung von Notizen ins Leben gerufen, die ich für mich selbst gemacht habe. Als ich realisierte, dass sie auch für andere NetBSD- User von Nutzen sein könnten, begann ich die Notizen zu sammeln und produzierte die erste Version mit dem groff-Formatierer. Um mit wenig Aufwand einen grösseren Bereich an Formaten abdecken zu können (z.B. HTML, RTF), machte ich den »Fehler«, auf SGML/DocBook umzusteigen, welches derzeit das Format der Sourcen ist.

Diese Open-Source- Tools wurden zum Schreiben und Formatieren des Guides benutzt:

- der Standardeditor von NetBSD (nvi.)
- Jade, das DSSSL- Umwandlungen ausführt. Damit werden HTML- und RTF- Dokumente direkt hergestellt.
- Das TeX-System aus der Packages-Collection. TeX ist das Backend, mit dem das PS- und das PDF- Format erzeugt wird.
- Tgif für die Zeichnungen.
- Gimp und xv, um die Bildformate zu konvertieren und kleinere Änderungen an den Zeichnungen vorzunehmen.

Vielen Dank an die Leute, die sich mit der Entwicklung dieser grossartigen Werkzeuge befassen.

Inhaltsverzeichnis

Anhang B. SGML/Docbook: Eine kleine Einführung

Inhalt:

Dieser Anhang beschreibt die Installation der Tools, die gebraucht werden, um eine formatierte Variante des NetBSD- Guides zu produzieren. SGML/Docbook und die DSSSL werden hier nicht beschrieben, aber am Ende dieses Anhanges gibt es einen Absatz mit ein paar Links zu nützlichen Dokumenten, die Dir beim Start helfen.

Die SGML-Docbook- Umgebung kann mit dem »netbsd-docs«- Metapaket installiert werden. Das ist der einfachste Weg, und Du weisst, wie das funktioniert. Dieser Anhang beschreibt die Installation der Komponenten Stück für Stück und kann genutzt werden, um eine besser abgestimmte Installation zu bekommen oder um ein Troubleshooting durchzuführen, wenn etwas nicht so funktioniert wie erwartet.

> **Hinweis:** Das »netbsd-docs«- Metapaket installiert einige Pakete, die in diesem Dokument nicht beschrieben werden, weil sie für den NetBSD- Guide nicht gebraucht werden:

- iso12083-1993
- unproven-pthreads-0.17nb2
- opensp-1.4
- html-4.0b

In diesem Dokument werden zu die vorkompilierten Pakete zur Installation verwendet. Details zu Paketen im Allgemeinen und deren Installation stehen in Kapitel 8.

> **Hinweis:** Die Versionsnummern der Tools, die wir installieren werden, können sich ändern, wenn neue Versionen zum Packages- System dazukommen.

nach oben

B.1. Was ist SGML/DocBook?

SGML (Standard Generalized Markup Language) ist eine Sprache, die benutzt wird, um andere Sprachen zu definieren, die auf Markups basieren (Beispielsweise die gültigen Konstruktoren). HTML (noch ein Beispiel) kann mittels SGML definiert werden. Wenn Du ein Programmierer bist, stelle Dir SGML wie das BNF (Backus-Naur-Form) vor; ein Tool, mit dem man Grammatiken definieren kann.

Docbook ist ein Markup Template, das mit SGML definiert wird. Docbook listet die gültigen Tags, die in einem Docbook- Dokument benutzt werden können und wie sie miteinander

kombiniert werden. Wenn Du Programmierer bist, stelle Dir Docbook als eine Sprache vor, die mit dem BNF spezifiziert worden ist. Als Beispiel besagt es, dass die Tags

```
<para> ... </para>
```

einen Absatz definieren und dass ein <para> in einem <sect1> sein kann, aber nicht ein <sect1> in einem <para> stehen darf und kann.

Deshalb; wenn Du ein Dokument in Docbook schreibst und nicht in SGML, ist in dieser Betrachtungsweise Docbook das Gegenstück zu HTML (wobei die Auswahl an Markups grösser ist und sich die konzepte unterscheiden).

Die Docbook- Spezifikation (z.B.: eine Liste mit Tags und Regeln) wird eine DTD genannt (Document Type Definition).

In Kürze: Eine DTD definiert, wie Deine Quelldokumente aussehen, aber sie gibt Dir keine Hinweise auf das Format der endgültigen (kompilierten) Dokumente. Ein weiterer Schritt ist da gefordert: Die Docbook- Dokumente müssen in eine andere Repräsentationsform konvertiert werden; beispielsweise HTML oder PDF. Diesen Schritt erledigt ein Tool namens Jade, das die DSSSL- Transformierungen den Quelldokumenten hinzufügt. DSSSL (Document Style Semantics and Specification Language) ist ein Format, das benutzt wird, um die Stylesheets hinzuzufügen, die benötigt werden, um aus Docbook in andere Formate zu konvertieren.

Das Leben eines DocBook- Dokumentes ist wie folgt:

- Docbook Quelldokument
- Das Docbook- DTD wird von nsgmls benutzt um die Dokumente »gültig zu machen«
- Jade wird benutzt, um die DSSL- Stylesheets in die Dokumente einzubauen un ein neues Dokument zu generieren.

  Es ist jetzt immer noch nicht möglich, das Dokument zu drucken oder es anzuschauen. Das neue Dokument ist nur das Originaldokument, dem die Direktiven für die Formatierung hinzugefügt und die Docbook- Tags entnommen wurden. Es kann HTML, RTF, TeX oder sonstwas sein.
- Ein Formatierer (HTML- Viewer, Word oder einen anderen RTF- Prozessor, TeX,...) wird benutzt, um die endgültige Version des Quelldokuments zu erstellen.

Daher brauchst Du das hier um zu starten:

- einen DTD für Docbook.
- Die DSSSL-Stylesheets für Jade, um HTML/RTF etc. zu generieren.
- Das Jade- Programm und den nsgmls- Parser.

nach oben

B.2. Jade

Jade ist ein SGML/XML- Parser, der die DSSSL- Engine implementiert. Das Jade- Paket

schliesst den Parser mit ein (nsgmls).

> **Hinweis:** OpenJade ist eine neuere Version von Jade. Sie lässt sich zu Zeit nicht mit NetBSD kompilieren. Für den NetBSD- Guide brauchst Du das allerdings auch nicht.

Installation von Jade aus einem vorkompiliertem Paket:

```
# pkg_add jade-1.2.1.tgz
```

Du wirst etwas Dokumentation in »/usr/pkg/share/doc/jade/index.htm« finden, aber das wichtigste installierte Verzeichnis ist »/usr/pkg/share/sgml/jade/«: Da findet sich die »catalog«-Datei von Jade.

nach oben

B.3. Docbook

Das nächste Ding, das Du installieren musst, ist das Docbook DTD (Beispiel: Die Schablone, die benutzt wird, um Docbook- Dokumente zu schreiben).

Dieses Paket benötigt das Package mit den Zeichenschablonen aus ISO 8879:1986. Wir installieren also zuerst die Zeichenschablonen:

```
# pkg_add iso8879-1986.tgz
```

Diese Schablonen werden im »usr/pkg/share/sgml/iso8879«- Verzeichnis installiert, wobei »/usr/pkg/share/sgml/iso8879/catalog« die Katalogdatei ist.

Jetzt können wir das Docbook- DTD installieren:

```
# pkg_add docbook-4.1.tgz
```

Ungeachtet seines Namens installiert dieses Paket diverse Versionen von Docbook DTD (z.B.: 2.4.1, 3.0, 3.1, 4.0, 4.1). Damit kannst Du auch Dokumente bearbeiten, die verschiedene Versionen des DTD verwenden.

> **Hinweis:** Die aktuelle Version des NetBSD- Guides benutzt Version 4.1 des Docbook- DTD. Deshalb sind die anderen Versionen für diesen Guide nicht unbedingt nötig. Die einzelnen Versionen benötigen jeweils weniger als 250 KB. Du solltest sie behalten, um andere Dokumente bearbeiten zu können.

Die Wurzel der Installation ist »usr/pkg/share/sgml/docbook/4.1«. Jede Version benutzt ein separates Verzeichnis und jede hat ihren eigenen Katalog; z.B.: »/usr/pkg/share/sgml/docbook/4.1/catalog«.

nach oben

B.4. Die DSSSL Stylesheets

Jetzt sollten wir die DSSSL- Stylesheets installieren:

```
# pkg_add dsssl-docbook-modular-1.57.tgz
```

Die Stylesheets installieren auch ihren Katalog; in »/usr/pkg/share/sgml/docbook/dsssl/modular/catalog«. Du Dokumentation über die modularen Stylesheets findest Du in »/usr/pkg/share/sgml/docbook/dsssl/modular/doc/index.html«.

nach oben

B.5. Die Benutzung der Tools

Lass uns mal versuchen, mit den neu installierten Tools eine HTML- Version des englischen Guides herzustellen.

Gehe in das Basisverzeichnis und dann:

```
$ cd en
$ make netbsd.html
```

Du wirst eine lange Liste mit Fehlermeldungen bekommen, weil nsgmls, der SGML- Parser, die Katalogdateien nicht findet. Gib dagegen mal folgende Kommandos ein (und ergänze Deine »~./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_CAT
export SGML_CATALOG_FILES
```

> **Hinweis:** Modifiziere die Kommandos, wenn Du eine C- Shell benutzst.

Wenn die »SGML_CATALOG_FILES«- Umgebungsvariable aktiv ist, mach ein neues

```
$ 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
```

Diesmal geht alles glatt und die HTML- Version des Guides wird generiert. Die RTF- Version entsteht auf demselben Weg.

```
$ 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
```

Mit dieser Installation kannst Du nur HTML- und RTF- Versionen herstellen. Das Generieren von PS- und PDF- Formaten verlagt die Installation von TeX und Jadetex.

nach oben

B.6. Ein alternativer Zugang zu den Katalogdaten

In meinen Installationen erstelle ich im Normalfall einen »Masterkatalog«, der die Referenz zu allen anderen Katalogen darstellt. Wenn Du diese Art des Zugriffs magst, erstelle eine »/usr/pkg/share/sgml/catalog«- Datei, die aus folgenden Zeilen besteht:

```
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"
```

Wenn Du diese Datei erstellt hast, kannst Du Deine »~/.profile«- Datei so modifizieren:

```
SGML_CATALOG_FILES=/usr/pkg/share/sgml/catalog
export SGML_CATALOG_FILES
```

nach oben

B.7. PostScript-Output

Für eine druckfähige Version des Guides musst Du so vorgehen:

- TeX installieren
- Die Silbentrennung für die italienische Sprache einschalten
- Das hugelatex-Format für jadetex erstellen
- jadetex installieren

In den folgenden Sektionen wird jeder Schritt im Detail beschrieben.

nach oben

B.7.1. TeX installieren

Du musst nicht spezielles tun, um TeX zu installieren; Es ist ein Riesenpaket, aber dank des Package-Systems trotzdem einfach zu installieren. Das geht mit diesen Kommandos (Die Versionsnummern können abweichen):

```
# pkg_add teTeX-share-1.0.2.tgz
# pkg_add teTeX-bin-1.0.7nb1.tgz
```

nach oben

B.7.2. Die Silbentrennung

Den NetBSD- Guide gibt es in SGML derzeit in drei Sprachen: Englisch, französisch und italienisch. Von diesen wird nur in der englischen und der französischen Version die Silbentrennung durchgeführt. Um diese Option für die italienische Sprache einzuschalten, sind ein paar kleine Schritte notwendig:

Öffne in Deinem Lieblingseditor »usr/pkg/share/texmf/generic/config/language.dat« und entferne das Kommentarzeichen »%« aus der Zeile für die italienische Silbentrennung. Beispiel:

```
%italian ithyph.tex
```

wird

```
italian ithyph.tex
```

> **Hinweis:** Je mehr Übersetzungen des es gibt, desto mehr dieser Optionen wirst Du möglicherweise freischalten müssen.

Jetz müssen die latex- und pdflatex- Formate neu erzeugt werden:

```
# cd /usr/pkg/share/texmf/web2c
# fmtutil --byfmt latex
# fmtutil --byfmt pdflatex
```

Wenn Du kontrollierst, z.B. »latex.log« wirst Du etwas finden wie das hier:

```
Babel <v3.6Z> and hyphenation patterns for american, french, german,
ngerman, italian, nohyphenation, loaded.
```

> **Hinweis:** Es gibt viele Wege, diese Operationen durchzuführen; abhängig davon, wie weit Du mit TeX vertraut bist (Ich bin es nicht so sehr...). Du kannst, um mal ein anderes Beispiel zu nennen, auch das interaktive texconfig- Programm benutzen oder die Formate mit »tex« per Hand neu erstellen.
>
> Wenn Du einen besseren Weg für diese Vorgänge als hier beschrieben weisst, lass es mich bitte wissen.

nach oben

B.7.3. Das hugelatex- Format erstellen

Jadetex benötigt das hugelatex- Forma, das nicht in der Standardinstallation von teTeX enthalten ist. Erstelle eine Sicherungskopie von »/usr/pkg/share/texmf/web2c/texmf.cnf« und ergänze mit den folgenden Zeilen am Ende die Datei (Wir brauchen jadetex- und pdfjadetex- Einstellungen wenn wir jadetex später installieren):

```
% 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
```

So kann das hugelatex- Format erstellt werden, wenn man der Jatetex- Installationsanweisung folgt:

```
# 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
```

> **Hinweis:** Wie vorhin auch, gibt es mehr als einen Weg, das hugelatex-Format zu erzeugen. Das obige Verfahren ist im Jadetx- Guide einfach durchgehend beschrieben.
>
> Eine andere Möglichkeit ist es, folgende Zeilen, die das hugelatex- Format beschreiben in »fmutil.cnf« einzufügen (liegt in »/usr/pkg/share/texmf/web2c«):
>
> ```
> # hugelatex format created for jadetex
> ```

```
hugelatex       tex             language.dat    latex.ini
```

Speichere die Datei und führe das hier aus:

```
fmtutil --byfmt hugelatex.
```

nach oben

B.7.4. Installation von Jadetex

Notiz: Jadetex gibt es auf http://www.tug.org/applications/jadetex/

Greife Dir die aktuele Version (derzeit »jadetex-3.6.zip«), packe sie aus und dann:

```
# cd jadetex
# make install
# mktexlsr
```

Wenn Du die jadetex- pdfjadetex-Formate installierst, werden die Formatierungsdateien zusammen mit einigen anderen Utilities in den TeX- Baum kopiert.

Die Jadetex- Distribution beinhaltet zwei Manpages, die nicht automatisch installiert werden. Du kannst sie nur manuell kopieren; Beispiel:

```
# cp jadetex.1 pdfjadetex.1 /usr/local/man/man1
```

Jetzt sind wir soweit, dass wir die Postscript- Version des NetBSD- Guides erzeugen können (Und jedes andere Dokument, das Du willst, versteht sich).

nach oben

B.7. Linkliste

Eine schlichte und gut geschriebene Einführung in SGML/Docbook und eine Beschreibung der Tools findet sich auf SGML comme format de fichier universel

Die offizielle DocBook Homepage ist die Stelle, an der Du die definive Dokumentation von Docbook findest. Du kannst sie online lesen oder ein Kopie von DocBook: The Definitive Guide das von Norman Walsh und Leonard Muellner geschrieben wurde, ziehen.

Für DSSSL startest Du auf nwalsh.com.

Jade/OpenJade- Quellen finden sich auf der OpenJade Home Page.

Wenn Du Postscript- oder PDF- Dokumente aus Deinen Docbook- Quellen herstellen willst, kannst Du einen Blick auf diese Seite werfen: JadeTex.

Die Homemepage von Markus Hoenicka erklärt alles, was Du wissen musst, wenn Du mit DocBook auf der Windows-NT- Plattform arbeiten willst.

nach oben

Inhalt

Appendix C: Danksagungen

Für die Entwicklung dieses Shortguides erhielt ich von einigen Leuten, bei denen ich mich bedanken und die ich weiterempfehlen möchte. Sie haben Material oder Anregungen für den Guide geliefert. Sie haben den Guide überprüft oder halfen beim Aufsetzen der Programme, die für die Produktion nötig waren.

Mein besonderer Dank gilt:

- Manolo De Santis
- Eric Delcamp
- Hubert Feyrer
- Jason R. Fink
- Reinoud Koornstra (Mipam)
- Guillain Seuillot

Falls ich vergessen haben sollte, jemanden zu erwähnen, lasst es mich bitte wissen.

Inhaltsverzeichnis