Unix I – Einführung in das Unix Betriebssystem

Stephan Thesing

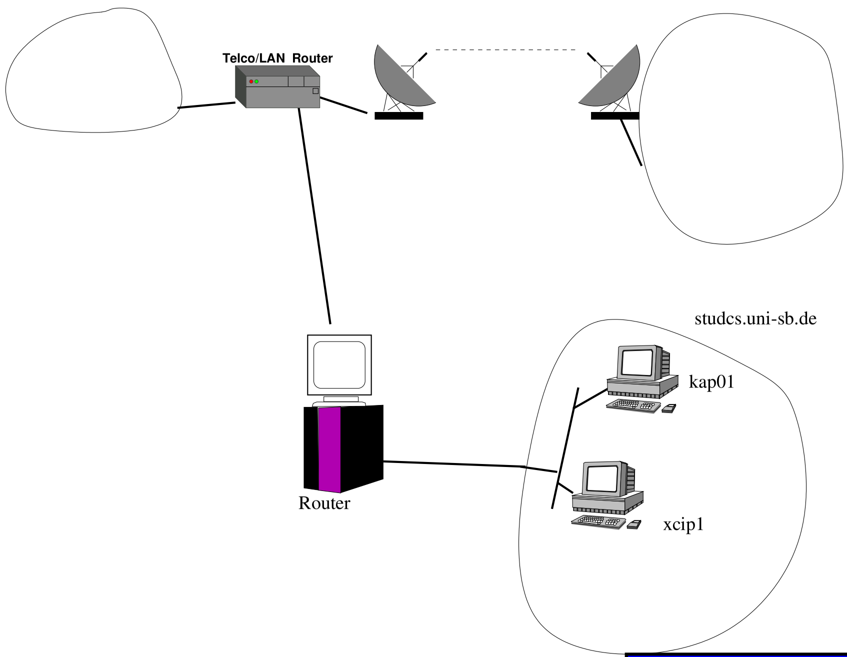
Netzwerke

- Grundlagen des Internet
- Dienste
  - telnet
  - FTP

’The’ Internet

Eine Zusammenschaltung vieler kleinerer LANs über Router.

Benutzen einheitliche Protokolle zur Datenübermittlung.



Universität des Saarlandes

Daten müssen in einheitlicher Form versandt werden, damit Rechner sie erkennen können.

Dazu definiert man *Protokolle*, die das Aussehen der Daten festlegen.

Es gibt mehrere Ebenen von Protokollen:

DNS	NTP	NFS	WWW	FTP	SMTP	NNTP
UDP			TCP			
IP Protokoll						
Physikalisch (Ethernet)						

Die Daten gehen in Paketen über das Netz.

Diese enthalten neben den Nutzdaten auch noch Protokollinformation:

- Absender, Adressat
- Checksummen
- Optionen

$$\overbrace{\texttt{SRC|DST|}\underbrace{\texttt{SRC IP| DST IP|}\underbrace{\texttt{Ports|Data}}_{TCP}}_{IP}}^{Ethernet}$$

Im Internet werden Rechner intern ausschliesslich Über ihre *Adresse* identifiziert.

IP Adressen sind 4 Bytes lang:

127.0.0.1
134.96.220.1
134.96.252.20

Globale IP Adressen sind eindeutig.

Ausnahmen: $127.x.y.z$, $10.x.y.z$, ...

Eine IP Adresse besteht aus einer *Netz-Kennung* und einer *Host-Kennung*.

Zur leichteren Identifizierung (vom menschlichen Standpunkt aus) von Rechnern benutzt man *Namen*.

Das Namensschema ist hierarchisch, damit Konflikte vermieden werden können.

Es besteht aus *Domain* Suffixen und einem *Hostnamen*:

`www.cs.uni-sb.de`

- Top-Level Domain `de`: Deutschland
- `uni-sb.de`: Uni-Saarbrücken, in Deutschland
- `cs.uni-sb.de`: Informatik, Uni-SB, Deutschland
- `www.cs.uni-sb.de`: Host `www` an der Informatik, Uni-SB, Deutschland

Um einen Rechner anzusprechen, der denselben Domain-Suffix teilt, reicht der Hostname.

`kap01` statt `kap01.studcs.uni-sb.de`.

Top-Level Domains:

de Deutschland

com Commercial (Firmen)

org Organisationen

edu Educational (Unis+Schulen in den USA)

gov Goverment (US Regierungsbehörden)

Jedes Land hat sein eigenes 2-stelliges Länderkürzel.

Allerdings sind in den USA nur wenige Domains unter `.us` registriert.

Die meisten laufen unter `.org`, `.com` und `.edu`.

Man kann *nicht* von dem Namen eines Rechners auf seinen Standort schliessen.

Die Zuordnung Name↔IP Adresse ist beliebig (solange sie eindeutig bleibt).

Diese Zuordnung wird über Datenbankserver abgewickelt, die die Zuordnungen speichern.

Lokale Zuordnungen stehen in der Datei `/etc/hosts`

Andere muss man bei einem *Nameserver* erfragen.

Dies läuft über das *DNS* Protokoll ab.

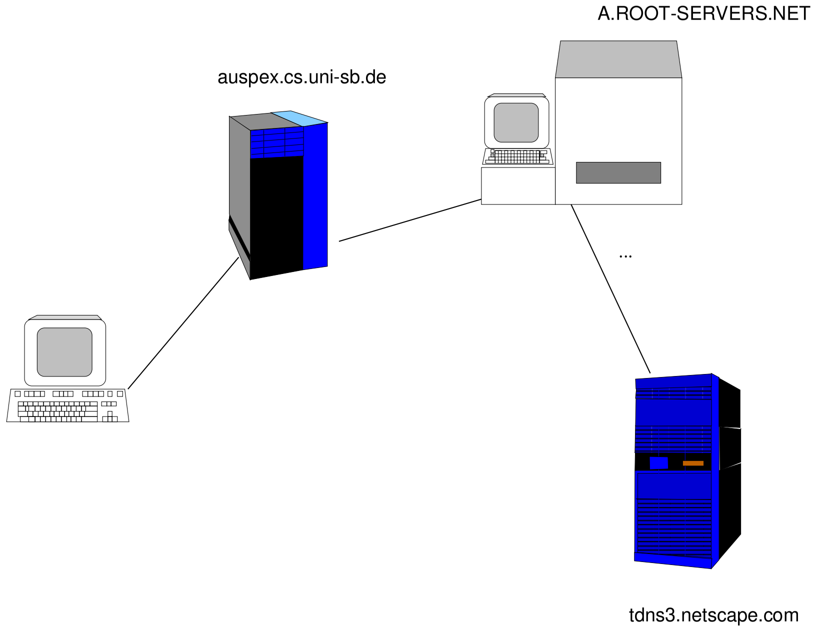
Das Programm `nslookup` erlaubt einem den einfachen Zugriff:

```
> nslookup www.cs.uni-sb.de
Server:   auspex.cs.uni-sb.de
Address:  134.96.220.254

Name:     www.cs.uni-sb.de
Address:  134.96.254.20
```

Man kann umgekehrt auch manchmal den Namen zu einer IP-Adresse finden.

Dies funktioniert nur, wenn die Server dieses *Reverse-Mapping* unterhalten.



TCP

Um eine stabile, fehlerfreie Verbindung aufzubauen, benutzt man das TCP-Protokoll.

Es ermöglicht den Aufbau eines sog. *Streams* zwischen zwei Rechnern.

Es beinhaltet automatisches Aussortieren von duplizierten Paketen und Neuübertragung von verlorenen Paketen.

Die Verbindung muss dazu natürlich eindeutig sein.

Damit man mehr als eine Verbindung zwischen zwei Rechnern aufmachen kann, gibt es die *Ports*.

Sie dienen als Haltestellen an einem Rechner.

TCP ist das meist benutzte Protokoll für Dienste im Internet.

Dienste werden dabei etwa auf einem Server bereitgestellt (etwa WWW).

Ein Client nimmt nun Verbindung zu dem Server und einem bestimmten Port auf diesem Server auf.

Damit der Client weiss, welchen Port er ansprechen muss, existiert eine Liste von sog. *well known ports*.

Diese sind in Standards (RFCs) festgelegt.

So hat z.B. WWW den well known Port 80.

In der Datei `/etc/services` sind einige Dienste mit ihren Port-Nummern aufgelistet.

Auf dem Server läuft ein Daemon, der auf Verbindungen an bekannten Ports wartet (inetd(8)).

Bei einem Verbindungsaufbau wird dann der zugehörige Server-Prozess gestartet.

Dieser empfängt Anforderungen vom Client und sendet Resultate zurück.

Die Art und das Aussehen der Anforderungen und Resultate hängt nun natürlich von dem betroffenen Dienst ab.

Dienste

Ein sehr nützlicher Dienst ist *telnet*. Er dient dazu, eine Terminal-Verbindung zu einem anderen Rechner aufzubauen.

Telnet hat die Port-Nummer 23.

Der Telnet Server auf der Zielmaschine stellt nun auf dem dortigen Rechner ein Terminal zur Verfügung und startet an diesem die Login-Prozedur.

```
> telnet kap01.studcs.uni-sb.de
Trying 134.96.220.1...
Connected to kap01.studcs.uni-sb.de.
Escape character is '^]'.

NetBSD/sparc (kap01) (ttyp1)

login:
```


Man kann mit **`telnet`** auch Verbindungen zu anderen Ports herstellen.

Dazu muss man nur die Port-Nummer als zweites Argument angeben.

```
> telnet www.cs.uni-sb.de 80
GET /index.html
```

Diese Verbindung überträgt aber die Zeichen nur literal zum Zielhost und zurück.

Telnet kann aber mehr.

Durch Optionen kann man bei einer normalen telnet-Verbindung Parameter aushandeln.

Etwa wird automatisch die `TERM` Umgebungsvariable auf dem Zielrechner gesetzt.

Durch einen Escape-Character (CTRL-]) kommt man bei laufender telnet-Verbindung in das Kommando-Menu.

```
telnet> help
Commands may be abbreviated.  Commands are:

close           close current connection
logout          forcibly logout remote user and close the connection
display         display operating parameters
mode            try to enter line or character mode ('mode ?' for more)
open            connect to a site
quit            exit telnet
send            transmit special characters ('send ?' for more)
set             set operating parameters ('set ?' for more)
unset           unset operating parameters ('unset ?' for more)
status          print status information
toggle          toggle operating parameters ('toggle ?' for more)
slc             change state of special charaters ('slc ?' for more)
auth            turn on (off) authentication ('auth ?' for more)
encrypt         turn on (off) encryption ('encrypt ?' for more)
z               suspend telnet
```

Dadurch kann man die Verbindung schliessen, Parameter setzen (etwa Terminal Einstellungen)

Zu beachten ist, dass `telnet` auf den meisten Rechnern keine Möglichkeit bietet, die Daten bei der Übertragung zu verschlüsseln.

So geht etwa das Passwort beim Einlogen im Klartext über das Netz.

Das telnet unter NetBSD kann die Daten verschlüsseln, wenn das telnet auf der anderen Seite dies auch kann.

Trotzdem sollte man sich nicht darauf verlassen, dass der Datenstrom über diesen Kanal nicht direkt abgefangen werden kann.

FTP

Ein weiteres wichtiges Protokoll ist FTP (File Transfer Protocol).

Es läuft ebenfalls über TCP und hat die Port-Nummer 21 auf dem Server.

Es dient dazu, Files von einem anderen Rechner auf den eigenen (und umgekehrt) zu transferieren.

Dazu muss man sich zuerst beim anderen Rechner einlogen.

Viele Server stellen einen sog. *anonymen Account* bereit, in den sich jeder einlogen kann und Files transferieren.

Als Passwort gibt man dazu seine email-Adresse an. Als Username gibt man entweder `anonymous` oder `ftp` an.


```
> ftp ftp.cs.uni-sb.de
Connected to cs.uni-sb.de.
220- ftp.cs.uni-sb.de FTP server (Version wu-2.6.0(2) 2000-01-17 13:34) ready.

     Hi,  this machine allows anonymous ftp access
          You may use the shorthand "ftp" for logging in
220       The ftp daemon understands the reget command
331 Please give your email address as password.
230- Guest login ok, access restrictions apply.
...
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Einlogen kann man sich, wenn entweder der Server automatisch nach dem Namen fragt, oder man gibt das `user` Kommando.

```
ftp> user ftp
331 Please give your email address as password.
Password:
230- Guest login ok, access restrictions apply.
```

Natürlich kann man sich auch als anderer User anmelden, wenn man einen Account auf der Maschine hat.

Auch hier gilt: Das Passwort geht im Klartext über das Netz!

Um Files zu übertragen, etc, gibt man am `ftp>` Prompt Komamndos.

- Das `help` Kommando gibt eine Übersicht über die Kommandos.
- `cd` *`Pfad`* wechselt in das angebenene Directory auf dem Server.
- `ls` gibt den Inhalt des Directories auf dem Server aus.
- `get` *`Filename`* holt das angegebene File vom Server.
- `put` *`Filename`* stellt das File vom eigenen Rechner auf den Server.
- `bye` beendet die FTP Verbindung.

Man kann auch noch mittels des `hash` Kommandos die Fortschrittsanzeige während der Übertragung einschalten.

Wichtig ist, dass Files im richtigen Modus übertragen werden.

FTP kennt zwei Modi: ASCII und BINARY. Im ASCII Modus werden Zeichen auf 7 Bit gekürzt und auch ansonsten umgesetzt (CR+LF→LF, etc)

Man sollte also immer den BINARY Modus benutzen.

Dieser wird mit dem Kommando `bin` eingeschaltet.

Wenn man mehrere File übertragen will, kann man die Befehle `mget` und `mput` benutzen, die ein Pattern als Argument nehmen.

Will man nicht vor jedem File gefragt werden, ob es übertragen werden soll, gibt man das `prompt` Kommando.


```
ftp> bin
200 Type set to I.
ftp> prompt
Interactive mode off.
ftp> get index.Z
local: index.Z remote: index.Z
227 Entering Passive Mode (134,96,254,254,5,112)
150 Opening BINARY mode data connection for index.Z (102783 bytes).
100% |***************************************| 100 KB 886.27 KB/s 00:00 ETA
226 Transfer complete.
102783 bytes received in 00:00 (867.19 KB/s)
ftp> bye
221 Goodbye.
```

Man kann in dem File `$HOME/.netrc` Kommandos ablegen, die der FTP Client sofort beim Einloggen, etc. ausführt.

Hier kann man auch einen Default für den Login Namen und das Passwort setzen, so dass man sich nicht dauernd als ftp selbst einlogen muss:

```
default login anonymous password thesing@cs.uni-sb.de
macdef init
prompt off
hash on
```

Mit dem FTP Client kann man auch Files auf dem Server löschen und umbennenen.

Ausserdem kann man abgebrochene Transfers fortsetzen (`reget` statt `get`).

Das lokale Directory auf dem Client kann man mit `lcd` wechseln.

In einer Subshell kann man auf dem Client Programme ausführen, durch das `!` Kommando:

```
ftp> !ls -l
```

Auch noch nett: Statt ein File einfach nur zu übertragen, kann man es umbenennen oder als Eingabe an ein Kommando schicken:

```
ftp> get index.z ftpindex
ftp> get README "| less"
ftp> get index.z "|zcat| less"
```

Das NetBSD `ftp` kann auch URLs direkt von der Kommandozeile holen:

```
> ftp ftp://ftp.cs.uni-sb.de/index.Z
> ftp http://www.cs.uni-sb.de/index.html
```