H Rechnerkommunikation (TCP/IP)

H.1 Überblick

- Ende der 70er Jahre in den USA Entwicklung neuer Netzwerkprotokolle, gefördert durch ***DARPA*** *(Defense Advanced Research Project Agency)*
- daraus entstandene Protokollfamilie: **TCP/IP**
  (nach den Namen der zentralen Protokolle -
  ***T**ransmission **C**ontrol **P**rotocol* und ***I**nternet **P**rotocol*)
- ab 1983 Standardprotokoll im ARPANet
- in Berkeley Implementierung der Protokolle für UNIX (4.1c bsd)
- ab 1985/86 Einsatz von TCP/IP in den Netzen der **NSF** *(National Science Foundation)*, **NASA** und anderer Institutionen
- in den 90er Jahren Übergang zu kommerziellen Netzprovidern (ISPs)
  (z. B. USA: MCI, Sprint, AT&T, D: EUnet, XLink, MAZ)

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.1 Überblick (2)

- Implementierungen der Protokolle sind heute für alle in für Netzbetrieb relevanten Betriebssysteme verfügbar
- die TCP/IP-Protokolle sind unabhängig von der verwendeten Netzwerktechnologie - verwendbar sind z. B.
  - ◆ Ethernet, Token Ring, FDDI, ATM
  - ◆ serielle Leitungen, Modemverbindungen (PPP), ISDN
- dieTCP/IP-Protokolle sind keine Standards im Sinn internationaler Standardisierungsgremien (ISO, CCITT), aber ***de facto Standards***
- Arbeitsberichte, Protokollvorschläge und -standards werden in einer Serie von technischen Berichten veröffentlicht, den ***Requests for Comments - RFC**s*

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.1 Überblick (2)

- IP ermöglicht den Verbund von Netzen,
  das entstandene Gesamtnetz wird als **Internet** bezeichnet
- Größe: 1987: ca. 20.000 Rechner, mehrere hundert Netze
  1994: über 3 Mio. Rechner in 61 Ländern
  Ende 98: >1,6 Mio. in D, > 8 Mio. in EU, ? weltweit
  (sichtbare Rechner), exponentieller Anstieg
- seit Ende der 80er Jahre starke Ausbreitung des **Internet** in Europa
  (NORDUnet, XLINK, SURFnet, ...)
- in Deutschland zuerst regionale Cluster (Bayern und Baden Württemberg), seit Ende 1989 bundesweites Internet mit dem WIN (Wissenschaftsnetz) als Transportnetz

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.2 Einordnung in das ISO/OSI Referenzmodell

7	Anwendungsschicht	*ftp, telnet, rlogin, rsh, ...*
6	Darstellungsschicht	-
5	Sitzungsschicht	
4	Transportschicht	*TCP, UDP*
3	Netzwerkschicht	*IP*
2	Verbindungsschicht	*Ethernet*
1	physikalische Schicht	

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.3 Internet Protocol - IP

- Netzwerkprotokoll zur Bildung eines virtuellen Netzwerkes auf der Basis mehrerer physischer Netze
- definiert Format der Datengrundeinheit - IP-Datagramm
- enthält Regeln, wie Pakete verarbeitet und Fehler behandelt werden (Fragmentierung, maximale Lebenszeit, ...)
- unzuverlässige Datenübertragung
- Routing-Konzepte

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

1 Adressierung

Grundprinzipien

- einheitliche Adressierung, unabhängig von Netzwerk-Technologie (Ethernet, Token-Ring)
  - ➥ virtuelles Netz: ***Internet***
- Netzwerkeinheitlich (über Router) - transparent für Benutzer
- Aufbau einer IP-Adresse:
  - 4 Byte: **a.b.c.d**, unterteilt in Netzwerk- und Rechneradresse

a < 128:	**a** = Netzwerk, **b.c.d** = Rechner *(Class A Net)* große Organisationen
127 < **a** < 192:	**a.b** = Netzwerk, **c.d** = Rechner *(Class B Net)* Organisationen mittlerer Größe
a > 191:	**a.b.c** = Netzwerk, **d** = Rechner *(Class C Net)* kleinere Organisationen

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

1 Adressierung (2)

Probleme des Adressierungsschemas

- Mehrere physische Netze in einem Class A oder B Netz
  - ***Subnetting***: Der Rechneranteil der Adresse kann durch sog. ***Subnetmask*** in einen Unternetz- und den eigentlichen Rechneranteil unterteilt werden (Bit-Maske)
  - Nach "außen" wird nur das Class A/B-Netz bekanntgegeben, intern hat man mehrere Netze
- Nur 127 Class A- und ca. 17.000 Class B-Netze, aber über 2 Mio Class-C-Netze ➞ *ROADS*-Problem
  - Mehrere Class-C-Netze werden zusammengefaßt (***Supernetting***) — statt einem Class-B-Netz z. B. 256 Class-C-Netze
  - Beim Routing wird der Netzblock gemeinsam behandelt

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

1 Adressierung (2)

IP-Adressen an der Uni Erlangen:

- **REVUE-Netz**
  Class B Netz 131.188
- Subnetmask: 0xFFFFFF00
  (teilweise auch 0xFFFFFFC0)
- Beispiele:

Informatik Hauptnetz:	131.188.2
Inf.-CIP Subnetz:	131.188.30
Inf. 4 Subnetz:	131.188.34
faui01:	131.188.2.1, 131.188.30.1
faui04a:	131.188.30.240

LRS-Netz 1:	131.188.137.0
LRS-Netz 2:	131.188.137.64
LRS-Netz 3:	131.188.137.128
LRS-Netz 4:	131.188.137.192
hal.e-technik:	131.188.137.6, 131.188.137.65

- zusätzlich Medizin-internes Netz
  UNI_ERL_MED
  Class B Netz 141.69

➥ Rechner mit mehreren IP-Adressen unerwünscht — besser spezielle Geräte (Router) für die Verbindung zwischen

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

Motivation

- **Routing** bezeichnet die Aufgabe, einen Weg zu finden, über den IP-Pakete geschickt werden
- Prinzipiell gibt es zwei Arten des Routings:
  - **Direktes Routing:** Ein Paket wird direkt von einem Rechner zu einem anderen geschickt (**auf einem physischen Netz**, Grundlage für alle Internet-Kommunikation)
  - **Indirektes Routing:** Der Zielrechner ist **nicht auf demselben physischen Netz**, so daß das Paket über (einen) Router geschickt werden muß

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

Direktes Routing

- Der Absender verpackt das IP-Datagramm in ein Paket des physischen Netzes (Encapsulation)
- Die Ziel-IP-Adresse wird in eine Adresse des physischen Netzes abgebildet
- Das Paket wird mittels der Netzwerk-Hardware übertragen
- häufigstes Verfahren: **ARP** *(address resulution protocol)* (auf Broadcast-Basis)

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

Indirektes Routing

- Beim indirekten Routing muß der Absender das verpackte Datagramm **an einen Router schicken**, der selbst wiederum das Datagramm zum Ziel-Netzwerk weiterleiten muß
- **Router** im Internet bilden eine **kooperative, verbundene Struktur**
- **Datagramme werden von Router zu Router weitergereicht**, bis sie das Ziel-Netzwerk erreicht haben.
- Da sowohl der Absender-Rechner als auch ein Router wissen müssen, welchen Router sie für ein bestimmtes Ziel-Netzwerk benutzen müssen, brauchen sie **Routing-Informationen.**
- Routing-Informationen werden in Rouing-Tabellen gehalten

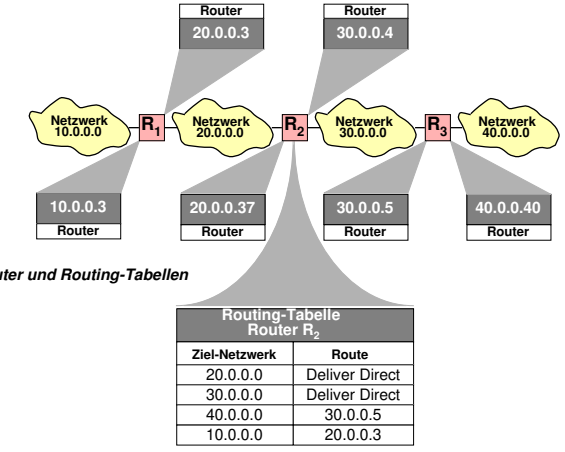
Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

Routing-Informationen

- Um die **Routing-Tabellen so kompakt** wie möglich zu halten, beziehen die Routing-Informationen sich (im allgemeinen) **nicht auf Ziel-Adressen, sondern auf Zielnetzwerke**
- IP-Software schickt das jeweilige Datagramm **entsprechend dem Netzwerk-Teil der IP-Adresse** an einen bestimmten Router
- Routing-Tabelleneinträge sind normalerweise ein **Tupel (Netw,Router)**
- Um den Unfang der Routing-Tabellen zu begrenzen, kann ein ***Default Router*** angegeben werden, der für alle nicht explizit aufgeführten Netze benutzt wird.

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Routing (5)



Router und Routing-Tabellen

Routing-Tabelle Router R_2

Ziel-Netzwerk	Route
20.0.0.0	Deliver Direct
30.0.0.0	Deliver Direct
40.0.0.0	30.0.0.5
10.0.0.0	20.0.0.3

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Routing (6)

Routing-Protokolle

- **Autonomes System** (AS): Menge von Netzen und Routern einer Administrationseinheit
  (z. B. alle Netze im deutschen Wissenschaftsnetz WiN)
- Autonome Systeme tauschen Informationen über Netze und zugehörige AS-Pfade untereinander aus (*Exterior Gateway Protocol — EGP*, speziell *Border Gateway Protocol* — **BGP**)
- Innerhalb eines AS werden Routing-Informationen mit *Interior Gateway Protocols* ausgetauscht (vor allem **RIP** und **OSPF**)

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Routing (7)

Routing-Protokolle

- **RIP** — Vektordistanz-Algorithmus, Distanz-Metrik = *hop-count*
  - jeder aktive RIP-Rechner überträgt seine Routing-Tabelle regelmäßig per Broadcast
  - jeder RIP-Rechner übernimmt Routing-Information, falls er nicht schon mindestens eine gleichwertige Route kennt
  - Routing-Information von RIP wird nach einer *timeout*-Phase ungültig, wenn sie nicht erneuert wurde
- **OSPF** — Shortest Path First-Algorithmus
  - es werden Informationen über die Netzanschlüsse jedes Knotens ausgetauscht, jeder OSPF-Rechner erstellt für sich daraus den Verbindungsgraphen des Netzes und routet danach
  - viele administrative Einflußmöglichkeiten (*load balancing*, etc.)

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.4 User Datagram Protocol - UDP

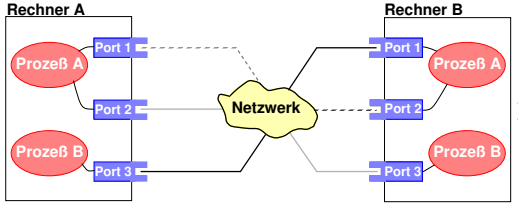
1 Motivation

- bei IP definiert eine Adresse einen Rechner
- keine Möglichkeit, einen bestimmten Benutzer oder Prozeß (Dienst) anzusprechen
- die intuitive Lösung, als Ziel einen Prozeß zu nehmen hat Nachteile
  - ➤ Prozesse werden dynamisch erzeugt und vernichtet
  - ➤ Prozesse können ersetzt werden - die *PID* ändert sich dadurch
  - ➤ Ziele sollten aufgrund ihrer Funktion (Dienst) ansprechbar sein
  - ➤ Prozesse könnten mehrere Dienste anbieten (vgl. *inetd*)

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 *Protocol-Port* Konzept

- *Protocol-Ports* sind abstrakte Kommunikations-Endpunkte
- Pakete, die einem Port zugestellt werden, werden in einer Warteschlange gehalten, bis sie abgeholt werden
- Empfangsanforderungen an einen Port werden blockiert, bis ein Paket eintrifft
- zur Kommunikation mit einem anderen Port müssen **Port-Nummer** und IP-Adresse bekannt sein



Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 … *Protocol-Port* Konzept (2)

- für eine Reihe von Standard-Diensten (route-Protokoll, rwho, ...) sind Portnummern festgelegt (UNIX → Datei **`/etc/services`**)
- UDP agiert als Multiplexer/Demultiplexer auf IP
- UDP bietet sonst keine zusätzliche Funktionalität gegenüber IP - UDP-Pakete können
  - ◆ verloren gehen
  - ◆ in falscher Reihenfolge beim Empfänger ankommen

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.5 Transmission Control Protocol - TCP

- IP stellt nur unzuverlässige Datenübertragung zur Verfügung
- TCP sichert die Übertragung gegen Fehler (Datenverlust, Paketverdopplung, falsche Reihenfolge)
- TCP versucht die zur Verfügung stehende Netzkapazität möglichst "*freundlich" und optimal* auszunutzen
- das Port-Konzept von UDP ist auch in TCP realisiert
- TCP-Kommunikation hat folgende Eigenschaften
  - ◆ ***Stream*-Orientierung** - die Daten werden als Byte-Strom entgegengenommen / abgeliefert
    - – unstrukturiert (keine Paketgrenzen!)
    - – gepuffert (IP-Pakete werden - wenn möglich - gefüllt)
  - ◆ **virtuelle Verbindung** - vor Beginn der Datenübertragung muß eine Verbindung zum Kommunikationspartner aufgebaut werden
  - ◆ voll-duplex Verbindung

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.6 Systemschnittstelle

- **Internet-Domain Sockets**

1 Adressierung

- Rechner-Adresse (32-bit, ***IP-Address***) und Portnummer
- Include-Datei: `<netinet/in.h>`,
  Struktur *sockaddr_in* :

```
struct sockaddr_in {
    short   sin_family;                 /* =AF_INET */
    short   sin_port;                   /* Port */
    struct in_addr sin_addr;            /* IP-Addr */
    char    sin_zero[8];
};
```

sin_port: Portnummer (1 bis 1023 reserviert!)
ab 1024 frei wählbar
0 = Port wird vom System gewählt

sin_addr: IP-Adresse, mit ***gethostbyname(3)*** zu finden

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Rechnernamen / IP-Adressen

- IP-Adressen sind für den Benutzer unhandlich
  → Vergabe von Rechnernamen
- Form der Rechnernamen bei größerer Rechnerzahl wichtig
  - Abbildung Rechnername ↔ Adresse muß möglich sein
  - es dürfen keine Namenskonflikte auftreten
- flacher Namensraum
  - Namen sind Sequenz von Zeichen, ohne Strukturierung
  - zentrale Vergabe notwendig
  - bei großem Namensraum unmöglich zu verwalten
- hierarchischer Namensraum
  - vergleichbar mit UNIX-Dateinamen
  - Namensvergabe dezentral möglich
  - Abbildung Name ↔ Adresse ebenfalls dezentral möglich

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

3 weitere Verwaltungsdaten

★ analog zu den Rechnerdaten gibt es Zuordnungsmechnismen für:

- Netzwerkadressen
  Datei: */etc/networks*
  Funktionen: ***getnetent(3), getnetbyname(3), getnetbyaddr(3)***
- Protokollnamen (TCP, UDP, ...)
  Datei: */etc/protocols*
  Funktionen: ***getprotoent(3), getprotobyname(3), getprotobynumber(3)***
- Service ⇔ Portnummer
  Datei: */etc/services*
  Funktionen: ***getservent(3), getservbyname(3), getservbyport(3)***

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

2 Rechnernamen / IP-Adressen (2)

- dezentrale Abbildung über ***Nameserver***
  - Beispiel: `faui04.informatik.uni-erlangen.de`
- Abbildung Rechnernamen ↔ IP-Adressen (Schnittstelle)
- Funktionen: ***gethostent(3)***
  gethostbyname(3)
  gethostbyaddr(3)
- Include-Datei: `<netdb.h>`
  Struktur *hostent* :

```
struct hostent {
    char *h_name;               /* official name of host
*/
    char  **h_aliases;          /* alias list */
    int   h_addrtype;           /* host address type
 (AF_INET) */
    int   h_length;             /* length of address */
    char  **h_addr_list         /* list of addresses from
                                   name server */
```


Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

H.7 spezielle Funktionalität

- Broadcast-Pakete
  - nur bei Datagramm-Sockets möglich
  - kann von allen Rechnern auf dem Netz empfangen werden
  - netzwerkunabhängige Programmierung kompliziert, da zuerst die angeschlossenen Netzwerke erfragt werden müssen
- Out of band data
  - dringende Daten, die bevorzugt empfangen werden sollen
  - nur bei Stream-Sockets
  - senden mit ***send(2)***, empfangen mit ***recv(2)***
  - bewirkt beim Empfänger das Signal **SIGURG**

Reproduktion jeder Art oder Verwendung dieser Unterlage, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.