Linux im Netz

Pierre Burri
Michel Bisson

Linux im Netz

Inhaltsverzeichnis

TCP/IP....4
Warum TCP/IP so populär ist....4
Das ARPA Schichten-Model....4
Die Anwendungsschicht....5
Die Transportschicht....5
Die Internetschicht....5
Die Netzwerkschicht....5
Das OSI-Referenzmodell....5
Unterschiede zwischen UDP und TCP....7
ICMP - Internet Control Message Protocol....7
3-Wege-Handshake im TCP Protocoll....8
Was Sockets sind....9
Netzwerkkabel....9
Netzwerkkarte einrichten....10
Statisches Routing....13
Dynamisches Routing....15
TCP/IP Netzwerk-Klassen....16
Subnetz-Berechnung....17
Subnetze....18
Ein Netz unterteilen:....19
Übung mit einem großen Netzwerk....20
Einige wichtige Netzwerk-Konfigurationsdateien....21
/etc/HOSTNAME, /etc/hostname oder /etc/sysconfig/network....21
/etc/hosts....21
/etc/host.conf....21
/etc/nsswitch.conf....22
/etc/resolv.conf....22
DNS - Domain Name System....23
Zuerst ein bißchen Theorie:....23
Einen DNS-Server konfigurieren:....30
DNS-Testprogramme....34
nslookup....34
host....34
dig....34
Die Datei root.hint aktualisieren....34
Drucken unter Linux mit CUPS....35
Zwei wichtige Merkmale von CUPS:....35
CUPS-Verwaltung über den Browser....35
CUPS-Verwaltung über Kommandos....35
Neue PPD für einen Drucker erstellen....35
Andere CUPS-Treiber-Quellen....36
Einige CUPS-Befehle....36
Druckbefehle....36
Drucker und Jobs anschauen....36
Job löschen....36
Druckeroptionen anzeigen....36

Administration von CUPS als Server ... 36
CUPS mit Samba ... 37
CUPS-GUI-Werkzeuge ... 37
kprinter, qtcups & xpp ... 37
kups ... 38
Drucken mit LPD - das BSD Printing Spool System ... 38
TCP/IP Services ... 39
Der Internet Server oder Super Server - inetd ... 39
Der TCP Wrapper - tcpd ... 39
FTP ... 40
Textorientierte FTP-Klienten ... 40
Graphische FTP-Klienten ... 41
Einige FTP-Server ... 41
vsftpd - Very Secure FTP Server ... 41
Pure-FTPd ... 42
proFTPd ... 42
Der Shell-Zugang (ssh, telnet) für einen FTP-Benutzer blockieren: ... 43
Telnet ... 44
BSD-Remote Befehle ... 45
SSH - Secure Shell ... 46
Dateien und Verzeichnisse mit rsync und rdist synchronisieren ... 50
Weitere (veraltete) TCP/IP-Dienste ... 51
Finger ... 51
Talk ... 51
NFS - Network File System ... 52
DHCP - Dynamic Host Configuration Protocol ... 56
DHCP-Server ... 56
DHCP-Klienten ... 57
Netzwerk-Fehlersuche - Troubleshooting ... 58
Linux und Unix Bücher fürs Netzwerk ... 60

- ## TCP/IP

Transmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol

Warum TCP/IP so populär ist

(nach dem Buch "the Linux Network", M&T Books)

- TCP /IP wurde von der US Regierung / Pentagon finanziell unterstützt und verbreitet
- TCP/IP ist allgemeiner Besitz: es gehört der US Regierung also auch dem Volk, und nicht nur einer einzige Firma (kein Wettbewerb Problem)
- die TCP/IP Implementierung steht frei zur Verfügung. Der Berkerly Unix Code hat zwar ein Copyright, ist aber frei anwendbar.
- weil TCP/IP vor dem Hintergrund eines möglichen Atomkrieges entstand, hat das Internet keine zentrale Autorität. Niemand kann es plötzlich zerstören oder kontrollieren.
- das Internet hat zentrale Koordinationen - Federalisten Zugang.
- TCP/IP ist gut konzipiert. Es ist einfach besser als alle privaten Netzwerk-Protokolle, besonders die Art wie die "Namenauflösung" gemacht wird.

(nach dem Buch "Linux Intern", Data Becker)

- Hardware- und Softwareunabhängige Protokollspezifikationen.
- Unabhängigkeit von der Netzwerkhardware.
- Standardprotokolle in den höheren Schichten (z.B. Telnet, FTP)
- weltweit einheitliches Adressierungsmuster.

Das ARPA Schichten-Model

ARPA = **A**dvanced **R**esearch **P**roject **A**gency

Application / Anwendungsschicht (FTP, telnet, ssh, ping usw.)
Host-to-Host / Transportschicht (TCP, UDP, usw.)
Internet (IP) (Namen-Auflösung und Adressen-Protokoll)
Network Access / Netzwerkschicht (Ethernet, FDDI, PPP)

Die Anwendungsschicht

Anwendungsebene, Interaktion mit dem Benutzer. (Telnet, FTP, ssh)
Daten.

Die Transportschicht

Einpacken und auspacken der Daten nach einem bestimmten Protokoll :
die drei bekanntesten sind TCP, UDP und ICMP.
Quell- und Ziel-Port
für **TCP**: Sequenz-Nummer, Headerlänge, Flags (SYN, ACK),
window size (Puffergrösse des Empfängers),checksum.
für **UDP**: length, checksum

Die Internetschicht

(IP)

verantwortlich um das Paket zum Ziel zu bringen, routing
Quell- und Ziel-IP-Adresse
Protokollversion, Header checksum, Time to live, Headerlänge

Die Netzwerkschicht

(z.B. Ethernet)

Übersetzung von IP-Adressen in Hardwareadressen
Quell- und Ziel-Hardwareadresse
Protokolltyp: IP

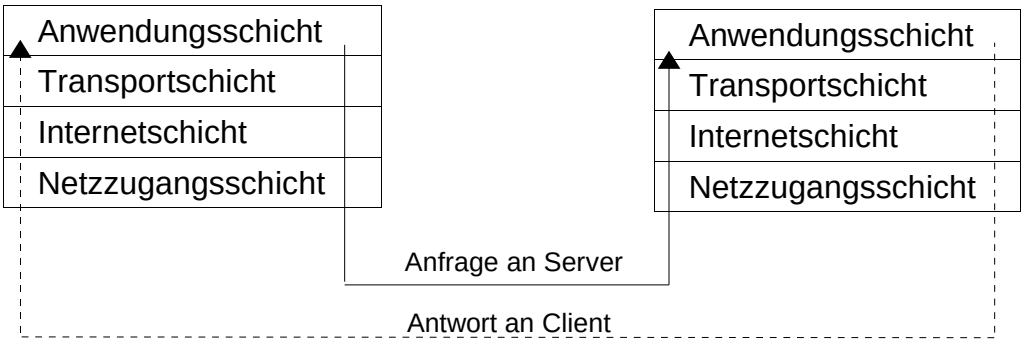
Das OSI-Referenzmodell

7.	Die Anwendungsschicht	Application Layer	
6.	Die Darstellungsschicht	Presentation Layer	conversion ebcdic-ascii
5.	Die Kommunikationsschicht	Session Layer	who is sender & receiver
4.	Die Transportschicht	Transport Layer	TCP, UDP, SPX...
3.	Die Vermittlungsschicht	Network Layer	IP, IPX...
2.	Sicherungsschicht	Data Link Layer	Fehlerbereinigung
1.	Physikalische Schicht	Physical Layer	

Beispiele von Protokollen:

Anwendungsschicht		***Transportschicht***
telnet	Telnet Protocol	TCP - Transmission Control Protocol
ftp	File Transfer Protocol	UDP - User Datagram Protocol
pop	Post Office Protocol	ICMP - Internet Control Message Protocol
smtp	Simple Mail Transfer Protocol	
http	Hypertext Transfer Protocol	
ssh	Secure Shell	
ipp	Internet Printing Protocol	

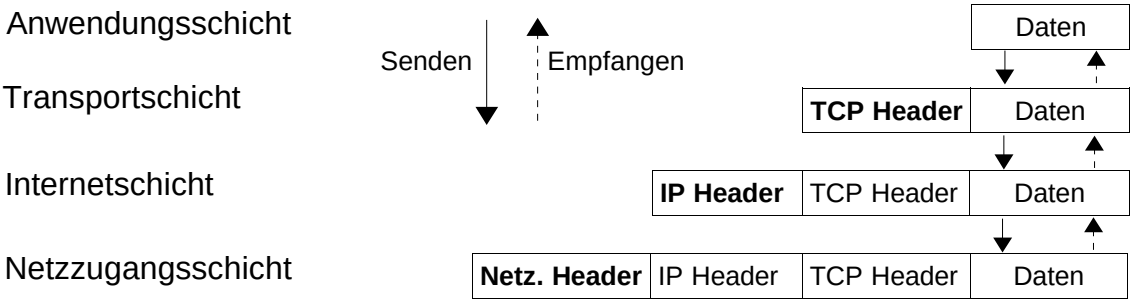
Datenweg zwischen Client und Server



Begriffe die in den verschiedenen Schichten von TCP/IP gebraucht werden:

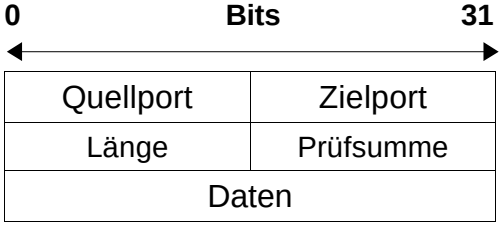
	TCP	**UDP**
Anwendungschicht:	Stream	Nachricht
Transportschicht:	Segment	Paket
Internetschicht:	Datagram	Datagramm
Netzzugangsschicht:	Frame	Frame

Der Datenaufbau mit den verschiedenen Headers:

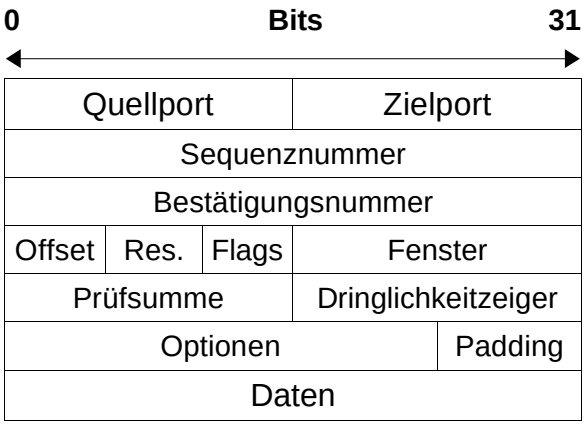


Headers auf der Transportschicht:

UDP:



TCP:



Header auf der Internetschicht:

Version	IHL	Type of Service	Gesamtlänge	
Identifikation			Flags	Fragmentation Offset
Time to Live		Protocol	Header-Prüfsumme	
Quelladresse				
Zieladresse				
Optionen				Padding
Daten				

Unterschiede zwischen UDP und TCP

(aus Linux im Netz von Dr. B. Röhrig):

UDP:
- ein verbindungsloses Protokoll
- es stellt nicht sicher, daß versendete Pakete beim Empfänger ankommen beziehungsweise in der richtigen Reihenfolge ankommen
- dafür sehr schnell und erzeugt eine geringere Netzbelastung als TCP
- eignet sich gut für hochgeschwindigkeitsanwendungen wie verteilte Dateisysteme (NFS)

TCP:
- verbindungsorientiertes Protokoll: erstellt voll duplexfähige, bidirektionale Verbindung
- gewährleistet einen sicheren Transport der Daten im Netzwerk
- sorgt dafür, daß kein Datenpaket verlorengeht und dass alle Pakete in der richtigen Reihenfolge beim Empfänger ankommen: durch Sequenznummer, Prüfsummenbildung mit Empfangsquittungen, Quittung mit Zeitüberwachung, Segment-Wiederholung usw.
- aus Benutzersicht Übertragen als Datenstrom, nicht blockweise
- wird unter anderem für interaktive Verbindungen zwischen zwei Rechnern verwendet (z.B. telnet,ssh)

ICMP - Internet Control Message Protocol

Diese Protokoll hat die Aufgabe, Fehler und Diagnoseinformationen für das IP im Netz zu transportieren.

Wichtige Felder dieses Protokolls:

Type	Nachrichtentyp
Code	Weitere Spezifizierung von Type

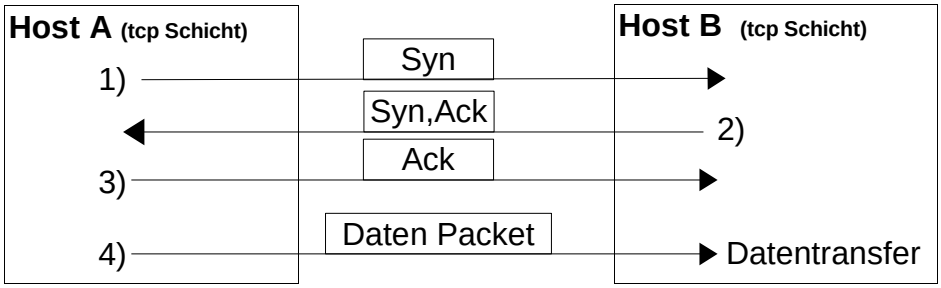
einige ICMP-Types:

0	= echo reply	Antwort auf ein echo request
3	= destination unreachable	Datagramm nicht zustellbar
4	= source quench	Puffer Kapazität des Empfängers erreicht (kann pakete nicht so schnell verarbeiten)
5	= redirect message	Vorschlag: Routenwechsel zur Wegverkürzung
8	= echo request	(ping)
11	= time exceeded	Lebenszeit des Pakets ist abgelaufen

einige ICMP-Codes im Zusammenhang mit Type 3:

0 = network unreachable
1 = host unreachable
3 = port unreachable
6 = destination network unknown
7 = destination host unknown

3-Wege-Handshake im TCP Protocoll



Portnummer und die "Well Known Ports"

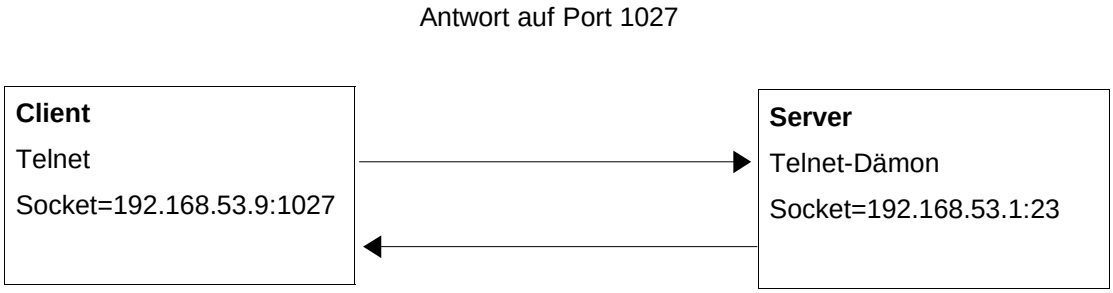
Beschreibung der "Well Known Ports" unter Linux: Datei `/etc/services`

z.B.

```
ftp-data        20/tcp          # File Transfer [Default Data]
ftp-data        20/udp          # File Transfer [Default Data]
ftp             21/tcp          # File Transfer [Control]
fsp             21/udp          # UDP File Transfer
ssh             22/tcp          # SSH Remote Login Protocol
ssh             22/udp          # SSH Remote Login Protocol
telnet          23/tcp          # Telnet
telnet          23/udp          # Telnet
smtp            25/tcp  mail    # Simple Mail Transfer
smtp            25/udp mail     # Simple Mail Transfer
www             80/tcp          # Worl Wide Web HTTP
www             80/udp          # Worl Wide Web HTTP
pop3            110/tcp         # Post Office Protocol - Version 3
pop3            110/udp         # Post Office Protocol - Version 3
imap            143/tcp  imap2  # Internet Message Access Protocol
imap            143/udp imap2   # Internet Message Access Protocol
ipp             631/tcp         # IPP (Internet Printing Protocol)
ipp             631/udp         # IPP (Internet Printing Protocol)
```

Was Sockets sind

Socket = IP-Adresse + Port



• Netzwerkkabel

Thin coaxial (auch thin coax oder 10base-2 Ethernet genannt)

- 50 Ohms UG 88 BNC-Stecker, RG 58 U Kabel, BNC-T-Adapter, BNC-Endstecker (Terminator).
- Maximale Länge: 185m dann muß ein Repeater eingefügt werden.
- Maximal 30 Stationen pro Segment.
- Maximale Übermittlungsgeschwindigkeit: 10 Megabits /Sek.
- + gut geschützt vor äußeren Störungen
- + billig
- -- ein Unterbruch lähmt das ganze Netzwerk.

Twisted pair (auch 10base-T Ethernet genannt)

- RJ-45 Stecker, Abgeschirmten 8 Poligen-Kabel
- Ethernet Hub (€ >= 35,- 10MBits/sec) oder Switch (teurer)
- Maximale länge: 90m bis 100m
- Maximal 1024 Stationen
- Maximale Übermittlungsgeschwindigkeit: 100 Megabits /Sek.
- ++ leicht zu verwalten
- - nicht geschützt vor äußeren Störungen
- - teurer als thin coax, ein kleiner 10Mbits/sec Hub kostet min. 35,- €, (Wenn schon, wäre ein Switch sinnvoller, der kostet ab ca. 50,- €)

FDDI - Fiber Distributed Data Interface

- 100 Megabits / Sek.
- Maximale länge: 200Km
- über Glasfaser oder Kupfer
- mit "token ring" Technologie
- Zwei Karten werden unter Linux unterstützt: die DEFPA (pci) und die DEFEA (isa) von Digital (Compaq)

Gigabit Ethernet

- 1000 Megabits / Sek.
- Treiber für Alteon AceNIC, 3Com 3C985, Yellowfin Gigabit-NIC, SysKonnect SK-98xx und Himachi GNIC-II

- ## Netzwerkkarte einrichten

  - **Netzwerkkarte mit YaST1 installieren** (bis SuSE 7.3)**:**

    yast - Administration des Systems ->
    Hardware in System integrieren -> Netzwerkkarte konfigurieren
    Typ des Netwerks : eth0
    Art der Netzwerk-Karte: [NE 2000 / NE 1000 (ISA)]
    Optionen zum Laden des Moduls
    :io=0x300 (für eine RealTek RTL8129/8139: module rtl8139)

    < Weiter >

  - **Netzwerkkarte mit YaST2 installieren:**

    yast2 über die KDE-Leiste: K - SuSE - System - Konfiguration - YaST2
    Netzwerkwerk/Basis anklicken
    Konfiguration der Netzwerkkarte anklicken.

    Der Vorteil von YaST2 gegenüber YaST1 ist die automatische Hardware-Erkennung, mindestens wenn es funktioniert...

  - **Eine Netzwerkkarte unter RedHat einrichten:**

    `/usr/sbin/netconfig [-d` *`Devicenamen`*`]`
    Konfigurationsdateien:
    `/etc/sysconfig/network-sripts/ifcfg-`*`Devicenamen`*

  - **Netzwerk-Modulen in der Datei `/etc/modules.conf` anschauen:**

    ```
    alias eth0 ne
    options ne      io=0x300
    ```

  - **Die geladene Treibermodule anschauen**

    `lsmod`

  - **Mehr Informationen über ein Modul bekommen**

    `modinfo` *`Modulnamen`* z.B. `modinfo 8139too`

  - **Einen Triebermodul laden oder entfernen**

    `modprobe [-r]` *`Treibername`* (bevor das Modul geladen wird, überprüft modprobe IRQ, IO-Port und ob das Modul nicht schon geladen ist.
    Treiberame = z.B. ne2k-pci, ne, 8139. `-r` für remove = entfernen)

    `insmode` *`Treibername`* (versucht der Treiber/das Modul zu laden)
    `rmmode` *`Treibername`* (entfernt der Treiber/das Modul)

  - **Abhängigkeiten und Pfade der Modulen:**

    Es ist möglich ein Modul/Treiber über `modprobe` direkt mit dem Namen zu laden oder zu entfernen, weil die Datei `/lib/modules/$(uname -r)/modules.dep` die effektiven Pfade und Abhängigkeiten unter den Modulen definiert.

- **Eine Netzwerkkarte mit ifconfig anschauen und einrichten**
  - **Die Netzwerkkonfiguration anschauen:**

    `ifconfig` (zeigt alle Netzwerkkarten an)
    `ifconfig` *`Device`* (zeigt nur eine bestimmte Netzwerkkarte an, Device = lo, eth0, eth1, ippp0, ippp1, ppp0 usw.)
  - **Eine Netzwerkkarte einrichten:**
    `ifconfig` *`Gerät IP-Adresse`* `[netmask` *`Netmaske`*`] [up]`

    z.B. `ifconfig eth0 192.168.10.14 netmask 255.255.255.0 up`
    oder nur `ifconfig eth0 192.168.10.10`
    nachdem die Karte konfiguriert ist, kann sie gestoppt oder gestartet werden
    `ifconfig eth0 down`
    `ifconfig eth0 up`
  - **Eine virtuelle Netzwerkkarte konfigurieren:**
    `ifconfig eth0:`*`xxxx <IP-Adr.>`* `[netmask <255.255.255.0>] [up]`
  - **Eine virtuelle Netzwerkkarte mit YaST1 einrichten:**
    (nur bis SuSE 7.3)
    Administration des Systems -> Netzwerk konfigurieren ->
    Netzwerk Grundkonfiguration
    (Nächste freie Nummer mit dem Cursor auswählen z.B. [1])
    F5=Device Ethernet oder <Andere Device eingeben> z.B. eth0:0
    F6=IP-Adresse IP-Adresse Ihres Rechners: 192.168.xx.yy
    F4=Aktivieren
    F10=Speichern
  - **Eine virtuelle Netzwerkkarte ab SuSE 8.0 einrichten:**
    Im Verzeichnis `/etc/sysconfig/network` wechseln
    Die Datei `ifcfg-eth0` nach z.B. `ifcfg-eth0:0` kopieren
    Die folgenden Parameter anpassen:
    `BROADCAST='192.168.xx.255'`
    `IPADDR='192.168.xx.yy'`
    `NETWORK='192.168.xx.0'`
    Das Netzwerk mit `rcnetwork restart` neu starten. Die Fehlermeldung `interface eth0:10 is not available` können Sie ignorieren.
  - **Nur als Info, andere ifconfig Parametern:**
    broadcast *addr ,* pointopoint *addr ,* io_addr *addr ,* irq *addr*
    MTU = Maximum Transfer Unit in Bytes =>1500 für Ethernet
    metric = nur für dynamic routing => Anzahl von Gateways / Hops
    Multicast = erlaubt das Broadcasting über alle Intranets
  - **Das Loopback-Device**
    Für die interne Kommunikation und Testzweck, gibt es das `Loopback-Device`.
    Das Loopback-Device (Devicenamen `lo`) hat die IP-Adresse `127.0.0.1` und den Rechnernamen `localhost`.

- **Das Dummy-Device**
  Ein `Dummy-Device` (dummy = Attrape, Schein...) einzurichten ist interessant wenn eine richtige IP-Adresse gebraucht wird, aber keine Netztwerkkarte vorhanden ist. z.B. Tests von Apache oder einen DNS-Server.
  `ifconfig dummy0` *`IP-Adresse`*

  Über YaST (bis SuSE 7.3):
  Administration des Systems -> Konfigurationsdatei verändern
  SETUPDUMMYDEV = yes (+ IPADDR_0)

  Als Kernel-Parameter:
  Netzwerkkartentreiber - CONFIG_DUMMY=m oder y

- **Das Netzwerk stoppen / starten**
  bis SuSE 7.0: `init 1` = Netzwerk stoppen `init 2 (5)` = Netzwerk starten
  ab SuSE 7.1: `init 1` = Netzwerk stoppen `init 3 (5)` = Netzwerk starten

  oder: `/etc/init.d/network restart` oder `rcnetwork restart`

  RedHat: `init 1` zum stoppen und `init 3` und `init 5` um zu starten

- ## Statisches Routing
  - **Routen anschauen:**
    `route`
    `route -n` (-n = nur IP-Adressen zeigen statt die Rechnernamen, viel schneller!)
    `netstat -r`

    Flags Beschreibung: U=Up, H=das Ziel ist ein Host, G=Gateway
  - **Statische Route permanent einfügen:**
    - Bei SuSE (bis Version 7.3) sind die Routen in der Datei `/etc/route.conf`

      z.B.:

Ziel	Gateway	Netmaske	Device
192.168.10.0	0.0.0.0	255.255.255.0	eth0
192.168.2.0	192.168.10.1	255.255.255.0	eth0
192.168.67.0	0.0.0.0	255.255.255.0	eth0:0
192.168.3.2	0.0.0.0	255.255.255.0	ippp0
192.168.4.2	0.0.0.0	255.255.255.0	ippp1
default	192.168.10.253		eth0

    - **Die SuSE-Routing-Tabelle aktualisieren** (nur bis SuSE 7.3)**:**
      `rcroute start | stop`
      oder `/etc/init.d/route start | stop`
    - **Die Routen über YaST2 eingfügen** (ab SuSE 8.0)**:**
      yast2 - Netzwerk/Erweitert - Routing -☑ Konfiguration für Experten - Hinzufügen - dann zum Beispiel:

Zeil:	192.168.2.0
Dummy oder Gateway	192.168.10.1
Netmaske	255.255.255.0
Gerät (optional)	eth0

      auf OK klicken -Beenden - Schliessen.

      YaST2 trägt die neue Route in der Datei `/etc/sysconfig/network/routes` ein.
    - **Die Routen manuell eintragen** (ab SuSE 8.0):
      Die Datei `/etc/sysconfig/network/routes` editieren und mindestens 4 Spalten pro Routen müssen eingetragen werden:

Zieladresse	Gateway	Netmaske	Gerät
z.B.			
`192.168.30.0`	`192.168.10.14`	`255.255.255.0`	`eth0`
`192.168.20.0`	`192.168.10.14`	`255.255.255.0`	`eth0`
`default`	`172.19.13.214`	-	-

      Die Spalten können auch mit einem " " ersetzt werden. Für eine Route die über eine virtuelle Netzwerkkarte geht, kann man das Gerät mit einem " " angeben.

      z.B.
      `192.168.30.0 192.168.10.14 255.255.255.0 -`

- **Permanente Statische Routen unter RedHat zufügen**

  Routen können mit dem Administrationswerkzeug `linuxconf` zugefügt werden:

  linuxconf -Verwaltung -Netzwerk - Routing und Gateway - Einstellen Routen zu anderen Netzwerken Neu
  Gateway 192.168.10.1
  Ziel 192.168.2.0
  Netmask (opt.)
  Bestätigen - Dismiss - Aktivieren - ..(Do it)... - Beenden.

  Die Routen werden in der Datei `/etc/sysconfig/static-routes` mit dem folgedem Format geschrieben:

  ```
  eth0 net 192.168.2.0 netmask 255.255.255.0 gw 192.168.10.1
  eth0 net 0.0.0.0 netmask 0.0.0.0 gw 192.168.10.15
  ```

- **Eine Route über einen Router/Gateway zufügen:**
  `route add -net 192.168.2.0 netmask 255.255.255.0 gw moon dev eth0`

  Erklärung: alle unbekannten TCP & UDP Pakete für das Netz 192.168.2.0 werden an den Gateway "moon" und über die Netzwerkkarte eth0 weitergeleitet.

- **Einen einzigen Host/Rechner zufügen:**
  `route add -host 192.168.10.10 dev eth0`
  oder `route add -host 192.168.10.10 netmask` **`0.0.0.0`** `dev eth0`

- **Einen Default Gateway (Router) zufügen / entfernen:**
  `route add default gw 192.168.70.9 [dev eth0]`
  `route del default`

  Erklärung: alle unbekannten TCP & UDP Pakete für das Internet (dynamische IP, also noch unbekannte IPs) werden an den Gateway "sirius" und über die Netzwerkkarte eth0 weitergeleitet.

- **Eine Route verfolgen:** (zeigt wieviel Gateways und Hops nötig sind)
  `traceroute` *`Zielrechner`*
  `traceroute -s` *`IP-Adresse_des_Senders Zielrechner`*

  z.B. traceroute www.suse.de

- **Netzwerkadresse-Hardwareadresse mit ARP auflösung**
  (ARP = Address **R**esolution **P**rotocol**)**
  `arp` **Wichtig:** zuerst noch ein ping 192.168.xx.255 ausführen!
  `arp -a` `(a=all)`
  `cat /proc/net/arp`

  Flags-Beschreibung: C=Complete Entry, M=Permanent Entry, P=Published Entry

  `arp -s` *`hostname hardwareadresse`* = Eintrag einfügen
  `arp -d` *`hostname`* = Eintrag löschen
  `arping -D` *`Hostname`* erlaubt IP-Adressen-Probleme zu entdecken!

- **Das IP-Forwarding dynamisch ein- und ausschalten**

  Der Forwarding-Zustand anschauen: `cat /proc/sys/net/ipv4/ip_forward`
  Das Forwarding einschalten: `echo 1 > /proc/sys/net/ipv4/ip_forward`
  Das Forwarding ausschalten: `echo 0 > /proc/sys/net/ipv4/ip_forward`

• Dynamisches Routing

RIP-Protokoll für internes Routing (**R**outing **I**nformation **P**rotocol)
BGP, EGP & OSFP sind externe Routingprotokolle (fürs Internet)

Programm **routed**:
- (passiv) lauscht am UDP-Port 520 nach "routing informationen packets"
- (aktiv) sendet seine eigene Routingtabelle an alle direkt angeschlossenen Netze
- pflegt dynamisch die eigene Kernel-Routingtabelle (Zeitverzögerung 30 sec)

Start von routed:
Variable `START_ROUTED = yes`
für einen Router muss noch `IP_FORWARD=yes` sein !
oder `echo 1 > /proc/sys/net/ipv4/ip_forward`

```
rcrouted start | stop | restart
```

zusätzlich können statische Routen in `/etc/gateways` eingetragen werden.
Die Routen werden beim starten von routed gelesen und in der Kernel Routingtabelle eingetragen.

Format:
```
<net | host> name1 gateway name2 metric value <passive | active | external>
```
z.B.
```
net 192.168.2.0 gateway 192.168.10.1 metric 1 passive
```

passive = der Gateway tauscht keine Routing Informationen aus
active = der Gateway hat einen Routing-Dämon
extern = ist wie `passive` aber es wird kein Eintrag in der Kernel Routingtabelle eingetragen.

Programm **gated**:
- für routing im Internet
- erhältlich unter `www.gated.org` (von NextHop Technologies)
- konfigurierbar mit der Datei `/etc/gated.conf`
- RIP-Diagnose mit dem Befehl `ripquery -r` *`hostname`*

• TCP/IP Netzwerk-Klassen

TCP = **T**ransmission **C**ontrol **P**rotocol
IP = **I**nternet **P**rotocol

Netzwerkadressen:

Klasse A	**1.**	0.	0.	0.	-	**127.**	255.	255.	255
Klasse B	**128.**	**0.**	0.	0.	-	**191.**	**255.**	255.	255
Klasse C	**192.**	**0.**	**0.**	0.	-	**223.**	**255.**	**255.**	255

Reservierte Adressen:

Klasse A	**10.**	0.	0.	0.	-	**10.**	255.	255.	255
Klasse B	**172.**	**16.**	0.	0.	-	**172.**	**31.**	255.	255
Klasse C	**192.**	**168.**	**0.**	0.	-	**192.**	**168.**	**255.**	255

Klasse D 224 - 239 (Multicasting)
Klasse E 240 - 255 (Internet Eigenbedarf)

Anzahl der Netzwerke und Computer:

	Netzwerke	**Computer**
Klasse A	125	2^{24}= 16'777'216
Klasse B	16'382	2^{16}= 65'536
Klasse C	2'097'150	2^{8} = 256
Total	2'113'658	3'724'410'368

Berechnung des Netmasks:

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
bit:	|	|	|	|	|	|	|	|	
	128	**64**	**32**	**16**	**8**	**4**	**2**	**1**	
Netmask									Computers *
128	|								128-2=126
192	|	|							64-2=62
224	|	|	|						32-2=30
240	|	|	|	|					16-2=14
248	|	|	|	|	|				8-2=6
252	|	|	|	|	|	|			4-2=2
254	|	|	|	|	|	|	|		2
255	|	|	|	|	|	|	|	|	1

* -2 weil: 192.168.x.0=Netzwerkadresse & 192.168.x.255=Broadcast sind reserviert

- **Subnetz-Berechnung**

4	4	8	16	32	64
0 - 3	128 - 131	0 - 7	0 - 15	0 - 31	0 - 63
4 - 7	132 - 135	8 - 15	16 - 31	32 - 63	64 - 127
8 - 11	136 - 139	16 - 23	32 - 47	64 - 95	128 - 191
12 - 15	140 - 143	24 - 31	48 - 63	96 - 127	192 - 255
16 - 19	144 - 147	32 - 39	64 - 79	128 - 159	
20 - 23	148 - 151	40 - 47	80 - 95	160 - 191	
24 - 27	152 - 155	48 - 55	96 - 111	192 - 223	
28 - 31	156 - 159	56 - 63	112 - 127	224 - 255	
32 - 35	160 - 163	64 - 71	128 - 143		
36 - 39	164 - 167	72 - 79	144 - 159		
40 - 43	168 - 171	80 - 87	160 - 175		
44 - 47	172 - 175	88 - 95	176 - 191		
48 - 51	176 - 179	96 - 103	192 - 207		
52 - 55	180 - 183	104 - 111	208 - 223		
56 - 59	184 - 187	112 - 119	224 - 239		
60 - 63	188 - 191	120 - 127	240 - 255		
64 - 67	192 - 195	128 - 135			
68 - 71	196 - 199	136 - 143			
72 - 75	200 - 203	144 - 151	**Berechnungsmethode:**		
76 - 79	204 - 207	152 - 159	Host/Netmask: 192.168.10.102 / 255.255.255.192		
80 - 83	208 - 211	160 - 167	255 - Netmask + 1 = 255 - 192 + 1 = 64		
84 - 87	212 - 215	168 - 175	Jetzt der letzter Block der IP-Adresse des Hosts im 64.		
88 - 91	216 - 219	176 - 183	Block suchen: 102 im 64. Block = 64 - 127		
92 - 95	220 - 223	184 - 191	Netzwerkadresse = 64		
96 - 99	224 - 227	192 - 199	Broadcastadresse = 127		
100 - 103	228 - 231	200 - 207	Rechnerbereich = 65 - 126		
104 - 107	232 - 235	208 - 215			
108 - 111	236 - 239	216 - 223			
112 - 115	240 - 243	224 - 231			
116 - 119	244 - 247	232 - 239			
120 - 123	248 - 251	240 - 247			
124 - 127	252 - 255	248 - 255			

Standard	CIDR	Standard	CIDR	Standard	CIDR	Standard	CIDR
128.0.0.0	/1	255.128.0.0	/9	255.255.128.0	/17	255.255.255.128	/25
192.0.0.0	/2	255.192.0.0	/10	255.255.192.0	/18	255.255.255.192	/26
224.0.0.0	/3	255.224.0.0	/11	255.255.224.0	/19	255.255.255.224	/27
240.0.0.0	/4	255.240.0.0	/12	255.255.240.0	/20	255.255.255.240	/28
248.0.0.0	/5	255.248.0.0	/13	255.255.248.0	/21	255.255.255.248	/29
252.0.0.0	/6	255.252.0.0	/14	255.255.252.0	/22	255.255.255.252	/30
254.0.0.0	/7	255.254.0.0	/15	255.255.254.0	/23	255.255.255.254	/31

```
255.0.0.0 /8     255.255.0.0 /16    255.255.255.0 /24    255.255.255.255 /32
```

- **Subnetze**

1. Den relevanten Rechner-Teil in binär konvertieren
2. Den Netmasken-Teil, der nicht gleich 255 ist, in binär konvertieren
3. Die beiden Resultate übereinander schreiben (rechtsbündig)
4. Einen Strich nach den Einern der Netzmaske ziehen
5. Den Teil der Rechner-Zahl, die sich vor dem Strich befindet, 2x kopieren
6. mit "0" für die Netzwerkadresse füllen
7. mit "1" für die Broadcastadresse füllen
8. beide Resultate wieder in dezimal zurück konvertieren

z.B. Rechner 192.168.10 .35
Netmask 255 .255.255.240

```
35 =        1 0 | 0 0 1 1
240 = 1 1 1 1 | 0 0 0 0
--------------------------
              1 0 | 0 0 0 0  => Netzwerkadresse  = 32
              1 0 | 1 1 1 1  => Broadcastadresse = 47
```

mögliche Rechneradressen: 33 bis 46

z.B. Rechner 192.168.10 .70
Netmask 255 .255.255.192

```
70 =   0 1 | 0 0 0 1 1 0
192 = 1 1  | 0 0 0 0 0 0
--------------------------
           0 1 | 0 0 0 0 0 0  => Netzwerkadresse  = 64
           0 1 | 1 1 1 1 1 1  => Broadcastadresse = 127
```

mögliche Rechneradressen: 65 bis 126

Übung:
Rechneradresse = 192.168.2.251 Netmaske = 255.255.255.224
Resultat: Netz =
Broadcast =
Rechner =

Rechneradresse = 192.168.190.115 Netmaske = 255.255.255.128
Resultat: Netz =
Broadcast =
Rechner =

Rechneradresse = 192.168.190.178 Netmaske = 255.255.255.128
Resultat: Netz =
Broadcast =
Rechner =

- **Ein Netz unterteilen:**

z.B. Produktionabteilung = 90 Leute / Rechner
Marketing = 13 Leute / Rechner
Sekretariat & Leitung = 5 Leute / Rechner

Netzwerk 192.168.23.x

90 Leute passen in eine Gruppe von 128/126, N.M. = 128
13 Leute passen in eine Gruppe von 16/14 (zu knapp, also 32), N.M = 224
5 Leute passen in eine Gruppe von 8/6 (auch zu knapp, also 16) N.M. = 240

Produktion (128):	N.A. = 0	B.A. = 127	Hosts = 1 - 126
Marketing (32):	N.A. = 128	B.A. = 159	Hosts = 129 - 158
Sekretariat 16):	N.A. = 160	B.A. = 175	Hosts = 161 - 174

Rechnung:

1. Größte Gruppe -> B.A. 128 - 1 = 127
2. Zweit größte Gruppe -> B.A. 128 + 32 -1 = 159
3. Dritt größte Gruppe -> B.A. 160 + 16 - 1 = 175

2. Beispiel (Netzwerk Klasse C)

Gruppe A	= 120	Netzgruppe = 128	N.M. = 128
Gruppe B	= 4	Netzgruppe = 16	N.M. = 240
Gruppe C	= 52	Netzgruppe = 64	N.M. = 192
Gruppe D	= 25	Netzgruppe = 32	N.M. = 224

Gruppe A (128):	N.A. = 0	B.A. = 127	Hosts = 1 - 126
Gruppe C (64):	N.A. = 128	B.A. = 191	Hosts = 129 - 190
Gruppe D (32):	N.A. = 192	B.A. = 223	Hosts = 193 - 222
Gruppe B (16):	N.A. = 224	B.A. = 239	Hosts = 225 - 238

Übung:

Gruppe A	= 77	Netzgruppe =	N.M. =
Gruppe B	= 8	Netzgruppe =	N.M. =
Gruppe C	= 15	Netzgruppe =	N.M. =
Gruppe D	= 21	Netzgruppe =	N.M. =

Gruppe A (128):	N.A. =	B.A. =	Hosts =
Gruppe C (32):	N.A. =	B.A. =	Hosts =
Gruppe D (32):	N.A. =	B.A. =	Hosts =
Gruppe B (16):	N.A. =	B.A. =	Hosts =

- **Übung mit einem großen Netzwerk**

  Subnetz mit 17.000 Leute / Rechner

Anz. Abt.	Angestellten	(Min Größe)	Effekt. Größe	Total Adressen
1	7,000	/ 256 = 28	32	1 x 32 = 32
8	450	/ 256 = 2	2 (besser 4)	8 x 4 = 32
7	400	/ 256 = 2	2 (besser 4)	7 x 4 = 28
10	300	/ 256 = 2	2	10 x 2 = 20
4	100	/ 256 = 1	1	4 x 1 = 4
6	50	/ 256 = 1	1	6 x 1 = 6
2	20	/ 256 = 1	1	2 x 1 = 2

Total:				= 124 Adressen

Größe	**Netmask**	**Addressbereich**
32	255.255.**224.0**	192.168.**0.0** - 192.168.**31.255**
	8 + 7=15 x 4 Adressen	
4	255.255.**252.0**	192.168.**32.0** - 192.168.**35.255**
4	255.255.**252.0**	192.168.**36.0** - 192.168.**39.255**
		..... - .....
		bis..... - 192.168.**91.255** (32+(15x4)-1)
	10 x 2 Adressen	
2	255.255.	192.168. - 192.168.
2	255.255.	192.168. - 192.168.
		..... - .....
		bis..... - 192.168.
	4+6+2=12 x 1 Adresse	
1	255.255.	192.168. - 192.168.
1	255.255.	192.168. - 192.168.
		..... - .....
		bis..... - 192.168.

- ## Einige wichtige Netzwerk-Konfigurationsdateien

  - ### /etc/HOSTNAME, /etc/hostname oder /etc/sysconfig/network

    Der Rechnername mit oder ohne Domainname ist je nach Distribution in einer der folgenden Dateien:

    /etc/HOSTNAME für SuSE

    /etc/hostname für Debian

    /etc/sysconfig/network für RedHat

    z.B. sirius.stars.priv

    mit YaST: Administration des Systems -> Netzwerk konfigurieren ->
    Rechnername ändern

    Rechnername : sirius
    Domainname : stars.priv

    < Weiter >

    mit YaST2: Netzwerk/Erweitert - Hostname und Domainname - Beenden.

  - ### /etc/hosts

    Lokale Renchername-Datenbank, Windows kennt auch die Datei HOSTS.

    IP-Adresse *vollständiger Rechnername Aliases
    z.B.

127.0.0.1	localhost			
192.168.10.1	moon.stars.priv	moon		
192.168.10.10	sirius.stars.priv	sirius	news	isdn

    *vollständiger Rechnername = Fully Qualified Hostname

    Achtung!! Das Programm `SuSEconfig` verändert die Datei `/etc/hosts`, wenn die Variablen `CHECK_ETC_HOSTS` und `BEATIFY_ETC_HOSTS` auf `yes` gesetzt sind.

    Bis SuSE 7.3 befinden sich diese Variablen in `/etc/rc.config`,
    ab SuSE 8.0 in `/etc/sysconfig/suseconfig`.

  - ### /etc/host.conf

    Diese Datei definiert die Reihenfolge der Dienste, die zur Namenauflösung angefragt werden. Diese Datei wird nur noch von älteren Programmen, die mit der C-Bibliotheken `libc4` oder `libc5` gelinkt sind, benutzt.

    z.B.

    order hosts, bind
    multi on

    einige mögliche Parameter:

`order`	`hosts`	= die Datei /etc/hosts
	`bind`	= DNS Server
	`nis`	= Yellow Pages / Network Information System
`multi`	`on`	= mehrere Namen möglich für Rechner in `/etc/hosts`
	`off`	= nur einen einzigen Namen pro Rechner möglich

- **/etc/nsswitch.conf**

  `nsswitch` steht für Name Service Switch (ursprünglich von Sun/Solaris).
  Diese Datei hat einen ähnlichen Zweck wie die Datei `/etc/hosts.conf` aber in einer allgemeineren Form. Viel mehr Dienste können damit definiert werden. Ab der Version 2 der GNU-C-Bibliothek (`glibc2`), ersetzt diese Datei die `/etc/host.conf`. Früher war diese Datei unter SuSE nur im Zusammenhang mit dem Paket ypserv (NIS) vorhanden.

  Datei-Format (siehe man 5 switsch.conf):

```
passwd:         files nis
group:          files nis

hosts:          files dns
networks:       files dns

services:       files
protocols:      files
rpc:            files
ethers:         files
```

- **/etc/resolv.conf**

  Diese Datei definiert welcher DNS-Server für die Namenauflösung benutzt wird. Sie ist für eine PPP-Verbindung (Modem / ISDN / ADSL) wichtig, und auch wenn einen lokalen DNS-Server vorhanden ist.

  z.B.

```
# /etc/resolv.conf
#
domain  stars.priv
search   stars.priv  elop.temp.berlin
nameserver 192.168.10.10
nameserver 192.168.10.1
```

  domain lokale oder eigene Domäne, fügt diese Domäne an einen einfachen Rechnernamen (muß nicht unbedingt angegeben werden)
  search ähnlich wie "domain", aber eine ganze Reihenfolge von Domänen kann angegeben werden. Verlangsamt die Auflösung.
  nameserver IP-Adresse des DNS-Servers, lokal oder entfernt im Fall einer PPP-Verbindung. Maximal 3 nameserver können angegeben werden.

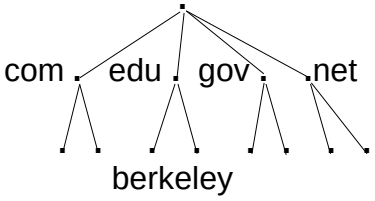
  bei SuSE wird diese Datei temporär ersetzt, wenn man eine PPP-Verbindung aufbaut, um die Name-Server des ISP einzusetzten.

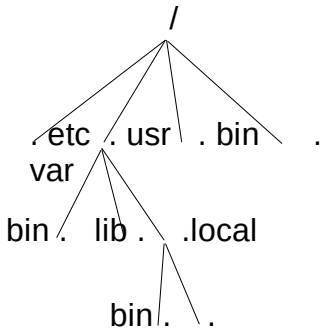
- ## DNS - Domain Name System

Zuerst ein bißchen Theorie:

(Online-Dokumentation DNS-HOWTO)

- **Aufbau der DNS Struktur im Vergleich zu einem Unix Dateisystem**





- **DNS Namen lesen und Unix Verzeichnisse und Dateien Lesen:**

`winnie.corp.hp.com` : von unten nach oben, oben ist rechts!
`/usr/local/bin/imake` von oben nach unten, oben ist links!

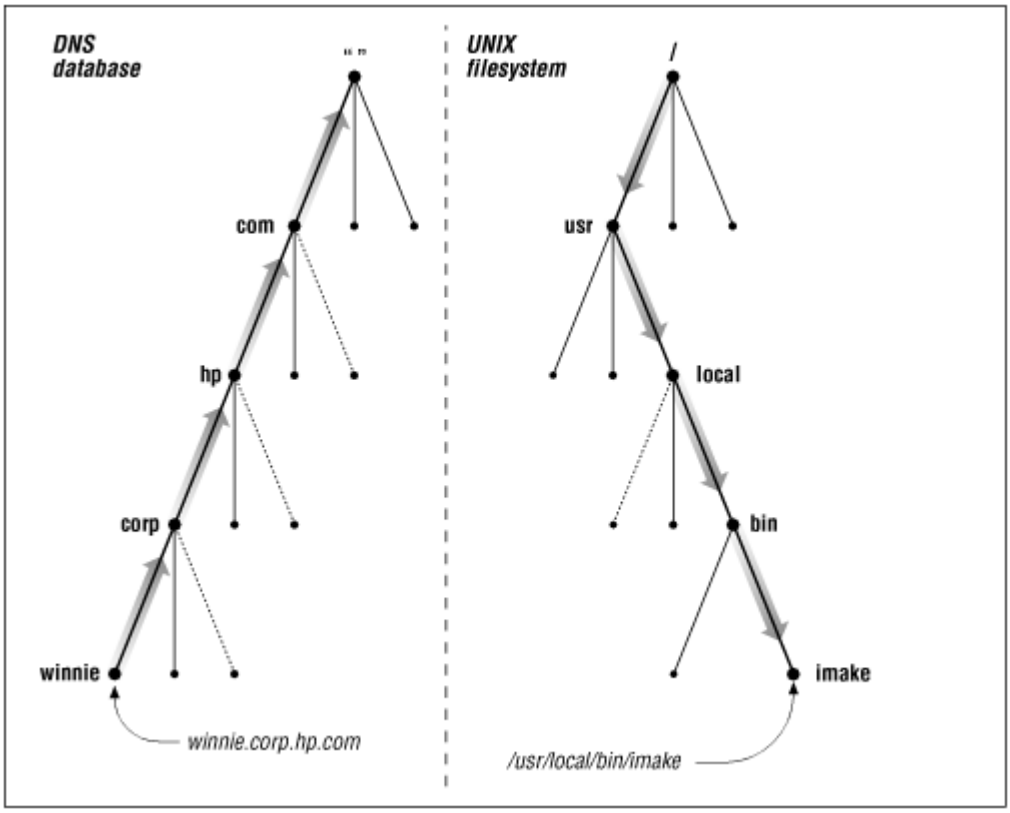


Bild 1.2 aus DNS and Bind von O'Reilly

- in DNS spricht man von Knoten (nodes) z.B. perdue.edu und Domäne (domain) z.B. perdue.edu mit allen Subdomänen (subdomains)

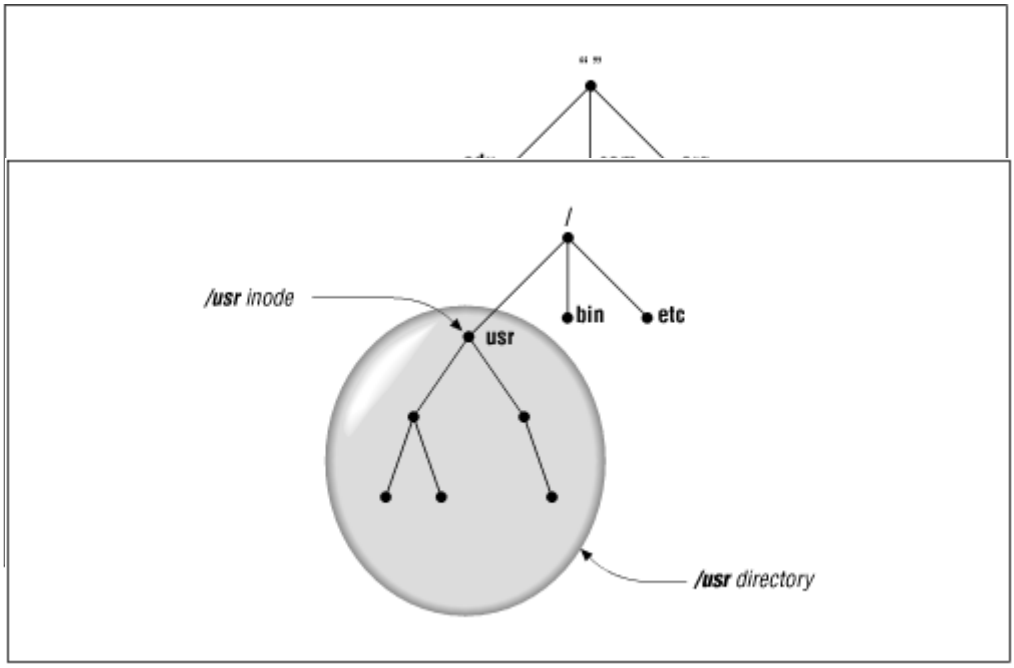


Bild 2.3 aus "DNS and Bind" von O'Reilly: Die Domäne `perdue.edu`

was unter Unix heisst inode (`/usr`) und Verzeichniss (`/usr` mit allen Unterverzeichnissen)

Bild 2.4 aus "DNS and Bind" von O'Reilly: Inodes und Verzeichnisse

- **Die Verwaltung von Domänen werden in Zonen delegiert** (wie die verschiedene Aufgaben in einem Projekt). Domäne und Zonen sind nicht das gleiche!
  z.B. Zone edu
  Zone berkeley.edu
  Zone cc.berkeley.edu
  Zone ce.berkeley.edu
  Zone cs.berkeley.edu
  Zone me.berkeley.edu

Bild 2.8 aus "DNS and Bind" von O'Reilly: Die Domäne `edu` in Zonen unterteilt.

Bild 2.9 aus "DNS and Bind" von O'Reilly: Die Domäne `berkeley.edu` in Zonen unterteilt.

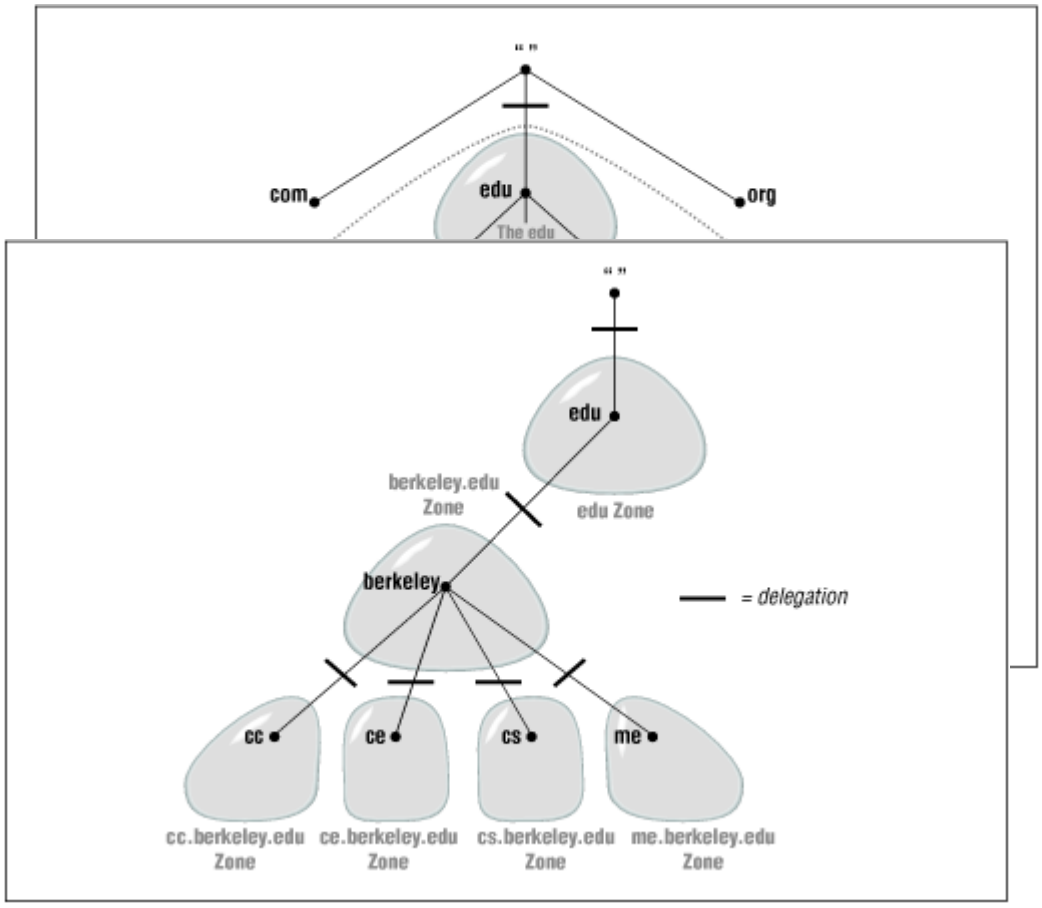
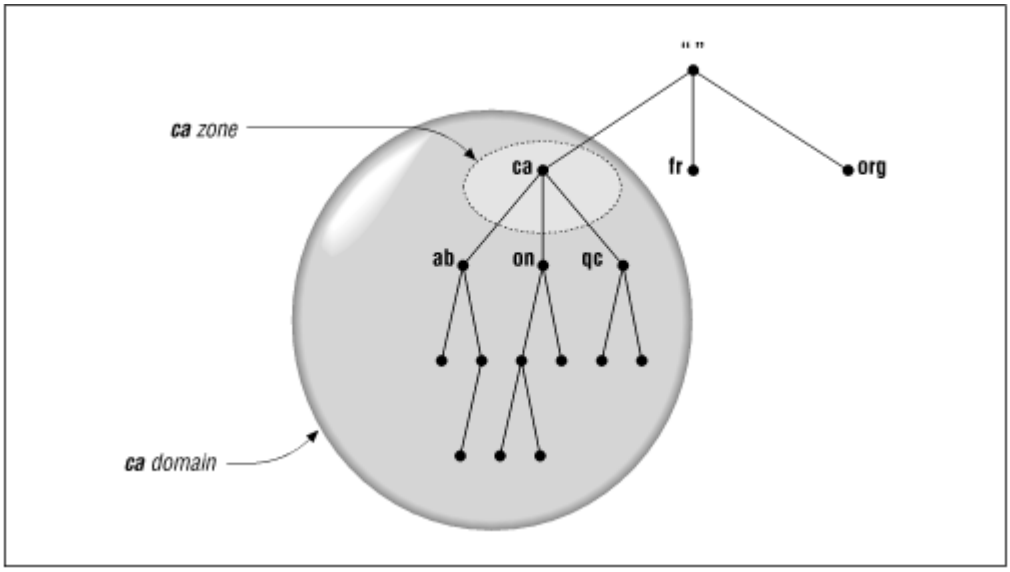


Bild 2.10 aus "DNS and Bind" von O'Reilly: Die Domäne ca...
Bild 2.11 aus "DNS and Bind" von O'Reilly: Im Gegensatz zur Zone ca

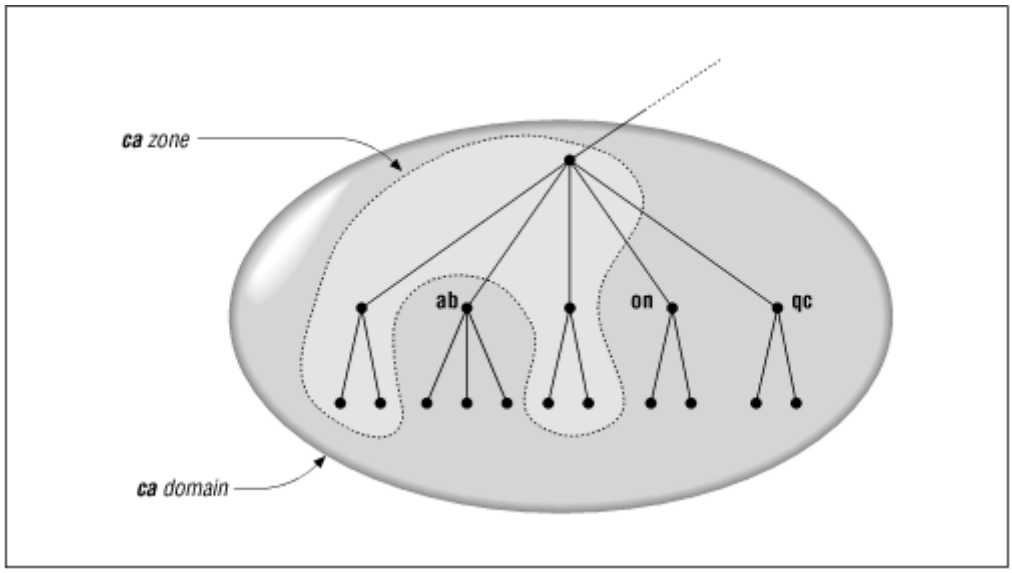
Rekursive und iterative Auflösung (resolution)

Es gibt 13 Root Name Servers, die in der Welt verteilt sind. Sie sind für die Auflösung der "top-level domains" verantwortlich

Es gibt zwei Arten eine Adresse aufzulösung: die rekursive und die iterative. Die rekursive Auflösung gibt die Hauptlast dem ersten Name-Server (NS) der die Anfrage des "resolvers" bekommen hat. Der Name-Server muss dann durch alle Domäne-Ebenen sich sozusagen durcharbeiten, bis der letzte NS ihm die



Gewünschte Antwort liefert. Die iterative Auflösung kann viel schneller gehen weil jeder NS die beste mögliche Antwort zurück gibt, bzw. zum besten bekannten NS weiterleitet. Auf dieser Weisen können mehrere Domäne-Ebenen übersrpungen



werden.

Beispiel einer rekursive Auflösung:

1. der lokale Name-Server bekommt die Anfrage vom Resolver für die Adresse: `girigiri.gbrmpa.gov.au`
2. der lokale Name-Server fragt zuerst ein Root Name-Server und bekommt die Adresse von dem Name-Server zuständig für `au`
3. der lokale NS fragt jetzt den NS au und bekommt die Adresse von dem NS zuständig für `gov.au`
4. der lokale NS fragt jetzt den NS `gov.au` und bekommt die Adresse von dem NS zuständig für `gbrmpa.gov.au`
5. der lokale NS fragt jetzt den NS `gbrmpa.gov.au` und bekommt die gewünschte Adresse von dem Rechner `girigiri.gbrmpa.gov.au`
6. der lokale NS gibt die Antwort dem Resolver zurück.

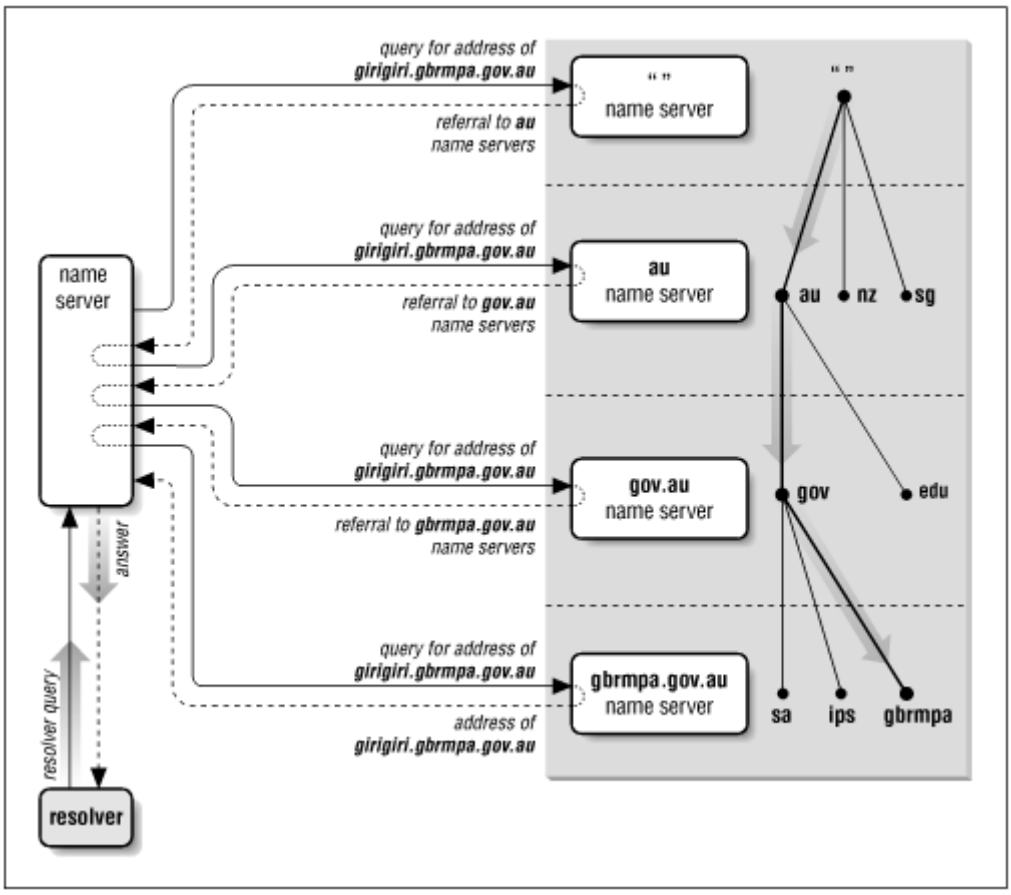


Bild 2.12 aus "DNS and Bind" von O'Reilly: Rekursive Auflösung von `girigiri.gbrmpa.gov.au` im Internet

Beispiel mit eine iterativen Auflösung:

1. der lokale Name Server bekommt die Anfrage vom Resolver für die Adresse: "labor1.forschung.uni-berlin.de"
2. der lokale Name Server fragt den nähesten Name Server (z.B. NS B), der NS B liefert die beste Antwort die er kann (z.B. NS "C")
3. der lokale NS fragt jetzt den NS "C", der NS "C" liefert die beste Antwort die er kann (z.B. NS "D")
4. der lokale NS fragt jetzt den NS "D" und bekommt die gewünschte Adresse von dem Rechner "labor1.forschung.uni-berlin.de"
5. der lokale NS gibt die Antwort dem Resolver zurück

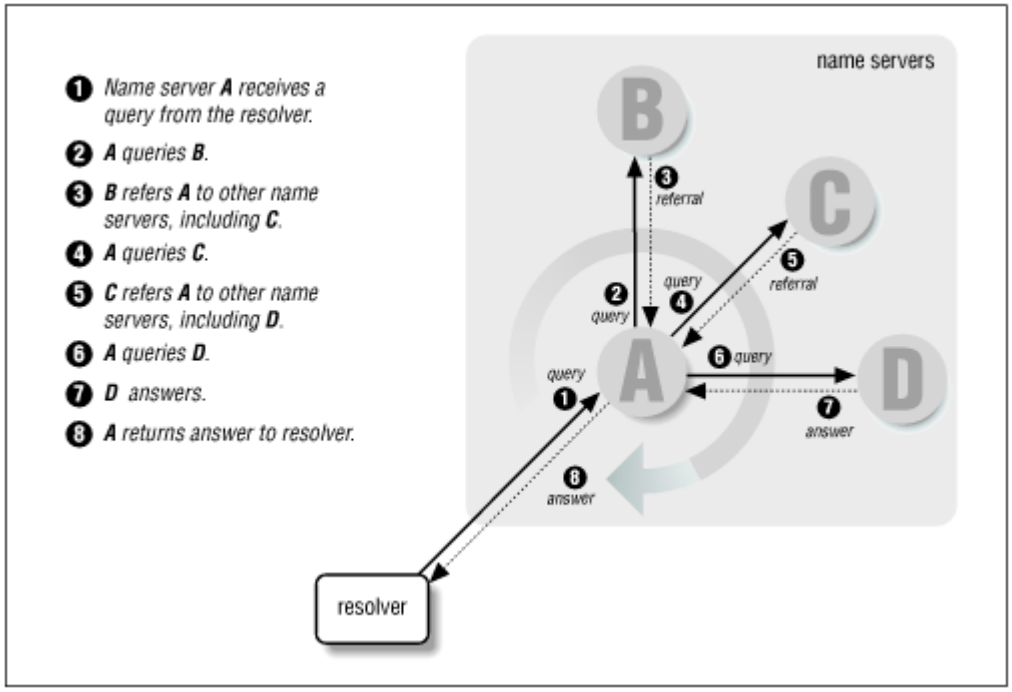


Bild 2.13 aus "DNS and Bind" von O'Reilly: Iterative Auflösung

- **Umgekehrte Auflösung (Rückwärts): IP-Adressen in Namen**
  `15.16.192.152` = von oben nach unten! oben ist links!
  `"." - arpa - in-addr - 15 - 16 - 192 - 152 = winnie.corp.hp.com`

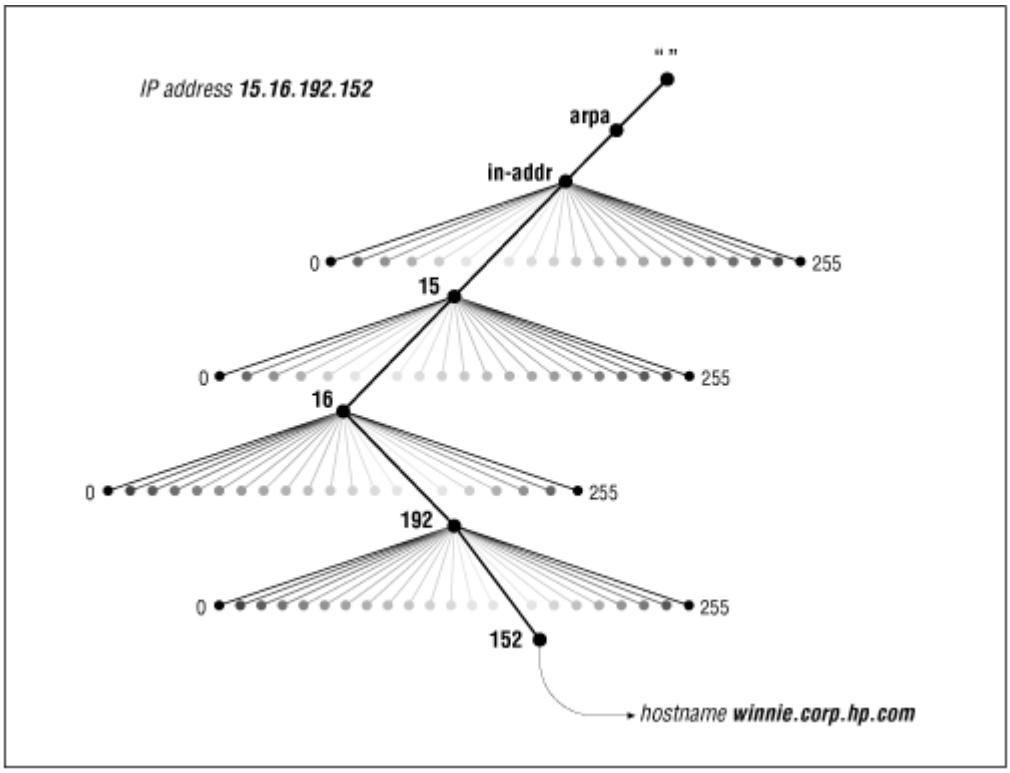


Bild 2.14 aus "DNS and Bind" von O'Reilly: Rückwärts-Auflösung.

- **Caching**
  - ist eine der wichtigsten Funktionen eines Name Servers (schnelligkeit, last)
  - positives caching (besitzt die Richtige Antwort, Adressen von verschieden NS)
  - negatives caching (eine bestimmte Adresse gibt es nicht)
  - durch caching, kürzeren/schnelleren Weg für die nächsten Anfragen (muss nicht bis zum Root Name Server gehen)
  - durch negatives caching (frühere Anfrage), kan ein NS Server die Antwort (Adresse existiert nicht) aus seinem Cache schneller geben wenn die Adresse auch nicht existiert.

- **Cache-Only Servers**
  seine einzige Aufgabe ist das caching, und kann dadurch die Requests ziemlich beschleunigen.

```
/etc/named.conf:
options {
        directory "/var/named";
};
zone    "." in {
        type hint;
        file "root.hint";
};
zone    "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};
```

Einen DNS-Server konfigurieren:

Mit YaST: Paket bind9 und bind9-utils (Serie n) installieren

Einstellungen so dass der DNS-Server nach dem Hochfahren automatisch started:
Bis SuSE 7.3: Variable `START_NAMED=yes` in `/etc/rc.config` setzten
Ab SuSE 8.0: Mit YaST - System -Runlevel-Editor - Runlevel-Eigenschaften...
auf die Zeile die mit "named" anfängt klicken - in die klienen
Vierecken mit 3 und 5 klicken - auf den Pfeil ▼ neben
Starten/Anhalten/Aktualisieren klicken - Jetzt starten... - Beenden - OK -Schließen.

Konfigurationsbeispiel:

Domäne:	stars.priv
Subnetz1:	192.168.10.0/24
Subnetz2:	192.168.2.0/24
Hostname des Master-DNS-Servers:	sirius
Hostname des Slave-DNS-Servers:	sun
Hostname eines zweiten Slave-DNS-Servers:	moon
Hostname eines Rechners im 10. Subnetz:	venus
Hostname eines Rechners im 2. Subnetz:	earth

Notwendige Konfigurationsdateien:

/etc/named.conf	Hauptkonfigurationsdatei des DNS-Servers
/var/named/localhost.zone	macht die Direkt-Auflösung über das Loopback-Device: localhost -> 127.0.0.1
/var/named/127.0.0.zone	macht die Rückwärts-Auflösung über das Loopback-Device 127.0.0.1 -> localhost
/var/named/stars.priv.zone	macht die Auflösung Rechner -> IP-Adressen
/var/named/192.168.10.zone	macht die Rückwärts-Auflösung für das 10. Netz IP-Adresse -> Rechnername
/var/named/192.168.2.zone	macht die Rückwärts-Auflösung für das 2. Netz IP-Adresse -> Rechnername
/var/named/root.hint	Liste aller Root-Name-Server (13 in der ganze Welt)

/etc/named.conf (auf dem Master, Rechner sirius)

```
options {

    directory "/var/named";
    forwarders { 217.5.100.1; 194.25.2.129; };
    #listen-on port 53 { 127.0.0.1; };
    listen-on-v6 { any; };
    allow-query { 127.0.0.1; 192.168.10.0/24; 192.168.2.0/24; };
    notify yes;
};

zone "." in {
        type hint;
        file "root.hint"; };
zone "localhost" in {
          type master;
          file "localhost.zone";
```

```
          check-names fail;
          allow-update { none; };
};

zone "0.0.127.in-addr.arpa" in {
          type master;
          file "127.0.0.zone";
          check-names fail;
          allow-update { none; };
};

# a master  zone (das ist nur ein Beispiel!)
#
zone "stars.priv" in {
          type master;
          file "stars.priv.zone";
          notify yes;
};

zone "10.168.192.in-addr.arpa" in {
          type master;
          file "192.168.10.zone";
          notify yes;
};

zone "2.168.192.in-addr.arpa" in {
          type master;
          file "192.168.2.zone";
          notify yes;
};

#a slave zone
#
zone "elop.temp.berlin" in {
          type slave;
          file "slave/elop.temp.berlin.zone";
          masters { 192.168.71.37; };
};

zone "70.168.192.in-addr.arpa" in {
          type slave;
          file "slave/192.168.70.zone";
          masters { 192.168.71.37; };
};

zone "71.168.192.in-addr.arpa" in {
          type slave;
          file "slave/192.168.71.zone";
          masters { 192.168.71.37; };
}
```

/etc/named.conf (auf dem Slave, Rechner sun oder moon)

```
#
# a slave zone (das ist nur ein Beispiel!)
#
zone "stars.priv" in {
        type slave;
        file "slave/stars.priv.zone";
```

```
        masters { 192.168.10.10; };
};

zone "10.168.192.in-addr.arpa" in {
        type slave;
file "slave/192.168.10.zone";
        masters { 192.168.10.10; };
};

zone "2.168.192.in-addr.arpa" in {
        type slave;
        file "slave/192.168.2.zone";
        masters { 192.168.10.10; };
};
```

/var/named/stars.priv.zone (auf dem Master, Rechner sirius):

```
;  file /var/named/stars.priv.zone
; SOA Record                    SOA=Start of Authority
$TTL 2D
@         IN SOA            sirius root.localhost  (
          2000121501        ; serial , todays date YYYY MM DD + todays serial  XX
          24H               ; refresh= how often the slave has to check that its data are up
                            ; to date
          2H                ; retry=if the slave fails to reach his master, tries to reconnect
          ; every retry seconds
          30D               ; expire=if the slave fails to contact his master for expire sec.
                            ; it stop to give out his data.
          4D  )             ; minimum TTL (time to live)=the maximum time other servers
          ; are allowed to cache these data
;
; NS Record
;
          IN NS             sirius        ; Inet Address of name server
          IN NS             moon    ; Inet Address of slave name server
          IN NS             sun     ; Inet Address of slave name server
;
; Address and Alias Records
;
moon      IN A              192.168.2.1
moon      IN A              192.168.10.1
sirius    IN A              192.168.10.10
ssv046    IN A              192.168.10.11
sun       IN A              192.168.10.14
venus     IN A              192.168.10.15
;
globeall  IN CNAME SIRIUS
vivation  IN CNAME SIRIUS
;
earth     IN A              192.168.2.12
```

/var/named/192.168.10.zone (auf dem Master, Rechner sirius)

```
;  file /var/named/192.168.10.zone
$TTL 2D
@         IN SOA            sirius.stars.priv.   root.localhost.  (
                            2000071501    ; serial
                            24H           ; refresh = 86400 sec
```

```
                                2H            ; retry     = 7200 sec
                                30D           ; expiry      = 2592999 sec
                                4D  )         ; minimum TTL = 345600 sec

          IN NS                 sirius.stars.priv.
          IN NS                 moon.stars.priv.
          IN NS                 sun.stars.priv.
;
; PTR Record
;
1         IN PTR                moon.stars.priv.
10        IN PTR                sirius.stars.priv.
11        IN PTR                ssv046.stars.priv.
14        IN PTR                sun.stars.priv.
15        IN PTR                venus.stars.priv.
```

/var/named/192.168.2.zone (auf dem Master, Rechner sirius):

```
;  file /var/named/192.168.2.zone
$TTL 2D
@         IN SOA                sirius.stars.priv.   root.localhost.  (
                                2000071501    ; serial
                                24H           ; refresh = 86400 sec
                                2H            ; retry     = 7200 sec
                                30D           ; expiry      = 2592999 sec
                                4D  )         ; minimum TTL = 345600 sec

          IN NS                 sirius.stars.priv.
          IN NS                 moon.stars.priv.
          IN NS                 sun.stars.priv.
;
; PTR Record
;
1         IN PTR                moon.stars.priv.
12        IN PTR                earth.stars.priv.
```

Mail Exchanger Record (für einen Mail-Server wie Sendmail oder Postfix):

Domainname z.B.	IN	MX	Priorität-Nummer	Rechnername
stars.priv.	IN	MX	10	sirius.stars.priv.
stars.priv.	IN	MX	20	moon.stars.priv.
stars.priv.	IN	MX	30	sun.stars.priv.

Die kleinste Nummer hat die höchste Priorität !!!

letzter Schritt auf dem Server:
eventuell die IP-Adressen & Rechner aus `/etc/hosts` auskommentieren

`rcnamed start` oder `restart`

Auf dem Klienten:

- die IP-Adressen und Rechnernamen in /etc/hosts auskommentieren
- Konfiguration des Nameserver mit YaST (bis SuSE 7.3):
  Administration des Systems -> Netzwerk konfigurieren ->
  Konfiguration Nameserver (erstellt die Datei `/etc/resolv.conf`)
- Mit YaST2: Netzwerk/Erweitert - Hostname und DNS - Liste der Nameserver und Domain-Suchliste - Beenden.

- **DNS-Testprogramme**

  - **nslookup**

    `nslookup <return>` (nslookup wird irgendwann verschwinden)
    `>`*`Hostname oder IP-Adresse`*
    `>ls -d` *`Domainname`*
    `>server` *`ServerName`*
    `>help`
    `>exit oder <Str>d`

  - **host**

`host [-v] Rechnername`	versucht der Rechnernamen aufzulösen. -v = verbose, die Ausgabe ist dann ähnlich wie mit `dig`.
`host Rechnername DNS-Server`	benutzt den angegeben DNS-Server für die Auflösung
`host IP-Adresse`	versucht die IP-Adresse aufzulösen.
`host -l Domäne`	zeigt alle Rechner einer DNS-Domäne.
`host -t mx Domäne`	zeigt der Mail-Exchange-Server einer Domäne.

  - **dig**

`dig [@server] name [type]` (type = any, a, mx, ns usw.)	`dig` wie `host` erlaubt einen Rechnernamen aufzulösen, aber gibt mehr Informationen.
z.B. `dig sun.linux.local`	versucht `sun.linux.local` aufzulösen
`dig @dozlinux sun`	versucht `sun` vom DNS-Server `dozlinux` aufzulösen.
`dig linux.local any`	zeigt die ganze Domäne `linux.local` an.
`dig -x IP-Adresse`	versucht eine IP-Adresse aufzulösen.

  - **Die Datei root.hint aktualisieren**

    ```
    cd /var/named
    dig @e.root-servers.net . ns > root.hint.new
    ```
    oder `dig @a.root-servers.net . ns > root.hint.new`
    ```
    mv root.hint root.hint.old
    mv root.hint.new root.hint
    rcnamed restart
    ```

- ## Drucken unter Linux mit CUPS

  CUPS = Common Unix Printing System, ist (ganz unparteiisch :-) das beste Drucksystem unter Linux und kann komplett das ursprüngliche Drucksystem `LPD` (`BSD printing spool system`) oder das neuere Paket `LPRng` ersetzen.

  Die Firma Easy Software Products `http://www.easysw.com` hat ursprünglich das kommerzielle Produkt `ESP Print Pro` für Unix-Kunden entwickelt. Später wurde der Kern von `ESP Print Pro` (ohne GUI-Frontend und nur mit wenige Treiber) als GPL-Software frei gegeben.
  CUPS ist ab SuSE 7.1 enthalten, Mandrake hat es schon länger als ihr Standard-Drucksystem, RedHat 7.3 hat es neben `LPRng` endlich auch zur Auswahl und Debian hat es auch schon länger als mögliches Drucksystem.
  Die letzte Version von CUPS kann als Tarball von `http://www.cups.org` bezogen werden. Das Paket muss allerdings noch kompiliert werden und es werden nur wenige generische Treiber mitgeliefert.

  Zwei wichtige Merkmale von CUPS:

  - CUPS funktioniert mit dem Standard Protokoll IPP = Internet Printing Protocol (Port 631) an dem sich die meisten Drucker-und Software-Hersteller (sogar Microsoft) angeschlossen haben. Das Verwalten von Lokale- und Netzwerk-Drucker ist dadurch sehr leicht.
  - CUPS verwendetet für jeden Drucker eine PPD-Datei (Postscript Printer Definition). Der PPD-Format wurde von Adobe® kreiert und erlaubt die Beschreibung eines Postscript-Druckers (wieviel DPI, wieviel Papierfächer, Farben, Duplex-Einheit usw.). Das revolutionäre ist, dass CUPS sogar eine PPD-Datei für einen "nicht-Postscript-Drucker" verwendet.

  - **CUPS-Verwaltung über den Browser**

    Weil CUPS einen kleinen Web-Server mitliefert, kann die ganze Warteschlage-Verwaltung, wie z.B. eine neue Drucker-Warteschlange zufügen oder entfernen, Jobs löschen, Dokumentation lesen, über einen Browser gemacht werden.

    Die Administrations-Seite von CUPS ist mit der URL `http://localhost:631` zu erreichen.

  - **CUPS-Verwaltung über Kommandos**

    Es gibt auch die Möglichkeit, CUPS über Befehle zu administrieren, hauptsächlich über den Befehl `lpadmin`. Andere Befehle: `lp, lpstat, lpinfo, cancel, enable, disable, accept, reject` und `cupsaddsmb.`

  - **Neue PPD für einen Drucker erstellen**

    Über die Seite `http://www.linuxprinting.org` und das Skript `PPD-O-Matic`, kann für fast alle bekannten Drucker, eine PPD für CUPS erstellt werden. Bevor eine PPD benutzt werden kann, muss zuerst das Skript `cupsomatic` in `/usr/lib/cups/filter/` kopiert werden und mit `chmod 755` ausführbar gemacht werden (bei SuSE und Mandrake schon enthalten). Dann kann mit `PPD-O-Matic` eine PPD erstellt werden: "Printer Listing" oder Driver Listings" dann einen Treiber auswählen PPD-O-Matic ausführen. Das Resultat speichern und unter `/usr/share/cups/model/` kopieren. Den CUPS-Dämon neu starten.

- **Andere CUPS-Treiber-Quellen**
  - `http://gimp-print.sourceforge.net`
    Sehr gute Treiber für Gimp und Titenstrahler von HP, Epson, Lexmark und Canon (bei SuSE schon enthalten).
  - `http://www.easysw.com/printpro/`
    Die kommerzielle Variante von CUPS mit sehr viele Treiber, Support usw.

- **Einige CUPS-Befehle**

  - **Druckbefehle**
    - `lp` *`Datei`* oder `lp -d` *`Drucker Datei`* oder `lpr -p` *`Drucker Datei`*
    - `lp -o prettyprint` *`Datei`* druckt die Seite mit einem Header
    - `lp -o media=Upper` (Fach 1) oder `lp -o media=A4,upper`
    - `lp -o media=Lower` (Fach 2) oder `lp -o media=A4,lower` *`Datei`*
    - `lp -o page-ranges=1` (oder `1-4` oder `1,3,7`) *`Datei`*
    - `lp -o page-set=odd` (oder `even`) *`Datei`* druckt nur die Ungeraden/ Geraden Seiten
    - `lp -n` *`AnzahlKopien Datei`*
    - `lp -o landscape` *`Datei`*

  - **Drucker und Jobs anschauen**
    - `lpstat -t` zeigt alle möglichen Angaben: Drucker-Status, Jobs
    - `lpstat -o` zeigt alle Jobs in den Warteschlagen
    - `lpstat -p` zeigt alle Drucker-Warteschlangen
    - `lpstat -o -p` Drucker und Jobs anschauen

  - **Job löschen**
    - `cancel` *`Jobnummer`*
    - `cancel -a` *`Warteschlange`* (löscht alle Jobs in dieser Warteschlange)

  - **Druckeroptionen anzeigen**
    - `lpoptions -l Warteschlange` (zeigt Seitengrösse, dpi usw.)

- **Administration von CUPS als Server**
  - Hauptkonfigurationsdatei für den CUPS-Server: `/etc/cups/cupsd.conf`

    Nur 2 Parameter müssen für einen CUPS-Server gesetzt werden:

    `BrowseAddress 192.168.x.255` so dass die Klienten die Drucker sehen.

    ```
    <Location />
      Allow From 192.168.x.0/24 so dass Klienten auf dem Server zugreifen
                                können / drucken
    </Location>
    ```

  - eine Druckerwarteschlange mit dem Befehl `lpadmin` installieren:

    ```
    [/usr/lib/]lpadmin -p Drucker \
    -m de/HP/LaserJet_5L-ljet4.ppd.gz -D "Drucker Info-Text" \
    ```

`-L "Ort vom Drucker" -v parallel:/dev/lp0 -E` (-E für Enable)

`-m` steht für `model` (meistens das Verzeichnis /`usr/share/cups/model/` und enthält alle verfügbaren Treiber. Die Unterverzeichnisse von `model` müssen mit dem Namen des Treibers angegeben werden.

- `lpinfo -m` zeigt all verfügbaren Treiber.
- eine Druckerwarteschlange löschen: `lpadmin -x Drucker`
- eine Druckerwarteschlange als Standard setzen: `lpadmin -d Drucker`
- Start u. Stop einer Queue: `/usr/bin/enable Drucker`
  `/usr/bin/disable Drucker`
- Jobs starten & stoppen: `/usr/sbin/accept Drucker`
  `/usr/sbin/reject Drucker`
- Die Dokumentation in PDF-und HTML-Format befindet sich in:
  `/usr/share/cups/doc`
  z.B. `sum.pdf` (software user manual) & `sam.pdf` (software administration manual)
- Die Logdateien befinden sich in: `/var/log/cups/` und heißen `access_log` und `error_log` (wie bei apache)
- **Klassen**
  Es ist möglich mehrere gleichartige Drucker in einer Klasse zu gruppieren. Wenn beim Drucken die Klasse als Warteschlange genommen wird, dann wird automatisch der erste freie Drucker dieser Klasse genommen.
  `lpadmin -p Drucker -c Klasse` fügt einen Drucker zu einer Klasse. Wenn die Klasse noch nicht existiert, wird sie erstellt.
  Die Klasse muss noch mit `/usr/bin/enable Klasse` gestartet werden.
  `lpadmin -p Drucker -r Klasse` entfernt einen Drucker von einer Klasse.

- **CUPS mit Samba**
  - Samba-Konfigurationsdatei `/etc/smb.conf`
    (ab SuSE 7.3 `/etc/samba/smb.conf`):
    ```
    load printers = yes
    printing = cups
    printcap name = /etc/printcap
    ```
    oder ab Samba 2.2 `printcap name = cups`

    mit printcap name = cups muss noch `cupsaddsmb [-a]` ausgeführt werden.

- **CUPS-GUI-Werkzeuge**
  - **kprinter, qtcups & xpp**

    Seit KDE 2.2 ist CUPS sehr gut integriert. CUPS kann direkt aus dem KDE-Kontrollzentrum konfiguriert werden (mindestens mit KDE 3, SuSE 8.0). `kprinter` ist eine sehr schöne Schnittstelle zum Befehl `lp/lpr` und erlaubt ein Dokument mit vielen verschiedenen Optionen zu drucken. `qtcups` gibt es schon länger und hat grundsätzlich das gleiche Ziel wie `kprinter`, hat aber weniger Optionen. `xpp` hat auch das gleiche Ziel wie die zwei anderen Programme, ist aber von KDE

unabhängig, und hat einen typischen X-Programm-Look.

- **kups**

  `kups` ist ein Administrationsprogramm, das auch schon länger existiert. Mit der Integration von CUPS in KDE, ist eigentlich `kups` überflüssig geworden.

• Drucken mit LPD - das BSD Printing Spool System

Paket `lprold` oder `LPRng` Serie n

- lpr [-P*Druckerschlange*] *Datei*
- lpq -l [-P*Druckerschlange*]
- lprm [-P*Druckerschlange*] [*JobNummer*] [*Benutzer*]
- lpc status

mit YaST eine LPD-Drucker konfigurieren:
Administration des Systems - Hardware in System integrieren - Drucker konfigurieren

Beschreibung der Druckerwarteschlangen: Datei `/etc/printcap`

Übersetzungsfilter von Postscript in PCL, ESP: `Ghostscript, apsfilter`

- **LPD-Drucker mit lprsetup konfigurieren**

```
lprsetup
ENTRY       Add/Overwrite/Delete an apsfilter entry
DEVICE      Which printer interface
PARALLEL    Parallel printer interface     z.B. -> /dev/lp0  (Port 1)
PRINTER     Which printer driver  z.B. -> POSTCRIPT oder OTHER
PAPER       Which paper type               z.B. -> a4
COLOR       Monochrome/colorfull           z.B. -> MONO  Mono  printer
ADD         add the printer definition
```

- **LPD-Netzwerk-Drucker installieren**
  - **Auf dem Server:** Rechnernamen aller Klienten in der Datei `/etc/hosts.lpd` einfügen
  - **Auf jedem Klient-Rechner:** `lrpsetup` aufrufen:

```
ENTRY           Add/Overwrite/Delete an apsfilter entry
DEVICE          Which printer interface
REMOTE          Printer forwarding queue
    What's the host name of your remote printer?     Rechnername
    What's the name of your remote printer?          Druckername (lp)
ADD             add the printer definition
DEVICE          Which printer interface
PREFILTER To another queue (bypass)
    remote      remote=Rechnername queue=Druckername -> OK
PRINTER         Which printer driver  -> POSTCRIPT oder OTHER
PAPER           Which paper type           -> a4
COLOR           Monochrome/colorfull       -> MONO  Mono  printer
ADD             add the printer definition
```

  - **Auf dem Klienten drucken:**
    lpr -Pascii (oder wahrscheinlich lp2) *Dateiname*

TCP/IP Services

Der Internet Server oder Super Server - inetd

(kommt aus dem Paket nkitb, Serie a)
die Konfigurationsdatei für den inetd: **/etc/inetd.conf**

Zeilenaufbau : **service type protocol wait user server cmdline**

`service` Name des Dientes wie in /etc/services, z.B. ftp

`type` Verbindungsart: *stream* für Verbindungsorientierte (TCP), *dgram* für verbindungslose (UDP) Protokolle.

`protocol` das verwendete Protokoll, meistens TCP oder UDP

`wait` immer *`nowait`* für stream, inetd hört direkt weiter auf dem Port. *`wait`* für dgram, inetd wartet bis der Server fertig ist.

`user` Eigentümmer (UID) des Serverprozesses

`server` Pfad zur ausführbaren Programmdatei des Servers oder das Schlüsselwort *`internal`* für interne Funktionen von inetd (time,echo)

`cmdline` Parameter für den Server, Servername + Optionen

Beispiele: (mit TCP-Wrapper)

```
ftp    stream tcp nowait     root usr/sbin/tcpd  wu.ftpd -a
telnet stream tcp nowait     root /usr/sbin/tcpd in.telnetd
```

(400=max Server pro Min)

```
swat   stream tcp nowait.400 root /usr/sbin/swat swat
```

Nach jeder Änderung muss inetd neu gestartet werden: rcinetd restart

Der TCP Wrapper - tcpd

tcpd tested die Dateien `/etc/hosts.allow` & `/etc/hosts.deny` (in dieser Reihenfolge). Es gibt drei Möglichkeiten:

1. Ein Eintrag wird in hosts.allow gefunden: tcpd sucht nicht weiter in hosts.deny
2. Ein Eintrag wird in hosts.deny gefunden: der Dienst wird abgelehnt
3. Keine Einträge werden gefunden: der Dienst wird akzeptiert

Syntax in den Dateien `hosts.allow` und `hosts.deny`:

```
Dienstliste:  Hostliste  [ :Shellbefehl ]
```

Dienstliste einen Dienst aus /etc/inetd.conf
Hostliste Hostnamen, IP-Adressen, ALL, LOCAL (ohne Punkte), UNKNOWN (lookup kann es nicht finden), PARANOID (die IP kann aus dem Namen nicht zurück gefunden werden)
Shellbefehl Befehl auszuführen wenn der Eintrag übereinstimmt

Beispiele für `/etc/hosts.deny`:

```
in.telnetd: bts0412.linux.local, bts0413.linux.local
in.telnetd, wu.ftpd: ALL EXCEPT .linux.local LOCAL
in.talkd, in.fingerd : ALL EXCEPT LOCAL
ALL: .stars.priv
ALL EXCEPT finger, tftp: LOCAL
ALL: 192.168.10.  oder ALL: 192.168.10.0/255.255.255.0
```

- **FTP**

  FTP-Server-Port = 21

  FTP-Befehle:

  Syntax: ftp Rechnername oder IP-Adresse oder nur ftp

  Die wichtigsten FTP Befehle:

`open`	eine Verbindung zu einem entfernten Rechner erstellen
`close`	eine Verbindung wieder schließen, ohne FTP zu beenden
`user`	sich neu einloggen, nach einen loggin Fehler
`bin`	binary oder bitweise, der Verbindungsmodus ist transparent
`ascii`	Textmodus, praktisch wenn die Betriebssysteme nicht gleich sind
`put`	eine Datei zu einem entfernten Rechner senden
`mput`	mehrere Dateien zu einem entfernten Rechner senden
`get`	eine Datei vom einem entfernten Rechner holen
`mget`	mehrere Dateien vom einem entfernten Rechner holen
`ls/dir`	den Inhalt des Verzeichnisses auf dem entfernten Rechner anschauen
`pwd`	zeigt das aktuelle Verzeichnis auf dem entfernten Rechner an
`lpwd`	zeigt das aktuelle Verzeichnis auf dem lokalen Rechner an
`cd`	das Verzeichnis auf dem entfernten Rechner wechseln
`lcd`	das Verzeichnis auf dem lokalen Rechner wechseln
`!`	erlaubt Befehle auf dem lokalen Rechner auszuführen, ohne FTP zu beenden, *exit* um wieder zu FTP zurückzukommen
`more/less`	zeigt den Inhalt einer Datei auf dem entfernten Rechner an
`delete`	löscht eine Datei auf dem entfernten Rechner
`rename`	umbenennen einer Datei auf dem entfernten Rechner
`mkdir`	erstellt ein Verzeichnis auf dem entfernten Rechner
`rmdir`	löscht ein Verzeichnis auf dem entfernten Rechner
`help/?`	zeigt alle möglichen Befehle an
`help xyz`	kleine Beschreibung des Befehls "`xyz`"
`quit/exit`	beendet FTP

  Die Datei `/etc/ftpusers` bestimmt auf der Server-Seite welche Benutzer ftp nicht benutzen dürfen, wie normalerweise der Benutzer root!

  - **Textorientierte FTP-Klienten**
    - **ftp**

      Der klassische Unix/BSD FTP-Klient
    - **lukemftp**

      Ein Ersatz für den Standart BSD-ftp, mit viel mehr Möglichkeiten / Befehlen. Siehe `man ftp` unter `History`.
    - **ncftp**

      Eine konfortable Alternative zum klassischen BSD-ftp, mit der Möglichkeit Lesezeichen zu setzten und Konfigurationen zu speichern.

- **Graphische FTP-Klienten**
  - **gftp**

    Effizientes und einfaches Programm. Das Aussehen ist sehr ähnlich wie ws_ftp pro unter Windows. `gftp` ist ein Programm des Gnome-Projektes.
  - **kbear**

    `Kbear` ist ein sehr elegantes und leistungsfähiges Programm. Es kann mehrere Verbindungen gleichzeitig öffnen. Es kommt mit dem KDE(2)-Projekt.
  - **iglooftp**

    Ein anderes einfaches und leistungsfähiges (aber kommerzielles) FTP-Programm. Es kann unter anderem Verbindungen über SSL, TSL und SRP aufbauen. Home-Page: `http://www.iglooftp.com/linux/`
  - **xftp**

    Älteres Programm mit dem Typischen (grauen...) X-Look.

- **Einige FTP-Server**
  - **vsftpd - Very Secure FTP Server**

    `vsftpd` ist ein sicherer (insofern FTP überhaupt sicher sein kann) kleiner und schneller FTP-Server. Die Konfiguration ist ziemlich einfach und wird über die Datei `/etc/vsftpd.conf` konfiguriert.

    Einige Parameter, die konfiguriert werden können sind:

    # erlaubt / verbietet einen "anonymous login", Benutzername: anonymous oder
    # ftp, Passwort: eine E-Mail-Adresse, leider auch eine fiktive)
    `anonymous_enable=NO`

    # erlaubt / verbietet den Zugang für lokale Benutzer (über localhost)
    `local_enable=YES`

    # erlaubt / verbietet jegliche Schreibe-Operationen.
    `write_enable=YES`

    # ändert die Willkommens-Meldung, und verbirgt auf dieser Weise welcher FTP-Server gerade seine Dienste anbietet (Software und Version)
    `ftpd_banner=Welcome to FTP service.`

    # die drei folgenden Parameter funktionieren zusammen. `chroot` (change root) ist
    # eine Art Gefängnis (jail), das dem Benutzer verbietet in das übergeordnete
    # Verzeichnis zu wechseln (cd .. funktioniert von dem Heimatverzeichnis nicht).
    # alle lokale Benutzer werden in einem `chroot` begrenzt.
    `chroot_local_user=YES`

    # wenn der vorherigen Parameter auf `YES` ist, ist die Chroot-Liste, die Liste der Benutzer, die nicht mit chroot begrenzt werden. Wenn der vorherigen Parameter auf `NO` ist, dann ist die Chroot-Liste die Liste der Benutzern die mit chroot begrenzt werden.
    `chroot_list_enable=YES`

    # Dateiname der Chroot-Liste.
    `chroot_list_file=/etc/vsftpd.chroot_list`

Es sit möglich eine Konfigurationsdatei pro Benutzer anzulegen. Der Parameter `user_config_dir` definiert das Verzeichnis wo sich die Konfigurationsdateien der Benutzer befinden. Die Benutzer-Konfigurationsdateien heissen gleich wie die Benutzernamen. Der Inhalt ist gleich wie die Datei `/etc/vsftpd.conf` aufgebaut.
`user_config_dir=/etc/vsftpd_user_conf`

z.B für den Benutzer `ftpwebuser` würde die
Datei `/etc/vsftpd_user_config/ftpwebuser` heissen.

`local_root` definiert in welchem Verzeichnis der FTP-Benutzer nach dem anmelden sich befinden soll. Dieser Parameter ist nur sinvoll wenn dieses Verzeichnis anders als das Heimatsverzeichnis dass in `/etc/passwd` definiert ist sein soll.
`local_root=/srv/www/meine_domaene`

- `vsftpd` wird über den `inetd` gestartet:

  `ftp stream tcp nowait root /usr/sbin/tcpd vsftpd`
- Dokumentation: `man vsftpd` und `man 5 vsftpd.conf` gefunden werden
- Home-Page: `http://vsftpd.beasts.org`
- Bei SuSE ab Version 8.0 enthalten

- **Pure-FTPd**

  `pure-ftpd` ist ein anderer kleiner und schneller FTP-Server.
  Er wird direkt mit Optionen statt mit einer Konfigurationsdatei gestartet:

  z.B. `pure-ftpd -A` (chroot) `-I` *`Timeout`* (in Min) `-s` (antiwarez) usw.

  Bei SuSE 8.0 wird `pure-ftpd` über das Skript `/etc/init.d/pure-ftpd` gestartet (und eben nicht über inetd). Die Optionen werden über das Perl-Skript `/usr/sbin/pure-config-args` aus der Datei `/etc/pure-ftpd.conf` gelesen, und an `pure-ftpd` als Variable übergeben.

  - Dokumentation: `man pure-ftpd`
  - Home-Page: `http://www.pureftpd.org`
  - Bei SuSE ab Version 8.0 enthalten.

- **proFTPd**

  proFTPd hat sehr viel Konfigurationsmöglichkeiten. Die Konfigurationsdatei ist ähnlich wie die von Apache aufgebaut.

  - Home-Page: `http://www.proftpd.org`
  - Bei SuSE bis zur Version 7.3 enthalten

- **Der Shell-Zugang (ssh, telnet) für einen FTP-Benutzer blockieren:**

  Es gibt mindestens zwei Möglichkeiten, um FTP-Benutzerkonten für den Zugang über ssh oder telnet (rlogin) zu blockieren:

  - Für jeden FTP-Benutzerkonto, die Shell-Spalte (7. Spalte) in der Datei `/etc/passwd` auf `/bin/false` setzten. Die Zeile `/bin/false` muss in der Datei `/etc/shells` vorhanden sein oder zugefügt werden.
  - Eine sogenannte "dummy shell" erstellen:

    1. Mit einen Texteditor die Datei `/etc/dummy.c` erstellen:

       ```
       /* dummy shell */
       #include<stdio.h>
       void main(void)
       {
        fprintf(stderr,"\n Sie haben keine Berechtigung sich
       einzuloggen!\n\n");
       }
       ```

    2. Die Datei `dummy.c` kompilieren:
       `gcc -s -static dummy.c -o /bin/dummy`
    3. Die Zeile `/bin/dummy` in der Datei `/etc/shells` zufügen
    4. In der Datei `/etc/passwd`, für jedes FTP-Konto, die Shell-Spalte `/bin/bash` in `/bin/dummy` ändern, z.B.: `public:x:518:32::/public:/bin/dummy`

Telnet

Telnet-Server-Port = 23

Telnet-Klient
Syntax:
`telnet [-l` *`Benutzer`*`] `*`Rechnername`* oder *`IP-Adresse`* `[`*`Port`*`]`

`exit` beendet die Telnet-Verbindung auf dem entfernten Rechner.
`<Str>d` beendet, wenn noch kein Benutzernamen eingegeben worden ist.
Telnet kann in zwei Modi arbeiten, der Eingabe-Modus (Standart) und der Komando-Modus. Der Komando-Modus wir nur in seltenen fällen benutzt.
`<Str>]` erlaubt den Wechsel von dem Eingabe-Modus zum Komando-Modus, dann `2x <Return>` kehrt zum Eingabe-Modus zurück
Komando-Modus:
telnet

telnet > **?**	zeigt alle möglichen Befehle an
telnet > quit	beendet die telnet-Verbindung

Es ist normalerweise nicht möglich sich als `root` auf einem entfernten Recher einzuloggen. Es ist mit YaST möglich, aber nicht ratsam, dies einzuschalten:
`root_login_remote=yes`

Telnet ist einfach zu bedienen. Es ist meistens oder sogar immer auf jedem Betriebssystem vorhanden. Der große Nachteil ist, daß der Benutzername und das Passwort in Klartext übertragen werden!

Mit telnet kann man auch Dämone wie Apache oder Sendmail testen.
Für den Web-Server Apache:

```
telnet localhost 80
GET /
```

Für den Mail-Server Sendmail oder Postfix:

```
telnet localhost 25
helo localhost
help
mail from: pierre@localhost
rcpt to: root@localhost
data
to:root
subject: sendmail testen
Dies ist nur einen Test mit telnet und sendmail
.
quit
```

Für ein E-Mail mit SuSE 8.0 von andere Rechner empfangen zu können, muss zuerst die Variable `SMPD_LISTEN_REMOTE="yes"` gesetzt werden.
Das kann mit `YaST - System - Sysconfig-Editor -+--etc` gesetzt werden, oder direct über die Datei `/etc/sysconfig/mail`. Nach der Veränderung muss der Mail-Server `Sendmail` mit `rcsendmail restart` neu gestartet werden.

• BSD-Remote Befehle

Die BSD-Remote-Befehle oder r*-Befehle sind in einer Zeit entstanden, wo es Netzwerkmissbräuche noch kaum gab. Das Hauptziel der r*-Befehle war Komfort im Netzwerk, die Möglichkeit sich leicht auf einem entfernten System einzuloggen, Dateien von einem System zum anderen leicht zu kopieren usw. Diese Befehle sind nicht mehr zeitgemäß, weil die Benutzernamen und Passwörter leider in Klartext übermittelt werden.
Alle Funktionen der r*-Befehle können durch das Paket ssh (secure shell) ersetzt werden.
Die folgenden Informationen sind hier nur für den Fall, dass Sie auf einem Rechner arbeiten müssen, auf dem diese Werkzeuge noch in Funktion sind, und eventuell so, dass Sie sie ausschalten oder sogar entfernen können.

rlogin remote login - auf einem entfernten Rechner sich einloggen
rsh ein Shell-Befehl auf einem entfernten Rechner ausführen
rcp eine Datei auf oder von einem entfernten Rechner kopieren

Die benutzten Protokolle der `r*-Befehle` (Datei `/etc/services`):

```
exec        512/tcp         # remote process execution
login       513/tcp
shell       514/tcp
```

Ein- und Ausschalten der r*-befehle in `/etc/inetd.conf`:

```
shell   stream  tcp     nowait  root    /usr/sbin/tcpd  in.rshd -L
# shell stream  tcp     nowait  root    /usr/sbin/tcpd  in.rshd -aL
login   stream  tcp     nowait  root    /usr/sbin/tcpd  in.rlogind
# login stream  tcp     nowait  root    /usr/sbin/tcpd  in.rlogind -a
exec    stream  tcp     nowait  root    /usr/sbin/tcpd  in.rexecd
```

Die Dateien `/etc/hosts.equiv` (systemweit) oder `~/.rhosts` (benutzerweit) erlauben z.B. mit `rlogin` das Benutzer sich einloggen können ohne das Passwort eingeben zu müssen. Für `rcp` und `rsh` muss eine von beiden Datei präsent sein.

z.B. die Datei `/etc/hosts.equiv` auf dem Rechner `sun`:

```
sirius
earth
```

bedeutet, dass alle Benutzer von den Rechnern `sirius` und `earth` sich ohne Passwort auf dem Rechner `sun` einloggen können.
Es ist empfehlenswert statt der Datei `/etc/hosts.equiv`, die Datei `~/.rhosts` zu benutzten, weil sie mindestens nur für einen einzigen Benutzer gültig ist.

Benutzung:

```
    rlogin [-l Benutzername] Rechnername
*   rcp [-r -p] Datei Rechnername:[Pfad]Dateiname
    (r=recursive, p=preserve)
    rcp Rechnername:[Pfad]Dateiname Dateiname
    rsh Rechnername Befehl
```

* **Achtung**, der Befehl `rcp` funktioniert nur wenn alle `echo-Meldungen` in den Dateien `/etc/profile`, `/etc/profile.local`, `~/.bashrc` und `~/.profile` weg sind!

- ## SSH - Secure Shell

Prinzip:
Der **ssh Dämon** funktioniert nach dem Client-Server Prinzip:
Jeder Rechner besitzt einen rechnerspezifischen Schlüssel (1024 Bits), um sich identifizieren zu können. Wenn der Dämon startet, wird ein zusätzlicher Server-Schlüssel generiert (768 Bits). Wenn dieser Schlüssel benutzt wurde, wird er normalerweise jede Stunde neu generiert. Er wird nur im Arbeitsspeicher deponiert (also nie auf der Festplatte).
Wenn ein Client sich mit dem ssh Dämon verbindet, sendet der Dämon seine öffentlichen Rechner- und Server-Schlüssel. Der Client kontrolliert den Rechner-Schlüssel mit seiner eigenen Datenbank, ob er sich nicht verändert hat. Der Client generiert dann eine zufällige 256 Bit-Zahl. Er verschlüsselt diese zufällige Zahl mit den beiden Rechner- und Server-Schlüsseln und sendet diese verschlüsselte Zahl dem Server zurück. Beide Seiten (Server und Client) benutzen jetzt diese verschlüsselte Zahl als Session-Schlüssel. Der Session-Schlüssel wird ab jetzt gebraucht, um alle folgenden Kommunikationen zu verschlüsseln.

RSA-Authentifizierung: Jeder Benutzer besitzt einen öffentlichen und einen privaten (oder geheimen) Schlüssel. Der öffentliche Schlüssel wird dem Server gegeben, aber nur der Benutzer kennt seinen privaten Schlüssel.
Der Server macht die Verschlüsselung mit dem öffentlichen Schlüssel des Benutzers, die Entschlüsselung kann aber nur von dem Benutzer gemacht werden, mit seinem privaten Schlüssel.

Installation & Konfiguration:
Pakete `ssh` (mit RSA) oder `openssh` (mit RSA oder DSA) beide in der Serie `sec`
Der ssh Dämon benutzt den Port 22.
Mit YaST `start_sshd` auf `yes` setzten (nur bis SuSE 7.3).

(open) sshd Konfigurationsdatei: `/etc/ssh/sshd_config` oder `/etc/sshd_config`
ssh client systemweite Konfigurationsdatei: `/etc/ssh_config` oder `/etc/ssh_config`
Die Schlüssel werden mit ssh-keygen generiert.
Privater RSA-Schlüssel: `~/.ssh/identity` oder `id_rsa`
Öffentlicher RSA-Schlüssel: `~/.ssh/identity.pub` oder `id_rsa.pub`
Standart Schlüsselgrösse: 1024 Bits
Liste der bekannten Rechner: `~/.ssh/known_hosts`
Öffentlicher Schlüssel des Rechners: `/etc/ssh_host_key.pub`

sshd starten & stoppen: `rcsshd start | stop | restart | status`

Anwendung:
Wenn man sich das erste Mal mit `ssh` oder `slogin` auf einem entfernten Rechner einloggt, wird die folgende Meldung kommen:

```
ssh moon
```
The authenticity of host 'moon' can't be established.
RSA key fingerprint is eb:c8:f4:b5:9f:d2:c7:c9:17:e9:d1:44:77:f4:64:88.
Are you sure you want to continue connecting (yes/no)? `yes`
Warning: Permanently added 'moon,192.168.11.15' (RSA) to the list of known hosts.

Das bedeutet, dass der öffentliche Schlüssel des entfernten Rechners sich noch nicht in der Datei "known_hosts" des lokalen Rechners befindet. Nach der Antwort "yes" wird dann diese Datei kopiert.

- **Sich auf einem entfernten Rechner einloggen:**
  `ssh` oder `slogin` *`Rechnername`* slogin ist ein symbolischer Link auf ssh
  `ssh [-l` *`Benutzername`*`] Rechnername`
  oder `ssh` *`Benutzername@Rechnername`*

  `ssh -P` *`Rechnername`* -P = benutzt nicht privilegierte Ports >= 1024
- **Einen Befehl auf einem entfernten Rechner ausführen:**
  `ssh` *`Rechnername Befehl`*
  oder `ssh` *`Benutzername@Rechnername`*
- **Eine lokale Datei auf einen entfernten Rechner kopieren:**
  `scp [-p]` *`Datei Rechnername`*`:[`*`Verzeichnis/`*`][`*`Datei`*`]`
  -p = Preserve
  oder zusätzlich mit Benutzername:
  `scp Datei` *`Benutzername@Rechnername`*`:[`*`Verzeichnis/`*`][`*`Datei`*`]`
- **Eine Datei von einem entfernten Rechner auf den eigenen Rechner kopieren:**
  `scp` *`Rechner`*`:[`*`Verzeichnis/`*`][`*`Datei`*`] [`*`Verzeichnis/`*`][`*`Datei`*`]`
- **Eine Struktur (Verzeichnisbaum) kopieren**
  `scp -r` *`Verzeichnis Rechnername`*`:`*`Verzeichnis/`* -r = Rekursiv
  oder
  `scp -r` *`Rechnername`*`:`*`Verzeichnis Verzeichnis/`*

Authentifizierung

Authentifizierung bedeutet die Erkennung oder der Beweis der Benutzeridentität. Autorisierung im Gegensatz bedeutet, dass ein authentifizierter Benutzer bestimmte Rechte hat oder bekommt. Wenn keine Schlüsseln erstellt worden sind, dann wird die Authentifizierung über den regulären Benutzernamen und das Passwort gemacht. Das ist heutzutage kein sehr sicheres Verfahren.
Sicherer ist das Dopelschlüssel-Verfahren (Public Key & Privat Key). Die Verschlüsselung-Algorithmen sind RSA und DSA.
Das Schlüsselpaar kann mit dem Befehl `ssh-keygen` erstellt werden.
`ssh-keygen` verlangt eine sogenannte `Passphrase`. Das kann ein ganzer Satz sein, der sogar Leerzeichen enthalten kann. Bei der Authentifizierung wird dann diese Passphrase statt des Benutzernamens und des Passworts verlangt. Die Passphrase wir nicht übers Netzwerk übermittelt, sondern wird lokal überprüft.

- **Schlüssel erstellen und verteilen:**
  Es ist komfortabel Dateien von einem auf einen anderen Rechner kopieren zu können ohne dauernd ein Passwort eingeben zu müssen, ist aber eine potenzielle Sicherheitsgefahr. Wenn ein Rechner kompromittiert worden ist, werden die anderen Rechnern es sehr bald auch sein. Eine Lösung um Komfort und Sicherheit zu haben ist der `ssh-agent`.
  Vorgehensweise:
  1. `ssh-keygen -t dsa` oder `ssh-keygen -t rsa` ausführen, die Passphrase eingeben (oder leer lassen, aber Achtung Sicherheitsloch!) *****

die folgenden Dateien werden somit erstellt: `~/.ssh/id_dsa` & `id_dsa.pub` bzw. `id_rsa & id_rsa.pub`
mit `-t rsa1` können auch die Schlüssel für das Protokoll Version 1 RSA erstellt werden, die Dateien heißen dann `identity` & `identity.pub`.

2. Das erste Mal, bevor die Datei `authorized_keys` auf dem entfernten Rechner existiert:
```
scp ~/.ssh/id_dsa.pub \
 entfernten-Rechner:~/.ssh/authorized_keys
```
oder wenn `authorized_keys` schon einen Eintrag hat:
```
scp ~/.ssh/id_dsa.pub \
entfernter-Rechner:~/.ssh/last_key
```
auf dem entfernten Rechner:
```
cd ~/.ssh
cat  last_key  >>  authorized_keys
```
3. auf dem entfernten Rechner:
```
chmod 600 ~/.ssh/authorized_keys
```

***** Die Passphrase ist sinnvoll, wenn man keine root-Rechte auf einen Rechner besitzt, um sich also vor dem Administrator schützen zu können oder einfach als Sicherheitsgrund.

Wenn man eine Passphrase eingegeben hat, kann man es trotzdem temporär (so lange man eingeloggt ist) ausschalten:
ssh-agent aufrufen.
etwas ähnliches wird am Bildschirm angezeigt:

```
SSH_AUTH_SOCK=/tmp/ssh-XX91bJ2S/agent.18359; export SSH_AUTH_SOCK;
SSH_AGENT_PID=18360; export SSH_AGENT_PID;
echo Agent pid 18360;
```

die zwei ersten Zeilen separat mit der Maus kopieren, dann einfügen und mit <Return> jede einzelne Zeile ausführen.
ssh-add [`~/.ssh/id_dsa`] aufrufen, die Passphrase eingeben und das ist es schon! :-)

- **X-Forwarding mit ssh**
  Mit ssh kann man sich auch auf einem entfernten Rechner einloggen, eine X-Anwendung starten und die Ausgabe direkt auf dem eigenen (lokalen) Rechner bekommen ohne ein `export DISPLAY=:0.0` machen zu müssen!

  `ssh -X` *`Rechnername`*

  Wenn man `echo $DISPLAY` eingibt, sieht man das folgende oder ähnliche Resultat: `localhost:10.0`

  Es ist auch möglich das X-Forwarding global einzuschalten sodass es nicht mehr notwendig ist die Option -X einzugeben (was ist der Vorteil?)
  In der Datei `/etc/ssh/ssh_config` (die systemweite Konfigurationsdatei des ssh-Klienten), muss `ForwardX11 yes` gesetzt werden.

- **Tunnelling - TCP Port-Forwarding mit ssh:**

  Eine phantastische Eigenschaft von ssh ist das Tunnelling von Protokollen. Das bedeutet, dass eine unsichere Verbindung (Übermittlung der Daten in Klartext) mit ssh ummantelt wird, und dadurch wieder sicherer wird. Zum Beispiel kann die Authentifizierung des Programmes Webmin auf dem Port 10000 mit ssh verschlüsselt werden, oder die Authentifizierung von SWAT (Port 901), oder die von CUPS (Port 631) usw.

  ```
  ssh [-f] entfernten-Rechner -L lokaler-Port:entferten-Rechner:entferten-Port [sleep Sekunden]
  ```

  Beispiel:
  ```
  ssh -f laptop -L 7777:laptop:901 sleep 600
  ```
  (Passwort oder Passphrase eingeben)

  -f = fork⇒ sendet der Prozess im Hintergrund, -L= Local Forward
  Dann mit einem Browser die folgende Adresse eingeben: `http://localhost:7777`

  Der erste Befehl öffnet eine ssh-Verbindung für 600 Sekunden und leitet gleichzeitig alle potentiellen Verbindungen vom lokalen Port 7777 auf den entfernten Rechner "laptop" Port 901 (Samba Management Programm swat) weiter.
  Mit dem Browser, wird der lokale Port 7777 geöffnet, ssh macht dann automatisch die Weiterleitung auf den Rechner "laptop" port 901.

  Dieser Befehl könnte noch einfacher aussehen:

  ```
  ssh laptop -L 7777:laptop:901
  ```

  Der einzige Unterschied ist, dass der X-Term besetzt bleibt.

  Windows-SSH-Klienten:

  `putty.exe` (telnet und ssh), `pscp.exe` (scp), erhältlich von:
  `http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html`

• Dateien und Verzeichnisse mit rsync und rdist synchronisieren

Es ist möglich Verzeichnisse oder Dateien von zwei Rechnern zu synchronisieren, das bedeutet, dass ein Verzeichnis mit Inhalt und Unterverzeichnis von einem Rechner auf einen anderen Rechner exakt kopiert wird (Befehl `rsync`), oder sogar auf mehrere Rechner kopiert wird (Befehl `rdist`).

- **rsync**

  Syntax:

  `rsync [-Optionen] -e ssh` *`/lokale-Quelle`* `[`*`Benutzer@`*`]`*`entfernten-Rechner`*`:`*`/entferntes-Ziel`*

  `rsync [-Optionen] -e ssh [`*`Benutzer@`*`]`*`entferten-Rechner`*`:`*`/entfernte-Quelle /lokales-Ziel`*

  Beispiel:

```
rsync -vazu --delete --force --exclude=Klausur -e ssh \
/home/linux/ pierre@sun:/home/linux/
```

  Der Befehl Macht ein rekursives Update vom Inhalt des Verzeichnisses `/home/linux/` vom lokalen Rechner zum Rechner `sun` im gleichnamigen Verzeichnis `/home/linux`

  Achtung: wenn das Quellverzeichnis mit einem `/` endet, bedeutet das, dass der Inhalt des Verzeichnisses kopiert wird und nicht das Verzeichnis selbst!!

  `-a` = Archive-Modus - bewahrt die Zugriffsrechte, Eigentümer usw.
  ähnlich wie die Option `-a` von dem `cp`-Befehl
  `-u` = Update - Überschreibt nicht neuere Dateien
  `-v` = Verbose - zeigt den Vorgang an
  `-z` = gzip-Komprimierung - es wird vor dem Transfer komprimiert
  `-n / --dry-run` = zeigt was transferiert würde aber ohne es auzuführen
  `--delete` = löscht alle Dateien, die auf dem Zeilverzeichnis existieren aber nicht auf dem Quell-Verzeichnis
  `--force` = zwingt das Löschen von Verzeichnissen, auch wenn sie nicht leer sind.
  `--exclude` = Datei-Ausnahmen die nicht zu kopieren sind.

  Anmerkung: `--force` im Zusammenhang mit `--delete` erlaubt eine exakte Kopie von Verzeichnissen, in Englisch wird das `mirroring` (Spiegelung) genannt.

- **rdist**

  noch in Berarbeitung...

- ## Weitere (veraltete) TCP/IP-Dienste

  - ### Finger

    `finger` zeigt Benutzer-Informationen an, auch über's Netzwerk

    Syntax:
    `finger [-Optionen] [`*`Benutzer`*`][@`*`Rechnername`*`]`
    `finger`
    `finger michel`
    `finger -l user1@bts02doz` `-l` = langes Format
    `finger @sun`
    der obere Befehl zeigt alle Benutzer die auf dem Rechner `sun` eingeloggt sind.
    Damit dieser Dienst funktioniert, muss der Finger-Dämon installiert sein und in `/etc/inetd.conf` eingeschaltet sein:

    ```
    finger stream tcp nowait  nobody /usr/sbin/tcpd in.fingerd -w
    ```

    Aus Sicherheitsgründen ist der Finger-Dämon (endlich) ab SuSE 8.0 weder installiert, noch in /etc/inetd.conf eingeschaltet.

  - ### Talk

    `talk` ist der Urvater des Chattings. In der Zeit von E-Mail, Handys & Co, sind solche Werkzeuge kaum mehr benutzt.

    Syntax:
    `talk` *`Benutzername`*`[@`*`Rechnername`*`] [`*`Terminal`*`]`

    antworten mit:
    `talk` *`anrufender-Benutzername`*`[@`*`Rechnername`*`]`

    Beispiel: `talk pierre@sun pts/0`

    mit `tty` kann der eigene Terminalname herausgefunden werden.

    mit `finger` *`Benutzername@Rechnername`* oder `ssh` *`Rechnername`* `w` können die Benutzer- und Terminalnamen auf dem entfernten Rechner herausgefunden werden.
    Damit `talk` funktioniert, muss der Talk-Dämon installiert sein und die folgende Einträge in `/etc/inetd.conf` eingeschalted sein:

    ```
    talk  dgram udp wait root /usr/sbin/tcpd in.talkd
    ntalk dgram udp wait root /usr/sbin/tcpd in.talkd
    ```

NFS - Network File System

- **Dokumentation:**
  NFS-HOWTO, man nfs, man mount

- **NFS-Server**
  Es gibt zwei verschiedene Typen von NFS-Servern unter Linux, den Kernel NFS-Server und den User-Space NFS-Server. Der Kernel NFS-Server läuft als Modul im Kernel, ist dadurch schneller als der andere, aber noch nicht alle Funktionen des User-Space NFS-Servers sind implementiert. Der User-Space NFS-Server ist weniger gefährlich als der Kernel NFS-Server, weil er der Kernel nicht zum Abstürzen bringen kann, benutzt aber mehr Prozessor-Ressourcen. Unter SuSE wird standardmässig der Kernel NFS-Server installiert.

- **Kernel NFS-Server:**
  Ab SuSE 7.2 Paket `nfs-utils` Serie n
  Bis SuSE 7.3 sind die Parameter in `/etc/rc.config`
  Ab SuSE 8.0 sind die Parameter in `/etc/sysconfig/nfs`

  `NFS_SERVER = "yes"` (nur bis SuSE 7.3)
  `USE_KERNEL_NFSD_NUMBER = "4"`

- **User-Space NFS-Server:**
  Ab SuSE 7.2 Paket `nfs-server` Serie n
  Parameter in `/etc/rc.config`:
  `NFS_SERVER = "yes"` (bis SuSE 7.3) sonst ab SuSE 8.0 über den Runlevel-Editor.

- **Konfiguration des NFS-Servers**

  Die Verzeichnisse, die für andere Rechner zur Verfügung gestellt werden, werden in der NFS-Terminologie `exportiert`. Alle diese Verzeichnisse werden in der Datei `/etc/exports` definiert.

  Beispiele für die Datei `/etc/exports`:

```
/home/pierre  *.linux.local(rw)
/home/user1   192.168.70.0/255.255.255.0(ro)
/home/marc    0.0.0.0/0.0.0.0(rw)        ist nicht gleich wie nur (rw) !
/root         SUN(rw,no_root_squash) EARTH(ro,root_squash)
/cdrom        (ro)
```

  Erklärung:

  ro und rw = read only (nur lesen) und read/write (lesen und schreiben)
  root_squash = der Benutzer `root` bekommt die UID von `nobody` und die GID von `nogroup` auf dem NFS-Server (Standard).
  no_root_squash = root bekommt wirklich die root-Rechte.

  Nach einer Änderung der Datei `/etc/exports` muss der NFS-Server neu gestartet werden:
  `rcnfsserver restart` oder `reload`
  oder `rcnfsserver stop` und dann `rcnfsserver start`

- **NFS-Klient**
  Die Benutzung eines exportierten Verzeichnises aus der Sicht des NFS-Klienten wird über den Befehl `mount` gemacht. Das kann normalerweise nur der Benutzer `root` machen. `root` kann aber einen Eintrag in die Datei `/etc/fstab` einfügen, so dass ein regulärer Benutzer das "mounten" auch ausführen kann.
  Der `Mountpoint` oder Anhängepunkt ist ein **existierendes** und **leeres** Verzeichnis (wenn der Mountpoint noch nicht existiert, muss logischerweise das Verzeichnis z.B. mit `mkdir` erstellt werden...).

  Mount-Befehl(als root):

  `Mount [-t nfs] Rechner:/Exportiertes-Verzeichnis /Mountpoint`

  z.B. `mount nfsserver:/public /mnt/nfs`

  Eintrag in der Datei /etc/fstab:

  `Rechner:/Exportiertes-Verzeichnis /Mountpoint nfs Optionen 0 0`

  z.B. `nfsserver:/public /mnt/nfs nfs noauto,user 0 0`

  Mount-Befehl als Benutzer:
  Nachdem `root` den entsprechenden Eintrag in die `fstab` eingefügt hat, kann der reguläre Benutzer den Mount-Befehl nur noch mit dem Mountpoint eingeben:

  `mount /Mountpoint`

  Ein Verzeichnis wieder abhängen:
  Als root und als regulärer Benutzer

  `umount /Mountpoint`

  Alle angehängten Partitionen und Verzeichnisse anschauen:

  `mount` (ohne Parameter)

  NFS-Mount-Optionen

  Beispiel-Zeile in `/etc/fstab`:

  `server:/pub /mnt/pub nfs rsize=8192,wsize=8192,timeo=14,intr`

  oder über den Mount-Befehl:

  `mount -o rsize=8192,wsize=8192,intr server:/pub /mnt/pub`

`rsize=8192`	Die Größe in Byte, die NFS benutzt, um Dateien von einem NFS-Server zu lesen.
`wsize=8192`	Die Größe in Byte, die NFS benutzt, um Dateien auf einen NFS-Server zu schreiben. Diese beiden Optionen (rsize & wsize) können NFS stark beschleunigen.
`timeo=14`	Timeout in Zehnteln von Sekunden (Standard ist 7), bevor ein neuer Verbindungsversuch gesendet wird, nach einem RPC-Timeout. (Siehe bitte `man nfs` für die ganze Erklärung dieses komplizierten Mechanismuses).

`intr`	Der Klient-Prozess kann, wenn der NFS-Server nicht antwortet, unterbrochen werden. Normalerweise versucht ein Prozess ca. eine Woche, bevor er aufgibt!
`retrans=6`	Die Anzahl der "minor timeouts", die passieren müssen , bevor ein "major timeout" passiert. Der Standardwert ist 3. Nach einem "major timeout" wird der Verbindungsversuch mit der Meldung "server not responding" endgültig beendet.

Der Portmapper auf dem NFS-Klient:
Der sogenannte Portmapper-Dämon muss auf dem Klientrechner laufen. Der Portmapper findet über RPC (Remote Procedure Call) heraus, auf welchem Port sich der angefragte Dienst befindet. Der NFS-Server befindet sich meistens auf dem Port 2049.

Der Portmapper ist bei SuSE im Paket `portmap` Serie n
Konfigurationsdatei (bis SuSE 7.3): `/etc/rc.config`:
`START_PORTMAP = "yes"`

- den Portmapper starten / stoppen: `rcportmap start / stop`

- **NFS-Hilfsprogramme:**

`showmount -a` (alles=Hosts + Verzeichnisse) `-d` (Verzeichnisse) `-e` (Exportliste)

`showmount -a` *`Rechnername`*	zeigt die exportierten Verzeichnisse eines entfernten Rechners
`nfsstat`	(Statistik über RPC, NFS Server und NFS Client)
`exportfs [-Optionen]`	(erlaubt, einen Mountpoint dynamisch zuzufügen oder wegzunehmen ohne die Datei `/etc/exports` zu verändern)
`exportfs -v`	zeigt alle exportierten Verzeichnisse mit den Optionen (besser als showmount)
`rpcinfo -p`	zeigt alle RPC-Programme und ihre Ports (p für Programme)

- **Wenn die UIDs auf Server und Klienten ungleich sind:**

  Normalerweise sollten die UIDs (Benutzer ID) auf dem Server und auf dem Klienten gleich sein, so dass nach dem "mounten" der Benutzer die gleichen Zugriffsrechte auf die Dateien, die auf dem Server sind, bekommt. Wenn wegen einer schlechten Netzwerk-Planung das nicht der Fall ist, ist es möglich, eine Übersetzungstabelle mit der Hilfe eines zusätzlichen Dämons zu erstellen.

  Soweit ich weiß, funktioniert das nur mit dem User-Space NFS-Server!

  Auf dem Server muss für jedes exportierte Verzeichnis ein zusätzlicher Eintrag in die Datei `/etc/exports` eingefügt werden:
  `/home/user1 (rw,map_daemon)`

  Auf dem Klient muss der UID-GID-Mapper-Dämon manuell gestartet werden:
  `rpc.ugidd`

  oder bis SuSE 7.3 in `/etc/rc.config`: `NFS_SERVER_UGID = "yes"`

- **Troubleshooting, NFS-Caches:**

`/var/lib/nfs/xtab`	Cache aller Klienten-Verbindungen. Kann gelöscht werden, aber dann sollte der NFS-Server "reloaded" werden (`rcnfsserver reload`).
`/var/lib/nfs/rmtab`	Klienten-Tabelle, darf nicht gelöscht werden aber kann geleert werden, hilfreich wenn der NFS-Server nicht mehr starten will, nachdem viele Klienten Verbindungen hatten.

DHCP - Dynamic Host Configuration Protocol

DHCP-Server

SuSE-Paket: `dhcp-server` (+ Paket `dhcp-base`)
Programm: `dhcpd`
Start/Stop-Script: `/etc/init.d/dhcpd` oder `rcdhcpd`
Konfigurationsdatei: `/etc/dhcp.conf`
Systemvariablen: `/etc/syscondig/dhcpd` (Startparameter für dhcpd)
Dynamische Daten: `/var/lib/dhcp`
Hilfe: `man dhcpd.conf` und `man dhcpd`

SuSE-Paket: `dhcp-tools`
Programme: `dhcpdump` und `dhcping`

Konfigurations des Servers (Version 3)

Unter SuSE 8.0 muss man zuerst eine Konfigurationsdatei erstellen. Eine Beispielkonfigurationsdatei kann von `/usr/share/doc/packages/dhcp-server` kopiert werden.

Beispiel einer minimale Konfiguration von dhcp.conf:

```
option domain-name "stars.priv";
option domain-name-servers venus.stars.priv;
default-lease-time 600;        die Werte sind in Sek. = 10 Minuten
max-lease-time 7200;           = 2 Stunden
authoritative;                 wenn das der Haupt DHCP-Server ist.
subnet 192.168.10.0 netmask 255.255.255.0 {
range 192.168.10.100 192.168.10.150;
option routers venus.stars.priv;
}
ddns-update-style none;
```

und das reicht schon für ein ersten Versuch!

Erster Versuch - der DHCP-Server starten:

`/etc/init.d/dhcpd start` oder `rcdhcpd start`

Auf dem Klienten der DHCP-Klient starten:

(auch als root) `dhcpcd -d` oder `dhclient` ausführen, je nachdem welchen DHCP-Klienten installiert worden ist.

- **Kontrolle:** `ifconfig` sollte jetzt die IP-Adresse die an `eth0` vergeben worden ist zeigen.
  Wenn `dhcpcd` als Klienten gestartet worden ist, kann man auch viele DHCP-Informationen in der Datei `/var/lib/dhcpcd/dhcpcd-eth0.info` anschauen.

- **Fixe IP-Adresse für ein DHCP-Klienten-Rechner**

  Konfiguration in /etc/dhcp.conf:

```
host sun {
         hardware ethernet 00:40:F4:3F:5A:A8;
         fixed-address sun.stars.priv;
}
```

- **DHCP-Klienten**

  SuSE-Paket: `dhcpcd` (is standardmässig bei SuSE installiert)
  Programm: `dhcpcd`
  Hilfe: `man dhcpcd`

  oder

  SuSE-Paket: `dhcpcd-client`
  `Programm`: `dhclient`
  Konfigurationsdatei: `/etc/dhclient.conf`
  Dynamische Daten: `/var/lib/dhcpcd`
  Hilfe: `man dhclient.conf` und `man dhclient`

  Für beide Pakete:
  Systemvariablen: `/etc/syscondig/network/dhcpd`

  Bei RedHat und bei Debian wird das Programm `dhclient` eingesetzt. Das Paket heißt `dhclient` unter RedHat, und `dhcp-client` unter Debian.

- ## Netzwerk-Fehlersuche - Troubleshooting
  - **ethereal**

    `etheral` ist ein schönes Netzwerkanalyse-Werkzeug (SuSE Paket `ethereal` Serie n). Um alle Funktionalitäten von `ethereal` nutzen zu können, muss es als root gestartet werden. Die folgende Anleitung ist für die Version 0.9.0 gültig.

    ```
    xhost +localhost
    su - (Passwort von root eingeben)
    export DISPLAY=:0.0
    ethereal &
    ```

    auf Capture... klicken
    auf Start... Ctl+K klicken
    Ethereal: Capture Preferences
    Interface auswählen, z.B. `eth0`
    ev. auf `Filter:` klicken

    Ethereal: Capture Filter
    Filter name: `smb`
    Filter string: `udp port 137 or udp port 138 or tcp port 139`
    auf `New` klicken
    auf `Save` klicken
    auf `Ok` klicken

    ❒ Capture packets in promiscuous mode
    ❒ Use ring buffer
    ☑ Update list of packets in real time
    ☑ Automatic scrolling in live capture
    ❒ Enable MAC name resolution
    ❒ Enable network name resolution
    ❒ Enable transport name resolution
    auf `OK` klicken

    Ein "Display-Filter" kann direkt auf dem unteren Fensterrand definiert werden: `Filter:`
    zuerst `Filter name:` und `Filter string:` eingeben,
    und nur dann auf `New`, auf `Save` und auf `Apply` klicken!

    "Display-Filter" Beispiele:
    ```
    udp.port == 137 || udp.port == 138 || tcp.port == 139
    ```

    `Capture-Filters` haben das gleiche Format wie `tcpdump`.
    `man tcpdump` zeigt viele Filter-Beispiele.
    `Display Filters` werden im Help von `ethereal` erklärt, auch mit vielen Beispielen.

    Ein TCP-Paketfluss (stream) in Klartext lesen: eine der "schönste" Funktion von ethereal ist die Möglichkeit eine ganze TCP-Verbindung in Klartext anzuschauen. Im Resultat-Fenster von etheral mit der linken Maustaste ein Paket markieren, und dann mit der rechten Maustaste auf das Menü "Follow TCP Stream" dücken!

- **tethereal** die Terminal-Version von ethereal
  `tethereal -i eth0`
  `tethereal -i eth0 -f port 23` (capture filter im tcpdump Format)
  `tethereal -i ippp0 -R "tcp.port==23"` (read filter)

- **ngrep**
  `ngrep -d eth0`
  `ngrep -d eth0 port 23` gut um eine Telnet-Session zu überwachen :-)
  `ngrep -d eth0 port 23 and \(host sun or laptop\)`
  `ngrep -d eth0 port 23 and net 192.168.10`
  `ngrep -d eth0 port 23 and net 192.168.10.0 mask 255.255.255.0`
  `ngrep -d eth0 icmp[0] = 0 or icmp[0] = 8` (0=echo request/ping, 8=echo reply)
  `ngrep -d eth0 icmp[0] != 0 and icmp[0] != 8`

- **tcpdump**
  `tcpdump-i lo` tcpdump hat sehr viele Filter Möglichkeiten. siehe man tcpdump.
  z.B. `tcpdump -i eth0 ip host sirius and not sun`

- **iptraf** muss in einem Terminal aufgerufen werden
  Menüs:
  `IP traffic monitor` überwacht die IPs und Ports
  `TCP/UDP service monitor` überwacht die Protokolle und die Ports

- **netstat**
  `netstat -a /--all` zeigt alles an, TCP, UDP, Unix Sockets
  `netstat -p /--programs` zeigt nur Programme an
  `netstat -n /--numeric` macht keine Namenauflösung, zeigt die IP-Adressen und Ports numerisch an
  `netstat -r /--route` zeigt die Routing-Tabelle an
  `netstat -l /--listening` zeigt nur Verbindungen wo ein Server zuhört
  `netstat -t /--tcp` zeigt nur TCP-Verbindungen an
  `netstat -u / --udp` zeigt nur UDP-Verbindungen an
  `netstat --inet /--ip` zeigt die Internet Protkoll-Familie: TCP, UDP an

- **knetdump** schönes KDE-Programm. Zeigt graphisch die Verbindungen zwischen Rechnern.
  Steuerung über `Einstellungen` und `Ansicht`.

- **ksniffer** auch ein KDE-Programm, gut für Statistics

- **tleds -d 100 ippp0** Tastatur-Leds blicken wenn Daten übertragen werden

- **ntop -i eth0 -r 1** -r = refresh.

- **traceroute** *Hostname* zeigt die Gateways/Router die auf dem Weg bis zum zum Zeil-Rechner sind an.

Linux und Unix Bücher fürs Netzwerk

Titel: Linux - Wegweiser für Netzwerker, 2. Auflage (gut bis sehr gut)
Author: Olaf Kirch & Terry Dawson
Verlag: O'Reilly
ISBN: 3-89721-135-1, Preis: 38,-€

Titel: Linux im Netz (ganz OK bis gut)
(Aktualisiert und erweiterte Neuausgabe)
Author: Dr.Bernhard Röhrig
Verlag: C&L
ISBN: 3-932311-61-2 Preis: 50,11 €

Titel: Linux Intern (sehr gut, nur über "second hand" erhältlich)
Author: M.Wielsch, J.Prahm, H.G. Eßer
Verlag: Data Becker
ISBN: 3-8158-1292-5

Titel: Linux im Windows-Netzwerk (2002, gutes Buch)
Author: Debacher, Kretschmer, Burre & Schultz
Verlag: Franzis'
ISBN: 3-7723-6066-1 Preis: 45,95 €

Titel: DNS und Bind (gut, 2002, 4. Auflage, mit Bind 9)
Author: Paul Albitz & Cricket Liu
Verlag: O'Reilly
ISBN: 3-89721-290-0 Preis; 46,- €

Titel: Professional Apache (englisch / eins des beste für Apache)
Author: Peter Wainwright
Verlag: Wrox Press Ltd.
ISBN: 1-861003-02-1 Preis: 62,95 €

Titel: Apache Web-Server (2000, 3. Auflage, nicht das beste für Anfänger)
Author: Lars Eilebrecht
Verlag: mtip
ISBN: 3-8266-0612-4 Preis: bei Amazon nicht mehr erhältlich?

Titel: Apache Web Server Administrarion (soll gut sein)
Author: Charles Aulds
Verlag: Sybex
ISBN: 3-8155-0322-1 Preis: 46,- €

Titel: Samba. (2. Auflage)
Author: Jens Kühnel
Verlag: mtip
ISBN: 3-8266-0620-5 Preis: 30,70 €

Titel: Das Samba Buch (2001, 3. Auflage, beinhaltet Samba 2.2.2)
Author: Olaf Borkner-Delcarlo
Verlag: SuSE PRESS
ISBN: 3-935922-15-9 Preis: 51,- €

Titel: Samba. Ein Datei- und Druckserver für heterogene Netzwerke.
Author: Robert Eckstein, David Collier-Brown, Peter Kelly
Verlag: O'Reilly
ISBN: 3-89721-161-0 Preis: 38,- €