BSD Mythen

Es gibt viele moderne Mythen und Vorurteile um BSD. Hier einige der am häufigsten anzutreffenden:

Hardware

„BSD unterstützt keine normale Hardware.“ Unterstützt Linux Hardware, die BSD nicht unterstützt? Macht es was aus? - Nur wenn Sie diese Hardware besitzen. Unterstützt Windows Hardware, die Linux nicht unterstützt? BSD unterstützt fast alle Hardware, die in einem Server, einer Workstation oder einem Desktop steckt. Es gibt Lücken, die aber mit jedem Release kleiner werden. Zu allen BSDs gibt es eine Liste der unterstützten Hardware, die Sie vor der Installation konsultieren können. Oder Sie testen die Kompatibilität mit einer LiveCD wie FreeSBIE, OliveBSD, NewBIE etc.

Verfügbare Software

„Aber für Linux gibt es mehr Software!“ Ganz einfach falsch. Wenn ein Programm einigermassen portabel geschrieben ist, so wird es auf 98% aller POSIX-konformen System kompiliert werden können. FreeBSD und Debian haben die grössten Sammlungen an installierbarer Open Source Software. Alle BSDs haben eine Linux Emulationsschicht (ABI), die binärkompatibel mit Linux ist. Dies funktioniert meist hervorragend.

Verbreitung

„Aber Linux ist beliebter!“ Und? Windows ist noch beliebter und verbreiteter als Linux. Dieses Argument zielt vorrangig darauf ab, dass es bei Linux einfacher sei Unterstützung bei Problemen zu finden. Es gibt aber genug Firmen, die kommerziellen Support für BSD anbieten. Zudem ist die BSD-Gemeinschaft in Foren und Mailinglisten allgemein auf hohem technischem Niveau und kann in fast allen Fragen weiterhelfen.

Einfachheit

„BSD ist schwer, für Fortgeschrittene, komplizierter und weniger benutzerfreundlich.“ Es ist *anders* als Windows oder Linux, na und? In den meisten Fällen finden es die Nutzer *viel* logischer und darum einfacher. Hier kommt die Konsistenz und konzeptionelle Klarheit voll zum Zug. Die BSDs sind nicht schwerer, sie sind anders.

Elitismus

„BSD User sind elitäre, selbstherrliche und rüde Snobs.“ BSD-User sind ganz normale Menschen, wie alle anderen auch. Sie lieben Unix, BSD und Beastie.

Wieso sollte ich BSD versuchen?

Soll ich BSD oder Linux benutzen? Eine Frage, die nur schwer zu beantworten ist. Hier ein paar Richtlinien:

1. Es funktioniert einfach. Sobald das System steht, kann man loslegen.

2 „Wenn es nicht kaputt ist, soll man es nicht flicken.“ Wenn Sie schon ein Open Source OS benutzen, und Sie damit glücklich sind, so gibt es wahrscheinlich keinen Grund, wieso Sie wechseln sollten.

3. BSD Systeme sind merklich schneller, aber nicht immer. Oft gibt es keinen Unterschied. In manchen Fällen könnte Linux sogar schneller sein.

4. Die BSDs haben einen besseren Ruf bezüglich Zuverlässigkeit wegen der reiferen Code-Basis.

5. Die BSD-Lizenz mit weniger Einschränkungen könnte attraktiver sein, als die GPL.

6. BSD kann die meisten Linux Programme ausführen, aber Linux nicht die BSD Programme.

Wichtige Websites und Links:

Alle BSDs sind hervorragend dokumentiert:

http://www.FreeBSD.org/

http://www.NetBSD.org/

http://www.OpenBSD.org/

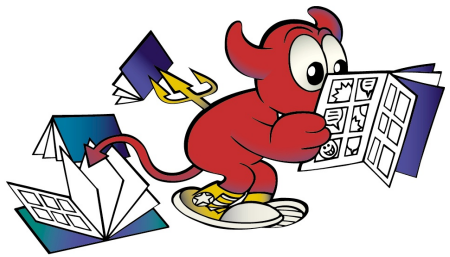
http://www.DragonFlyBSD.org/

http://www.over-yonder.net/~fullermd/rants/bsd4linux/bsd4linux1.php

http://www.freebsd.org/doc/en_US.ISO8859-1/articles/linux-comparison/

http://sites.inka.de/mips/unix/bsdlinux.html

http://www.freebsd.org/doc/en_US.ISO8859-1/articles/explaining-bsd/index.html



BSD vs. Linux

Ein kurzer Vergleich

Eine häufig gestellte Frage ist die, wie sich BSD von Linux unterscheidet. Wir wollen diese Frage hier kurz beantworten. Wir geben zu, dass wir BSD vorziehen, aber das ist nur aus persönlicher Erfahrung. Wir werden nicht allzu sehr ins Detail gehen, wir wollen Gemeinsamkeiten und Unterschiede erklären. Dieser Flyer basiert mehrheitlich auf der Arbeit von Matthew D. Fuller, Greg Lehey und Dru Lavigne. Siehe dazu die Links am Ende.

Was ist Unix?

Unix ist ein Betriebssystem, das in den späten 60er Jahren in den Bell Labs von Ken Thompson und Dennis Ritchie entwickelt wurde. Vielleicht ist Unix das Betriebssystem, das die moderne EDV am meisten beeinflusst hat. Jedes Multifunktionsgerät zur Datenverarbeitung, das Sie finden, und auch viele spezialisierte Geräte bedienen sich der Ideen, Konzepte und sogar des Codes im Unix Stammbaum. Wenn wir von Unix sprechen, meinen wir viel mehr das generelle Konzept, als das bestimmte Betriebssystem, das Unix(TM) heisst. Das generelle Konzept soll heissen, „jedes Betriebssystem, das im Aufbau, in der Ausführung, an den Schnittstellen und vom allgemeinen Gefühl her sehr einem Unix System ähnelt“. Das bedeutet alle BSDs, Linux, Solaris, AIX, HP/UX und mehrere Dutzend andere.

Was ist Linux?

Linux bedeutet auch mehrere Dinge. Zum einen den Kernel, ursprünglich von Linus Torvalds geschrieben. Linux bezeichnet auch eine Familie von Betriebs-systemen. Wenn wir „Linux“ sagen, meinen wir Red Hat, Slackware, Debian, Gentoo und hunderte andere Distributionen um den Linux-Kernel herum, mit sehr ähnlichem Userland; die meist auf GNU Tools basieren.

Was ist BSD?

BSD steht für „Berkeley Software Distribution“. Es wurde an der University of California Berkeley (CSRG) entwickelt und der Code stand immer unter der BSD-Lizenz. Die Lizenz besagt mehr oder weniger „Tun Sie damit, was immer Sie wollen. Zollen Sie uns einfach Anerkennung dafür.“ Später startete das 386BSD-Projekt und ermöglichte es auf der Intel i386-Plattform zu laufen. Als das 386BSD-Projekt langsam zu Ende ging, formierten sich zwei Hauptgruppen: FreeBSD und NetBSD. 1995 spaltete sich OpenBSD von NetBSD und 2003 DragonFlyBSD von FreeBSD ab.

Wenn wir „BSD“ sagen, so meinen wir den generellen BSD-Charakter. Es gibt 4 grosse freie BSDs:

- FreeBSD läuft auf vielen Plattformen, aber der Fokus liegt darauf so robust und effizient wie möglich für Server und Desktopsysteme auf Intel und AMD zu sein.
- NetBSD hat das Ziel auf so vielen Plattformen wie möglich zu laufen. Es soll das portabelste OS auf diesem Planeten sein.
- OpenBSD fokussiert auf Security und verwandte Themen. Die nahtlose Integration von Security, Auditability und Kryptographie sind das primäres Ziel.
- DragonFly BSD legt den Schwerpunkt auf eine SMP-fähige Infrastruktur, die einfach zu verstehen, warten und entwickeln ist.

All diese Ziele sind unter den BSDs austauschbar. *Jedes* BSD kümmert sich um Security. *Jedes* BSD kümmert sich um Performanz. Grosse Teile des Sourcecodes werden untereinander ausgetauscht. Viele Entwickler arbeiten an mehreren Systemen. Je genereller der Lösungsansatz, desto eher wird es für alle BSDs der gleiche sein. Philosophisch sind sich alle BSDs sehr ähnlich, im Gegensatz zur Linux Methodologie. Und um genau diese Philosophie geht es in diesem Flyer.

Die Systembasis

Das Konzept einer „Systembasis“ kann eventuell schwer zu verstehen sein, denn dieses Konzept gibt es in der Linux-Welt nicht. Linux war von Anfang an immer nur ein Kernel, und ein Kernel allein ist nicht sehr nützlich. Man braucht alle Userland-Programme, um ihn zu nutzen. Linux war schon immer ein Konglomerat; ein Kernel von hier, ein 'ls' von dort, vim, Perl, gzip, tar und alle anderen auch. Linux unterschied nie zwischen einer Systembasis und den Userland-Programmen. Das *ganze System* besteht aus Userland-Programmen.

Ganz im Gegensatz dazu hatte BSD *immer* schon ein zentralisiertes Entwicklungsmodell. BSD benutzt kein GNU ls oder GNU libc, es hat ein BSD ls und ein BSD libc, die direkte Nachfahren der BSD-Releases der CSRG sind. Das Sytem als Ganzes ist aus einem Guss, nicht eine Ansammlung kleiner Stücke. X ist nicht Teil der FreeBSD Systembasis. Es ist ein Zusatzpaket, wie xterm, KDE, GNOME, Mozilla etc., das natürlich nicht Teil der Systembasis ist. Der primäre Unterschied liegt darin, wo sie entwickelt werden. NetBSD und OpenBSD *haben* eine X-Implementation in der Basis, weil sie es mit den Konsolen-Treibern integrieren. Sie benutzen beide stark modifizierte Versionen.

Die ganze Systembasis wird zusammen entwickelt. Teile der Systembasis wie sendmail, BIND, ssh und weitere werden aber ausserhalb entwickelt. Es gibt sogar ein paar GNU Pakete, wie GCC, die jedem Linux-Nutzer bekannt sein sollten. Diese werden aber speziell behandelt, insofern, dass Versionen in den Source-Baum importiert und anschliessen ins System eingepasst werden. Es sind früher einige sogar nur für BSDs verfügbar gewesen. Sendmail und BIND waren ursprünglich Bestandteile von BSD und wurden erst später separat verfügbar.

Der Hauptgrund, wieso ein ursprünglich extern entwickeltes Paket importiert wird, ist der, dass es auf seine Weise genug wichtig ist, dass es einfacher ist, es als integralen Bestandteil der Systembasis zu betrachten. GCC und die binutils sind solche Pakete, weil sie unter anderem zum Bau der Systembasis an sich benötigt werden. GNOME, KDE etc. sind *nicht* Bestandteile der Basis, und werden es ziemlich sicher nie werden, weil sie nicht zum Betrieb des Systems an sich benötigt werden.

Die Basis muss nur die Werkzeuge bereitstellen, um das System zu betreiben, zu aktualisieren und zusätzliche Pakete zu installieren. Dann erst installieren Sie, was Sie für ihre Aufgaben benötigen. Jedes BSD hat seine eigene Basis, NetBSD und OpenBSD zum Beispiel, haben viel breitere Kriterien, was sie in ihre Basis aufnehmen (in der OpenBSD-Basis ist Apache).

Ports/pkgsrc versus rpm/packages

Dann gibt es noch die zweite Kategorie, die Programme, die Zusatzpakete sind. In der BSD-Welt sind das meist sogenannte „Ports“ (FreeBSD und OpenBSD) oder „pkgsrc“ (NetBSD und DragonFlyBSD).

Traditionell war es so, dass man ein Paket kompilieren musste, wenn man es installieren wollte. Und bevor man es kompilieren konnte, musste man meistens noch daran herumbasteln. Das System wollte seine eigenen Header. Manchmal musste man sogar Pfade umschreiben, damit das Programm an den richtigen Ort kam. Man musste es also „portieren“. Das „Portssystem“ übernimmt diese Aufgabe für Sie. Es automatisiert den Bau, die Installation und die Erstellung von Paketen für Sie.

Die meisten Linux-User installieren Binärpakete und die meisten BSD-User kompilieren die Pakete selbst. Dies hat zum Teil mit den Werkzeugen zu tun; das Portssytem ist um das Konzept der Kompilation von Quelltexten herum entstanden, mit der Fähigkeit Binärpakete zu bauen und zu installieren eher als Nebengedanken. Währenddessen sind die Linux-Werkzeuge (z.B. rpm und apt) um das Konzept der Binärpakete herum entstanden. Die Kompilation von Quelltexten war eher ein Nebengedanke dabei. Es gibt aber für beide Methoden Vor- und Nachteile. Binärpakete sind schneller zu installieren und benötigen weniger Platz, da die Kompilation entfällt. Dafür hat die Kompilation aus Quelltexten den Vorteil, dass nicht so viele verschiedene Bibliotheken nötig sind und die Software genau auf den Verwendungszweck zugeschnitten werden kann. Man kann Binärdateien auf BSD, wie auch Linux installieren, man kann aber auch auf beiden selbst kompilieren.

Es kann natürlich auch was schief gehen. Vielleicht verschwindet eine Abhängigkeit von einem Server, so dass niemand den Quelltext herunterladen kann. Vielleicht beschädigt ein drittes Programm ein bestehendes, von dem wiederum andere abhängig sind. Das Portssystem löst nicht *alle* Probleme. Aber das Problem von „Ich will A, das ist von B abhängig, das ich aber nicht finde, weil...“ wird fast nie so oft anzutreffen sein wie bei dezentralisierten Systemen wie rpm.