Netzwerktechnologien
3 VO

Univ.-Prof. Dr. Helmut Hlavacs
helmut.hlavacs@univie.ac.at

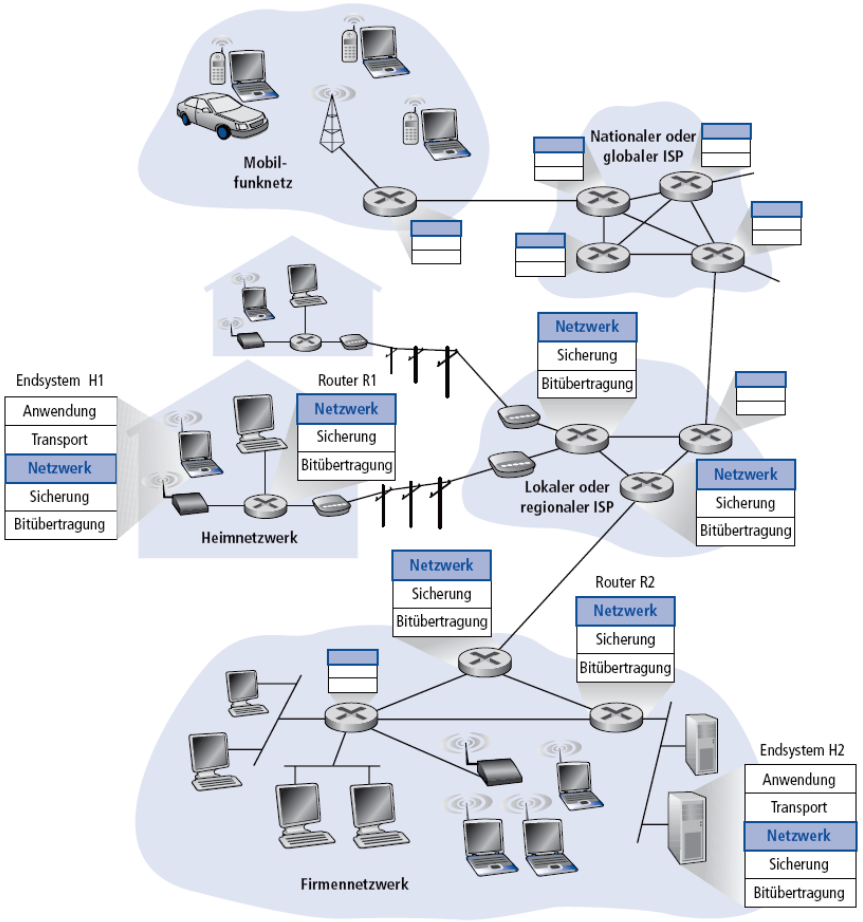
Dr. Ivan Gojmerac
gojmerac@ftw.at

Bachelorstudium Medieninformatik
SS 2012

Kapitel 4 – Netzwerkschicht

4.1 Die Netzwerkschicht

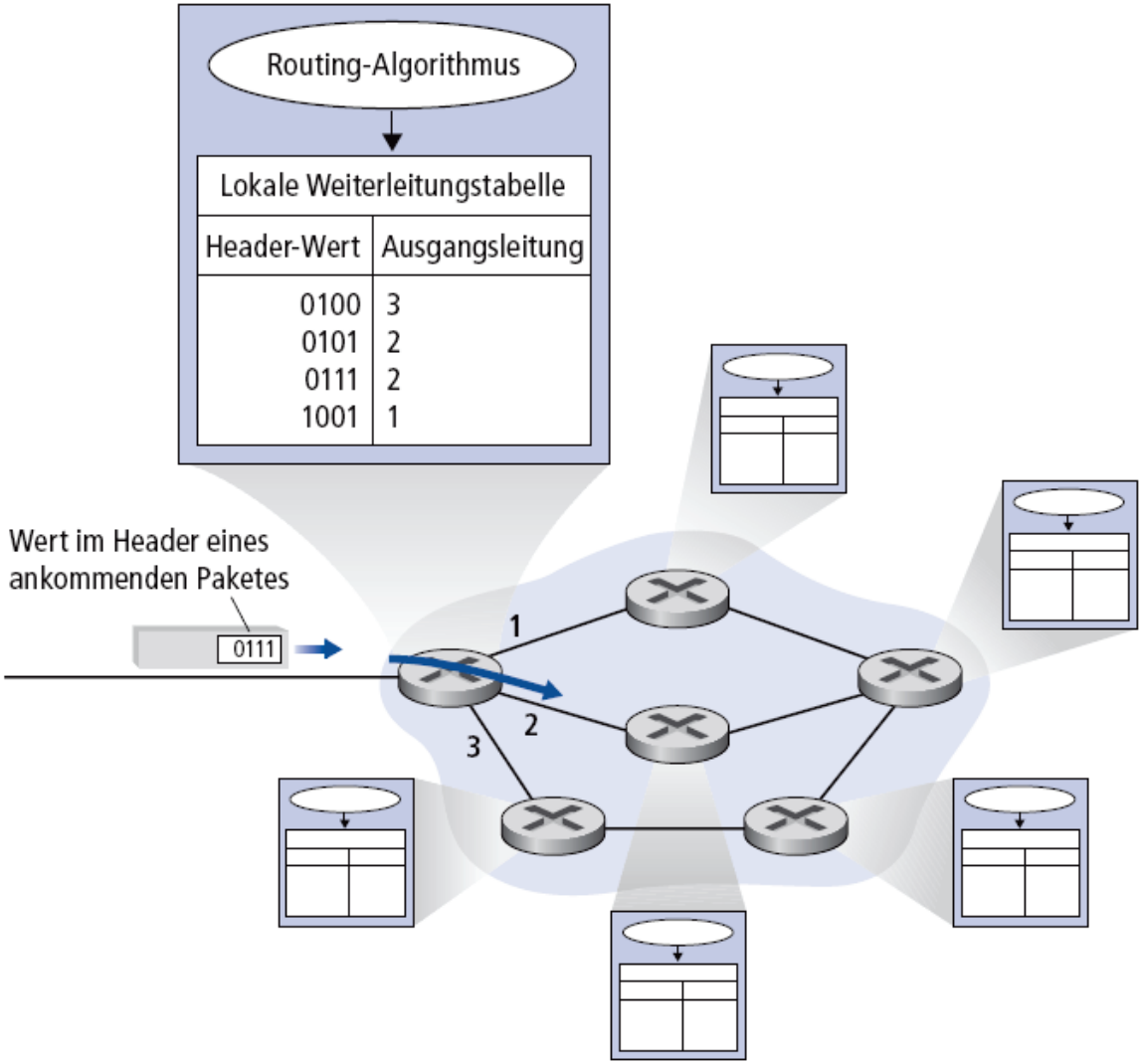
- Auch: **Vermittlungsschicht** oder **Network Layer**
- Daten von der nächsthöheren Schicht (Transportschicht) des Senders entgegennehmen
- In Datagramme verpacken
- Durch das Netzwerk leiten
- Auspacken des Vermittlungspakets beim Empfänger
- Ausliefern der Daten an die nächsthöhere Schicht (Transportschicht) des Empfängers
- Netzwerkschicht existiert in jedem Host und Router!



4.1.1 Funktionen der Netzwerkschicht

- Weiterleiten von Paketen (Forwarding):
  - Router nimmt Paket auf einer Eingangsleitung entgegen
  - Router bestimmt die Ausgangsleitung anhand lokaler Informationen (z.B. Routing-Tabelle)
  - Router legt das Paket auf die Ausgangsleitung
- Wegewahl (Routing):
  - Router kommunizieren miteinander, um geeignete Wege durch das Netzwerk zu bestimmen
  - Als Ergebnis erhalten sie Informationen, wie welches System im Netzwerk zu erreichen ist (z.B. wird eine Routingtabelle mit Einträgen gefüllt)
- Metapher:
  - Routing = Planen einer Strecke für eine Autofahrt
  - Weiterleitung = Verhalten an einer Autobahnkreuzung

4.1.1 Zusammenspiel von Routing und Forwarding



4.1.1 Verbindungsauf- und Verbindungsabbau

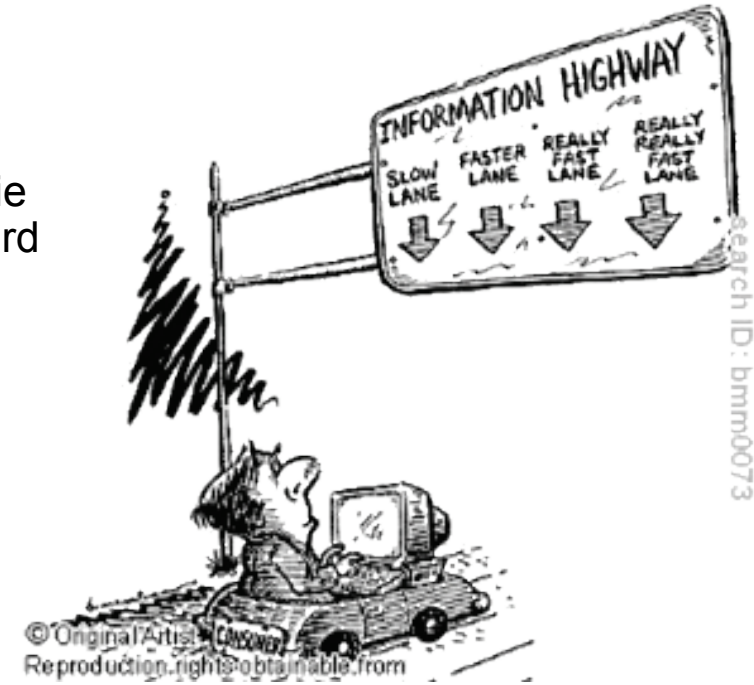
- Dritte wichtige Funktion in einigen Netzwerkarchitekturen:
  - ATM (Asynchronous Transfer Mode), Frame Relay, X.25
- Bevor Daten übertragen werden können, wird eine virtuelle Verbindung vom Sender zum Empfänger aufgebaut
  - Nach der Übertragung wird diese Verbindung wieder abgebaut
- Unterschied Verbindungen auf der Transportschicht und der Netzwerkschicht:
  - *Netzwerkschicht*: zwischen zwei Hosts (dazwischenliegende Router können involviert sein)
  - *Transportschicht*: zwischen zwei Prozessen

4.1.2 Dienstmodelle der Netzwerkschicht

Unterschiedliche Anforderungen verlangen unterschiedliche Dienstmodelle:

- Beispiele für einzelne Datagramme:
  - Garantierte Zustellung
  - Garantierte Zustellung in weniger als 40 ms

- Beispiele für einen Fluss von Datagrammen:
  - Reihenfolgeerhaltende Auslieferung
  - Garantierte minimale Datenrate
  - Beschränkung der Schwankungen in der Zeit, die für den Transport von Datagrammen benötigt wird



© Original Artist
Reproduction rights obtainable from
search ID: bmm0073

4.1.2 Dienstmodelle der Netzwerkschicht

Netzwerk-architektur	Dienst-modell	Band-breiten-garantien	Garantie der Ver-lustfreiheit	Reihen-folge	Zeit-garantien	Hinweis auf Überlast
Internet	Best Effort	keine	nein	beliebige möglich	nicht unterstützt	keiner
ATM	CBR	garantiert eine kon-stante Rate	ja	in korrekter Reihenfolge	unterstützt	Überlast tritt nicht auf
ATM	ABR	garantier-tes Mini-mum	nein	in korrekter Reihenfolge	nicht unterstützt	Überlasthin-weise werden verwendet

4.2 Virtuelle Leitungen und Datagrammnetzwerke

Verbindungsorientierter und verbindungsloser Netzwerkschichtdienst

- Ein Datagrammnetzwerk verwendet eine verbindungslose Netzwerkschicht
- Ein Netzwerk mit virtuellen Leitungen verwendet eine verbindungsorientierte Netzwerkschicht
- Analog zur Transportschicht. Aber:
  - ! Dienst: Host-zu-Host (nicht Prozess-zu-Prozess)
  - ! Keine Wahlmöglichkeit: Ein Netzwerk bietet das eine oder das andere an
  - ! Implementierung: im Inneren des Netzwerkes

→ Achtung! In höheren Schichten kann immer noch ein verbindungsorientierter Dienst (z.B. TCP) über eine verbindungslose Vermittlungsschicht (z.B. IP) realisiert werden!

4.2.1 Netzwerke mit virtuellen Leitungen

→ Der Pfad zwischen Sender und Empfänger verhält sich wie eine Telefonleitung

- Aufbau einer Verbindung, bevor Daten transportiert werden können
  Danach: Abbau der Verbindung
- Jedes Paket beinhaltet einen VC-Identifier (Virtual Channel = Virtuelle Leitung) und keine Zieladresse
- *Jeder Router* auf dem Pfad vom Sender zum Empfänger verwaltet einen Zustand für diese Verbindung
- Ressourcen von Links und des Routers können einer virtuellen Leitung zugeordnet sein (zugeordnete Ressourcen = vorhersagbare Dienstgüte)

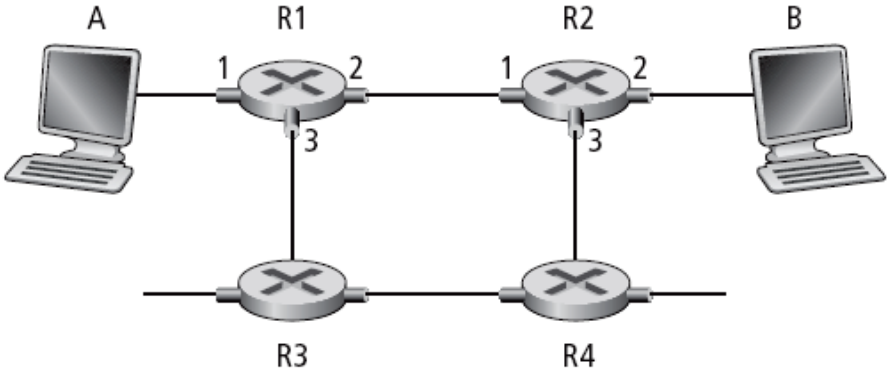
4.2.1 Implementierung virtueller Leitungen

Eine virtuelle Leitung besteht aus:

1. Einem Pfad vom Sender zum Empfänger
2. VC-Identifier, ein Identifier für jeden Link entlang des Pfades
3. Einträgen in den Weiterleitungstabellen der Router auf dem Pfad

- Pakete, die zu einem VC gehören, sind durch einen VC-Identifier (nicht durch die Zieladresse!) gekennzeichnet
- Der VC-Identifier desselben Paketes kann von Link zu Link unterschiedlich sein
  - Die neuen VC-Identifier stehen in der Weiterleitungstabelle der Router

4.2.1 Implementierung virtueller Leitungen

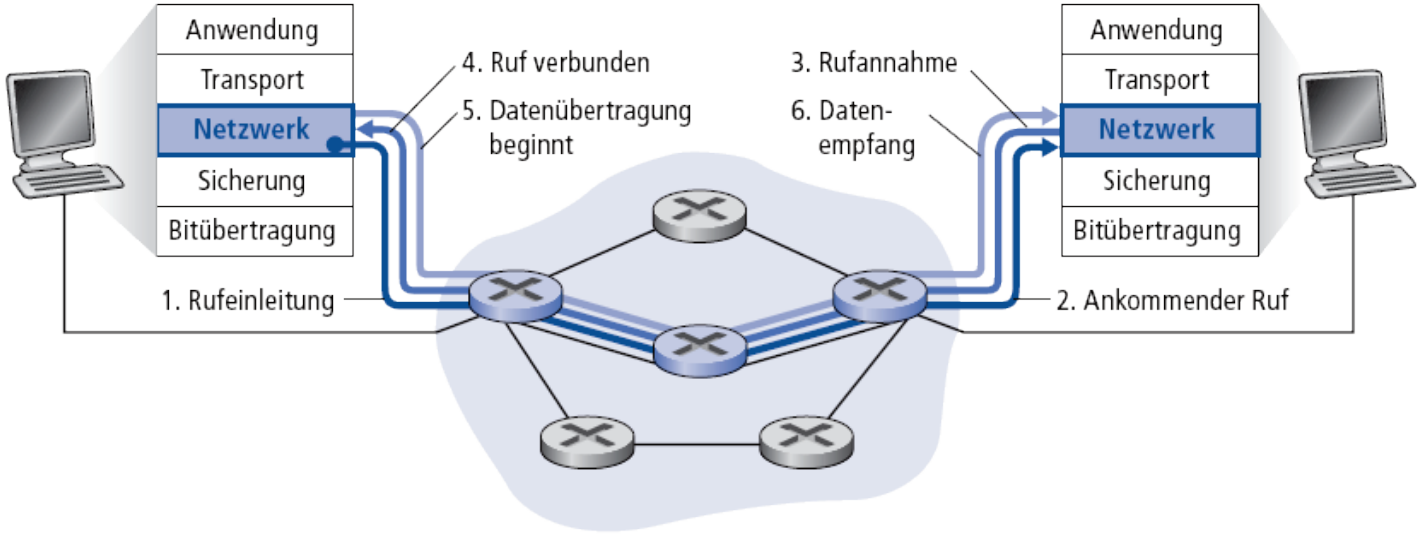


Weiterleitungstabelle in R1:

Eingehende Schnittstelle	Eingehende VC-Nummer	Ausgehende Schnittstelle	Ausgehende VC-Nummer
1	12	2	22
2	63	1	18
3	7	2	17
1	97	3	87
...	...	...	...

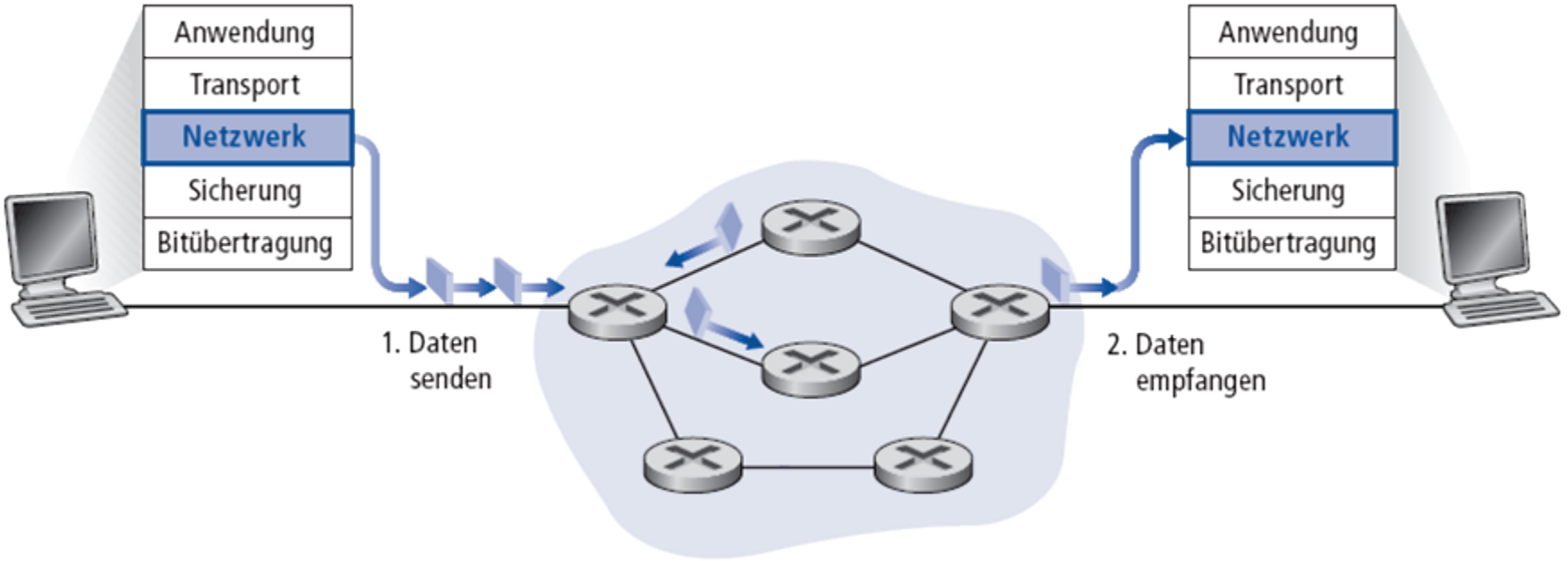
→ Router verwalten Zustand für jede Verbindung!

4.2.1 Virtuelle Leitungen: Signalisierungsprotokolle



- Verwendet zum Aufbau, Aufrechterhalten und Abbau von virtuellen Leitungen
- Verwendet in ATM, Frame Relay und X.25
- Nicht im Internet!

4.2.2 Datagrammnetzwerke



- Kein Verbindungsaufbau auf der Netzwerkschicht
- Router halten keinen Zustand für Ende-zu-Ende-Verbindungen
  - Auf Netzwerkebene gibt es das Konzept einer “Verbindung” nicht!
- Pakete werden unter Verwendung einer Zieladresse weitergeleitet
  - Pakete für dasselbe Sender-Empfänger-Paar können je nach Konfiguration in unterschiedlichen Richtungen einen unterschiedlichen Pfad nehmen

4.2.2 Weiterleitungstabelle eines Routers

Zieladressbereich	Schnittstelle
11001000 00010111 00010000 00000000	
bis	0
11001000 00010111 00010111 11111111	
11001000 00010111 00011000 00000000	
bis	1
11001000 00010111 00011000 11111111	
11001000 00010111 00011001 00000000	
bis	2
11001000 00010111 00011111 11111111	
sonst	3

4.2.2 Longest Prefix Matching

Passender Präfix	Schnittstelle
11001000 00010111 00010	0
11001000 0 0010111 00011000	1
11001000 00010111 00011	2
sonst	3

Beispiele

Adresse: 11001000 00010111 00010110 10100001 → **Schnittstelle 0**

Passt zum 3. Eintrag! → Schnittstelle 2

Adresse: 11001000 00010111 00011000 10101010

Passt zum 2. Eintrag! → **Schnittstelle 1**

Gibt es mehrere Treffer, verwendet der Router die **Longest-Prefix-Matching-Regel** (längstes übereinstimmendes Präfix).

4.2.2 Datagramme und virtuelle Verbindungen

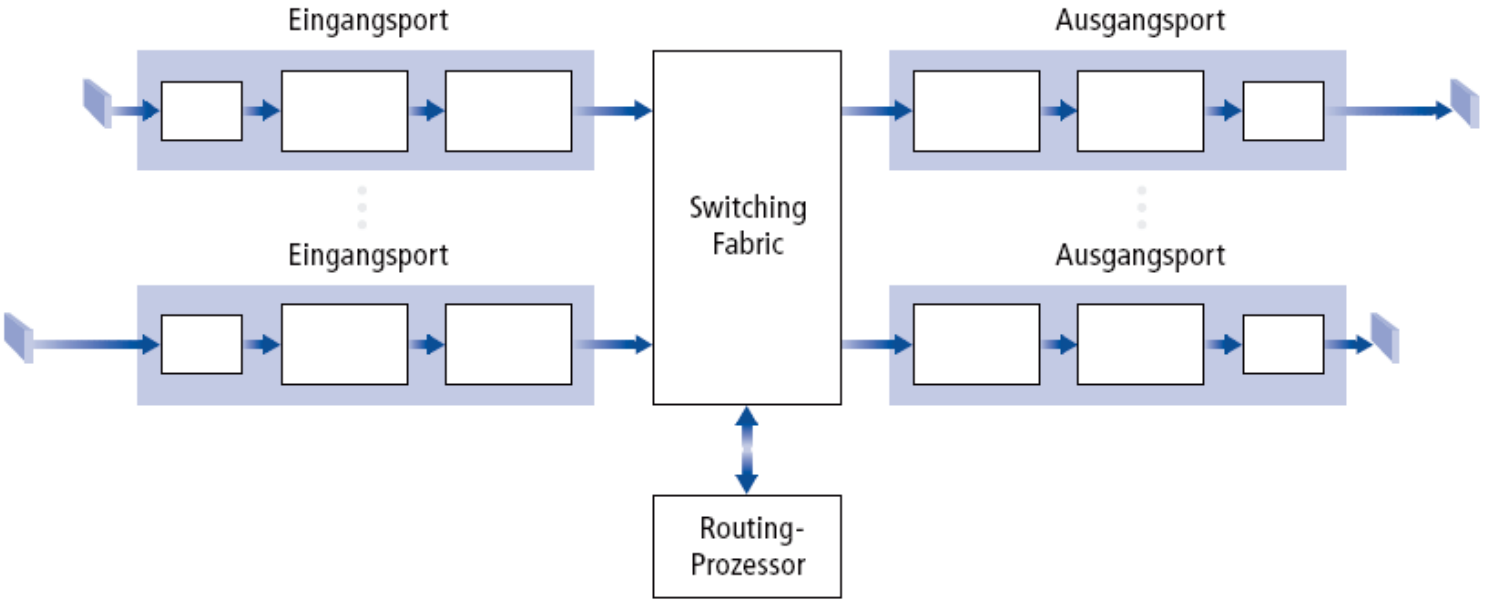
Internet

- Datenaustausch zwischen Computern
  - Keine Echtzeitanforderungen
- Mächtige Endsysteme
  - Können sich anpassen und Fehler beheben
  - Konsequenz: einfaches Netzwerk, Komplexität in den Endsystemen
- Vielzahl verschiedener Links
  - Unterschiedliche Charakteristika
  - Einheitlicher Dienst schwierig zu realisieren

ATM (Asynchronous Transfer Mode)

- Stammt von der klassischen Telefontechnologie ab
- Menschliche Kommunikation
  - Hohe Anforderungen an Echtzeit und Zuverlässigkeit
  - Dienstgarantien sind notwendig
- Einfache Endsysteme
  - Telefone
  - Konsequenz: einfache Endsysteme, Komplexität im Netzwerk

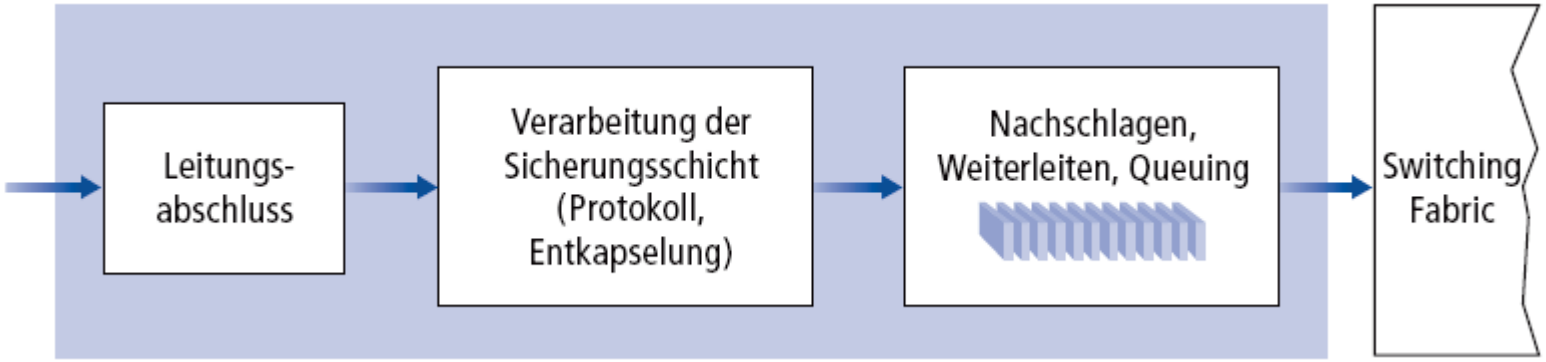
4.3 Router



Zwei wichtige Aufgaben eines Routers:

- Ausführen von Routing-Algorithmen und -Protokollen
  - RIP, OSPF, BGP
- Weiterleiten von Datagrammen von einem eingehenden zu einem ausgehenden Link

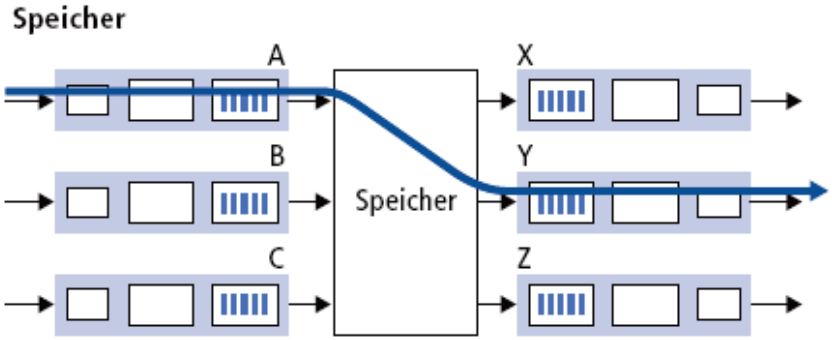
4.3.1 Eingangsports

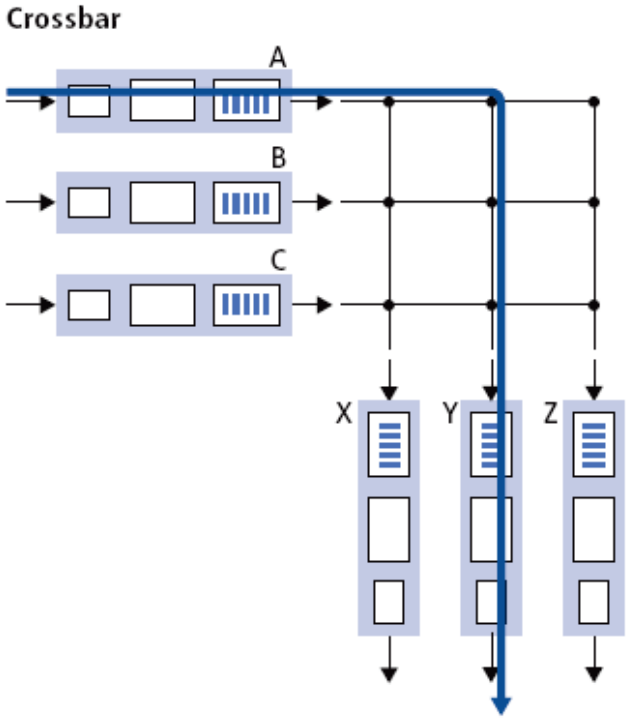


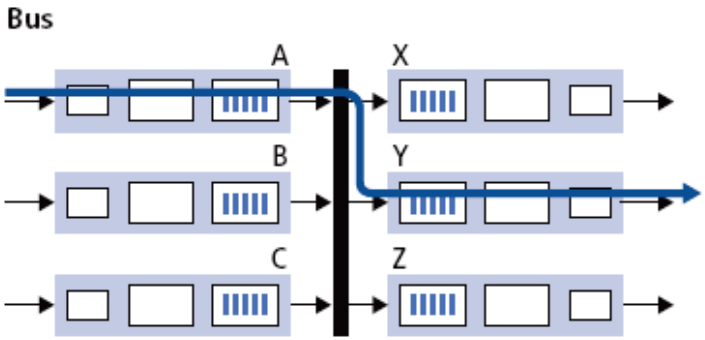
- Leitungsabschluss: physikalische Schicht, Bits empfangen
- Sicherungsschicht: z.B. Ethernet (s. Kapitel 5)
- Nachschlagen, Weiterleiten, Queuing:
  - Suche nach einem geeigneten Ausgangsport
  - Dezentral, Kopie der Routing-Tabelle (oder Teile davon) notwendig
  - Ziel: Behandlung der Pakete mit „line speed“, also mit der Geschwindigkeit der Eingangsleitung des Ports
  - Puffern von Paketen, wenn die Switching Fabric belegt ist

4.3.2 Das Switching Fabric

Drei Switching-Techniken:

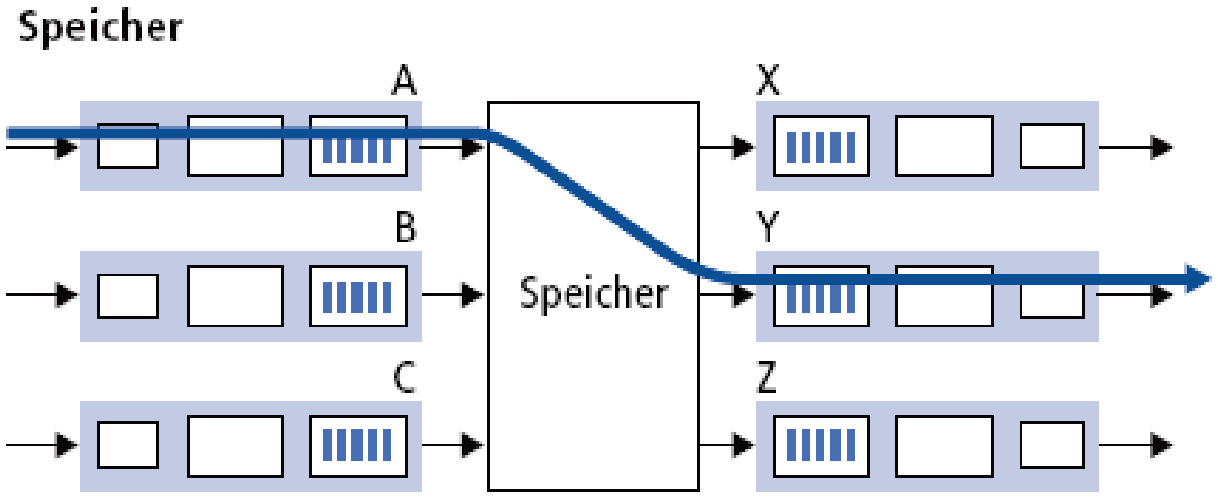






4.3.2 Das Switching Fabric

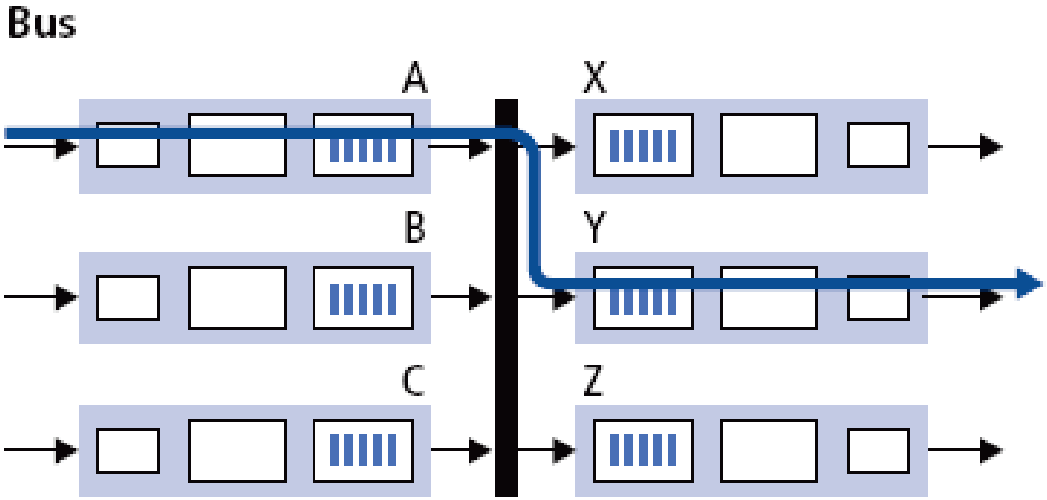
1. Switching über den Speicher:



- Erste Routergeneration:
  - „Normale“ Rechner, Switching wird über die CPU durchgeführt
  - Paket von Eingangsport in den Hauptspeicher kopieren
  - Paket vom Hauptspeicher in den Ausgangsport kopieren
  - Geschwindigkeit durch Speicherbus beschränkt!
    - Zwei Speicherzugriffe: einer zum Schreiben, einer zum Lesen

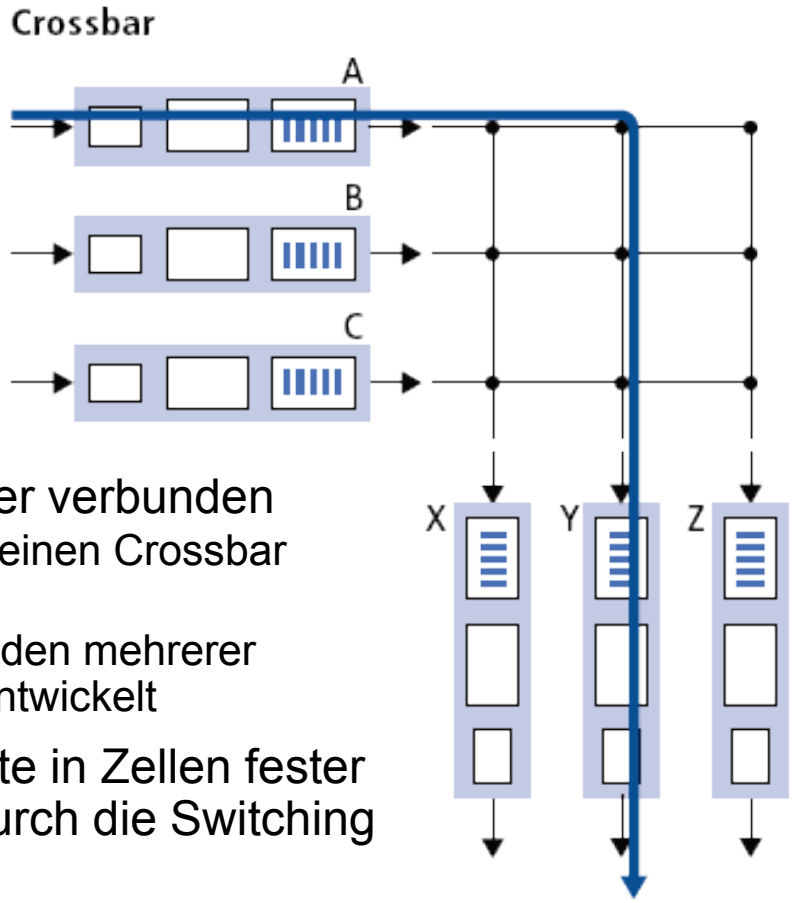
4.3.2 Das Switching Fabric

2. Switching über einen Bus:



- Alle Ports teilen sich einen gemeinsamen Bus
- **Bus Contention**: Die gesamte Kommunikation erfolgt über den Bus, dieser beschränkt die Bandbreite des Routers
  - Aber: nur eine Busoperation (nicht zwei!)
  - *Beispiel*: 32-Gbps-Bus, Cisco 5600, ausreichend für Zugangsrouter und Router für Firmennetze (nicht geeignet im Backbone!)

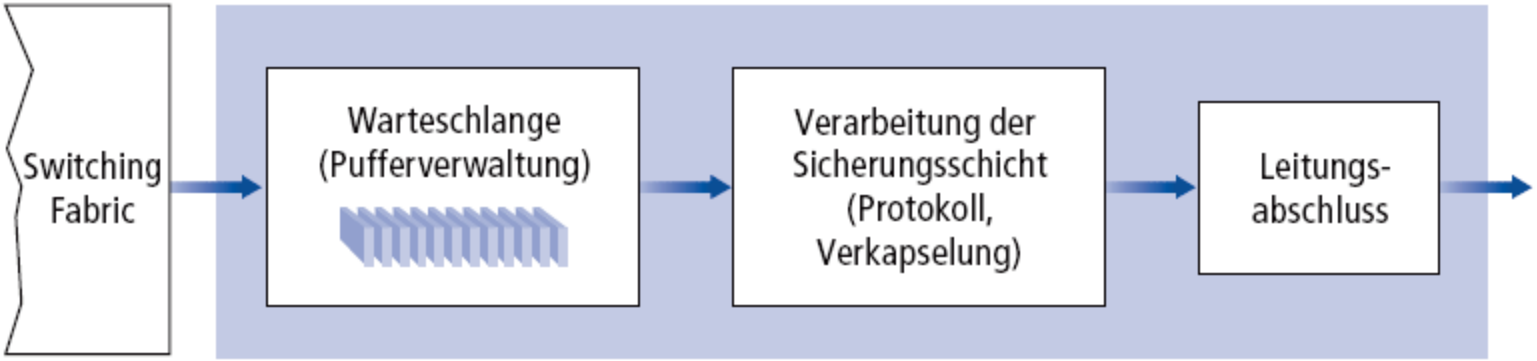
4.3.2 Das Switching Fabric



3. Switching über ein Spezialnetz:

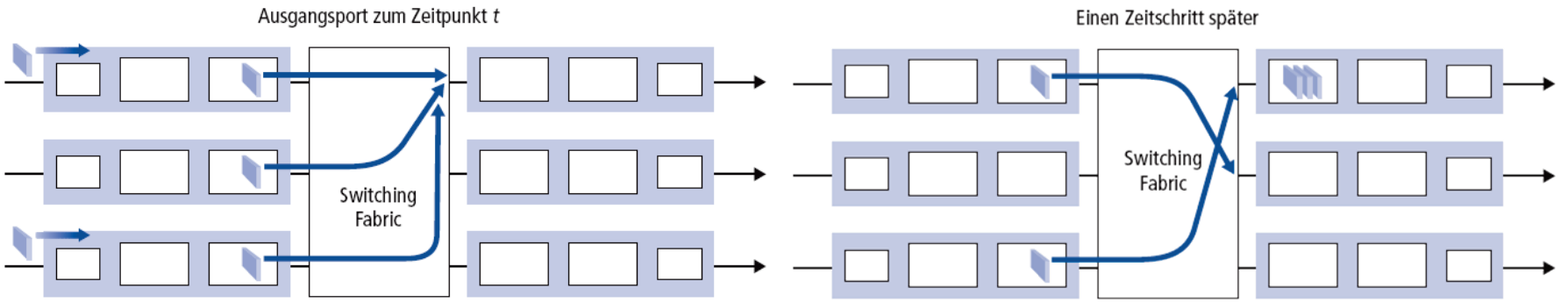
- Ports sind über ein Netzwerk miteinander verbunden
  - Beispielsweise alle Eingangsports über einen Crossbar mit allen Ausgangsports
  - Technologie ursprünglich für das Verbinden mehrerer Prozessoren in einem Parallelrechner entwickelt
- Weitere Fortschritte: Zerlegen der Pakete in Zellen fester Größe, Zellen können dann schneller durch die Switching Fabric geleitet werden
  - *Beispiel*: Cisco 12000, Switching von 60 Gbps durch das interne Netz

4.3.3 Ausgangsports



- Prinzipiell: analog zum Eingangsport!
- Einfacher, da die Entscheidung über die Weiterleitung schon getroffen ist

4.3.4 Puffern im Ausgangsport



- Puffern von Paketen, wenn sie schneller aus der Switching Fabric kommen, als sie auf die Leitung gelegt werden können
- Auswirkungen:
  - Gepufferte Pakete werden verzögert
  - Wenn der Puffer überläuft, müssen Pakete verworfen werden
- **„Scheduling Discipline“:** bestimmt die Reihenfolge, in der gepufferte Pakete auf die Leitung gelegt werden

4.3.4 Puffern im Ausgangsport

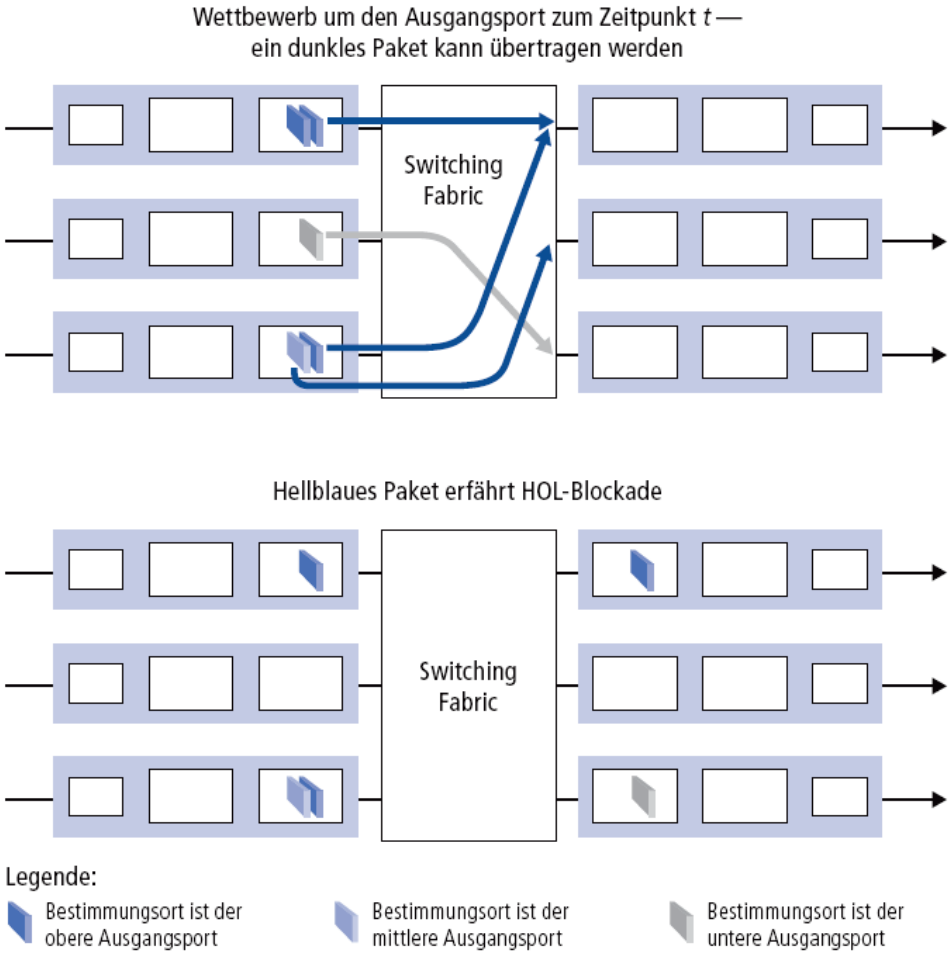
Wie groß sollten die Puffer sein?

- RFC 3439 beschreibt folgende **Faustregel**: Die Größe des Puffers sollte der Rundlaufzeit (RTT, z.B. 250 ms) multipliziert mit der Datenrate des Links entsprechen
  - Bei einem 10 Gps Link, 250 ms RTT, ergibt das 2,5 Gbit Puffer
- Neuere Empfehlungen: bei N Datenflüssen und Link-Datenrate C:

$$\frac{RTT \cdot C}{\sqrt{N}}$$

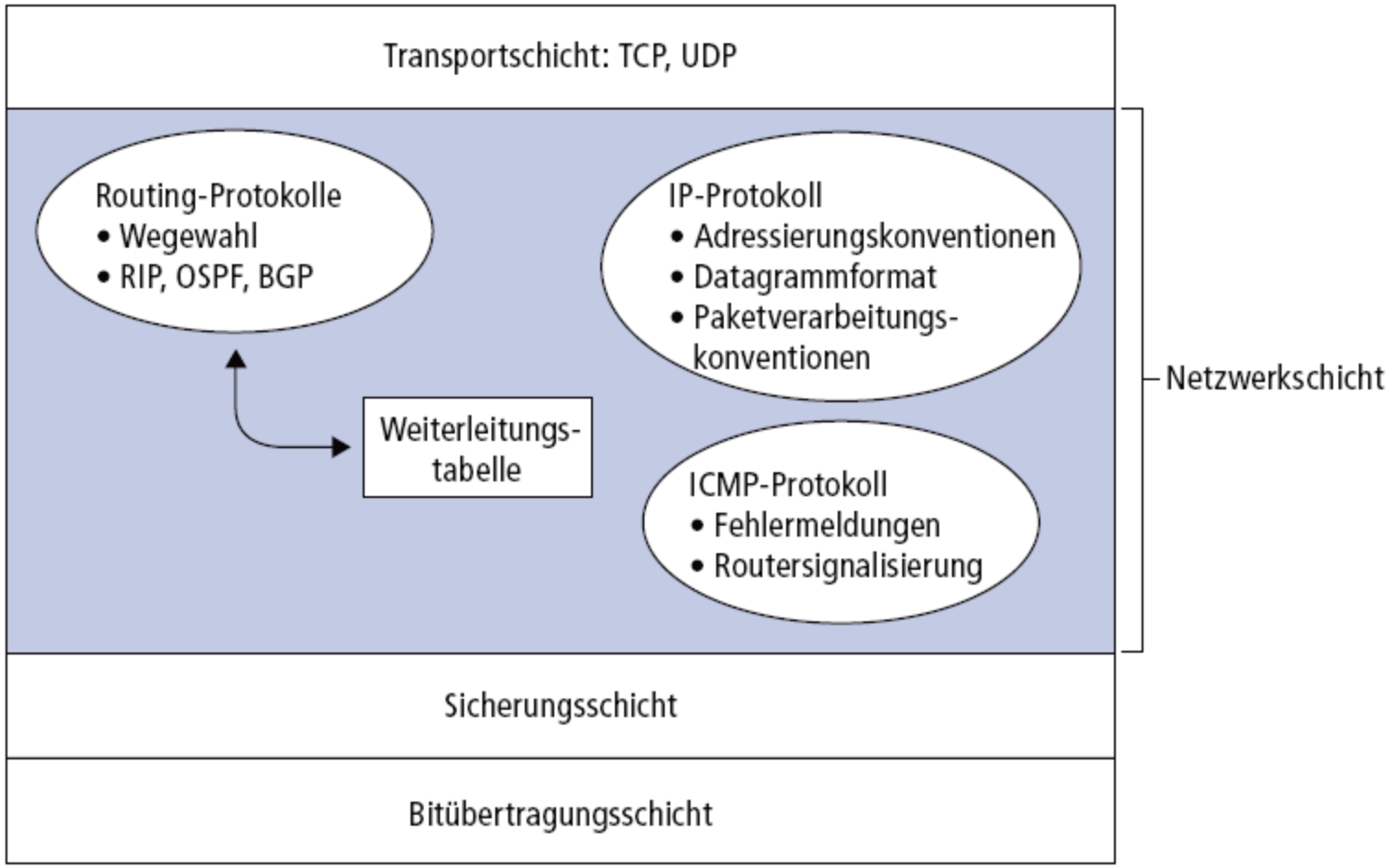
4.3.4 Puffern im Inputport

- Wenn die Switching Fabric ein Paket nicht direkt weiterleiten kann, muss dieses im Eingangsport gepuffert werden
- Dort kann es ein Paket blockieren, welches eigentlich bereits durch die Switching Fabric geleitet werden könnte
  → **Head-of-Line (HOL) Blocking**

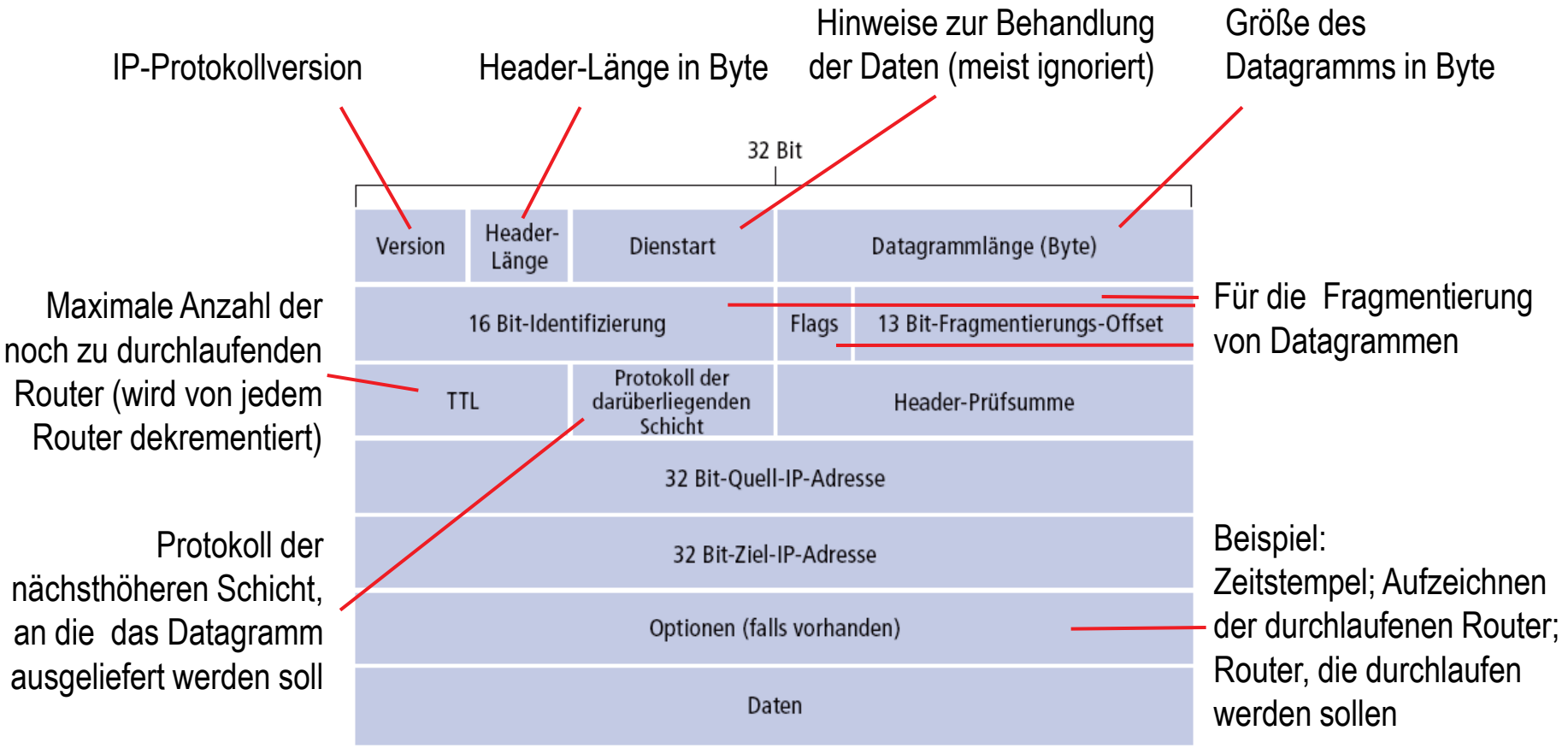


4.4 IP: Internetprotokoll

Die Netzwerkschicht des Internets:



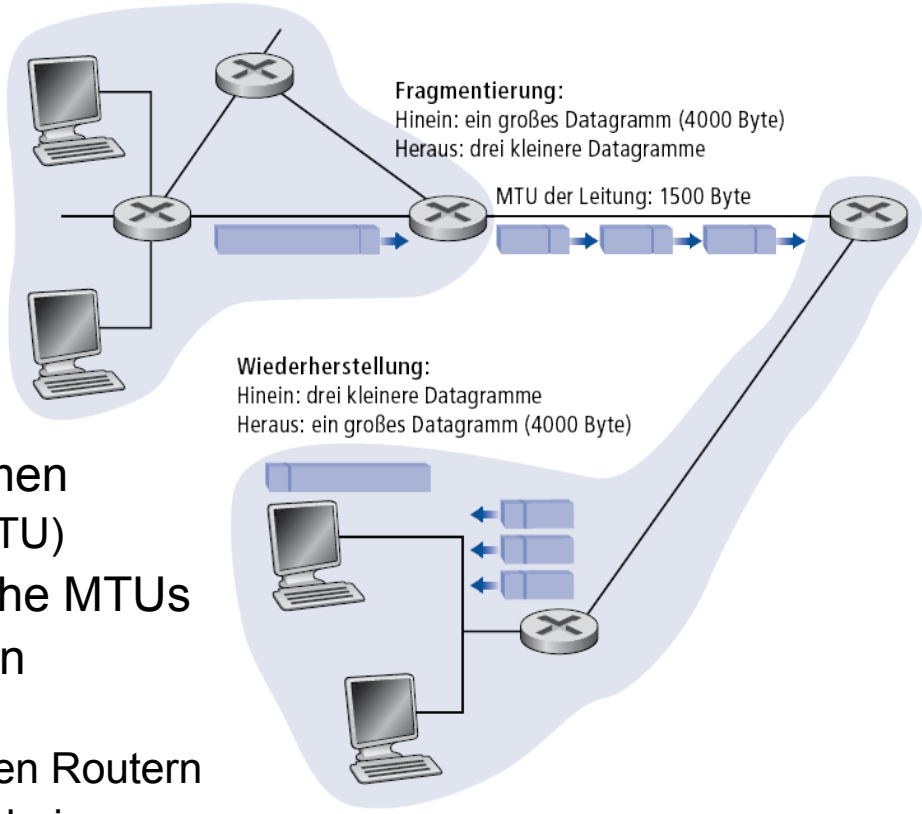
4.4.1 IP-Datagrammformat



Wie viel Overhead entsteht bei Verwendung von TCP?

→ 20 Byte für den TCP-Header, 20 Byte für den IP-Header= 40 Byte + Overhead auf der Anwendungsschicht

4.4.1 IP-Datagramm-Fragmentierung



- Links haben eine Maximalgröße für Rahmen
  - Genannt Maximum Transmission Unit (MTU)
- Verschiedene Links haben unterschiedliche MTUs
- IP-Datagramme müssen unter Umständen aufgeteilt werden
  - Aufteilung (Fragmentierung) erfolgt in den Routern
  - Zusammensetzen (Reassembly) erfolgt beim Empfänger
  - IP-Header enthält die notwendigen Informationen hierzu

4.4.1 IP-Datagramm-Fragmentierung

Beispiel: IP-Datagramm mit 4000 Byte (inklusive 20 Byte IP-Header), MTU des nächsten Links = 1500 Byte

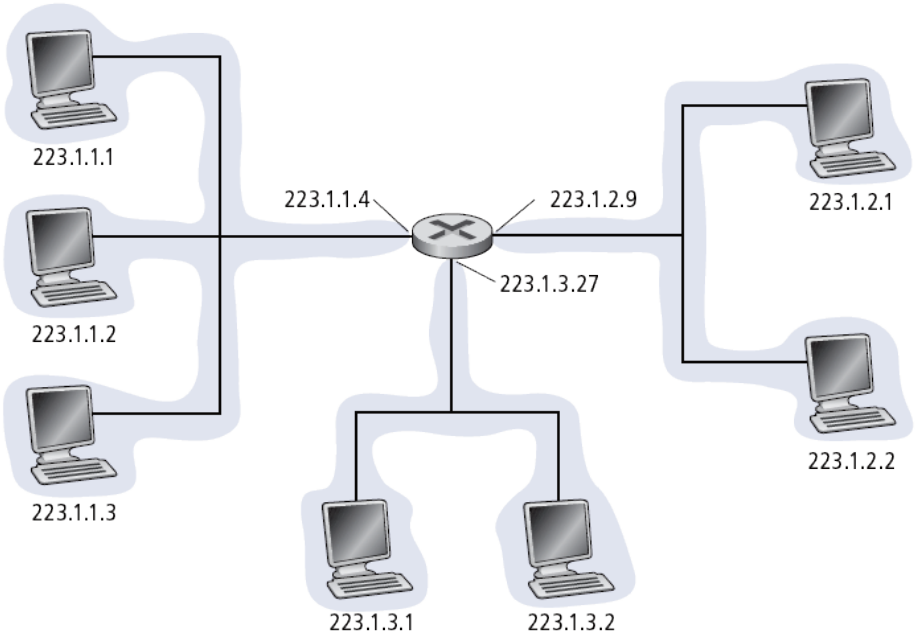
Fragment	Bytes	ID	Offset	Flag
1. Fragment	1.480 Byte im Datenfeld des IP-Datagramms	Identifizierung = 777	Offset = 0 (d.h., die Daten sollten beginnend bei Byte 0 eingefügt werden)	Flag = 1 (d.h., da kommt noch mehr)
2. Fragment	1.480 Datenbytes	Identifizierung = 777	Offset = 185 (d.h., die Daten sollten bei Byte 1.480 beginnend eingefügt werden; beachten Sie, dass 185 · 8 = 1.480)	Flag = 1 (d.h., da kommt noch mehr)
3. Fragment	1.020 Datenbytes (= 3.980 – 1.480 – 1.480)	Identifizierung = 777	Offset = 370 (d.h., die Daten sollten beginnend bei Byte 2.960 eingefügt werden; beachten Sie, dass 370 · 8 = 2.960)	Flag = 0 (d.h., es ist das letzte Fragment)

4.4.1 IP-Datagramm-Fragmentierung

- Praktisch:
  - Endsystem/Anwendung muss sich keine Gedanken über die Größe von MTUs verschiedener Links auf dem Weg vom Sender zum Empfänger machen
- Entspricht dem Prinzip einer geschichteten Architektur
- Aber:
  - Aufwand in den Routern
  - Wenn ein Fragment verloren geht, ist das ganze Datagramm verloren
- Daher: *Fragmentation considered harmful!*
- Lösung: Bestimmen der kleinsten MTU des Weges (Path MTU)
  - Setze **DF (Don't-Fragment-Bit)** im Header des IP-Paketes
  - Wenn fragmentiert werden soll, wird das Paket verworfen und der Sender per ICMP benachrichtigt
  - Sender wählt dann kleinere MTU
  - Wiederholen, bis akzeptable MTU gefunden wurde

4.4.2 IPv4-Adressierung

- IP-Adresse: 32-Bit-Kennung für das Interface (Schnittstelle) eines Endsystems oder eines Routers
- Interface: Verbindung zwischen dem System und dem Link
  - Wird normalerweise durch eine Netzwerkkarte bereitgestellt
  - Router haben typischerweise mehrere Interfaces
  - Endsysteme können ebenfalls mehrere Interfaces haben
  - Jedes Interface besitzt eine IP-Adresse

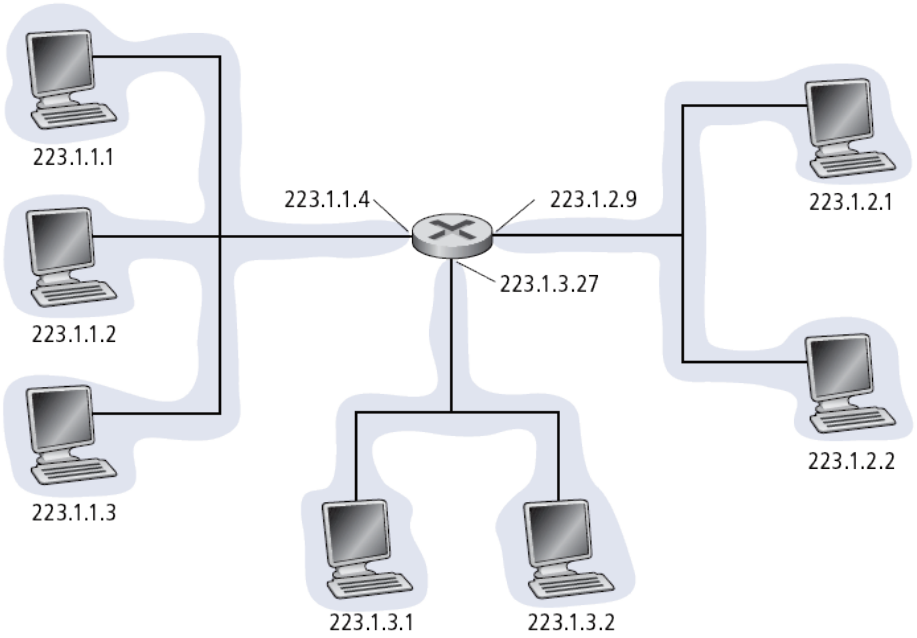


223.1.1.1 = 11011111.00000001.00000001.00000001

223 1 1 1

4.4.2 IPv4-Adressierung

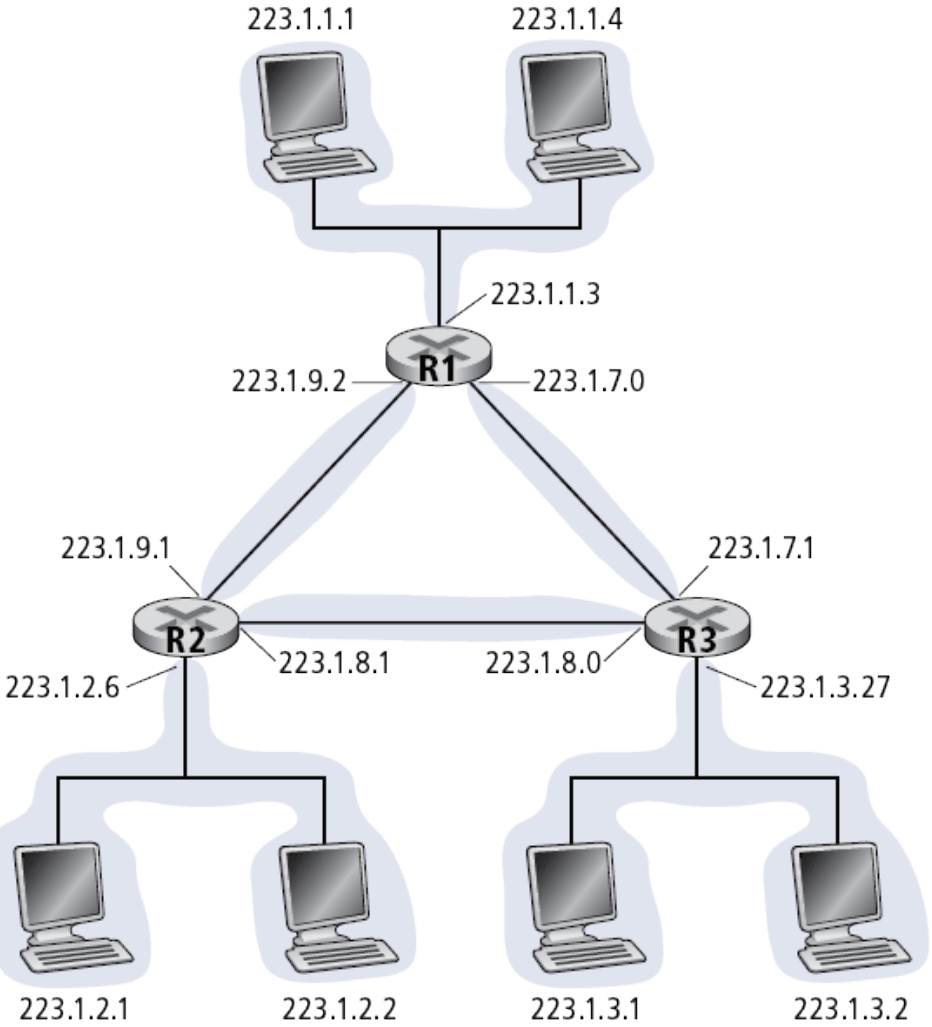
- IP-Adresse:
  - Zwei Bestandteile:
    - **netid**: die oberen Bits der Adresse, identifiziert ein Netzwerk
    - **hostid**: die unteren Bits der Adresse, identifiziert ein Interface eines Systems
- Was ist ein (Sub-)Netzwerk?
  - Alle Interfaces mit derselben netid formen ein Netzwerk
  - Alle Interfaces eines Netzwerkes können sich direkt (ohne einen Router zu durchqueren) erreichen



Drei IP-Netzwerke, die mit einem Router verbunden sind. Die netid steht hier in den oberen 24 Bit.

Subnetzmaske: `/24` oder `255.255.255.0`

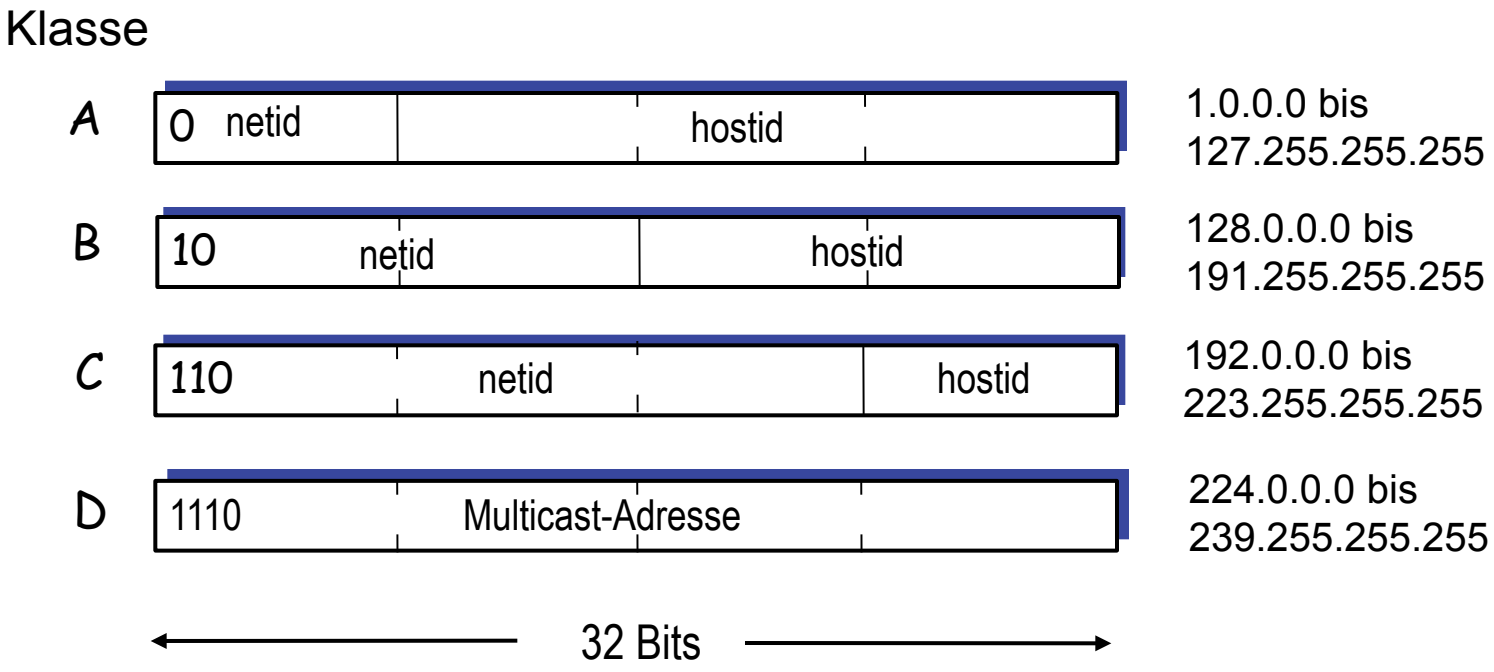
4.4.2 IPv4-Adressierung



Drei Router verbinden sechs Subnetzwerke untereinander.

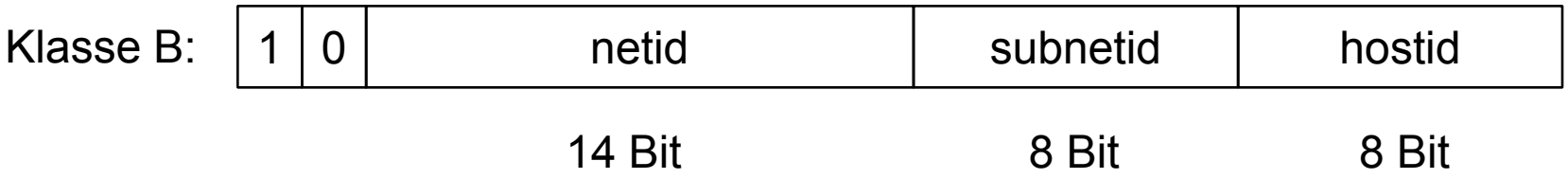
4.4.2 IP Adressierung - Adressklassen

- Früher wurden IP-Adressen in Adressklassen aufgeteilt
- Die Klasse bestimmte das Verhältnis der Längen netid/hostid
- Dies nennt man „classfull“ addressing (klassenbasierte Adressierung)



4.4.2 Adressierung von Subnetzen

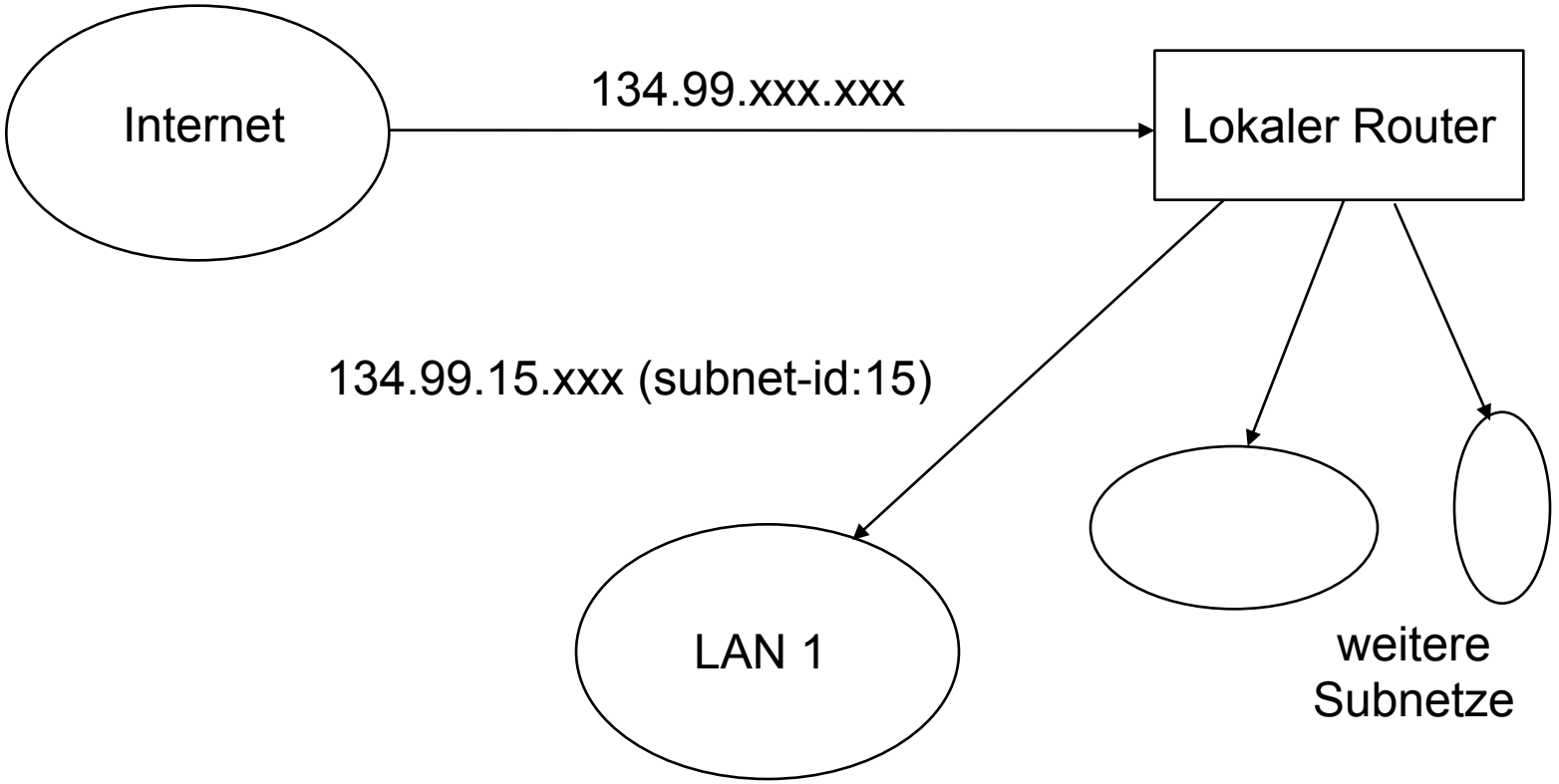
- Klasse-A- und –B-Adressen haben Platz für mehr Endsysteme, als man in einem Netzwerk sinnvoll unterbringen kann
- Daher teilt man die hostid weiter auf, z.B. so:



- Die Unterteilung (subnetid, hostid) ist eine lokale Entscheidung und wird von der Organisation vorgenommen, der die netid zugeordnet wurde

4.4.2 Adressierung von Subnetzen

- Die subnetid ist außerhalb des Netzwerkes, für das sie verwendet wird, nicht sichtbar:



4.4.2 Adressierung von Subnetzen

- Subnetzmaske (subnet mask)
  - Wird für jede IP-Adresse eines Systems im System gespeichert
  - Sie identifiziert, welcher Teil der Adresse zur subnetid und welcher zur hostid gehört

	16 Bit	8 Bit	8 Bit
Beispiel	1111111111111111	11111111	00000000

subnet mask: 0xffffff00=255.255.255.0
oder auch /24

- Die eigene IP-Adresse in Verbindung mit der Subnetzmaske erlaubt Rückschlüsse darüber, wo sich eine andere IP-Adresse befindet:
  - im selben Subnetz (also direkt erreichbar)
  - im selben Netzwerk, aber in einem anderen Subnetz
  - in einem anderen Netzwerk

4.4.2 Beispiel für die Verwendung von Subnetzmasken

- Gegeben:
  - Eigene IP-Adresse: `134.155.48.10`
  - Subnetzmaske: `255.255.255.0`
  - Adresse A: `134.155.48.96`, Adresse B: `134.155.55.96`

- Überprüfen der beiden Adressen:
  - `134.155.48.10` & `255.255.255.0` = `134.155.48.0`
  - `134.155.48.96` & `255.255.255.0` = `134.155.48.0` → identisch, gleiches Subnetz
  - `134.155.55.96` & `255.255.255.0` = `134.155.55.0` → verschieden, anderes Subnetz

4.4.2 Subnetzmasken variabler Länge

- Problem: Gegeben sei ein Klasse-C-Netzwerk, welches in zwei Subnetze mit 50 Endsystemen und ein Subnetz mit 100 Endsystemen unterteilt werden soll.

- Das funktioniert nicht mit einer einzelnen Subnetzmaske!
  - `255.255.255.128`: zwei Netze mit je 128 hostids
  - `255.255.255.192`: vier Netze mit je 64 hostids

- Lösung: Subnetzmasken variabler Länge
  - Unterteile den Adressraum zunächst mit der kürzeren Subnetzmaske (1 Bit im Beispiel)
  - Unterteile eine Hälfte davon weiter mit der längeren Subnetzmaske (2 Bit im Beispiel)
  - Resultat: Subnetze verschiedener Größe

4.4.2 Subnetzmasken variabler Länge

Beispiel:

- Klasse-C-Netzwerk: `193.43.55.x`
- Subnetzmaske für das Subnetz mit der ID 0 (100 Endsysteme):
  - `255.255.255.128`
  - Adressen in diesem Subnetz: `193.43.55.0-127`
- Subnetzmaske für die Subnetze mit den IDs 2 und 3 (je 50 Endsysteme):
  - `255.255.255.192`
  - Adressen im Subnetz 2: `193.43.55.128-191`
  - Adressen im Subnetz 3: `193.43.55.192-255`

4.4.2 Klassenbasierte Adressierung

- Verteilung der Adressen:
  - Durch zentrale Organisationen (z.B. IANA)
  - Netzweise (also z.B. Klasse-B-Netz für ein Unternehmen)
  - Relativ chaotisch:
    - Zuteilung der numerisch nächsten netid an den nächsten Nachfrager
- Probleme:
  - Verschwendung von IP-Adressen:
    - Das Unternehmen könnte 2^{16}, also mehr als 65.000 Adressen vergeben
    - Nur ein Bruchteil davon wird genutzt
  - *Classful* Routing-Tabellen werden schnell sehr groß:
    - Ein separater Eintrag für eine jede netid der Klasse C, auch wenn mehrere Klasse C Netze zusammengefasst werden könnten!
      → Routing-Tabellen müssen mit hoher Frequenz aufgefrischt werden
      » Immer, wenn ein Klasse C Netzwerk hinzukommt, wegfällt oder sich verändert, muss dies im ganzen Internet bekanntgegeben werden

4.4.2 Klassenlose Adressierung

- Idee:
  - Die Aufteilung nach netid/hostid wird immer explizit per Subnetzmaske durchgeführt
  - Keine explizite Unterscheidung zwischen netid und subnetid
- Schreibweise:
  - a.b.c.d/x, wobei x die Länge der netid (Prefix) bestimmt
  - Alternative Schreibweise zur Subnetzmaske
  - Beispiel:
    - 192.48.96.0/23
    - 11000000.00110000.01100000.00000000 (netidhostid)

4.4.2 Adressvergabe bei Hosts - DHCP

- Bei Hosts erfolgt die Adressvergabe durch die Konfiguration von:
  - IP-Adresse
  - Subnetzmaske
  - Weiteren Parametern

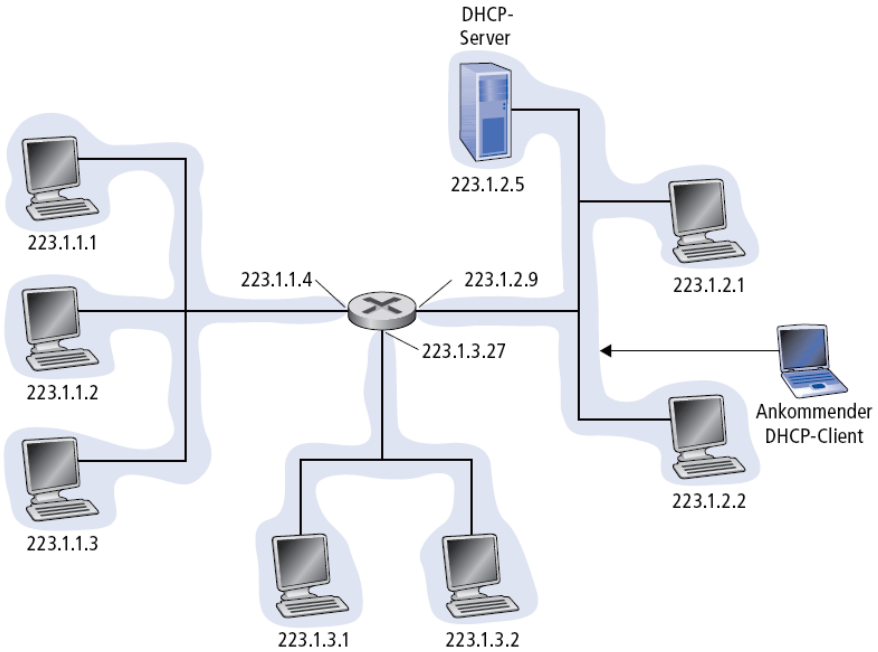
- Problem: Wie bekommen Endsysteme ihre IP-Adresse (und andere Parameter der Netzwerkschicht, wie z.B. die Subnetzmaske)?
  - Manuelle Konfiguration ist fehleranfällig
  - → Lösung: Automatische Zuweisung mithilfe von DHCP [RFC 2131]

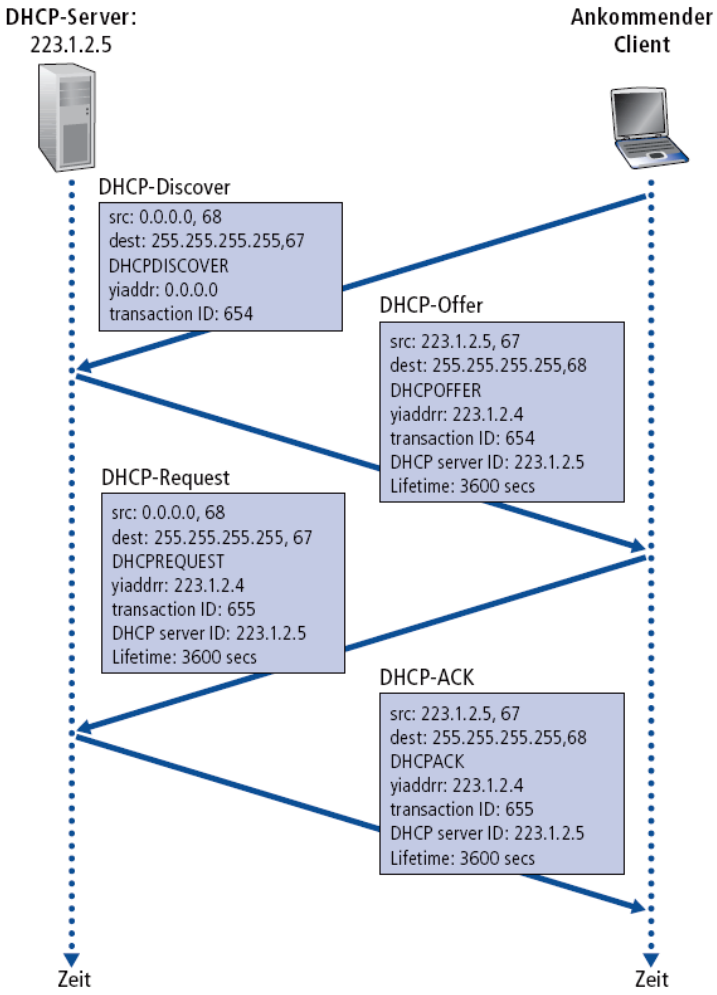
→ **DHCP**: Dynamic Host Configuration Protocol: dynamisches Beziehen der Adresse von einem Server
  - "Plug-and-Play"

4.4.2 DHCP

- Ziele von DHCP:
  - Automatische Vergabe von Adressen und Parametern
  - Keine Konfiguration der Endsysteme notwendig
  - Unterstützung von „nomadischen“ Benutzern
- Prinzipieller Ablauf
  - Endsystem schickt eine **DHCP-Discover-Nachricht** per IP-Broadcast (Adresse 255.255.255.255)
  - DHCP-Server antwortet mit einer **DHCP-Offer-Nachricht**
  - Endsystem beantragt eine IP-Adresse: **DHCP-Request-Nachricht**
  - DHCP-Server vergibt Adresse: **DHCP-Ack-Nachricht**
- Funktionen von DHCP [definiert in RFC2131]:
  - Verschiedene Arten der Adresszuweisung (manuell, permanent, temporär)
  - Verlängerung und Rückgabe der Adresse durch den Client
  - Konfigurieren von Parametern

4.4.2 DHCP Szenario





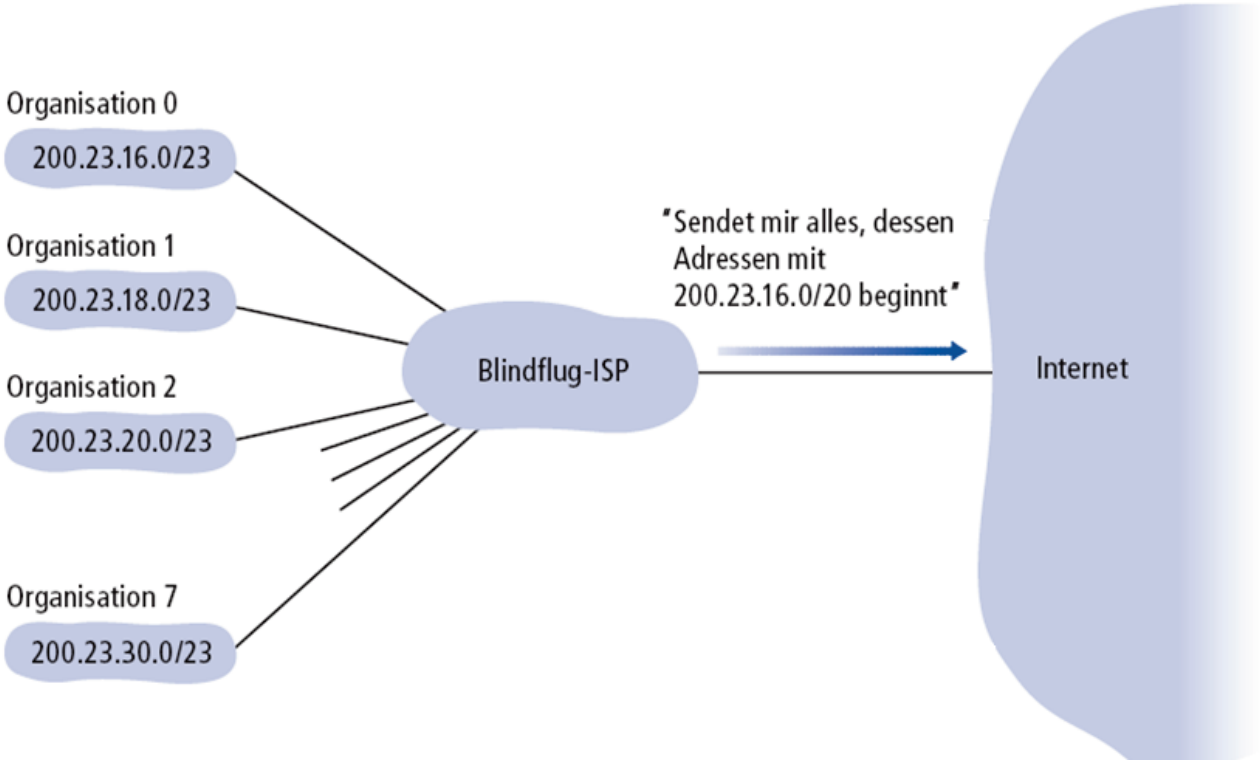
- DHCP verwendet UDP
- DHCP-Nachrichten werden an die MAC-Broadcast-Adresse geschickt
- Es gibt ein Feld, in dem eine eindeutige Kennung des Clients verpackt ist → dies ist meist die MAC-Adresse.

4.4.2 Adressvergabe bei Netzwerken

- Bei der klassenlosen Adressierung (CIDR) werden zusammenhängende Adressbereiche von der **Internet Assigned Numbers Authority (IANA)** an die **Regional Internet Registries (RIR)** vergeben
  - APNIC (Asia Pacific Network Information Centre) - Asien/Pazifik
  - ARIN (American Registry for Internet Numbers) - Nordamerika und Afrika südlich der Sahara
  - LACNIC (Regional Latin-American and Caribbean IP Address Registry) – Lateinamerika und einige karibische Inseln
  - RIPE NCC (Réseaux IP Européens) - Europa, Mittlerer Osten, Zentralasien und afrikanische Länder nördlich des Äquators
- Diese vergeben zusammenhängende Adressbereiche an ISPs
- ISPs vergeben zusammenhängende Adressbereiche an ihre Kunden
  - Dadurch: Aggregation in Routing-Tabellen teilweise möglich

4.4.2 Adressvergabe bei Netzwerken

ISP	11001000 00010111 00010000 00000000	200.23.16.0/20
Organisation 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organisation 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organisation 2	11001000 00010111 00010100 00000000	200.23.20.0/23
…		
Organisation 7	11001000 00010111 00011110 00000000	200.23.30.0/23



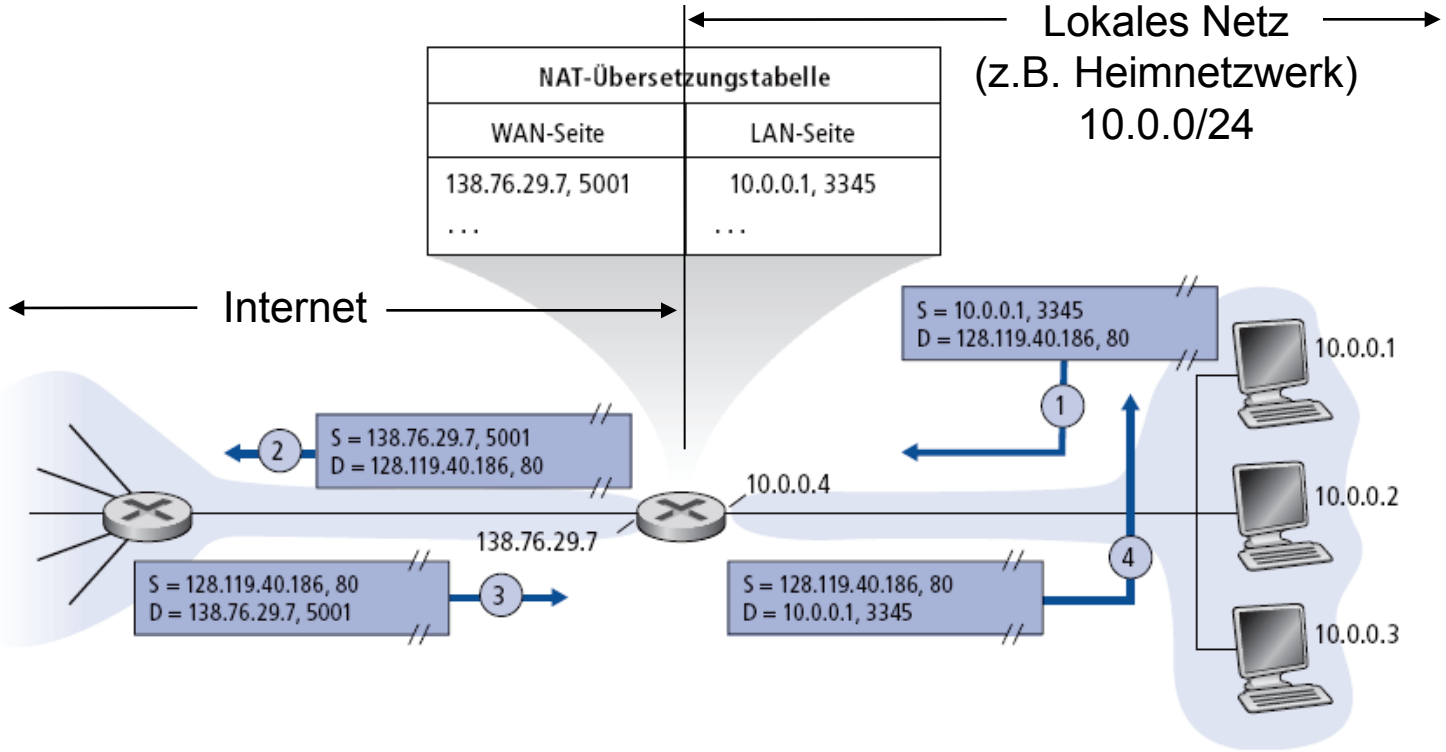
4.4.2 Adressvergabe bei Netzwerken

Address Block	Present Use	Reference
0.0.0.0/8	"This" Network	[RFC1700, page 4]
10.0.0.0/8	Private-Use Networks	[RFC1918]
14.0.0.0/8	Public-Data Networks	[RFC1700, page 181]
24.0.0.0/8	Cable Television Networks	---
39.0.0.0/8	Reserved but subject to allocation	[RFC1797]
127.0.0.0/8	Loopback	[RFC1700, page 5]
128.0.0.0/16	Reserved but subject to allocation	---
169.254.0.0/16	Link Local	---
172.16.0.0/12	Private-Use Networks	[RFC1918]
191.255.0.0/16	Reserved but subject to allocation	---
192.0.0.0/24	Reserved but subject to allocation	---
192.0.2.0/24	Test-Net	---
192.88.99.0/24	6 to 4 Relay Anycast	[RFC3068]
192.168.0.0/16	Private-Use Networks	[RFC1918]
198.18.0.0/15	Network Interconnect Device Benchmark Testing	[RFC2544]
223.255.255.0/24	Reserved but subject to allocation	---
224.0.0.0/4	Multicast	[RFC3171]
240.0.0.0/4	Reserved for Future Use	[RFC1700, page 4]

4.4.2 Network Adress Translation (NAT)

- Motivation:
  - Häufig hat man nur eine IP-Adresse, aber mehrere Endsysteme
  - Diese ist meist nur temporär (per DHCP) zugewiesen
  - Man möchte bei einem Provider-Wechsel nicht die IP-Adressen der Endsysteme verändern
  - IP-Adressen im eigenen Netzwerk sollen aus Sicherheitsgründen nicht vom Internet aus sichtbar sein
  - Interne IP-Adressen sollen veränderbar sein, ohne dass der Rest des Internets darüber informiert werden muss
- Idee:
  - Vergebe lokale (weltweit nicht eindeutige) Adressen an die Systeme im eigenen Netzwerk
  - Router zur Anbindung an das Internet übersetzt diese Adressen in eine gültige, weltweit eindeutige IP-Adresse
  - Dazu wird die Adressierung auf der Transportschicht gebraucht: Ports

4.4.2 Network Adress Translation (NAT)



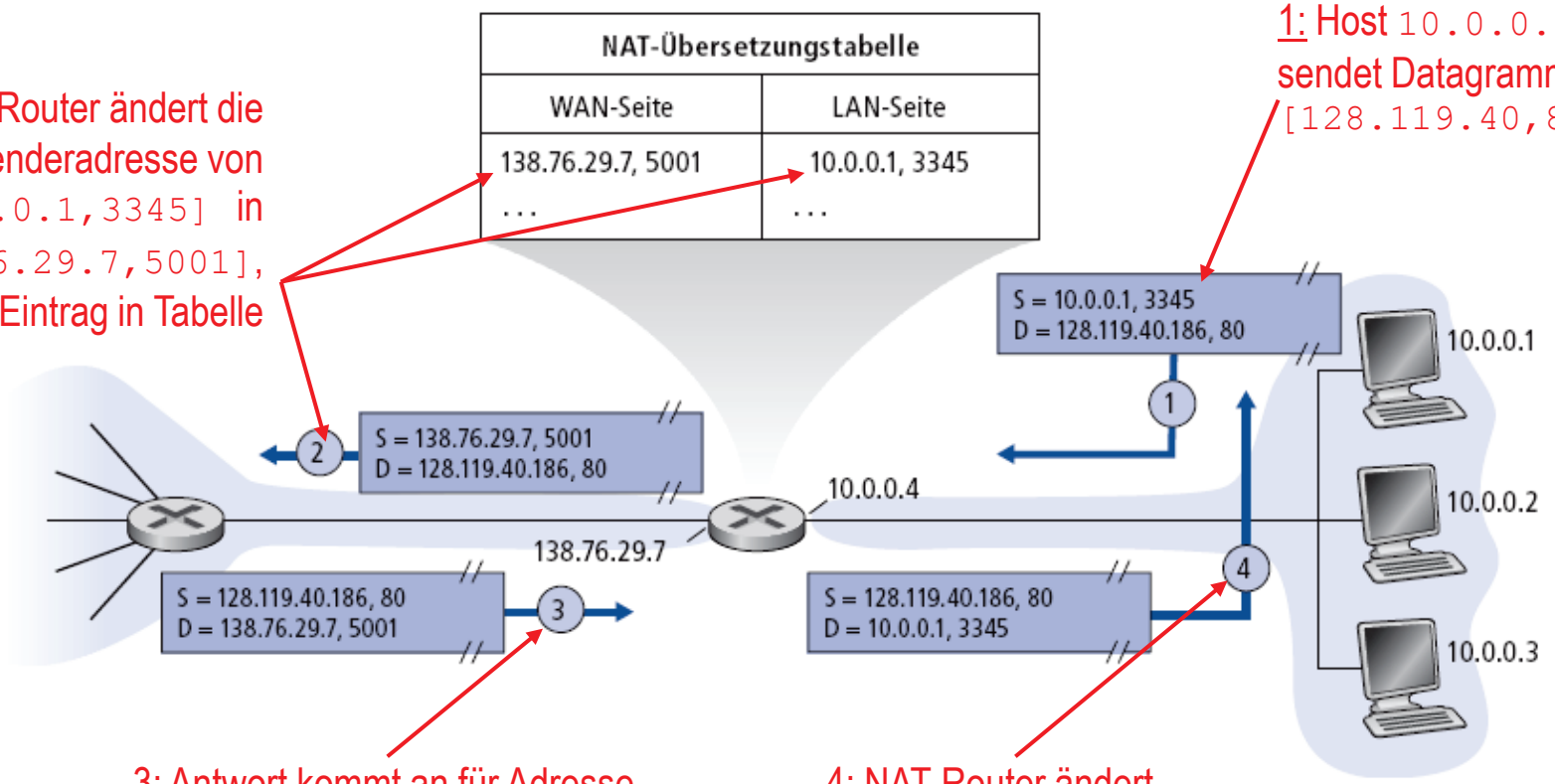
Alle Datagramme die das lokale Netz *verlassen*, haben die *gleiche* NAT-IP-Adresse als Absender: 138.76.29.7, unterschieden werden sie über die Portnummern.

→ Datagramme mit Sender oder Empfänger in diesem Netzwerk haben 10.0.0/24 als Adresse für diesen Sender/Empfänger.

4.4.2 Network Adress Translation (NAT)

- Implementierung:
  Ein NAT-Router muss Folgendes tun:
  - *Ausgehende Datagramme*:
    Ersetze `[Sender-IP-Adresse,Portnummer]` im Absenderfeld für jedes ins Internet geleitete Datagramm durch `[NAT-IP-Adresse,neue Portnummer]`
    → Kommunikationspartner wird die Antworten an `[NAT-IP-Adresse,neue Portnummer]` schicken
  - Speichere in einer NAT-Tabelle die Abbildung zwischen `[Sender-IP-Adresse, Portnummer]` und `[NAT-IP-Adresse,neue Portnummer]`
  - *Ankommende Datagramme*:
    Ersetze `[NAT-IP-Adresse,neue Portnummer]` im Empfängerfeld durch `[Sender-IP-Adresse,Portnummer]` aus der NAT-Tabelle

4.4.2 Network Adress Translation (NAT)



4.4.2 Network Adress Translation (NAT)

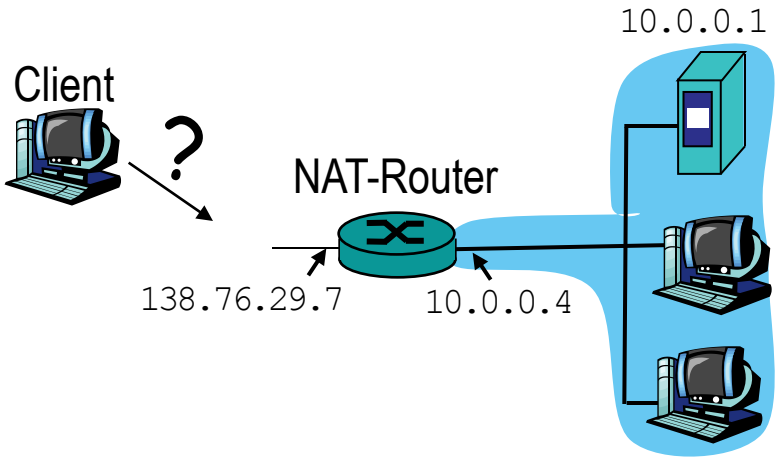
- 16-Bit-Port Number-Feld:
  - Mehr als 60.000 gleichzeitige Verbindungen mit einer IP-Adresse sehr einfach möglich
- NAT ist nicht unumstritten:
  - Router sollten nur Informationen der Schicht 3 verwenden
  - Verletzung des End-to-End Principle (Ende-zu-Ende Prinzip):
    - Transparente Kommunikation von Endsystem zu Endsystem nicht möglich, da im Netz die Adressen durch die NATs übersetzt werden!
    - Der Anwendungsentwickler muss die Präsenz von NAT-Routern berücksichtigen
      - Beispiel: Verwenden der Host-IP-Adresse als weltweit eindeutige Nummer nicht möglich!
  - NAT dient hauptsächlich der Bekämpfung der Adressknappheit im Internet. Dies sollte besser über IPv6 (mehr dazu später) erfolgen!

4.4.2 NAT-Traversal

→ NAT-Traversal nennt man das Durchqueren von Routern, die NAT einsetzen.

Angenommene Situation:

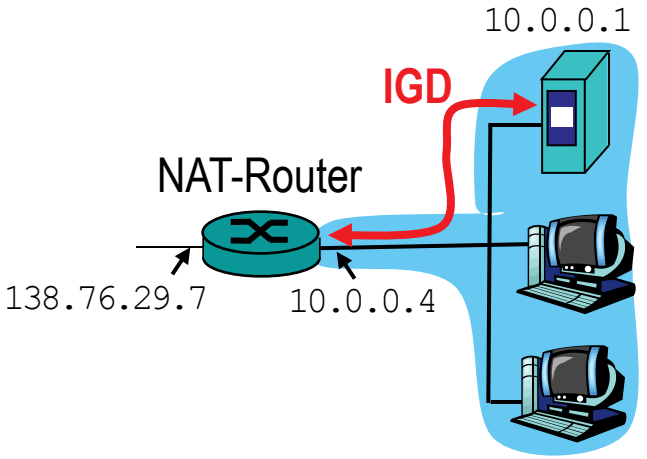
- Der Client möchte den Server mit der Adresse 10.0.0.1 kontaktieren
  - Die Adresse 10.0.0.1 ist eine lokale Adresse und kann nicht als Adresse im globalen Internet verwendet werden
  - Die einzige nach außen sichtbare Adresse ist: 138.76.29.7
- **Lösung 1**: Statische Konfiguration von NAT, so dass eingehende Anfragen angemessen weitergeleitet werden
  - Beispiel: [123.76.29.7, Port 2500] wird immer an [10.0.0.1, Port 25000] weitergeleitet



4.4.2 NAT-Traversal

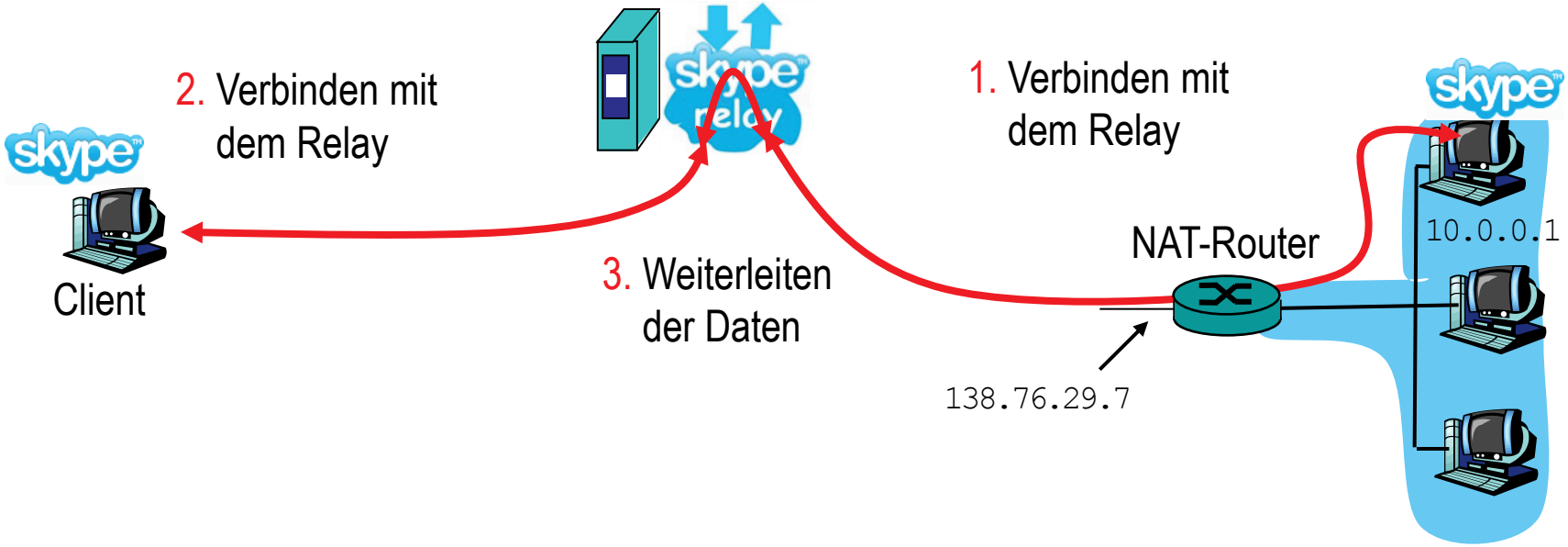
- **Lösung 2**: Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Dies ermöglicht dem Host hinter dem NAT Folgendes:
  - Herausfinden der öffentlichen IP-Adresse des NAT-Routers (138.76.29.7)
  - Auffinden existierender Abbildungen in der NAT-Tabelle
  - Einträge in die NAT-Tabelle einfügen oder aus ihr löschen

  → Das heißt automatische Konfiguration von statischen NAT-Einträgen!



4.4.2 NAT-Traversal

- **Lösung 3**: Relaying (von Skype verwendet)
  - Server hinter einem NAT-Router baut eine Verbindung zu einem Relay auf (welches nicht hinter einem NAT-Router liegt)
  - Client baut eine Verbindung zum Relay auf
  - Relay leitet die Pakete vom Client zum Server und umgekehrt weiter



4.4.3 Internet Control Message Protocol (ICMP)

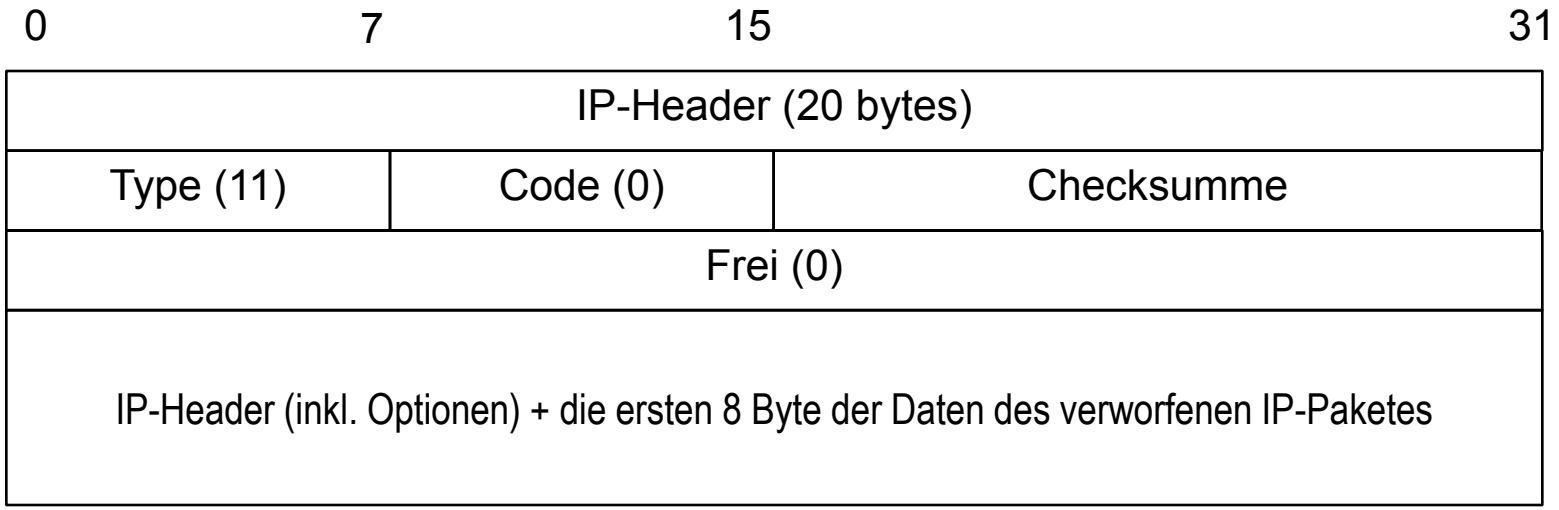
- Wird von Hosts und Routern verwendet, um Informationen über das Netzwerk selbst zu verbreiten
  - Fehlermeldungen: Host, Netzwerk, Port, Protokoll nicht erreichbar
  - Echo-Anforderung und Antwort (von ping genutzt)
- Gehört zur Netzwerkschicht, wird aber in IP-Datagrammen transportiert
- **ICMP-Nachricht**: Type, Code und die ersten 8 Byte des IP-Datagramms, welches die Nachricht ausgelöst hat

Type	Code	Beschreibung
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

4.4.3 Traceroute

- Aufgabe von Traceroute:
  - Traceroute bestimmt Informationen über alle Router, die auf dem Weg zu einer IP-Adresse liegen
  - Dabei wird auch die RTT zu jedem Router bestimmt
- Funktionsweise von Traceroute:
  1. Traceroute schickt ein UDP-Paket an die Adresse, für die der Weg untersucht werden soll; TTL im IP-Header wird auf 1 gesetzt
  2. Der erste Router verwirft das IP-Paket (TTL = 1!) und schickt eine ICMP-Time-Exceeded-Fehlermeldung an den Absender
  3. Traceroute wiederholt dies mit TTL = 2 etc.

4.4.3 ICMP-Time-Exceeded-Nachricht



4.4.3 Traceroute und ICMP

Wie erkennt man, ob das Paket schließlich beim Empfänger angekommen ist?

- Traceroute sendet UDP-Pakete an einen Port, der wahrscheinlich nicht verwendet wird, und erwartet eine ICMP-Port-Unreachable-Nachricht vom Empfänger!
- Demo:
  - `traceroute <host>`

→ So lernt das Quellsystem Anzahl und Identitäten der Router kennen, die zwischen ihm und dem Zielhost liegen und es kann die RTT zwischen den beiden Hosts messen.