IP- UND TCP - GRUNDLAGEN

Inhalt

1. DAS IP – INTERNET PROTOCOL

Das Internet-Protokoll definiert die Grundlage der Datenkommunikation auf der untersten Ebene. Es ermöglicht unabhängig von den verwendeten physikalischen Medien das Zusammenfügen vieler unterschiedlicher Netzwerk- und Hardwarearchitekturen zu einem einheitlichen Netz.

Das Internet-Protokoll gewährleistet die Übertragung der Daten durch einen verbindungslosen, nicht abgesicherten Transport. Für Sicherheitsmechanismen sind übergeordnet Protokolle wie TCP verantwortlich.

Grundlagen für netzübergreifende Verständigung:

- Adressierungsmechanismus, um Sender und Empfänger eindeutig zu benennen
- Konzept zum Transport der Datenpakete über Knotenpunkte (Routing)
- Format für Datenaustausch (definierter Header mit wichtigen Informationen)

1.1. Die Protokollschichten des Internet

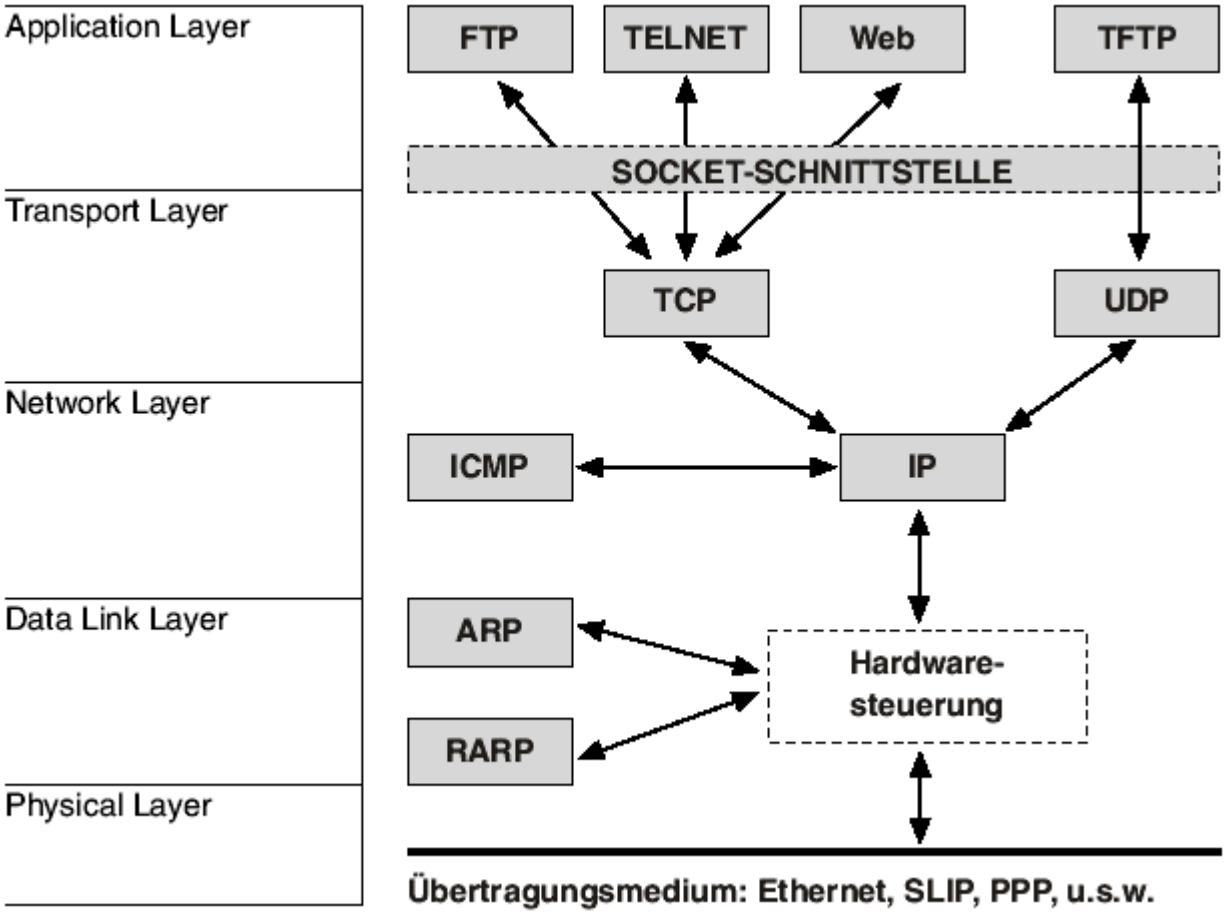


Bild: Die Protokollschichten des Internets

1.2. Internetadressen

Jeder Host im Internet hat eine weltweit einmalige Adresse. Diese IP-Adresse ist ein 32-Bit-Wert, der üblicherweise zur besseren Lesbarkeit in Dot-Notation - also byteweise durch Punkte getrennt - angegeben wird.

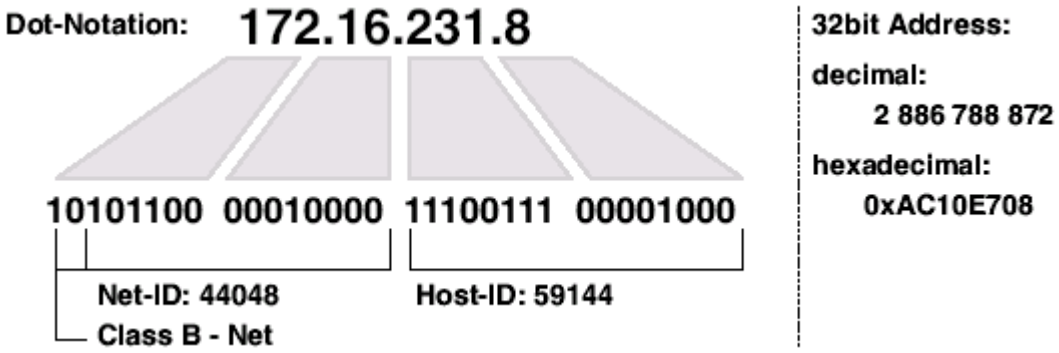


Bild: Format der IP-Adresse

Die IP-Adresse ist in die Netzwerk- und die Host-ID unterteilt. Wieviel Bits jeweils für Netzwerk- und Host-ID verwendet werden, hängt von der Klasse des IP- Netzwerks ab.

Diese Netzwerkklasse kann - wie in der unter en Tabelle gezeigt - an den höchsten Bits der Adresse abgelesen werden:

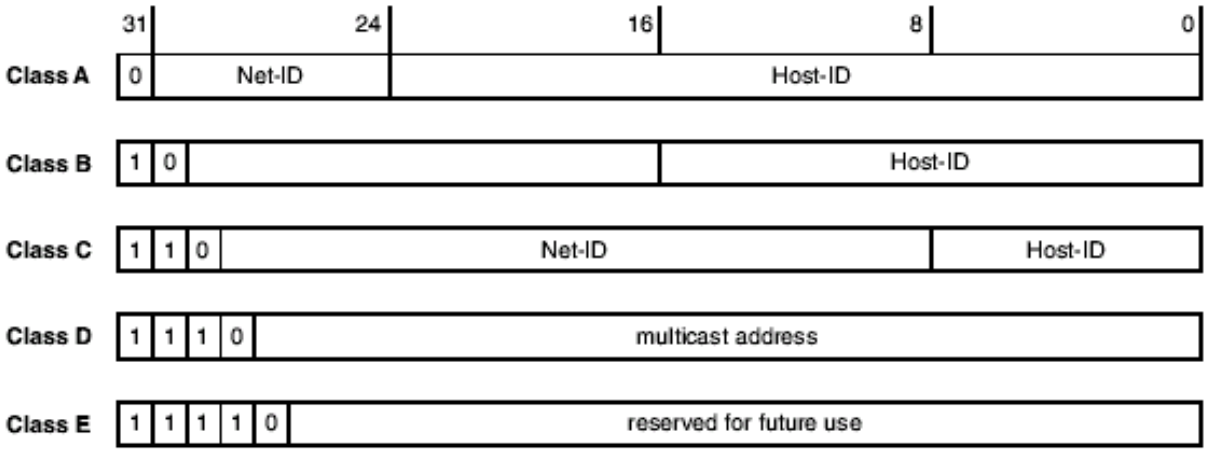


Bild: Netzwerkklassen

Aus der Definition der Netzwerkklassen ergeben sich damit die folgenden Adressräume:

Class	Lowest Net-ID	Highest Net-ID
A	0.1.0.0	126.0.0.0
B	128.0.0.0	191.255.0.0
C	192.0.1.0	223.255.255.0
D	224.0.0.0	239.255.255.255
E	240.0.0.0	247.255.255.255

Bild: Adressräume der jeweiligen Netzwerkklassen

Üblicherweise werden jedoch nur IP-Adressen der Klassen A bis C vergeben. Mit den Klassen D und E kommt man nicht in Berührung: Die Klasse D umfasst Netze für Multicasting und die Klasse E ist für Forschungszwecke reserviert.

Die folgenden Internetadressen haben eine besondere Bedeutung und dürfen nicht als Adresse an einen Internet-Host vergeben werden:

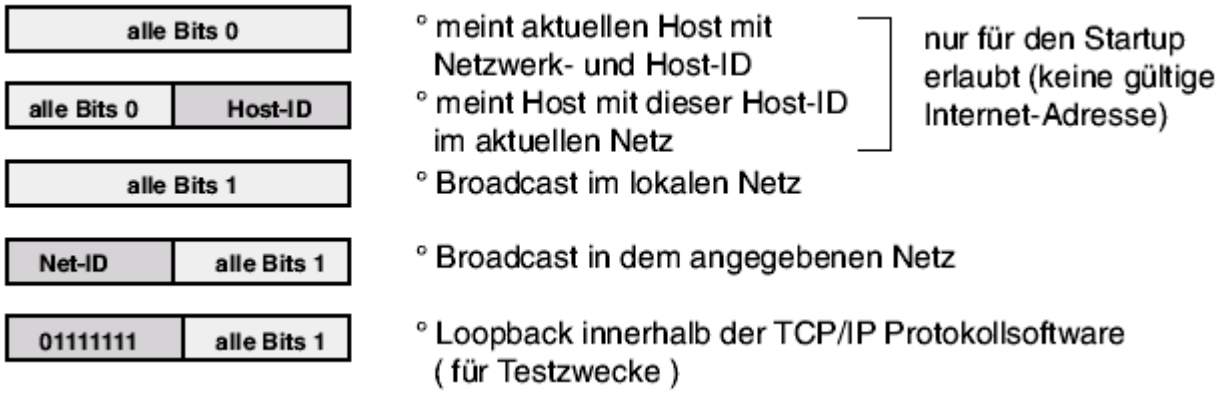


Bild: Besondere IP-Adressen

1.3. Das Paketformat von IP

Ein Datagramm setzt sich aus einem Paketkopf (Header) und dem Datenbereich (Data Area) zusammen. Der Header enthält Informationen über das Datagramm.

Im Datagramm sind u. a. enthalten:

- die Adressen von Sender und Empfänger,
- Routing-Informationen,
- die Nummer des übergeordneten Protokolls zur Weiterleitung des Datagramms
- sowie spezielle Optionen.

datagram header	datagram data area

Format IP-Datagram-Header:

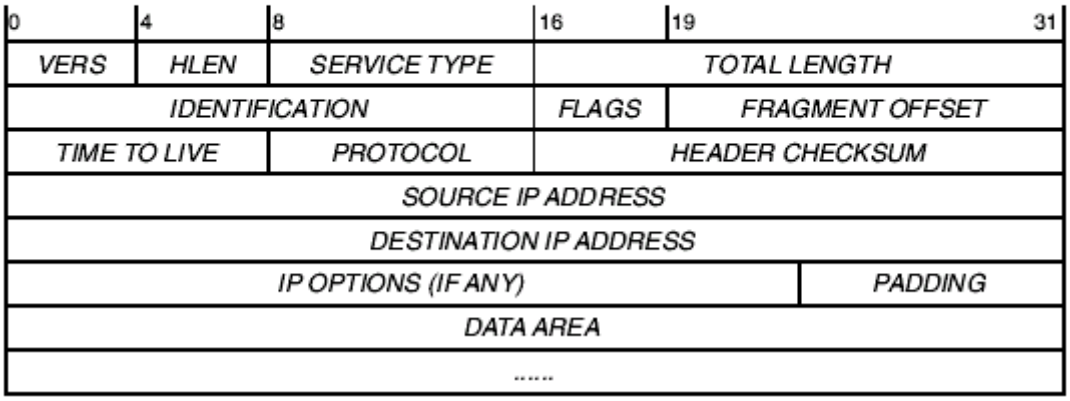


Bild: IP-Paketformat

version:	binärkodierte Version des IP-Protokolls (aktuell 4.0)
hlen:	Länge des IP-Headers in DWORDs (32 Bit)
service type:	Priorität eines Pakets und Merkmale des gewünschten Übertragungswegs
total length:	Gesamtlänge des IP-Pakets aus Kopf und Nutzdaten in Bytes (8 Bit)
identification:	Vom Sender gesetzter Wert zur Identifizierung der einzelnen Fragmente
flags (3bit):	Bit 2: Fragmentierung erlaubt 0=ja, 1=nein; Bit 3: 0=letztes Fragment, 1=weitere Fragmente folgen
time to live:	Zähler, der bei jedem Router dekrementiert wird. Wird der Wert 0 erreicht, wird das Paket verworfen.
protocol:	Nr. des übergeordneten Protokolls (z.B. TCP=6, UDP=17, ...)
header checksum:	Checksumme über den Header
source IP address:	IP-Adresse des Absenders
destina tion IP addr.:	IP-Adresse des Empfängers
IP options (variab.):	IP-Optionen, sofern benötigt
padding:	Füllbytes, um die Headerlänge auf ein Vielfaches von DWORDs zu bringen

2. ROUTING VON IP-PAKETEN

Routing ist der Transport eines Datagramms vom Sender zum Empfänger. Wir unterscheiden zwischen direktem und indirektem Routing. Das direkte Routing erfolgt innerhalb eines lokalen Netzes, wobei kein Router benötigt wird. Indirektes Routing erfolgt zwischen zwei Stationen in unterschiedlichen Netzen, wobei der Sender das IP- Paket dem nächsten Router übergibt.

Ob das Paket direkt oder indirekt geroutet werden muss, ist leicht zu entscheiden: Die Software vergleicht die Net-ID des Empfängers mit der aktuellen Net-ID; sind sie nicht identisch, wird das Paket dem Router übergeben.

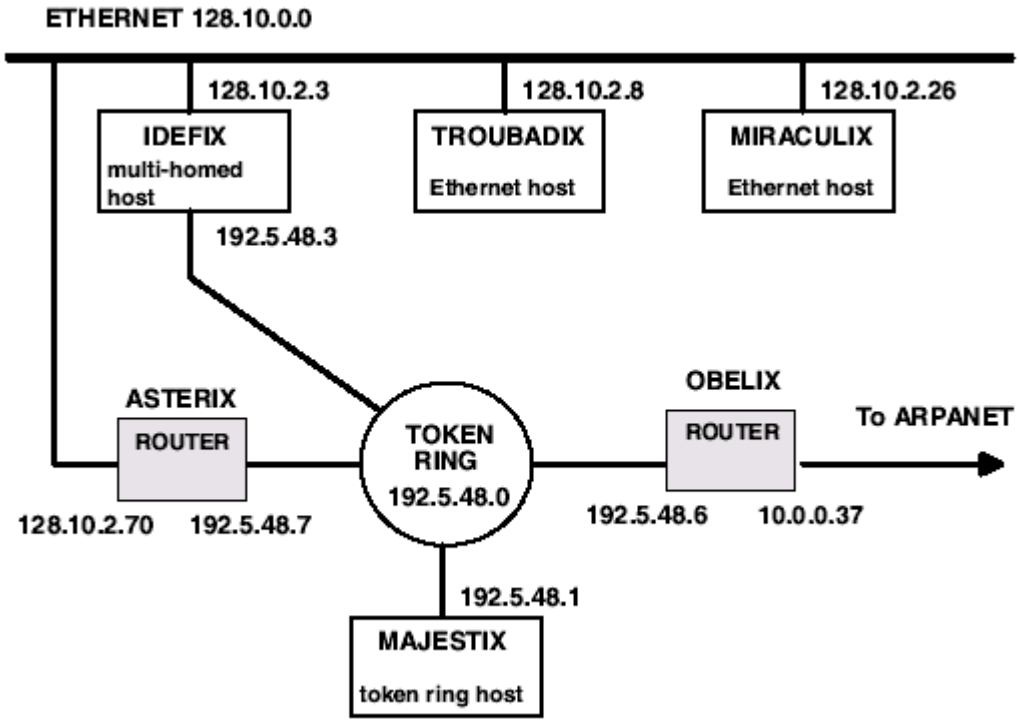


Bild: Beispiel eines heterogenen Netzwerks

Die obenstehende Abbildung zeigt das Beispiel eines Netzwerks mit Hosts und Routern.

Der Host IDEFIX ist ein „multi-homed Host“, d. h. er hat Zugang zu mehreren Netzwerken (z. B. über zwei Ethernetkarten), verfügt aber über keine Routersoftware.

Die Hosts IDEFIX, TROUBADIX und MIRACULIX gehören einem Class-B- Netz (128.10.0.0) an.

Das Token-Ring-Netz ist ein Class-C-Netz (192.5.48.0), welches durch den Router OBELIX mit dem Arpanet (Class-A-Netz 10.0.0.0) verbunden ist.

3. SUBNETS

Wenn ein lokales Netz nicht ausreicht oder wegen seiner Größe (z. B. bei Class-A-Netzen mit über 16 Millionen Hosts) zu unhandlich ist, wird es in weitere Teilnetzwerke – sogenannte Subnets – untergliedert. Unterschiedliche Netztechnologien in den einzelnen Abteilungen, Beschränkungen hinsichtlich der Kabellänge und der Anzahl der ange-

schlossenen Stationen sowie Performance-Optimierung sind weitere Gründe, Netzwerke in kleine Segmente aufzuteilen.

Da die Struktur der IP-Adresse keine Möglichkeit mehr bot, diese zusätzliche Kodierung in der Adresse selbst unterzubringen, wurde die Subnet-Mask eingeführt. Sie definiert, welche Bits der Host-ID dazu verwendet werden, die Subnet-ID zu kodieren und welche die ID des Hosts bezeichnen.

Die Subnet Mask wird vom Administrator festgelegt und wie die IP-Adresse in Dot-Notation (z.B. 255.255.255.128) dargestellt.

Werden Subnets gebildet, muss der beschriebene Routing-Algorithmus erweitert werden, da die Net-ID des Empfängers mit der des aktuellen Hosts identisch sein kann, obwohl sich beide in unterschiedlichen lokalen Netzen befinden. Deshalb müssen zusätzlich auch die Subnet-IDs von Empfänger und aktuellem Host verglichen werden. Sind diese ebenfalls identisch, kann direkt geroutet werden.

Binäroperationen mit der Subnet Mask:

Host-ID = IP-Address AND (NOT (Subnet-Mask))

Net-IDS = IP-Address AND Subnet-Mask (Net-IDS: Kombination aus Net- und Subnet-ID)

Subnet-ID: setzen Sie in der Net-IDS die Net-ID gleich 0

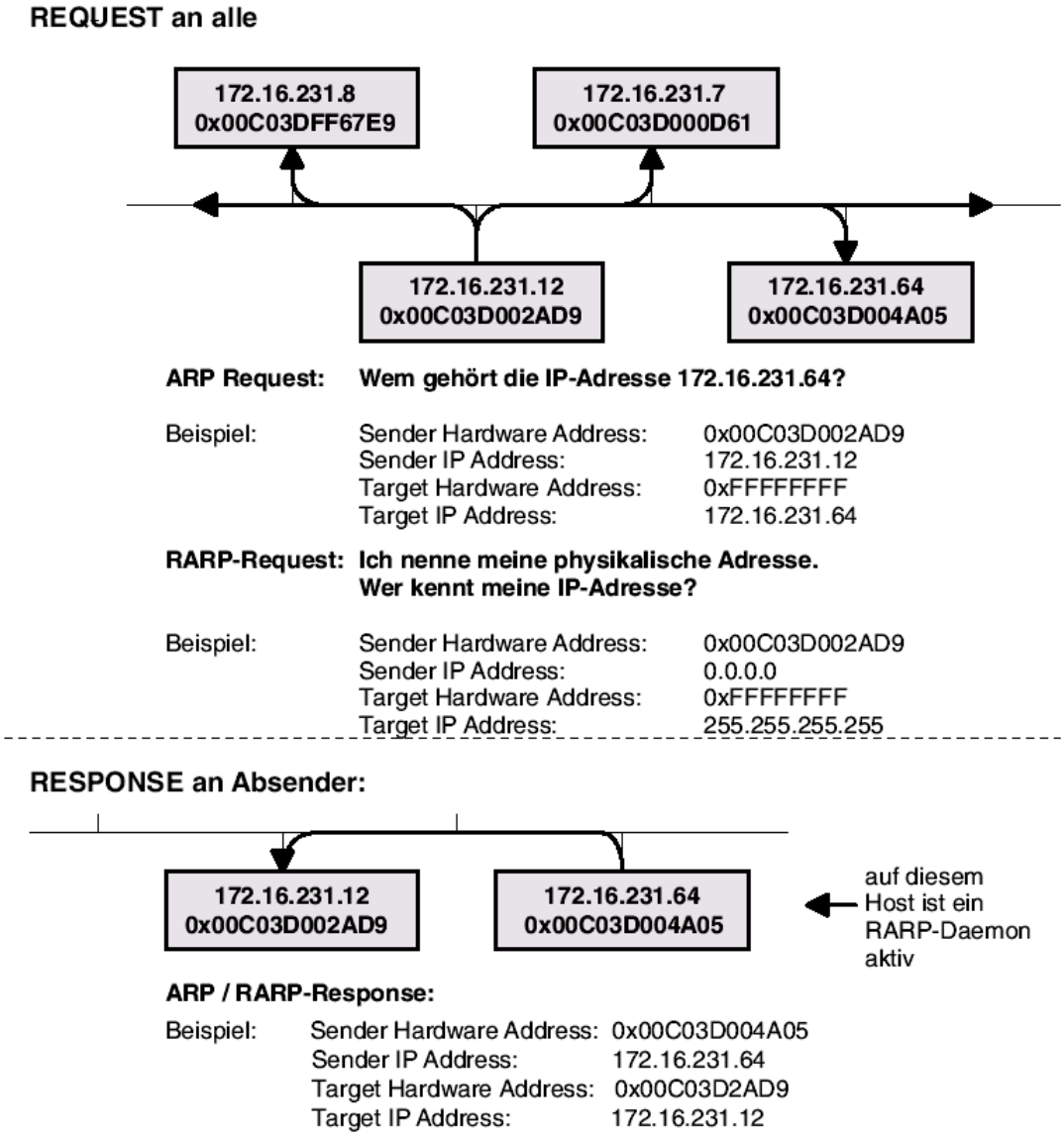
IP-Adresse:	172.16.233.200	10101100	00010000	11101001	11001000
Subnet-Mask:	255.255.255.128	11111111	11111111	11111111	10000000
Host-ID:	72	00000000	00000000	00000000	01001000
Net-ID:	172.16.0.0	10101100	00010000	00000000	00000000
Net-IDS:	172.16.233.128	10101100	00010000	11101001	10000000
Subnet-ID:	0.0.233.128	00000000	00000000	11111111	10000000

Bild: Beispiel für eine IP-Adresse aus einem Class B-Netz

4. ARP UND RARP

ARP und RARP (letzteres findet sich nur unter UNIX) liefern Mechanismen, mit denen sich IP-Adressen auf die physikalischen Netzadressen abbilden lassen, die man beim direkten Routing benötigt. Jedes hardwarenahe Protokoll (Ethernet, X.25, ISDN ...) hat sein eigenes Adressformat und versteht keine IP-Adressen.

Befindet sich der Empfänger nicht im lokalen Netz, benötigt man die physikalische Adresse des Routers, der das Paket in ein anderes Netz weiterreicht.



Die Zuordnung von Ethernet-Adr esse und IP-Adresse wird in einer Tabelle gespeichert und erst nach einem Timeout gelöscht.

Achtung: *Bei Änderung dieser Zuordnung (z. B. Zuweisen der gleichen IP-Adresse an ein Austauschgerät) werden Sie u. U. keine Verbindung zum Empfänger erhalten. Steht der Befehl* „arp“ *nicht zur Verfügung, schafft hier nur Warten, ein Neustart des Rechners oder die Vergabe einer neuen IP-Adresse Abhilfe.*

5. TRANSPORT LAYER

5.1. Adressierung der Applikationen mit Portnummern

Die IP-Adresse adressiert einzig und allein den Host. Auf jedem Host können jedoch gleichzeitig mehrere Applikationen aktiv sein, z.B. ein Web-Browser, ein Telnet Client und andere. Die notwendigen Mechanismen zur Adressierung der Applikationen bieten die Protokolle TCP und UDP.

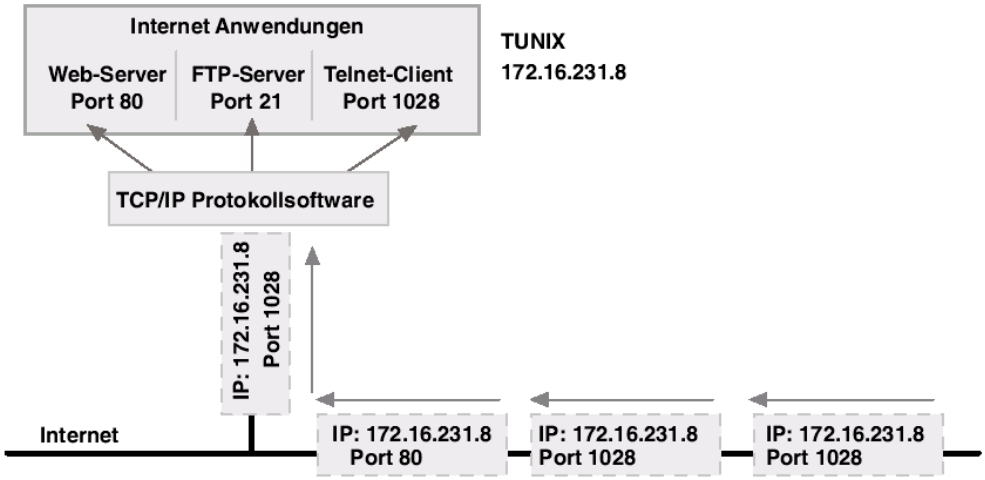


Bild: Mechanismen zur Adressierung der Applikationen eines Hosts

Für bekannte Anwendungen werden feste Ports vergeben, auf die sich jede Anwendung beim Verbindungsaufbau beziehen kann. Der Bereich von 0 bis 1023 enthält deshalb reservierte Portnummern. Diese dürfen unter keinen Umständen für eigene Anwendungen verwendet werden. Die komplette Liste der „normierten Dienste“ finden Sie im RFC 1700 (1994).

ftp://ftp.isi.edu/in-notes/iana/assignments/port-numbers // Ports

http://sunsite.auc.dk/RFC // RFCs

Anwendung	Port	Protokoll	Beschreibung
ftp	21	udp / tcp	File-Transfer
telnet	23	udp / tcp	Telnet
smtp	25	udp / tcp	Simple Mail Transfer
domain	53	udp / tcp	Domain Name Server
tftp	69	udp / tcp	Trivial File-Transfer
www-http	80	udp / tcp	World Wide Web HTTP
sftp	115	udp / tcp	Simple File Transfer Protocol
snmp	161	udp / tcp	SNMP
. . .			

Bild: Aufstellung von Applikationen und deren Port-Nummern:

5.2. Das Format von UDP

Die Leistungsmerkmale von UDP beschränken sich auf die Trennung der Kommunikationskanäle der Applikationen. Die Zustellung der Datagramme im Netz ist nicht abgesichert.

Das Protokoll bietet keine Gewähr für die Einhaltung der Reihenfolge.

Eine Internetanwendung, die auf UDP aufsetzt, muss selbst für die Absicherung der Datenübertragung sorgen. UDP ist deshalb für Applikationen mit eigenen Sicherungs-

mechanismen geeignet. Es spart „Protokoll-Overhead“ und bietet dadurch höhere Übertragungsraten als TCP. Zusätzlich entfallen alle Mechanismen für Verbindungsaufbau und -abbau.

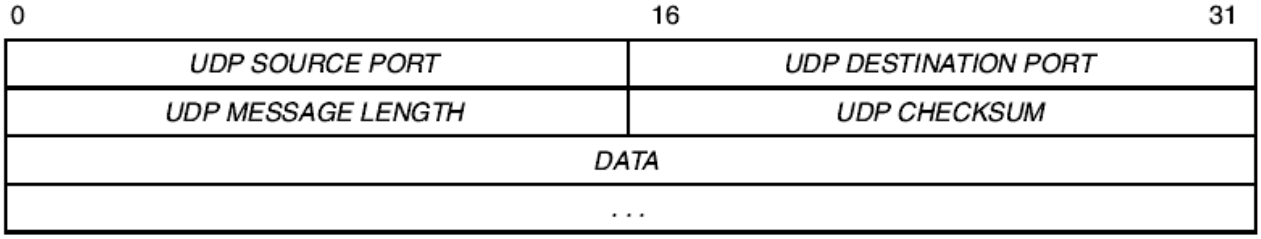


Bild: Format eines UDP-Datagram Headers

Source Port: Port des Absenders; wird für das Rücksenden von Daten durch den Empfänger benötigt.

Destination Port: Ziel-Port, an den das Datenpaket beim Empfänger übertragen werden soll.

Länge: Größe des UDP-Datagramms in Byte (Header und Daten).

Checksumme: Checksumme über das UDP-Datagramm, jedoch optional (wird sie nicht genutzt, erscheint in diesem Feld eine 0).

5.3. TCP – Transport Control Protocol

TCP befreit die Internetanwendung von Sicherungsmechanismen und realisiert im Gegensatz zu UDP einen sicheren Kommunikationskanal. Deshalb basieren nahezu alle wichtigen Internetanwendungen (HTTP, E-Mail usw.) auf TCP.

Die Endpunkte einer TCP-Verbindung bilden zwei Tupel aus IP-Adresse und Portnummer. Es wird eine virtuelle Verbindung zwischen den beiden Endpunkten aufgebaut.

Die Kommunikation ist *Vollduplex,* d. h. beide Kommunikationspartner können gleichzeitig Daten senden und empfangen.

Das Protokoll ist für die Applikation transparent - Daten, die an die TCP- Schnittstelle übergeben werden, kommen beim Empfänger auch so an.

Die Paketgrößen sind frei wählbar. Sofern es keine Hardwareeinschränkungen gibt, ist hier von einem Byte bis hin zu mehreren Megabyte alles erlaubt.

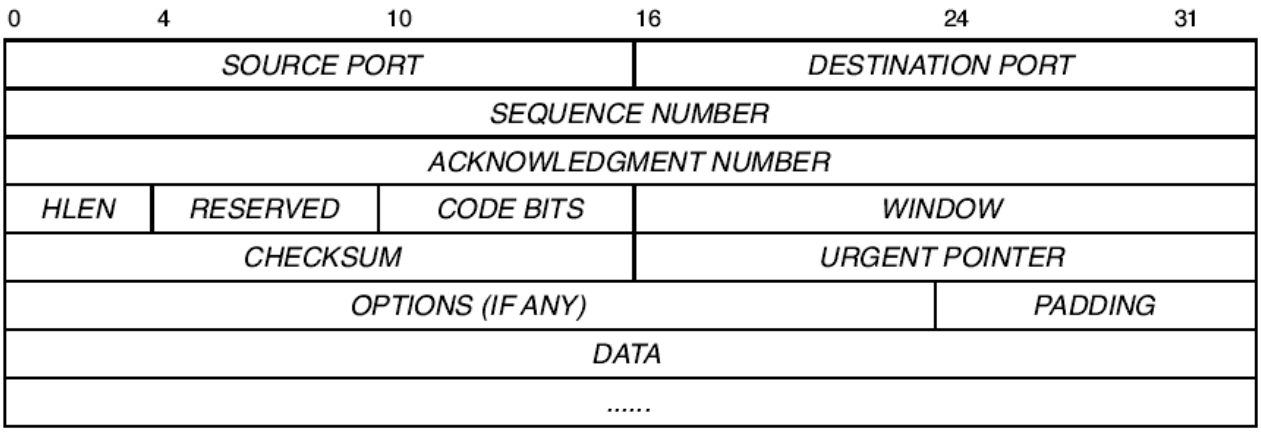


Bild: Das Format eines TCP-Packets

Source Port: Portnummer der Applikation des Absenders

Destination Port: Portnummer der Applikation des Empfängers

Sequence No: Offset des ersten Datenbytes relativ zum Anfang des TCP-Stroms (garantiert die Einhaltung der Reihenfolge)

Acknowl. No: im nächsten TCP-Paket erwartete Sequence-No. (ACK für die erhaltenen Bytes)

HLEN: Größe des TCP-Headers in DWORDs (32 Bit), Beginn des Datenbereichs

Code Bits: bezeichnen Aufgabe und Inhalt des Pakets:

Bit 1:	URG	das Segment enthält dringende Daten, Feld Urgent Pointer beachten SOCKET-INTERFACE: Out Of Band Data
2:	ACK	das Segment enthält ein Acknowledge
3:	PSH	empfangene Daten sofort weitergeben (push)
4:	RST	Verbindung zurücksetzen (reset)
5:	SYN	Verbindungsaufbau und Synchronisation der Sequence Numbers
6:	FIN	keine weiteren Daten vom Sender, Verbindung schließen

Bild: Bedeutung der Code Bits

Window: Anzahl der Bytes (ausgehend von dem im Feld Acknowledgement-No. angezeigten Byte), für deren Entgegennahme der Empfänger bereit ist

Checksum: Checksumme über das TCP-Datagramm und einen Pseudoheader

Urgent Pointer: wenn das URG-Flag gesetzt ist, gibt dieser Wert den Offset des ersten Bytes hinter den dringenden Daten am Anfang der Segment-daten an

Options: Optionen (die wichtigste Option: maximale Segmentgröße)

5.3.1. Auf- und Abbau einer TCP-Verbindung

TCP verfügt über feste Mechanismen zur Einrichtung einer Verbindung zwisc hen Client und Server. Der Aufbau einer Verbindung dient unter anderem dazu, beide Teilnehmer auf den zu übertragenden Datenstrom zu synchronisieren und Übertragungsparameter wie Paketlänge und Empfangsspeichergröße auszutauschen.

Aufbau:

Der Client sendet ein Paket, in dem in den *Code-Bits* das Flag SYN gesetzt ist. Der Server bestätigt den Empfang mit dem Flag ACK und setzt seinerseits das Flag SYN.

Beide synchronisieren sich auf die *Sequence-No.* der Gegenstation.

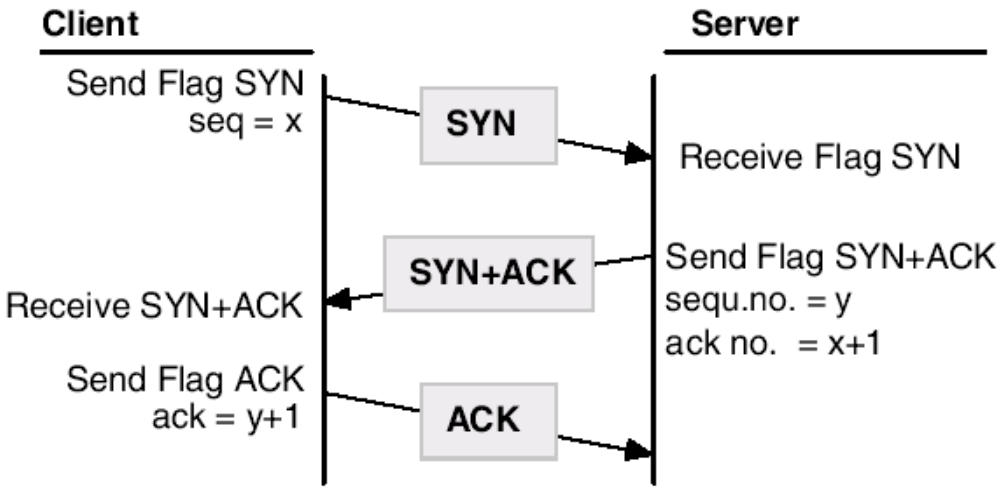


Bild: Ablauf zum Aufbau einer TCP-Verbindung

In der Option *Maximum Segment Size* kann jede Seite festlegen, wie viel Byte sie maximal in einem Segment nach dem TCP-Header empfangen kann.

Abbau:

Das Schließen einer Verbindung kann vom Client oder vom Server veranlasst werden. Hierzu wird in den *Code-Bits* das Flag FIN gesetzt. Erst wenn beide Seiten dieses Flag gesetzt haben, gilt die Verbindung als geschlossen.

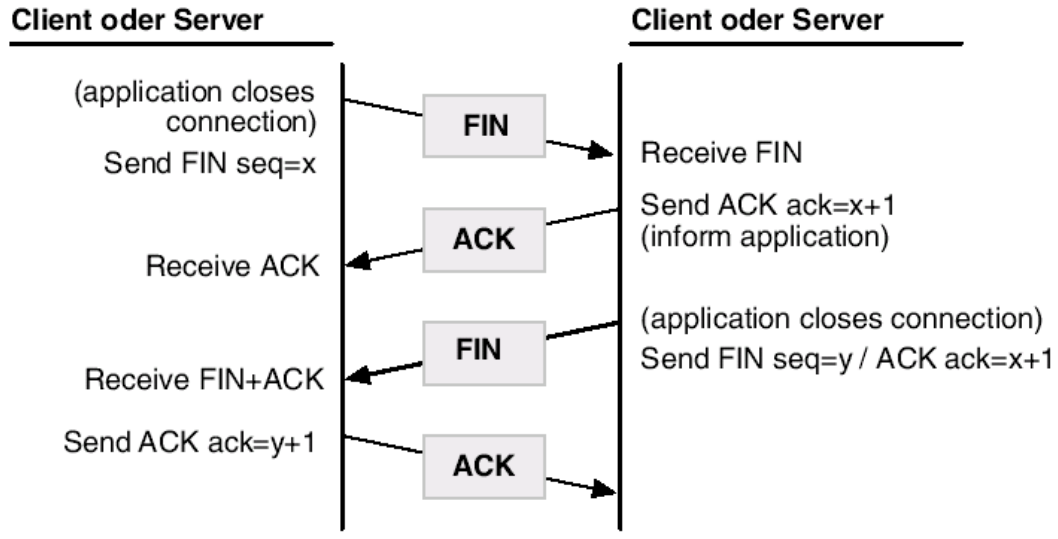


Bild: Ablauf zum Abbau einer TCP-Verbindung

Flusssteuerung

TCP verfügt über mehrere Mechanismen, zur sicheren und effizienten Übertragung von Daten. Hier ein paar der wichtigsten Regeln:

- Der Sender muss alle Daten solange bereithalten, bis sie vom Empfänger bestätigt wurden.
- Bei fehlerhaften Paketen (z.B. Paketen mit fehlerhafter Checksumme) gibt der Empfänger die letzte Acknowledgement-Number zurück, woraufhin der Sender das Paket wiederholt.
- Gehen Pakete verlor en, wiederholt der Sender nach Ablauf en eines Timeouts alle Pakete nach dem zuletzt empfangenen Acknowledgement.
- Mit dem Feld *Windows* teilt der Empfänger in jedem Paket mit, wie viel Empfangsspeicher er noch hat. Enthält das Feld den Wer t 0, stellt der Sender die Übermittlung ein, bis er vom Empfänger ein Paket mit dem Wert *Windows* ungleich Null erhält.
- Da der Sender ständig über die aktuelle Buffergröße des Empfängers informiert ist, muss er nicht auf die Bestätigung jedes einzelnen Paketes warten, sondern kann so viele Daten senden, bis der Buffer des Empfängers gefüllt ist. Der Empfänger bestätigt dabei immer einen Teil des Empfangs-Bytestroms und nicht die einzelnen Pakete. Dadurch können Teile von Paketen oder auch mehrer e Pakete auf einmal bestätigt werden. Diese Methode nennt man *Windowing*.

6. LITERATURVERZEICHNIS

- **INERNET intern**

Tischer und Jennrich

Ver lag: DATA Becker

ISBN 3-8158-1160-0

- **Inside Visual C++**

David J. Kruglinski

Ver lag: Microsoft Press

ISBN 3-86063-394-5

- **Internetworking with TCP/IP**

Verlag: PRENTICE HALL

Volume I: Principles, Protocols and Architecture

Douglas E. Comer

ISBN: 0-13-216987-8

Volume II: Design, Implementation and Inter nals

Douglas E. Comer, David L. Stevens

ISBN: 0-13-125527-4

Volume III: Client-Server Programming and Applications

Douglas E. Comer, David L. Stevens

ISBN: 0-13-260969-X