

rubbish.ch

Chris Cavegn

HAVE YOU EVER READ
**SUCH
RUBBISH**
!??



rubbish.ch

Veröffentlicht 07.11.2004

Copyright © 2004 Chris Cavegn

Diese Dokumentation untersteht der folgenden Lizenz:

English:

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Deutsch:

Dieses Programm ist freie Software. Sie können es unter den Bedingungen der GNU General Public License, wie von der Free Software Foundation veröffentlicht, weitergeben und/oder modifizieren, entweder gemäss Version 2 der Lizenz oder (nach Ihrer Option) jeder späteren Version.

Die Veröffentlichung dieses Programms erfolgt in der Hoffnung, dass es Ihnen von Nutzen sein wird, aber OHNE IRGEND EINE GARANTIE, sogar ohne die implizite Garantie der MARKTREIFE oder der VERWENDBARKEIT FÜR EINEN BESTIMMTEN ZWECK. Details finden Sie in der GNU General Public License.

Sie sollten eine Kopie der GNU General Public License zusammen mit diesem Programm erhalten haben. Falls nicht, schreiben Sie an die Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.

Die komplette Lizenz kann unter <http://www.gnu.org/copyleft/gpl.html> eingesehen werden.

Inhaltsverzeichnis

Einleitung	vii
Anmerkungen	viii
Konventionen	ix
Probleme	x
ToDo	xi
I. Serverkonfiguration	1
1. Anforderungen	2
2. Basiskonfiguration	4
/etc/init.d/networking	4
/etc/mailname	4
apm	4
Tools	4
inetd	5
Login	5
apt	5
3. OpenLDAP	7
Installation	7
Konfiguration	7
/etc/nsswitch.conf	7
/etc/pam.d/	7
/etc/ldap/slapd.conf	8
samba.schema	9
Abschluss	9
Verzeichnisstruktur erstellen	9
OpenLDAP testen	10
Troubleshooting	11
Auflösen der ID's	11
Passwort ändern	12
4. Samba	13
Installation	13
Konfiguration	13
Abschluss	13
Samba testen	14
5. PowerDNS	15
Installation	15
Konfiguration	15
Abschluss	15
LDAP anpassen	15
PowerDNS testen	16
dns lokal einrichten	16
6. dhcpd	17
dhcpd ohne LDAP-Support	17
Installation	17
Konfiguration	17
dhcpd mit LDAP-Support	17
Installation	17
Konfiguration	18
LDAP anpassen	18
dhcpd testen	20
7. Web	21
Apache 2	21
Installation	21
Konfiguration	21
Apache und SSL	21
Apache testen	22
Authentifizierung	22
MySQL	23

Installation	23
Konfiguration	23
Abschluss	23
php	23
Installation	23
Konfiguration	24
Abschluss	24
Testen	24
phpmyadmin	24
Installation	24
Konfiguration	24
Tomcat	25
Java	25
Installation	26
Test 1	27
Apache Tomcat Connector	27
Test 2	27
8. Mail	28
Logging umbiegen	28
Postfix und Procmail	28
Installation	28
Postfix Konfiguration	28
Procmail Konfiguration	29
Abschluss	29
Courier IMAP	30
Installation	30
Konfiguration	30
Abschluss	30
Fetchmail	30
Installation	30
Konfiguration	30
Abschluss	30
Aliases	31
Einrichten von /etc/skel	31
Test 1	31
Virenschanner	32
Installation	32
Testen	32
Zweiter Scanner	33
Spamassassin und Razor	34
Installation	34
Konfiguration	34
AMaViS	35
Installation	35
Konfiguration	35
Abschluss	36
Test 2	36
Trouble Shooting	37
SquirrelMail	37
Installation	37
Konfiguration	37
Abschluss	37
Adressbücher	38
OpenLDAP Zugriffsrechte setzen	38
LDAP Addressbooks-Ast anlegen	38
Adressbuch anlegen	38
9. vmail	40
vmail-User	40
OpenLDAP anpassen	40
Konfiguration	40
Verzeichnisstruktur erstellen	41
jamm einrichten	41
Tomcat anpassen	41

Installation	42
Test 1	42
Postfix anpassen	43
Courier anpassen	43
Squirrelmail	44
Test 2	44
10. vcs	45
Subversion	45
Installation	45
Apache anpassen	45
Testen	45
cvs	46
II. Linux Tips und Tricks	47
11. Arbeiten mit Linux	48
.bashrc & .bash_profile	48
Bash-Prompt einfärben	48
Ausgabe von ls einfärben	48
Anwendungsbeispiel	48
12. Security	50
Authentifikation über private SSH Keys	50
Schlüsselerzeugung	50
Schlüssel Installieren	50
Abschluss	50
Portforwarding mit SSH	50
Portforwarding verhindern	51
SSL Zertifikate & Apache	51
Certificate Authority einrichten	51
Zertifikate für Apache erstellen	52
13. System	54
Magic SysRQ	54
Benützung von Magic SysRQ	54
Vorhandene Kommandos	54
System rebooten	54
Screen	54
Vorhandene Kommandos	55
A. Das LDAP Layout	56
Verzeichnisbaum	56
B. Benötigte Ports	58
C. LDAP Browser	59

Einleitung

Während der Installations meines Servers bin ich über diverse Unklarheiten in vorhandenen Howto's und anderen Dokumenten gestolpert, habe sinnlos Zeit vertrödelt mit eigentlich simplen Sachen (wenn man sie einmal begriffen hat) und mich daran gestört, dass es so wenig wirklich brauchbare deutsche Dokumentation zu diesem Thema gibt. Ich habe zwar kein Problem mit Englisch, aber Deutsch ist mir halt immer noch lieber ...

Ich habe mich entschlossen, meine Installations- und Konfigurationsanleitungen in dieser Form schriftlich festzuhalten, um mir und anderen Interessierten stundenlange Suchaktionen im Internet zu ersparen. Das vorliegende Dokument ist in zwei Sektionen aufgeteilt. Der erste Teil behandelt das eigentliche Installieren und Konfigurieren eines Servers, wobei als Distribution Debian Sarge zum Einsatz kommt. (Sarge ist momentan noch *testing*, sollte aber in Kürze das neue Debian *stable* werden. Ausserdem liefert das Security-Team bereits Sicherheitsupdates.) Im zweiten Teil habe ich einige Tipps und Tricks gesammelt, die ich ganz nützlich finde und die eine Ergänzung zum ersten Teil darstellen.

Diese Dokumentation wird von mir regelmässig ergänzt, erweitert und korrigiert. Ich bin natürlich immer offen für Ergänzungen, Anmerkungen, Bugfixes oder Erweiterungsvorschläge (Die per Mail an <chris@rubbish.ch> gesendet werden können). Die jeweils neuste Version dieses Dokumentes ist irgendwo auf meinem Webserver (natürlich <http://www.rubbish.ch/>) erhältlich.

Viel Spass beim Installieren, Konfigurieren und Testen

Chris Cavegn

Anmerkungen

Anmerkung1:

Diese Anleitung ist **nicht** für Linux Anfänger geeignet! Gewisse Dinge werden einfach vorausgesetzt oder können im Rahmen dieses Dokumentes gar nicht erklärt werden. Es ist auch nicht die Absicht, die Howtos der hier installierten Software zu ersetzen. Es wird vorausgesetzt, dass der Leser bereit ist, sich selber ein paar Stunden mit der Materie zu befassen, sonst sollten konsequenterweise die Finger von diesem Dokument gelassen werden. Für Fragen, Ergänzungen und Berichtigungen bin ich unter <chris@rubbish.ch> erreichbar.

Anmerkung2:

Bei einigen der nachfolgend eingebundenen Konfigurationsdateien sind einige Zeilen so lang, dass sie über den Rand des pdf's hinausschauen würden. Um dies zu verhindern, habe ich die entsprechenden Zeilen mit einem \ umgebrochen. Da dies unter Linux die Standardmethode ist um Zeilenumbrüche zu kennzeichnen, deshalb sollte es eigentlich keine Probleme verursachen. Im Zweifelsfalle sind die Zeilen einfach wieder zusammensetzen, dann sollte eigentlich alles funktionieren.

Anmerkung3:

Ich übernehme **keine** Garantie für die Korrektheit und Sicherheit der vorliegenden Dokumentation. Jeder ist selber verantwortlich für seine Installationen. Wenn dies jemandem nicht zusagt, soll er bitte diese Anleitung beiseitelegen und sich anderweitig Support organisieren.

Konventionen

Um die Lesbarkeit dieser Dokumentation zu erhöhen halte ich mich an die folgenden Konventionen was das Benützen der verschiedenen Schriftarten anbelangt:

- Wenn Eingaben in der Shell gemacht werden müssen, sieht das so aus:

```
mkdir tmp  
ls -lha
```

- Ausgaben von ausgeführten Kommandos sowie Inhalte von Konfigurationsdateien werden so dargestellt:

```
# The primary network interface  
auto eth0  
iface eth0 inet static  
address 10.1.0.13
```

- Drei aufeinanderfolgende Punkte (. . .) stehen für einen Platzhalter. Dies kann zum Beispiel bedeuten, dass Teile eines Logfile-Eintrages weggekürzt wurden oder ein Abschnitt in einem Konfigurationsfile übergangen wird.
- Wenn eine Zeile in einem Konfigurationsfile zu lang ist, so wird dies durch einen Backslash (\) gekennzeichnet. Beim Editieren einer Datei sollte dies als *eine* Zeile behandelt werden. Das sieht dann so aus:

```
Diese Zeile wird zu lang, deshalb wird sie mit einem \  
Backslash getrennt.
```

- Dateinamen sind folgendermassen markiert: `/etc/network/interface`
- Wenn im Text auf etwas wichtiges hingewiesen werden soll, wird dies mit *kursiver Schrift* gemacht.
- Wird im Text von einem Programm gesprochen, so wird dies wie hier am Beispiel von **mkdir** dargestellt.

Alles Andere sollte eigentlich selbsterklärend oder aus dem Zusammenhang ersichtlich sein.

Probleme

vhosts und SSL:

Wenn bei Apache mehrere SSL-Zertifikate für verschiedene vhosts verwendet werden, kann zwar für jeden der vhosts ein eigenes Zertifikat angegeben werden, effektiv wird aber nur eines verwendet. Eine einfache Lösung wäre das verwenden von mehreren IP's (also für jeden vhost der ssl verwendet eine eigene Adresse), dies scheint mir aber ein wenig unrealistisch, vielleicht gibt's ja noch eine andere Lösung.

Zugriffsrechte der Adressbücher:

Es ist momentan noch möglich, die eingerichtete Zugriffskontrolle für die Adressbücher zum umgehen. Bei den virtuellen Usern wird die Domäne nicht überprüft. Das heisst, wenn ich auf das Adressbuch des Users <a@b.c> zugreifen will, und ich die Kontrolle über eine andere Virtuelle Domäne habe, muss in nur den User <a@b.c> meiner eigenen Domäne hinzufügen. Dieser User wird zwar nie Mails empfangen können, aber er hat die notwendigen Rechte, um auf das Adressbuch des richtigen Users <a@b.c> zuzugreifen.

ToDo

apache-logs filtern

nimda und anderes zeugs sollte aus den apache-logs gefiltert werden, so dass die wirklich relevanten sachen auch sichtbar sind.

mail forwarder einrichten

Es gibt Provider, die von dial-up usern keine mails entgegennehmen. Deshalb sollte ein (optionaler, für gewisse domänen gültiger) forwarder eingerichtet werden.

chkrootkit, rootkit hunter, tripwire

zum aufspüren von rootkits, allerdings sollten die binaries und die db's der tools irgendwo schreibgeschützt abgelegt sein (=> usb-stick?)

postmaster / jamm

Jamm sollte für die Verwaltung der Domänen nicht dem admin, sondern einen eigenen Postmaster verwenden. Ausserdem sollten die automatisch eingerichteten postmaster für die einzelnen Domänen Schreibrechte haben. sollten

Anhänge

ldapbrowser vorstellen (screenshot, etc)

dazuko (www.dazuko.org)

Ist dat als On-Access scanner/modul brauchbar?

Liste mit nicht gewarteten Paketen

Einige Pakete werden nicht automatisch mit apt gewartet. Davon sollte eine Liste erstellt werden.

- dhcp3-common, dhcp3-server - j2re, j2sdk

Teil I. Serverkonfiguration

Dieser Teil behandelt die eigentlich Installation und Konfiguration des Servers, wobei die Anleitung auf einen Einsatz des Servers in einer kleineren Umgebung angepasst ist.

Teils Konfigurationen mögen für eine solche kleinere Installation wie Overkill erscheinen (Z.B. ein OpenLDAP Server für ~10 Clients), für meine Zwecke sind sie jedoch optimal und das System wird dadurch erweiterbarer.

Immer wenn ein Konfigurationsbeispiel gezeigt wird, ist die Konfiguration den lokalen Gegebenheiten anzupassen. Beispiel wird immer die Domäne *rubbish.ch* verwendet, was bei einer angepassten Installation kaum je der Fall sein wird.

Kapitel 1. Anforderungen

Beim Installieren meines Servers hatte ich einige Anforderungen aufgestellt und thematisch zusammengefasst. Folgende Tabelle zeigt diese Anforderungen sowie deren Status:

Mailserver

Die Mailuser sollen keine Systemuser sein müssen. Das heisst, sie haben kein home auf dem Server. Dadurch sind die verschiedenen Mailordner zentral gespeichert und können einfach gesichert werden. Wenn möglich sollen diese User in einem LDAP-Server abgelegt sein.

Alle eMails sollen standardmässig auf Viren und Spam gescannt werden.

Die Mails sollen über das Web per Squirrelmail, Horde oder einem anderen Webmail zugänglich sein.

Adressbücher

Jeder User soll über ein persönliches Adressbuch verfügen, ausserdem soll es pro Domäne ein globales Adressbuch geben. Wenn möglich werden alle diese Adressbücher auf dem LDAP-Server abgelegt und sollen auch von einem externen Mailclient wie Evolution abrufbar sein.

Webserver

Soll sowohl über http als auch über https erreichbar sein. PHP muss unterstützt werden. htaccess sollte über LDAP abgewickelt werden. (mod_auth_ldap)

Das Webmail soll unter webmail.rubbish.ch erreichbar sein, wobei sicherzustellen ist, dass der Zugang nur per https gewährt wird.

Dateien sollten über ein Webinterface verwaltet werden können. Zu jeder Datei sollten Metainformationen angezeigt werden. Wenn möglich soll dies im Webmail integriert sein.

VCS-Server

Für meine Softwareprojekte benötige ich einen VCS-Server wie cvs oder subversion. Dies ermöglicht dann auch ein zentrales Backup meiner Projekte.

Die verwalteten Projekte sollen über das Web zugänglich gemacht werden. Für die Projekte muss jeweils separat definiert werden können, wer mit welchen Rechten darauf zugreifen darf.

Die obenstehenden Anforderungen sind komplett erfüllt und funktionieren Einwandfrei. Die unten aufgeführten Punkte sind mindestens teilweise noch unvollständig.

Notizen & Memos

Im Webmail sollen Memos und Notizen abgespeichert werden können. Diese Einträge sollten wenn möglich auch über ein Standalone-Programm abrufbar sein.

Kalender

Es soll pro User ein Kalender zur Verfügung stehen, der sowohl über das Web als auch über ein Programm wie Korganizer oder Evolution abgerufen werden kann.

Samba

Die Samba-User-Verwaltung soll über LDAP abgewickelt werden.

Die Shares sollen wenn möglich durch einen Virenschanner, der 'On-Access' scannt, vor Viren geschützt werden.

Benutzerverwaltung

Über ein Webinterface soll die gesamte Benutzerverwaltung abgewickelt werden können. Also sowohl System- als auch virtuelle Mailuser anlegen, modifizieren und löschen. Ausserdem sollen Domänen angelegt werden können, Sambapasswörter müssen geändert werden können, etc.

Den grössten Teil meiner Ziele habe ich mit der vorliegenden Anleitung erreicht. Die fehlenden Punkte sind hauptsächlich kleinere Mankos welche ich aber mindestens teilweise noch zu beheben gedenke.

Kapitel 2. Basiskonfiguration

Wie in der Einleitung erwähnt, setzte ich als OS für meine Server Debian ein. Dieses lässt sich am einfachsten direkt über das Internet installieren. Dafür muss zuerst eine Boot-CD erstellt werden welche unter <http://www.debian.org/devel/debian-installer/> heruntergeladen werden kann.

Mit dieser NetinstCD kann jetzt ein gewöhnliches Debian stable ohne zusätzliche Pakete installiert werden. (Wie in der Einleitung erwähnt, ist Sarge zum jetzigen Zeitpunkt noch nicht stable, deshalb muss *testing* ausgewählt werden.) Folgende Dinge sollten beachtet werden:

- Dem Server sollte eine fixe IP konfiguriert werden. (Server und IP per dhcp verträgt sich nicht wirklich.)
- Der Servernamen sollte gerade korrekt gesetzt werden.
- Es sollte kein normaler Useraccount angelegt werden.
- Es sollen keine zusätzlichen Pakete installiert werden.
- Der Mail Transfer Agent sollte nicht konfiguriert werden, da er zu einem späteren Zeitpunkt durch Postfix ersetzt wird.

/etc/init.d/networking

Es kann vorkommen, dass der Server beim Ausführen von `/etc/init.d/networking`, also z.B. beim Rebooten oder Herunterfahren des Servers, sehr lange (~10min) hängt. Um dies zu verhindern muss die folgende Zeile in `/etc/init.d/networking` eingefügt werden:

```
echo -n > /etc/network/ifstate
```

Dies führt zu folgendem Datei-Inhalt:

```
stop)
...
else
    echo -n "Deconfiguring network interfaces..."
    echo -n > /etc/network/ifstate
    ifdown -a
    echo "done."
fi
```

/etc/mailname

Damit der Mailserver, der später konfiguriert wird, richtig funktioniert, muss der Inhalt der Datei `/etc/mailname` auf den Wert der primären Domäne gesetzt werden. Im Falle dieser Beispielinstallation ist dies der folgende Wert:

```
rubbish.ch
```

apm

Damit der Server nach dem Herunterfahren auch wirklich abschaltet, muss das Modul *apm* in der Datei `/etc/modules` eingetragen werden. Damit wird erreicht, dass *apm* vom nächsten Reboot an bei jedem Systemstart automatisch geladen wird. Für jetzt muss das Modul von Hand geladen werden:

```
modprobe apm
```

Tools

Bei einem frisch installierten Debian sollten folgende Tools nachträglich installiert werden:

```
1: apt-get install nmap
2: apt-get install lsof
3: apt-get install screen
4: apt-get install less
5: apt-get install bzip2
6: apt-get install ssh
7: apt-get install ntpdate
8: apt-get install rsync
9: apt-get install w3m
```

1. Der beste verfügbare Portscanner ;-)
2. Zeigt offene Sockets und Streams an
3. Zeigt Text-Dateien an
4. Ein Windowmanager, der mehrere virtuelle Terminals über eine physikalische Verbindung verwalten kann. (Für mehr Informationen siehe Beschreibung von Screen)
5. Unterstützung für die bzip2-Komprimierung
6. Ermöglicht die Administration des Servers über das Netzwerk. SSH wird zu einem späteren Zeitpunkt noch richtig konfiguriert.
7. Ermöglicht Zeitsynchronisation z.B. mit `time.ethz.ch`
8. Ein Synchronisationstool, das den Datenabgleich von entfernten Servern ermöglicht.
9. Ein Konsolebrowser der auch Frames unterstützt. Damit auch Tools aus dem Internet heruntergeladen werden können wenn kein graphischer Browser vorhanden ist.

inetd

Debian lässt standardmässig einige Services laufen, die nicht benötigt werden. In `/etc/inetd.conf` sollten deshalb alle Services auskommentiert werden. Inetd muss anschliessend neu gestartet werden. Ein Portscann auf `localhost` sollte jetzt nur noch zwei geöffnete Ports anzeigen. (Port 22, der vom vorhin installierten OpenSSH belegt wird, sowie Port 25, der zum vorinstallierten Mailserver gehört und der zu einem späteren Zeitpunkt ersetzt wird).

```
/etc/init.d/inetd restart
nmap localhost
```

Login

Um klarzustellen, auf welchen Rechner gerade zugegriffen wird, sollte die default-Begrüssung ausgewechselt werden:

```
nano /etc/motd
```

Dabei ist der Dateinhalt z.B. durch folgenden Text zu ersetzen:

```
#####
##### Welcome to #####
##### intern.rubbish.ch #####
#####
```

Um zu verhindern, dass Debian die alte `motd` beim Systemstart wieder erstellt und somit die angepasste Version überschrieben wird, muss zudem in `/etc/default/rcS` folgender Eintrag geändert werden:

```
EDITMOTD=no
```

apt

Um in Debian auch Sourcepakete verwenden zu können, muss der folgende Eintrag in `/etc/apt/sources.list` gemacht werden. (Sourcepakete sind nicht-vorkompilierte Pakete, die aber im Gegensatz zu original Quellen automatisch debian-konform konfiguriert werden.)

```
deb-src http://mirror.switch.ch/ftp/mirror/debian/ sarge main
```

Ausserdem sollten unbedingt security Updates verwendet werden. Die folgende Zeile sorgt dafür,

dass diese benützt werden:

```
deb http://ftp.de.debian.org/debian-security sarge/updates main
```

Um diese Änderungen wirksam zu machen, müssen die der apt-sourcen aktualisiert werden:

```
apt-get update
```

Kapitel 3. OpenLDAP

Ich will möglichst keine Passwörter und sonstige Benutzerinfos auf dem Dateisystem des Servers haben, deshalb habe ich mich dazu entschlossen, die Benutzerverwaltung (und einiges mehr) in einen OpenLDAP Server (<http://www.openldap.org/>) auszulagern.

Für die 'low-level Verwaltung' von OpenLDAP hat sich **LDAP Browser** (siehe Beschreibung von LDAP Browser) bewährt, ein Programm, mit dem sich der Inhalt des LDAP-Servers wie mit einem File-Browser bearbeiten lässt.

Installation

```
1: apt-get install slapd
2: apt-get install ldap-utils
3: apt-get install libldap2-dev
4: apt-get install libpam-ldap
5: apt-get install libnss-ldap
```

1. Der OpenLDAP Server wird nachher manuell konfiguriert, es spielt also keine Rolle was für Angaben gemacht werden.
2. Die Client-Tools (die auch auf dem Server benötigt werden
3. OpenLDAP Entwicklungsbibliotheken, werden benötigt, wenn Software mit ldap Unterstützung selber kompiliert werden soll (Siehe später dhcp mit ldap Patch).
4. Pam-Modul für LDAP: Bei libnss-ldap sollte als *LDAP Server host* „127.0.0.1“ und als *distinguished name* „o=System“ angegeben werden. „md5“ wird als Methode für das Hashen der Passwörter verwendet. Ansonsten können die Voreinstellungen übernommen werden.
5. Benötigt für Authentifikation der User: Als *Root login account* sollte „cn=admin,o=System“ angegeben werden. Anschliessend muss ein Passwort für den konfigurierten User angegeben werden. Dieses Passwort muss gleich sein wie das Passwort, das später als LDAP Admin-Passwort gesetzt wird.

Konfiguration

/etc/nsswitch.conf

Die Untenstehenden Zeilen müssen auf die angegebenen Werte geändert werden:

```
passwd:      ldap files
group:       ldap files
shadow:      ldap files
```

Damit wird erreicht, dass für die Auflösung von Namen, Gruppen und Passwörtern zuerst der LDAP Server und erst nachher die lokalen Files verwendet werden.

/etc/pam.d/

Um Pam LDAP beizubringen, müssen im Verzeichnis `/etc/pam.d/` drei Dateien angepasst werden. (Es ist jeweils der Dateiname sowie der relevante Ausschnitt aus der Datei angegeben.)

common-account:

```
account sufficient pam_ldap.so
account required   pam_unix.so
```

common-auth:

```
auth sufficient pam_ldap.so
auth required   pam_unix.so try_first_pass nullok_secure
```

common-password:

```
password sufficient pam_ldap.so
password required   pam_unix.so nullok obscure min=4 max=8 md5
```

/etc/ldap/slapd.conf

```
# This is the main slapd configuration file. See slapd.conf(5) for
# more info on the configuration options.

#####
# Global Directives:

# Features to permit
#allow bind_v2

# Schema and objectClass definitions
include      /etc/ldap/schema/core.schema
include      /etc/ldap/schema/cosine.schema
include      /etc/ldap/schema/nis.schema
include      /etc/ldap/schema/inetorgperson.schema
include      /etc/ldap/schema/samba.schema

# Schema check allows for forcing entries to
# match schemas for their objectClasses's
schemacheck  on

# Where the pid file is put. The init.d script
# will not stop the server if you change this.
pidfile      /var/run/slapd/slapd.pid

# List of arguments that were passed to the server
argsfile     /var/run/slapd.args

# Read slapd.conf(5) for possible values
loglevel     0

# Where the dynamically loaded modules are stored
modulepath   /usr/lib/ldap
moduleload   back_bdb

backend      bdb
database     bdb

# The base of your directory in database #1
suffix       "o=System"

# Where the database file are physically stored for database #1
directory    "/var/lib/ldap"

# Indexing options for database #1
index        objectClass eq

# Save the time that the entry gets modified, for database #1
lastmod      on

# Where to store the replica logs for database #1
# relogfile  /var/lib/ldap/replug

# The userPassword by default can be changed
# by the entry owning it if they are authenticated.
# Others should not be able to see it, except the
# admin entry below
# These access lines apply to database #1 only
access to attribute=userPassword,sambaNTPassword,sambaLMPassword
           by dn="cn=admin,o=System" write
           by anonymous auth
           by self write
           by * none

# Some attributes that shouldn't be changed
access to attribute=homeDirectory,loginShell,uidNumber,gidNumber, cn,uid,objectClass
           by dn="cn=admin,o=System" write
           by anonymous read
           by self read
           by * read

# Sasl Stuff
access to dn.base="" by * read

# The admin dn has full write access, everyone else
# can read everything.
access to *
           by dn="cn=admin,o=System" write
           by dn="cn=admin1,o=System" write
           by self write
           by * read

rootdn cn=admin1,o=System
```

```
rootpw hallo
```

Die ACL die mit *# Some attributes that ...* beginnt, ist sehr wichtig. Wenn diese Einschränkungen nicht gemacht werden, kann jeder User theoretisch durch manipulieren seines Eintrages root-Rechte erhalten (indem er seine uid auf 0 setzt). Ausserdem sind andere Manipulationen nicht auszuschliessen. Deshalb müssen diese Attribute für den User auf read-only gesetzt werden.

Momentan ist in `slapd.conf` ein zweiter admin (`cn=admin1,o=System`) inklusive Passwort hardcodet. Dies ist nur ein temporärer Zustand und wird anschliessend behoben.

samba.schema

In `/etc/ldap/slapd.conf` wurde vorhin ein Schemafeld für Samba eingebunden (der Eintrag `include /etc/ldap/schema/samba.schema`). Damit dies auch wirklich funktioniert, muss `samba.schema` heruntergeladen und unter `/etc/ldap/schema/samba.schema` abgespeichert werden. Ein aktuelles Schema für Samba ist unter <http://cvs.samba.org/cgi-bin/cvsweb/samba/examples/LDAP/samba.schema> erhältlich.

Während meiner Installation trat beim Neustart von OpenLDAP (kommt später) ein Fehler auf (AttributeType inappropriate matching rule: "caseIgnoreMatch"). Dieser kann sehr leicht behoben werden. Bei den Zeilen 309 sowie 335 von `samba.schema` muss jeweils **caseIgnoreMatch** durch **caseIgnoreIA5Match** ersetzt werden.

Abschluss

Das Startscript des OpenLDAP Servers hat bei mir manchmal nicht korrekt funktioniert, wenn ich den Server mit `/etc/init.d/slapd restart` neu starten wollte. Das Problem war, dass der Server neu gestartet wurde bevor er richtig heruntergefahren war. Auch das lässt sich einfach beheben, in `/etc/init.d/slapd` muss bei der Direktive **restart** zwischen `start` und `stop` ein **sleep 1** eingebaut werden:

```
restart|force-reload)
    stop
    sleep 1
    start
    ;;
```

Um die veränderte Konfiguration zu berücksichtigen, muss OpenLDAP neu gestartet werden. Da auch das Verhalten von pam angepasst wurde, ist auch ein Neustart von SSH notwendig. (Wird SSH nicht neu gestartet, so ist nicht garantiert dass er die aktuelle Konfiguration von pam verwendet.)

```
/etc/init.d/slapd restart
/etc/init.d/ssh restart
```

Auf einer anderen Konsole sollte jetzt getestet werden, ob es immer noch möglich ist, sich als root einzuloggen. (Am Besten von lokal, per SSH, ausserdem sollte das Passwort mit **passwd** geändert werden können). Diese Tests schützen vor unliebsamen Überraschungen ;-)

Verzeichnisstruktur erstellen

Ein LDAP Server ist intern in einer Baumstruktur organisiert (siehe Beschreibung des LDAP Baumes), in diesem Abschnitt geht es darum, diese Grundstruktur zu erstellen und im Server abzuspeichern. Ausserdem wird jetzt der richtige LDAP-Admin angelegt und der Temporäre aus `/etc/ldap/slapd.conf` entfernt.

Zuerst muss das Passwort für den LDAP-Admin gehasht werden. Hier ist es wichtig, dass dasselbe Passwort wie bei der Konfiguration von libnss verwendet wird, sonst wird das System später nicht korrekt funktionieren. Das folgende Beispiel zeigt den Einsatz von **slappasswd** und was die entsprechende Ausgabe sein sollte:

```
sarge:~# slappasswd
New password:
Re-enter new password:
{SSHA}2o3m+OcXe8k0DZRfdrzx+jVj/1XqjWCG
```

Die folgende Datei `struktur.ldif` enthält die Basisstruktur des LDAP Servers. Der angegebene Wert des `userPassword` muss durch den vorhin berechneten Hash ersetzt werden.

```
##### struktur.ldif #####
dn: o=System
o: System
objectClass: organization

dn: ou=People,o=System
ou: People
objectClass: top
objectClass: organizationalUnit

dn: ou=Group,o=System
ou: Group
objectClass: top
objectClass: organizationalUnit

dn: cn=admin, o=System
cn: admin
userPassword:: {SSHA}g0i6OSJ1DpIjWqAkVkFe5dSpNp56b1SB
description: LDAP administrator
objectClass: organizationalRole
objectClass: simpleSecurityObject
```

Diese Konfiguration wird jetzt durch den Befehl **ldapadd** dem LDAP Server hinzugefügt. Die Ausgabe des Kommandos muss wie folgt aussehen, sonst ist ein Fehler aufgetreten der zuerst behoben werden muss.

```
sarge:~# ldapadd -F -D cn=admin1,o=System -f struktur.ldif -x -W
Enter LDAP Password:
adding new entry "o=System"

adding new entry "ou=People,o=System"
adding new entry "ou=Group,o=System"
adding new entry "cn=admin,o=System"
```

Jetzt muss getestet werden, ob der eingerichtete admin korrekt funktioniert:

```
ldapsearch -D "cn=admin1,o=System" -b "o=System" -x -W
```

Dieses Kommando sollte die gesamte Struktur des OpenLDAP Servers auflisten. Falls dies funktioniert, wird *admin1* nicht mehr benötigt. Es können die folgenden drei Zeilen aus `/etc/ldap/slapd.conf` entfernt werden:

```
by dn="cn=admin1,o=System" write # ( bei access to *)
...
rootdn cn=admin1,o=System
rootpw hallo
```

OpenLDAP testen

Nach fertiggestellter Installation ist es das Ziel, dass die User vollständig per Web-Interface administriert werden können. Um aber die OpenLDAP Installation einmal zu testen, kann von Hand ein Testuser angelegt werden.

Zuerst wird die Datei `TestUser.ldif` mit folgendem Inhalt erzeugt:

```
dn: cn=testing, ou=Group, o=System
gidNumber: 1111
userPassword:: e2NyeXB0fXg=
objectClass: posixGroup
objectClass: top
cn: testing

dn: uid=testing, ou=People, o=System
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
shadowLastChange: 12053
uid: testing
uidNumber: 1111
cn: ,,,
loginShell: /bin/bash
gidNumber: 1111
shadowMax: 99999
```

```
gecos: ,,,
homeDirectory: /home/testing
shadowWarning: 7
```

TestUser.ldif enthält den Demo-User *testing*. Dieser soll dem LDAP hinzugefügt werden, was mit folgendem Kommando geschieht:

```
ldapadd -F -D cn=admin,o=System -f testUser.ldif -x -W
```

Folgendermassen kann überprüft werden, ob alles Erwartungsgemäss funktioniert:

```
sarge:~# cd /home
sarge:~# mkdir /home/testing
sarge:~# chown testing:testing /home/testing
sarge:~# ls -l | grep testing
drwxr-xr-x  2 testing testing    4096 Aug 31 20:07 testing/
```

Anhand dieser Kommandos wird sichergestellt, dass der neu angelegte User *testing* dem System auch wirklich bekannt ist, dass also die LDAP Auflösung korrekt funktioniert.

Anschliessend sollten noch ein paar weitere Tests durchgeführt werden, welche hauptsächlich auf das Passwort abzielen. Folgende Dinge werden geprüft: Kann root das Passwort eines Users ändern ohne dessen altes Passwort zu kennen, kann root **su** benützen ohne das Passwort des Users zu kennen, kann *testing* sein eigenes Passwort ändern sowie kann *testing* **su** benützen.

```
sarge:~# passwd testing
New password:
Re-enter new password:
LDAP password information changed for testing
passwd: password updated successfully
sarge:~# su testing
testing@sarge:~# passwd
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for testing
passwd: password updated successfully
testing@sarge:~# su
Password:
testing@sarge:~# exit
```

Anschliessend sollte noch versucht werden, ob sich *testing* per ssh einloggen kann. Falls dies nicht möglich ist, ist sicherzustellen, dass der ssh Server nach der OpenLDAP-Konfiguration neu gestartet wurde.

Nach Abschluss dieses Tests sollte der angelegte User und die Gruppe *testing* nicht gelöscht werden, sie werden beim Testen des Samba Servers noch einmal benötigt.

Troubleshooting

Bei der Installation von OpenLDAP ist sehr schnell einmal ein Fehler gemacht und dann wird die Fehlersuche und -behebung sehr mühsam. Nachfolgend sind die Probleme und deren Ursachen aufgelistet, denen ich bis jetzt begegnet bin.

Auflösen der ID's

Symptome: Als root funktioniert alles wunderbar, aber als normaler User können die UID's und GID's nicht aufgelöst werden.

```
testing@sarge:/home# whoami
whoami: cannot find username for UID 1111

testing@sarge:/home# ls -lh
insgesamt 4.0K
drwxr-sr-x  2 1111    1111    4.0K 2004-08-07 15:04 testing

sarge:/home# su

sarge:/home# whoami
root

sarge:/home# ls -lh
insgesamt 4.0K
drwxr-sr-x  2 testing testing 4.0K 2004-08-07 15:04 testing
```

Fehlerbehebung: Die Ursache für dieses Problem sind falsche Zugriffsrechte auf die Datei `/etc/libnss-ldap.conf`. Root hat Lese- und Schreibrechte, aber die User können nicht lesend darauf zugreifen. Das heisst, wenn ein User angemeldet ist, hat das System keine Möglichkeit mehr, eine Anfrage an den OpenLDAP Server abzusetzen, da dessen Konfiguration nicht bekannt ist.

```
sarge:/etc# cd /etc
sarge:/etc# ls -lh libnss-ldap.conf
-rw----- 1 root root 6.7K 2004-08-06 21:22 libnss-ldap.conf
sarge:/etc# chmod 644 libnss-ldap.conf
```

Passwort ändern

Symptome: Weder als root noch als User kann das LDAP-Passwort geändert werden. Es wird zwar nett gesagt, das Passwort sei erfolgreich gesetzt worden, aber effektiv ist nichts passiert. Wie aus dem Logfile `/var/log/auth.log` hervorgeht, ist da einiges nicht erfolgreich abgelaufen:

```
sarge:/etc# passwd testing
Enter new UNIX password:
Retype new Unix password:
passwd: password updated successfully

aber:

less /var/log/auth.log
... passwd[2429]: pam_ldap: error trying to bind
                        (Invalid credentials)
... PAM_unix[2429]: check pass: user unknown
... PAM_unix[2429]: authentication failure; root(uid=1111)
                        -> testing for passwd service
```

Fehlerbehebung: Der Dialog, der hier präsentiert wird, ist der falsche - die LDAP-User werden gar nicht berücksichtigt. Die Ursache für dieses Problem liegt darin, dass in der Datei `/etc/pam_ldap.conf` der Wert `rootbinddn` falsch gesetzt ist. Dieser sollte `cn=admin,o=System` entsprechen. Eine weitere mögliche Fehlerursache wäre das Fehlen der Datei `/etc/ldap.secret`, in der das Passwort des LDAP-Admin gespeichert werden muss. (Mit den Zugriffsrechten 600.)

Der funktionierende Passwort-Dialog sieht folgendermassen aus:

```
Als root:
sarge:/home# passwd testing
New password:
Re-enter new password:
LDAP password information changed for testing
passwd: password updated successfully

Als User testing:
testing@sarge:/home# passwd
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for testing
passwd: password updated successfully
```

Kapitel 4. Samba

Allfällige Windows Clients sollen unter anderem auf die Homes auf dem Server zugreifen können. Um dies zu erreichen, muss Samba (<http://www.samba.org/>) installiert werden. Wie die Unix-User werden auch die Samba-Benützer im OpenLDAP Server gespeichert.

Installation

Samba wird sehr einfach über apt installiert werden. Was in den Dialogen angegeben wird ist wie bei LDAP nicht so wichtig, da auch Samba manuell konfiguriert wird.

```
apt-get install samba
```

Konfiguration

Samba wird durch das Editieren von `/etc/samba/smb.conf` konfiguriert. Die Datei sollte in etwa so aussehen:

```
[global]
workgroup = rubbish.ch
server string = %h server (Samba %v)

load printers = no

guest account = nobody
invalid users = root

log file = /var/log/samba/log.%m
max log size = 500

dns proxy = no

socket options = TCP_NODELAY, SO_RCVBUF=8196, SO_SNDBUF=8196
panic action = /usr/share/samba/panic-action %d

client code page = 850
character set = ISO8859-15

; Folgende Zeilen beim Hauptserver einkommentieren:
; local master = yes
; os level = 127
; preferred master = yes
; wins support = yes

##### Security #####
security = user
encrypt passwords = true

passdb backend = ldapsam:ldap://localhost, guest
ldap admin dn="cn=admin,o=System"
ldap suffix = "o=System"
ldap user suffix = "ou=People,o=System"

#===== Share Definitions =====

[homes]
hide files = /*.*/
veto files = /*.*/
comment = Home Directories
browseable = no
writable = yes
create mask = 0600
directory mask = 0700
```

Abschluss

Zum Abschluss muss Samba neu gestartet werden. Dannach muss Samba das LDAP admin Passwort bekannt gemacht werden, was mit `smbpasswd` geschieht:

```
/etc/init.d/samba restart
smbpasswd -w <ldap-admin-passwort>
```

Samba testen

Vom OpenLDAP test her sollte noch der User und die Gruppe *testing* vorhanden sein. Dieser User muss jetzt Samba-fähig gemacht werden:

```
smbpasswd -a testing
```

Jetzt sollte mit dem User *testing* auf den Samba Server zugegriffen werden können.

ToDo: Linux Tips & Tricks: Clientseite

Kapitel 5. PowerDNS

PowerDNS ist ein leistungsfähiger DNS Server, der als Backend LDAP verwenden kann. Dies kommt meine Vorhaben zu gute, möglichst wenig Flatfiles verwalten zu müssen. Die Homepage von PowerDNS ist <http://www.powerdns.com/>, das LDAP-Backend wird unter <http://www.linuxnetworks.de/pdnslldap/index.html> entwickelt.

Installation

Die Installation von PowerDNS ist sehr simpel, ein einfaches Kommando genügt:

```
apt-get install pdns pdns-backend-ldap
```

Konfiguration

PowerDNS ist sehr einfach zu Konfigurieren, mit folgender minimaler `/etc/powerdns/pdns.conf` funktioniert PowerDNS bereits inklusive rekursivem DNS:

```
# Erlaube Rekursives DNS vom Lokalen Netzwerk aus
allow-recursion=10.1.0.0/8 127.0.0.0/8

# Ziel fuer Rekursives DNS
recursor=62.2.32.5

# LDAP Backend verwenden
launch=ldap

# LDAP Einstellungen
ldap-host=127.0.0.1
ldap-basedn=ou=dns,o=System
```

Abschluss

```
/etc/init.d/pdns restart
```

startet PowerDNS neu, mit einem lokalen Portscan sollte kontrolliert werden, ob er wirklich läuft. Falls der UDP Port 53 geschlossen ist, sollte das Logfile `/var/log/syslog` konsultiert werden (PowerDNS gibt Fehler beim Starten nicht auf der Konsole aus).

LDAP anpassen

Die DNS-Einträge werden wie bereits erwähnt nicht in Flatfiles, sondern im OpenLDAP Server konfiguriert. Das folgende File zeigt die Struktur der Einträge, diese müssen natürlich den Bedürfnissen entsprechend angepasst werden.

```
dn: ou=dns,o=System
objectClass: organizationalUnit
ou: dns

# Domain
dn: dc=ch,ou=dns,o=System
objectclass: top
objectclass: dnsdomain
dc: ch

dn: dc=rubbish,dc=ch,ou=dns,o=System
objectclass: top
objectclass: dnsdomain
objectclass: domainrelatedobject
dc: rubbish
soarecord: ns.rubbish.ch hostmaster@rubbish.ch 2002010401 1800 3600 604800 84600
nsrecord: ns.rubbish.ch
mxrecord: 10 mail.rubbish.ch
associateddomain: rubbish.ch

# intern
dn: dc=sarge,dc=rubbish,dc=ch,ou=dns,o=System
objectclass: top
```

```
objectclass: dnsdomain
objectclass: domainrelatedobject
dc: sarge
arecord: 10.1.0.10
associateddomain: sarge.rubbish.ch

# sarge-aliases
dn: dc=sarge-aliases,dc=rubbish,dc=ch,ou=dns,o=System
objectclass: top
objectclass: dnsdomain
objectclass: domainrelatedobject
dc: sarge-aliases
cnamerecord: sarge.rubbish.ch
associateddomain: mail.rubbish.ch
associateddomain: ns.rubbish.ch
associateddomain: www.rubbish.ch
associateddomain: webmail.rubbish.ch

# a host
dn: dc=ahost,dc=rubbish,dc=ch,ou=dns,o=System
objectclass: top
objectclass: dnsdomain
objectclass: domainrelatedobject
dc: ahost
arecord: 10.1.0.200
associateddomain: ahost.rubbish.ch
```

Die verschiedenen Einträge sollten eigentlich selbserklärend sein, ansonsten gibt es auf der angegebenen PowerDNS Homepage weitere Dokumentation. Sobald im LDAP eine Änderung gemacht wird, ist sie sofort aktiv, es ist im Gegensatz zu Bind mit Flatfiles kein Neustart erforderlich. Falls PowerDNS eine Anfrage nicht selber beantworten kann, wird sie an den konfigurierten **recursor** weitergeleitet. PowerDNS kann gleichzeitig mit mehreren Domänen umgehen, das ldif-File muss einfach entsprechend angepasst werden.

PowerDNS testen

Am einfachsten wird die Funktionstüchtigkeit von PowerDNS von einem anderen Computer aus mithilfe von *nslookup* überprüft. Innerhalb von **nslookup** kann mit Hilfe des Kommandos **server <serverip>** auf den neu installierten Server umgeschaltet werden. Jetzt können beliebige hosts eingegeben werden, PowerDNS sollte immer die passende Antwort zur Anfrage liefern. (Sowohl selber definierte hosts als auch irgendwelche aus dem Internet.)

dns lokal einrichten

Wenn sichergestellt ist, dass PowerDNS korrekt funktioniert, kann in der Datei `/etc/resolv.conf` der eingetragene Nameserver durch `127.0.0.1` ersetzt werden.

Kapitel 6. dhcpd

Als dhcp Server kommt der dhcpd von ISC zum Einsatz (<http://www.isc.org/index.pl?dhcpd/>). Wahrscheinlich gibt es auch noch andere dhcp-Server für Linux, dieser bildet aber den Quasi-Standard und hat sich überall durchgesetzt. Nachfolgend sind zwei Varianten zur Installation von dhcpd aufgeführt. Die erste verwendet kein LDAP, funktioniert dafür sehr stabil, die zweite kann Dank einem Patch auf den LDAP-Server zugreifen, muss aber nach jedem neustart von LDAP selber auch neu gestartet werden. Auf den weiteren Verlauf dieser Installationsanleitung hat diese Wahl keinen Einfluss.

dhcpd ohne LDAP-Support

Wenn kein LDAP-Support notwendig ist, kann im Gegensatz zu der nachfolgend beschriebenen Variante das normale Debian-Package installiert werden. Es ist also kein Patchen notwendig.

Installation

Nach der Eingabe von

```
apt-get install dhcp3-server
```

ist der dhcp-Server bereits installiert und muss nur noch konfiguriert werden.

Konfiguration

Die Konfiguration des dhcpd geschieht durch die Konfigurationsdatei `/etc/dhcp3/dhcpd.conf`. Der nachfolgende Auszug zeigt, wie diese Datei etwa aussehen sollte:

```
ddns-update-style none;

option domain-name "rubbish.ch";
option domain-name-servers 10.1.0.10;

option subnet-mask 255.255.255.0;
option ip-forwarding false;
default-lease-time 86400;
max-lease-time 604800;

deny bootp;

subnet 10.1.0.0 netmask 255.255.255.0 {
    option routers 10.1.0.2;
    option broadcast-address 10.1.0.255;

    host work {
        hardware ethernet 00:00:e8:7f:7f:49;
        fixed-address work.rubbish.ch;
    }
    host mobile {
        hardware ethernet 00:08:74:3e:46:a2;
        fixed-address mobile.rubbish.ch;
    }

    range 10.1.0.200 10.1.0.249;
}
```

Nun muss die Konfiguration noch mit einem neustarten des dhcpd-Servers abgeschlossen werden:

```
/etc/init.d/dhcp3-server restart
```

dhcpd mit LDAP-Support

Die nachfolgende Installationsanleitung ist teilweise von <http://xenux.danstesoreilles.com/?article=28&skin=skin1> übernommen, wer Spass an Französisch hat findet dort eine entsprechende Version ;-)

Installation

dhcpcd bringt von Haus aus keine Unterstützung für ldap mit. Es gibt aber Patches die ihm diese Funktionalität beibringen. Das bedeutet, dass dhcpcd selber kompiliert werden muss. Dabei ist folgendermassen vorzugehen:

```
apt-get build-dep dhcp3-server
```

Diese Kommando installiert alle Pakete, die benötigt werden, um dhcpcd aus den Quellen zu kompilieren. Die eigentlichen dhcpcd Quellen werden dabei noch nicht heruntergeladen. Der Patch, mit dem dhcpcd um die ldap Funktionalität erweitert wird, kann unter <http://www.lunytune.net/isc-ldap.html> heruntergeladen werden. Zum Zeitpunkt meiner Installation war der dortige Patch veraltet, ich habe unter <http://home.ntelos.net/~masneyb/> eine aktuellere Version gefunden. Der dhcp Server wird mit folgenden Kommandos zuerst heruntergeladen, dann gepatcht, kompiliert und schliesslich in ein `.deb` verpackt: (Die Kommandos sind den aktuellen Versionen des dhcp Paketes und dem Patch anzupassen.)

```
mkdir /root/dhcp
cd /root/dhcp/
apt-get source dhcp3-server
wget http://home.ntelos.net/~masneyb/dhcp-3.0.1rc14-ldap-patch
mv dhcp3-3.0.1 dhcp-3.0.1rc14
patch -p0 < dhcp-3.0.1rc14-ldap-patch
mv dhcp-3.0.1rc14 dhcp3-3.0.1
cd dhcp3-3.0.1
dpkg-buildpackage -uc -b
cp contrib/dhcp.schema /etc/ldap/schema/
```

In der letzten Zeile wird das `dhcp.schema` in den Schemaordner des OpenLDAP kopiert. Dieses wird in dessen anschliessender Konfiguration benötigt. Das Umbenennen vor und nach dem Patchen ist darauf zurückzuführen, dass erstens die Namensgebung vom `isc` und `debian` nicht genau gleich ist und zweitens die Version des Patches und des dhcp Servers nicht ganz übereinstimmt. Die eigentliche Installation des Servers ist jetzt nicht mehr kompliziert und geschieht mit folgenden Kommandos:

```
dpkg -i dhcp3-common_3.0.1-1_i386.deb
dpkg -i dhcp3-server_3.0.1-1_i386.deb
```

Die Installationsroutine wird versuchen, den Server zu Starten und dabei scheitern. Dies ist normal, da noch keine Konfiguration existiert und dhcpcd deshalb nicht lauffähig ist.

Seltsamerweise will **apt-get upgrade -s** den eben installierten dhcp Server aktualisieren. Da diese 'aktuelle' Version keinen ldap-Support enthalten würde, ist das natürlich unbrauchbar. Als work-around kann man mit **dpkg** einzelnen Paketen den Status *hold* verpassen. Diese Pakete werden dann beim Ausführen von **apt-get upgrade** erwähnt, aber nicht geupdated.

```
cd root
dpkg --get-selections | grep dhcp3 > dhcp3-install.txt
sed -e s#install#hold#g dhcp3-install.txt > dhcp3-hold.txt
dpkg --set-selections < dhcp3-hold.txt
rm -rf dhcp3-install.txt dhcp3-hold.txt
```

Durch die Eingabe von **dpkg --get-selections | grep dhcp3** sollte jetzt verifiziert werden, ob die gewünschten Pakete wirklich auf *hold* gesetzt wurden. Ein **apt-get upgrade** sollte jetzt zwar noch dhcp3 Pakete anzeigen, sie sollten aber als 'zurückgehalten' gekennzeichnet sein.

Konfiguration

Auch bei dhcpcd ist dank LDAP die Konfiguration wieder relativ einfach, die `/etc/dhcp3/dhcpcd.conf` sollte so ähnlich aussehen:

```
ldap-server "127.0.0.1";
ldap-port 389;
ldap-username "cn=dhcpmanager,o=System";
ldap-password "dhcp-stuff";
ldap-base-dn "ou=dhcp,o=System";
ldap-method dynamic;
ldap-debug-file "/var/log/dhcp-ldap.log";
```

LDAP anpassen

Um LDAP als Backend für dhcp zu benutzen, muss an ihm einiges angepasst werden. In einem

ersten Schritt muss ein User angelegt werden, der Zugriff auf die Daten hat, dann muss ein Schema für dhcp eingebunden werden, aus Performancegründen werden dann noch neue Indexe definiert und am Schluss noch ein paar Testdaten eingespielt.

Verzeichnisstruktur

Das folgende File enthält den Subtree *dhcp* sowie den dhcp Manager. (User, mit dem der dhcp Server Zugriff auf die dhcp Daten hat.) Das Passwort des dhcp Managers muss mit dem vorhin konfigurierten übereinstimmen und kann wieder mit **slappasswd** generiert werden. Anschliessend muss das File dem LDAP hinzugefügt werden.

```
dn: ou=dhcp,o=System
objectclass: organizationalunit
ou: dhcp

dn: cn=dhcpmanager,o=System
objectclass: top
objectclass: person
cn: dhcpmanager
sn: dhcpmanager
userpassword: {SSHA}+f4DR37Fv9qMs41wDFtBMSdIblevY87q
```

```
ldapadd -F -D cn=admin,o=System -f dhcpd.ldif -x -W
```

/etc/ldap/slapd.conf

Wie beim Installieren des dhcp Servers erwähnt, benötigt dhcp ein zusätzliches Schema im LDAP. Ausserdem müssen für den User *dhcpmanager* neue Zugriffsregeln konfiguriert werden. Um das Arbeiten des dhcpd ein wenig zu beschleunigen, werden zwei zusätzliche Indexe definiert. Die mit einem ++ gekennzeichneten Zeilen müssen zusätzlich hinzugefügt werden.

```
...
include          /etc/ldap/schema/inetorgperson.schema
++include        /etc/ldap/schema/dhcp.schema
...

...
# Indexing options for database #1
index objectClass eq
++index dhcpHWAddress eq
++index dhcpClassData eq
...

...
# Some attributes that shouldn't be changed
access to attribute=homeDirectory,loginShell,uidNumber,gidNumber, cn,uid,objectClass
        by dn="cn=admin,o=System" write
        by anonymous read
        by self read
        by * read

# dhcpd stuff
++access to dn="ou=dhcp,o=System"
++ by dn="cn=admin,o=System" write
++ by dn="cn=dhcpmanager,o=System" write
++ by * read
...
```

Anschliessend ist ein Neustart des OpenLDAP Servers erforderlich. Da neue Indexe definiert worden sind, müssen diese gebildet werden:

```
/etc/init.d/slapd restart
slapindex
```

Jetzt ist der OpenLDAP bereit, die Daten des dhcp Servers zu verwalten.

Testdaten

Das unten stehende File mit Testdaten entspricht vom Aufbau her ziehlich 1:1 einem dhcpd Configfile. Es sollte eigentlich klar sein was welcher Eintrag bedeutet, ansonsten hilft die Manpage zu dhcpd.conf sicher weiter. Auch dieses File kann wieder analog zu den Anderen dem LDAP hinzugefügt werden.

```
dn: cn=sarge,ou=dhcp,o=System
```

```

objectClass: top
objectClass: dhcpServer
cn: sarge
dhcpServiceDN: cn=DHCP Config,ou=dhcp,o=System

dn: cn=DHCP Config,ou=dhcp,o=System
cn: DHCP Config
objectClass: top
objectClass: dhcpService
objectClass: dhcpOptions
dhcpPrimaryDN: cn=sarge,ou=dhcp,o=System
dhcpStatements: ddns-update-style none
dhcpStatements: default-lease-time 86400
dhcpStatements: max-lease-time 604800
dhcpOption: domain-name "rubbish.ch"
dhcpOption: subnet-mask 255.255.255.0
dhcpOption: domain-name-servers 10.1.0.10
dhcpOption: routers 10.1.0.2

dn: cn=10.1.0.0,cn=DHCP Config,ou=dhcp,o=System
cn: 10.1.0.0
objectClass: top
objectClass: dhcpSubnet
objectClass: dhcpOptions
dhcpNetMask: 24
dhcpRange: 10.1.0.200 10.1.0.249
dhcpOption: broadcast-address 10.1.0.255

dn: cn=work,cn=DHCP Config,ou=dhcp,o=System
cn: work
objectClass: top
objectClass: dhcpHost
dhcpHWAddress: ethernet 00:00:e8:7f:7f:49
dhcpStatements: fixed-address 10.1.0.100

dn: cn=mobile,cn=DHCP Config,ou=dhcp,o=System
cn: mobile
objectClass: top
objectClass: dhcpHost
dhcpHWAddress: ethernet 00:08:74:3e:46:a2
dhcpStatements: fixed-address 10.1.0.101

```

Nachdem die Daten dem LDAP hinzugefügt und dieser neu gestartet wurde, ist alles bereit für den Einsatz von dhcpcd. Falls ein Fehler vorliegt, gibt das Logfile `/var/log/syslog` über die Ursache Auskunft.

```
/etc/init.d/dhcp3-server restart
```

Achtung:

Wenn bereits ein DHCP Server am Netz hängt, sollte sichergestellt werden, dieser vor dem Starten von dhcpcd deaktiviert wird. Wenn mehr als ein DHCP Server gleichzeitig betrieben wird, steht ziemlich sicher Chaos bevor ...

dhcpcd testen

Zum Testen sollten zwei PC's eine neue IP beziehen, wobei der eine mit seiner MAC Adresse im LDAP eingetragen sein soll und der andere nicht. Ich hatte zuerst das Problem, dass auch der eingetragene PC eine beliebige IP und nicht die ihm zugewiesene erhielt. Die Ursache war, dass ich nach dem Eintragen der Indexe in `/etc/ldap/slapd.conf` vergessen habe, **slapindex** auszuführen.

tail -f /var/log/syslog ist nützlich, falls irgendetwas nicht korrekt funktioniert. (Damit können in Echtzeit die Logmeldungen des dhcpcd mitverfolgt werden.)

Kapitel 7. Web

Als Webserver kommt selbstverständlich Apache2 mit php und einer MySQL-Datenbank im Hintergrund zum Einsatz.

Apache 2

Installation

```
apt-get install apache2-mpm-prefork
mkdir /home/httpd
mkdir /home/httpd/html
chown www-data:www-data /home/httpd/html
```

Konfiguration

Die folgenden Einträge müssen am Ende der `/etc/apache2/apache2.conf` eingefügt werden:

```
NameVirtualHost <Server-IP>:80
NameVirtualHost <Server-IP>:443
```

Für die Konfiguration eines virtuellen Hosts muss jeweils die unten stehende Vorlage den Anforderungen entsprechend angepasst und nach `/etc/apache2/sites-available/` kopiert werden. Anschliessend wird der neu angelegte vhost mittels eines Symlinks nach `sites-enabled` aktiv geschaltet. Nach einem Neustart von apache sollte die neu konfigurierte Domäne erreichbar sein.

```
<VirtualHost 10.1.0.10:80>
    ServerName www.rubbish.ch
    ServerAdmin webmaster@rubbish.ch

    DocumentRoot /home/httpd/www.rubbish.ch/html
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /home/httpd/www.rubbish.ch/html>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>

    LogLevel warn
    ErrorLog /var/log/apache2/www.rubbish.ch-error.log
    CustomLog /var/log/apache2/rubbish.ch-access.log combined
    ServerSignature On
</VirtualHost>
```

Falls ein vhost unter mehr als einer Adresse erreichbar sein soll, so kann dies durch folgende Zeile konfiguriert werden:

```
ServerAlias rubbish.ch *.rubbish.ch
```

Der vhost ist jetzt also zusätzlich unter `http://rubbish.ch/` sowie jeder beliebigen Subdomain von `rubbish.ch`, die nicht anderweitig konfiguriert ist, erreichbar.

```
cd /etc/apache2/sites-available/
nano www.rubbish.ch
ln -s /etc/apache2/sites-available/www.rubbish.ch \
    /etc/apache2/sites-enabled/www.rubbish.ch
rm /etc/apache2/sites-enabled/000-default
/etc/init.d/apache2 restart
```

Das `rm` löscht den Default-vhost, der nicht mehr benötigt wird.

Apache und SSL

Zuerst muss Apache gesagt werden, dass er auf Port 443 auf Verbindungen warten soll. Dies

geschieht durch editieren der Datei `/etc/apache2/ports.conf`. Es muss einfach die folgende Zeile eingefügt werden:

```
Listen 443
```

Anschliessend muss Apache noch gesagt werden, dass das Modul **mod-ssl** geladen werden soll:

```
a2enmod ssl
```

Diese Konfiguration macht natürlich nur Sinn, wenn auch etwas per HTTPS erreichbar ist. Ein Beispiel für einen virtuellen Host mit SSL unterstützung ist nachfolgend aufgeführt. (Für weitere Hinweise zu diesem Thema siehe [SSL Zertifikate & Apache aus Linux Tips & Tricks](#).)

```
<VirtualHost 10.1.0.10:443>
    ServerName www.rubbish.ch
    ServerAdmin webmaster@rubbish.ch

    DocumentRoot /home/httpd/www.rubbish.ch/html
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /home/httpd/www.rubbish.ch/html>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>

    LogLevel warn
    ErrorLog /var/log/apache2/www.rubbish.ch-ssl-error.log
    CustomLog /var/log/apache2/rubbish.ch-ssl-access.log combined
    ServerSignature On

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/www.rubbish.ch.crt
    SSLCertificateKeyFile /etc/apache2/ssl/server.key
</VirtualHost>
```

Apache testen

Nachdem Apache konfiguriert wurde, muss er neu gestartet werden. Anschliessend kann mit einem Browser getestet werden, ob die korrekten Daten angezeigt werden und ob https funktioniert.

Authentifizierung

Um die Sicherheit zu erhöhen soll es möglich sein, einzelne Verzeichnisse so zu schützen, dass nur ausgewählte Benutzer Zugriff haben sollen. Natürlich sollen auch hier wieder die entsprechenden Informationen im LDAP Server abgelegt sein.

mod_auth_ldap

Die Apache-User sollen sich bei Bedarf wie erwähnt gegenüber dem OpenLDAP identifizieren. Diese Funktionalität wird vom Modul `mod_auth_ldap` erbracht, welches standardmässig mit Apache installiert wird.

Mit folgenden Kommandos wird Apache so konfiguriert, dass `mod_auth_ldap` beim nächsten Start des Webservers geladen wird.

```
cd /etc/apache2/mods-enabled
ln -s /etc/apache2/mods-available/auth_ldap.load
```

htaccess

Es gibt zwei Möglichkeiten `mod_auth_ldap` zu benutzen. Die erste ist das Verwenden von `.htaccess`-Files, die zweite die Konfiguration der geschützten Bereiche direkt im Konfigurationsfile des entsprechenden vhosts. Welche Variante gewählt wird ist Geschmacksache, ich ziehe die zweite vor. Anschliessend eine Konfiguration welche nur einem User Zugriff gewährt, welcher ein gültiges Kennwort besitzt und in einer definierten Gruppe ist.

```
<Location /test>
  AuthName "JustTesting"
  AuthType Basic
  AuthLDAPURL ldap://localhost:389/ou=People,o=System?uid
  AuthLDAPGroupAttribute memberUid
  AuthLDAPGroupAttributeIsDN off
  require group cn=www-test,ou=Group,o=System
  Options Indexes
  AllowOverride None
  Order allow,deny
  Allow from all
</Location>
```

Ein User muss hier also der Gruppe *www-data* angehören.

Es gibt natürlich noch weitere Konfigurationsmöglichkeiten für `mod_auth_ldap`, diese können dem nachfolgenden Link entnommen werden: http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html

Damit die Änderungen wirksam werden, muss Apache neu gestartet werden.

```
/etc/init.d/apache2 restart
```

Falls Fehler auftreten oder sich User nicht korrekt authentifizieren können obwohl sie eigentlich die Berechtigung hätten, sind die Logfiles im Ordner `/var/log/apache2/` sowie `/var/log/syslog` die erste Anlaufstelle. Ausserdem kann es nützlich sein, die Anfragen an den LDAP Server sowie dessen Antworten unter die Lupe zu nehmen. (Loglevel von OpenLDAP anpassen!)

MySQL

MySQL wird als Datenbank für Webanwendungen und sonstige Applikationen verwendet.

Installation

```
apt-get install mysql-server
```

MySQL soll selbstverständlich beim Booten gestartet werden, also diese Option auswählen.

Konfiguration

MySQL ist mit sinnvollen Optionen vorkonfiguriert. Allerdings ist die Datenbank standardmässig aus Sicherheitsgründen nur über Unix Sockets, nicht aber über Netzwerk erreichbar. Falls dies verändert werden soll, muss die Option `skip-networking` im Konfigurationsfile `/etc/mysql/my.conf` auskommentiert werden.

Abschluss

Falls an der Konfiguration von MySQL etwas verändert wurde, muss der Dienst neu gestartet werden.

```
/etc/init.d/mysql restart
```

Als Abschluss sollte jetzt unbedingt noch ein Passwort für den MySQL Administrator gesetzt werden.

```
mysqladmin password <mysql-admin-password>
```

php

php gehört natürlich auch dazu ...

Installation

Zuerst wird das eigentliche php4 installiert, und anschliessend noch ein paar Zusatzpakete:

```
apt-get install libapache2-mod-php4
```

```
apt-get install php4-mysql
apt-get install php4-ldap
apt-get install php4-mcrypt
apt-get install php4-pear
```

Bis jetzt ist noch kein php4-imap Support vorhanden. Bei diesem Packet gibt es irgendwelche Probleme mit den Abhängigkeiten, bei der Installation will es ebenfalls *libapache-mod-ssl* installieren, welches wiederum von anderen Paketen abhängt. Momentan gibt es scheinbar für dieses Problem keine Lösung, deshalb muss man damit Leben oder die Installation von php4-imap sein lassen.

```
apt-get install php4-imap
```

Konfiguration

Eigentlich sollte keine Konfiguration notwendig sein, allerdings habe ich das Problem, dass irgend etwas mit `/etc/mime.types` nicht stimmt. Die Auswirkung davon ist, dass die php-Scripts nicht ausgeführt sondern mir zum Download angeboten werden. Um dies zu beheben, ist sicherzustellen, dass in `/etc/mime.types` nur die folgenden Einträge zu php existieren. Alle Anderen (x-httpd-php3, etc) sind zu löschen.

```
...
application/x-httpd-php          phtml pht php php3 php4
application/x-httpd-php-source  phps
...
```

Abschluss

```
/etc/init.d/apache2 restart
```

Testen

Die php Installation kann sehr einfach getestet werden. Im Document Root des Servers (zum Beispiel `/home/httpd/html/` oder was konfiguriert ist) wird eine Datei mit dem Namen `test.php` mit dem Inhalt

```
<&php phpinfo(); &>
```

angelegt. Wenn alles funktioniert und mit einem Browser auf die Datei zugegriffen wird, sollte eine php-Statistik angezeigt werden, die unter anderem die installierten Optionen anzeigt.

phpmyadmin

Zur Verwaltung des mysql-Servers soll phpmyadmin verwendet werden. Aus Gründen der Sicherheit soll es aber nur aus dem lokalen Netzwerk 10.0.0.0/8 zugänglich sein.

Installation

```
apt-get install phpmyadmin
```

Auf dem Server ist nur apache2 installiert, das soll beim Installieren von phpmyadmin auch so angegeben werden.

Konfiguration

apt-get installiert phpmyadmin unter `/usr/share/phpmyadmin`. Da dieses Verzeichnis nicht im webroot liegt, muss es es für jede Domäne konfiguriert werden, unter der es erreichbar sein soll. In diesem Beispiel soll phpmyadmin nur über https erreichbar sein. Die folgenden Einträge werden also im vhost `www.rubbish.ch-ssl` eingetragen:

```
Alias /phpmyadmin /usr/share/phpmyadmin
<DirectoryMatch /usr/share/phpmyadmin/>
    Options +FollowSymLinks
```

```

AllowOverride None
order allow,deny
allow from 10.0.0.0/8
</DirectoryMatch>

```

Falls jemand aber die url `http://www.rubbish.ch/phpmyadmin/` eingibt, soll er keine Fehlermeldung erhalten, sondern auf die richtige Seite weitergeleitet werden. Dies wird durch den folgenden Eintrag im `vhost www.rubbish.ch` erreicht:

```
Redirect /phpmyadmin https://www.rubbish.ch/phpmyadmin/
```

Anschliessend muss apache neu gestartet werden und phpmyadmin sollte erreichbar sein:

```
/etc/init.d/apache2 restart
```

Tomcat

Falls der Server in der Lage sein soll, jsp's und Servlets zu verarbeiten, muss Tomcat installiert werden. Da Debian von Haus aus ohne Java-Unterstützung daherkommt, ist dies leider nicht so unkompliziert wie die Installation der meisten anderen Pakete. Bevor Tomcat installiert werden kann, muss ein funktionierendes Java vorhanden sein.

Java

Java-Pakete sind in Debian nicht vorhanden und ich habe auch sonst keine fertigen Pakete gefunden (ok, da gibts noch blackdown, aber das überzeugt mich irgendwie nicht wirklich). Allerdings gibt's ein Shellscript namens *j2se-package* von Hubert Schmid (siehe <http://z42.de/debian/>) welches aus Original Sun-Paketen `.deb`'s generiert.

Installation

1. Benötigte Pakete besorgen und installieren

Zuerst müssen die benötigten Pakete von <http://z42.de> heruntergeladen werden. Ausserdem werden noch Hilfspakete und Bibliotheken benötigt, die ebenfalls installiert werden:

```

wget http://z42.de/debian/old/j2se-package_0.9_all.deb
wget http://z42.de/debian/old/sun-j2re1.4debian_0.9_all.deb
wget http://z42.de/debian/old/sun-j2sdk1.4debian_0.9_all.deb

apt-get install debhelper
apt-get install java-common
apt-get install libxp6
apt-get install libxt6
apt-get install libxtst6
apt-get install libx11-6

```

Anschliessend kann *j2se-package* installiert werden:

```
dpkg -i j2se-package_0.9_all.deb
```

2. **j2re und j2sdk besorgen**

Unter <http://java.sun.com/j2se/downloads.html> muss j2re und j2sdk heruntergeladen werden. Dabei ist darauf zu achten, dass die Linuxversion die auf `.bin` endet, benötigt wird.

3. **Pakete erzeugen**

Die folgenden Befehle installieren *j2se-package* und erzeugen die Debian Pakete:

```

j2se-package j2re-1_4_2_05-linux-i586.bin
j2se-package j2sdk-1_4_2_05-linux-i586.bin

```

(Wobei die genaue Versionsnummer natürlich angepasst werden muss.)

4. **Pakete installieren**

Die vorhin erzeugten Pakete müssen jetzt noch installiert werden:

```
dpkg -i sun-j2re.4_1.4.2+05_i386.deb
dpkg -i sun-j2sdk1.4_1.4.2+05_i386.deb
dpkg -i sun-j2re1.4debian_0.9_all.deb
dpkg -i sun-j2sdk1.4debian_0.9_all.deb
```

Die zusätzlichen Pakete, die mit dpkg installiert werden, sind spezielle Anpassungen an Debian, die von <http://z42.de/debian/> angeboten werden.

Konfiguration

Um Java benutzen zu können, müssen noch ein paar Umgebungsvariablen gesetzt werden. Dies geschieht durch Anpassen der folgenden zwei Konfigurationsfiles:

/etc/profile

```
...
# Set Java-Stuff
export CLASSPATH=./usr/lib/j2sdk1.4-sun/lib:/usr/lib/j2sdk1.4-sun/jre/lib
export JAVA_COMPILER=javacomp
export JAVA_HOME=/usr/lib/j2sdk1.4-sun
export PATH=$PATH:/usr/lib/j2sdk1.4-sun/bin
...
```

/root/.profile

```
...
export CLASSPATH=./usr/lib/j2sdk1.4-sun/lib:/usr/lib/j2sdk1.4-sun/jre/lib
export JAVA_COMPILER=javacomp
export JAVA_HOME=/usr/lib/j2sdk1.4-sun

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/bin/X11
PATH=$PATH:/usr/lib/j2sdk1.4-sun/bin
export PATH
...
```

Jetzt sollte Java installiert und funktionsbereit sein.

Testen

Java wird am einfachsten getestet, in dem eine simple Javaklasse erst kompiliert und anschliessend ausgeführt wird. Diesem Zweck dient die nachfolgende Datei `Test.java`.

```
public class Test{
    public static void main(String argv[]){
        System.out.println("Java funktioniert");
    }
}
```

Der Test wird folgendermassen ausgeführt:

```
javac Test.java
java Test
```

Falls alles erfolgreich verlaufen ist, sollte jetzt die Ausgabe

```
Java funktioniert
```

auf der Konsole erschienen sein.

Falls die Java-Binaries nicht funktionieren, kann der folgende Befehl ausprobiert werden:

```
source ~/.profile
```

Installation

Wenn Java erst einmal funktioniert ist die Tomcatinstallation sehr einfach, er lässt sich ganz normal mit apt installieren.

```
apt-get install tomcat4
mkdir /home/tomcat/
```

```
cd /var/lib/tomcat4
mv webapps/ /home/tomcat/
ln -s /home/tomcat/webapps
nano /etc/default/tomcat4
/etc/init.d/tomcat4 restart
```

In `/etc/default/tomcat4` ist die folgende Zeile entsprechend anzupassen:

```
JAVA_HOME=/usr/lib/j2sdk1.4-sun
```

Tomcat ist ein *contrib*-Paket. Wenn also eine Fehlermeldung im Sinne von „Das Paket ist unbekannt“ kommt, ist sicherzustellen, dass apt auch die contrib-Pakete berücksichtigt.

Test 1

Um das Funktionieren von Tomcat zu überprüfen kann entweder ein Test-jsp geschrieben oder das Paket *tomcat4-webapps* installiert werden. Die Variante mit *tomcat4-webapps* ist sicher die einfachere, auch können so komplexere Tests durchgeführt werden.

Apache Tomcat Connector

Bis jetzt sind Apache und Tomcat zwei unabhängige Server welche logischerweise auf verschiedenen Ports laufen. Das Ziel ist jedoch, dass Apache der Primäre Webserver ist und den grössten Teil aller Anfragen beantwortet. Wenn er jedoch einen Request empfängt, der auf eine jsp-Seite oder ein Servlet zielt, so soll dieser automatisch und transparent an Tomcat weitergeleitet werden. Diesen Zweck erfüllt das Apache Modul *libapache2-mod-jk2*.

Installation

```
apt-get install libapache2-mod-jk2
```

Konfiguration

TODO ja das wüsste ich auch gerne ...

Test 2

TODO kommt noch

Kapitel 8. Mail

Als SMTP-Server kommt Postfix in der Version 2 zum Einsatz. Den IMAP-Part übernimmt Courier IMAP, ausserdem werden Spamassassin, AMaViS sowie procmail für verschiedene Filteraufgaben hinzugezogen.

In diesem Kapitel werden noch keine virtuellen Domänen eingerichtet, dies wird im nachfolgenden Teil behandelt.

Einige Teile der Abschnitte Spamassassin und AMaViS sind von <http://workaround.org/articles/ispmail-sarge/> übernommen, dieses Tutorial ist sicherlich einen Blick wert.

Logging umbiegen

Das Mailsystem generiert verschiedene logfiles. Diese sollen aus Gründen der Übersichtlichkeit in einem Unterordner von `/var/log` zusammengefasst werden. In einem ersten Schritt muss dafür `/etc/syslog.conf` angepasst werden. Die folgenden vier Zeilen sind entsprechend anzupassen:

```
mail.*                -/var/log/mail/mail.log
...
mail.info             -/var/log/mail/mail.info
mail.warn             -/var/log/mail/mail.warn
mail.err              /var/log/mail/mail.err
```

Anschliessend sind folgende Kommandos auszuführen um den Unterordner zu erstellen und Syslog neu zu starten:

```
rm -rf /var/log/mail.*
mkdir /var/log/mail
chmod 777 /var/log/mail
/etc/init.d/sysklogd restart
```

Postfix und Procmail

Postfix ist ein reiner SMTP-Server, der Lookups in LDAP machen kann. Ausserdem unterstützt er Maildir's, was von Courier IMAP vorausgesetzt wird. Procmail ist ein Filtertool, das es erlaubt, einkommende Mails zum Beispiel auf verschiedene Ordner zu verteilen. Da Postfix die Mails direkt an Procmail weiterleiten soll, werden die Pakete zusammen installiert.

Installation

```
apt-get install postfix
apt-get install procmail
```

Postfix wird anschliessend manuell konfiguriert, es kann deshalb die Option *Keine Konfiguration* gewählt werden.

Postfix Konfiguration

Die Datei `/etc/postfix/main.cf` sollte in etwa so aussehen:

```
# Command Directories
command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix

# mailuser settings
mail_owner = postfix
default_privs = nobody
setgid_group = postdrop

# SENDING MAIL
myorigin = /etc/mailname
append_dot_mydomain = no
myhostname = sarge.rubbish.ch

# RECEIVING MAIL
```

```
inet_interfaces = all
mydestination = rubbish.ch, $myhostname, localhost.$mydomain, localhost

# REJECTING MAIL FOR UNKNOWN LOCAL USERS
local_recipient_maps = $alias_maps unix:passwd.byname
unknown_local_recipient_reject_code = 550

# TRUST AND RELAY CONTROL
mynetworks = 127.0.0.0/8, 10.1.0.0/8

# ALIAS DATABASE
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases

# ADDRESS EXTENSIONS (e.g., user+foo)
recipient_delimiter = +

# DELIVERY TO MAILBOX
mailbox_command = /usr/bin/procmail -a "$EXTENSION"
mailbox_size_limit = 0

# JUNK MAIL CONTROLS
#header_checks = regexp:/etc/postfix/header_checks
body_checks = regexp:/etc/postfix/body_checks

# SHOW SOFTWARE VERSION OR NOT
smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
```

Niemand verschickt heutzutage noch exe's oder andere ausführbare Daten, die nicht gepackt sind - ausser natürlich Viren und Trojaner. Deshalb sollten Dateien mit einer solchen Endung vom Mailserver aus verboten werden. Um dies zu erreichen, genügt es, die Datei `/etc/postfix/body_checks` zu erstellen und folgende beiden Zeilen einzutragen:

```
/(.*)name=\\"(.*)\".(hta|com|pif|vbs|vbe|js|jse|exe|bat|cmd|vxd|scr|chm)\"$/ REJECT
/(filename|name)=\".*\".(asd|chm|exe|dll|hlp|hta|js|ocx|pif)\"/ REJECT
```

Achtung:

Falls diese Body-Checks nicht gemacht werden sollen, so ist in der Datei `main.cf` die Zeile `body_checks ...` auszukommentieren. Wenn die Zeile nicht auskommentiert ist und `body_checks` nicht existiert, wird Postfix beim Starten einen Error loggen und nicht funktionieren.

Procmail Konfiguration

Die folgende Datei zeigt ein Beispiel einer Procmail Konfiguration für einen User. Die `.procmailrc` wird im Home des Users abgelegt und bei jedem Mail verarbeitet, dass dieser erhält.

```
:0:
* ^X-Spam-Status: Yes
$HOME/Maildir/.Eingang.SPAM/

:0:
* 1^0 ^FROM:.*(root|webmaster|postmaster|virus)@rubbish.ch
* 1^0 ^TO_(root|webmaster|postmaster|virus)@rubbish.ch
$HOME/Maildir/.Eingang.System/

:0:
*
$HOME/Maildir/.Eingang/
```

Der erste Filter kopiert alle erkannten Spam-Mails in den Unterordner SPAM. Der zweite kümmert sich um Systemmails, die von und zu den aufgeführten Usern gesendet wurden. Normalerweise werden mehrere Filterzeilen *und*- Verknüpft. Durch das zusätzliche `1^0` am Anfang der Zeile kann aber auch eine *oder*-Verknüpfung realisiert werden. Der dritte Filter kopiert die restlichen Mails in den normalen Posteingang.

Mit Procmail sind auch weitere Aktionen wie zum Beispiel Weiterleitungen machbar. Eine komplette Dokumentation ist unter <http://www.procmail.org/> erhältlich.

Abschluss

Zum Abschluss muss Postfix neu gestartet werden:

```
/etc/init.d/postfix restart
```

Courier IMAP

Von Courier wird in meiner Email-Installation nur der IMAP Server verwendet. (Die gesamte Courier-Suite umfasst auch einen SMTP sowie einen POP Server.) Auch hier war wieder einer der ausschlaggebenden Gründe dass Courier eine LDAP-Anbindung ermöglicht.

Installation

```
apt-get install courier-imap
```

Dummerweise überschreibt die Installation von Courier IMAP das File `/etc/pam.d/imap` so dass der neu installierte Server nicht richtig funktioniert. Dem kann leicht abgeholfen werden:

```
cd /etc/pam.d/  
cp other imap
```

Anschliessend sollte Courier korrekt funktionieren.

Konfiguration

In der Datei `/etc/courier/imapd` sollten die folgende Zeile geändert werden:

```
IMAP_CHECK_ALL_FOLDERS=1
```

Diese Änderung sorgt dafür, dass in allen Ordnern geschaut, ob neue Mails angekommen sind. Dies macht insbesondere dann Sinn, wenn die einkommenden Mails mit Procmail auf Unterordner verteilt werden.

Abschluss

```
/etc/init.d/courier-imap restart
```

Fetchmail

Fetchmail ist ein Tool, das Emails von anderen POP- und IMAP-Servern holt und sie lokal wieder dem Mailsystem übergibt. Dies ist ideal, da dann alle externen Mailkonti nicht mehr manuell geprüft wrden müssen sondern auch deren Mails automatisch auf dem zentralen Mailserver landen.

Installation

```
apt-get install fetchmail  
touch /var/log/mail/fetchmail  
chown fetchmail /var/log/mail/fetchmail
```

Die Verzeichnisse für die *WWW-Administration* sollten nicht angelegt werden.

Konfiguration

Es muss ein neues File mit dem Namen `/etc/fetchmailrc` angelegt werden, das die abzurufenden Emailkonti enthält. `fetchmailrc` hat folgendes Format:

```
set logfile /var/log/mail/fetchmail  
set no bouncemail  
  
poll pop.swissonline.ch with proto pop3  
user "<remote-user>" pass "<remote-pass>" to <local-user>;  
mda "/usr/lib/sendmail -i -oem -f %F %T";
```

Abschluss

```
/etc/init.d/fetchmail restart
```

Aliases

In `/etc/aliases` können sogenannte Mail-Alias eingetragen werden. Falls ein User, der dort eingetragen ist, ein Mail bekommt, wird dieses an den konfigurierten Account weitergeleitet. Für die Mails der folgenden Systemaccounts sollte auf jeden Fall ein Alias auf einen User eingerichtet werden:

```
# /etc/aliases
##### System #####
mailer-daemon: postmaster

www: root
abuse: root
nobody: root
security: root
webmaster: root
hostmaster: root
postmaster: root

##### System to User #####
root: testing
```

`testing` steht dabei für den User, der die Systemmails erhalten soll und ist entsprechend anzupassen. Damit die Änderungen in `aliases` gültig werden, muss es mit dem Kommando **newaliases** in eine Binärdatei umgewandelt werden.

```
newaliases
```

Einrichten von /etc/skel

Der folgende Abschnitt gilt nur für lokale User. Bei den im nächsten Kapitel eingerichteten vmail-User hat diese Konfiguration keinen Einfluss. Da aber wahrscheinlich auch lokale User verwendet werden, macht die Konfiguration von `skel` trotzdem Sinn.

Damit neuen Usern eine einheitliche Mail-Ordnerstruktur zur Verfügung gestellt wird, kann `/etc/skel` folgendermassen eingerichtet werden:

```
cd /etc/skel
maildirmake Maildir
maildirmake Maildir/.SPAM
maildirmake Maildir/.Drafts
maildirmake Maildir/.Sent
maildirmake Maildir/.Trash
```

Ausserdem sind die folgenden beiden Dateien anzulegen:

```
# /etc/skel/Maildir/courierimapsubscribed:

INBOX.Sent
INBOX.Trash
INBOX.Drafts
INBOX.SPAM
```

```
# /etc/skel/.procmailrc

:0:
* ^X-Spam-Status: Yes
$HOME/Maildir/.SPAM/

:0:
*
$HOME/Maildir/
```

Wird jetzt ein neuer lokaler User angelegt, so wird der komplette Inhalt von `/etc/skel` in dessen Home kopiert und damit hat der neue User eine vorbereitete Maildir-Struktur zur Verfügung.

Test 1

Jetzt ist es an der Zeit, die bisherige Installation des Mailservers zu testen:

1. Zuerst muss der User *testing* auf das empfangen von Mails vorbereitet werden:

```
cd /etc/skel
cp -r .procmailrc Maildir /home/testing/
cd /home/testing
chown -R testing:testing .procmailrc Maildir
```

2. Anschliessend muss ein Mailprogramm für das benützen des Installierten Servers konfiguriert werden. (Sowohl IMAP als auch SMTP) Wenn man sich per IMAP auf dem Server einloggen kann, ist der Test schon halb bestanden ;-)
3. Wenn jetzt ein Mail an *testing* gesendet wird, sollte es im Posteingang auftauchen.
4. Falls es Probleme gibt, können folgende Logfiles helfen:

```
/var/log/mail/mail.log
/var/log/syslog
/var/log/auth.log
```

Virens Scanner

AMaViS ist selber nicht ein Virens Scanner, sondern ein Daemon, der Mails entgegennimmt, sie von externen Tools scannen lässt und sie dann wieder dem SMTP-Server übergibt. Deshalb muss zusätzlich zu AMaViS ein richtiger Virens Scanner installiert werden, wobei ich den Scanner *clamav* verwende. Informationen zu diesem Scanner sind unter <http://www.clamav.net/> erhältlich.

Installation

```
apt-get install arj unzoo zoo lzop arc
apt-get install clamav-testfiles
apt-get install clamav
apt-get install clamav-daemon
```

Zuerst werden ein paar (Ent)Pack-Programme installiert, so dass clamav auch komprimierte Anhänge scannen kann. Anschliessen wird der Scanner selbst sowie Testfiles für diesen installiert.

Bei der Installation von clamav sind folgende Punkte zu beachten:

1. Als Aktualisierungsmethode ist *daemon* empfehlenswert.
2. Es sollte ein möglichst naher Mirror ausgesucht werden.
3. Die Datenbank des Daemon soll selbstverständlich bei updates aktualisiert werden.

Testen

Um zu überprüfen, ob die Installation funktioniert, sollte folgendes Kommando ausgeführt werden:

```
/etc/init.d/clamav-freshclam restart
```

Im Logfile `/var/log/clamav/freshclam.log` sollte ersichtlich sein, dass die Datenbanken „up to date“ sind.

Anhand der installierten Testfiles kann kontrolliert werden, ob *clamscan* korrekt funktioniert:

```
cd /usr/share/clamav-testfiles
clamscan test
clamscan test.bz2
clamscan test.msc
```

```
clamscan test.rar
clamscan test.zip
```

Alle Scanns sollten mit der Meldung enden, dass die *ClamAV-Test-Signature* gefunden wurde.

Zweiter Scanner

Falls aus Sicherheitsgründen ein zweiter Scanner installiert werden soll, bietet sich *F-Prot* an.

Installation

```
apt-get install f-prot-installer
```

Konfiguration

F-Prot ist leider nicht so komfortabel ausgestattet wie clamav, deshalb muss für automatische Updates ein wenig mehr konfiguriert werden:

Für automatische Datenbank-Updates muss die folgende Zeile in `/etc/crontab` eingetragen werden:

```
46 1,7,13,19 * * * root /usr/sbin/update-f-prot > /var/log/mail/f-prot-update.log
```

Das F-Prot Update-Script wird also alle 6 Stunden jeweils in der 46'ten Minute aufgerufen. Um sicherzustellen, dass der neue Eintrag in der crontab beachtet wird, sollte cron neu gestartet werden:

```
/etc/init.d/cron restart
```

Bis jetzt wird nur die Datenbank mit den Virendefinitionen geupdated. Das folgende Script sorgt dafür, dass auch der Virenschanner selber aktualisiert wird. Dieses Script ist nicht unbedingt notwendig, mit **update-f-prot -i** liesse sich dasselbe erreichen. Allerdings führt das Script während dem Updaten ein Locking des Scanners durch, so dass verhindert wird während dem Updaten gescannt wird.

```
/usr/sbin/f-prot-pgm-autoupdate:
```

```
#!/usr/bin/perl
use Sys::Syslog;

$updateDir = shift || "/usr/sbin";
$lockFile = "/tmp/FProtBusy.lock";

$LOCK_SH = 1;
$LOCK_EX = 2;
$LOCK_NB = 4;
$LOCK_UN = 8;
$FProtIsLocked = 0;

Sys::Syslog::openlog("update-f-prot", 'pid, nowait', 'mail');

BailOut("Installation dir \"\$updateDir\" does not exist!")
  unless $updateDir ne "" && -e $updateDir;

&lockFProt();
system("$updateDir/update-f-prot -i");
&unlockFProt();
Sys::Syslog::syslog('info', "F-Prot program updated");
Sys::Syslog::closelog();
exit 0;

sub BailOut {
  &unlockFProt();
  Sys::Syslog::openlog("F-Prot program update", 'pid, nowait', 'mail');
  Sys::Syslog::syslog('err', @_);
  Sys::Syslog::closelog();
  warn "@_, $!";
  exit 1;
}

sub lockFProt {
  open(LOCK, ">$lockFile") or return;
  flock(LOCK, $LOCK_EX);
```

```

print LOCK "Locked for updating F-Prot program files by $$\n";
$FProtIsLocked = 1;
}

sub UnlockFProt {
return unless $FProtIsLocked;
print LOCK "Unlocked after updating F-Prot program files by $$\n";
unlink $LockFile;
flock(LOCK, $LOCK_UN);
close LOCK;
}

```

Dieses File stammt von Bengt Thuree, es kann im Original unter <http://lists.debian.org/debian-user/2003/12/msg03016.html> heruntergeladen werden.

Um die Program-Updates zu aktivieren muss ein Symlink vom Program ins `cron.daily` gesetzt werden:

```

cd /etc/cron.daily
ln -s /usr/sbin/f-prot-pgm-autoupdate

```

Spamassassin und Razor

Für die Spamfilterung wird Spamassassin zusammen mit Razor verwendet. Während Spamassassin aufgrund von vordefinierten Regelsätzen filtert, steht hinter Razor ein Anti-Spam-Netzwerk, in dem reale Spammails erfasst und deshalb zuverlässig erkannt werden. Zusammen bieten diese beiden Tools eine Trefferrate von über 90%.

Installation

```

apt-get install razor
apt-get install spamassassin
apt-get install razor
apt-get install dcc-client

```

Konfiguration

1. Razor ist ein Antispam-Netzwerk, in dem Hashes von bekannten Spammails gesammelt werden. Dies erlaubt eine einfache Prüfung, ob ein einkommendes Mail als Spam klassifiziert wurde oder nicht. Um razor zu konfigurieren, sind die folgenden Kommandos auszuführen:

```

razor-client
razor-admin -home=/etc/amavis/.razor -create
razor-admin -home=/etc/amavis/.razor -discover
razor-admin -home=/etc/amavis/.razor -register

```

In `/etc/razor/razor-agent.conf` sollten die folgenden Zeilen eingefügt respektive angepasst werden:

```

logfile = /var/log/mail/razor.log
razorhome = /etc/amavis/.razor/
debuglevel = 0

```

(Mehr Informationen über spamassassin und razor sind unter <http://wiki.apache.org/spamassassin/UsingRazor> erhältlich.)

2. Für das Verwenden des bayes-Filter muss folgendermassen vorgegangen werden. Zuerst wird in der Spamassassin-Konfigurationsdatei `/etc/spamassassin/local.cf` der Pfad für die Datenbank von bayes gesetzt:

```

bayes_path /etc/spamassassin/bayes

```

Anschliessend wird der Filter über das Kommando **sa-learn** trainiert.

```

sa-learn --spam <Spam-Folder>
sa-learn --ham <Ham-Folder>
chgrp amavis /etc/spamassassin/bayes_*

```

```
chmod 664 /etc/spamassassin/bayes_*
```

Es ist sicherzustellen, dass „Spam-Folder“ und „Ham-Folder“ jeweils auf einen Pfad zeigt, der mindestens 200 Spammails respektive 200 Nicht-Spammails enthält. Ansonsten ist bayes nicht korrekt trainiert.

3. Spamassassin muss in der Datei `/etc/default/spamassassin` aktiviert werden. Dabei ist die folgende Zeile entsprechend anzupassen:

```
ENABLED=1
```

4. Da Spamassassin durch AMaViS eingebunden wird, muss an dieser Stelle keine weitere Konfiguration für die Zusammenarbeit mit Postfix vorgenommen werden. Spamassassin sollte nur noch gestartet werden:

```
/etc/init.d/spamassassin start
```

AMaViS

Installation

```
apt-get install amavisd-new
```

Konfiguration

Die Konfiguration von AMaViS geschieht über das File `/etc/amavis/amavisd.conf`. Folgende Zeilen sollen entsprechend angepasst werden:

```
$mydomain = 'rubbish.ch';
# @bypass_spam_checks_acl = qw( . );
read_l10n_templates('de_DE', '/etc/amavis');
$final_spam_destiny = D_PASS; # (defaults to D_REJECT)
$warnvirusrecip = 1; # (defaults to false (undef))
$warnbannedrecip = 1; # (defaults to false (undef))
$remove_existing_x_scanned_headers = 1; # remove existing headers
$sa_tag_level_deflt = -1000;
```

Falls F-Prot ebenfalls installiert wurde, muss er in `amavisd.conf` nicht speziell konfiguriert werden. Per default wird er als backup-VirusScanner verwendet. Wenn also clamav ausfallen sollte werden die Mails von f-prot geprüft.

Anschliessend muss Postfix so konfiguriert werden, dass AMaViS verwendet wird. Dazu müssen folgende Anpassungen gemacht werden:

```
/etc/postfix/main.cf
```

```
# JUNK MAIL CONTROLS
...
content_filter = amavis:[127.0.0.1]:10024
```

```
/etc/postfix/master.cf
```

```
##### VirusScanner #####
amavis unix - - n - 2 smtp
    -o smtp_data_done_timeout=1200
    -o disable_dns_lookups=yes

localhost:10025 inet n - n - - smtpd
    -o content_filter=
    -o local_recipient_maps=
    -o relay_recipient_maps=
    -o smtpd_restriction_classes=
    -o smtpd_client_restrictions=
    -o smtpd_helo_restrictions=
    -o smtpd_sender_restrictions=
    -o smtpd_recipient_restrictions=permit_mynetworks,reject
    -o mynetworks=127.0.0.0/8
    -o strict_rfc821_envelopes=yes
```

Abschluss

Aus irgendwelchen Problemen mit Berechtigungen muss der User *clamav* der Gruppe *amavis* hinzugefügt werden:

```
adduser clamav amavis
/etc/init.d/clamav-daemon restart
```

Das Alias *virus* muss jetzt noch in */etc/aliases* eingetragen und mit **newaliases** bestätigt werden. Anschliessend ist Postfix und AMaViS neu zu starten:

```
newaliases
/etc/init.d/postfix restart
/etc/init.d/amavis restart
```

In */var/log/mail/mail.log* sollte jetzt überprüft werden, dass beide Antivirenprogramme als primäre Scanner erkannt wurden und auch sonst keine Fehler aufgetreten sind.

Test 2

In diesem zweiten Test geht es darum, die Funktionstüchtigkeit von Spamassassin, clamav sowie AMaViS zu überprüfen. Zuerst sollte ein ganz normales Mail an `<testing@rubbish.ch>` verschickt werden. Im Header sollte dann folgendes sichtbar sein:

```
X-Virus-Scanned: by amavisd-new-20030616-p10 (Debian) at rubbish.ch
X-Spam-Status: No, hits=0.0 tagged_above=-1000 required=6.3 tests=
```

Um zu schauen, ob Spamassassin wirklich funktioniert, kann das folgende Mail mit dem Betreff *junk* versendet werden: (Quelle:

```
/usr/share/doc/spamassassin/examples/sample-spam.txt)
```

```
This is the GTUBE, the
  Generic
  Test for
  Unsolicited
  Bulk
  Email
```

```
If your spam filter supports it, the GTUBE provides a test by which
you can verify that the filter is installed correctly and is
detecting incoming spam. You can send yourself a test mail
containing the following string of characters (in upper case and
with no white spaces and line breaks):
```

```
XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-EMAIL*C.34X
```

```
You should send this test mail from an account outside of your
network.
```

Dieses Mail sollte als Spam gekennzeichnet sein (und je nach *.procmailrc* in den Ordner SPAM verschoben worden sein). Die Header des Mails sollten etwa folgendermassen aussehen:

```
X-Virus-Scanned: by amavisd-new-20030616-p10 (Debian) at rubbish.ch
X-Spam-Status: Yes, hits=1000.0 required=6.3 tests=GTUBE
X-Spam-Level: *****
X-Spam_Flag: YES
```

Um festzustellen, ob Spamassassin die installierten Tools razor, dcc-client sowie den bayes-Filter auch tatsächlich verwendet, soll das Kommando

```
spamassassin --lint -D
```

ausgeführt werden. Die folgenden Einträge sollten in der Ausgabe enthalten sein:

```
debug: bayes corpus size: nspam = x>200, nham = x>200
debug: Razor2 is available
debug: DCC is available: /usr/bin/dccproc
```

Abschliessend muss nur noch der Virens Scanner geprüft werden. Dies geschieht durch das Versenden des EICAR-Teststrings, der per Mail an **testing@rubbish.ch** geschickt wird.

```
X50!P%@AP[4\PZX54(P^)7CC]7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Dieses Mail sollte zwei Antworten zur Folge haben. Erstens erhält der Empfänger eine Benachrichtigung und zweitens bekommt der Postmaster eine Virus-Warmmeldung, die noch mehr Details enthält.

Trouble Shooting

Falls zwar im Header steht, dass das Mail mit AMaViS gescannt wird, aber im Logfile `/var/log/mail/mail.log` die Fehlermeldung `Clam Antivirus-clamd FAILED - unknown status: ... /parts: Access denied. ERROR` auftaucht, ist sicherzustellen, dass der User `clamav` der Gruppe `amavis` hinzugefügt wurde. Ist dies der Fall, so sollten die folgenden Daemons neu gestartet werden:

```
/etc/init.d/clamav-daemon restart
/etc/init.d/amavis restart
```

SquirrelMail

Als Webmail verwende ich Squirrelmail (<http://www.squirrelmail.org/>), dies hauptsächlich deshalb weil SquirrelMail sehr einfach zu bedienen, übersichtlich und klar strukturiert ist. Alternativen wären zum Beispiel Horde (<http://www.horde.org/>) welches auch direkt über `apt-get` installiert werden kann oder OpenExchange (<http://mirror.open-xchange.org/>) welches aber eben erst freigegeben wurde und noch nicht in einer öffentlichen Version vorliegt.

Installation

```
apt-get install squirrelmail
cd /home/httpd
ln -s /usr/share/squirrelmail
```

Konfiguration

Das Webmail soll unter `https://webmail.rubbish.ch/` erreichbar sein, deshalb muss eine neuer virtueller Host angelegt sowie ein passendes SSL-Zertifikat erzeugt werden. (Für mehr Details siehe Apache2 Konfiguration sowie Linux Tips und Tricks - SSL Zertifikate & Apache.

Der Inhalt der Konfigurationsdatei für den neuen vhost sieht dabei etwa so aus:

```
<VirtualHost 10.1.0.10:443>
  ServerName webmail.rubbish.ch
  ServerAdmin webmaster@rubbish.ch

  DocumentRoot /home/httpd/squirrelmail

  LogLevel warn
  ErrorLog /var/log/apache2/webmail.rubbish.ch-ssl-error.log
  CustomLog /var/log/apache2/webmail.rubbish.ch-ssl-access.log combined
  ServerSignature On

  SSLEngine on
  SSLCertificateFile /etc/apache2/ssl/webmail.rubbish.ch.crt
  SSLCertificateKeyFile /etc/apache2/ssl/server.key
</VirtualHost>
```

Da `/home/httpd/squirrelmail` ausserhalb der `DocumentRoot`'s der normalen Domänen liegt, kann SquirrelMail nur über den soeben konfigurierten vhost erreicht werden. Wenn jemand das `https` vergisst und auf `http://webmail.rubbish.ch/` surft, bekommt er eine Fehlermeldung. Schöner wäre eine automatische Umleitung auf die richtige URL. Genau dafür sorgt ein zweiter virtueller Host:

```
<VirtualHost 10.1.0.10:80>
  ServerName webmail.rubbish.ch
  ServerAdmin webmaster@rubbish.ch

  Redirect / https://webmail.rubbish.ch/
</VirtualHost>
```

Abschluss

Anschliessend ist Apache neu zu starten, ausserdem sollte geprüft werden, ob die neuen vhosts funktionieren und Squirrelmail erreichbar ist.

Adressbücher

Die anschliessend angelegten Adressbücher werden auf dem OpenLDAP Server gespeichert. Dadurch sind ihre Einträge im Gegensatz zu normalen, lokalen Kontakten von überallher abrufbar. Das heisst, diese Adressbücher können sowohl mit einem Mailclient wie Evolution oder Thunderbird als auch durch SquirrelMail benützt werden.

Zur Verwendung mit SquirrelMail ist ein separates Plugin notwendig. Dieses befindet sich noch im alpha-Status, funktioniert aber mehr oder weniger fehlerfrei. Es wird in Zukunft irgendwo auf <http://www.rubbish.ch/> erhältlich sein.

Die hier eingerichteten LDAP-Adressbücher funktionieren sowohl mit den normalen System-Usern als auch mit den später konfigurierten vmail-Usern.

OpenLDAP Zugriffsrechte setzen

In `/etc/ldap/slapd.conf` müssen neue ACL's definiert werden, um den Zugriff auf die Adressbücher zu regeln: (Die Einträge sind dabei vor denjenigen ACL's zu machen, die den restlichen Zugriff Regeln, also etwa drittletzte Stelle.)

```
# Adressbook rules
access to dn=".*,ou=(^[^,]+),ou=Addressbooks,o=System"
  by dn="uid=$1,ou=People,o=System" write
  by dn="mail=$1,jvd=.*,ou=Hosting,o=System" write
  by dn="cn=admin,o=System" write
  by * none

access to dn=".*ou=(^[^,]+),ou=Addressbooks,o=System"
  by dn="uid=$1,ou=People,o=System" write
  by dn="mail=$1,jvd=.*,ou=Hosting,o=System" write
  by dn="cn=admin,o=System" write
  by * none
```

Der erste access-Abschnitt ist notwendig, dass Einträge des Adressbuches modifiziert und angeschaut werden dürfen. Der zweite ist für das Erstellen von neuen Einträgen zuständig.

`dn="uid=$1,ou=People,o=System"` trifft auf Systemuser zu,
`dn="mail=$1,jvd=.*,ou=Hosting,o=System"` erkennt die vmail-Accounts.

LDAP Addressbooks-Ast anlegen

Alle Adressbücher sollen zentral unter einem Ast im LDAP abgelegt werden. Dieser wird durch das nachfolgende `addressbooks.ldif` definiert.

```
dn: ou=Addressbooks, o=System
ou: Addressbooks
objectClass: top
objectClass: organizationalUnit
```

```
ldapadd -F -D cn=admin,o=System -f addressbooks.ldif -x -W
```

Adressbuch anlegen

Für jeden User, der ein Adressbuch haben soll, muss jetzt ein solches angelegt werden. Grundsätzlich muss das Adressbuch einfach gleich heissen, wie der Name des Users. Dabei ist zu beachten, dass virtuelle User immer ein @ und die Domäne enthalten, Systemuser dagegen bestehen nur aus dem Usernamen. Anschliessend je ein Beispiel für einen System- und für einen vmail-User:

```
# vmail-user:
dn: ou=test@rubbish.ch,ou=Addressbooks, o=System
ou: test@rubbish.ch
objectClass: top
objectClass: organizationalUnit
description: privat addressbook
```

```
# System-user
dn: ou=testing,ou=Addressbooks, o=System
ou: testing
objectClass: top
objectClass: organizationalUnit
description: privat addressbook
```

Ein einzelner Eintrag sieht dabei etwa folgendermassen aus:

```
dn: cn=Ursli Schellen, ou=testing, ou=Addressbooks, o=System
mail: ursli@heidi.ch
mail: schellen_ursli@alponline.ch
mobile: +41 89 123 45 67
labeledURI: www.alponline.ch
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
homePhone: +41 12 345 67 89
sn: Schellen
cn: Ursli Schellen
```

Sobald das Adressbuch angelegt ist, kann es durch SquirrelMail, Evolution oder andere LDAP fähige Mailclients benützt werden.

Kapitel 9. vmail

Der Mailserver soll für mehrere Domänen Mails empfangen können, ohne dass jede einzeln in `/etc/postfix/main.cf` konfiguriert werden muss. Dies wird kann durch sogenannte 'virtuelle Domänen', die in LDAP abgespeichert sind, erreicht werden. Die User dieser virtuellen Domänen sollen keine 'normalen' Systemuser sein, sondern ebenfalls virtuelle User, die nur dem Empfangen ihrer Mails dienen. Das Ziel dieser Unterscheidung liegt darin, dass diese User auch keine anderen Rechte als das Mailen haben, was eine bessere Rechteverteilung ermöglicht. Ausserdem sollen die Mails der User aller dieser Domänen zentral unter einem Verzeichnis abgelegt werden.

Um möglichst kompatibel mit anderen Installationen zu bleiben, habe ich mich sehr nahe an das `vmail-howto` von *jamm* gehalten. Das original Howto mit teilweise ausführlicheren Erklärungen kann unter <http://jamm.sourceforge.net/> gefunden werden.

vmail-User

Da die Mails aller User der virtuellen Domänen zentral gespeichert werden und diese User dem System selber nicht bekannt sind, muss ein zusätzlicher User angelegt werden, unter dessen Kennung die Mails aller virtuellen User abgelegt werden. Folgende Kommandos legen diesen User und die benötigten Verzeichnisse mit den korrekten Zugriffsrechten an:

```
ldapadd -x -D "cn=admin,o=System" -f vmail.ldif -w
mkdir -p /home/vmail/domains
chown vmail:vmail /home/vmail/domains
```

Wobei `vmail.ldif` in etwa so aussehen sollte:

```
dn: uid=vmail, ou=People, o=System
userPassword:: e2NyeXB0fXg=
loginShell: /bin/false
gidNumber: 4000
uidNumber: 4000
shadowMax: 99999
uid: vmail
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
gecos: ,,,
shadowLastChange: 12053
cn: vmail
homeDirectory: /home/vmail/domains
shadowWarning: 7

dn: cn=vmail, ou=Group, o=System
gidNumber: 4000
userPassword:: e2NyeXB0fXg=
objectClass: posixGroup
objectClass: top
cn: vmail
```

OpenLDAP anpassen

Konfiguration

Zum Speichern der Strukturen der virtuellen Domänen im OpenLDAP Server wird das File `jamm.schema` benötigt. Dieses Schema ist unter <http://jamm.sourceforge.net/> erhältlich und muss im Verzeichnis `/etc/ldap/schema/` abgelegt werden. (Es sollten die Binaries heruntergeladen und entpackt werden, das Schema ist darin enthalten.)

Anschliessend müssen folgende ACL's in die Datei `/etc/ldap/slapd.conf` eingefügt werden. (Die Include-Direktive kommt zu den anderen Includes, der Rest relativ am Schluss der Datei, aber noch vor den zwei allgemeinen Regeln.)

```
include /etc/ldap/schema/jamm.schema
...
##### Jamm Virtual Domains #####
```

```

access to dn=".*jvd=([^\,]+),ou=Hosting,o=System" attr=userPassword
  by group/jammPostmaster/roleOccupant="cn=postmaster,\
  jvd=$1,ou=Hosting,o=System" write
  by dn="cn=admin,o=System" write
  by anonymous auth
  by self write
  by * none

access to dn="ou=Hosting,o=System"
  by group/jammPostmaster/roleOccupant="cn=postmaster,\
  jvd=*,ou=Hosting,o=System" write
  by dn="cn=admin,o=System" write
  by * read

access to dn=".*jvd=([^\,]+),ou=Hosting,o=System"
  by group/jammPostmaster/roleOccupant="cn=postmaster,\
  jvd=$1,ou=Hosting,o=System" write
  by dn="cn=admin,o=System" write
  by self write
  by * read

```

Zum Schluss muss der LDAP-Server neu gestartet werden.

```
/etc/init.d/slaped restart
```

Verzeichnisstruktur erstellen

Nachdem OpenLDAP für die Aufnahme der virtuellen Domänen vorbereitet wurde, muss jetzt im Server noch ein Ast angelegt werden, der dann die Domänen enthalten wird. Das folgende File `hosting.ldif` enthält diesen Ast und muss in die LDAP-Struktur integriert werden.

```

dn: ou=Hosting, o=System
ou: Hosting
objectClass: top
objectClass: organizationalUnit

```

```
ldapadd -x -D "cn=admin,o=System" -f hosting.ldif -W
```

jamm einrichten

jamm (jamm.sourceforge.net) ist ein Manager für virtuelle Domänen. Diese können auch mit dem schonmal erwähnten `ldapbrowser` konfiguriert werden, jamm ist einfach angenehmer. Da jamm eine Java Webapplikation ist, wird Tomcat benötigt.

Tomcat anpassen

Damit jamm anschliessend funktioniert, gibt es zwei Möglichkeiten. Entweder wird der Security-Manager deaktiviert (einfache Variante) oder es werden die korrekten Berechtigungen gesetzt (nach ein paar Stunden suchen: auch einfache Variante). Da das deaktivieren des Security-Managers auf einem Server der im Internet hängt nicht zu empfehlen ist und ich jetzt eh weiss, welche Berechtigungen jamm benötigt, habe ich mich für die zweite Variante entschlossen.

jamm setzt auf struts auf, welches wiederum eine spezielle Berechtigung benötigt. Natürlich ist diese im Debian Tomcat defaultmässig nicht aktiviert und muss manuell konfiguriert werden. Ausserdem muss jamm die Berechtigung gegeben werden, auf Port 389 des LDAP-Server zuzugreifen. Dazu wird die Datei `/etc/tomcat4/policy.d/05jamm.policy` mit dem folgenden Inhalt angelegt:

```

grant codeBase "file:${catalina.home}/webapps/jamm/-" {
  // struts
  permission java.lang.RuntimePermission "accessDeclaredMembers";

  // jamm
  permission java.net.SocketPermission "localhost:389", "connect";
};

```

Anschliessend muss Tomcat neu gestartet werden.

```
/etc/init.d/tomcat4 restart
```

Installation

jamm benötigt einige Libraries, die nicht alle in Debian enthalten sind:

```
cd /path/to/jamm-x.y.z-bin.tar.gz/  
tar xzf jamm-x.y.z-bin.tar.gz  
unzip jamm-x.y.z/jamm-x.y.z.war -d /home/tomcat/webapps/jamm  
cd /home/tomcat/webapps/jamm/WEB-INF  
cp jamm.properties.dist jamm.properties  
nano jamm.properties  
/etc/init.d/tomcat4 restart
```

Wobei `jamm.properties` so aussieht:

```
jamm.ldap.host = localhost  
jamm.ldap.port = 389  
jamm.ldap.search_base = ou=Hosting,o=System  
  
jamm.ldap.root_login = root  
jamm.ldap.root_dn = cn=admin,o=System  
  
jamm.vmail.homedir = /home/vmail/domains
```

Anschliessend ist jamm funktionsbereit und kann unter `http://www.rubbish.ch:8180/jamm/` erreicht werden. Im folgenden Abschnitt wird jamm für ein paar erste Tests gebraucht.

Test 1

Dieser erste Test dient hauptsächlich dafür da, das korrekte Funktionieren von OpenLDAP und jamm sicherzustellen. Es werden Domänen und User angelegt die im zweiten Test dann benötigt werden.

Folgendes sollte gemacht werden und die bisherige Installation zu testen und die notwendigen Voraussetzungen für den zweiten Test zu schaffen:

1. Einloggen

Ich weiss, dieser Punkt hat es in sich ... Der Loginname ist der in `jamm.properties` konfigurierte, also `root`. Das Passwort entspricht dem `admin` Passwort des OpenLDAP Servers.

2. rubbish.ch anlegen

Es soll die neue Domäne `rubbish.ch` angelegt werden. Wenn ein Passwort eingegeben wird, erstellt jamm automatisch den User `<postmaster@rubbish.ch>` der die Berechtigungen für die Verwaltung der Domäne hat.

3. testing@rubbish.ch anlegen

Dies ist ein Testaccount der nur dazu dient, Testmails zu empfangen.

4. test@rubbish.ch anlegen

`<test@rubbish.ch>` ist ein Alias, das auf `<testing@rubbish.ch>` weitergeleitet wird.

5. test@rubbish.ch anlegen

`<test@rubbish.ch>` ist ein Alias, das auf `<testing@rubbish.ch>` weitergeleitet wird.

6. testing2@rubbish.ch anlegen

Dies ist ein zweiter Testaccount, der als 'Catch-All' Account dienen soll.

7. Catch-All konfigurieren

Der Catch-All Account soll alle Mails an `<testing2@rubbish.ch>` weiterleiten.

8. postmaster ausprobieren

<postmaster@rubbish.ch> sollte sich mit dem vorhin definierten Passwort einloggen können und sollte die Berechtigung haben, die User seiner Domäne verwalten zu können. Allerdings scheint es da noch Probleme zu geben, die entweder mit den ACL in `/etc/ldap/slapd.conf` zusammenhängen oder aber es gibt noch Fehler in `jamm`.

9. testing.ch anlegen

Es sollte eine zweite Domäne angelegt werden, die mindestens einen Account enthalten soll. So kann getestet werden, ob Mails zwischen den Domänen hin und hergeschickt werden können.

Wenn das alles funktioniert ist dieser Teil der Installation in Ordnung, jetzt kommen die grösseren Umkonfigurationen dran.

Postfix anpassen

Um Postfix LDAP beizubringen, sind der Postfix-Konfigurationsdatei `/etc/postfix/main.cf` folgende Zeilen anzupassen ...

```
mydestination = intern.rubbish.ch, $myhostname, localhost, localhost.$mydomain
```

(Die 'Hauptdomäne' **rubbish.ch** ist nicht mehr in der Postfix-Konfigurationsdatei drin. Dies ist gewollt da diese Domäne später über LDAP konfiguriert werden soll.) ... respektive hinzuzufügen:

```
domains_server_host = localhost
domains_search_base = ou=Hosting,o=System
domains_query_filter = (&(objectClass=JammVirtualDomain)(jvd=%s) \
    (accountActive=TRUE)(delete=FALSE))
domains_result_attribute = jvd
domains_bind = no
domains_scope = one

aliases_server_host = localhost
aliases_search_base = ou=Hosting,o=System
aliases_query_filter = (&(objectClass=JammMailAlias)(mail=%s) \
    (accountActive=TRUE))
aliases_result_attribute = maildrop
aliases_bind = no

accounts_server_host = localhost
accounts_search_base = ou=Hosting,o=System
accounts_query_filter = (&(objectClass=JammMailAccount)(mail=%s) \
    (accountActive=TRUE)(delete=FALSE))
accounts_result_attribute = mailbox
accounts_bind = no

accountsmmap_server_host = localhost
accountsmmap_search_base = ou=Hosting,o=System
accountsmmap_query_filter = (&(objectClass=JammMailAccount)(mail=%s) \
    (accountActive=TRUE)(delete=FALSE))
accountsmmap_result_attribute = mail
accountsmmap_bind = no

virtual_alias_maps = ldap:accountsmmap, ldap:aliases

virtual_transport = virtual
virtual_mailbox_base = /home/vmail/domains
virtual_mailbox_maps = ldap:accounts
virtual_mailbox_domains = ldap:domains
virtual_minimum_uid = 4000
virtual_uid_maps = static:4000
virtual_gid_maps = static:4000
```

Anschliessend muss noch das Postfix LDAP Backend installiert und das ganze neu gestartet werden:

```
apt-get install postfix-ldap
/etc/init.d/postfix restart
```

Courier anpassen

Im Gegensatz zu Postfix muss bei Courier nur relativ wenig umkonfiguriert werden, allerdings muss auch hier zuerst noch mit dem Paket `courier-ldap` die LDAP-Unterstützung installiert werden.

```
apt-get install courier-ldap
```

Anschliessen muss Courier so konfiguriert werden, dass er für die Authentikation der User zusätzlich zu PAM auch den LDAP Server verwendet. Dazu ist die folgende Zeile in `/etc/courier/authdaemonrc` anzupassen:

```
authmodulelist="authldap authpam"
```

Die genaue Konfiguration des LDAP Servers findet in der Datei `/etc/courier/authldaprc` statt, welche ungefähr so aussehen sollte (Die restlichen Zeilen sind auszukommentieren):

```
LDAP_SERVER          localhost
LDAP_PORT            389
LDAP_PROTOCOL_VERSION 3
LDAP_BASEDN          ou=Hosting,o=System
LDAP_TIMEOUT         5
LDAP_AUTHBIND        1
LDAP_MAIL             mail

LDAP_GLOB_UID         vmail
LDAP_GLOB_GID         vmail
LDAP_MAILDIR          mailbox
LDAP_HOMEDIR          homeDirectory
LDAP_CRYPTPW          userPassword
LDAP_FILTER           (objectClass=JammMailAccount)\
                     (accountActive=TRUE)(delete=FALSE)
```

Anschliessend muss Courier neu gestartet werden:

```
/etc/init.d/courier-authdaemon restart
/etc/init.d/courier-imap restart
```

Squirrelmail

An Squirrelmail muss nichts verändert werden da es von Haus aus mit virtuellen Domänen umgehen kann. Respektive: Allgemein merken IMAP-Clients nicht, ob es sich bei einem Account um einen virtuellen User handelt oder nicht. Der einzige Unterschied ist, dass einige User ein @ in ihrem Namen haben und die anderen nicht.

Test 2

Den im ersten Test angelegten Usern sollte es jetzt möglich sein, sich in Squirrelmail einzuloggen. Falls beim Einloggen die Meldung *EROR: Connection dropped by imap-server* kommt, muss nur schnell ein Mail an den betreffenden User geschickt werden,anschliessend sollte es möglich sein sich einzuloggen. Dies hängt damit zusammen, dass jamm zwar einen Account einrichten kann, der dann theoretisch auch aktiv ist, allerdings wird das Maildir für den User nicht direkt eingerichtet. Wird aber ein Mail an den Account geschickt so sorgt Postfix/Procmail dafür, dass alle notwendigen Ordner und Files erzeugt werden.

Daraus folgt: Wenn ein neuer User angelegt wird, sollte sobald möglich ein Mail an ihn gesendet werden.

Anmerkung zum Testen: Der zweite Test funktioniert nur, wenn entweder die Mails über das Webmail gesendet werden oder ein Mailclient so eingerichtet wird, dass er die Mails über den eingerichteten Mailserver versendet. Wird von einem beliebigen Client aus ein Mail an eine der eingerichteten Domänen verschickt, wird es ziehlich sicher nicht ankommen da das DNS-System nicht entsprechend konfiguriert wurde.

Kapitel 10. vcs

Als Version Control System verwende ich neuerdings Subversion (<http://subversion.tigris.org/>). Aus Kompatibilitätsgründen installiere ich aber weiterhin ein CVS (<http://www.cvshome.org/>).

Subversion

Subversion wird als Nachfolger von cvs bezeichnet. Es hat wesentliche Verbesserungen im Umgang mit Binärdateien, unterstützt Versionierung von Verzeichnissen, etc.

Ein grafischer Client für Windows kann unter <http://tortoissvn.tigris.org/> heruntergeladen werden.

Installation

```
1: apt-get install subversion
2: apt-get install subversion-tools
3: apt-get install libapache2-svn
```

1. Das eigentliche Subversion
2. Hilfstools z.B. zum Backupen eines Repositories
3. Subversion Modul für Apache2

Apache anpassen

Es gibt verschiedene Varianten, auf Subversion-Repository's zuzugreifen. Wie bei cvs ist es möglich, einen ssh-Tunnel zu öffnen und mit Subversion darüber zu arbeiten. Dies würde aber bedeuten, dass alle Subversion-User auf dem Server einen SSH-Account benötigen würden. Aus Sicherheitsgründen sollten jedoch solche Accounts nur vergeben werden, wenn dies unvermeidlich ist. Im diesem Falle gibt es die einfache Möglichkeit, Subversion über Apache und webdav zu benutzen. Dies bedeutet einfach, dass alle Repository's in der Apache Konfiguration eingetragen werden müssen. Die Authentifikation wird dabei über die bekannten Methoden durchgeführt.

```
<Location /demo>
  DAV svn
  SVNPath /home/projects/subversion/demo

  AuthName "Demo Repository"
  AuthType Basic

  AuthLDAPURL ldap://localhost:389/ou=People,o=System?uid
  AuthLDAPGroupAttribute memberUid
  AuthLDAPGroupAttributeIsDN off
  require group cn=svn-test,ou=Group,o=System
</Location>
```

Anschließend muss Apache neu gestartet werden (**/etc/init.d/apache2 restart**). Weitere Konfigurationsmöglichkeiten sind dem Subversion Handbuch zu entnehmen, welches unter <http://svnbook.red-bean.com/> abrufbar ist.

Testen

Das Testen von Subversion sollte nach folgendem Schema geschehen:

1. Test-Repository anlegen (auf dem Server):

```
svnadmin create /path/to/repository
chown -R www-data:www-data /path/to/repository
```

2. Ein paar Testdaten erfassen:

```
cd /root
mkdir tmp
cd tmp
```

```
touch dummy.txt
cd ..
svn import tmp file:///path/to/repository -m "initial import"
```

3. Die importierten Daten sollten natürlich auch wieder ausgecheckt werden können:

```
cd /root
mkdir tmp2
svn checkout file:///path/to/repository/ tmp2
```

CVS

Teil II. Linux Tips und Tricks

hier kommt die beschreibung von linux tips und tricks hin

Kapitel 11. Arbeiten mit Linux

.bashrc & .bash_profile

~/ .bashrc ist wie schon aus dem Namen hervorgeht das Konfigurationsfile für die BASH. Über diese Datei kann deren Verhaltensweise beeinflusst werden, zum Beispiel können die folgenden Dinge umkonfiguriert werden:

Bash-Prompt einfärben

Wenn mehrere SSH-Verbindungen zu verschiedenen Rechnern geöffnet sind, kann es leicht vorkommen, dass die Übersicht verloren geht und Kommandos auf der falschen Shell eingetippt werden. Diese Gefahr kann drastisch minimiert werden, wenn die Rechner farblich 'codiert' werden. Zum Beispiel können die root-Shells der Server rot eingefärbt werden, diejenigen der normalen User werden blau, etc. Für die blaue Variante sieht die .bashrc folgendermassen aus:

```
red='\e[0;31m'  
RED='\e[1;31m'  
blue='\e[0;34m'  
BLUE='\e[1;34m'  
cyan='\e[0;36m'  
CYAN='\e[1;36m'  
NC='\e[0m' # No Color  
PS1="[$blue\u@\h$NC \W]\\$ "  
  
export PS1
```

Beim Superuser sollten diese Einträge wie erwähnt in .bashrc, bei normalen Usern in .bash_profile gemacht werden. Dies aus dem Grund da das jeweils andere File je nachdem nicht automatisch berücksichtigt wird.

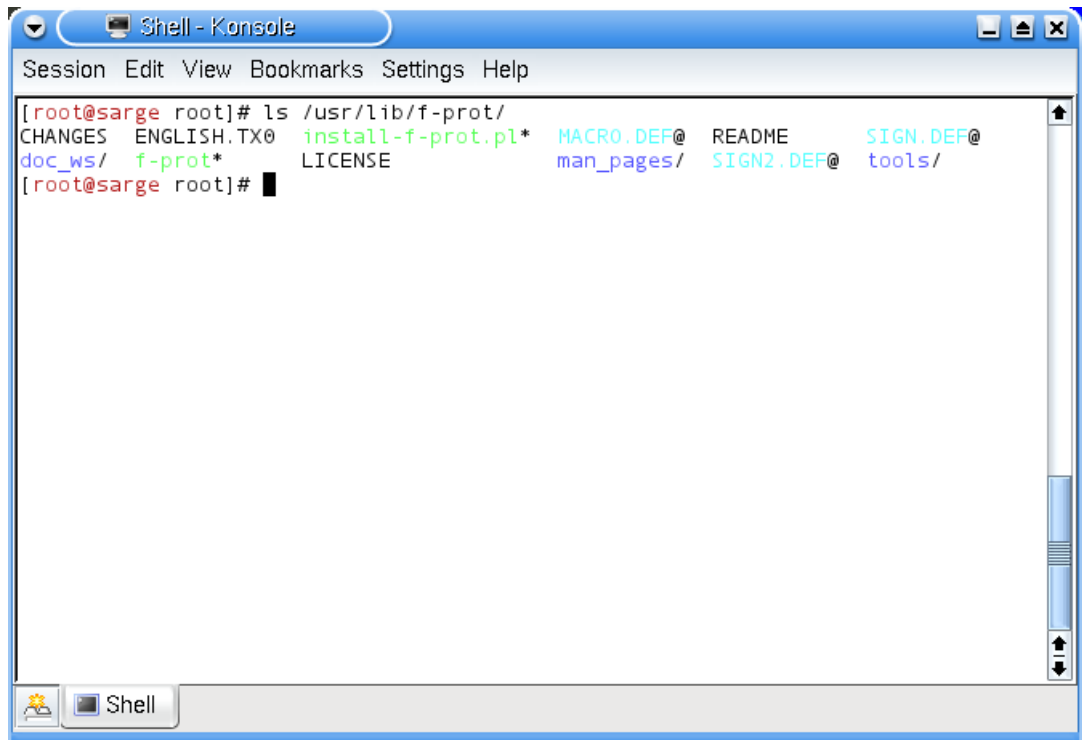
Ausgabe von ls einfärben

Wenn die Ausgabe von ls eingefärbt werden soll, so kann dies durch das Eintragen der folgenden Zeile in .bashrc respektive .bash_prompt erreicht werden:

```
alias ls='ls -F --color'
```

Anwendungsbeispiel

In der Praxis sieht das dann so aus:



```
[root@sarge root]# ls /usr/lib/f-prot/  
CHANGES  ENGLISH.TX0  install-f-prot.pl*  MACRO.DEF@  README  SIGN.DEF@  
doc_ws/   f-prot*      LICENSE             man_pages/  SIGN2.DEF@  tools/
```

The screenshot shows a terminal window with a blue title bar and a menu bar containing 'Session', 'Edit', 'View', 'Bookmarks', 'Settings', and 'Help'. The terminal content shows a root user at a machine named 'sarge' executing the 'ls' command on the directory '/usr/lib/f-prot/'. The output lists various files and subdirectories, including 'CHANGES', 'ENGLISH.TX0', 'install-f-prot.pl*', 'MACRO.DEF@', 'README', 'SIGN.DEF@', 'doc_ws/', 'f-prot*', 'LICENSE', 'man_pages/', 'SIGN2.DEF@', and 'tools/'. The prompt returns to the root user.

Kapitel 12. Security

Authentifikation über private SSH Keys

Das Verwenden von privaten SSH Schlüsseln ermöglicht es, die Sicherheit von SSH weiter zu steigern. Dies aus dem Grund, da dann nie mehr ein Login-Passwort über das Netzwerk übermittelt wird, und sei es auch in verschlüsseltem Zustand. Das Verfahren beruht darauf, dass mit einem Public Key eindeutig gesagt werden kann, ob eine Nachricht von einem bestimmten Privat Key signiert wurde oder nicht. Wenn also der Public Key auf dem Server liegt, und sich ein User per Privat Key authentifizieren will, kann der Server über den Public Key feststellen, wer Zugang will und ob derjenige berechtigt ist oder nicht.

Schlüsselerzeugung

Bevor sich ein User mit einem Key authentifizieren kann, muss er diesen natürlich zuerst erzeugen:

```
ssh-keygen -t dsa
```

ssh-keygen erzeugt ein Schlüsselpaar, das aus einem privaten und einem öffentlichen Schlüssel besteht. Der öffentliche Schlüssel muss auf den Server kopiert werden, beim privaten sollte sichergestellt werden, dass er nicht in falsche Hände gerät.

Schlüssel Installieren

Nachdem das Schlüsselpaar erzeugt wurde, muss jetzt der öffentliche Schlüssel auf jeden Server kopiert werden, auf den der Besitzer des Schlüssels zugriff haben soll.

`~/.ssh/authorized_keys` enthält alle öffentlichen Schlüssel, mit denen sich ein User authentifizieren kann:

```
mkdir /home/testing/.ssh  
cat /path/to/key.pub >> /home/testing/.ssh/authorized_keys
```

Bei diesen Kommandos wurde davon ausgegangen, dass der öffentliche Schlüssel bereits auf dem Server liegt. Per SSH kann eine Datei folgendermassen über das Netzwerk kopiert werden:

```
scp testing.pub testing@sarge.rubbish.ch:/home/testing/
```

Abschluss

Wenn sich der User jetzt auf dem Server einloggen will, sollte er nach dem Passwort von seinem Privaten SSH Schlüssel gefragt werden. Nach dessen Eingabe sollte ihm automatisch der Einlass in das System gewährt werden.

Portforwarding mit SSH

Es kommt vor, dass man per SSH auf einen Server zugreifen kann, aber eine anderer Service, den der Server eigentlich anbieten würde, durch eine Firewall gesperrt ist (zum Beispiel IMAP). In einem solchen Fall kann man sich mit der Portforwarding Funktionalität von SSH einen Tunnel zum Zielport erstellen lassen und trotzdem auf den eigentlich gesperrten Service zugreifen.

```
ssh testing@sarge.rubbish.ch -L 10143:localhost:143
```

Der lokale Port 10143 wird auf den Port 143 (IMAP) des entfernten Servers weitergeleitet. Das *localhost* ist aus der Sicht des entfernten Servers anzuschauen. Der lokale und entfernte Port muss wie im Beispiel ersichtlich nicht übereinstimmen, es kann ein beliebiger Port auf einen beliebigen Port gemappt werden. Root darf einen Tunnel nach einem beliebigen lokalen Port graben, andere User haben nur das Recht, lokale Ports mit einer Nummer grösser als 1023 zu verwenden.

```
Privileged ports can only be forwarded by root.
```

Diese Fehlermeldung deutet darauf hin, dass ein normaler User versucht einen Tunnel auf einen lokalen Port mit einer Nummer kleiner 1024 zu erstellen.

Portforwarding verhindern

Aus Sicherheitsgründen kann es Sinn machen, einzelnen Usern das Verwenden von Portforwarding zu verbieten. (Denn mit Portforwarding wird eigentlich die Firewall komplett umgangen, was nicht jedem gestattet sein soll.) Dies kann erreicht werden, wenn der Eintrag *no-port-forwarding* in der Datei *authorized_keys* gemacht wird:

```
cd /home/testing/.ssh
less authorized keys

no-port-forwarding ssh-dss AAAAB3NzaC1kc3MAA...
```

SSL Zertifikate & Apache

Der Sicherheit willen sollte für Dinge wie z.B. ein Webmail das verschlüsselte Protokoll *https* dem *http* vorgezogen werden. Dieser Abschnitt behandelt das Einrichten von *https* und der entsprechenden *>pache* Konfiguration.

Die nachfolgende Anleitung ist grösstenteils von http://httpd.apache.org/docs-2.0/ssl/ssl_faq.html übernommen. Wer also das Englisch nicht fürchtet, findet dort das vielleicht aktuellere Original.

Certificate Authority einrichten

1. Für die Certificate Authority (nachfolgend CA genannt) muss ein Key erzeugt werden. Hier muss ein Passwort mit mindestens vier Stellen gesetzt werden.

```
mkdir /root/ca
cd /root/ca
openssl genrsa -des3 -out ca.key 1024
```

2. Vom erstellten Schlüssel wird jetzt ein selbst signiertes Zertifikat erstellt:

```
openssl req -new -x509 -days 10000 -key ca.key -out ca.crt
```

Mit **openssl x509 -noout -text -in ca.crt** können die Details zum erstellten Zertifikat angeschaut werden.

3. Von den erstellten Dateien sollte unbedingt ein Backup erstellt werden, ausserdem wäre es noch ganz praktisch, das Passwort auch in einem Jahr noch zu wissen.

sign.sh

Eigentlich sollte Debian ein Script namens *sign.sh* enthalten. Dieses habe ich aber nicht gefunden, deshalb hier eine Version aus dem Internet:

```
#!/bin/sh
##
## sign.sh -- Sign a SSL Certificate Request (CSR)
## Copyright (c) 1998-2001 Ralf S. Engelschall, All Rights Reserved.
##
# argument line handling
CSR=$1
if [ $# -ne 1 ]; then
    echo "Usage: sign.sign <whatever>.csr"; exit 1
fi
if [ ! -f $CSR ]; then
    echo "CSR not found: $CSR"; exit 1
fi
case $CSR in
    *.csr ) CERT=`echo $CSR | sed -e 's/\.csr/.crt/'` ;;
    * ) CERT="$CSR.crt" ;;
esac

# make sure environment exists
if [ ! -d ca.db.certs ]; then
    mkdir ca.db.certs
fi
```

```

if [ ! -f ca.db.serial ]; then
    echo '01' >ca.db.serial
fi
if [ ! -f ca.db.index ]; then
    cp /dev/null ca.db.index
fi

# create an own SSLeay config
cat >ca.config <<EOT
[ ca ]
default_ca          = CA_own
[ CA_own ]
dir                 = .
certs               = \${dir}
new_certs_dir       = \${dir}/ca.db.certs
database            = \${dir}/ca.db.index
serial              = \${dir}/ca.db.serial
RANDFILE            = \${dir}/ca.db.rand
certificate         = \${dir}/ca.crt
private_key         = \${dir}/ca.key
default_days        = 2002
default_crl_days    = 30
default_md          = md5
preserve            = no
policy              = policy_anything
[ policy_anything ]
countryName         = optional
stateOrProvinceName = optional
localityName        = optional
organizationName    = optional
organizationalUnitName = optional
commonName          = supplied
emailAddress         = optional
EOT

# sign the certificate
echo "CA signing: $CSR -> $CERT:"
openssl ca -config ca.config -out $CERT -infiles $CSR
echo "CA verifying: $CERT <-> CA cert"
openssl verify -CAfile ca.crt $CERT

# cleanup after SSLeay
rm -f ca.config
rm -f ca.db.serial.old
rm -f ca.db.index.old

# die gracefully
exit 0

```

Am einfachsten funktioniert das ganze, wenn `sign.sh` in das gleiche Verzeichnis kopiert wird, das auch `ca.key` enthält.

Zertifikate für Apache erstellen

1. Nachdem eine CA eingerichtet ist, muss zuerst ein Schlüssel für den Server erstellt werden:

```
openssl genrsa -des3 -out server.key 1024
```

2. Jetzt muss ein **Certificate Signing Request** (anschliessend nur noch CSR genannt) erstellt werden:

```
openssl req -new -key server.key -out www.rubbish.ch.csr
```

Es ist unbedingt sicherzustellen dass die richtige URL angegeben wird wenn OpenSSL nach dem **CommonName** fragt. Falls der Webserver später unter `https://www.rubbish.ch/` erreichbar sein soll, muss als CommonName **www.rubbish.ch** angegeben werden.

3. Der erstellte CSR muss jetzt noch von einer CA zertifiziert werden. (Entweder von der eigenen oder von einer offiziellen.)

```
./sign.sh www.rubbish.ch.csr
```

Das Ergebnis dieser Zertifizierung ist die Datei `www.rubbish.ch.crt` welche wie erwartet das endgültige Zertifikat enthält. Das `www.rubbish.ch.csr` wird nicht mehr benötigt und kann gelöscht werden.

4. Dass ein Key ein Passwort hat, ist eigentlich eine gute Sache. Hier stört es aber, da der Key bei jedem Serverstart entsperrt werden müsste. Folgendermassen wird man das Passwort los:

```
cp server.key server.key.org  
openssl rsa -in server.key.org -out server.key
```

5. Das erstellte Zertifikat sollte jetzt zusammen mit dem Server-Key in den Ordner /etc/apache2/ssl kopiert werden.

```
cp server.key www.rubbish.ch.crt /etc/apache2/ssl
```

Kapitel 13. System

Magic SysRQ

Auch wenn Linux sehr stabil ist, können trotzdem Abstürze oder Kernel-Panics auftreten. In einem solchen Fall den Resetknopf drücken kann zu Datenverlust führen. Dies zu verhindern ist das Ziel dieses Abschnittes. Magic SysRQ sind 'magische' Tastenkombinationen, auf die der Kernel antwortet, egal was er sonst gerade am machen ist. (Ausser natürlich wenn er sich total aufgehängt hat.)

Um Magic SysRQ verwenden zu können, muss es in den Kernel kompiliert sein. Die entsprechende Option ist unter *Kernelhacking - Magic SysRQ* zu finden. Wenn Sicherheitsüberlegungen das deaktivieren von Magic SysRQ erforderlich machen, so kann dies mit **echo "0" > /proc/sys/kernel/sysrq** erreicht werden.

Benützung von Magic SysRQ

Durch das gleichzeitige Drücken von **ALT-SysRQ-Kommando** wird der Magic SysRQ Befehl „Kommando“ ausgeführt. Auf einem x86'er entspricht „SysRQ“ wenn vorhanden *SysRQ* oder sonst der *Print Screen* Taste.

Vorhandene Kommandos ¹

- b* Rebooted das System unmittelbar. Achtung: Die Dateisysteme werden weder unmounted noch synchronisiert!!!
- e* Sendet ein SIGTERM an alle Prozesse ausser init. (Programme haben die Möglichkeit, sich zu beenden.)
- i* Sendet ein SIGKILL an alle Prozesse ausser init. (Prozesse werden beendet, entspricht einem **kill -9 <pid>**.)
- k* Killt alle Programme, die auf der aktuellen (virtuellen) Konsole laufen. Damit kann man zum Beispiel sicherstellen, dass man wirklich das echte Login vor sich hat und nicht sein Passwort einem Trojaner verrät.
- r* Stellt den 'raw-mode' der Tastatur ab und setzt sie in den 'XLATE' Modus. Nimmt zum Beispiel einem abgestürzten X-Server die Kontrolle über Maus und Tastatur weg und ermöglicht ein Wechseln in die Konsole.
- s* Versucht, alle eingehängten (gemounteten) Dateisysteme zu synchronisieren.
- u* Versucht, alle gemounteten Dateisysteme als read-only zu mounten.

System rebooten

Nach einem Absturz kann das System durch folgende Magic SysRQ Befehle sauber neu gestartet werden:

1. **ALT-SysRQ-e** Sendet ein SIGTERM an alle Prozesse
2. **ALT-SysRQ-u** Alle Dateisysteme werden read-only remounted
3. **ALT-SysRQ-i** Sendet ein SIGKILL an alle Prozesse
4. **ALT-SysRQ-b** System rebooten

Screen

Screen ist ein Windowmanager, der über ein physikalisches Terminal (zum Beispiel eine

¹ Dies sind nur die wichtigsten Kommandos. Die vollständige Referenz ist unter `/usr/src/linux/Documentation/sysrq.txt` zu finden.

SSH-Verbindung) mehrere virtuelle Terminals simulieren kann. Diese virtuellen Terminals können zwischengespeichert und später wieder geöffnet werden.

Das erlaubt es zum Beispiel, über eine SSH-Verbindung einzuloggen, screen zu starten, einen lange dauernden Installationsvorgang zu starten und anschliessend die Verbindung zu schliessen. Zu einem beliebigen späteren Zeitpunkt kann diese Verbindung wieder geöffnet werden um weiterzuarbeiten.

Vorhandene Kommandos ²

C-aC-c	(Create) Erstellt ein neues Fenster
C-aC-0	Wechselt zum ersten Fenster
C-aC-n	Wechselt zum n-ten Fenster
C-aK	(Kill) Schliesst das aktuelle Fenster
C-aC-d	(Detach) Schliesst Screen (Läuft aber im Hintergrund weiter)
C-aC-x	Sperrt alle Konsolen und zeigt einen Passwort-Prompt an

Falls eine geschlossene (detached) Session wieder geöffnet werden soll, ist folgendermassen vorzugehen:

screen -ls	zeigt die Liste der vorhandenen Screen-Sessions an
screen -r	(Reattach) öffnet die erste Session dieser Liste
screen -r <id>	Session mit der angegebenen id wieder öffnen
screen -RR	Öffnet auf jeden Fall eine Screen Sitzung. Wenn möglich wird eine existierende hervorgeholt, ansonsten eine neue erzeugt.

² Dies sind nur die wichtigsten Kommandos. Als vollständige Referenz ist die Manpage von screen respektive *screen -help* zu verwenden.

Anhang A. Das LDAP Layout

Ich verwende das anschliessend dargestellte Layout aus folgenden Gründen:

- Es ist relativ simpel, einfach verständlich und logisch was wo hingehört.
- Der Domainname des Netzwerks, in dem der Server steht, ist in der *basedn* nicht enthalten, das vereinfacht das Kopieren von Daten zwischen verschiedenen OpenLDAP Servern erheblich. (Zum Beispiel das Verschieben oder Kopieren einer virtuellen Maildomäne von einem OpenLDAP Server zum anderen.)

Verzeichnisbaum

```
1. o=System
   |
   | 2. \-ou=People
   |     \-uid=chris
   |        \-uid=test
   |
   | 3. \-ou=Group
   |     \-cn=chris
   |        \-cn=test
   |
   | 4. \-ou=Addressbooks
   |     |
   |     | 5. \-ou=chris
   |     |     \-cn=Christian Cavegn
   |     |        \-cn=Hans Müller
   |     |
   |     | 6. \-ou=user@rubbish.ch
   |     |     \-cn=Tschau Sepp
   |     |
   |     | 7. \-ou=Hosting
   |     |     |
   |     |     | 8. \-jvd=rubbish.ch
   |     |     |     \-mail=user@rubbish.ch
   |     |     |        \-mail=alias@rubbish.ch
   |     |     |
   |     |     | 9. \-jvd=testing.ch
   |     |     |     \-mail=test@testing.ch
   |     |     |
   |     |     | 10. \-ou=dhcp
   |     |     |     |
   |     |     |     | 11. \-cn=sarge
   |     |     |     |     \-cn=DHCP Config
   |     |     |     |
   |     |     |     | 12. \-ou=dns
   |     |     |     |     |
   |     |     |     |     | 13. \-dc=ch
   |     |     |     |     |     \-dc=rubbish
   |     |     |     |     |         \-cn=sarge
   |     |     |     |     |            \-cn=sarge-aliases
   |     |     |     |     |
   |     |     |     |     | 14. \-dc=ch
   |     |     |     |     |     \-dc=testing
   |     |     |     |     |        \-cn=www
   |     |     |     |     |
   |     |     |     |     | 15. \-sambaDomainName=SARGE
```

1. Root des Verzeichnisbaumes
2. Speicherort für die UNIX-User (also System-User)
3. Gleich wie ou=People, einfach für die Gruppen
4. Subtree für die Adressbücher sowohl der System- als auch der vmail-User. Jeder Eintrag unterhalb von ou=Addressbooks enthält das persönliche Adressbuch eines Users.
5. Subtree für die Virtuellen Maildomänen
6. Ein Beispiel für eine virtuelle Domäne. Enthält wiederum die entsprechenden User und Aliase
7. Subtree für die DHCP Informationen. Enthält einerseits einen Eintrag mit Informationen über den DHCP-Server selber, andererseits ist pro Subnet das verwaltet werden soll wiederum ein Eintrag vorhanden.
8. Subtree für alle dns-Informationen

9. Eine Domäne, die vom Server effektiv verwaltet wird. Enthält wiederum pro DNS-Eintrag oder Aliase darauf einen separaten Eintrag.

Der Eintrag *sambaDomainName* wird wie aus dem Name hervorgeht von **samba** benötigt und ist an dieser Stelle nicht weiter von Bedeutung.

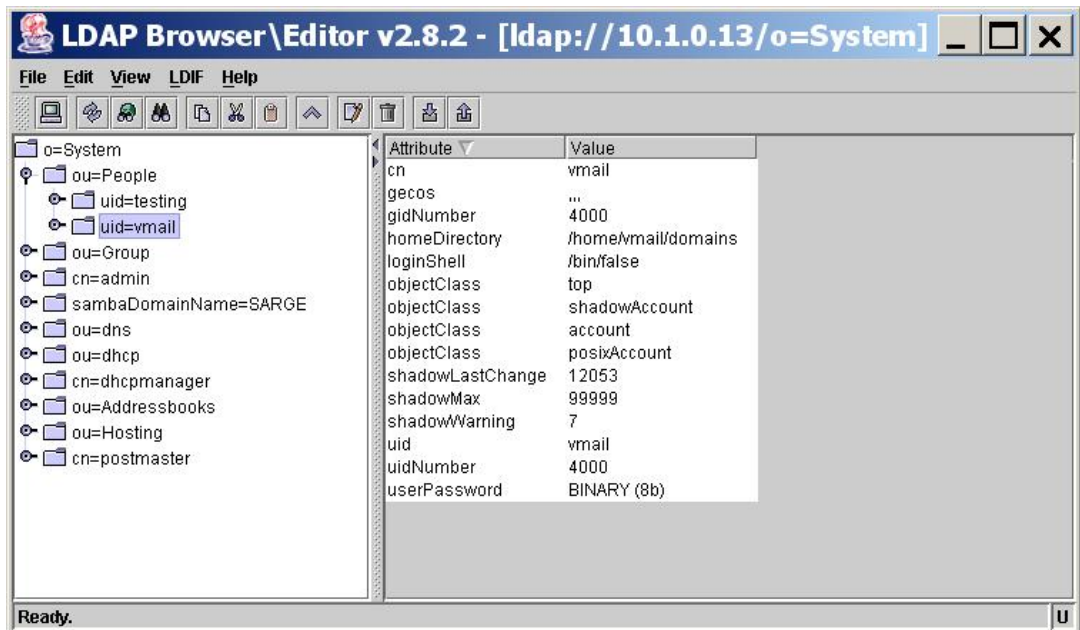
Anhang B. Benötigte Ports

Jeder der installierten Server ist über (mindestens) einen Port erreichbar. Die folgende Liste zeigt die Ports an, die auf meinem System nach der Installation geöffnet sind. Es ist gut möglich, dass diese Liste nicht abschliessend ist, sie soll nur als Referenz verwendet werden.

PortNr	Protokoll	Dienst	Beschreibung	
22	tcp	sshd	OpenSSH Server	
25	tcp	postfix	Postfix SMTP Mailserver	
53	tcp	udp	pdns	PowerDNS Nameserver
67	udp	dhcpd	Der ICS dhcp Server	
80	tcp	apache2	Apache2 Webserver	
137	udp	samba	Samba Fileserver	
138	udp	samba	Samba Fileserver	
139	tcp	samba	Samba Fileserver	
143	tcp	courier	Courier IMAP Mailserver	
389	tcp	slapd	OpenLDAP Verzeichnisdienst	
443	tcp	apache2	Apache2 Webserver, SSL	
445	tcp	samba	Samba Fileserver	
3306	tcp	mysql	MySQL Datenbank	
8005	tcp	tomcat4	Tomcat Servletcontainer	
8009	tcp	tomcat4	Tomcat Servletcontainer	
8180	tcp	tomcat4	Tomcat Servletcontainer	
10024	tcp	amavis	AMaViS	
10025	tcp	postfix	Postfix, Pfad für AMaViS	

Die Files `/etc/services` und `/etc/protocols` enthalten eine komplette Übersicht über die *well known ports*.

Anhang C. LDAP Browser



Unter <http://www.iit.edu/~gawojar/ldap/> kann der LDAP Browser heruntergeladen werden.

LDAP Browser ist das Programm meiner Wahl, wenn es um die lowlevel Verwaltung eines LDAP Servers geht. LDAP Browser erlaubt es, durch die Baumstruktur von OpenLDAP zu navigieren wie der Konqueror/Explorer durch eine lokales Dateisystem. Mit ihm können Einträge und Attribute erstellt, modifiziert und gelöscht werden, ausserdem können Passwörter auf verschiedene Arten (crypt, SHA, ...) gehasht und verifiziert werden.

Ein weiterer Vorteil von LDAP Browser ist die integrierte Import- und Exportfunktion. Damit lassen sich (Sub)Trees im .ldif-Format verarbeiten. Natürlich ist das auch mit Kommandozeilentools möglich, aber per Drag'n'Drop ist es einfach bequemer.

Mit anderen Worten, die Verwaltung eines LDAP Servers macht ohne LDAP Browser nur halb so viel Spass ...