

Das OpenBSD Projekt

Alexander von Gernler

<grunk@openbsd.org>



4. Erlanger Linuxtage, 1./2. April 2006

Inhaltsübersicht

1 Einleitung

- Über den Vortrag
- Einordnung von OpenBSD

2 Das Projekt

- Geschichte
- Struktur
- Ziele
- Spin-Offs aus OpenBSD

3 Das Produkt

- Technische Daten
- Unterschiede zu Linux
- Safety und Security
- Letzte Neuigkeiten

4 Bemerkungen

- Ausblicke
- Literatur
- About

Inhaltsübersicht

1 Einleitung

- Über den Vortrag
- Einordnung von OpenBSD

2 Das Projekt

- Geschichte
- Struktur
- Ziele
- Spin-Offs aus OpenBSD

3 Das Produkt

- Technische Daten
- Unterschiede zu Linux
- Safety und Security
- Letzte Neuigkeiten

4 Bemerkungen

- Ausblicke
- Literatur
- About

Über den Vortrag

- **Dieser Vortrag ist...**

- eine Vorstellung des OpenBSD-*Projektes*
- eine Vorstellung des OpenBSD-*Betriebssystems*
- ein subjektiver Erfahrungsbericht

- **Er ist nicht...**

- ein Installations-HOWTO für OpenBSD
- eine technische Referenz
- ein heiliger Kreuzzug

- **Der Autor...**

- ist Mitglied im OpenBSD-Projekt
- benutzt OpenBSD seit Version 2.9
- betreibt den OpenBSD Mirror an der Uni Erlangen



Läuft da auch alles, was ich von Linux kenne?

- Beide implementieren POSIX-Standards und sind damit unixoide Betriebssysteme.
- Quellcode, der auf Linux kompiliert, baut auch unter BSD, zur Not mit kleinen Anpassungen
- Komplette freie Softwareauswahl steht gewohnt zur Verfügung
 - Mozilla, MPlayer, XMMS, KDE, Apache, MySQL, Gimp, ...

Wieviele Distributionen gibt es denn?

- Es gibt keine Distributionen!
- Drei große Projekte (sog. Flavors)
 - FreeBSD
 - OpenBSD
 - NetBSD
- Jedes Projekt stellt exakt eine Distribution bereit
- Trennung von Kernel und Userland wenig sinnvoll, trotz Versuchen wie Debian mit BSD-Kern o. ä.
- Kein Distributionen-Wildwuchs wie bei Linux (dreistellige Zahl, so genau weiss das niemand)

Die BSD Flavors

• FreeBSD

- <http://www.FreeBSD.org/>
- Stabilität, Performance, Vielzahl von Anwendungen



• NetBSD

- <http://www.NetBSD.org/>
- Portabilität – läuft auf über 55 Plattformen



• OpenBSD

- <http://www.OpenBSD.org/>
- Sicherheit, Kryptographie, korrekte Implementation



Inhaltsübersicht

- 1 Einleitung
 - Über den Vortrag
 - Einordnung von OpenBSD
- 2 **Das Projekt**
 - Geschichte
 - Struktur
 - Ziele
 - Spin-Offs aus OpenBSD
- 3 Das Produkt
 - Technische Daten
 - Unterschiede zu Linux
 - Safety und Security
 - Letzte Neuigkeiten
- 4 Bemerkungen
 - Ausblicke
 - Literatur
 - About

Geschichte von OpenBSD

1969 Entwicklung des Ur-UNIX an den Bell Labs durch KEN THOMPSON und DENNIS M. RITCHIE



1974 Weitergabe des Quellcodes an Universitäten, darunter auch die UC Berkeley (**B**erkeley **S**oftware **D**istribution)

1992 Rechtsstreit AT&T ./ UC Berkeley et al., Entwicklung von 386BSD ohne strittigen Code

1993 Erscheinen von NetBSD 0.8 und FreeBSD 1.0

1994 Streit im NetBSD Core-Team

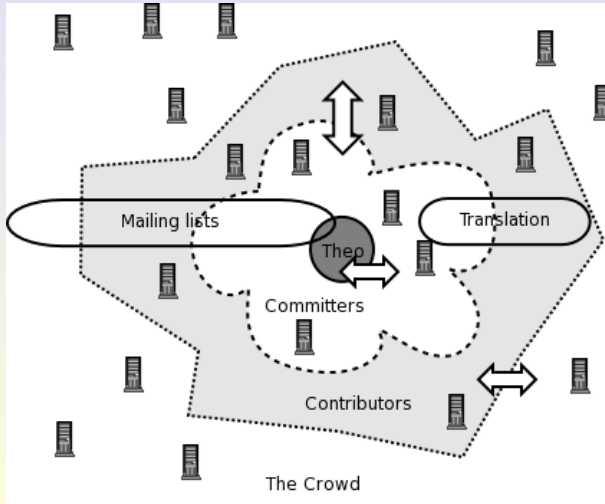
1995 THEO DE RAADT wird aus dem NetBSD Team ausgeschlossen und gründet OpenBSD als abgespaltenes Projekt



Strukturen im Projekt

- Project Leader** THEO DE RAADT als freundlicher Diktator
- Committers** Team der Entwickler mit CVS-Schreibzugriff
- Contributors** Jeder, der vernünftigen Code mit freier Lizenz einreicht, kann beitragen
- Mirrors** Server, von denen OpenBSD gezogen werden kann
- Hackathon** Jährliches Treffen der Entwickler in Calgary

Organigramm



Entwicklungsmodus

- Quellcodebaum per CVS versionsverwaltet
- Kommunikation über internen Chat und interne Mailingliste
- Kein Commit ohne OK von anderen Entwicklern (wenige Ausnahmen)
- Letztes Wort hat Theo de Raadt
- Fehlverhalten wird mit Accountsperrung belohnt
- Demokratie ist prinzipiell toll, funktioniert aber nicht bei Softwareprojekten: Machen statt reden
- Benutzeranfragen werden meist mit "Submit a diff" beantwortet

Philosophie von OpenBSD

- Absolut freie Lizenz erlaubt Verwendung des Quellcodes für *jeden* Zweck
- Oberste Priorität auf Sicherheit
- Integration starker Kryptographie mit der Möglichkeit, sie überall hin zu exportieren
- Die beste Entwicklungsplattform überhaupt bereitstellen
- Standards verfolgen und korrekt implementieren (POSIX, ANSI, X/OPEN etc.)
- Zielgruppe: Eigene Entwickler
- Absolut konservative Entwicklungsstrategie: Features lieber nicht als schlecht implementieren

Lizenzdiskussion: GNU GPL vs. BSD-style

• GNU GPL

- Will größtmögliche Freiheit für die **Community**
- Weitergabe der Software nur mit Weitergabe des Quellcodes, dies muss ebenfalls unter GPL erfolgen
- Hält die Quellen rechtlich gesehen für immer offen
- Aber: Prozess gegen Router-Hersteller zeigt, dass GPL auch von Firmen absichtlich missachtet wird

• BSD-style

- Will größtmögliche Freiheit für den **Einzelnen**
- Weitergabe der Software unter jeder beliebigen Lizenz, auch Closed Source möglich
- Wichtig nur: Wahrung des Copyrights
- Große Verbreitung von BSD Code bis zu Windows (TCP Stack)

Mit OpenBSD hab ich selber gar nichts zu tun!

Oh doch! Jeder von uns benutzt täglich OpenBSD:

- OpenSSH** Freie Implementation der Secure Shell von `ssh.com`. Einsatz auf praktisch allen Linux-Distributionen und den anderen BSDs
- OpenBGPD** Freie Implementation des Border Gateway Protokolles. Einsatz bei vielen Internet Service Providern und grossen Internet-Exchanges
- OpenNTPD** Freie Implementation des Network Time Protokolls. Schlanker und stabiler als der alte XTNPd.
- OpenCVS** Momentan laufende Anstrengung, einen sicheren und freien Ersatz für das uralte und schlecht gewartete Versionsmanagement-System zu schreiben

Inhaltsübersicht

- 1 Einleitung
 - Über den Vortrag
 - Einordnung von OpenBSD
- 2 Das Projekt
 - Geschichte
 - Struktur
 - Ziele
 - Spin-Offs aus OpenBSD
- 3 **Das Produkt**
 - Technische Daten
 - Unterschiede zu Linux
 - Safety und Security
 - Letzte Neuigkeiten
- 4 Bemerkungen
 - Ausblicke
 - Literatur
 - About

Technische Daten

- Freies Open Source UNIX, basierend auf den 4.4BSD Quellen
- Eigenes Binärformat, aber Emulation für Linux, FreeBSD, Solaris (SVR4), BSD/OS, SunOS und HP/UX
- Plattformen: i386, amd64, macppc, sparc, sparc64, zaurus, alpha, vax, mac68k, sgi, hp300, hppa, mvme68k, mvme88k, cats, luna88k
- Standardmäßig IPv6- und IPsec-fähig
- Minimalinstallation ca. 100 MB, Standardinstallation bei ca. 300 MB, Desktop-System des Autors bei 1.6 GB
- Software installierbar als Binärpakete oder aus Ports Collection. Derzeit über 3402 Ports

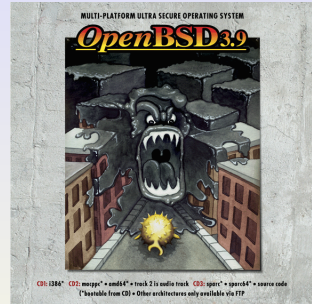


Wo kann man OpenBSD einsetzen?

- Sicherheitskritische Anwendungen
 - Internet Service Provider
 - Firmennetzwerke
- Firewalls
 - Selbst aufgesetzte Firewall
 - auf OpenBSD basierende Produkte (z.B. GeNUGate)
- Sichere Workstations
 - Installation ist *secure by default*

OpenBSD beziehen

- Auslieferung auf 3 CDs jedes halbe Jahr (feste Releasezyklen im Mai und November)
 - Schickes Artwork mit Aufklebern
 - Neuer OpenBSD Release Song
 - Kostenpunkt ca. EUR 45,-
 - Erhältlich von openbsd.org oder über Buchhandel
- Per Diskette, CD oder PXE bootbar, Installation der Pakete aus dem Netz kostenlos
- Großes Netz an tagesaktuellen Mirrors weltweit verfügbar
- Bereitstellung von Security-Patches und Bekanntgabe auf security-announce@openbsd.org



Technische Gemeinsamkeiten

“BSD = Linux with a twist” (CHRISTIAN WEISGERBER)

- Oberflächlich: Einschalten, Bootloader, Kernel, Init, Textkonsole mit Login, X11 möglich
- Alle Standardbefehle vorhanden (`ps(1)`, `top(1)`, `ifconfig(8)`, `ping(8)`, `vi(1)`, ...)
- Wichtige Internet-Daemons im Basissystem: `httpd(8)`, `sshd(8)`, `ftpd(8)`, `named(8)`, `sendmail(8)`, `ntpd(8)`, `identd(8)`, `nfsd(8)`, `inetd(8)`
- Breite Hardwareunterstützung, auch Cryptohardware (Beschleuniger, z. B. `hifn`)
- Gängige Software als Ports vorhanden: MySQL, PostgreSQL, Gimp, `xmms`, MPlayer, Mozilla, KDE, GnuPG, `mutt`

Technische Unterschiede

- BSD Fast Filesystem `ffs` statt Linux `ext2/ext3`
- Erweiterung *SoftUpdates* sorgt für Geschwindigkeit und Konsistenz bei Absturz, ist aber nicht mit Journaling (vgl. `ext3`) gleichzusetzen
- Die meisten Linux Distros haben ein System `V init` mit Runlevels, die BSDs haben das einfachere BSD-style `init`
- Kernel und Userland bilden feste Einheit und können nur **gemeinsam** aktualisiert werden. Sie passen deshalb immer zusammen. Keine Probleme wie:
 - `lvmtools` passen nicht zum Kernel LVM
 - `iptables` will andere Netfilter Version im Kern
 - `reiserfsck` hat mit dem alten Kern doch noch getan?!

Andere Kernel-Philosophie

- Kernelmodule werden praktisch nicht benutzt
 - Kein einschleusen von Code ins laufende System
 - Probleme, immer passendes Modul zu laden (muss ohnehin auf gleicher Codebasis kompiliert werden) entfallen
 - Stabiler, sicherer
- Kernel customizen ist No-No: **GENERIC** erfüllt alle Wünsche
 - Kein Gewinn durch Weglassen von Komponenten (weder Geschwindigkeit, noch Platz) auf heutigen Maschinen
 - Keine Fehler, die nur durch falsch konfigurierte Kernel entstehen
 - Entwickler leisten nur Support für **GENERIC**
- Compilerflags und andere Geschwindigkeitshacks werden extrem abgelehnt
- Aufgaben von `/proc` und `/kern` werden von `sysctl(1)` übernommen

Keep it simple, stupid!

- nur ein Runlevel, dazu Reboot, Single-User und Halt
- exzellente und aktuelle Manpages
- motivierende Starthilfen: `man afterboot`, `man intro`
- Immer zwei Versionen von OpenBSD verfügbar:
 - `stable` Stand der halbjährlichen Release-Version mit aktuellen Sicherheitspatches. Geeignet für den Produktionsbetrieb
 - `current` Front der aktuellen Entwicklung, ändert sich permanent und ist kaum weniger stabil. Geeignet für neugierige und enthusiastische Anwender (z. B. der Autor), sowie Entwickler

Software besorgen – woher?

- **Von OpenBSD aus verfügbar**

- Installieren als Binärpaket vom Mirror, oder
- selber bauen aus der Ports Collection:

```
# cd /usr/ports/editors/vim/stable
# make install
```

- **Nicht in Ports-Collection**, aber Quellcode verfügbar

- Entweder gute Software, dann:

```
# ./configure && make && make install
```
- Oder halt per Hand kompilieren nach README

- **Nicht im Quellcode verfügbar**, aber als Linux-Binary der selben Plattform, z. B. für i386: Acrobat Reader

- Binäremulation einschalten, läuft:

```
# sysctl kern.emul.linux=1
```

Warum ist OpenBSD so sicher?

- **Passwörter** dürfen **nie auf Platte** landen: Swapspace verschlüsselbar
- **Kryptographie** braucht **guten Zufall**: OpenBSD nutzt viele Entropiequellen (Interrupts von Maus, Tastatur, Netzwerk, Festplatte)
- **Systemverhalten** für Angreifer **unvorhersagbar** zu **machen**: erzeugte Prozeß-IDs, IP Datagramm IDs, DNS Query-IDs, Inode-Nummern
- **Exploits verhindern**: ProPolice Stack Protection erschwert Buffer Overflows durch setzen eines Canarys (GCC Erweiterung).
- **Keine Daten** als Code **ausführen lassen**: W^X (Write XOR Execute) für Speicherseiten bzw. Segmente verhindert weitere Gemeinheiten

Sicherheit – still more

- Firewalling unter OpenBSD mit **pf(4)**
 - Mächtiger regelbasierter Paketfilter, *stateful filtering*
 - Kann Traffic normalisieren (z. B. Verschleierung von NAT mit `scrub reassemble tcp`)
 - Kann Entscheidungen aufgrund des OS der Gegenstelle treffen (*Passive OS Fingerprinting*)
 - `altq(9)` erlaubt regelbasierte Bandbreitenbegrenzung
 - CARP (Common Address Redundancy Protocol) erlaubt HA-Firewalling: Zustände auf der aktiven Firewall werden auf die Backup-Firewall übertragen
- *Secure by default* als Motto: Installation ist standardmäßig schlank und sicher. Trägt der Tatsache Rechnung, dass Rechner, die mal laufen, nicht mehr berührt werden
- Unterstützung von Kryptohardware (Hifn, Broadcom, Intel, 3com) kinderleicht: Einstecken, geht.

Sicherheit – BSD Securelevels

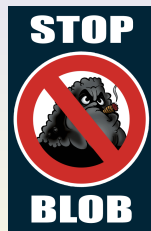
- `kern.securelevel=1`
 - Securelevel nur noch durch `shutdown(8)` gesenkt werden.
 - `/dev/mem`, `/dev/kmem` nicht mehr schreibbar
 - Gemountete Partitionen können nicht roh geschrieben werden
 - Dateiflags wie `immutable` oder `append-only` greifen und können nicht entfernt werden
 - Kernelmodule können weder geladen, noch entfernt werden
- `kern.securelevel=2`
 - Alle Beschränkungen von Level 1 gelten
 - Raw Disks dürfen *überhaupt nicht mehr* geschrieben werden
 - Die Systemzeit kann nicht mehr zurückgestellt werden
 - Paketfilter- und NAT-Regeln können nicht mehr verändert werden
 - Der Kerneldebugger kann nicht mehr eingeschaltet werden

Sicherheit – praktische Paranoia

- `authpf(8)` erlaubt Benutzern im lokalen Netz, die Firewall nach Authentifikation durch `ssh(1)` zu benutzen
- `systrace(1)` erlaubt Aufstellen und Durchsetzen von Syscall-Profilen für einzelne Programme
- Die meisten Dienste in OpenBSD
 - laufen in `chroot(8)`
 - führen *privilege revocation* durch: `ping`, `ping6`, `portmap`, `rpc.rstatd`, `rpc.rusersd`, `pppoe`, `traceroute`, `traceroute6`, `rwalld`, `pppd`, `spamd`, `afsd`, `authpf`, `ftpd`, `tftpd`, `httpd`, `dhcpcd`, `mopd`, `rbootd`
 - sind *privilege separated*: `sshd`, `syslogd`, `pflogd`, `isakmpd`, `bgpd`, `ntpd`, `tcpdump`, `named`, X server, `xdm`, `dhclient`
 - Nur ganz wenige Programme sind `setuid root`.

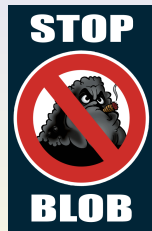
Offene Rechnungen

- Thema: *Blob*
 - Praktisch alle anderen Distributionen (auch BSDs) benutzen Binärtreiber (NVIDIA, Atheros, Broadcom, Adaptec, ICP Vortex, 3-Ware, Intel, IBM, ...)
 - Binärtreiber widersprechen **komplett** dem OpenSource Gedanken
 - Distributionen machen es trotzdem, weil es *bequem* für die Benutzer ist
- SHAME ON YOU!



Offene Rechnungen

- Thema: *Blob*
 - Praktisch alle anderen Distributionen (auch BSDs) benutzen Binärtreiber (NVIDIA, Atheros, Broadcom, Adaptec, ICP Vortex, 3-Ware, Intel, IBM, ...)
 - Binärtreiber widersprechen **komplett** dem OpenSource Gedanken
 - Distributionen machen es trotzdem, weil es *bequem* für die Benutzer ist
- **SHAME ON YOU!**



Offene Rechnungen (2)

- Thema: Finanzen
 - OpenSSH ist **der** de-facto Weg für sichere Logins
 - Linux, Solaris, HP-UX, *BSD, was-auch-immer
 - OpenSSH ist direkter Bestandteil von OpenBSD
- Grosse Firmen wie Sun, HP oder Cisco setzen OpenSSH in ihren Produkten ein und verdienen Millionen
- Der Großteil der Spenden kommt von Privatleuten wie uns.
- SHAME ON YOU!

Offene Rechnungen (2)

- Thema: Finanzen
 - OpenSSH ist **der** de-facto Weg für sichere Logins
 - Linux, Solaris, HP-UX, *BSD, was-auch-immer
 - OpenSSH ist direkter Bestandteil von OpenBSD
- Grosse Firmen wie Sun, HP oder Cisco setzen OpenSSH in ihren Produkten ein und verdienen Millionen
- Der Großteil der Spenden kommt von Privatleuten wie uns.
- **SHAME ON YOU!**

Was ist neu in OpenBSD 3.9?

- Noch bessere Hardwareunterstützung für Serverhardware (Opteron, IPMI, Dell ESM, I2C/SMBUS)
- Viele neue/verbesserte Netzwerkkartentreiber, gerade für Gigabit Ethernet
- Re-write von `ftp-proxy(8)` und `tftp-proxy(8)`
- Unterstützung für SpeedStep und verwandte Mechanismen in `apmd(8)`
- OpenSSH 4.3
- Viel neue Funktionalität im Host Access Point Daemon `hostapd(8)`

OpenBSD ist interessant für Firmen

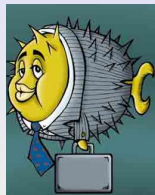
Universitäten Alberta/Canada, Minnesota, Michigan (CITI), Lund, Italian Institute of Nuclear Physics

Firmen Adobe, CORE SDI, Network Security Technologies (**NST**), Alteon Networks

Provider Calyx, BSWS, Globalwire Communications

Non-Profit Universität Helsinki Zentralkrankenhaus, Forcefield Art Installation, Universität Uppsala Krankenhaus

Geheimdienste oft behauptet, nicht nachprüfbar. OpenBSD wurde mal finanziell von der DARPA gefördert



Schwachpunkte von OpenBSD

- Ports-Tree kleiner als der von FreeBSD
- Updates von Version zu Version nicht vollautomatisch – erfordern Sachverstand
- Kein ACPI Support
- Filesystemchecks gecrashter `ffs` Partitionen dauern noch so lange wie bei `ext2`, obwohl SoftUpdates einen `fsck` im Hintergrund ermöglichen würden (vgl. FreeBSD)



Bücher zu OpenBSD

• Absolute OpenBSD

- von MICHAEL LUCAS
- gutes Kompendium, sehr humorvoll
- ISBN 1-886411-99-9

• Secure Architectures with OpenBSD

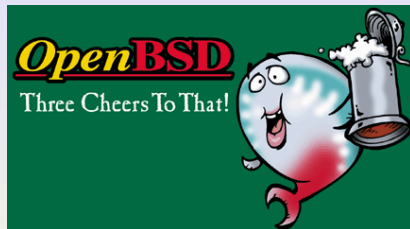
- von BRANDON PALMER und JOSE NAZARIO
- kein stringentes Werk, aber gutes Kochbuch
- ISBN 03-21193-66-0

• Building Firewalls with OpenBSD and PF [2nd edition]

- von JACEK ARTYMIK
- deckt ein Spezialthema ab
- ISBN 83-916651-1-9



Noch Fragen?



- 1 Folien erstellt mit \LaTeX , latex-beamer, make und CVS unter OpenBSD/i386
- 2 Folien erhältlich unter <http://pestilenz.org/~grunk/vortraege/2006/erlug/openbsd.pdf>
- 3 Quellcode der Folien auf Anfrage: `<grunk@pestilenz.org>`