Debian beim ISP

Marc Haber
toplink-plannet GmbH
Schönfeldstr. 8
76131 Karlsruhe
http://www.toplink-plannet.de/
marc.haber@toplink-plannet.de

Inhalt des Vortrags

- Anforderungen beim ISP
- Auswahl der Systemumgebung
- Über Debian
- Basissystem und Updates
- Kernelmanagement und Boot-Prozess
- Package-Verteilung
- Dienste bei toplinik-plannet
- Packageauswahl und Management

Über toplink-plannet

- Als Internet-Provider seit 1995 in Karlsruhe aktiv
- 2001 Ausdehnung nach Württemberg
- 2003 Vorbereitung bundesweiter Dienste
- 10 Mitarbeiter, davon 6 im technischen Bereich
- Leitungsgeschäft / Routing / Netzanbindung
- Klassische Internet-Dienste
- VPN
- Traditionell sehr stark mit freier Software arbeitend

Anforderungen beim ISP

- Viele verschiedene Dienste
- u.U. mehrere Server pro Dienst
- Hochverfügbarkeit: Redundanz/Fallback/Clustering
- Existierende Mechanismen zur Administration vieler gleichartiger Rechner nur eingeschränkt brauchbar:
  - Diese sind eher für Administration von Rechnerpools gemacht

OS-Auswahl:
- Windows scheidet aus
- UNIX -> Kommerziell oder nichtkommerziell
-> Hardwareentscheidung!
- Linux / BSD
- Suse/Redhat/Debian

Welches OS

- Windows
  - Weit verbreitet
  - Viele Anwendungen gibt es nur für Windows
  - Hohe TCO
  - Internetdienste unter Windows oft schlecht implementiert oder fehlerhaft
- Alternative zu Windows: unixoide OS

Welches unixoide OS

- Kommerzielle: Solaris, AIX, HPUX
  - Läuft oft nur auf teurer Hardware
  - Ist auf PC-Hardware sehr wählerisch
  - Keine/kaum Quelltexte verfügbar
- Linux
  - Open Source / GPL
  - Weit verbreitet
  - Know-How leicht verfügbar
  - Aber: Hype.
  - Viele „Möchtegern-Experten“

Welches OS

- Free|Open|NetBSD
  - Open Source / BSD-Lizenz
  - Sehr flexibel durch Ports-Struktur
  - Betrieb schlanker Serversysteme ist durch die Ports-Struktur komplexer
  - Sehr stabil
  - Verhält sich dank besserem Scheduler unter Last besser als Linux
  - Viele Benutzer von Linux abgewandert, elitäre Grundeinstellung

Welche Hardware

- PC-Hardware:
  - Billig
  - Leicht erweiterbar
  - Know-How leicht verfügbar
  - Für kleinere Bedürfnisse ausreichend
  - Mit hinreichend viel Aufwand auch hohe Ansprüche erfüllend:
    - Schlund
    - Google

Welche Linux-Distribution

- SuSE
  - Marktführer in Deutschland
  - Guter Support seitens Hardware- und Software-Anbietern
  - Kocht an vielen Stellen eigenes Süppchen
  - Immer grafischer orientierte Administration macht „schlanke“ Systeme schwer zu installieren
  - Alles in allem eher auf den Desktop eingerichtet

Welche Linux-Distribution

- Redhat
  - Marktführer international
  - Guter Support seitens Hardware- und Software-Anbietern
  - Spielt oft Technologievorreiter auf Kosten der Stabilität

Debian

- Komplett freies System
  - „if it's in Debian, you can use it and sell it“
- Entwickelt ohne kommerzielle Zwänge
  - „it will be released when it's ready“
- „gut abgehangen“ / veraltet
  - Direkte Folge der „release when ready“ Politik
- Drei verfügbare Distributionen:
  - stable – testing – unstable
  - stale – rusting – broken

Unstable: Neueste Packages
Testing: Nach Wartezeit ohne kritische Bugs
Stable: Entsteht nach Freeze aus Testing

Serverdienste bei tpl auf Basis von stable

Warum Debian

- Komplett freies System
- Sehr guter Support aus der Community
- Mitarbeit der Anwender erwünscht und willkommen
- Alle Bugs lückenlos offengelegt
- Entwicklungsprozess sehr transparent
- Öffentliche Entwicklerversionen
  - Vorbereitung auf neue „stable“-Versionen möglich

Was bei Debian noch fehlt

- Verifizierbar zurückverfolgbare Binary-Packages
- Umgang mit Kernel Capabilities
- Unterstützung von ro Filesystemen
- Ausblenden von Subtrees bei der Package-Installation
  - /usr/share/man
  - /usr/share/doc

Welches Debian

- unstable
  - Entwicklerversion
  - Sehr dynamisch
  - Hat oft wirklich böse Bugs
- testing
  - Packages kommen nach Wartezeit von unstable nach testing, wenn:
    - Keine schweren Bugs
    - Alle Dependencies in testing erfüllbar
  - Das gilt auch für Security-Updates!

Welches Debian

- Stable
  - Entsteht nach Feature Freeze aus testing
  - Releasezyklus zur Zeit > 1 Jahr
  - Nach Release nur noch Security-Fixes durch Backport des Fixes
  - Security-Fixes „schnell“ verfügbar
  - Point Releases verwässern dieses Konzept

Debian stable auf unseren Serversystemen

- Sehr stabile, ausgereifte Software
- Einzelne Pakete als Backport aus unstable
  - Ermöglicht aktuelle Software auf stabiler Basis.
  - Macht mehr Arbeit, da Eigenverantwortung bei Security-Advisories gegen backported packages.
- In-House entwickelte Software als .deb
    - Dependencies
    - Maintainer Scripts
    - Automatismen der Administration

Basissystem

- Für den Serverbetrieb abgespecktes Debian-Basissystem.
- Luxus-Packages: editoren, less
- Adminsitrationshilfen: cron-apt, aide, Backup-Client
- ssh-Server
- hosts.deny/hosts.allow

Installation des Basissystems

- Basissystem immer gleich
- basis.tar.gz als Master-Image für einen Server
- Boot von Knoppix(light) auf der Zielhardware
- Einrichtung von Massenspeicher und Filesystemen
- < basis.tar.gz ssh root@ziel tar − − extract − − file − − − preserve − − directory / mnt
- Letztes Customizing (hostname, passwords) über ein Script

Accountmanagement

- Alle Systeme haben alle Accounts lokal
- tpl-accounts.deb
  - fragt bei der Installation den Maschinentyp ab.
  - Legt die zum Maschinentyp passenden Mitarbeiter-Accounts an
  - Installiert Homedirectories (mit passendem ssh-Key)
  - Update der Package legt neue Mitarbeiter-Accounts an und sperrt die Accounts von Ex-Mitarbeitern.

Installation der Dienste

- Alle Software kommt als .deb auf die Systeme
- Oft benötigte Packages haben auf lokale Bedürfnisse angepasste Defaultkonfiguration

Manche Packages unterstützen komplett abgetrennte Konfiguration -> exim4

Distributions- und Security-Updates

- Optimale Unterstützung dank apt-get und cron-apt
- Bei wichtigen Maschinen Vorabtest auf einem staging system
- In den allermeisten Fällen mit minimaler Downtime aus der Ferne machbar

Apt-get zieht Dependencies automatisch hinterher
Maintainer scripts konvertieren ggf.
Konfigurations- und Datendateien
Staging System aus dem Backup des produktiven Systems => Notfallsimulation

Kernel

- Rechnerklassen nach Verwendung
  - Coreserver
  - Paketfilter
  - ATM-Router
- Kernel läuft auf jeder Hardware
  - Prozessortyp
  - IDE/SCSI-Treiber
  - Netzwerkkarten-Module

Kernel-Package

- Baut Kernel, Module, System.map als .deb-Package
- Package hat Maintainer-Scripts für korrekte Anmeldung im Bootmanager
- --append-to-version ermöglicht Unterscheidung der Kernel für unterschiedliche Rechnerklassen
- Package hat Kernel-Versionsnummer im Packagenamen:
  - kernel-image-$VERSION-$KLASSE
  - kernel-image-2.4.20-ac1-coreserver.deb
  - Deswegen: Keine automatischen Updates möglich

Update-Helper notwendig
lokales zehn-Zeilen-Script baut mit Hilfe von equivs den Update-Helper gleich mit

Optimierung der kernel-package

- Ergänzung der kernel-image-Package durch eine Dependency-Package
  - kernel-image-$KLASSE
  - Kernel-Versionsnummer als Version
  - Dependet auf kernel-image-$VERSION-$KLASSE
  - Ermöglicht auf diese Weise automatische Updates des Kernels

Bootmanager: Grub

- Hat Dateisystemkenntnisse
- Kann die Lage des zu bootenden Kernel-Images aus dem Dateisystem erkennen
- Kein Aufruf eines Scripts nach Kernelinstallation oder Konfigurationsänderung notwendig
- Kommandozeile erlaubt Auswahl des Kernels zur Boot-Zeit
- Große Hilfe bei Bootstörungen

Intrusion Detection

- Täglicher Dateiintegritätscheck
- Aide mit lokaler Datenbank
- Nicht optimale Sicherheit, da ein Angreifer die Datenbank kompromittieren könnte
- Andere Ansätze:
  - Regelmäßige Prüfung der Integrität aus einem Analysesystem
  - Übertragung der Datenbank von sicherem Medium

Eigener Debian-Mirror

- dank rsync kostenarm haltbar
- i386+sources
  - Speicherbedarf etwa 45G (rapide steigend)
  - Täglicher mirror pulse ca 400M
- Mirror von debian-security
  - Mirrorprotokoll gibt oft schon Hinweise auf anstehende Fixes vor erscheinen des Advisories.
  - Debian rät von öffentlich erreichbaren Security-Mirrors ab

Eigener Package Pool

- für Backports und lokale Packages
  - Voraussetzung für komplette Updates auch lokaler Packages via apt
- Lokale „Distributionen“ enthalten nur lokal veränderte Packages:
  - tpl/woody: produktiv freigegebene lokale Packages
  - tpl/testwoody: im Test befindliche lokale Packages
- Apt-ftparchive ermöglicht einfachen Bau der nötigen Verwaltungsfiles

Packageverteilung aus eigenem Distributions-Server per http
Mirror der offiziellen Debian-Distribution
Speicherbedarf (i386) ca 30 GB
täglicher Mirrorpuls etwa 400 MB

Package Pool Helper Scripts

- checknewreleases
  - Benachrichtigt, wenn eine Package im lokalen Pool hinter der Unstable-Package hinterherhängt
  - Exceptions halten die Benachrichtigungen kurz
  - Benachrichtigung kann auf ein fälliges Security-Update hinweisen
- mkdists
  - Baut aus im Package-Pool abgelegten Steuerfiles die für apt notwendige Packages.gz

Standardlösungen

- Bind
- Apache
- PHP
- MySQL
- exim/courier-[imap|pop]
- Amanda
- Netsaint
- Webrt
- mrtg/rrfw
- Samba
- Netfilter
- Zebra

Eigene lokale Lösungen

- netbase
- amavis-ng
- ulog-acctd
- netfilter-init

Backports

- Sourcen aus Unstable
- Bau im stable-chroot
  - Basiert auf dem Server-Image, nicht auf debootstrap oder pbuilder.
  - Sorgt für immer passende Dependencies
  - Bau hat keine Auswirkungen auf das Gastgebersystem

Backport dann notwendig, wenn Paket nicht in stable, bzw. Version in stable zu alt
Anstelle lokaler Übersetzung und Installation nach /usr/local: Erstellung einer Debian-Package

Backports

- Probleme
  - Build-Depends
    - Sind immer schwerer zu befriedigen, je weiter sich unstable von stable entfernt
    - Besonders problematisch:
      - Build-Tools wie debhelper oder debconf
      - Libraries (libc, libdb)
      - Perl
  - Library-Bedarf der Binary-Packages
    - Nicht so kritisch wie es scheint
      - iaR schon bei Build-Depends auffallend

Backports

- Veränderung der Versionsnummer im Changelog
  - Backports sollen auf den ersten Blick erkennbar sein
  - Lokale Package überschreibt stable-Package.
  - Neuere stable-Package überschreibt lokale Package
  - 0.3.1-4 wird 0.3.1-4tpl4
  - apt-pinning erlaubt aber heute feinere Kontrolle

Debian-Tools

- apt/dpkg
  - Ermöglicht saubere und einfache Updates
- Cron-apt
  - Kann potentiell apt-updates automatisch durchführen
  - Unverzichtbar zur Verteilung von Security-Updates
  - Wird bei uns dafür verwendet, mails zu verschicken,wenn update notwendig.

Zusammenfassung

- Debian ist die ideale Linux-Distribution für einen kleinen ISP
- Die mitgebrachten Mechanismen sind für die Arbeit auf dem System sehr hilfreich
- Man sollte sie auch für lokale Software und Packages konsequent benutzen
  - Hilft die Anzahl der Sonderfälle (=händisch zu behandelnde Altlasten) auf den Systemen gering zu halten

Grenzen

- Die Security der Softwareverteilung ist verbesserungswürdig
- Security-Features des Kernels werden nicht verwendet
- Bei > 10 wirklich gleichartigen Systemen braucht es andere Mechanismen

Grenzen

- Debian-Monokultur unterstützt Betriebsblindheit gegenüber anderen Betriebssystemen
- Man wird religiös
- Man flucht auf anderen Betriebssystemen nur noch
- Bei der Komplixität heutiger Systeme sind wirklich intime Kenntnisse nur noch für eine einzige Distribution möglich.

Ende des Vortrags

Debian beim ISP

Vielen Dank für Eure Geduld

Marc Haber
marc.haber@toplink-plannet.de

toplink-plannet GmbH
Schönfeldstr. 8
76131 Karlsruhe
http://www.toplink-plannet.de

OOo-Präsentation:
http://q.bofh.de/~mh/stuff/debian-beim-isp.sxi