Festplatten / Blockdevices

Unix gab es urpsrünglich auf anderen Plattformen als Intels x86, daher hatte es auch traditionell *Disk Labels* statt einem Master Boot Record (MBR). NetBSD hat dies beibehalten, und einige Plattformen (z.B. Sparc) benutzen immer noch nativ nur Disk Label und keinen MBR. Es ergeben sich dadurch einige Unterschiede bei der Adressierung von Partitionen.

Partitionierung

Zwar wird oft `gpt(8)` statt eines MBRs/Disk Labels benutzt, da es für große Platten geeignet ist, aber `disklabel(8)` trifft man trotzdem noch oft.

Diese editiert man mit dem Tool `disklabel(5)`. Man kann dabei das Disk Label direkt auf die Festplatte schreiben, wenn nicht davon gebootet werden soll oder die Plattform das unterstützt (z.B. sparc), oder man bettet es in eine native Partition (meist MBR) ein.

In diesem Fall wird zuerst eine MBR-Partition erstellt (*slice* genannt), und in diesem Slice wird ein Disk Label erstellt. Die Disk Label enthalten bis zu 16 Partitionen, mit *a*–*n* durchnummeriert. Dabei haben Partition *c* und ggf. *d* eine Sonderrolle: *c* wird zum Ansprechen des Slices benutzt, in dem das Disk Label liegt, oder bei nicht eingebetteten Disk Labels die gesamte Platte. *d* wird, falls *c* für das Slice benutzt wurde, für die gesamte Platte verwendet.

Raw Devices

NetBSD unterscheidet bei der Adressierung (bzw. der Abstraktionsebene) je nachdem, welche Device Nodes verwendet werden. Jedes Blockdevice (im folgenden sei immer `/dev/sd0` verwendet) ist entweder roh (raw) als Character Device (mit DMA) ansprechbar, dann mit einem *r* davor (`/dev/rsd0`), oder als Block Device normal.

Daher sollte man zum Mounten etc. immer das Block Device benutzen, aber für Operationen unter dem Dateisystem das Raw Device, z.B. zum Nullen einer Platte `dd of=/dev/rsd0d` (nicht `rsd0` oder `sd0d`).

Überwachung/Monitoring

Monitoring-Tools

NetBSD hat einige sehr mächtige Monitoring-Tools per Default mitgeliefert, die Linux so einheitlich nicht hat, bzw. nur mit dem relativ schwer benutzbaren Tool sar.

- `sysstat(1)` – zeigt eine Übersicht über alle möglichen Daten. Man kann auch die einzelnen Subsysteme aussuchen, die man monitoren will.
- `iostat(8)` – zeigt Statistiken über den I/O von Festplatten, Terminals und CPUs an
- `netstat(1)` – zeigt verschiedene Informationen, z.B. von Diensten geöffnete Ports oder die aktuelle Routing-Tabelle an; funktioniert größtenteils wie sein Linux-Äquivalent
- `vmstat(1)` – zeigt Informationen zur Speicherbelegung an
- `fstat(1)` und `sockstat(1)` – äquivalent zu lsof zeigt fstat Informationen über offene Dateien an, sockstat hingegen über offene Sockets.

Man kann zwar alle Tools auch ohne Parameter aufrufen, aber ein Blick in die Manpage gibt einem deutlich mehr Optionen, das Monitoring anzupassen.

Monitoring-Mails

Die Standard-Crontab von NetBSD hat drei Einträge: `daily`, `weekly` und `monthly`, die jeweils nachts ausgeführt werden.

Sie enthalten Informationen über Speicherplatzverbrauch, Systemmeldungen, Sicherheitslücken, Loginzeiten, etc. Man kann über ihre Konfigurationsdateien `daily.conf(5)` bzw. `weekly.conf(5)` und `monthly.conf(5)` konfigurieren, was für Checks sie machen, was sie anzeigen, wie sie den Output mailen.

Man sollte diese Mails wenigstens überfliegen, sonst nicht monitort, da wichtige Informationen wie bspw. Plattenausfälle im Raid oder Sicherheitslücken hierüber gemeldet werden. V.a. bei Rechnern mit Internetanbindung sollte man sie anpassen, da sonst keine Sicherheitslücken gemeldet werden.

NetBSD für Umsteiger

Heutzutage gibt es nur noch wenige, die mit Unix auf der Shell anfangen. Die meisten installieren ihr Betriebssystem grafisch, und in den allermeisten Fällen ein GNU/Linux, das wohl am weitesten verbreitete Unix.

Diese Einleitung ist für Leute gedacht, die schon Linux benutzen, und die wissen, wie es funktioniert und auf welchen Prinzipien es basiert, und beschreibt die Unterschiede denen man in der täglichen Benutzung begegnen wird.

Im folgenden wird der Begriff *Linux* äquivalent zum Begriff *GNU/Linux* benutzt.

Startskripte, init

Einer der großen historischen Unterschiede zwischen BSD und dem Rest der Unixwelt war das Init-System init(8), das erste vom Kernel geladene Programm. Dieses liest seine aus der Datei `rc.conf(5)` seine Konfiguration, die im wesentlichen aus Optionen und (De-)Aktivierungen von Diensten besteht (üblicherweise als `dienst=YES`, z.B. `sshd=YES`).

In `/etc/rc.d` liegen die Startskripte (wie Linux' `/etc/init.d`). Aus denen baut sich init eine Abhängigkeitsliste zusammen und eine Liste, in welcher Reihenfolge die aktivierten Dienste dann gestartet werden.

Das Starten/Stoppen von Diensten läuft wie unter Linux, nur mit anderem Pfad. Ein `/etc/rc.d/sshd start` startet den sshd, analog gibt es noch mindestens die Optionen `start`, `restart`, `stop`, `status`.

Wenn ein Dienst nicht in der rc.conf aktiviert ist, dann muss man ein `one` vor den Parameter hängen, also wenn man kein `sshd=YES` oder sogar ein `sshd=NO` hat, den sshd aber starten will, dann macht man das mit `/etc/rc.d/sshd onestart`.

Netzwerk

Das Netzwerk von NetBSD wird auch in der `rc.conf` mit dem Schlüsselwort `ifconfig_IFNAME` konfiguriert. `defaultroute` setzt das Default-Gateway, `hostname` den Hostnamen des Systems. Zusätzliche IP-Adressen können mit `ifconfig_IFNAME_alias` konfiguriert werden, genaueres steht in der Manpage rc.conf(5).

Manuell konfiguriert man das Netzwerk mit `ifconfig(8)`, dessen Unterschiede zu dem von Linux gering sind. Zusätzlich übernimmt es noch die Funktion von Linux' iwconfig mit einigen zusätzlichen Optionen.

Die Routen werden wie unter Linux mit `route(8)` konfiguriert, welches sich aber stark anders als die Linux-Variante bedient. Einiges kann bzw. muss man auch mit dem Tool `netstat(1)` erledigen, das sich größtenteils wie sein Linux-Äquivalent verhält.

Das Paketmanagement

Zusatzsoftware kann man bei NetBSD entweder als Binärpakete, die man sich aus einem Repository holt, oder als selbstkompilierte Software verwalten. Beide Wege sind gleichwertig, die Binärpakete sind genau so kompiliert, wie man es mit pkgsrc selber täte.

Zusätzlich zu diesem Text ist ein kurzer Blick in das entsprechenden Kapitel im *NetBSD Guide* zu empfehlen, und es gibt sogar einen *pkgsrc Guide*, der nochmal auf die Einzelheiten des Paketmanagements eingeht.

pkgsrc

Das System zur Verwaltung von Quellpaketen unter NetBSD heißt `pkgsrc`. Es stammt von FreeBSDs *ports* ab, aber hat sich dann davon gelöst und stellt heute ein plattform- und architekturübergreifendes Paketmanagement dar, und kann sogar als unprivilegierter User benutzt werden.

Normalerweise liegen die pkgsrc-Quellen unter `/usr/pkgsrc`, installiert werden die Pakete nach `/usr/pkg` (beides kann angepasst werden). Im Installationspfad hat man wieder eine Struktur ähnlich zu der in `/`: `bin/`, `sbin/`, `etc/`, usw. pkgsrc installiert keine Pakete außerhalb seines Installationspfades. Das heißt, dass jegliche zusätzliche Software auch in `/usr/pkg/etc` konfiguriert wird. Dort gibt es auch wieder ein Verzeichnis `rc.d`, welches die Startskripte für zusätzliche Software enthält. Man muss aber darauf achten, dass man den Start weiterhin auch *nur* in der rc.conf in /etc konfiguriert.

pkgsrc ist unterteilt in Kategorien wie *www*, *chat*, *mail*, usw., und in diesen Kategorien liegen dann die jeweiligen Pakete. Um ein Paket zu installieren, ruft man aus dem entsprechenden Verzeichnis (z.B. `misc/bsdstats`) ein `make install` auf, welches das Paket kompiliert und danach als Binärpaket installiert.

Um nach einem Paket zu suchen, empfiehlt sich entweder `find(1)` (`cd /usr/pkgsrc; find . -maxdepth 2 -name PKGNAME`, wenn man nur den Paketnamen sucht, oder ein einfaches `grep(1)`.

Jedes Paketverzeichnis enthält die Datei `DESCR`, die eine Beschreibung enthält, eine Volltextsuche wäre dann z.B. mit `cd /usr/pkgsrc; grep STRING */*/DESCR` möglich.

Nach jedem Installieren sollte man noch die Überreste vom Kompilieren beseitigen, was man mit `make clean` erledigt (bzw. primitiver `rm -r work/`). Falls man es mal vergessen haben sollte, kann man dies auch vom pkgsrc-Verzeichnis oder nur in einer Kategorie aufrufen, alle untergeordneten Verzeichnisse werden dann gesäubert.

Binärpakete

Die eigentliche Paketverwaltung (pkgsrc ist quasi nur Paketquelle) besteht aus den Tools `pkg_info(1)`, `pkg_add(1)`, `pkg_admin(1)`, `pkg_delete(1)` und `pkg_create(1)`.

pkg_add und pkg_delete (de-)installieren die als Parameter übergebenen Pakete, wobei man bei pkg_delete darauf achten muss, dass man immer auch die Version dazu angibt. pkg_info listet alle Pakete auf bzw. gibt einem Informationen zum speziellen Paket aus (z.B. enthaltene Dateien, Beschreibung, Version, etc.). pkg_admin übernimmt allgemeinere Aufgaben zur Administration von Paketen und der Paketdatenbank (diese liegt in `/var/db/pkg`) oder dem Checken von Sicherheitslücken.

mk.conf

pkgsrc wird über die Datei `mk.conf(5)` konfiguriert (diese existiert per default nicht, sie muss angelegt werden). pkgsrc selber zeigt einem bei der Installation oder mit `make show-options` an, welche Optionen man für es setzen kann.

Andere Tools

Es gibt zahlreiche weitere Tools, die einem das Paketmanagement erleichtern. Man kann sich einfach die pkgsrc-Kategorie `pkgtools` angucken, diese enthält einige Tools, die einem dies erleichtern, vor allem `nih`, `pkg_chk` und `pkgin` seien hier erwähnt.