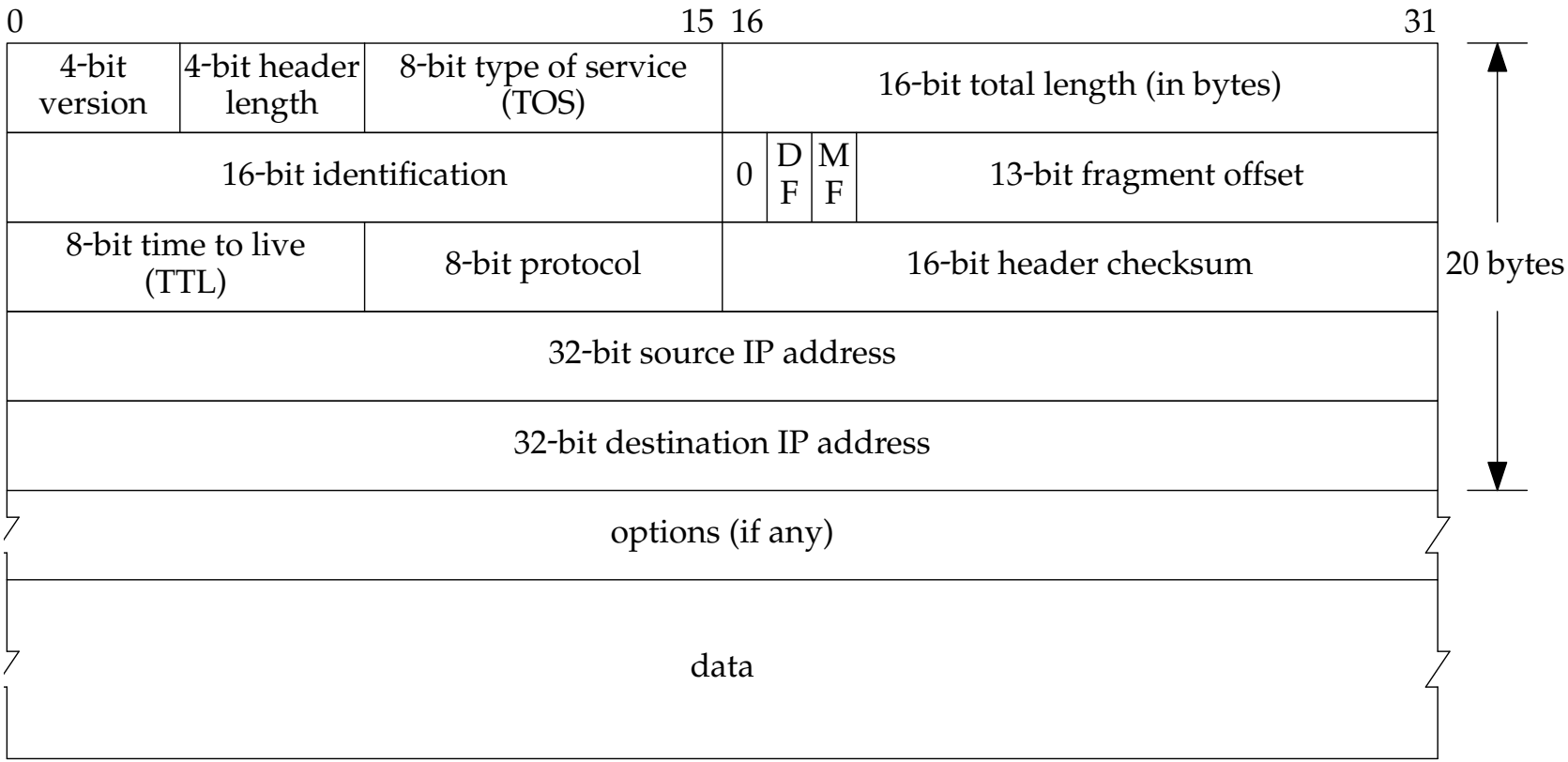
Internet Protokoll (IP)

IPv4 Rahmen


0
15
16
31
4-bit version
4-bit header length
8-bit type of service (TOS)
16-bit total length (in bytes)
16-bit identification
0
D F
M F
13-bit fragment offset
8-bit time to live (TTL)
8-bit protocol
16-bit header checksum
20 bytes
32-bit source IP address
32-bit destination IP address
options (if any)
data

- **Version:** 4, falls IPv4, 6 für IPv6
- **Header Length:** Länge des IP Headers inklusive Optionen gezählt in 4Byte Worten, d.h. mindestens 5 (für 20Bytes). Maximale Länge des IP Headers ist demnach 60Bytes.
- **Type of Service:** Hinweis an Router, wie der Paketpfad zu optimieren ist, früher 3Bit Priorität, 4Bit Optimierungsrichtlinie (RFC1349)

  1000 Minimiere Verzögerungen
  0100 Maximiere Durchsatz
  0010 Maximiere Zuverlässigkeit
  0001 Minimiere Kosten
  1111 Maximiere Sicherheit (RFC1455)

  Abgelöst durch RFC2474 für DS, RFC3168 für ECN:
  - 2Bit Explicit Congestion Notification (00: Knoten kann kein ECN, 01/10: Knoten unterstützt ECN, 11: Link/Knoten in Überlast)
  - 6Bit Differentiated Services

- **Total Length:** Anzahl Bytes im gesamten Paket (Header + Daten), max. 64kByte
- **Identification:** Identifikationsnummer eines (noch unfragmentierten) Paketes, soll vom höheren Layer festgelegt werden.
- **Don't Fragment Flag:** Flag, das es Knoten im Pfad verbietet, das Paket zu fragmentieren
- **More Fragments Flag:** Zeigt an, daß das Paket fragmentiert ist und mindestens ein weiteres Fragment nach nach diesem kommt.
- **Fragment Offset:** Offset eines Fragments im Gesamtpaket, in 8Byte Einheiten
- **TTL:** Time To Life, jeder Router im Datenpfad dekrementiert dieses Feld, ist TTL 0 erreicht, wird eine Fehlermeldung erzeugt.

- **Protocol:** Code für den verwendeten höheren Layer, vgl. RFC1700, RFC3232 und `http://www.iana.org/assignments/protocol-numbers`
- **Header Checksum:** Prüfsumme über (nur) den Header des Paketes
- **Source Address:** Netzwerkadresse (IP) des Senders
- **Destination Address:** IP des Zieles
- **Options:** Folge von Protokolloptionen oder deren Resultate. Die Länge des Feldes ergibt sich aus der Header Length. Muß auf 4Byte Grenze aufgefüllt werden.
- **Data:** Nachricht der darüberliegenden Schicht

Fragmentierung

- Die Rahmen der Sicherungsschicht können nur Pakete einer bestimmten maximalen Größe (MTU, Maximum Transfer Unit) übertragen.
- Muß ein Knoten auf dem Weg zwischen Quelle und Ziel ein Paket übertragen, das diese Größe überschreitet, kann er
  1. das Paket auf mehrere Rahmen aufteilen (fragmentieren).
  2. eine Fehlermeldung generieren.
- Ist das **Don't Fragment Flag** gesetzt, wird eine Fehlermeldung erzeugt, andernfalls wird so fragmentiert, daß es dem nächsten Link genügt.
- Bei Bedarf kann ein Rahmen mehrfach fragmentiert werden.

Beispiel Fragmentierung

Eine Nachricht von 3000 Bytes soll in einem IP Paket ohne Optionen mittels Ethernet (MTU 1500Bytes) übertragen werden.

Wir benötigen 3 Fagmente:

Nutzdaten	Länge	Fragment Offset	DF	MF
1480 Bytes	1500	0	0	1
1480 Bytes	1500	185	0	1
40 Bytes	60	370	0	0

Bemerkung: Fragmentierung ist die Ursache für eine Reihe von Problemen in der Netzwerkinfrastruktur, z.B durch

- Senden in umgekehrter Reihenfolge.
- Senden nur eines späten Fragmentes.

Prüfsummenbildung

Die Länge eines IP Headers ist immer durch 4 teilbar.

Zur Bildung der Prüfsumme im IP Protokoll wird der Header (ohne Prüfsumme oder mit Wert 0) in 2 Byte Blöcken geschrieben, dann für jede Spalte gerade Parität gebildet.

Das Ergebnis wird in das Feld **Checksum** übernommen. Es folgt ein synthetisches Beispiel mit einem verkürzten (6 Byte) Header:

`Headerdaten:`	`10101001`	`01101010`
	`00100001`	`00101010`
Checksum `Füller:`	`00000000`	`00000000`
Checksum:	`10001000`	`01000000`

Bemerkung: Jeder Knoten, der TTL oder Optionen ändert, muß die Prüfsumme neu berechnen.

IP Optionen

Für eine Liste der IP Optionen siehe
`http://www.iana.org/assignments/ip-parameter`.
Kodierung ist für Optionen 0 und 1 ein Byte, sonst
Code, Länge, Wert.
Einige ausgewählte Optionen aus RFC791:

- 0x00 End of Options
- 0x01 No Option (Füller)
- 0x07 Record Route
- 0x83 Loose Source Route
- 0x89 Strict Source Route

Kodierung Loose Source Routing:

Code	Len	Ptr	n Addresses
131	$n * 4 + 3$	$4, 8, 12, \ldots$	$\ldots$

Beispiel eines IP Rahmens

```
#nc -g 127.0.0.1 10.1.195.159 12345
```

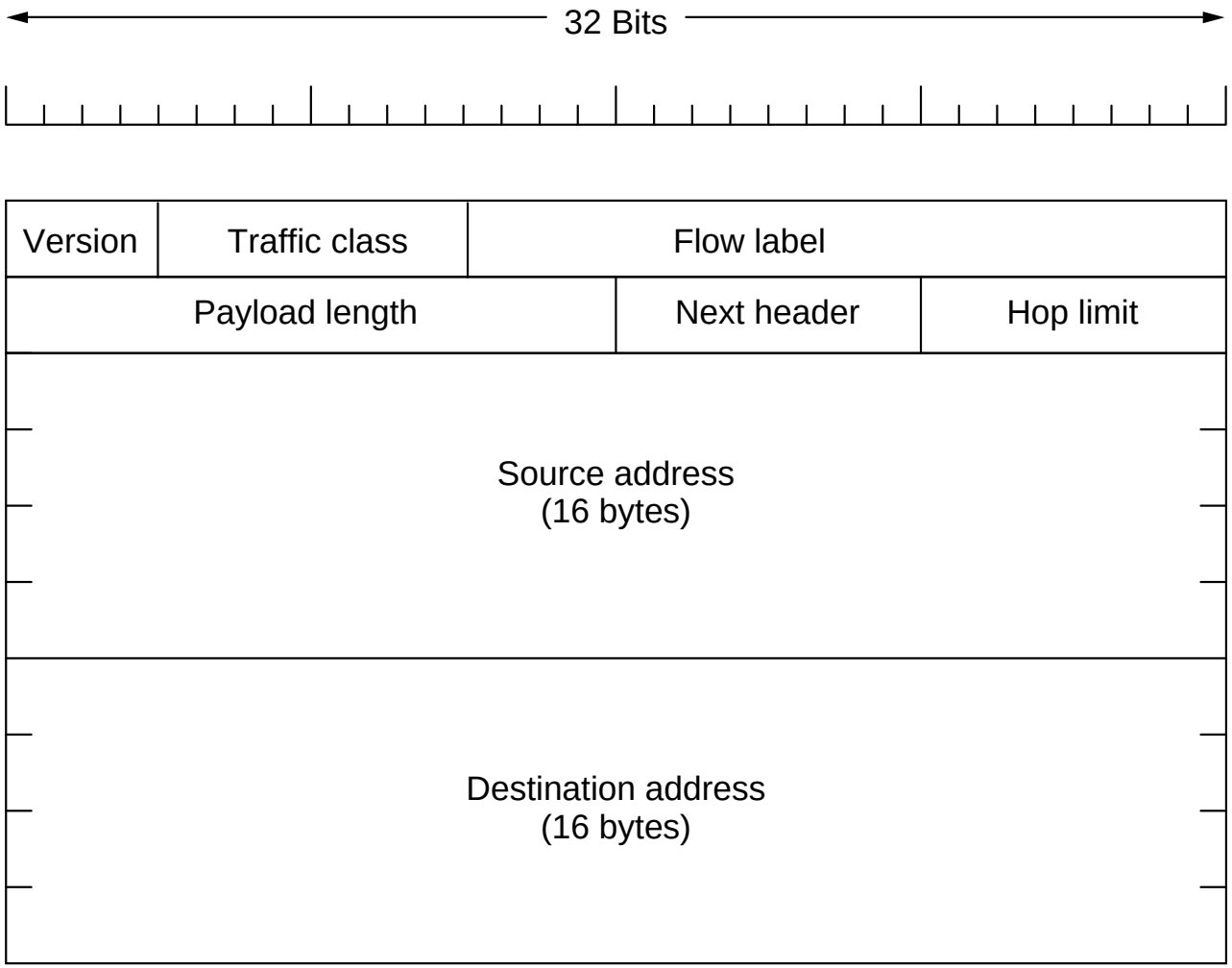
Bytes(hex)	Bedeutung
48	IPv4, 32 Bytes Header
00	Default DS, kein ECN
00 48	72 Bytes im Paket
b2 ed	Identification
40 00	Don't Fragment, Fragment Offset 0
40	Time To Live 64, Standardwert
06	Protocol: TCP
be 18	Checksum
7f 00 00 01	Source IP: 127.0.0.1
7f 00 00 01	Destination IP: 127.0.0.1

Beispiel eines IP Rahmens, Fortsetzung

IP Optionen

Bytes(hex)	Bedeutung
83	131, Loose Source Routing
0b	11 Bytes in dieser Option
04	Zeiger auf erste Adresse
0a 01 c3 9f	Erste Adresse: 10.1.195.159
0a 01 c3 9f	Zweite Adresse: 10.1.195.159
01	NOP

IPv6 Header



(c) Tanenbaum, Computer Networks

Felder im IPv6 Header

- **Version:** 4, falls IPv4, 6 für IPv6
- **Traffic class:** Unterscheidung verschiedener Klassen bezüglich zulässiger Verzögerungen
- **Flow label:** Soll pseudo-Verbindungen ermöglichen
- **Payload length:** Rahmenlänge ohne den 40Byte Header selbst
- **Next header:** Identifier für mögliche extension headers
- **Hop limit:** Maximale Anzahl an Knoten (Hops), ähnlich TTL bei IPv4
- **Source Address:** Netzwerkadresse (IP) des Senders
- **Destination Address:** IP des Zieles

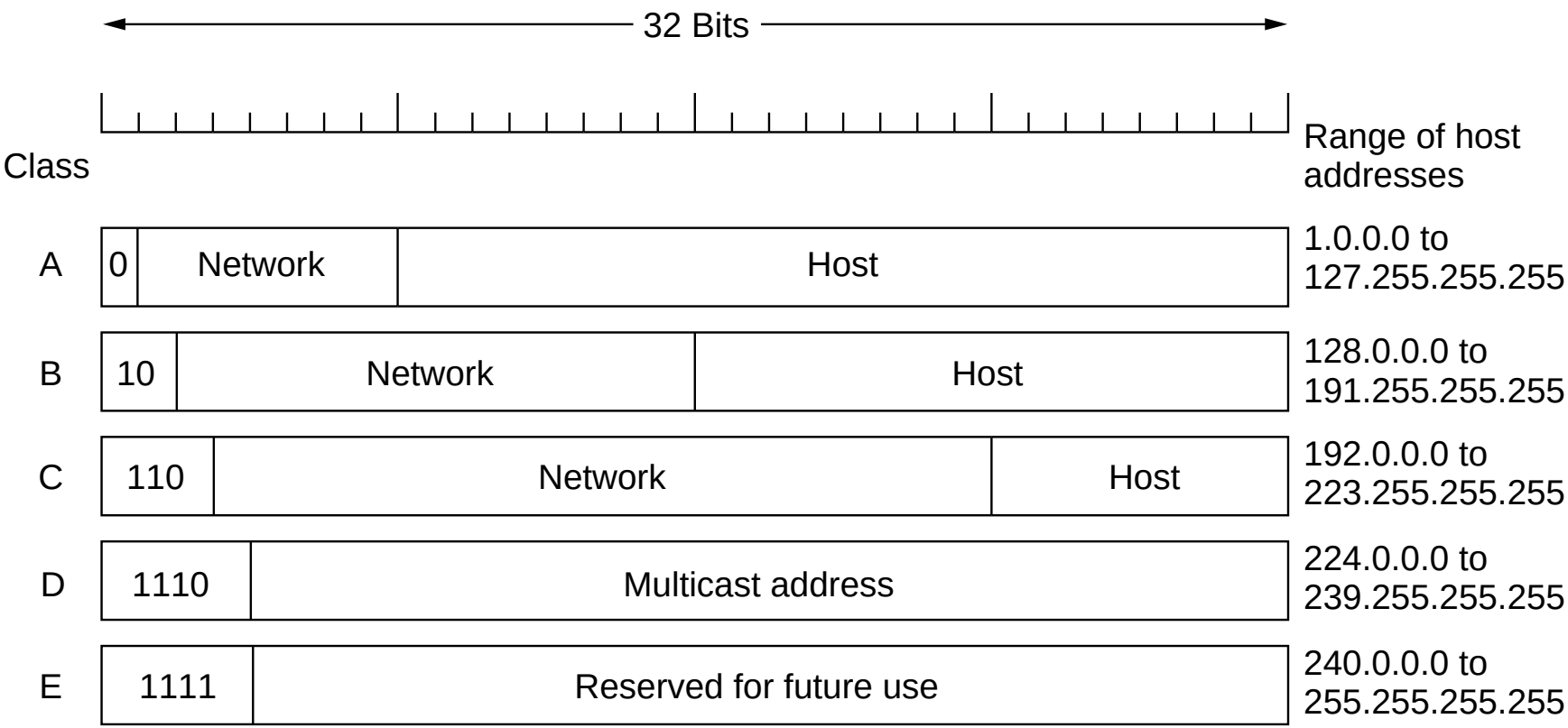
Netzklassen, RFC791

Der IP Adressbereich wurde eingeteilt in Subnetze, die - abhängig von ihrer Adresse - unterschiedliche Größen haben:

Klasse	Präfix	Type	Endadresse
A	0	127 Subnetze, je 24 Bit	127.255.255.255
B	10	16383 Subnetze, je 16 Bit	191.255.255.255
C	110	2097151 Subnetze, je 8 Bit	223.255.255.255
D	1110	Multicast Adressen	239.255.255.255
E	1111	reserviert	255.255.255.255

Beispiel: IP Adresse 192.168.10.1 bezeichnet eine Adresse in einem (privaten, vgl. RFC1918) Class-C Netzwerk.

Netzklassen, RFC791



(c) Tanenbaum, Computer Networks

Classless Inter-Domain Routing, CIDR, RFC1518

Problem:
Die Einteilung in Netzklassen konnte nicht beibehalten werden, da mit dem schnellen Wachstum des Internets die Tabellen in Routern schnell nicht mehr administrierbar waren (für Details siehe RFC1519).

Lösung:
Die Routingentscheidung wird nicht mehr allein anhand der Zieladresse gefällt, sondern zusätzlich mit einer Netzmaske, die die Größe des zugehörigen Netzes angibt.

Die Netzmaske besteht aus 32Bit, vorne 1 für den Präfix, 0 für die Knoten im Netz. Für ein Class-C Netz ergibt sich z.B. die Netzmaske 255.255.255.0.

Schreibweise für 192.168.10.1 sind bei CIDR: 192.168.10.1/24 oder 192.168.10.1/255.255.255.0.

Beispiel CIDR

Für eine Adresse wird anhand der Routingtabelle geprüft, welcher Eintrag paßt, d.h. (Bits der Adresse) AND (Netzmaske) ergibt das Netz. Passen mehrere Einträge, wird derjenige mit der längsten Maske gewählt.

#	Destination	Gateway	Genmask	Iface
1	10.1.195.0	0.0.0.0	255.255.255.0	eth0
2	192.168.42.0	10.1.195.1	255.255.255.0	eth0
3	10.135.228.0	0.0.0.0	255.255.254.0	eth1
4	10.1.0.0	10.1.195.1	255.255.0.0	eth0
5	0.0.0.0	10.135.229.252	0.0.0.0	eth1

10.1.180.33 paßt auf Zeilen 4 und 5, gerouted wird über eth0, nächster Knoten ist 10.1.195.1.

Spezielle Adressbereiche

Adressen	Zweck	Referenz
0.0.0.0/8	Zero Addresses	RFC1700
10.0.0.0/8	Private Adressen	RFC1918
127.0.0.0/8	Loopback Address	RFC1700
169.254.0.0/16	Zeroconf, Link Local	RFC3927
172.16.0.0/12	Private Adressen	RFC1918
192.0.2.0/24	Beispielnetze/Test Domain	RFC3330
192.168.0.0/16	Private Adressen	RFC1918
198.18.0.0/15	Test Netze	RFC2544
224.0.0.0/4	IP Multicast	RFC3171
240.0.0.0/4	Reserviert	RFC1700

Network Address Translation (NAT)

Viel Router haben die Möglichkeit, Adressen im IP Header umzuschreiben.

Ziele sind:

- Rechner in privaten Netzen können Rechner im Internet erreichen
- IP Pakete können nur über diese Router ins Internet
- Eingehende Verbindungen aus dem Internet auf Rechner im privaten Netz funktionieren nur, wenn der NAT-Router entsprechend konfiguriert ist.

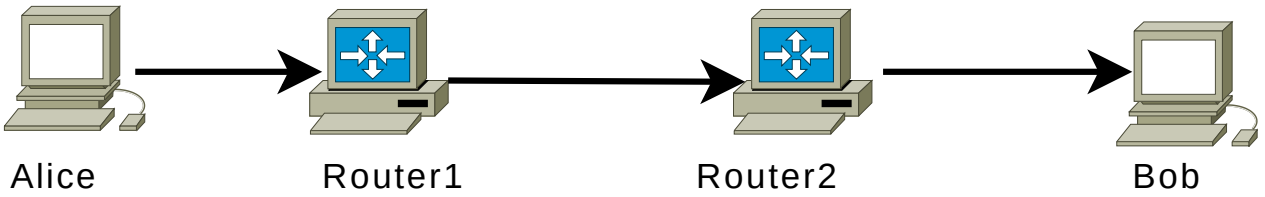
Man unterscheidet:

- **Source NAT:** Die Quelladresse wird verändert
- **Destination NAT:** Die Zieladresse wird verändert
- **Masquerading:** Mehrere Quelladressen werden hinter einer IP (oft der des Routers) versteckt.

Motivation Internet Control Message Protocol (ICMP)

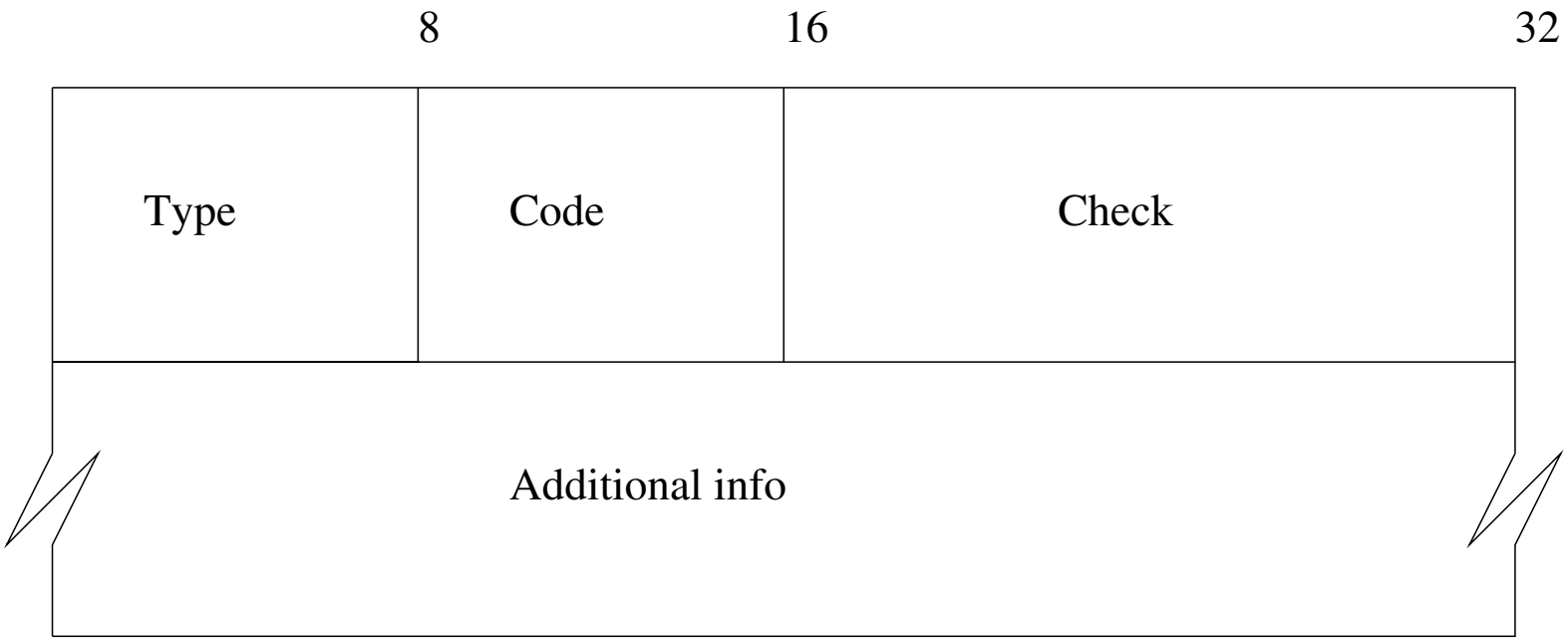
Wie kann ein Host über Probleme bei der Auslieferung von Daten informiert werden?

Beispiel:
Host Alice sendet ein Paket an Host Bob. Bob befindet sich in einem anderen Netzwerk, d.h. ist nicht über Adressen der Sicherungsschicht erreichbar.



Wie wird Alice informiert, wenn Router2 Bob nicht erreichen kann?

ICMP Rahmen



Type Typ der Nachricht, davon hängen die weiteren Felder ab.

Code Untergruppen für den jeweiligen Type

Check Prüfsumme, wie im IP Header berechnet

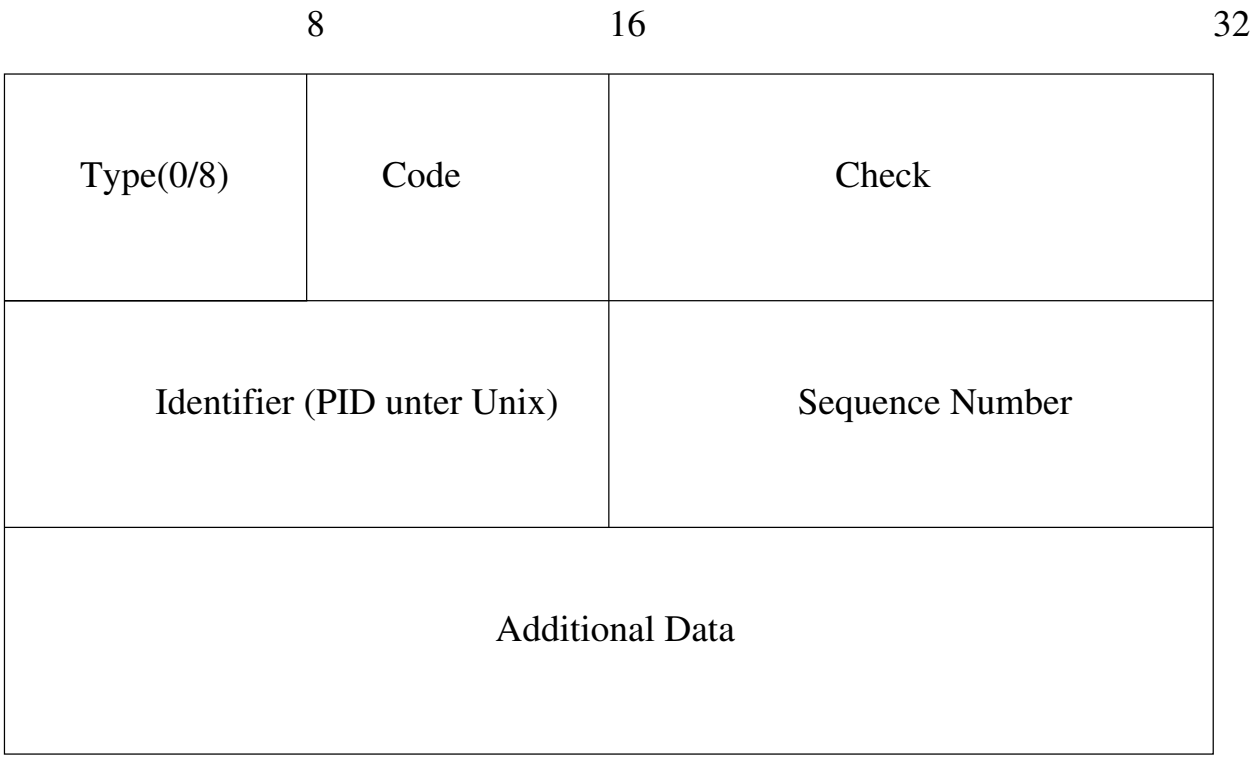
Rest Weitere Daten abhängig von Type und Code

Wichtige ICMP Typen

Typ	Name
0	Echo Reply
3	Destination Unreachable
4	Source Quench
5	Redirect
8	Echo Request
9	Router Advertisement
10	Router Solicitation
11	Time Exceeded
12	Parameter Problem
13	Timestamp Request
14	Timestamp Reply
17	Address Mask Request
18	Address Mask Reply

ICMP Echo Request/Reply (Type 0/8)

Test, ob die Gegenseite auf IP-Ebene erreichbar ist:
Type 8, Code 0 für Request, Type 0, Code 0 für Reply



ICMP Echo Request/Reply

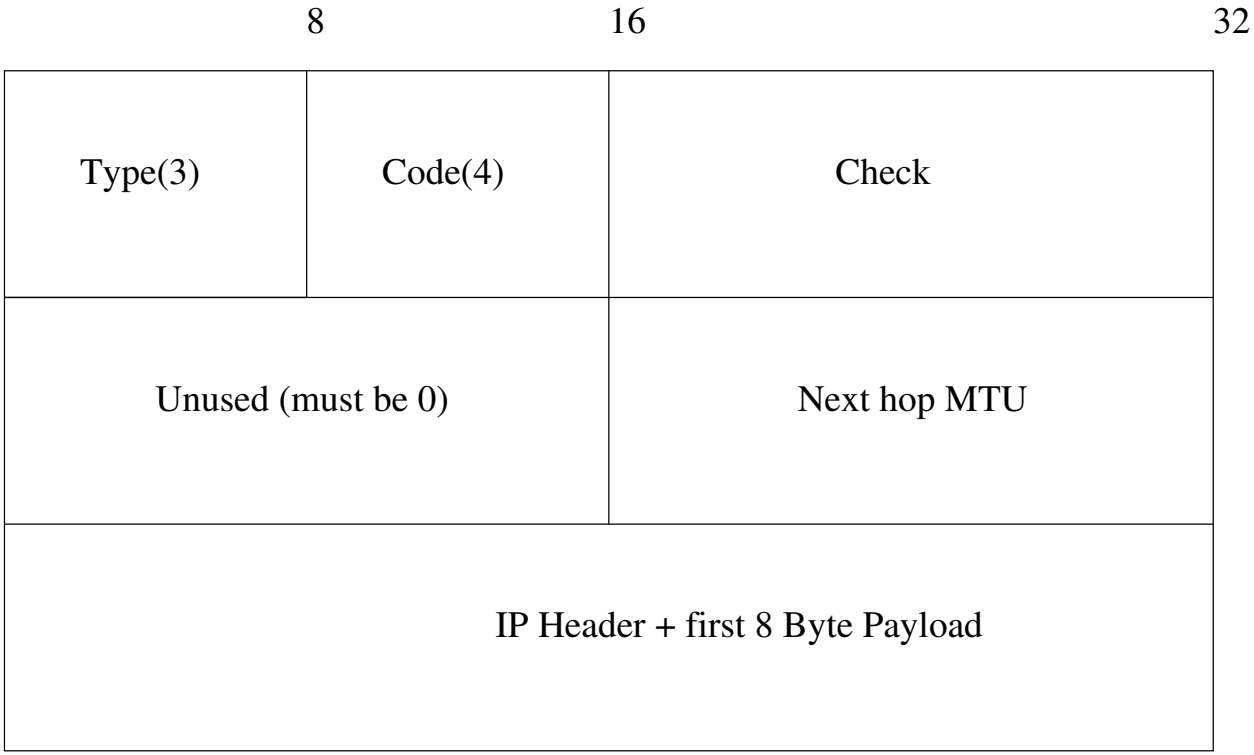
- Ein Host wird einen ICMP Echo Request beantworten, indem er die Felder **Indentifier**, **Sequence Number** und **Additional Data** in die Antwort kopiert.
- **Identifier:** Dieses Feld wird benutzt, damit ein Reply dem Sender des Requests zugeordnet werden kann. Unix benutzt oft die Process ID (Endianess beachten!).
- **Sequence Number:** Fortlaufende Nummer, die benutzt wird, um Request und Reply einander zuzuordnen. **ping** benutzt das zur Laufzeitmessung.
- **Additional Data:** Beliebige Daten, die vom Server zurückgeschickt werden.

ICMP Destination Unreachable (Type 3)

Code	Name
0	network unreachable
1	host unreachable
2	protocol unreachable
3	port unreachable
4	fragmentation needed but don't-fragment bit set
5	source route failed
6	destination network unknown
7	destination host unknown
9	destination network administratively prohibited
10	destination host administratively prohibited
11	network unreachable for TOS
12	host unreachable for TOS
13	communication administratively prohibited by filtering

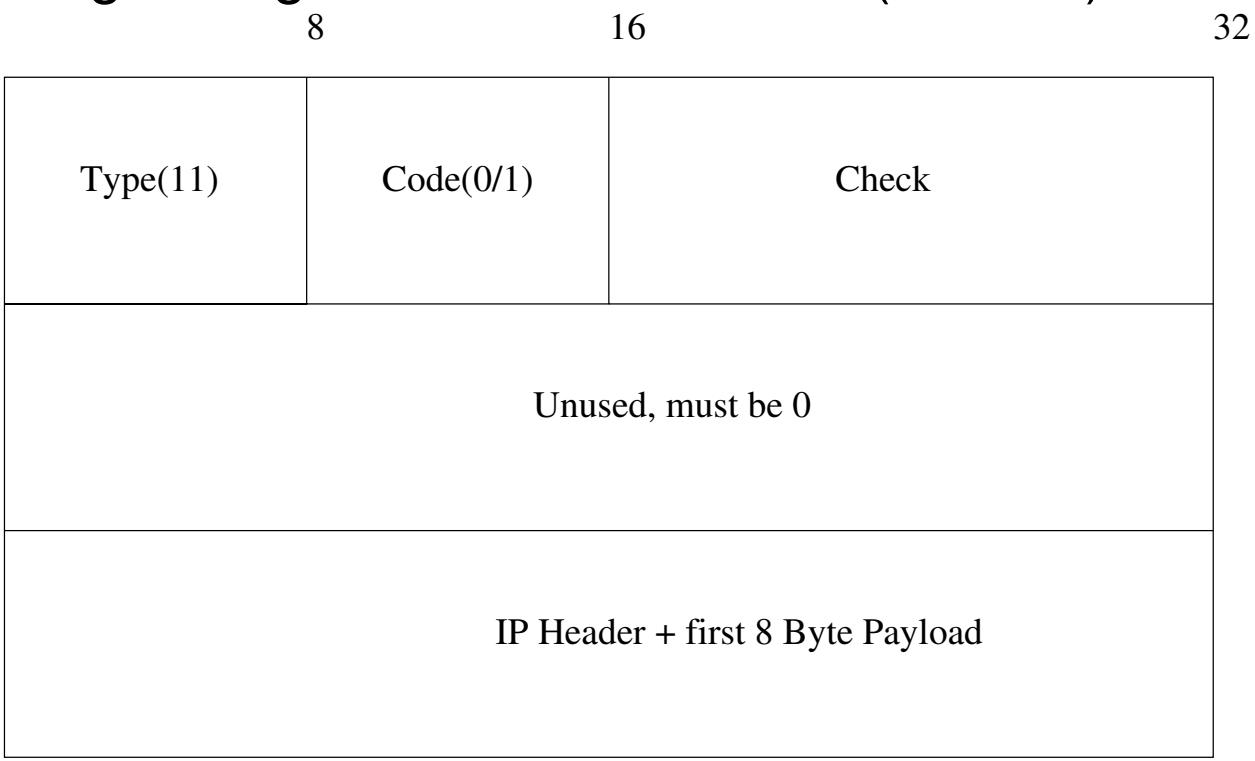
Beispiel: Fragmentation Required

Fragmentation Required (Code 4) wird erzeugt, wenn die MTU des nächsten Links nicht ausreicht, um das IP Paket zu übertragen, aber DF gesetzt ist.



ICMP Time Exceeded (Type 11)

Time Exceeded wird erzeugt, wenn in einem Router der Wert des TTL Feldes auf 0 fällt (Code 0), oder zu lange auf ein Fragment gewartet werden muß (Code 1).



Traceroute, Van Jacobson, 1988

- Programm zu Bestimmung der Route zu einem Zielhost
- Windows Implementation (tracert) benutzt ICMP Echo Request (Unix Impementationen benutzen UDP)
- **Algorithmus:**

```
Setze TTL = 1
Bis ICMP Echo Reply empfangen wurde:
  Sende drei ICMP Echo Request
  Gebe Laufzeit bis zu den Antworten aus
  TTL = TTL + 1
```

tracert Beispiel

Die Ausgabe von **tracert** von einem Windows Host:

```
C:\>tracert -d 10.1.180.33

Tracing route to 10.1.180.33
1 <10 ms <10 ms <10 ms 172.16.199.1
2 <10 ms <10 ms 16 ms 10.1.195.2
3 <10 ms <10 ms <10 ms 10.1.180.33

Trace complete.
```

tracert Netzwerktrace

Jeder der drei Dialoge wurde je dreimal aufgezeichnet mittels `tcpdump -ttt -vn`:

```
000000 IP (ttl 1) 172.16.199.122 > 10.1.180.33:
  ICMP echo request, id 512, seq 10240, length 72
000196 IP (ttl 64) 172.16.199.1 > 172.16.199.122:
  ICMP time exceeded in-transit, length 100

000000 IP (ttl 2) 172.16.199.122 > 10.1.180.33:
  ICMP echo request, id 512, seq 11520, length 72
001224 IP (ttl 254) 10.1.195.2 > 172.16.199.122:
  ICMP time exceeded in-transit, length 36

000000 IP (ttl 3) 172.16.199.122 > 10.1.180.33:
  ICMP echo request, id 512, seq 12032, length 72
000204 IP (ttl 253) 10.1.180.33 > 172.16.199.122:
  ICMP echo reply, id 512, seq 12032, length 72
```

Routing

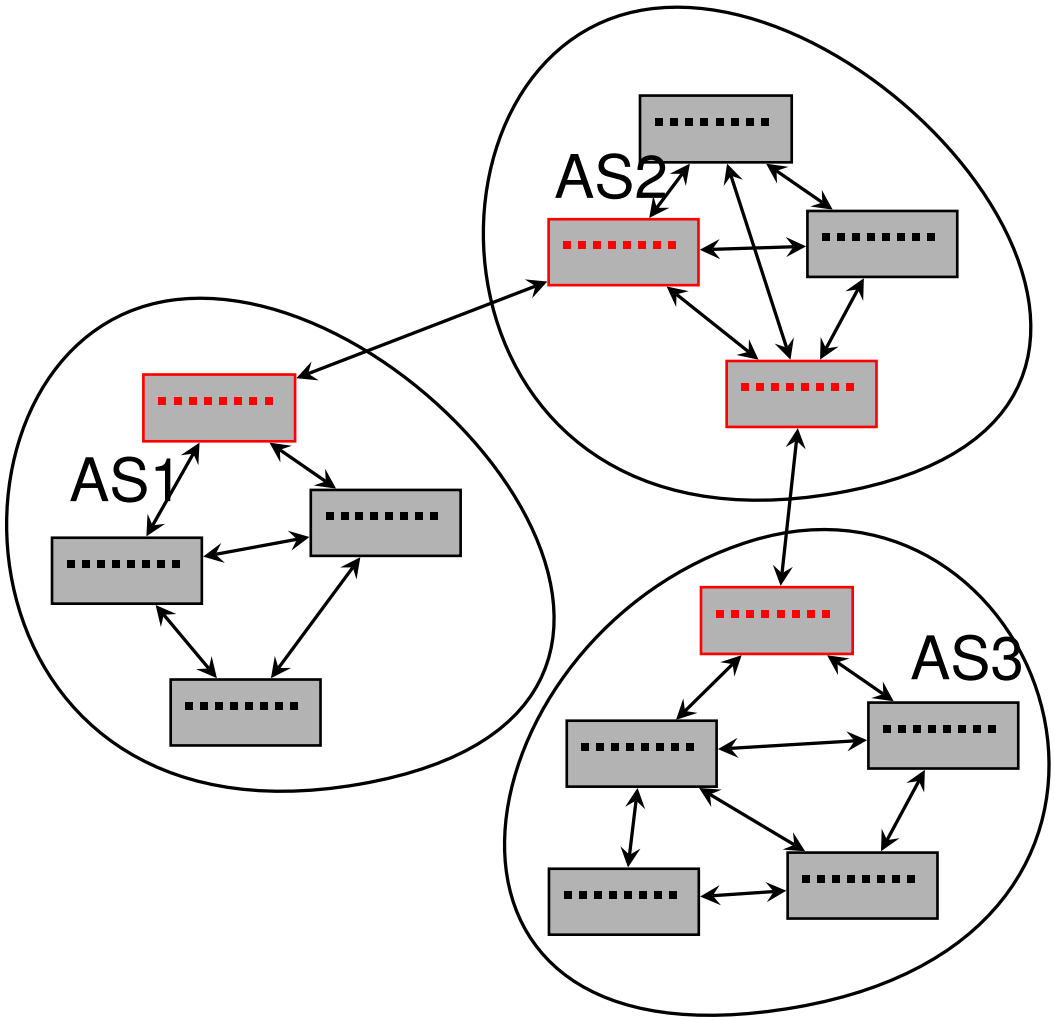
Grundlagen

- Das Internet gliedert sich in Bereiche unterschiedlicher administrativer Verantwortung (z.B. Veratwortung eines ISPs), sog. **autonome Systeme** (**AS**).
- Es muß unterschieden werden zwischen Routing innerhalb eines AS (intra-AS) und zwischen verschiedenen AS (inter-AS), da hier unterschiedliche Anforderung an den Austausch von Routinginformation gestellt werden.
- Die Knoten an den Verbindungsstellen zwischen AS heißen **Gateway Router**.

Anforderungen an Routingprotokolle

- **Skalierbakeit:** Besonders ein inter-AS Routingprotokoll muß mit wachsender Anzahl AS skalieren können, da es keine Möglichkeit gibt, die Anzahl zu kontrollieren/reduzieren.
- **Leistungsfähigkeit:** Das Protokoll sollte in der Lage sein, schnell auf Änderungen in der Netzstruktur zu reagieren, administrative Vorgaben in die Routenplanung zu integrieren und gute Routen zu generieren.
- **Flexibilität:** Das Protokoll muß in der Lage sein, administrative Vorgaben in die Planung einzubauen. Besonders wichtig ist das beim inter-AS Routing, da z.B. keine/unterschiedliche vertragliche Beziehung zwischen den Betreibern einzelner AS bestehen.

Beispielnetz



Bezeichnungen aus der Graphenthorie

- **Graph:** Ein 2-Tupel $G = (V, E)$ ist ein (ungerichteter) Graph, falls $V \neq \emptyset$ und $E = \{ \{v_1, v_2\} \mid v_1, v_2 \in V\}$.
- **Gewichteter Graph:** Ein Graph $G = (V, E)$ mit einer Funktion $c : E \rightarrow \mathbb{R}$ heißt gewichteter Graph.
- **adjazent:** Zwei Ecken $v_1, v_2 \in V$ heißen adjazent, falls $\{v_1, v_2\} \in E$.
- **zusammenhängend:** Ein Graph heißt zusammenhängend, falls es zu jedem Paar $w_1, w_2 \in V$ eine Folge $v_1, \ldots, v_n \in V$ gibt, so daß v_i, v_{i+1} $\forall i$ und w_1, v_1 bzw. v_n, w_2 adjazent sind.

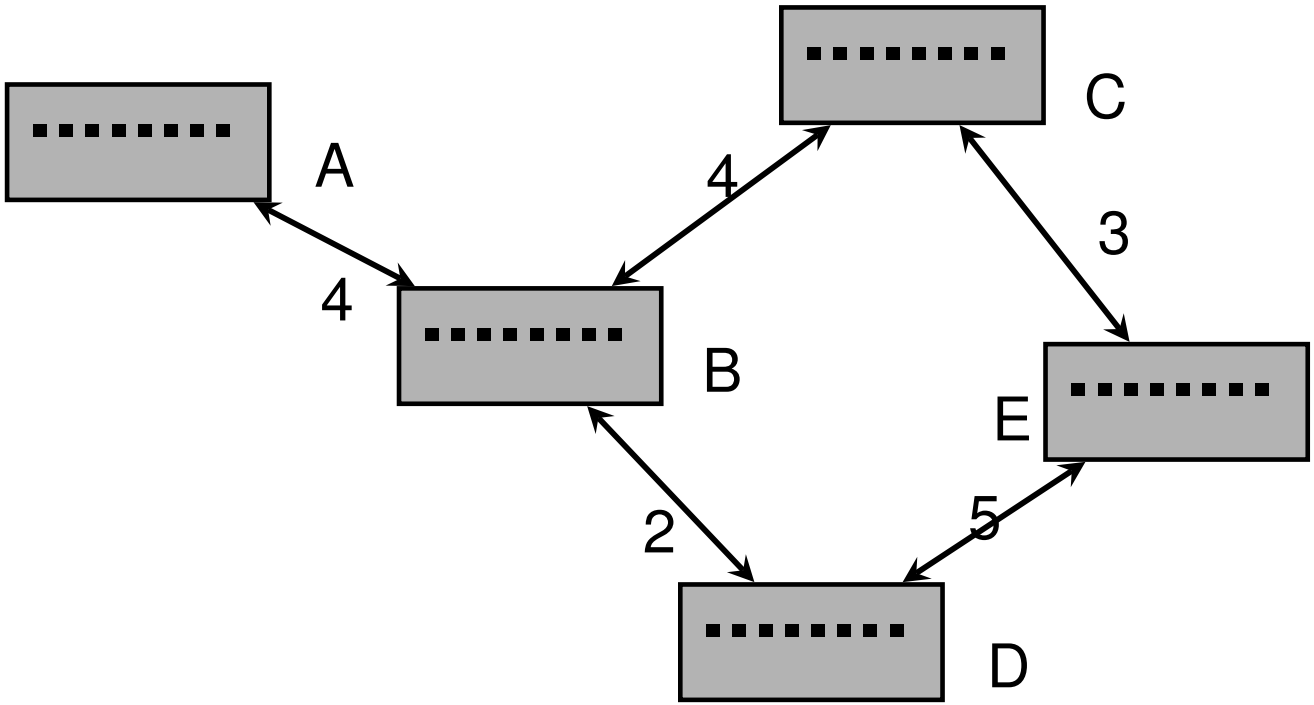
V ist die Menge der Ecken (Vertices) eines Graphen, E die Menge der Kanten. Die Funktion c ordnet jeder Kante $e \in E$ ein Gewicht zu. Oft läßt sich das Gewicht als Kosten oder Distanz interpretieren.

Übersicht über Routingverfahren

- **Distance Vector Algorithms:** Benutzt den “distributed Bellman Ford Algorithmus”, hat Probleme in großen Netzen. Kommunikation ist nur mit unmittelbaren Nachbarn notwendig.
- **Link State Algorithms:** Jeder Router bildet einen gewichteten Graph des gesamten Netzes. Die Routingtabelle ergibt sich durch den Shortest Path Algorithm. Das Verfahren hat Probleme in großen Netzen und erzeugt hohe Netz- und CPU-Last.
- **Path Vector Protocol:** Ähnlich den Distance Vector Algorithms, aber es werden nur ausgewählte Hosts (Speaker Nodes) einbezogen.

Beispiel eines AS

- 5 Router innerhalb des autonomen Systems
- 5 Links mit unterschiedlichen Kosten



Bellmann-Ford Algorithmus: Initialisierung

Wir betrachten ungerichteten Graphen $G = (V, E)$ mit Gewicht c. Ziel ist, zu jeder Ecke v die minimale Distanz $D_v(w)$ zu jeder anderen Ecke w zu bestimmen.

Initialisierung:

$$D_v(w) := \begin{cases} 0 & \text{falls } v = w \\ c(\{v, w\}) & \text{falls } v, w \text{ adjazent} \\ \infty & \text{sonst} \end{cases}$$

$D_w(y) := \infty$ für alle $w \in V,\ v, w$ adjazent. (Initialisierung der möglich Distanzvektoren der Nachbarn von v)

Sende Distanzvektor $\mathbf{D}_v := (D_v(y),\ y \in V)$ an alle Nachbarn von v.

Bellmann-Ford Algorithmus: Hauptschleife

Der Algorithmus besteht aus drei Schritten:

1. Empfange Distanzvektor(en) $\mathbf{D}_w$ von Nachbar(n) w.
2. Für alle $y \in V$ setzte
$$D_v(y) := \min_{w \in V} \left\{ c(\{v, w\}) + D_w(y) \right\}$$
3. Falls $\mathbf{D}_v$ verändert wurde, sende $\mathbf{D}_v$ an alle Nachbarn.

Wann immer ein Distanzvektor von einem Nachbarn w empfangen wird, prüfe, ob sich dadurch ein besserer Weg zu einem der Knoten im Netz ergibt, d.h. ist die Summe

- Weg nach w
- Weg von w zum Ziel

günstiger als der bisher bekannte Weg.

Bellmann-Ford im Beispiel AS

	A	B	C	D	E
A	0	**4**	∞	∞	∞
B	**4**	0	**4**	**2**	∞
C	∞	**4**	0	∞	**3**
D	∞	**2**	∞	0	**5**
E	∞	∞	**3**	**5**	0

C sendet Distanzvektor an B und E:

	A	B	C	D	E
A	0	**4**	∞	∞	∞
B	**4**	0	**4**	**2**	$7(C)$
C	∞	**4**	0	∞	**3**
D	∞	**2**	∞	0	**5**
E	∞	7(C)	**3**	**5**	0

Bellmann-Ford im Beispiel AS

B sendet Distanzvektor an A,C,D:

	A	B	C	D	E
A	0	**4**	8(B)	6(B)	11(B)
B	**4**	0	**4**	**2**	7(*C*)
C	8(B)	**4**	0	6(B)	**3**
D	6(B)	**2**	6(B)	0	**5**
E	∞	7(C)	**3**	**5**	0

D sendet Distanzvektor an B,E:

	A	B	C	D	E
A	0	**4**	8(B)	6(B)	11(B)
B	**4**	0	**4**	**2**	7(*C*)
C	8(B)	**4**	0	6(B)	**3**
D	6(B)	**2**	6(B)	0	**5**
E	11(D)	7(C)	**3**	**5**	0

Bellmann-Ford Algorithmus: Count-to-Infinity Problem

Distance Vector Algorithmen können sehr langsam konvergieren.

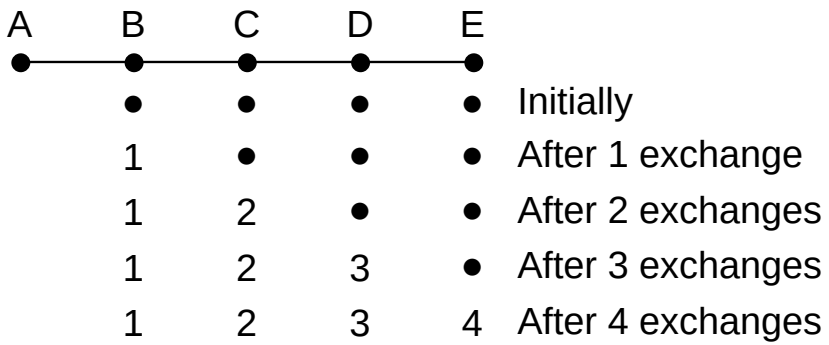
- Gute Nachrichten breiten sich schnell aus
- Schlechte Nachrichten breiten sich langsam aus

Beispiel:
5 Knoten A,B,C,D,E linear vernetzt
Distanz jeweils 1

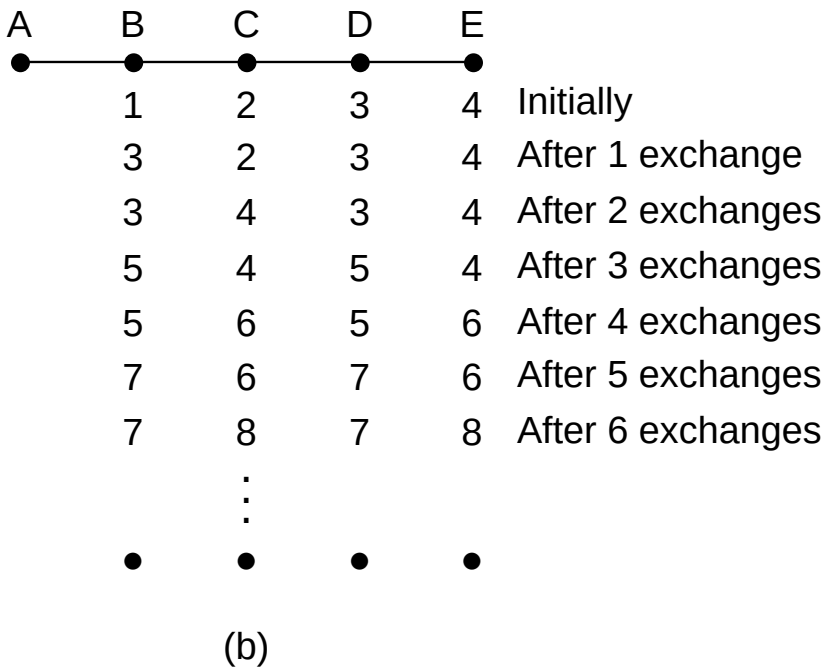
- a) Knoten A wird eingeschaltet
- b) Knoten A wird ausgeschaltet

Bellmann-Ford Algorithmus: Count-to-Infinity Problem

Angegeben ist die Distanz zu Knoten A.



(a)



(b)

Mögliche Lösung:
Abbruch wenn Distanz größer als längster Pfad + 1.

Dijkstra Algorithmus

Seien G und c wie im Bellmann-Ford Algorithmus. Sei ferner $u \in V$ eine Ecke des Graphen.

Initialisierung:

Setze $N := \{u\}$ und

$$D(v) := \begin{cases} 0 & \text{falls } u = v, \\ c(\{u, v\}) & \text{falls } u, v \text{ adjazent}, \\ \infty & (\textit{sonst}) \end{cases}$$

Dijkstra Algorithmus

do
finde $w \in V \setminus N$, so daß $D(w) \leq D(w') \; \forall \; w' \in V \setminus N$
$N := N \cup \{w\}$
Für alle v mit $v \in V \setminus N$, v, w adjazent bilde
$D(v) := \min\{D(v), D(w) + c(\{v, w\})\}$
while $N \neq V$

- Während des Ablaufes gibt $D(v)$ stets für alle $v \in V$ die beste bis dahin gefundene Distanz zwischen u und v an.
- N enthält die Ecken, für die das Ergebnis schon feststeht.
- Initial ist $N = \{u\}$ mit Distanz $D(u) = 0$.
- In jedem Schritt wird die nächstgelegene Ecke außerhalb N betrachtet und geprüft, ob Routing über diese Ecke den Weg zu anderen Ecken verkürzt.

Dijkstra Algorithmus im Beispiel AS

Wir betrachten Ecke A als Ausgangspunkt, in der Tabelle sind für jeden Schritt die aktuellen Werte von $D(v)$ für $v \in \{ABCDE\}$ aufgetragen.

Schritt	A	B	C	D	E	N
1	0	4(B)	∞	∞	∞	$\{A\}$
2	0	4(B)	8(B)	6(B)	∞	$\{A, B\}$
3	0	4(B)	8(B)	6(B)	11(B)	$\{A, B, D\}$
4	0	4(B)	8(B)	6(B)	11(B)	$\{A, B, C, D\}$
5	0	4(B)	8(B)	6(B)	11(B)	$\{A, B, C, D, E\}$

Dijkstra Algorithmus im Beispiel AS

Entsprechend für Ecke C als Ausgangspunkt:

Schritt	A	B	C	D	E	N
1	∞	4(B)	0	∞	3(E)	$\{C\}$
2	∞	4(B)	0	8(E)	3(E)	$\{C, E\}$
3	8(B)	4(B)	0	6(B)	3(E)	$\{B, C, E\}$
4	8(B)	4(B)	0	6(B)	3(E)	$\{B, C, D, E\}$
5	8(B)	4(B)	0	6(B)	3(E)	$\{A, B, C, D, E\}$

Broadcast Routing

Broadcast:
Übertragen einer Nachricht an alle Empfänger simultan

Mögliche Broadcast Routing Verfahren:

- Übertragen getrennter Nachrichten an jeden Empfänger
  - sehr einfach
  - Verschwendung von Bandbreite
  - alle Adressen müsen bekannt sein
- **Flodding:** Jeder Router sendet ein ankommendes Paket an alle Nachbarn, außer dem vorherigen Sender
  - Flodding generiert sehr viele Duplikate
  - Maßnahmen zur Begrenzung notwendig
  - sehr robust
  - benutzt immer den kürzesten Weg (mögliche Referenz)

Broadcast Routing

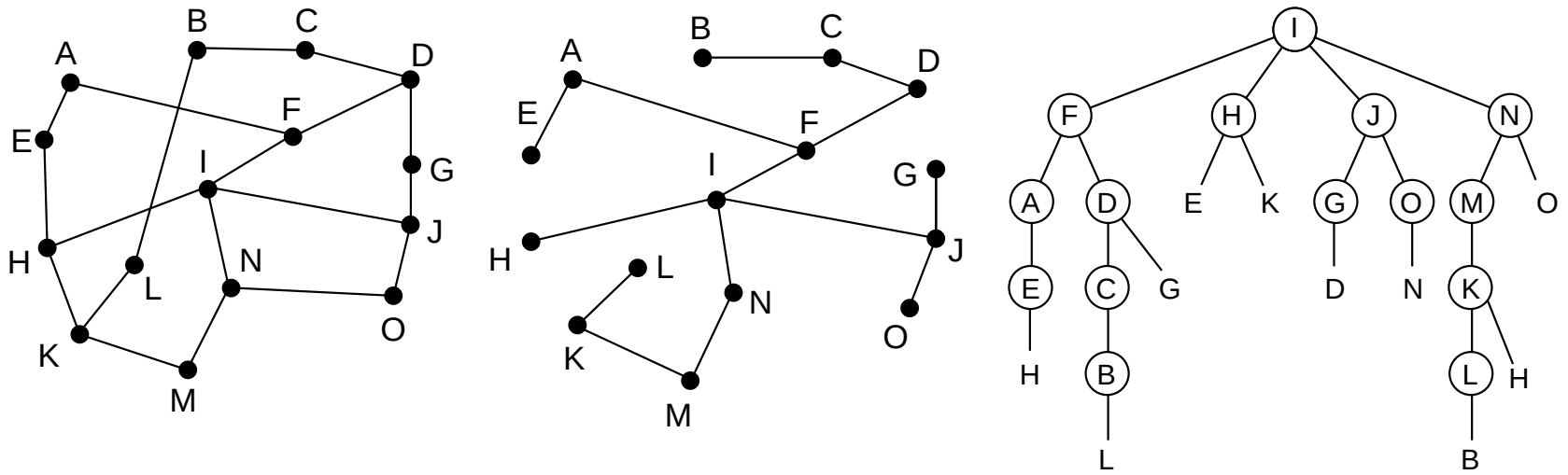
Mögliche Broadcast Routing Verfahren (Fortsetzung):

- **multidestination routing:** Jedes Paket enthält eine Liste mit seinen Zielen. Jeder Router prüft diese Liste und versendet Kopien mit angepaßter Liste an die notwendigen Nachbarn.
  - Wie “separates Senden”, nur das parallele Pakete zusammengefaßt werden
- **reverse path forwarding:** Ein Router prüft für ein ankommendes Paket ob es auf der Leitung ankommt, die üblicherweise zum Senden an die Quelladresse des Broadcast benutzt wird.
  - ankommende Leitung verschieden: Paket verwerfen
  - ankommende Leitung stimmt überein: Paket weiterleiten

Broadcast Routing: Reverse Path Forwarding

Beispiel:

- a) Gesamtes (Sub-)Netz mit Knoten A...O
- b) Baum von Knoten I
- c) Baum von Knoten I für Reverse Path Forwarding



(c) Tanenbaum, Computer Networks

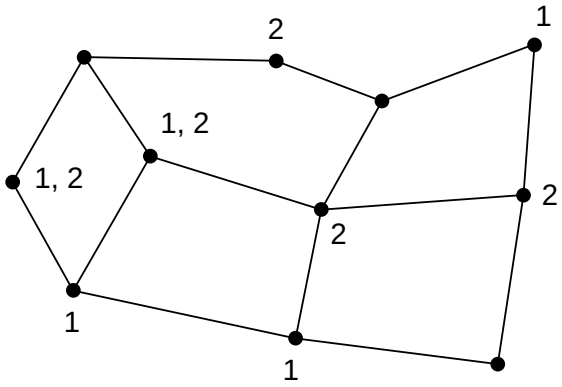
Multicast Routing

Multicast: Senden einer Nachricht an eine definierte Gruppe

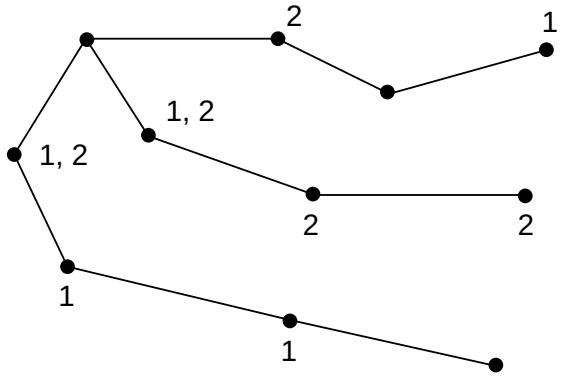
- sehr kleine Gruppe → einzelnes Routing für jeden Empfänger
- sehr große Gruppe → Broadcast Routing

Multicast Routing:

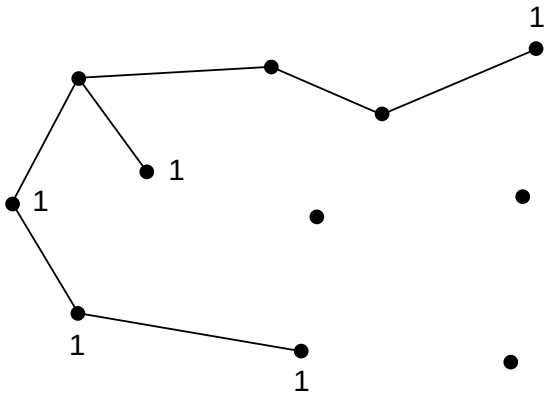
- erfordert Management der Gruppen durch die Router
- Router müssen über neue Gruppen informiert werden
- jeder Router berechnet Baum für jede Gruppe (hoher Speicheraufwand)
- jeder Router leitet Pakete nur an die Knoten weiter, die Teil des Baums der Gruppe sind (verkürzter Baum)

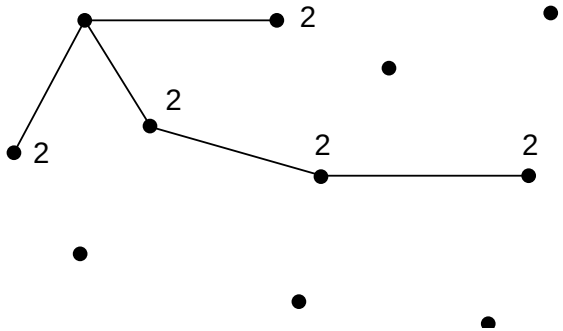


(a)



(b)





a) gesamte Netzwerk
b) Baum des linken Knoten
c) Baum für Multicast Gruppe 1
d) Baum für Multicast Gruppe 2

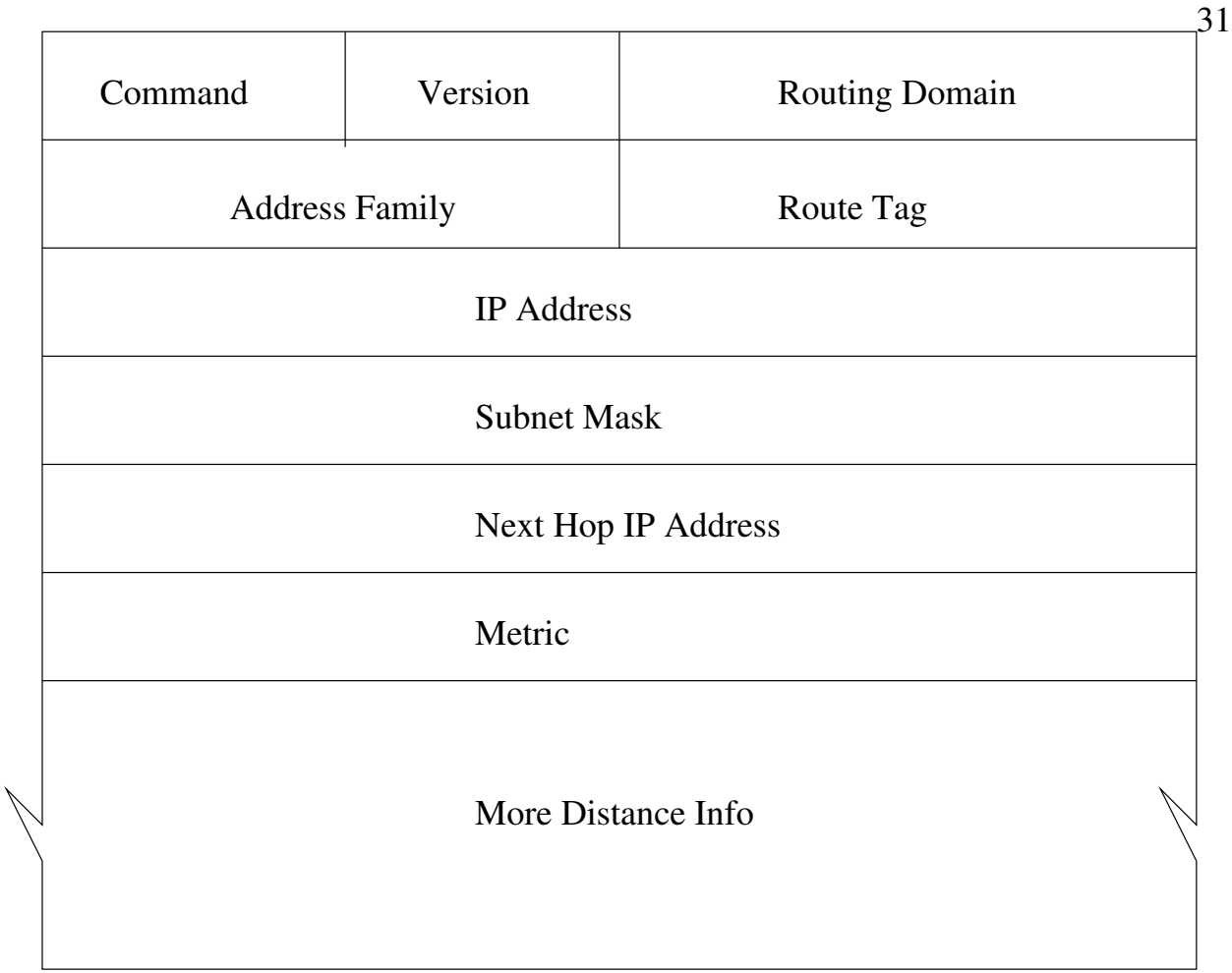
(c) Tanenbaum, Computer Networks

Beispiel: Routingtabelle des lokalen Rechners

Windows prompt `>route print`

```
===========================================================================
Schnittstellenliste
11 ...00 1c bf XY ZX YZ ...... Intel(R) PRO/Wireless 3945ABG Network Connec
10 ...00 16 d3 XY ZX YZ ...... Intel(R) 82566MM Gigabit Network Connection
1 ........................... Software Loopback Interface 1
21 ...00 00 00 00 00 00 00 e0 Microsoft-ISATAP-Adapter #2
23 ...00 00 00 00 00 00 00 e0 Microsoft-6zu4-Adapter
14 ...02 00 54 XY ZX YZ ...... Teredo Tunneling Pseudo-Interface
===========================================================================
IPv4-Routentabelle
===========================================================================
Aktive Routen:
     Netzwerkziel    Netzwerkmaske          Gateway    Schnittstelle  Metrik
          0.0.0.0          0.0.0.0    134.130.XX.YY    134.130.XX.YYY     10
          0.0.0.0          0.0.0.0      134.61.XX.Y       134.61.YY.Z     25
        127.0.0.0        255.0.0.0   Auf Verbindung         127.0.0.1    306
        127.0.0.1  255.255.255.255   Auf Verbindung         127.0.0.1    306
  127.255.255.255  255.255.255.255   Auf Verbindung         127.0.0.1    306
      134.61.XX.Z    255.255.248.0   Auf Verbindung       134.61.XX.Z    281
      134.61.XX.Z  255.255.255.255   Auf Verbindung       134.61.XX.Z    281
    134.61.YY.ZZZ  255.255.255.255   Auf Verbindung       134.61.XX.Z    281
    134.130.XX.XX  255.255.255.240   Auf Verbindung    134.130.XX.ZZZ    266
   134.130.XX.ZZZ  255.255.255.255   Auf Verbindung    134.130.XX.ZZZ    266
   134.130.XX.XYZ  255.255.255.255   Auf Verbindung    134.130.XX.ZZZ    266
        224.0.0.0        240.0.0.0   Auf Verbindung         127.0.0.1    306
...
  255.255.255.255  255.255.255.255   Auf Verbindung         127.0.0.1    306
```

RIP Version 2 (vgl. RFC2453)



RIP Version 2

RIP Version 2 ist ein Distance Vector basiertes Verfahren.

- **Command:** Entweder Request(1) oder Reply(2)
- **Version:** RIP Version 2 (RIP Version 1 unterstützt kein CIDR, das Rahmenformat ist gleich)
- **Routing Domain:** Indentifier für RIP process (in Version 2)
- **Address Family:** Für IP immer 2
- **Route Tag:** Tag zur Identifikation des AS (ASN, RFC1930)
- **IP Address:** Addresse des Subnetzes, zu dem die Information gehört
- **Subnet Mask:** Netzmaske des Subnetzes
- **Next Hop IP Address:** IP Addresse des Routers, über den das Subnetz erreicht werden kann
- **Metric:** Kosten, bei RIP Anzahl Hops zum Ziel, max. 15
- Pro Rahmen bis zu 25 Distanzinformationen je 20 Byte.

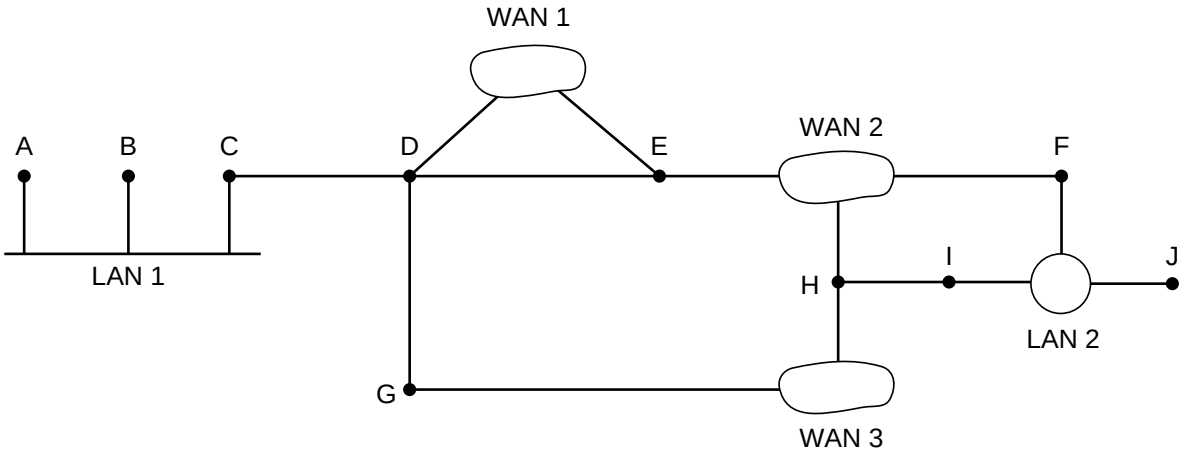
RIP Version 2

- Bei Start schickt der Router einen RIP2 Request mit Address Family 0 (statt 2) auf allen Interfaces. Dieser fordert von allen Routern den kompletten Satz Distanzvektoren an.
- In einem Request mit Address Family 2 wird jeder Eintrag im Rahmen bearbeitet. Falls eine Route vorhanden ist, setze Metric, andernfalls setze Metric auf 16 (unendlich).
- Wird ein Reply empfangen, verwende den Distance Vector Algorithm zum Neuaufbau der Routingtabelle.
- Alle 30 Sekunden wird die komplette Routingtabelle an alle Nachbarn verschickt.
- Bei jeder Veränderung der eigenen Routingtabelle werden die Änderungen der Metric übertragen.
- Jeder Routingeintrag verfällt nach max. 2 Minuten, wenn er nicht erneuert wird.

Open Shortest Path First (vgl. RFC2328)

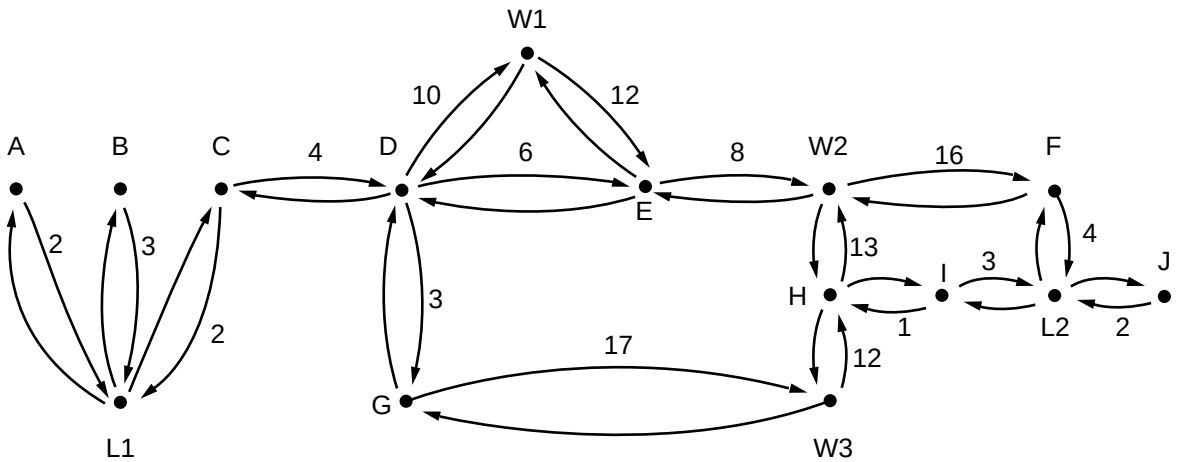
- OSPF Daten werden im IP Rahmen übertragen (Protocol 89)
- OSPF benutzt den Link State Algorithmus zur Berechnung der Routingtabelle.
- Anwendung im Intra-AS Routing
- Bei OSPF werden die Linkstati eines Routers zu allen Routern des AS übertragen.
- Jeder Router berechnet mit dem Dijkstra Algorithmus die Route geringster Kosten zu allen anderen Routern.
- Links werden mit OSPF Paketen auf Funktionalität geprüft.

Beispiel OSPF Graph



(a)

Beispiel AS



(b)

Beispiel Graph

(c) Tanenbaum, Computer Networks

Eigenschaften von OSPF

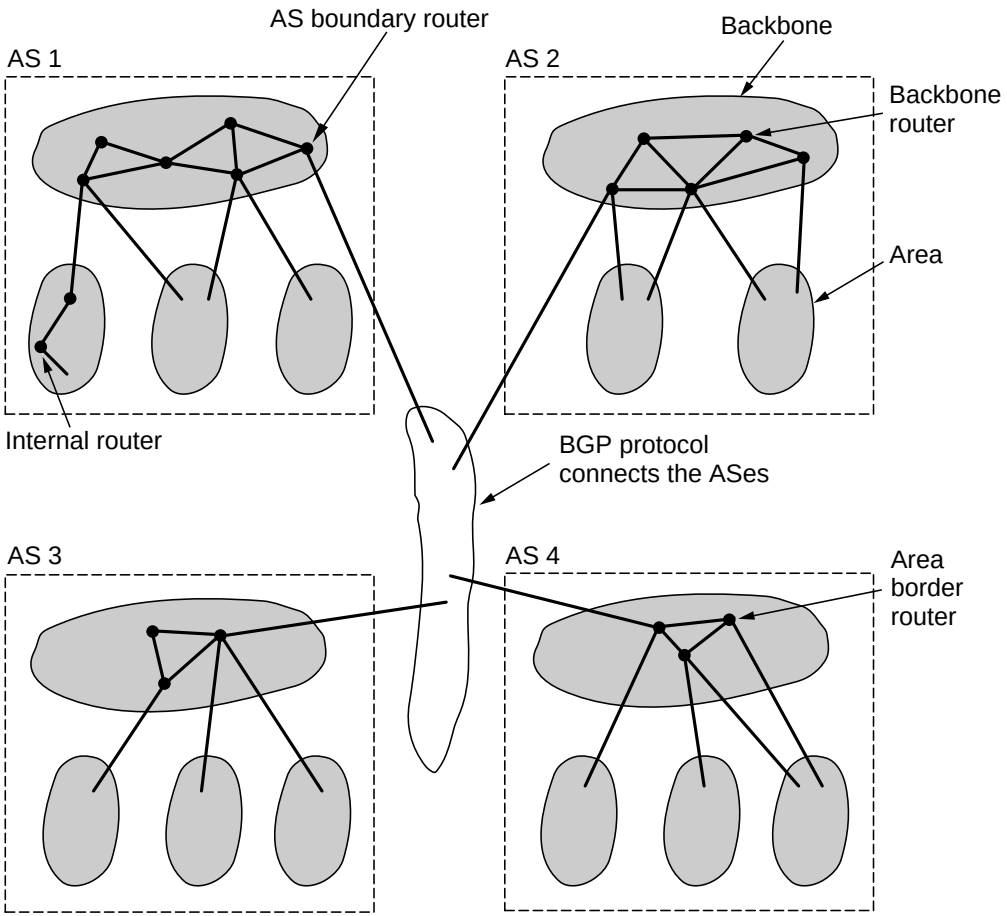
- OSPF unterstützt echte Authentifizierung.
- Mehrere Pfade mit gleichen Kosten können parallel benutzt werden (Load Balancing).
- OSPF unterstützt verschiedene Routen abhängig vom TOS Feld.
- Kosten eines Links sind dimensionslos, unterschiedliche Kosten können für unterschiedliche Werte des TOS Feldes benutzt werden.
- Routen müssen nicht durch IP Adressen identifiziert werden (vgl. PPP)
- OSPF erlaubt eine hierarchische Aufteilung des AS in kleinere AS (“Areas”), die durch **Area Border Router** mit dem **Backbone AS** verbunden sind.

Routertypen von OSPF

Vier Typen von Routern existieren in OSPF

- Internal Router: Router in einer Area, die nicht mit dem Backbone verbunden sind
- Area Border Router: Verbinden Area und Backbone
- Backbone Router: Interne Router im Backbone
- Boundary Router: Verbindung zu anderen AS

Beispiel OSPF Hierarchie



(c) Tanenbaum, Computer Networks

Border Gateway Protocol Version 4 (vgl. RFC4271)

- Beispiel eines Path-Vector Protokolls
- Dient als Inter-AS Routing Protokoll im Internet
- BGP4 muß von jedem ISP eingesetzt werden, um Routing zu anderen AS zu unterstützen.
- In Benutzung seit 1994 (Version 4 unterstützt im Gegensatz zu Version 3 CIDR)
- Routingentscheidungen zwischen AS basieren nicht auf Kosten sondern auf Regeln.

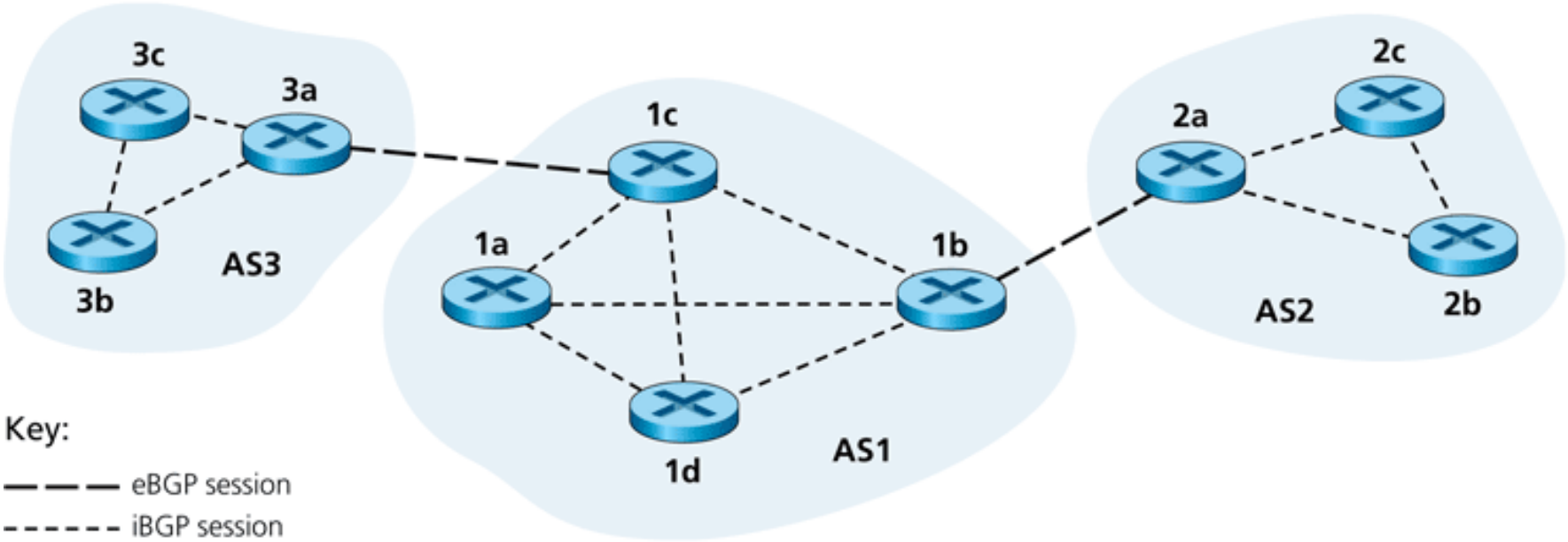
Bezeichnungen

- **BGP Speaker:** Knoten, der das BGP Protokoll implementiert.
- **eBGP Kommunikation:** BGP Datenaustausch zwischen AS
- **iBGP Kommunikation:** Datenaustausch zwischen BGP Speaker und seinem AS
- **Präfix:** Netzwerkidentifikation bestehend aus IP und Maske, z.B. 134.130.35.128/25
- **BGP Attribut:** Zusätzliche Information zu einem Präfix, besonders wichtig sind die Attribute AS-Path und Next-Hop
- **Route:** Kombination von Präfix und zugehörigen Attributen

Generelle Arbeitsweise

- Jede Router hat eine Liste seiner Nachbarn, zu denen er eine Verbindung aufbaut.
- Verbindungen werden durch einen Keepalive Mechanismus permanent überwacht.
- Gateway Router tauschen via eBGP Kommunikation die verwendeten Routen mit ihren Nachbarn aus.
- Router innerhalb des AS informieren sich gegenseitig über die verwendeten Routen via iBGP Kommunikation
- Aus den möglichen Routen wählt jeder Router anhand seiner Konfiguration die für ihn beste aus.

Beispiel BGP Sessions



(c) Kurose and Ross, Computer Networking

Routenerzeugung

- Wird ein Präfix von einem Router via eBGP an einen anderen Router gemeldet, fügt er seine AS Nummer ASN (vgl. RFC1930) an das AS-Path Attribut an und setzt das Next-Hop Attribut auf die IP Adresse des externen Interfaces.
- Jeder Router innerhalb eines AS wird eine Route zum Präfix über den Next-Hop berechnen. Dazu wird ein Intra-AS Routingverfahren (z.B. OSPF) verwendet.

Routenselektion

Ein Router kann verschiedene Routen zum selben Ziel über verschiedene BGP Speaker empfangen.

- Kann der Next-Hop nicht erreicht werden, verwerfe den Pfad
- Wähle die Route mit der höchsten Präferenz (wird über Regeln vom Administrator festgelegt).
- Bei gleicher Präferenz, wähle die kürzeste Route, d.h. mit dem kürzesten AS-Path Attribut.
- Unter den verbliebenen Routen, wähle die mit den günstigsten Kosten zum Next-Hop.