Linux Servertechnik

Christoph Ketelsen
Florian Vogler
Nico Hänsel

7. Februar 2006

Inhaltsverzeichnis

1 Client - Server - Grundlagen

1.1 Client

Ein Client nimmt Dienste in Anspruch, deshalb wird eine an den Server angeschlossene Arbeitsstation als Client bezeichnet. Der Client schickt Anfragen des Benutzers in einem speziellen Protokoll an den Server und stellt dessen Antworten in lesbarer Weise auf dem Bildschirm dar.

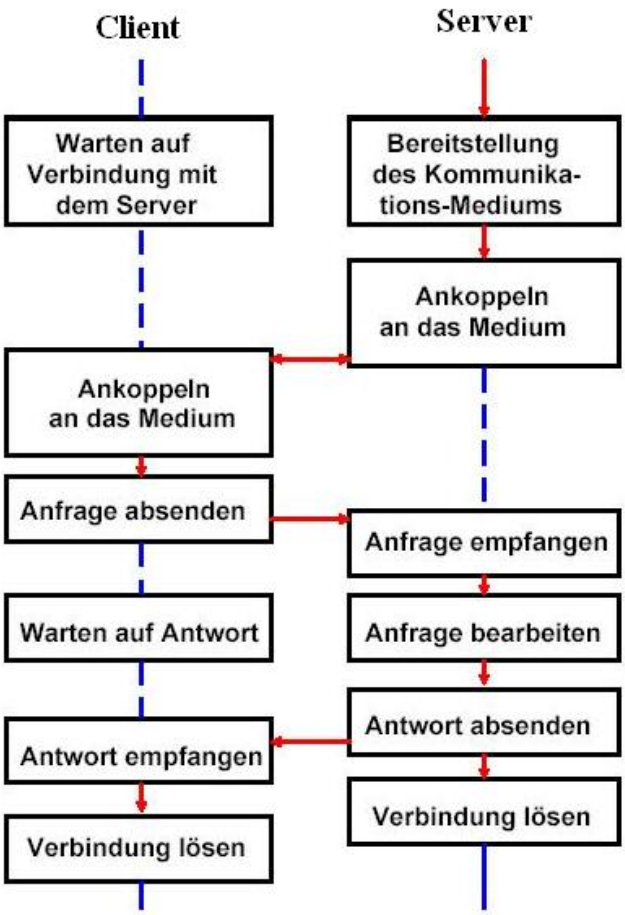
1.2 Server

Ein Server ist ein zentraler Rechner in einem Netzwerk, der den Arbeitsstationen oder Clients Daten, Speicher und Ressourcen zur Verfügung stellt. Auf dem Server ist das Netzwerk-Betriebssystem installiert, und vom Server wird das Netzwerk verwaltet. Im WWW sind Server Knotenpunkte des Netzes.

Ein Server kann aus einem Rechner mit zugehörigem Betriebssystem und einem Dienstprogramm bestehen. Gleichermaßen kann aber auch nur ein Programm gemeint sein, das einen bestimmten Dienst wie einen Domain-Name-Service (DNS) oder Web-Service bereitstellt. Aber diese Dienstprogramme sollen hier keine Rolle spielen, sind sie doch im Endeffekt nur als Anwendung agieren. Vielmehr gilt es, die unterschiedlichen Server-Klassen und die spezifischen Anforderungen zu beleuchten. „Server-Klasse“ bezieht sich in diesem Kontext nicht auf die Größe des Servers - Workgroup- oder Enterprise-Server - sondern ausschließlich auf die Aufgabengebiete, die abzudecken sind. Bei einer Klassifizierung nach dem Einsatzbereich ergeben sich sechs verschiedene Server-Klassen mit jeweils eigenem Anforderungsprofil:

- Ein **File-Server** stellt seinen Clients Dateien und Platz auf dem Dateisystem bereit. Zusätzlich übernimmt er die Sicherung der Benutzerdateien.
- Ein **Application-Server** ermöglicht den Anwendern den Zugriff auf ein oder mehrere Anwendungsprogramme.
- Auf einem **Datenbank-Server** läuft eine mehr oder weniger große Datenbank. Die Aufgabe des Servers ist die Verwaltung und Organisation der Daten, die schnelle Suche, das Einfügen und das Sortieren von Datensätzen.
- Ein **Compute-Server** bietet möglichst viel Rechenleistung. Typische Beispiele für Compute-Server sind Supercomputer à la Cray in Kernforschungsanstalten.
- Ein **Internet-Server** stellt Internet- und Intranet-Dienste bereit. Typische Dienste umfassen das World Wide Web, den Domain Name-Service, FTP sowie E-Mail.
- **(Streaming) Media-Server** stellen Multimedia-Daten (z.B. Audio- und Video-Clips) in Echtzeit und höchster Dienstqualität zur Verfügung.

1.3 Kommunikationsmodell



2 Apache

2.1 Geschichte

Der Apache - Webserver ist seit 1997 der bei weitem am häufigsten eingesetzte Webserver. Laut der Netcraft - Webserverstatistik war im August 2002 auf über 60 Prozent aller betrachteten Webserver ein Apache -Webserver im Einsatz.

Der Apache -Webserver entstand 1995 aus dem bis dahin meistgenutzten Webserver, dem NCSA httpd, der am National Center for Supercomputing Applications der University of Illinois entwickelt worden war. Da der bisherige Entwickler, Rob McCool, das NCSA verlassen hatte, war die Entwicklung ins Stocken geraten. Eine Gruppe von Webmastern fand sich zusammen, um den NCSA httpd weiter zu entwickeln. Da die Weiterentwicklung zunächst in der Form von Patches und Ergänzungen zum NCSA httpd erfolgte, bekam das Produkt den Namen Apache, von „A patchy server“.

Ende 1995 wurde die Version 1.0 des Apache - Webservers veröffentlicht.

Nach einer längeren Betatestphase für die Version 2, die sich seit ungefähr 1998 in der Entwicklung befand, wurde im April 2002 mit der Version 2.0.35 die erste „Produktionsversion“ (beim Apache -Webserver General Availability Release genannt) freigegeben.

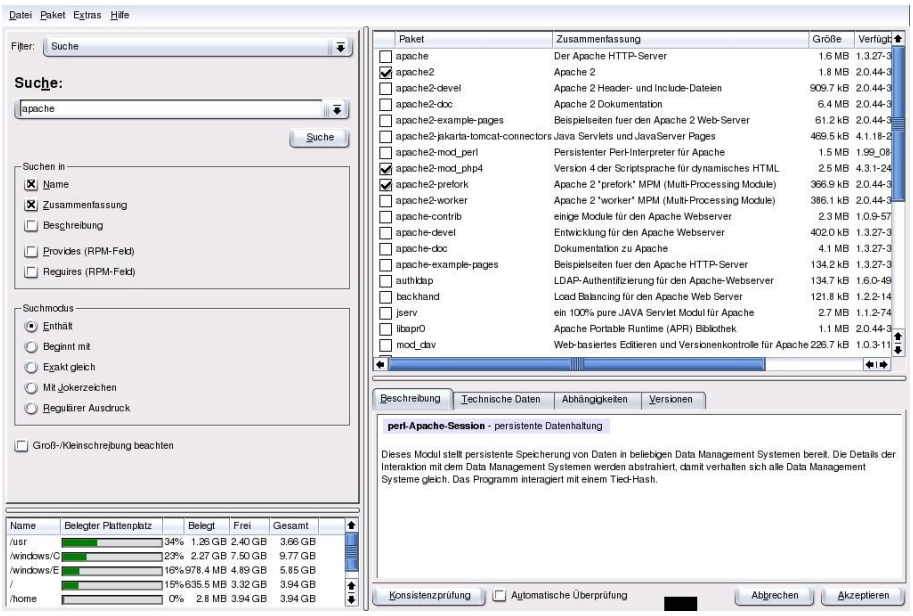
In der neuen Version des Apache - Webservers hat sich vor allem an der Architektur des Apache - Kerns einiges geändert. Bei der Entwicklung der neuen Version hatten die Autoren das Ziel, die Portierung auf neue Plattformen einfacher zu gestalten, und entwarfen eine modulare Architektur, in der die Apache Portable Runtime (APR) eine Abstraktionsschicht zwischen dem unterliegenden Betriebssystem und dem Apache 2.0 darstellt. Die APR stellt für die eigentlichen Apache - Module gewissermaßen ein virtuelles Betriebssystem dar, verwendet aber so weit wie möglich native Betriebssystemaufrufe, um eine bestmögliche Performance zu erzielen.

2.2 Allgemeines

- plattformunabhängiger Open - Source HTTP - Server
- meistgenutzter HTTP - Server weltweit
- sehr umfangreich konfigurierbar und durch Module erweiterbar
- Features:
  - DBM Datenbanken zur Authentifizierung
  - Bei Fehlern und Problemen konfigurierbare Reaktionen des Servers
  - Virtuelle Hosts
  - Konfigurierbares Logging

2.3 Installation

- YAST2 starten
- Software installieren oder löschen anklicken
- Filter: Suche
- Nach „apache“ suchen
- Apache2 , Apache2-Prefork und Apache2_mod_php4 - Modul auswählen
- Akzeptieren anklicken



2.4 Konfiguration

Der Apache WebServer ist nach der Installation sofort lauffähig. Als Root-Verzeichnis des WebServers ist (/srv/www/htdocs) voreingestellt. Auf dieses Verzeichnis greift der Server zu, wenn Anfragen an ihn gestellt werden. Damit der WebServer auch PHP Dateien verarbeiten kann, müssen nun noch zwei kleine Ergänzungen in der Konfigurationsdatei vorgenommen werden. Hierzu geben wir als root folgenden Befehl ein:

```
vi /etc/apache2/httpd.conf
```

Hier fügen wir nun zum einen die PHP - Typen ein:

```
AddType application/x-httpd-php .php .phtml .php3 .php4
```

Und zum anderen das PHP - Modul an sich:

```
LoadModule php4_module /usr/lib/apache2-prefork/libphp4.so
```

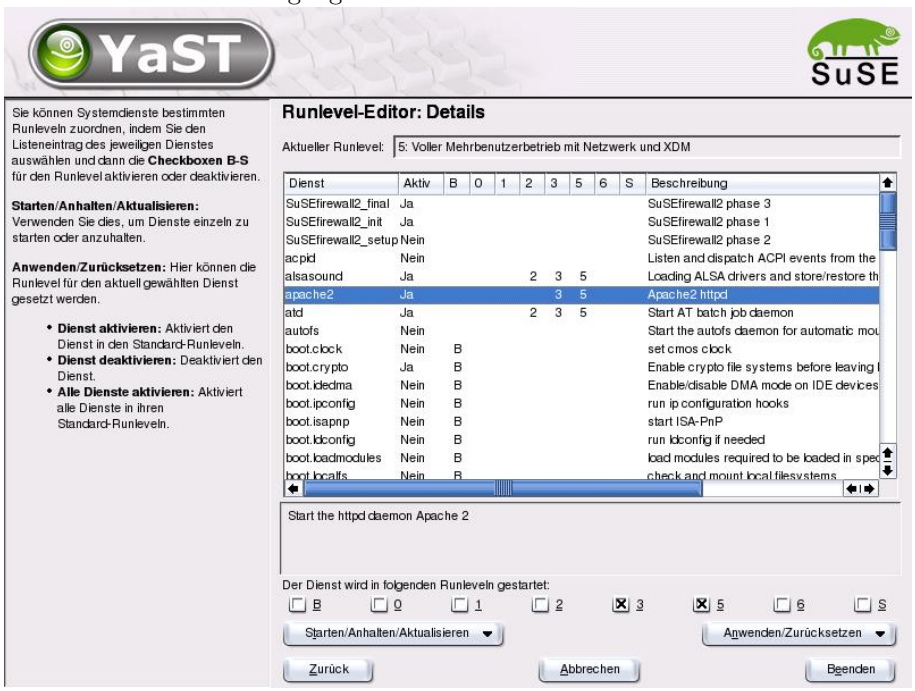
Nachdem dies erledigt ist, speichern wir die Konfigurationsdatei und kehren zur shell zurück. Nun muss der HTTP - Server gestartett werden, damit die von uns vorgenommenen Änderungen wirksam werden. Hierzu geben wir folgenden Befehl ein:

```
/etc/rc.d/apache2 start
```

Nach jeder Änderung, die an der Konfigurationsdatei vorgenommen wurde, muss der Server neugestartet werden. Die geschieht mit folgendem Befehl:

```
/etc/rc.d/apache2 restart
```

Was nun noch folgt, ist die Einbindung des Dienstes in die Runlevel. Dies geschieht mit YAST. Wir binden hier den apache2 - Dienst in die Runlevel 3 und 5 ein. Wenn der Rechner nun neu gestartet wird, erfolgt der Start des apache2 - Dienstes beim Bootvorgang automatisch.



2.5 Erstellen eines passwortgeschützten Bereichs

Um ein Verzeichnis in der Webstruktur mit einem Passwort zu schützen, sind 2 Dateien notwendig. Zum einen ist das die .htaccess - Datei, die in jedem Verzeichnis vorhanden sein muss, dessen Zugang vorher autorisiert werden muss und zum anderen die .htpasswd - Datei, die nur einmal vorhanden sein muss und in der die Usernamen und Passwörter der berechtigten Nutzer gespeichert werden.

Damit der HTTP - Server die Authentifizierung unterstützt, muss in der httpd.conf noch eine kleine Veränderung vorgenommen werden. Dazu öffnen wir die Konfigurationsdatei wieder:

```
vi /etc/apache2/httpd.conf
```

In der Datei ändern wir nun folgenden Eintrag von None zu AuthConfig, um die Unterstützung zu aktivieren:

```
AllowOverride AuthConfig
```

Wenn das erledigt ist, muss der Apache - Servers neu gestartet werden:

```
/etc/rc.d/apache2 restart
```

Um ein Verzeichnis mit einem Passwortschutz zu versehen müssen 2 Dateien vorhanden sein. Als nächstes muss die Datei .htpasswd erstellt werden. Apache stellt hierzu das Tool htpasswd2 zur Verfügung. Mit folgendem Befehl wird die User/Passwort - Datei erstellt:

```
htpasswd2 -c /srv/www/htdocs/schutz/.htpasswd benutzername
```

Für benutzername geben wir den Benutzer ein, der hinzugefügt werden soll, die Option -c steht dafür, dass eine neue Datei angelegt wird. Falls die Datei mit dem entsprechenden Pfad bereits existiert, braucht man diese Option nicht, die neue Benutzer werden in diesem Fall in die vorhandene Datei eingefügt.

Zum Schluss legen wir nun die .htaccess - Datei in einem Verzeichnis unserer Webstruktur an, das wir schützen wollen. Als Beispiel werden wir hier das Verzeichnis schutz schützen. Legen wir also zunächst die .htaccess - Datei an:

```
vi /srv/www/htdocs/schutz/.htaccess
```

In die Datei werden folgende Einträge geschrieben:

```
AuthType Basic
AuthName "Passwort erforderlich"
AuthUserFile /srv/www/htdocs/schutz/.htpasswd
require valid-user
```

Wenn man nun versucht, über den Browser auf das Verzeichnis schutz des Servers zuzugreifen, wird man aufgefordert, Benutzername und Kennwort einzugeben.

3 Samba

3.1 Einführung

Samba - Was ist das?
Kurz gesagt lässt Samba jeden Unixrechner in der Netzwerkumgebung von Windows erscheinen. Das heißt, man kann von Windows aus auf einen Unixrechner genau wie auf einen anderen Windowsrechner zugreifen. Der Clientrechner merkt gar nicht, dass er es nicht mit einem echten Windowsserver zu tun hat. Im Detail bedeutet das, dass sehr einfach Dateifreigaben erstellt werden können. Jeder Benutzer kann transparent Dateien auf seinem Heimatverzeichnis unter Unix und in anderen freigegebenen Verzeichnissen ablegen. Weiterhin kann man Drucker, die unter Unix ansprechbar sind, als Netzwerkdrucker in Windows ansprechen. Darüber hinaus bietet Samba viele Dienste, die sonst nur von Windows NT geleistet werden. Dazu gehören:

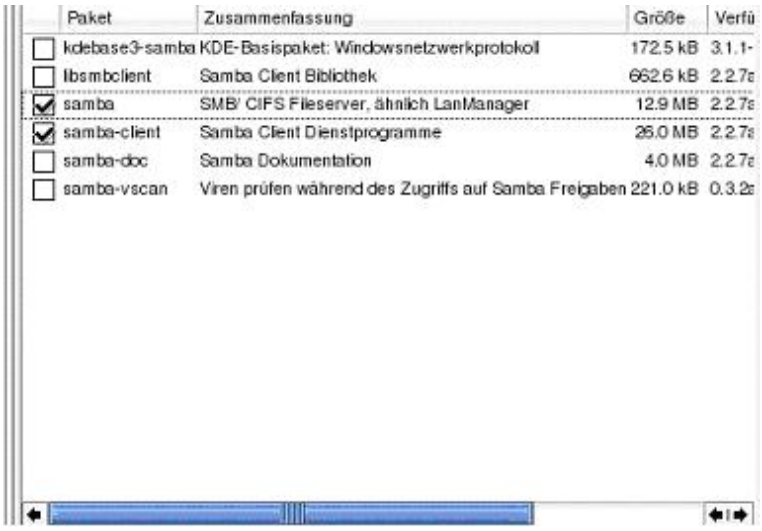
- **WINS-Server** Mit Samba kann sehr einfach ein WINS-Server eingerichtet werden. Dieser stellt Namensdienste für NetBIOS - Netze zur Verfügung, damit sich Windows-Maschinen über Subnetzgrenzen hinweg erreichen können
- **Computersuchdienst** Samba als sehr stabiler Server kann alle Aufgaben des Computersuchdienstes übernehmen. Die in Windowsumgebungen oft nicht sehr vorhersagbare Netzwerkumgebung kann so etwas stabiler gemacht werden.
- **Logon-Server** Für Windows-95/98/2K ist Samba Logon-Server, kann also die Domänenanmeldung für diese Systeme übernehmen.
- **PDC** Die Funktionalität des echten Primary Domain Controller ist nicht vollständig implementiert. Für viele Anwendungszwecke, insbesondere Authentifizierung von NT-Workstationbenutzern, reicht Samba jedoch völlig aus.
- **Diagnosewerkzeuge** Samba bietet eine Reihe von kleinen, aber sehr effektiven Werkzeugen, die die oft mühselige Suche nach Fehlern im Netz vereinfachen können. Samba bietet gegenüber anderen Implementationen des SMB-Protokolls einige Vorteile. Teilweise sind diese Vorteile von Unix geerbt, teilweise sind sie in der Architektur von Samba begründet.
- **Entfernte Administration** Der größte Vorteil von Samba in größeren Umgebungen ist die Möglichkeit, die gesamte Administration von der Kommandozeile aus durchzuführen. Damit bekommt man gegenüber grafischen Oberflächen sehr viel bessere Möglichkeiten, von entfernten Standorten aus zu administrieren. Werkzeuge wie PC Anywhere sind hier deutlich weniger flexibel. Zusätzlich bietet Samba die Möglichkeit der grafischen Administration über einen Webbrowser. Auch hier ist es unerheblich, wo sich Administrator und Server befinden.

- **Zentrale Konfiguration** Die gesamte Konfiguration von Samba befindet sich in einer einzigen Datei und ist nicht über viele Dialogfelder verteilt. Das erleichtert die Administration erheblich. So lässt sich eine funktionierende Konfiguration sehr einfach sichern und wieder einspielen.
- **Stabilität** Samba erbt von Unix eine hohe Stabilität. Unixrechner sind dafür ausgelegt, über Monate hinweg durchzulaufen und leisten dies auch. Samba als weiterer Prozess profitiert von dieser hohen Verfügbarkeit. Die modulare Struktur von Unix lässt es darüber hinaus zu, dass der Serverdienst Samba unabhängig von allen anderen Systemprozessen eigenständig neu gestartet werden kann, sofern hier ein Problem vorliegen sollte. Samba hat eine Architektur, die die Stabilität weiter fördert. Für jede Clientverbindung wird ein eigener Prozess gestartet. Verursacht also ein Client ein Problem auf Serverseite, wird möglicherweise der für diesen Client zuständige Prozess abstürzen. Die anderen Prozesse und damit Clients werden nicht gestört.
- **Skalierbarkeit** Samba kann von dem viel zitierten kleinen 386er unter Linux bis hin zu den größten heute verfügbaren Maschinen jede Hardware optimal ausnutzen. Die Architektur von Samba ermöglicht es, dass auch Multiprozessormaschinen ausgelastet werden. Multiprozessormaschinen können alle Prozessoren dann beschäftigen, wenn es viele unabhängige Prozesse im System gibt. Samba erstellt für jeden Client einen Prozess, der auf einem eigenen Prozessor ablaufen kann.
- **Flexibilität** Samba bietet eine riesige Anzahl von Konfigurationsoptionen, die zunächst einmal überwältigend wirkt. Die meisten Optionen werden nur für Spezialfälle benötigt, oder sind aus Kompatibilitätsgründen zu sehr exotischen Clients vorhanden. Soll Samba an spezielle Situationen angepasst werden, ist es durch ein sehr flexibles Schema von Makroersetzungen möglich, die Konfigurationsdatei weitgehend dynamisch zu verändern. Damit sind erheblich mehr Konfigurationsmöglichkeiten gegeben als mit Windows. Als Beispiel sei genannt, dass man sehr einfach einen Sambaserver unter zwei verschiedenen Namen in der Netzwerkumgebung erscheinen lassen kann, und beide virtuelle Server unterschiedlich konfigurieren kann. Zu Testzwecken ist es sogar möglich, zwei unterschiedliche Versionen gleichzeitig auf einer Maschine laufen zu lassen.

3.2 Installation

- YAST starten
- Software installieren oder löschen anklicken
- Filter: Suche
- Nach „samba“ suchen
- Samba und Samba-Client auswählen

- Akzeptieren anklicken



3.3 Eine „einfache“ Konfiguration

Für den Anfang soll hier eine einfache Konfiguration beschrieben werden, mit der ein Samba-Server im Netz erscheint und einige, wenige Dienste anbietet. Die einzelnen Parameter werden hier kurz erklärt.

Samba wird mit der Datei smb.conf konfiguriert. Je nach Unix oder Linux - Distribution kann diese Datei an unterschiedlichen Orten zu finden sein: /etc/smb.conf, /etc/samba/smb.conf oder auch /usr/local/samba/lib/smb.conf, wenn Samba selbst kompiliert wurde. Wurde die Datei smb.conf wie beschrieben angelegt, müssen zwei Dämonen gestartet werden: Der nmbd und der smbd. An dieser Stelle unterscheiden sich die Unix- und Linuxversionen erheblich, so dass keine allgemeinen Hinweise gegeben werden können. Verschiedene Möglichkeiten sind:

```
/etc/init.d/smb start

/sbin/init.d/smb start

/usr/local/samba/sbin/nmbd -D; /usr/local/samba/sbin/smbd -D

rcsmb start
```

oder einfach über die grafische Oberfläche das YAST starten und dann in „System“ im Run-Level-Editor die Dienste starten.

Die smb.conf für eine einfache Konfiguration könnte so aussehen:

```
[global]
workgroup = samba
netbios name = sambaserver
interfaces = eth0
encrypt passwords = yes

[homes]
valid users = %S
read only = no
browseable = no

[cdrom]
path = /cdrom

[public]
path = /pub
read only = no
```

Wenn man mit dieser Einstellung Zugriff auf den Server ermöglichen möchte, muss man für jeden Benutzer einen Eintrag in der Datei smbpasswd machen, da verschlüsselte Passwörter (encrypt passwords = yes) eingesetzt werden. Dies geschieht beispielsweise für den Benutzer **user1** über:

```
linux: # smbpasswd -a user1
New SMB password:
Retype new SMB password:
Added user user1.
linux: #
```

Die einzelnen Zeilen haben folgende Wirkung:

- [**global**] leitet globale Servereinstellungen ein. Alle anderen Abschnitte beschreiben Freigaben.
- **workgroup = samba** legt die Arbeitsgruppe fest, in der der Server auftauchen soll.
- **netbios name = sambaserver** gibt dem Server einen Namen, unter dem er im Netz erscheint.
- **interfaces = eth0** beschreibt das Netzwerk - Interface, auf dem Samba Dienste anbieten soll. Selbst wenn der Rechner nur ein einziges Netzwerkinterface hat, sollte dieser Parameter angegeben werden. Die vorhandenen Interfaces bekommt man bei den meisten Unixsystemen über den Befehl netstat -ian heraus.

- **[homes]** leitet die Freigabe der Heimatverzeichnisse sämtlicher Benutzer ein. Jeder Benutzer bekommt eine eigene Freigabe unter seinem eigenen Namen und hat damit einen eigenen Bereich, auf dem er schreiben kann.
- **valid users = %S** beschränkt den Zugriff auf den Benutzer, der sich verbinden möchte.
- **read only = yes/no** vergibt Schreibrecht auf die Freigabe. Standardmäßig wird nur Lesezugriff vergeben.
- **browseable = no** versteckt die Freigabe [homes] in der Netzwerkumgebung. Der Client zeigt sie nicht mehr als [homes] an, sondern nur noch unter dem Benutzernamen.
- **[cdrom]** leitet eine weitere Freigabe ein.
- **path = /cdrom** gibt den genannten Pfad frei. Dieser muss selbstverständlich im Dateisystem existieren.
- **[public]** macht noch eine Freigabe im Netz. Die Parameter sollten jetzt selbsterklärend sein.

Mit dieser minimalen smb.conf sollte es auf jeden Fall möglich sein, auf den Rechner zuzugreifen. Wenn man Probleme mit der Konfiguration weiterer Dienste bekommt, sollte man von einer möglichst einfachen Konfiguration ausgehen und dann Schritt für Schritt weitere Parameter hinzunehmen.

Oder einfach das ganze mit SWAT erledigen ;-)

3.4 Einen Samba-User adden

Die User, welche auf Samba zugreifen können sollen, müssen erst noch hinzugefügt werde. Dies wird mit dem Befehl smbadduser erreicht. Dazu wird zuerst der Befehl smbadduser gefolgt vom NT-Benutzernamen und vom Linux-Usernamen eingegeben, welche von einem : getrennt werden. Also könnte der Befehl so aussehen

```
smbadduser user1:user1
```

3.5 Samba Swat in Linux integrieren

Die Standardkonfigurationsdatei von Samba (/etc/samba/smb.conf), welche im Paket „Samba-client“ enthalten ist, beinhaltet die wichtigsten Einträge bereits. Nun können wir sie an unsere Bedürfnisse anpassen. Bei der Sambasuite ist bereits ein grafisches Administrationstool namens SWAT (SambaWebAdministrationsTool) dabei. SWAT ist ein Webkonfigurationstool, d.h. Von jedem beliebigen Browser kann der Sambaserver konfiguriert werden.

Damit SWAT verwendet werden kann, müssen einige Parameter geprüft und ggf. eingetragen werden.

3.5.1 Überprüfen der Datei /etc/services

Diese Datei enthält die offenen TCP/IP-Ports und sollte einen Eintrag in folgender Form enthalten:

```
swat            901/tcp
```

Wir durchsuchen die Datei /etc/services nach dem Eintrag swat mit dem Befehl

```
grep swat /etc/services
```

Die Ausgabe könnte wie folgt aussehen:

```
willi4:/ # grep swat /etc/services
swat 901/tcp # CONFLICT, not official assigned!
```

Aus dieser Zeile können sie entnehmen das SWAT auf TCP-Port 901 angesprochen wird. Wenn diese Zeile nicht vorhanden ist, benutzen wir den vi und fügen am Ende der Datei folgende Zeile ein:

```
swat  901/TCP
```

3.5.2 Überprüfen der Datei /etc/inetd.conf

Inetd, der Internet Metadämon, vereint viele TCP/IP-Dienste in sich. Weiterhin sollte auch ein Eintrag in der Datei /etc/inetd.conf für SWAT enthalten sein.

Wir durchsuchen die Datei /etc/inetd.conf nach dem Eintrag swat mit dem Befehl

```
grep swat /etc/inetd.conf
```

Die Ausgabe könnte folgendermaßen aussehen:

```
willi4:/ # grep swat /etc/xinetd.conf
# swat is the Samba Web Administration Tool
# swat  stream  tcp     nowait.400      root    /usr/sbin/swat  swat
swat stream tcp nowait.400 root usr/sbin/swat swat
willi4:/ \#
```

Sollten diese Zeile nicht in der inetd.conf vorhanden sein, fügen wir sie am Ende der Datei hinzu.

Damit SWAT gestartet werden kann müssen folgende Dienste aktiv sein:

Apache Webserver mit `/etc/rc.d/apache2 start`
Internet Metadämon mit `/etc/rc.d/xinetd start`
Sambaserver mit `rcsmb start`

3.5.3 Starten des Webkonfigurationstools SWAT

Die vorhandene Standardkonfigurationsdatei /etc/samba/smb.conf kann nun mit SWAT bearbeitet werden. Wir starten als User einen Webbrowser und geben „http://localhost:901“ in die Adressleiste ein. In der nun folgenden Anmeldemaske melden wir uns als root an.

Nun können wir Freigaben sehr leicht einrichten und verwalten. Wir erstellen nun zum Test eine Freigabe, die den anonymen Zugriff auf das Verzeichnis /tmp erlaubt.

3.6 Bestandteile von Samba

Das Programmpaket Samba besteht aus mehreren Programmen, von denen einige der Serverseite und andere der Clientseite zugeordnet werden können.

3.6.1 Die Servertools

- **smbd** ist der zentrale Serverprozess, der für die eigentlichen Datei- und Druckdienste zuständig ist. Sie werden mehrere smbds im System finden. Einer dieser Prozesse hört auf dem TCP-Port 139, und nimmt neue Verbindungen entgegen. Jede neue Verbindung stößt einen neuen smbd Prozess an. Wenn Sie einen Client vom Sambaserver trennen wollen, müssen Sie nur mit smbstatus die Prozessnummer des zuständigen smbd erfragen, und diesen einen Prozess töten. Jeder aktive Client benötigt etwa 1-2 MB Hauptspeicher auf dem Server. Clients, die gerade nicht aktiv Dateien mit dem Sambaserver austauschen, benötigen praktisch überhaupt keine Ressourcen. Viel Hauptspeicher kann von Samba selbstverständlich gut als Cache genutzt werden.

- **nmbd** ist für die NetBIOS Namens- und Datagrammdienste zuständig. Dieser Prozess reserviert beim Start von Samba die entsprechenden NetBIOS-Namen, er kann WINS-Server sein und ist für den Computersuchdienst zuständig.

- **testparm** Mit diesem Programm kann man die smb.conf auf syntaktische Korrektheit prüfen. Das Programm liest die Konfigurationsdatei ein und gibt Fehlermeldungen aus, sofern es unbekannte Parameter findet.

- **smbpasswd** wird zur Pflege der verschlüsselten Passwörter auf Serverseite verwendet.

- **smbcontrol** Mit diesem Programm lassen sich die Dämonen von Samba kontrollieren. Beispielsweise kann man für einzelne Dämonen den Debuglevel gezielt auf einen gewünschten Wert setzen.

3.6.2 Die Clients

- **smbclient** Mit dem Programm smbclient kann man auf Freigaben von NT-Rechnern zugreifen. Man kann auf von NT zur Verfügung gestellten Druckern drucken und man kann NT-Freigaben in tar-Dateien sichern. Weiterhin kann mit smbclient die Liste der Server im Netz erfragt werden, analog zu der Netzwerkumgebung unter Windows.

- **nmblookup** ist ein Diagnosewerkzeug für die NetBIOS-Namensauflösung. Wenn zwei Computer mit Windows sich nicht finden können, kann man mit nmblookup deren Versuche, sich gegenseitig zu finden, genau nachstellen. Ebenso können WINS-Server befragt werden und ein NetBIOS Node Status abgefragt werden. Das entsprechende Programm unter Windows ist das Kommandozeilenprogramm nbtstat.

- **smbcacls:** Mit diesem Programm lassen sich von Unix aus Access Control Lists auf Windows-Dateien auslesen und setzen. Ist Samba mit ACL-Support kompiliert, geht dies selbstverständlich auch für die auf Unix abgelegten Dateien.

3.6.3 Weitere Serverkomponenten

- **smb.conf:** Die zentrale Konfigurationsdatei von Samba. Ist Samba als fester Systembestandteil installiert, findet sie sich in der Regel unter /etc/smb.conf. oder unter /etc/samba/smb.conf. Wurde Samba selbst kompiliert, so liegt sie häufig unter /usr/local/samba/lib/smb.conf.

- **/var/log/samba:** Samba benötigt ein Verzeichnis, in dem es temporäre Logdateien und Datenbanken ablegen kann. Wird Samba ohne besondere Optionen selbst kompiliert, liegt dieses Verzeichnis unter /usr/local/samba/var.

- **/etc/smbpasswd** ist die Passwortdatenbank von Samba, sofern mit verschlüsselten Passwörtern gearbeitet wird. Bei selbst kompilierten Sambaversionen liegt diese Datei häufig im Verzeichnis /usr/local/samba/private/.

3.7 Verbindungsaufbau zum Samba-Server

3.7.1 Verbindungsaufbau unter Windows

Um eine Verbindung zum Samba-Server herzustellen genügt es, in der Adresszeile des Explorers die IP-Adresse des Servers einzugeben.

3.7.2 Verbindungsaufbau unter Linux

In der Konsole folgendes eingeben (mit root-Rechten):

```
mount -t smbfs //(IP des Sambaservers)/(Name der Freigabe) /mnt/samba
-o username=(Benutzername)
```

4 MySQL

4.1 Allgemeines zu SQL und MySQL

4.1.1 SQL

Abkürzung für „Structured Query Language“. In den 70er Jahren des 20. Jahrhunderts von der Firma IBM entwickelte Abfragesprache für die relationale Datenbank DB2. Es handelte sich dabei um eine nichtprozedurale (Programmier-) Sprache, die weder Schleifen, Unterprogramme noch Funktionen enthielt. SQL-Befehle setzen sich aus zwei Teilen zusammen,

- der Data Definition Language (DDL) zum Aufsetzen einer Datenbankstruktur und
- der Data Manipulation Laguage (DML) zur Manipulation der enthaltenen Daten.

Relationale Datenbanken arbeiten mengen- und gruppenorientiert. Wer sich also mit SQL beschäftigt, der muß sich ein wenig in Mengenlehre auskennen. Dr. E.F. Codd entwickelte 1970 Regeln, die eine relationale Datenbank definieren:

- Ein relationales Datenbankmodell muß in der Lage sein, Datenbanken vollständig über seine relationalen Fähigkeiten zu verwalten.
- Darstellung von Informationen: Alle Informationen in einer relationalen Datenbank (einschließlich Namen von Tabellen und Spalten) sind explizit als Werte in Tabellen darzustellen.
- Zugriff auf Daten: Jeder Wert in einer relationalen Datenbank muß durch Kombination von Tabellenname, Primärschlüssel und Spaltenname eindeutig zu finden sein.
- Behandlung von Nullwerten: Das DBMS behandelt Nullwerte durchgängig als unbekannte oder fehlende Daten und unterscheidet so von Standardwerten. Zahlen können also drei Werte annehmen, 0, einen Wert und NULL für „Wert nicht vorhanden“. Man spricht hier auch von der Dreiwertigkeit.
- Struktur einer Datenbank: Die Datenbank und ihre Inhalte werden in einem sogenannten Systemkatalog auf derselben logischen Ebene wie die Daten selbst - also in Tabellen - beschrieben. Demzufolge läßt sich der Katalog mit Hilfe der Datenbanksprache abfragen.
- Abfragesprache: Zu einem relationalen System gehört mindestens eine Abfragesprache mit einem vollständigen Befehlssatz für Datendefinition, Manipulation, Integritätsregeln, Autorisierung und Transaktionen.
- Aktualisieren von VIEWS (Sichten): Alle VIEWS, die theoretisch aktualisiert werden können, lassen sich auch vom System aktualisieren.

- Abfragen und Bearbeiten ganzer Tabellen: Das DBMS unterstützt nicht nur Abfragen, sondern auch die Operationen für Einfügen, Aktualisieren und Löschen in Form ganzer Tabellen.
- Physikalische Datenunabhängigkeit: Der logische Zugriff auf die Daten durch Anwendungen muß unabhängig von den physikalischen Zugriffsmethoden oder den Speicherstrukturen der Daten sein.
- Logische Datenunabhängigkeit: Änderungen der Tabellenstrukturen dürfen keinen Einfluß auf die Logik der Anwendungen haben.
- Unabhängigkeit der Integrität: Integritätsregeln müssen sich in der Datenbanksprache definieren lassen. Die Regeln müssen im Systemkatalog gespeichert werden. Es darf nicht möglich sein, die Regeln zu umgehen.
- Verteilungsunabhängigkeit: Der logische Zugriff auf die Daten durch Anwendungen darf sich beim Übergang von einer nicht verteilten zu einer verteilten Datenbank nicht ändern.
- Kein Unterlaufen der Abfragesprache: Integritätsregeln, die über die Datenbanksprache definiert sind, dürfen sich nicht mit Hilfe von Low-Level Sprachen umgehen lassen.

Dr. E. F. Codd hat in SQL alle Elemente der Algebra integriert, um Daten mengenmäßig zu erfassen, zu speichern, und diese in Relation zueinander zu setzen. Hierbei können Schnittmengen, Vereinigungsmengen, Restmengen u.s.w. gebildet und ausgegeben werden. Diese werden durch sogenannte JOINS durchgeführt.

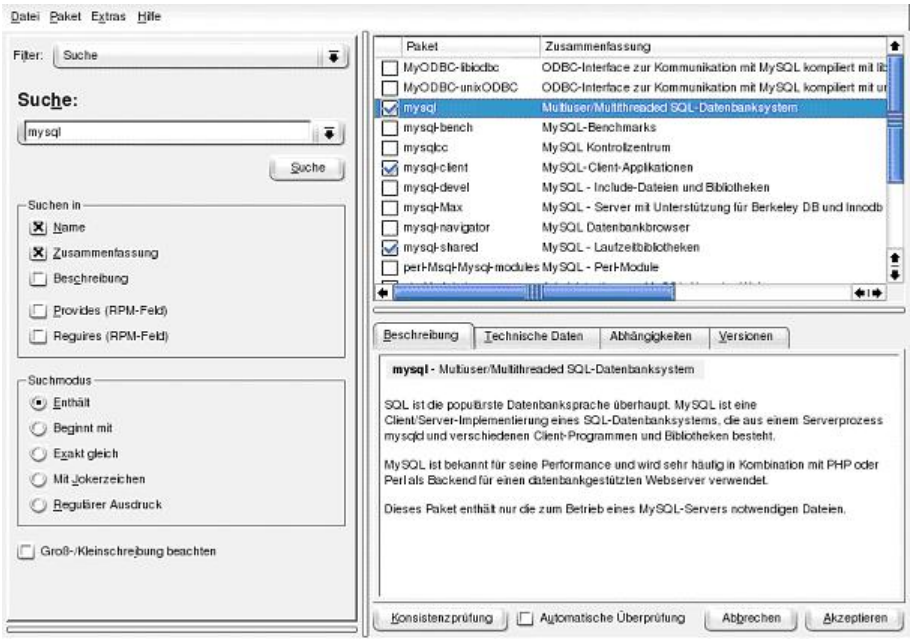
Bei dem Aufbau von einer SQL Datenbank müssen die Daten so aufgeteilt werden, daß sie voneinander verschiedene, eindeutige Datensätze bilden. Das praktische an SQL ist, daß wenn man die Abfragesprache einmal erlernt hat, kann man aus riesigen Datenmengen wirklich alle Informationen auslesen, die man braucht. Z.B. kann man Daten sortieren, verändern, filtern, Statistiken erheben, u.s.w. Mit Hilfe von Datenbankschnittstellen, wie z.B. ODBC und ASCII Im/Export kann man Daten mit beliebigen Datenbanken austauschen und sogar diese mit einbinden. Die ODBC Schnittstelle dient hier als Datenbank unabhängige Schnittstelle zu Applikationen unter Windows und UNIX.

4.1.2 MySQL

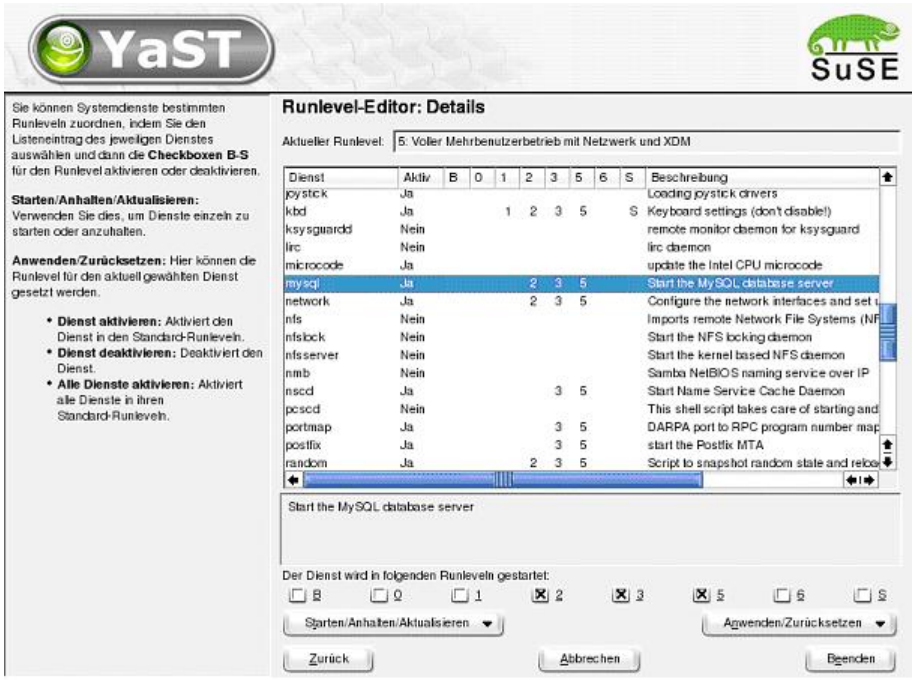
MySQL ist ein relationales Datenbank Managementsystem (rdbms), welches von der Firma T.c.X. DataKonsult in Schweden entwickelt wurde. MySQL unterstützt User mit einer leistungsstarken Multi-User, Multi-threaded SQL (Structured Query Language) Datenbanklösung, welche schnell, robust und einfach im Gebrauch ist. MySQL ist zudem eine Open Source Anwendung im Sinne der GNU.

4.2 MySQL Installation

- YAST starten
- Software installieren oder löschen anklicken
- Filter: Suche
- nach „mysql“ suchen
- mysql, mysql-client und mysql-shared auswählen
- Akzeptieren anklicken



4.3 MySQL-Server in die Runlevels eintragen und starten



4.4 Passwort für den Datenbankserver vergeben

In der Shell mit MySQL-Admin das Passwort für die Datenbank setzen:

```
mysqladmin password sommer
```

Wenn bereits ein Passwort vergeben ist, muss zusätzlich das alte abgefragt werden. Dazu muss hinter den Befehl noch die Option -p gesetzt werden. Diese verlangt dann die Eingabe des alten Passworts.

```
willi9:/ # mysqladmin password winter -p
Enter password:
willi9:/ #
```

4.5 Aufrufen der MySQL-Konsole

Um die MySQL-Abfrage Konsole zu öffnen, genügt es in der Shell `mysql -p` einzugeben. Die Option -p steht hierbei wieder dafür, dass das Passwort für den Datenbankserver abgefragt wird.

5 Router und Proxy

5.1 Router

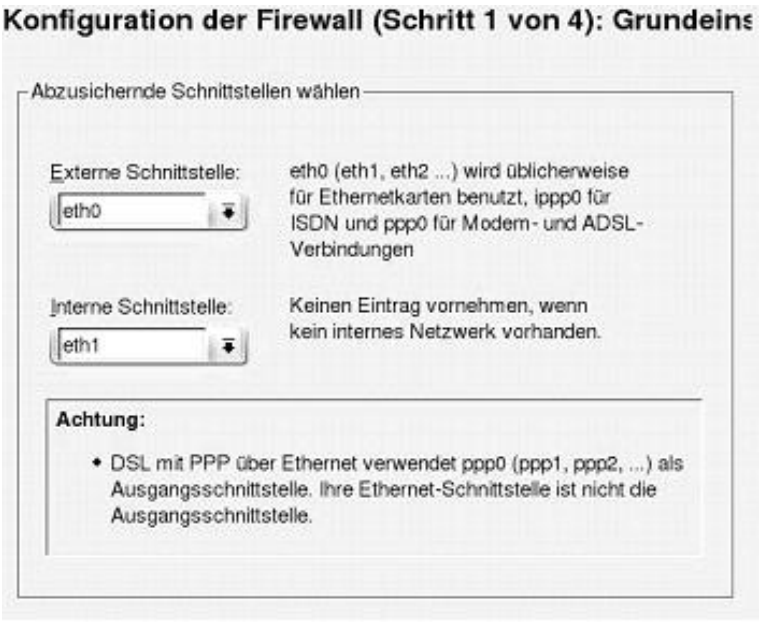
5.1.1 Allgemeines

Allgemein gesagt, verbindet ein Router verschiedene Segmente eines Netzwerks miteinander, d.h. wenn man das lokale Netzwerk mit dem Internet verbinden möchte, dann benötigt man einen Router (alternativ kann man auch einen Proxy benutzen).

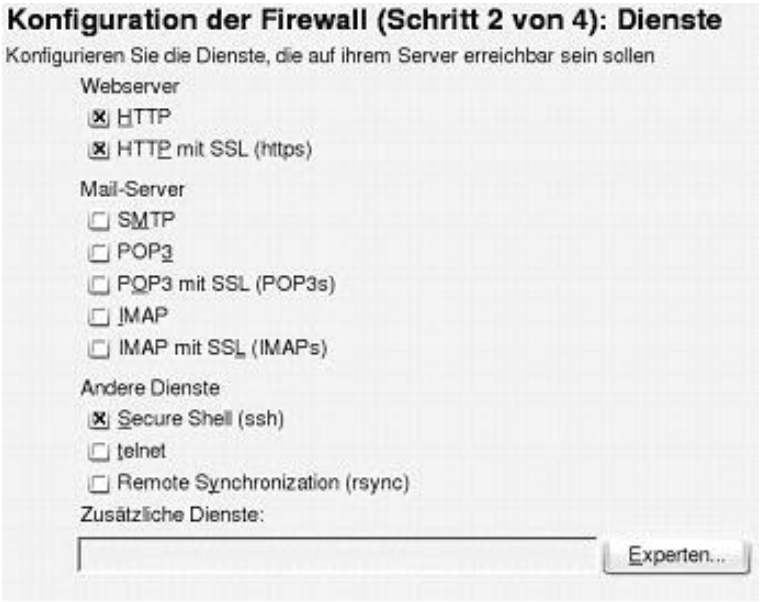
Ein Router hat den Vorteil (für Zocker o.ä.), dass man nicht jeden Port einzeln freischalten muss, sondern dass alle Anfragen ans Internet (gleich, welchen Typs sie sind) weitergeleitet werden, die nicht lokal verarbeitet werden können.

5.1.2 Routing/Firewall einrichten

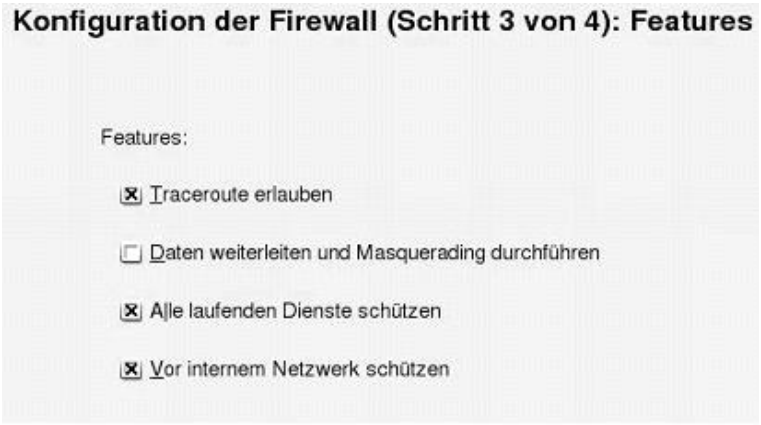
- Voraussetzung zum Routing sind natürlich 2 Netzwerkkarten
- unter Linux einfach das YAST starten
- unter Sicherheit und Benutzer dann die Firewall Einstellungen auswählen
- dann für die jeweiligen Schnittstellen jeweils eine Netzwerkkarte auswählen(Bsp.: eth0 und eth1 für Ethernetkarten)



- Im zweiten Schritt müssen die Dienste markiert werden, die weiterhin auf dem Server erreichbar seien sollen



- Im dritten Schritt muss dann Daten weiterleiten und Masquerading durchführen aktiviert werden



- Der vierte Schritt ist relativ unwichtig und wird hier nicht weiter behandelt.
- Nach einem weiteren Klick auf weiter wird der Router dann aktiviert.

5.2 Proxy Server - Squid

Sollen sich mehrere Surfer einen Web-Zugang teilen, der sicher, schnell und zudem flexibel ist, bietet sich ein Caching-Web-Proxy wie Squid an.

Squid steht unter der GNU GPL. Er ist sehr ausgereift, schnell und flexibel. Deshalb ist er von den Web-Proxies am weitesten verbreitet und wird gerade in sehr großen Umgebungen, wie Universitäten und großen, verzweigten Unternehmen, verwendet.

Warum sollten wir seine Vorteile nicht auch in kleineren Netzwerken nutzen?

5.2.1 Was macht ein Web-Proxy?

Zunächst die Vorteile eines Web-Proxies im allgemeinen:

1. Proxy heißt übersetzt **Stellvertreter**, und als solcher holt er für die Nutzer die Web-Seiten aus dem Netz. Nach außen ist netzwerktechnisch nur der Proxy zu sehen, der Zugreifende ist hinter ihm versteckt und dadurch geschützt.

2. Als weitere Fähigkeit kann ein Proxy meist statische Web-Inhalte zwischenspeichern, was **Caching** genannt wird. Ein erneuter Zugriff auf die gleichen Inhalte wird dadurch erheblich beschleunigt, und das bei gleichzeitig geringerer Netzlast!

5.2.2 Warum gerade den Squid als Web-Proxy verwenden?

Der Einsatz des Squid bringt mehrere Vorteile mit sich:

Freie Lizenz Squid ist eine Open-Source-Entwicklung unter der GNU GPL. Somit fallen keine Lizenz-Kosten an, der Quelltext ist frei verfügbar und an eigene Bedürfnisse anpassbar.

Stabilität Der Squid wird seit vielen Jahren entwickelt und hat sich auch gerade in größeren Umgebungen, wie Universitäten und großen Firmen, bewährt. Im Laufe der Entwicklung ist er schneller und vielseitiger geworden, so dass er sich hinter keinem kommerziellen Proxy verstecken muss.

Ein Blick in die gut dokumentierte **/etc/squid/squid.conf** zeigt, wie viele Optionen mit dem Squid offen stehen. Und es gibt eine sehr aktive Gemeinde, die diesen Proxy immer weiter entwicket.

Geschwindigkeit Statische Inhalte, welche einmal abgerufen wurden, können zwischengespeichert werden. Dazu gehören auch Grafiken von dynamisch generierten Seiten. Ein erneutes Abrufen solcher Inhalte, auch von einem anderen Benutzer, kann aus dem Zwischenspeicher bedient werden. Die Anfrage ist dadurch erheblich schneller beantwortet, der Internetzugang wird entlastet. Die Aktualität der Seiten wird durch sehr ausgeklügelte Methoden sichergestellt.

Die am häufigsten genutzten Seiten werden im Arbeitsspeicher gehalten (**hot object**). Die nicht so schnelle Festplatte wird für länger zurückliegende Zugriffe genutzt.

Eine zusätzliche Beschleunigung bewirkt das Zwischenspeichern der Zuordnung **Name zu IP-Adresse** (DNS-Caching).

Kontrolle Soll der Zugriff eingeschränkt werden, ist dies über die Rechtevergabe mit den **Zugriffs-Kontroll-Listen (Access Control Lists, ACL)** von Squid flexibel möglich. Sinnvoll kann dies sein, um die Ablenkung durch die Angebots-Flut des Internets einzuschränken, juristische Probleme zu vermeiden (z. B. durch pornografische Inhalte in Schulen) oder um die Online-Kosten im Griff zu halten.

Bereits getätigte Zugriffe können übersichtlich ausgewertet werden. So ist nachvollziehbar, wer welche Seiten aufgesucht und wer wie viel Daten übertragen hat. Ebenso kann ermittelt werden, welche URLs am häufigsten aufgesucht wurden oder auch wieviele Daten insgesamt übertragen wurden. Zusätzliche Tools, wie webalizer (http://www.webalizer.org) und **cachemanager**, helfen die Logfiles auszuwerten. So kann rechtzeitig ermittelt werden, wann die Proxy-Hardware nicht mehr ausreicht. Oder es können bestimmte Seiten gesperrt werden, die den Internet-Zugang überstrapazieren.

In diesen Kontrollmöglichkeiten liegt natürlich auch die Gefahr, in die Privatsphäre anderer einzugreifen. Deshalb ist die Informationsflut in die Logfiles abgestuft deaktivierbar (z. B. mit dem Parameter **client_netmask**).

Erhöhte Sicherheit Eine Firewall kann ein lokales Netz effektiver absichern, wenn sie einen Proxy wie den Squid verwendet, anstatt nur auf Paketfilterung zu vertrauen. Der Grund dafür liegt darin, dass Paketfilter auf TCP/IP-Ebene, nicht aber den Inhalt von HTTP- und FTP-Verbindungen analysieren können. Proxies können aber genau diese Inhalte erkennen.

Zusätzlich können Proxies die Clients des lokalen Netzes erheblich besser verbergen, als es durch Network Adress Translation (NAT) möglich wäre. Mit Hilfe des Squid kann genau definiert werden, was an den Webserver übertragen werden soll und was nicht (z. B. mit dem Parameter **forwarded_for**). Es können einige Viren geblockt werden (z. B. Nimda).

Vereinfachte Namensauflösung Praktisch ist auch, dass ein Proxy die Namensauflösung zu den IP-Adressen übernimmt. Es muss im internen Netz kein öffentlicher Name auflösbar sein, was die DNS-Konfiguration erleichtert.

Flexibilität Insbesondere in komplexeren Netzwerken ist der Squid-Proxy sehr flexibel.

So kann z. B. genau definiert werden, welche Seiten aus dem Internet, welche von einem anderen Proxy und welche direkt aus dem lokalen Netz geholt werden sollen.

Es kann über mehrere Standorte ein so genannter **Cache-Verbund** aufgebaut werden, was die Netzlast deutlich vermindern kann und die Ausfallsicherheit erhöht.

Des weiteren sind zusätzliche Module verfügbar, welche die Funktionalität des Squid erweitern können. Hierzu zählen z. B. SquidGuard oder auch Module für die Nutzung einer Windows-Benutzerverwaltung im Squid.

5.2.3 Squid - Konfiguration

Nach der Installation ist der Squid nicht ohne Anpassungen lauffähig, er muss zunächst über die Datei **/etc/squid/squid.conf** an die vorhandene Netzwerk-Situation angepasst werden. Diese Datei ist sehr gut kommentiert, aber aufgrund der vielen Einstelloptionen auch sehr umfangreich. Zum Glück können fast alle Optionen unverändert übernommen werden.

Um den Proxy erst einmal zum Laufen zu bringen, müssen die hier vorgestellten Parameter angepasst werden. Ergänzend kann auch in die QUICKSTART-Datei der Squid-Doku geschaut werden (meist unter **/usr/share/doc/squid**).

Netzwerkoptionen Hier sind einige Einstellungen zu den Netzwerkbindungen und den verwendeten Protokollen zu treffen. Netzwerkbindung:

```
#  TAG: http_port
#  Usage:  port
#  hostname:port
#  1.2.3.4:port
#  Default:
#  http_port 3128
http_port 3128
http_port 8080

# http_port 192.168.8.21:8080 Schreibweise mit IP
# http_port willi21:8080      Schreibweise mit Hostname
```

Hier wird festgelegt, auf welchem Port oder IP-Adresse/Port bzw. Hostname/Port Squid Verbindungen annimmt. Hierbei können mehrere Adressen in unterschiedlichen Zeilen angegeben werden. Die Standardeinstellung ist, dass Squid Verbindungen auf allen IP-Adressen des Rechners auf Port 3128 entgegen nimmt. Außerdem erwarten viele Anwender einen Proxy auf dem Port 8080, deshalb kann es recht sinnvoll sein, Squid auf beiden Ports lauschen zu lassen..

Zugriffs-Rechte Die Voreinstellung des Squid ist aus Sicherheitsgründen so, dass keiner surfen darf. Die Rechtevergabe ist sehr flexibel und damit leider auch kompliziert. Es soll der Einfachheit halber zunächst allen Nutzern im eigenen lokalen Netz Vollzugriff auf das Internet gewährt werden.

Für die Rechtevergabe muss zunächst die Zugriffs-Kontroll-Liste (Access Control List, ACL) definiert werden. Anschließend wird über den Namen dieser

ACL das Recht mit dem Schlüsselwort **http_access allow** zugewiesen. Weitere Details sind zu finden unter: Rechtevergabe unter Squid im Detail

Die Syntax der ACL:

```
acl (frei_definierbarer_Name) (acl_Typ) (Werte)
```

Angenommen, die PCs des eigenen Netzes verwenden alle IPs aus dem Bereich **192.168.8.***, könnte die Konfiguration für Vollzugriff des eigenen lokalen Netzes wie folgt aussehen:

```
acl erlaubte_rechner src 192.168.8.0/255.255.255.0
http_access allow erlaubte_rechner
http_access deny all
```

Die letzte Zeile ist schon vorhanden, die beiden darüber liegenden müssen neu angelegt werden. Die Reihenfolge der http_access-Zeilen ist wichtig! Für die eigene Nutzung muss natürlich die IP-Netz-Adresse **192.168.8.0/255.255.255.0** angepasst werden.

Größe des Festplatten-Zwischenspeichers Dieser Parameter ist nicht zwingend zu verändern, damit Squid lauffähig wird, aber so wichtig, dass er hier aufgeführt wird.

Soll der Zwischenspeicher (Cache) viele Objekte enthalten können, muss die knappe Voreinstellung von 100 MB vergrößert werden.

Doch Vorsicht: Es kann leider nicht der gesamte verfügbare Platz einer Partition genutzt werden, da zusätzlich noch Verwaltungsdaten Platz benötigen. Steht dem Squid nicht genügend Platz zur Verfügung, kann er nicht arbeiten. Die Angabe der Zwischenspeicher-Größe sollte deshalb immer mindestens 10% unter dem Wert des freien Platzes liegen.

Die Syntax ist:

```
cache_dir (Pfad zum Zwischenspeicher) (Größe) (nicht zu verändernde Parameter)
```

Soll die Größe z.B. auf 10000MB eingestellt werden, kann folgendes eingetragen werden:

```
cache_dir /var/cache/squid 10000 16 256
```

5.2.4 Squid-Daemon starten

Die Installation von Squid über RPM oder DEB ermöglicht eine relativ einfache Inbetriebnahme des Dienstes. In diesen Fällen ist bereits einiges vorkonfiguriert, was ansonsten von Hand gemacht werden muss (etwa das Anpassen von Dateisystemrechten oder das Erstellen eines Init-Skripts).

Bevor Squid das erste Mal gestartet wird, muss die Cache-Verzeichnisstruktur angelegt werden mit:

```
root@linux # squid -z
```

Danach kann Squid über das von der jeweiligen Linux-Distribution vorgesehene Startskript gestartet werden, meist durch:

```
root@linux # /etc/rc.d/squid start
```

Zum Neustart von Squid ist der Befehl analog dazu:

```
root@linux # /etc/rc.d/squid restart
```

Bei Problemen sollten die Logfiles /var/log/messages und cache.log untersucht werden. Wo die cache.log abgelegt ist, kann ermittelt werden mit:

```
root@linux # grep cache_log /etc/squid/squid.conf
```

5.2.5 Rechtevergabe unter Squid im Detail

Warum Zugriffsrechte für den Internet-Zugriff? Werden die Rechte lediglich so gesetzt wie im Abschnitt Squid-Konfiguration auf die Schnelle beschrieben, können alle Nutzer auf sämtliche Inhalte des Internet zugreifen.

Das mag als Konfiguration häufig reichen, es können sich daraus jedoch folgende Nachteile ergeben:

- Der Internetzugang ist schnell überlastet und damit langsam.
- Erhöhte Kosten können entstehen.
- Die Nutzer können sich in der Informationsflut des Internets verlieren (wichtig bei Schulungen/Schülern).

Umsetzung Das Setzen der Rechte unter Squid ist sehr flexibel, aber leider nicht selbsterklärend.

Mit einer ACL wird zunächst festlegt, auf was (Ziel) oder von was (Quelle) zugegriffen wird. Über http_access wird dieser ACL anschließend ein Verbot oder eine Erlaubnis zugewiesen.

ACL-Elemente - Ziel oder Quelle definieren Die Syntax einer ACL sieht folgendermaßen aus:

```
acl (frei_definierbarer_acl_Name) (acl-Typ) (Wert ...)
```

Über den frei definierten ACL-Namen kann mit Hilfe des unten erläuterten http_access das gewünschte Recht vergeben werden. Diese ACL-Namen dürfen natürlich nicht doppelt vergeben werden.

Es können verschiedene Arten von Quellen und Zielen zur ACL-Definition verwendet werden. Hier die für die Zugriffsverwaltung genutzten ACL-Typen:

- **src:** Absender-IP-Adressen der Client-PCs, die über Squid auf Web-Inhalte zugreifen wollen.
- **dst:** Ziel-IP-Adressen, auf die zugegriffen werden soll.

- **dstdomain:** Name der Ziel-Domain, auf deren Server im Internet zugegriffen werden soll. Hier kann auch direkt ein Server angegeben werden!
- **dstdom_regex:** Wie dstdomain, zusätzlich können Reguläre Ausdrücke verwendet werden, um die Liste der Server zu erweitern.
- **time:** Zur Festlegung bestimmter Zeitbereiche, in denen gesurft werden darf.
- **url_regex:** URLs können über Reguläre Ausdrücke definiert werden.
- **urlpath_regex:** Der Pfad, also alles, außer dem Protokoll (wie **http://**) und dem Rechnernamen (wie **www.selflinux.org**), wird mit dem angegebenen Regulären Ausdruck verglichen.
- **ident:** Vergleich der Namen, die von den Unix-Clients mit dem identd übertragen werden, mit denen in der Liste. So kann eine einfache Benutzerverwaltung genutzt werden. Den identd gibt es auch für Windows als Programm oder als Dienst.
- **external:** Einbinden eines externen Hilfsprogramms, das z. B. ermöglicht, zur Benutzerverwaltung einen NT-Domänen-Kontroller zu nutzen.

Weitere ACL-Typen stehen für spezielle Anwendungen des Squid zur Verfügung, die nicht für die Benutzerverwaltung wichtig sind.

Werden mehrere Werte hinter dem ACL-Typ aufgelistet, braucht nur einer der Werte zu passen, um das zugehörige http_access zu aktivieren (OR-Logik, siehe Logik der Rechtevergabe).

Zur besseren Übersicht können die aufzulistenden Werte auch in eine eigene Datei ausgelagert werden. Dort wird für jeden Eintrag eine eigene Zeile angelegt. Die Datei muss Squid wie folgt bekanntgegeben werden:

```
acl (frei_definierbarer_Name) (acl-Typ) "(Pfad_zur_Datei)"
```

Zugriffsrechte für die ACLs definieren Mit http_access wird in Kombination mit allow bzw. deny ein Recht für die definierten ACL-Elemente festgelegt. Die Syntax:

```
http_access allow/deny (acl-Name ...)
```

Das sieht zunächst sehr simpel aus. Einer definierten ACL wird über deny oder allow ein gewünschtes Recht zugewiesen, so wie im Beispiel des Abschnittes Squid-Konfiguration auf die Schnelle gezeigt:

```
acl erlaubte_rechner src 192.168.8.0/255.255.255.0
http_access allow erlaubte_rechner
```

Komplexer wird es, wenn mehrere ACL-Namen in einer Zeile aufgelistet sind. Dies bewirkt, dass alle aufgelisteten ACLs zutreffen müssen, damit das allow oder deny in Kraft treten kann. Beispiel:

```
acl selflin            dstdomain .selflinux.org
acl pcRestr            src    192.168.20.0/255.255.255.0
acl erlaubte_rechner   src    192.168.8.0/255.255.255.0
acl Zeitbesch  time   10:00-12:00
http_access allow  selflin pcRestr
http_access allow  erlaubte_rechner Zeitbesch
```

Die Angabe der zwei ACL-Namen **selflin** und **pcRestr** nach dem ersten http_access bewirkt, dass Rechner mit der IP **192.168.20.*** nur auf SelfLinux-Seiten surfen dürfen. Die **erlaubte_rechner** dürfen alles sehen, aber nur in der Zeit von 10 Uhr bis 12 Uhr, da diese mit der ACL **Zeitbesch** verknüpft sind.

Hier ist gleich eine wichtige Eigenart des ACL-Typs dstdom zu erkennen: Der '.' vor dem Domainnamen sagt Squid, dass auch Subdomains in diese ACL fallen, wie z. B. www.selflinux.org oder srv.sub.selflinux.org.

Logik der Rechtevergabe Es ist sehr wichtig für das Berechtigungsmodell von Squid, die OR/AND-Logik zu verstehen:

- Alle Elemente eines ACL-Eintrags werden mit OR verknüpft.
- Alle Elemente eines Access-Eintrags dagegen werden mit einem AND verknüpft.

Noch ein Beispiel, das fatalerweise überhaupt keinen Zugriff ermöglicht:

```
acl wir src 192.168.8.0
acl ihr src 192.168.20.0
http_access allow wir ihr
```

Hier würden Zugriffe dann erlaubt werden, wenn sich der Surfende zugleich mit den beiden Quell-IPs an den Proxy wendet, was nicht möglich ist. Somit werden keine Zugriffe mehr erlaubt.

Sollen dagegen beide IP-Adressen Zugriff erhalten, muss folgendes eingetragen werden:

```
acl wir src 192.168.8.0 192.168.20.0
http_access allow wir
```

Ein weiterer wichtiger Punkt ist die Reihenfolge der access-Listen. Ist erst einmal ein Zugriff erlaubt worden, kann er durch darunter liegende Zeilen nicht wieder zurückgenommen werden. Beispiel:

```
acl selflin  dstdomain .selflinux.org
acl verboten src      192.168.8.23
http_access allow selflin
http_access deny  verboten
```

Obwohl 192.168.8.23 in der letzen Zeile alles verboten wird, kann dieser PC die Seiten von SelfLinux sehen, da die Erlaubnis über dem kompletten Verbot vergeben wurde.

6 glFTPd

6.1 Allgemeines

Der glFTPd ist ein sehr fortschrittlicher und umfangreicher FTP-Server. Einer der Hauptunterschiede zu anderen FTP-Servern ist seine eigene Benutzerdatenbank, welche man über die site-Kommandos vollständig online verwalten kann. Diese site-Kommandos können auch dazu genutzt werden, um sich Statistiken oder Logdateien anzusehen, Scripte auszuführen und noch viele andere Dinge zu tun. Der glFTPd läuft in einer chroot-Umgebung, was ihn relativ sicher macht.

glFTPd bedeutet **G**rey**L**ine **F**ile **T**ransfer **P**rotocol **D**emon. Er wurde nach seinem ersten Entwickler GreyLine benannt. Diese erste Veröffentlichung dieser Software geschah Anfang des Jahres 1998. Seit damals wurde sie enorm weiterentwickelt und wird heute von vielen Menschen weltweit genutzt.

6.2 Features

Der Funktionsumfang des glFTPd macht viele komplexe und komplizierte Setups möglich. Hier eine Liste der wichtigsten Features:

- virtuelle Benutzer und Gruppen
- Bandbreitenbeschränkung (global und nutzergebunden)
- Upload/Download-Ratio Unterstützung
- Echtzeit CRC-Berechnung für hochgeladene Dateien
- Script-Support für fast alle Kommandos und Operationen
- Online User-Management
- eingebundene Statistiken über site-Kommandos
- Verschlüsselung durch TLS/SSL-Integration
- ACL Support

6.3 Installation

Die aktuelle Version des glFTPd (im Moment v1.32) kann man sich auf der Homepage des Projekts herunterladen.

```
www.glftpd.com
```

Nachdem wir das Archiv der Installationsdateien heruntergeladen und entpackt haben, führen wir als root das Installationsscript aus:

```
./installgl.sh
```

Dieses Script erledigt für uns alle Kompilierungs- und Konfigurationsschritte, wir müssen lediglich ein paar Abfragen bestätigen. Zuerst will das Script wissen, ob der TCP-Demon benutzt werden soll.

```
Use tcpd? [Y]es [N]o:
```

Diese Abfrage bestätigen wir mit „y“. Als nächstes erhalten wir die Abfrage, ob der glFTPd in einer Gefängnisumgebung (jailed) laufen soll. Da dies ein erheblicher Beitrag zur Sicherheit des FTP-Servers ist, bestätigen wir auch dieses.

```
Use a jailed environment? [Y]es [N]o:
```

Die nun folgende Abfrage verlangt die Eingabe des Verzeichnisses, in das der glFTPd eingeschlossen werden soll. Hier können wir das vergegebene Standardverzeichnis benutzen und mit der Eingabetaste fortsetzen.

```
Please enter the private directory to install glftpd inside [/jail]:
```

Jetzt will das Script wissen, ob für den glFTPd eine private Gruppe angelegt werden soll und nur der root-User Zugriff auf den glFTPd hat. Da der shell-Zugriff auf den glFTPd für uns nicht interessant ist, bestätigen wir auch diese Abfrage.

```
Use a private group? [Y]es [No]:
```

Wir verwenden für die Gruppe den Standardnamen und bestätigen mit der Eingabetaste.

```
What would you like your private group to be called? [glftpd]:
```

Nun folgt die Eingabe der Nutzer, die auf den glFTPd Zugriff haben. Den einzigen Benutzer, den wir eingeben, ist root.

```
Who should have access to glftpd? (separate with ,):
```

Die nun folgende Abfrage verlangt von uns einen Verzeichnisnamen innerhalb des jail-Verzeichnisses, in das der glFTPd installiert werden soll. Auch hier wählen wir mit der Eingabetaste

```
Please enter the directory inside /jail to install glftpd to [/glftpd]:
```

Als Port für den glFTPd benutzen wir den Standardport 21.

```
Enter the port you would like glftpd to listen on [21]:
```

Nachdem das Installaionsscript nun die benötigten Dateien kopiert und kompiliert hat, müssen wir nun den von uns verwendeten inetd angeben, in unserem Fall ist dies der xinetd.

```
Do you want to use [I]netd or [X]inetd?
```

Zum Schluß muß nun noch das Zertifikat für die TLS-Verschlüsselung erstellt werden, für das wir einen beliebigen Namen wählen können.

```
Create ftpd-dsa.pem now? [Y/N]:
Please specify location, inside /jail/glftpd, to install
ftpd-dsa.pem to [/etc]:

Please specify a generic name for this certificate.
This can be any name but should say something about the ftp
server like the name for it perhaps? :
```

Damit ist die Installation abgeschlossen und der glFTPd kann verwendet werden.

6.4 Benutzer- und Gruppenverwaltung

Jetzt, da unser FTP-Server läuft, können wir Benutzer anlegen und diese in verschiedenen Gruppen unterteilen. Die Verwaltung der Benutzer in Gruppen beruht dabei auf dem gleichen Prinzip wie im Linux. So kann man beispielsweise die Lese- und Schreibrechte bestimmter Gruppen und damit auch deren Mitglieder in der Verzeichnisstruktur des FTP-Servers festlegen.

Um Benutzer und Gruppen anlegen zu können, muß man sich auf dem glFTPd als SiteAdmin einloggen. Der Standardbenutzer glftpd ist als SiteAdmin voreingestellt. Zunächst also der login in unseren FTP-Server.

```
ftp localhost
```

Der FTP-Client verbindet sich nun mit dem glFTPd. In der folgenden Eingabeaufforderung müssen wir Username und Passwort für den login eingeben+++

```
Connected to localhost.
220 MY SITE NAME (glftpd 1.32\_Linux+TLS) ready.
Name (localhost:user1):
```

Als Username und Passwort muss nun „glftpd“ eingegeben werden.

6.4.1 Gruppen und Benutzer anlegen

Nun sind wir auf dem FTP eingeloggt. Um nun eine neue Benutzergruppe anzulegen, benutzen wir den site-Befehl

```
site grpadd (Gruppenname) (Beschreibung)
```

Um einen neuen FTP User anzulegen verwenden wir den Befehl

```
site adduser (Username) (Passwort) (IP-Bereich)
```

Um einen User in eine Gruppe aufzunehmen, verwenden wir den Befehl

```
site chgrp (Username) (Gruppenname)
```

Wenn man einen neuen User anlegen will und diesen sofort in eine Gruppe aufnehmen möchte, kann man dies auch mit einem Befehl tun:

```
site gadduser (Gruppenname) (Username) (Passwort) (IP-Bereich)
```

Als IP-Bereich sind mehrere Einstellungen möglich, beispielsweise:

- username@192.168.1.10
- *@192.168.*
- *@*

Auf diese Weise kann man sicherstellen, das sich Benutzer nur von einer bestimmten IP bzw. einem bestimmten IP-Bereich aus verbinden können, beim letzten Beispiel ist die Verbindung von jeder beliebigen IP aus möglich.

6.4.2 Rechte

Die Rechteverwaltung des glftpd entspricht der von Linux. Es können also nur diejenigen Benutzer aus Verzeichnissen lesen und in sie schreiben, in denen sie oder ihre Gruppe berechtigt sind. So muß das standardmäßig vorhandene incoming-Verzeichnis zunächst in seinen Rechten angepasst werden, bevor jeder beliebige User darin eigene Verzeichnisse und Dateien anlegen kann.

vorher:

```
drwxr-xr-x   3 glftpd   glftpd       4096 Sep 29 12:48 incoming
```

nachher:

```
drwxrwxrwx   3 glftpd   glftpd       4096 Sep 29 12:48 incoming
```

Die Änderung der Zugriffsrechte erfolgt wie bei Linux mit dem Befehl chmod, in diesem Fall also:

```
chmod 777 incoming
```

Innerhalb des incoming-Verzeichnisses können nun alle Benutzer eigene Dateien und Verzeichnisse anlegen und verwalten.

6.4.3 Flags

Nachdem ein Benutzer angelegt wurde, kann der SiteOp diesem verschiedene Flags zuteilen. Diese dienen als Berechtigung zur Ausführung bestimmter site-Kommandos. Die möglichen Flags sind:

- 1: User ist SiteOP
- 2: User ist GroupAdmin in seiner ersten Gruppe
- 3: User kann die Gruppe nicht wechseln
- 4: Login auch bei vollem Server, kein Transferlimit
- 5: color-Befehl verwendbar oder nicht

- 6: User ist gelöscht
- 7: User ist „Co-SiteOp“
- 8: User ist anonymous
- A: User darf site NUKE verwenden
- B: User darf site UNNUKE verwenden
- C: User darf site UNDUPE verwenden
- D: User darf site KICK verwenden
- E: User darf site KILL/SWHO verwenden
- F: User darf site TAKE verwenden
- G: User darf site GIVE verwenden
- H: User darf site USER/USERS verwenden
- I: User darf unbegrenzt idle sein

6.4.4 Ratio und Credits

Die Ratio eines Users ist das Verhältnis seines Up- und Downloads. Standardmäßig ist die Ratio jedes Users auf 1:3 eingestellt, das heißt, für jedes KB was er auf den Server lädt, darf er 3 KB herunterladen. Stellt man die Ratio eines Users auf 0, so hat er eine unbegrenzte Downloadmenge, ohne dafür etwas uploaden zu müssen. In diesem Fall spricht man von einem leech-Account.

Wenn ein neuer User angelegt wird hat er zu Beginn Credits in Höhe von 15000 KB. Wenn er keinen leech-Account hat, ändert sich diese Zahl je nachdem, wieviel Upload und Download der User hat. Die Credits repräsentieren das effektive Downloadvolumen des Users auf dem Server.

6.5 SITE-Kommandos

Die site-Kommandos machen den glftpd zu einem überaus mächtigen und flexiblen Server, können in den falschen Händen jedoch recht gefährlich werden. Es ist deshalb ratsam, die Flags für die User mit Bedacht zu wählen, da nicht alle site-Kommandos für sie empfehlenswert sind. Der Aufruf der site-Kommandos erfolgt über „site (Kommando)“. Welche site-Kommandos man als User ausführen kann, erfährt man, wenn man im FTP folgenden Befehl eingibt:

```
site help
```

Eine komplette Auflistung aller site-Kommandos findet sich in der Dokumentation des glFTPd unter /jail/glftpd/docs/glftpd.docs. Hier nun eine Auflistung der wichtigsten site-Kommandos für den SiteOp:

6.5.1 Die wichtigsten SITE-Kommandos

- site ADDUSER - einen User hinzufügen
- site DELUSER - einen User löschen
- site CHANGE - einzelne Werte eines Users ändern
- site ADDIP - einem User eine IP hinzufügen
- site GADDUSER - User innerhalb einer Gruppe hinzufügen
- site RENUSER - einen User umbenennen
- site CHPASS - Passwort eines Users ändern
- site User - Alle eingerichteten User anzeigen
- site User (username) - Informationen zu einem User anzeigen
- site WHO - wer ist online?
- site GRPADD - eine Gruppe hinzufügen
- site GRPDEL - eine Gruppe löschen
- site GRPREN - eine Gruppe umbenennen
- site color - Color-Mode ein/aus schalten

7 Quellcodepakete kompilieren

Die Open-Source Gemeinde bietet eine Vielzahl frei erhältlicher Programme an. Diese sind oftmals in Form des Programm - Quellcodes verfügbar und können vom Nutzer selbst kompiliert werden. Das hat den Vorteil, dass das erzeugte Programm auf das System, auf dem es kompiliert wurde, optimal angepasst ist.

Benutzt man Linux als Betriebssystem, so kann es passieren, dass es von einem Programm oder vom Kernel eine aktuellere Version gibt. Möchte man nun nicht warten, bis der Distributor Updates für diese Komponenten bereitstellt, so hat man die Möglichkeit, sich die Sourcen des entsprechenden Programms herunterzuladen und diese selbst zu kompilieren und ins System einzubinden. Im folgenden Beispiel werden wir anhand des freien Mediaplayers mplayer einmal genauer betrachten, wie dies vonstatten geht.

7.1 Sourcen besorgen

Zuerst benötigen wir natürlich die Sourcen des mplayer. Bei Open - Source - Quellcode unterscheidet man zwischen drei Kategorien:

- **stable releases** Stabile, getestete Version der Software
- **development releases** Weiterentwickelte, jedoch möglicherweise instabile und nicht fehlerfreie Softwareversion
- **daily snapshots** Tägliches Abbild der momentan in der Entwicklung befindlichen Programm-version, die sich unter Umständen überhaupt nicht kompilieren lässt.

Den Quellcode des mplayer laden wir zunächst von der Projekthomepage herunter:

```
http://www.mplayerhq.hu
```

Dort finden wir unter Download die aktuelle MPlayer v1.0pre5 source. Nach dem Herunterladen der Datei MPlayer-1.0pre5.tar.bz2 entpacken wir diese. Hierzu ist es ratsam, im home - Verzeichnis einen eigenen Ordner für Programmsourcen anzulegen, beispielsweise /home/user1/src, in den auch zukünftig weitere Sourcen untergebracht werden können.

7.2 Kompilieren

Nachdem wir nun die erforderlichen Sourcen entpackt haben, benötigen wir die Konsole. Dort wechseln wir zunächst in das Verzeichnis der mplayer - Sourcen:

```
cd src/MPlayer-1.0pre5
```

Im folgenden Schritt müssen wir das makefile konfigurieren. Diese Datei enthält die für den Compiler notwendigen Informationen über das System, auf dem wir den mplayer kompilieren werden. Da dies von Hand sehr mühsam und aufwändig

ist, wird dieser Vorgang über ein Script erledigt. Dieses wird mit folgendem Befehl gestartet:

```
./configure
```

Nun konfiguriert das Script das makefile. Nachdem dieser Vorgang beendet ist können wir nun endlich mit dem eigentlichen Kompilierungsvorgang beginnen. Alles was nötig ist, um ihn zu starten, ist folgender Befehl:

```
make
```

Nun beginnt der Compiler damit, den Quellcode in ein ausführbares Programm zu übersetzen. Da dieser Vorgang eine ganze Weile dauert, können wir die Wartezeit damit verbringen, uns die lustigen Ausgaben des Compilers anzuschauen, die durch die Konsole rauschen.

Nachdem der Compiler seine Arbeit erledigt hat, müssen wir die erzeugten Binaries in das System einbinden. Da wir dazu root - Rechte benötigen, müssen wir zunächst in den SuperUser - Modus wechseln. Dies geschieht mittels

```
su
```

oder

```
sudo su
```

wobei letzterer Befehl voraussetzt, dass der User, der diesen Befehl aufruft, die entsprechenden Berechtigungen besitzt. Diese können wir in der Datei /etc/sudoers eintragen:

```
user1 ALL(ALL) NOPASSWD:ALL
```

Dies ermöglicht dem User, direkt in den SuperUser - Modus zu wechseln, ohne das root - Passwort eingeben zu müssen oder als User direkt Programme im root - Modus aufzurufen. Nun aber zurück zur Installation der erzeugten Binaries. Nach dem Wechsel in den root - Modus geben wir nun nur noch folgenden Befehl ein:

```
make install
```

Das Ergebnis des Kompiliervorgangs wird nun in unser Linux eingebunden. Damit ist die Installation komplett und wir können den mplayer nun benutzen.

8 Impressum und Quellen

8.1 Impressum

Diese Schulungsunterlage entstand im Zuge unserer Ausbildung zum Fachinformatiker für Systemintegration.

Autoren:

- Christoph Ketelsen (christoph@prozone.de)
- Florian Vogler (davor87@gmx.de)
- Nico Hänsel (info@bikebau.de)

8.2 Handout-Quellen

Dieses Handout wurde aus mehreren Manuals und Anleitungen zusammengestellt, welche gründlich auf Nutzbarkeit geprüft und überarbeitet wurden. Ein ausführliches Link-Verzeichnis unserer Quellen wollen wir hier nicht angeben, da Links in der heutigen Zeit zu schnell „veralten“. Wer also etwas sucht, bemühe einfach www.google.de.

8.3 Software-Quellen

- **Apache** Im Suse Linux Paket enthalten oder www.apache.org
- **Samba** Im Suse Linux Paket enthalten oder www.samba.org
- **MySQL** Im Suse Linux Paket enthalten oder www.mysql.org
- **Squid** Im Suse Linux Paket enthalten oder www.squid-cache.org
- **glFTPdr** www.glftpd.com
- **mplayer** www.mplayerhq.hu

Auf den Internetseiten könnt ihr immer die aktuelle Version herunterladen und findet dort auch umfangreiche Manuals zu den aktuellen Versionen.

8.4 Lizenzbestimmung