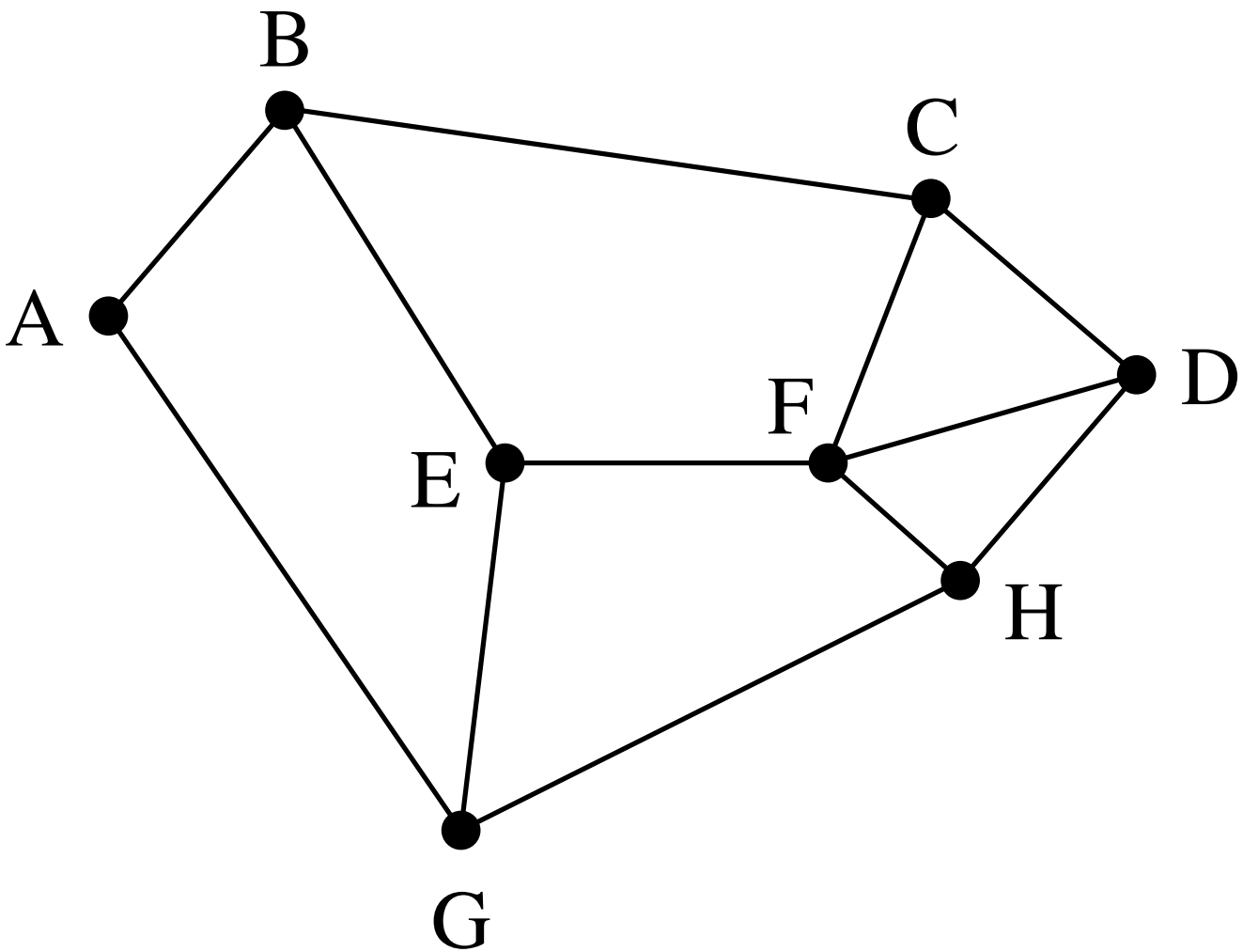
9. Vermittlungsschicht (Network Layer)

Bisher: Wir können nur Nachrichten zwischen direkt verbundenen Rechnern schicken: Entweder Punkt-zu-Punkt verbunden oder Broadcastmedium im LAN.

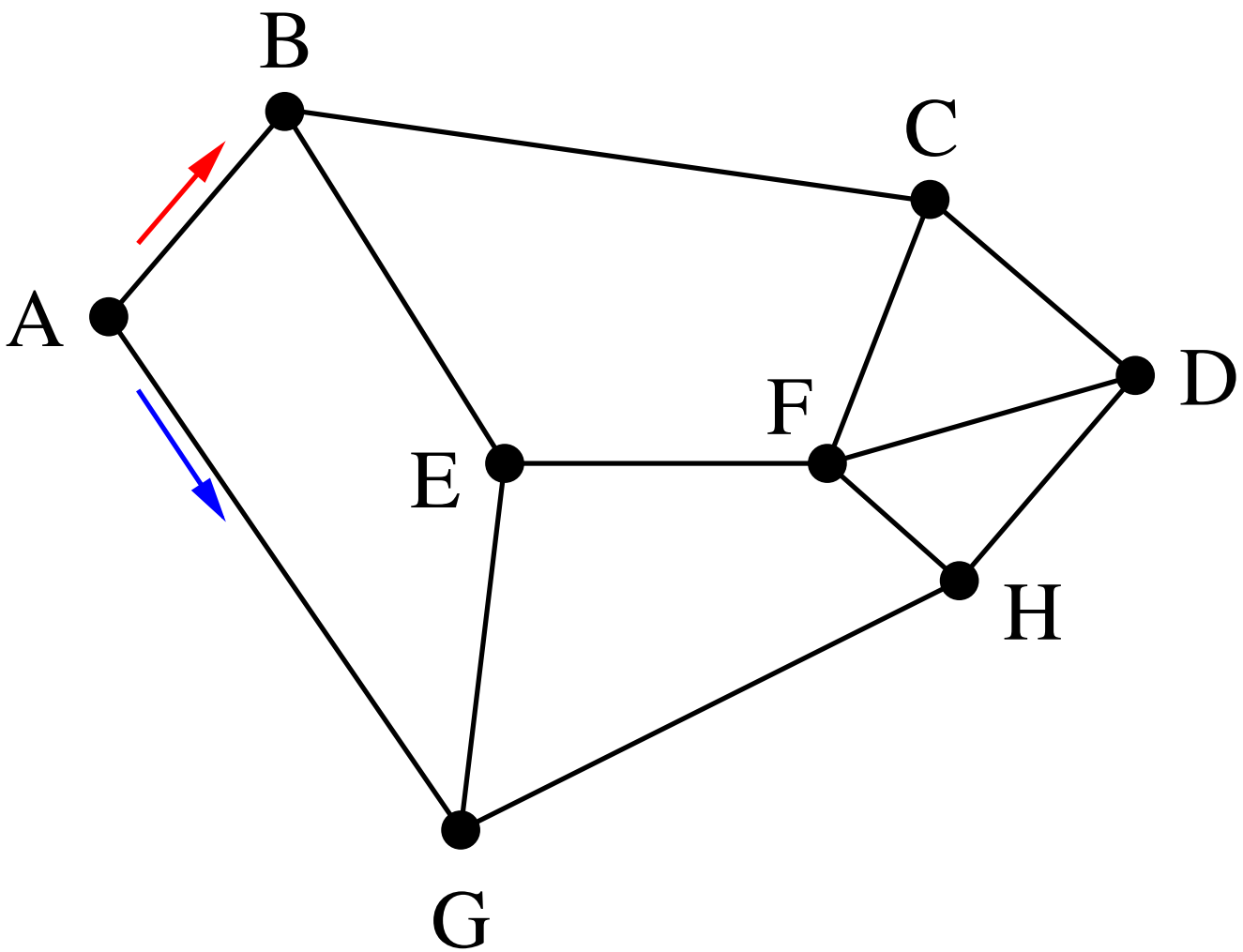
Die Vermittlungsschicht ist verantwortlich für den Transport von Nachrichten von einem Quellrechner zu einem Zielrechner über beliebig, untereinander verbundene Datenleitungen und Teilnetze. Die Nachrichten müssen an den Zwischenstationen vermittelt werden.


B
C
A
D
F
E
H
G

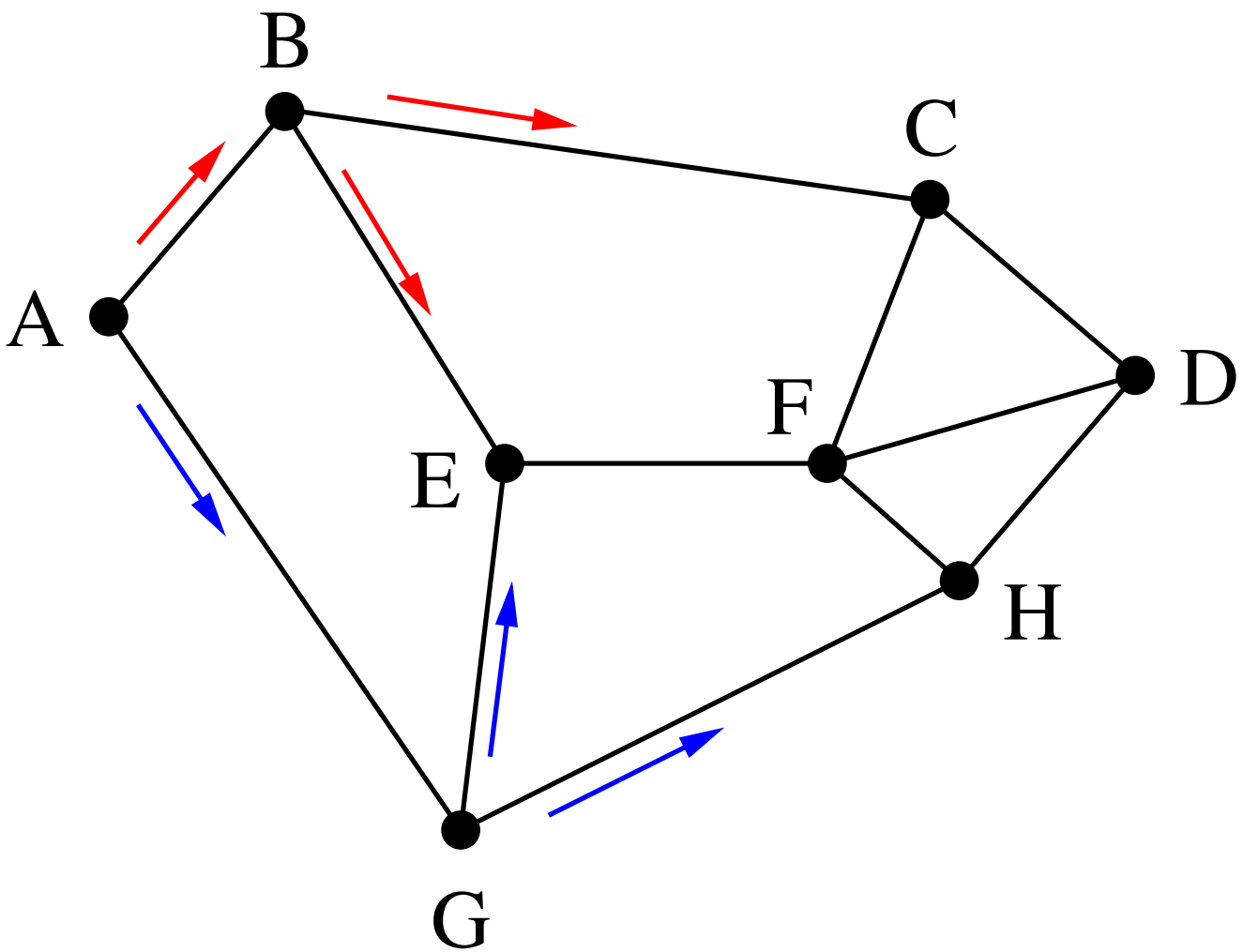
Fluten: Das Paket wird auf allen Leitungen mit Ausnahme der Empfangsleitung gesendet.

+ einfaches Verfahren
+ Paket findet den Weg zum Empfänger

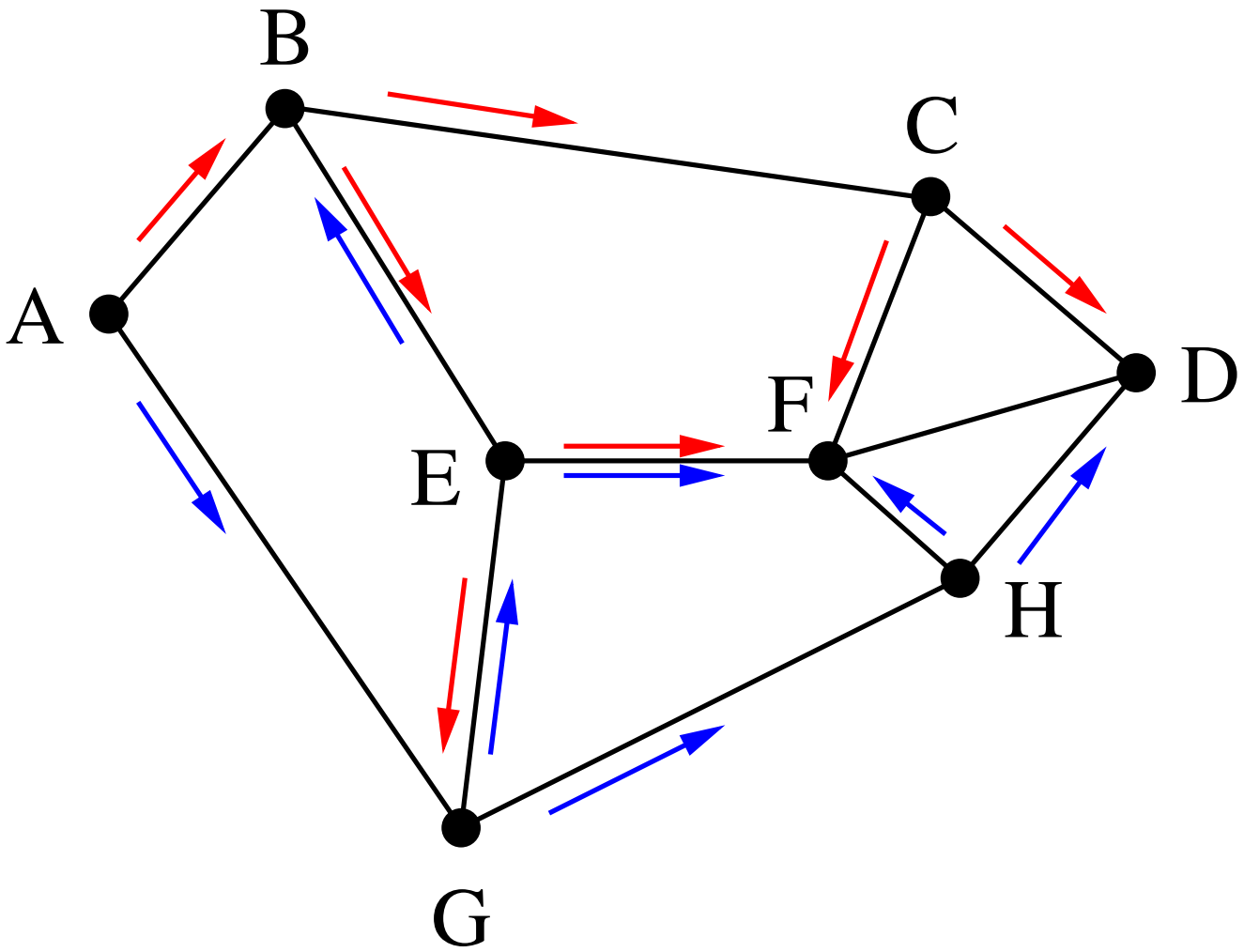
Beispiel: Nachricht von Knoten A an Knoten D



Beispiel: Nachricht von Knoten A an Knoten D

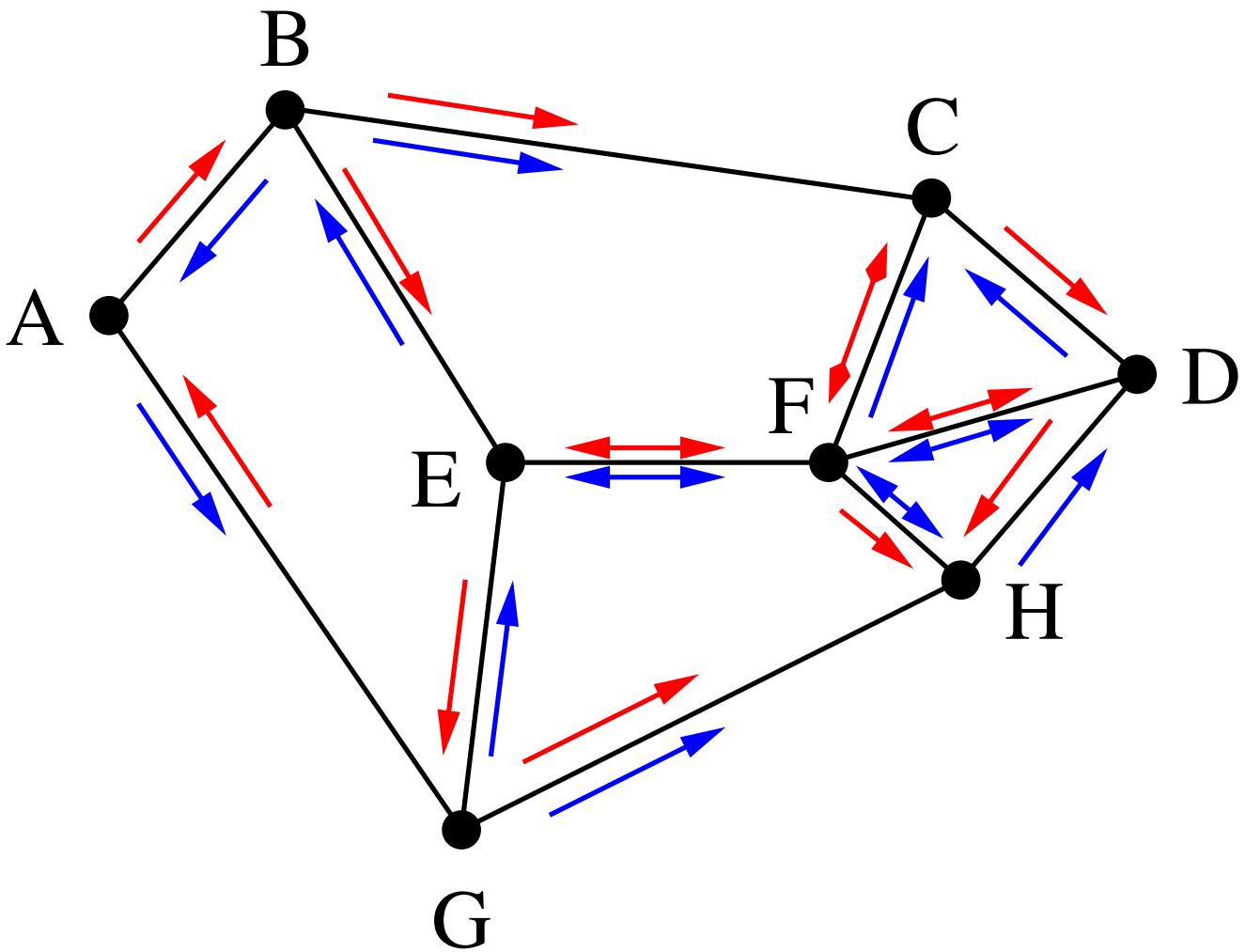


Beispiel: Nachricht von Knoten A an Knoten D



Die Nachricht hat Knoten D erreicht!

Beispiel: Nachricht von Knoten A an Knoten D



Fluten: Das Paket wird auf allen Leitungen mit Ausnahme der Empfangsleitung gesendet.

+ einfaches Verfahren
+ Paket findet den Weg zum Empfänger
• Empfänger muß Duplikate erkennen
- Das Replizieren der Pakete erzeugt eine *Flut* von Paketen

$\Longrightarrow$ Lebensdauer der Pakete im Netz begrenzen:

1. `Hops-Counter` := erwartete Anzahl Hops auf dem Weg zum Empfänger
2. Der `Hops-Counter` wird im Header H_V mitgeschickt.
3. In jedem Vermittlungsrechner wird das Feld `Hops-Counter` runtergezählt.
4. Falls `Hops-Counter` $== 0$, wird das Paket ggf. lokal zugestellt, aber nicht weiter repliziert.

Ziele eines Wegwahlverfahrens:

1. Übertragungszeit minimieren, d.h. den „schnellsten" Weg vom Sender zum Empfänger bestimmen,

   Der schnellste Weg hängt ab von:

   - Anzahl Vermittlungsrechner auf dem Weg (Anzahl Hops), da in jedem Rechner Verarbeitungszeit anfällt.
   - Leitungslänge und Bandbreite,
   - Auslastung der Vermittlungsrechner.

2. Gesamtdurchsatz maximieren,
3. Übertragungskosten minimieren.

Klassifikation von Wegwahlverfahren

1. statische Wegwahlverfahren: Wege werden im voraus berechnet und in sogenannten Wegwahltabellen gespeichert.

   Mögliche Parameter:

   - Netztopologie
   - Leitungslänge und Bandbreite
   - erwartete Auslastung der Leitungen

2. adaptive Wegwahlverfahren: Wegwahl paßt sich der aktuellen Netzsituation an:

   - Störungen (Leitungs-/Vermittlungsrechnerausfall)
   - aktuelle Lastsituation

1. Beispiel: Link-State-Routing-Algorithmus (statisch)

Ein Link-State-Routing-Algorithmus setzt die komplette Kenntnis des Netzwerk-Graphen voraus.
$\Longrightarrow$ Jeder Router kann die kürzesten Wege berechnen und damit seine Wegwahltabelle bestimmen.

Bestimmung des kürzesten Weges $\Longrightarrow$ Dijkstra-Algorithmus

Da volle Kenntnis des Netzwerk-Graphen vorausgesetzt wird, ist dieses Routingverfahren meist nur innerhalb des Netzes eines Netzwerkbetreibers anwendbar.

Dijkstra-Algorithmus zur Bestimmung des kürzesten Weges (Wiederholung)

Die Netztopologie entspricht einem zusammenhängenden Graphen $G = (E, K)$, wobei die Eckenmenge E die Menge der Vermittlungsrechner und die Kantenmenge K die Leitungen darstellt.
Der Graph ist bei Simplexbetrieb gerichtet, bei Duplexbetrieb ungerichtet. Im folgenden wird stets Duplexbetrieb betrachtet.
Aufgabe: Gesucht wird der kürzeste Weg von Vermittlungsrechner v_0 zu allen übrigen Knoten des Netzes.
Die Länge der Kanten wird mittels einer Gewichtsfunktion f gemessen:

$$f : E \times E \rightarrow R^+$$

Die Distanzfunktion gibt die Länge eines Weges $v_0, v_1, ..., v_k$ an:

$$d(v_0, v_k) = f(v_0, v_1) + f(v_1, v_2) + ... + f(v_{k-1}, v_k)$$

Initialisierung:

$S := \{v_0\}$ Menge derjenigen Knoten, zu denen ein kürzester Weg bereits gefunden wurde

Bestimme für alle $v \in E$ die Distanzfunktion

$$d(v) := \begin{cases} 0 & \text{falls } v = v_0 \\ f(v_0, v) & \text{falls die Kante } (v_0, v) \text{ existiert} \\ \infty & \text{sonst} \end{cases}$$

WHILE $(S \neq E)$

BEGIN Bestimme ein $w \in E \setminus S$ mit
$d(w) = min\{d(e) |\ e \in E \setminus S\}$;
//kürzester Weg von v_0 nach w gefunden
$S := S \cup \{w\}$;

FOR alle Nachbarknoten v von w mit $v \in E \setminus S$
DO $d(v) := min\{d(v), d(w) + f(w, v)\}$

END

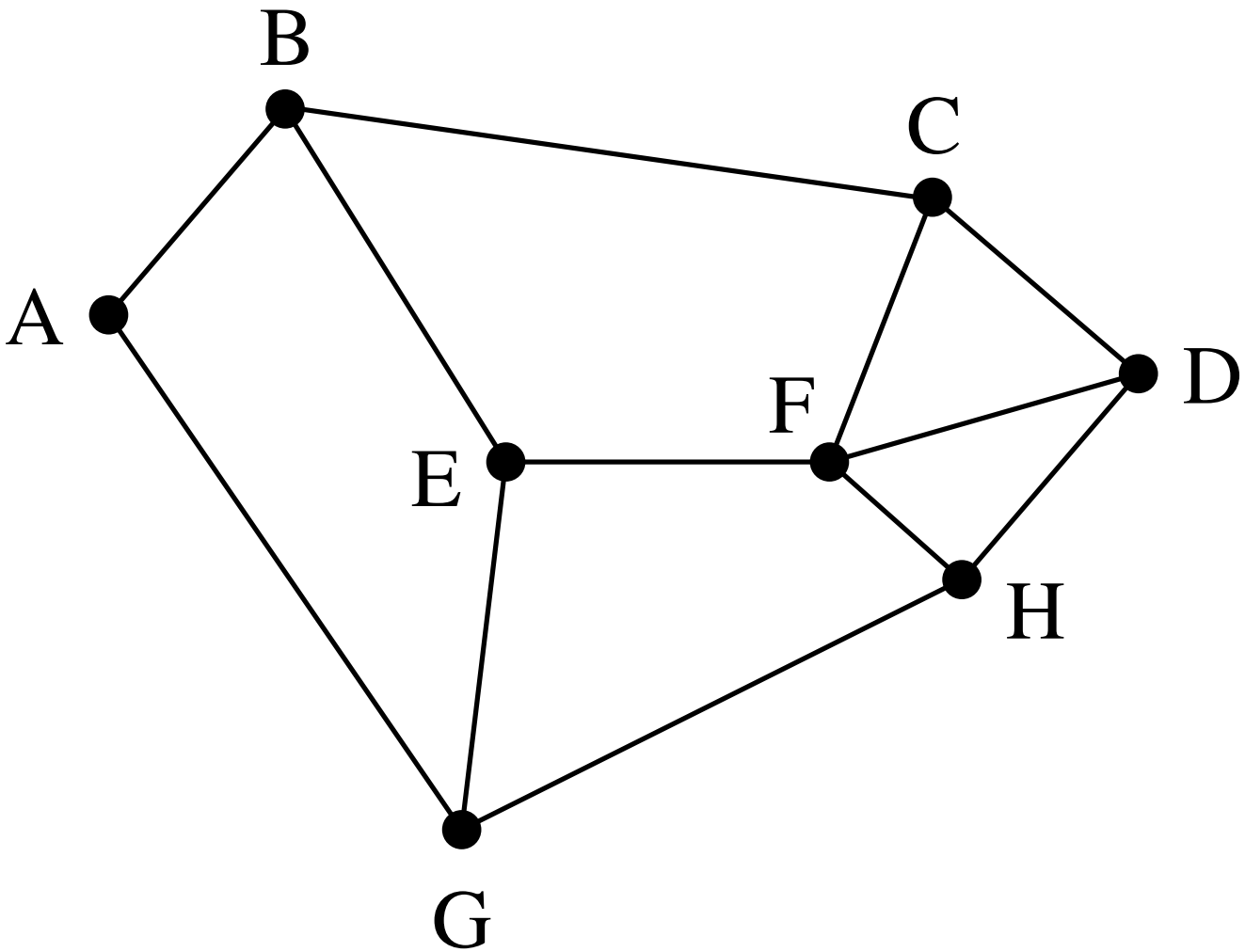
Bemerkung:

1. Der Algorithmus findet kürzeste Wege zu allen Knoten des Netzes.
2. Komplexität des Algorithmus: $O(n^2)$
3. In der Praxis wird als Gewichtsfunktion für die Kanten: $f = 1$ gewählt, d.h. es werden in der Distanzfunktion die Anzahl Hops gezählt.
4. $v_0, v_1, v_2, ..., v_k, w$ sei ein kürzester Weg von v_0 nach w. Dann ist $v_1, v_2, ..., v_k, w$ ein kürzester Weg von v_1 nach w.

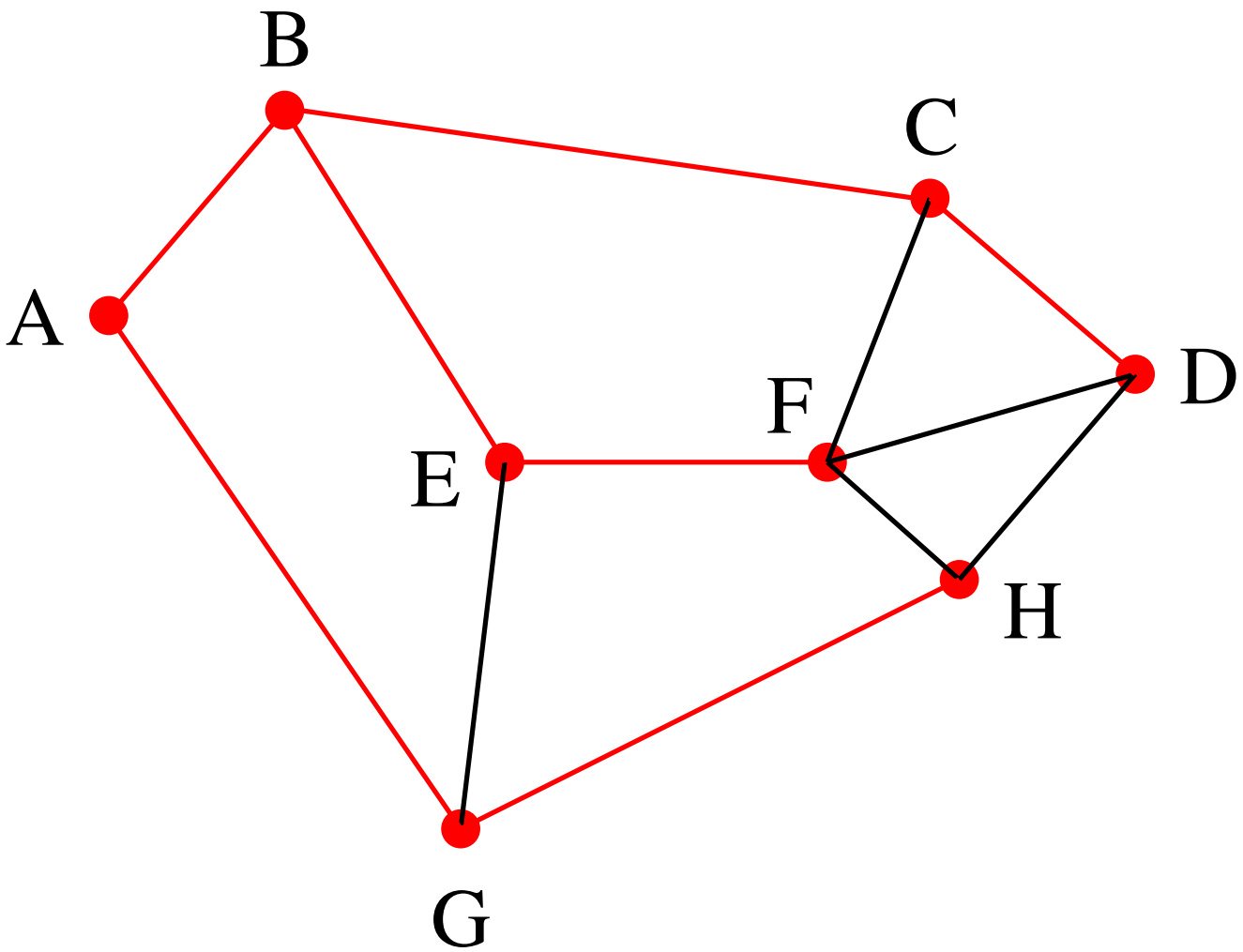
Folgerung:

Statt sich den kompletten Weg zum Zielknoten zu merken, genügt es sich einen Nachbarknoten zu merken, der auf einem kürzesten Weg zum Zielknoten liegt.
$\Longrightarrow$ kurze Wegwahltabellen

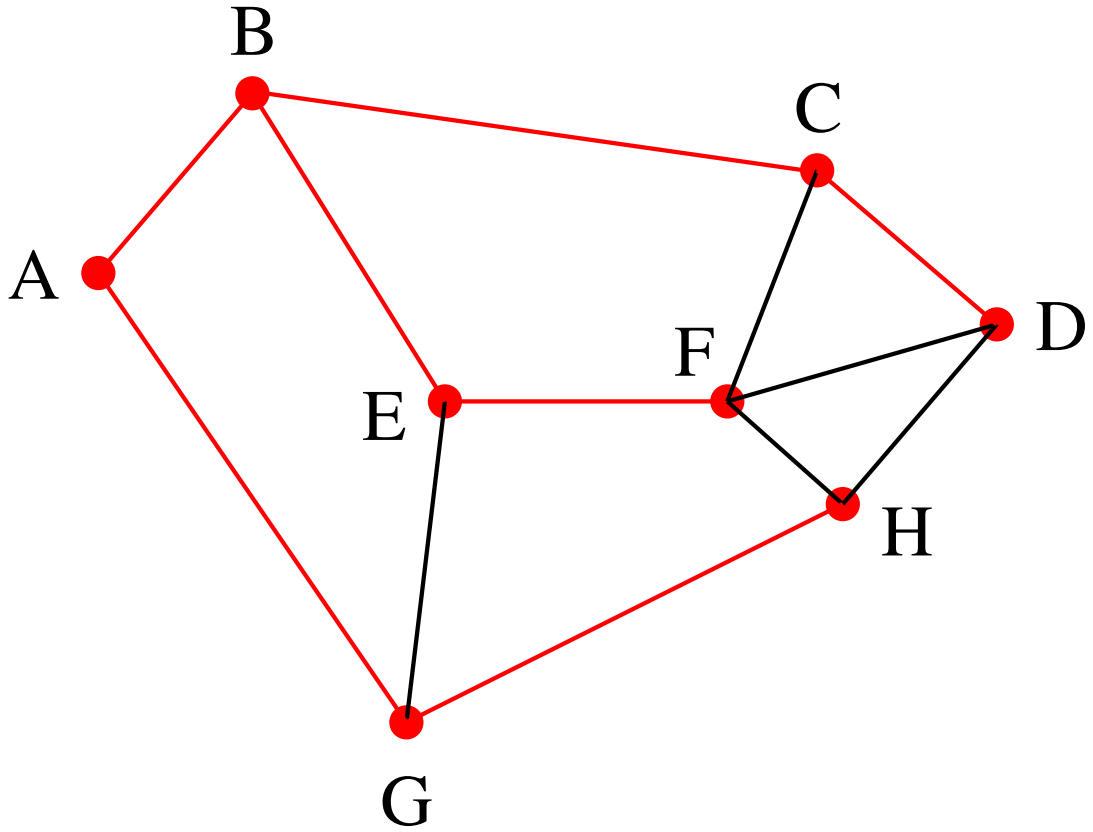
Beispiel: Wegwahltabelle für Knoten A:



Aufspannende Baum für Knoten A (zum Beispiel mittels Dijkstra-Algorithmus bestimmt):



Wegwahltabelle für Knoten A:

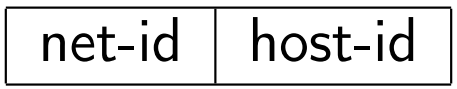


Ziel	**Link**
B	
C	
D	
E	
F	
G	
H	

Hierarchische Addressierung

Eine hierarchische Adressierung ermöglicht kleinere/effizientere Wegwahltabellen.

Beispiel:
Internet-Adressen sind 32-Bit Adressen, die aus zwei Teilen bestehen:



übliche Schreibweise: gepunktete Dezimalnotation, z.B. 141.83.21.121

Rechner mit mehreren Netzverbindungen (z.B. Gateways) haben mehrere IP-Adressen.

Einträge aus `/etc/route.conf`, Suse 7.3

```
192.168.0.0     0.0.0.0          255.255.255.0     eth1
141.89.59.0     0.0.0.0          255.255.255.0     eth0
default         141.89.59.254    0.0.0.0           eth0
```

1. Spalte: Ziel-Netz
4. Spalte: ausgehende Interface, auf dem das Paket weitergereicht werden soll

Isolierte (lokale) Verfahren von Baran (1964): Jeder Vermittlungsrechner entscheidet selber aufgrund von selbstgesammelten Informationen.

1. Hot-Potato-Routing: Die empfangene Nachricht wird so schnell wie möglich weitergegeben, d.h. an den Nachbarn mit der kürzesten Warteschlange vor der Ausgangsleitung.

   Modifikation: Wegwahltabelle, die nicht nur die kürzesten Wege, sondern auch Alternativwege enthält. Unter diesen wird dann gemäß Warteschlangenlänge entschieden.

2. Backward Learning: Die Pakete zählen unterwegs die Anzahl Vermittlungsrechner, die sie passieren (`hopscount`). Empfängt ein Rechner A über den Nachbarknoten N ein Paket von Rechner S mit `hopscount` $= k$, *lernt* A, daß es einen Weg der Länge k zu S über N gibt und ergänzt seine Wegwahltabelle:

Ziel	Nachbarknoten	Länge
...	...	...
S	N	k

Regeln:

1. **Wegwahl:** Ist der Empfänger in der Wegwahltabelle eingetragen, wird das Paket an den zuständigen Nachbarknoten geschickt.
   Sonst: Fluten
2. **Update:** Die Wegwahltabelle wird aktualisiert, falls ein Paket eintrifft, das einen kürzeren Weg angibt.

siehe auch: Wegwahlverfahren in Sensornetzen

3. Vector Distance Routing

Idee: gesteuertes/ Periodisches Lernen kürzester Wege

Ansatz: Austausch von Übertragungszeitvektoren:

$$\begin{pmatrix} \tilde{d}(A, Z_1) \\ \tilde{d}(A, Z_2) \\ \dots \\ \dots \\ \dots \\ \tilde{d}(A, Z_n) \end{pmatrix}$$

$\tilde{d}(A, Z_i)$ gibt die aktuelle Schätzung für die Übertragungszeit von A zum Ziel Z_i an (z.B. in Form der Hop-Count-Metrik).

$\tilde{d}(A, Z_i) = \infty$, falls kein Weg bekannt ist.

1. Initialisierung der Wegwahltabellen z.B. mittels Dijkstra-Algorithmus oder Backward Learning
2. Knoten A mißt periodisch die Laufzeit von Nachrichten zu seinen Nachbarknoten N: $d(A, N)$
3. A schickt entweder in regelmäßigen Zeitabständen (synchron) oder nach einer Aktualisierung (asynchron) den Übertragungszeitvektor an alle seine Nachbarn.

4. Neuberechnung der Wege: Hat A die Übertragungsvektoren aller seiner Nachbarn erhalten, so wird für jeden Zielknoten Z und jeden Nachbarn N die geschätzte Weglänge $d(A, Z)$ über N neu berechnet:

$$d_N(A, Z) := d(A, N) + \tilde{d}(N, Z)$$

und

$$d(A, Z) := \min\{d_N(A, Z) \mid N \text{ Nachbar von } A\}$$

bestimmt.

Nachteil: Die Länge der Übertragungsvektoren hängt von der Anzahl beteiligter Netze ab $\Longrightarrow$ Verfahren skaliert nicht!

Autonome Systeme

Definition 36:

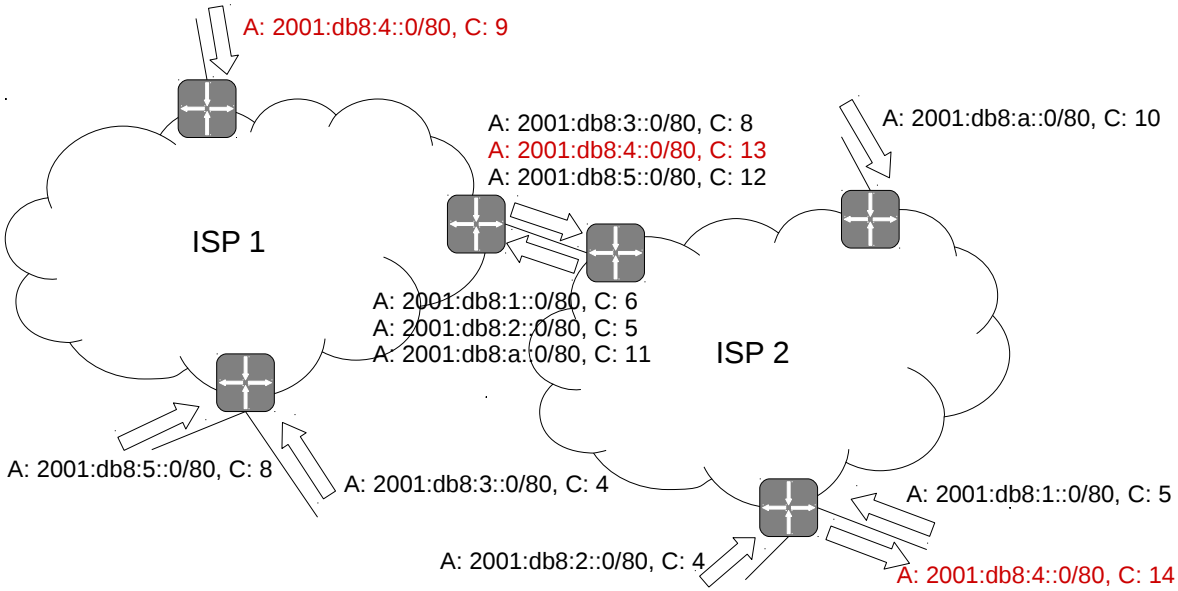
1. Eine Menge von Netzen und Routern, die administrativ zusammengehören, heißen Autonomes System. Mittels des Exterior Gateway Protocol (EGP) findet Wegwahl zwischen Autonomen Systemen statt.
2. Wege innerhalb eines autonomen Systems werden mit dem Interior Gateway Protokoll (IGP) bestimmt.

Beispiel für ein Autonomes System: DFN ist AS680

Beispiel für ein Exterior Gateway Protokoll: Border Gateway Protocol (BGP)

Tobias Tragsdorf: „*Sicheres BGP Routing*“, Diplomarbeit, Universität Potsdam, 2012.

Border Gateway Protocol



- Y. Rekhter, T. Li, S. Hares: *A Border Gateway Protocol 4 (BGP-4)*, RFC 4271, 2006
- Ziel: **Routing** über Netzgrenzen hinweg (OSI-Layer: 3)
- Benachbarte Router bauen Verbindung über TCP-Port 179 auf
- **Annoncierung** von Routeninformationen
- (ökonomische) **Routingentscheidungen**

Beispiele für IGP-Protokolle:

1. 80er Jahre: **Vector Distance Routing:** RIP (Routing Information Protocol)
   RFC 1058 (1988), RFC 2453 (1998)
   weit verbreitet: als `routed` im 4 BSD-UNIX
   Austausch der Vektoren alle 30 s, f = 1 (Hop-Count-Metrik)
   Vektoren enthalten einen Eintrag für jedes Zielnetz
   Nachteil: skaliert nicht!

2. **Link-State-Protokoll:** OSPF (Open Shortest Path First)
   basiert auf Dijkstra-Algorithmus, Vermittlungsrechner testen regelmäßig die Verbindungen zu ihren Nachbarn
   Fällt ein Link aus, geht die Update-Information an alle Router.
   J. Moy, OSPF Version 2 RFC 2328 (1998)

3. IS-IS (Intermediate System to Intermediate System Protocol)
   ISO-Norm 10589, Link-State-Protokoll ähnlich zu OSPF

Definition 37: Paketfragmentierung wird erforderlich, wenn ein Paket von Netz 1 in Netz 2 weitergeleitet werden soll und das Paket größer als die Rahmengröße von Netz 2 ist. Dann wird das Datenpaket aus Netz 1 in mehrere kleinere Datenpakete zerlegt, um der Rahmengröße von Netz 2 zu genügen.

Maximum Transfer Unit: maximale Größe des Datenfeldes (Payload) vom Rahmen der Sicherungsschicht

X.25 (1970-2012):	126 Byte, 496 Byte
IEEE 802.3 Ethernet:	1500 Byte
IEEE 802.11 Wireless LAN (WiFi):	2312 Byte
IEEE 802.15.4 (Sensornetze):	53-106 Byte

1. Transparente Fragmentierung:

Jedes Fragment wird an das gleiche „Ausgangsgateway" adressiert. Dort werden die Fragmente wieder zusammengesetzt, bevor sie ggf. in das nächste Netz geschickt werden.

+ Netze, in denen das Paket nicht fragmentiert werden muß, durchläuft das Paket als eine Nachricht. $\Longrightarrow$ weniger fehleranfällig
– Pro Netz muß ggf. einmal fragmentiert und reassembliert werden

$\Longrightarrow$ bei virtuellen Verbindungen geeignet

2. Internetfragmentierung für Datagrammnetzwerke: Fragmente werden erst am Zielrechner wieder zusammengesetzt.

\+ minimaler Reassemblierungsaufwand

-/+ Fehlerkontrolle liegt beim Zielrechner

\+ flexible Wegwahl: Fragmente können unterschiedliche Ausgangsgateways benutzen
$\Longrightarrow$ Header muß auf alle Fragmente kopiert werden

Kennzeichen:

- Schicht 3 Protokoll,
- Datagram Service,
- Wegwahl, Anpassen der Paketgröße, Adressen etc. beim Übergang in anderes Netz, Flußkontrolle, ...
- hierarchische Wegwahl: Routing orientiert sich am Zielnetzwerk, nicht am Zielrechner

Vinton Cerf, Robert Kahn und TCP/IP

Quelle: James F. Kurose und Keith W. Ross: *Computernetzwerke*, S. 273

In den frühen 1970ern wuchs die Zahl der Paketvermittlungsnetze stark an, wobei das ARPAnet - der Vorläufer des Internets - nur eines von vielen Netzen war. Jedes dieser Netze hatte sein eigenes Protokoll. Zwei Forscher, Vinton Cerf und Robert Kahn, erkannten die Wichtigkeit, diese Netze zusammenzuschalten, und entwickelten ein netzwerkübergreifendes Protokoll, das sie TCP/IP (für Transmission Control Protocol/Internet Protocol) nannten.

Während Cerf und Kahn ihr Protokoll zunächst als eine Einheit betrachteten, wurden später die Bestandteile TCP und IP getrennt.

Das TCP/IP-Protokoll, die Basis des heutigen Internets, wurde vor den modernen PCs und Workstations entworfen, bevor Ethernet und lokale Netzwerke Verbreitung gefunden hatten, vor dem Web, Audio-/Video-Streaming und Chat.

Cerf und Kahn erkannten die Notwendigkeit eines Netzwerkprotokolls, das einer Vielzahl von noch zu definierenden Anwendungen eine solide Basis bot, und gleichzeitig beliebige Kombinationen von Hosts und Sicherungsschichtprotokollen ermöglichte.

2004 erhielten Cerf und Kahn den ACM Turing Award, den „Nobelpreis der Informatik“, für ihre „Pionierarbeiten im Internetworking, einschließlich des Entwurfs und der Implementation der grundlegenden Kommunikationsprotokolle des Internets, TCP/IP, und für ihre inspirierte Führungsrolle im Bereich Computernetzwerke“.

Die Publikation über TCP/IP:

Vinton Cerf, Robert Kahn: *A Protocol for Packet Network Interconnection*, IEEE Transactions on Communications Technology, Vol. COM-22, No. 5, S. 627-641, **1974**.

(https://www.cs.princeton.edu/courses/archive/fall06/cos561/papers/cerf74.pdf

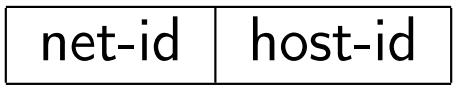
endet mit dem Ausblick: *"The next important step is to produce a detailed specification of the protocol so that some initial experiments with it can be performed."*

RFC 791, Jon Postel: INTERNET PROTOCOL **(1981)**

RFC 793, Jon Postel: TRANSMISSION CONTROL PROTOCOL **(1981)**

Internet-Adressen

Internet-Adressen sind 32-Bit Adressen, die aus zwei Teilen bestehen:



Vinton Cerf, einer der Väter des Internets, im Vortrag an der Universität Potsdam, 26.5.2011:

"How could we make open standards? It was the time of the cold war!
We did it and nobody noticed.

If you want to blame someone for the 32 bit IPv4 addresses, blame me! – I had to decide, and we expected the network to be ***experimental****."*

Übliche Schreibweise: gepunktete Dezimalnotation, z.B. 141.83.21.121

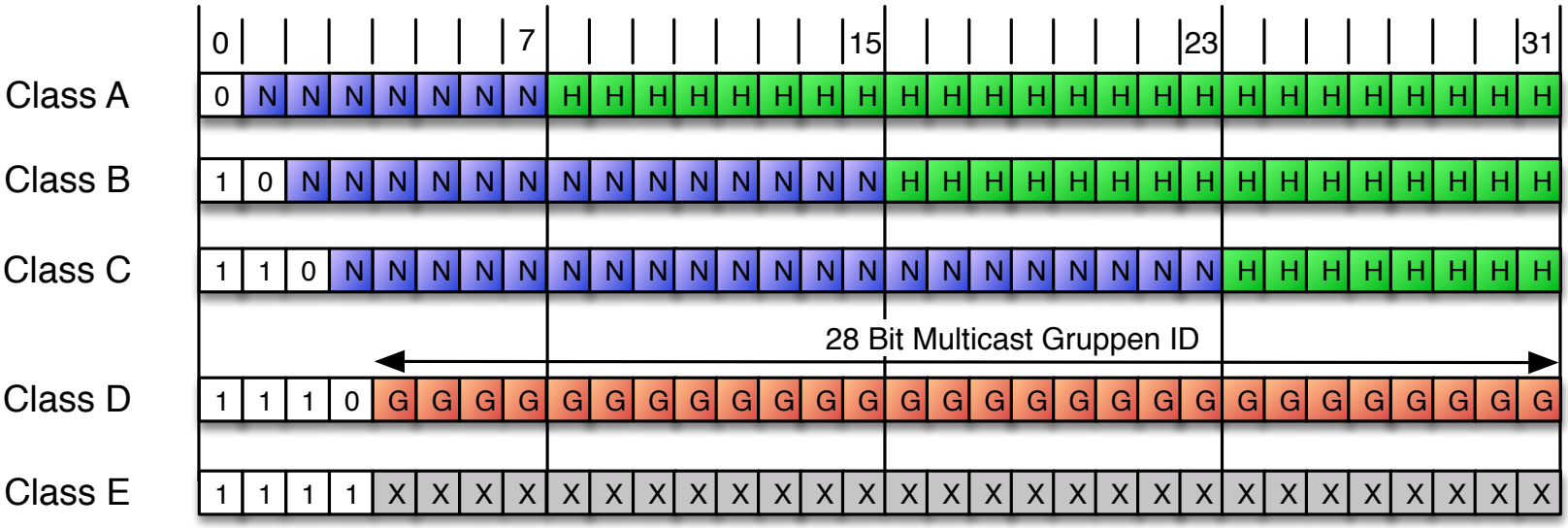
	net-id	host-id
Netzadresse	net-id	0 ... 0
Broadcast-Adresse	net-id	1 ... 1
Limited Broadcast (lokal im LAN)	1 ... 1	1 ... 1
Loopback-Interface[17]	127	beliebig (oft 1)

Rechner mit mehreren Netzverbindungen (z.B. Gateways) haben mehrere IP-Adressen.

Die non-profit Organisation ICANN (Internet Corporation for Assigned Names and Numbers) ist für das Management der Internetadressen zuständig (u.a. Zuteilung der Adressen, Koordination der DNS-Root-Server, ...)

[17] Die Loopback-Adresse wird nicht geroutet, sondern lokal zugestellt. Sie dient zum Testen.

Adressklassen von IPv4



- **Class A**: (`0.0.0.0 - 127.255.255.255`), **Class B**: (`128.0.0.0 - 191.255.255.255`) **Class C**: (`192.0.0.0 - 223.255.255.255`)
- Das Konzept der Adressklassen wurde 1993 aufgegeben und durch die Einführung des Classless Inter-Domain Routing (CIDR) ersetzt.
  $\Longrightarrow$ Durch die Verwendung von variablen Subnetzmasken lässt sich der IP-Adressraum besser ausnutzen.

CIDR:

- Um die Net-ID zu bestimmen werden die IP-Adresse und Subnetzmaske mittels bitweisem logischen UND verknüpft.

  Beispiel: 192.168.1.129 mit Subnetzmaske 255.255.255.0
  Net-ID: 192.168.1, Host-ID = 129
- Alternativ: Angabe der Subnetzmaske durch Angabe der gesetzten Bits: /n

  Beispiel: 192.168.1.129/24

$\Longrightarrow$ Umso länger der Präfix (hier 24), desto kleiner das Subnetz.

Private IP-Adressen

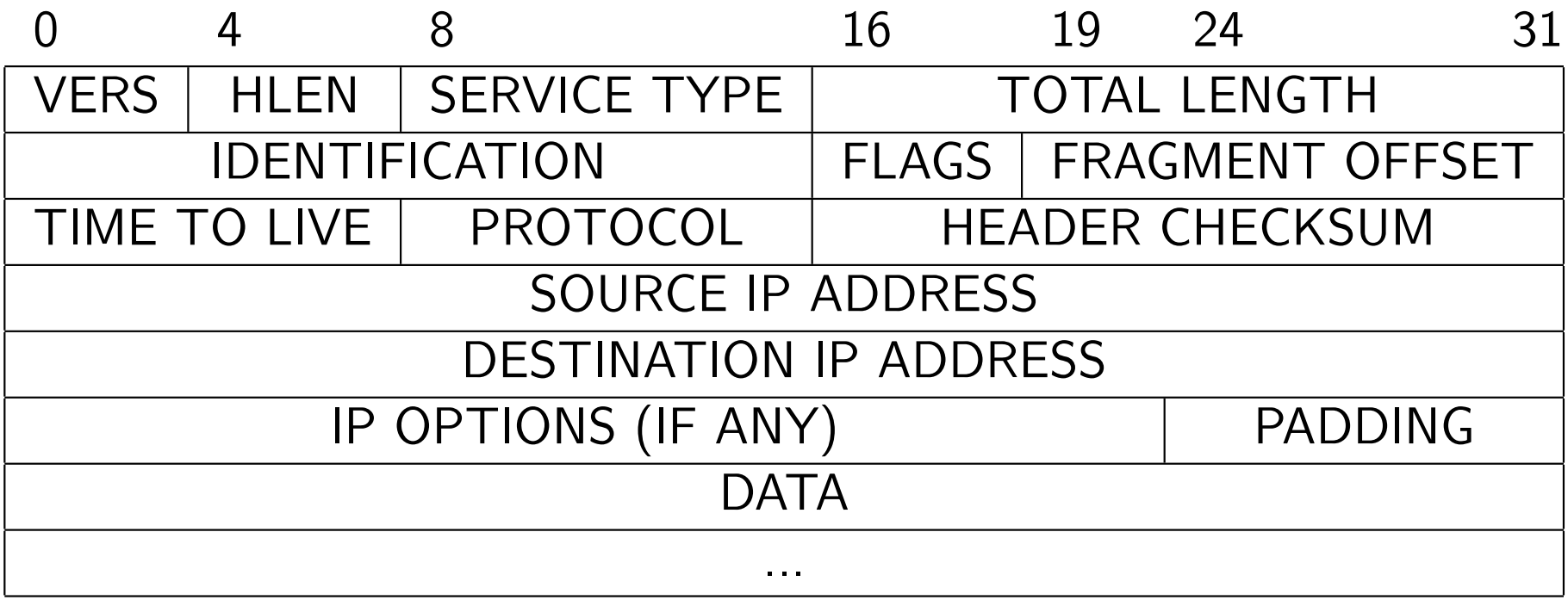
Im Gegensatz zu öffentlichen IP-Adressen sind private IP-Adressen frei verfügbar, werden aber nicht im Internet weitergeleitet (auch: nonroutable addresses).

Präfix	kleinste Adresse	größte Adresse
10/8	10.0.0.0	10.255.255.255
172.16/12	172.16.0.0	172.31.255.255
192.168/16	192.168.0.0	192.168.255.255
169.254/16	169.254.0.0	169.254.255.255

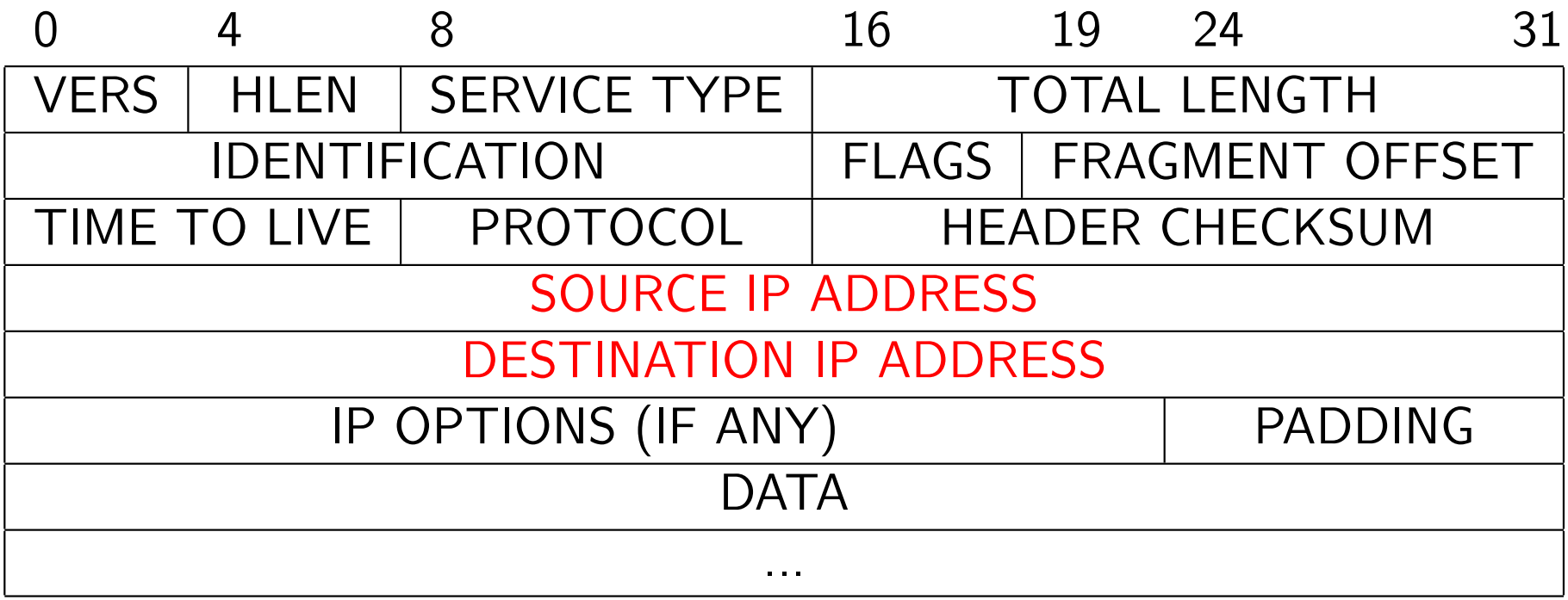
Vorteil: Mit privaten IP-Adressen lassen sich Netzbereiche aufbauen, die im Internet nicht sichtbar sind. $\Longrightarrow$ freie Designentscheidung

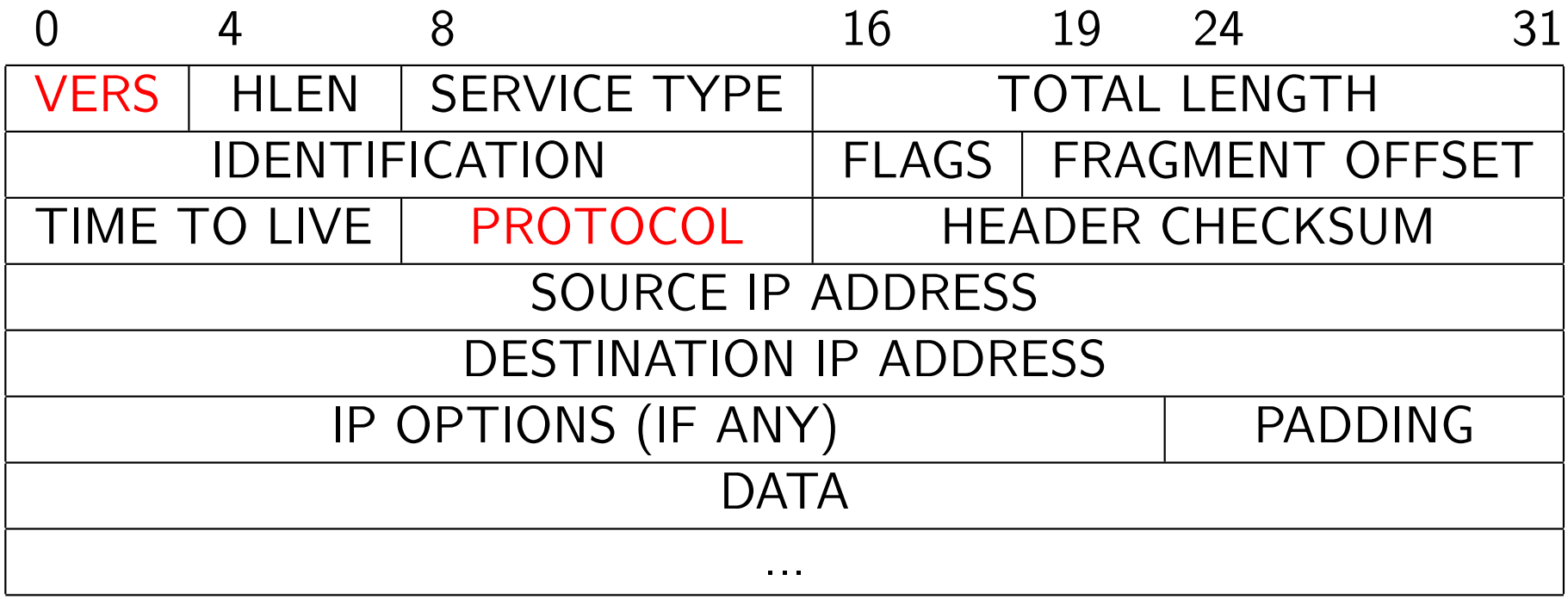
Nachteil: Erfordert *Network Address Translation* (NAT) für den Übergang in das Internet.

IP Datagrammformat:

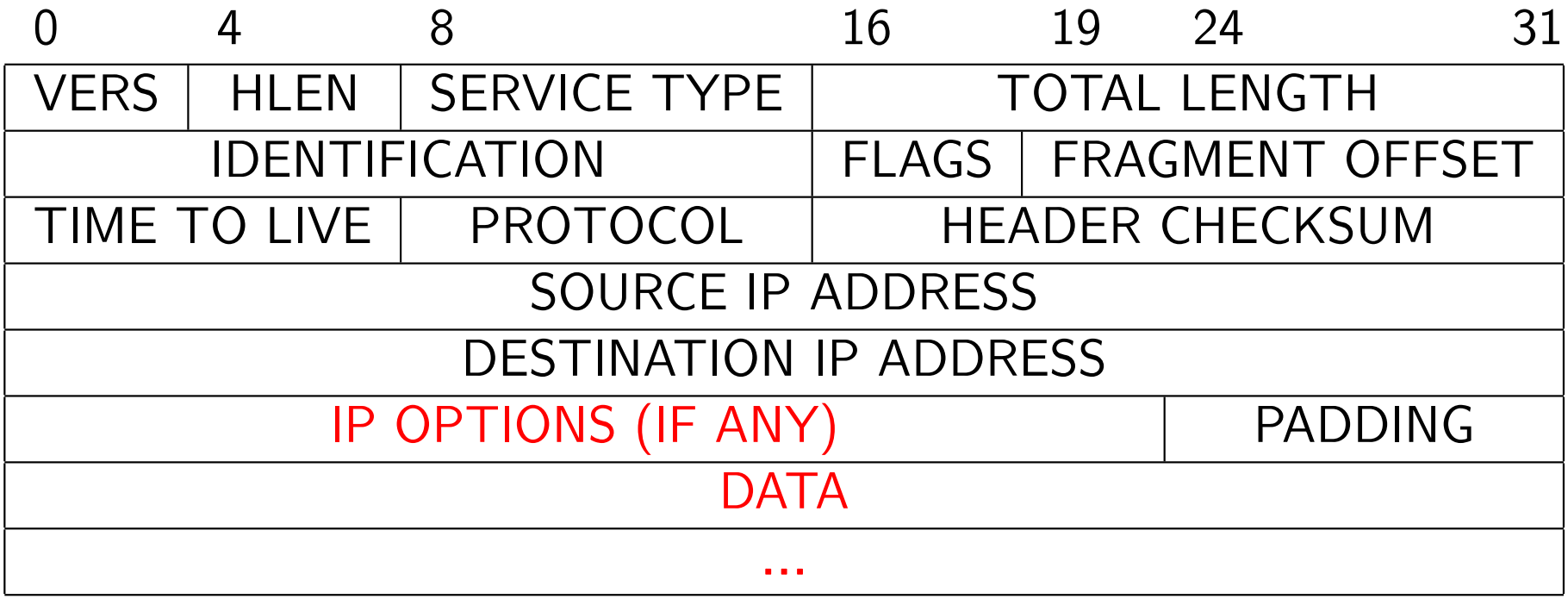


IP Datagrammformat:

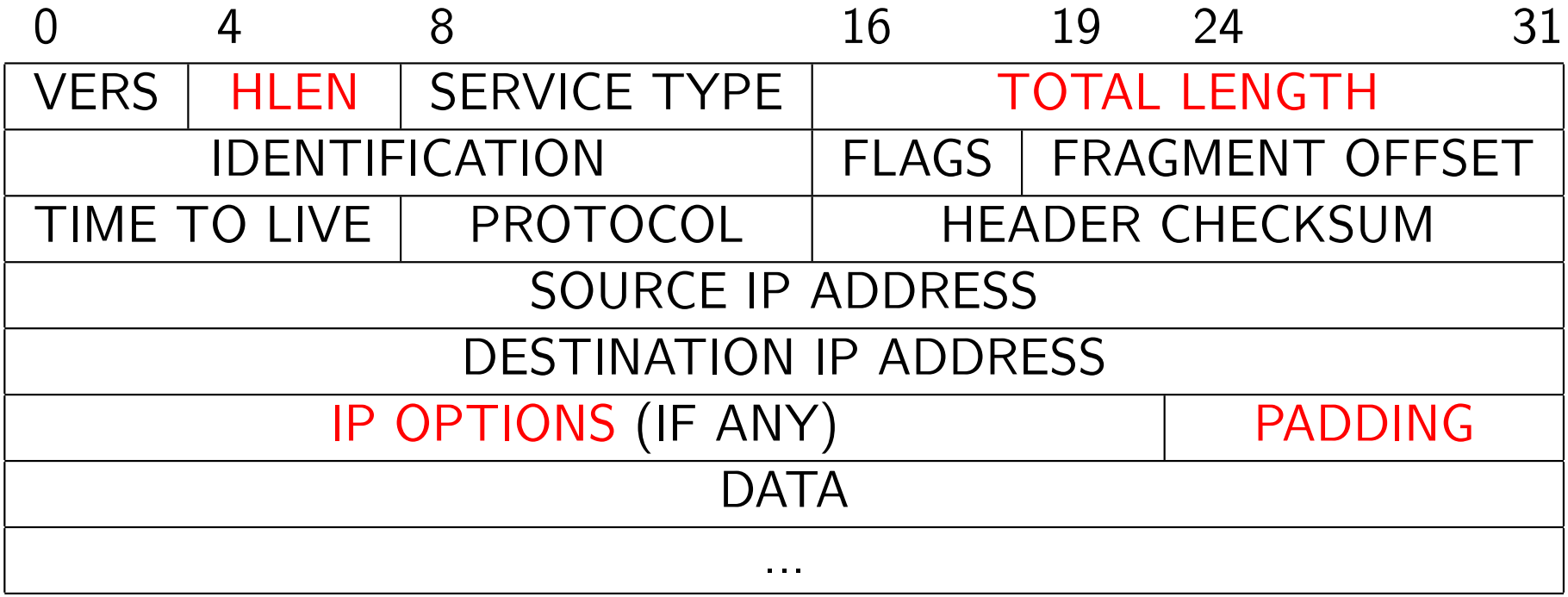




- VERS: Versionsnummer des benutzten IP-Protokolls: 4 oder 6
- PROTOCOL: Bestimmt das Transportprotokoll, das die Daten erzeugt hat (z.B. TCP oder UDP)



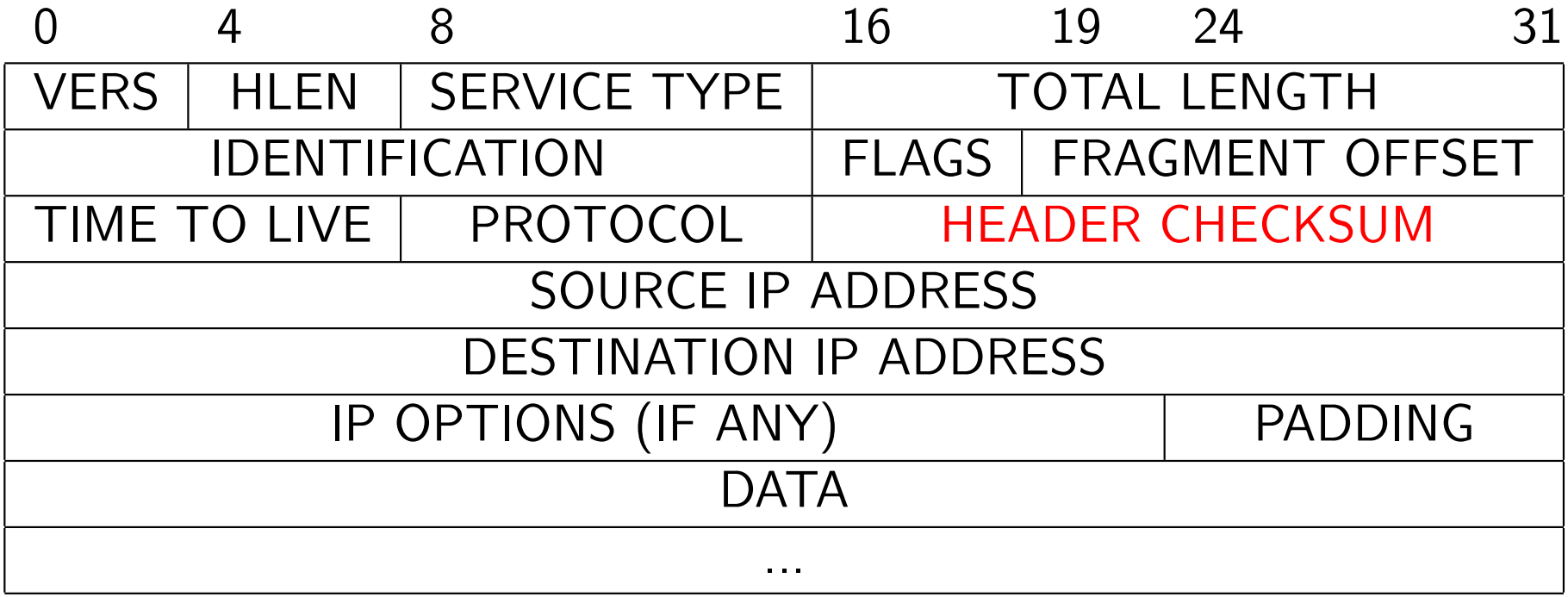
$\Longrightarrow$ **Variable Header- und Daten-Länge**



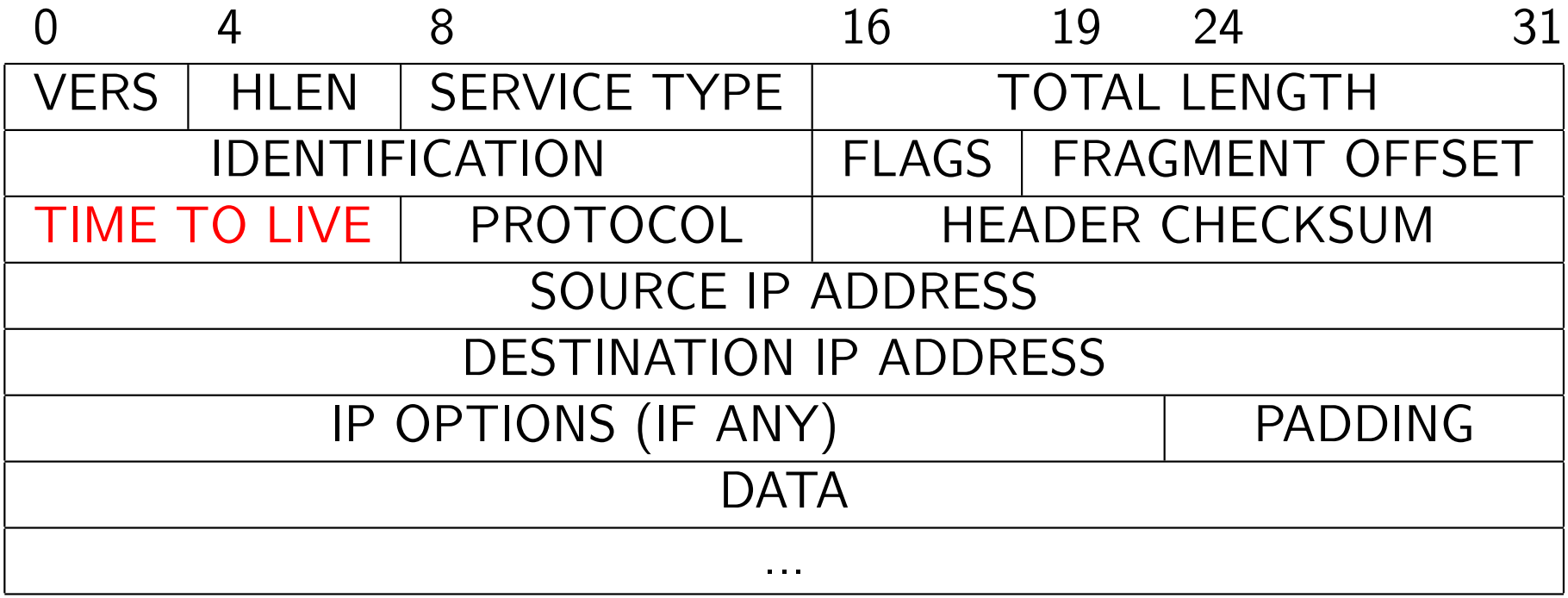
Variable Header- und Daten-Länge:

- HLEN: Header-Länge in 32-Bit-Worten $\Longrightarrow$ PADDING
- TOTAL LENGTH $< 2^{16} =$ **65 536 Byte**
- Datenlänge (Payload): TOTAL LENGTH – HLEN $\cdot$ 4 [Byte]
- optional sind die Felder: IP OPTIONS, PADDING

$\Longrightarrow$ **minimaler IP-Header: 20 Byte**

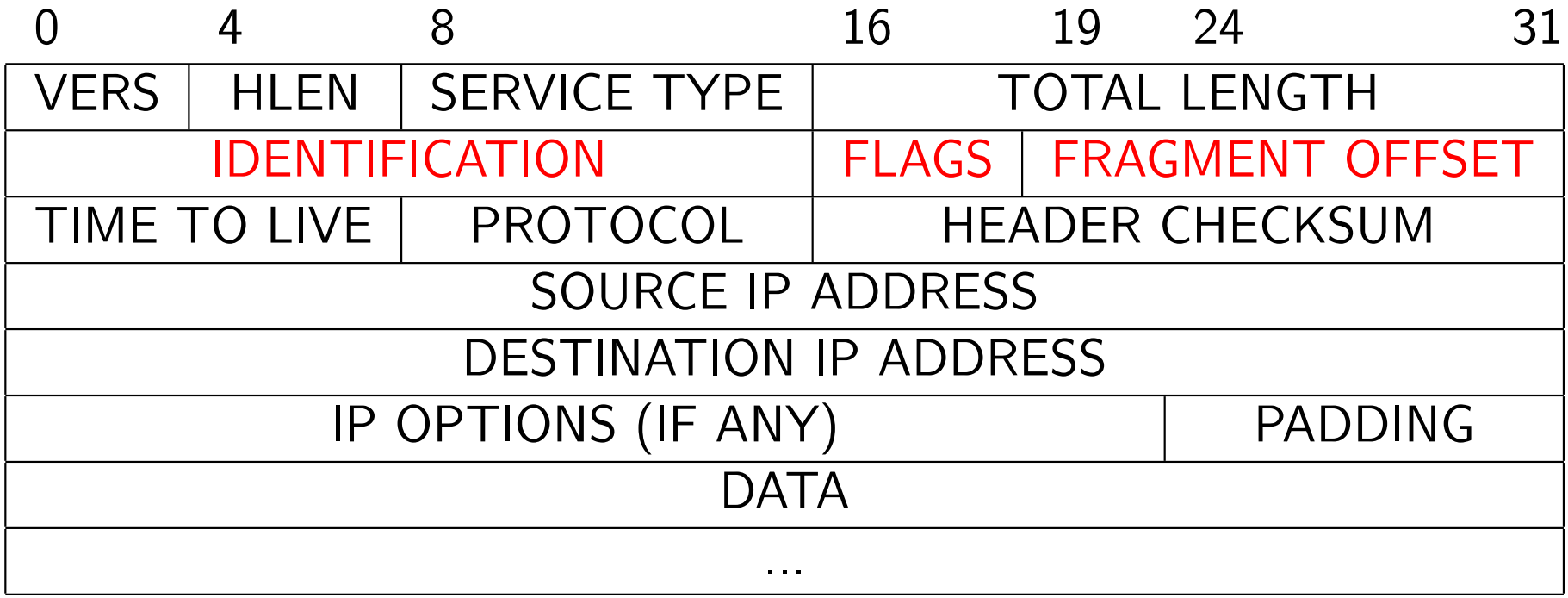


HEADER CHECKSUM: Prüfsumme überprüft nur den Header:
Einerkomplement von je 16-Bit $\rightarrow$ Einerkomplement der Summe



TTL: Zähler für die Lebensdauer des Datagramms:
In jedem Gateway wird die TTL um 1 und zusätzlich für jede Sekunde Wartezeit um 1 runtergezählt.
TTL $= 0 \Longrightarrow$ Das Datagramm wird vernichtet und der Absender mittels einer Internet Control Message Protocol (ICMP) Nachricht (11 - Time Exceeded) benachrichtigt.

Internet–Fragmentierung: Fragmente werden erst am Zielrechner wieder zusammengesetzt.

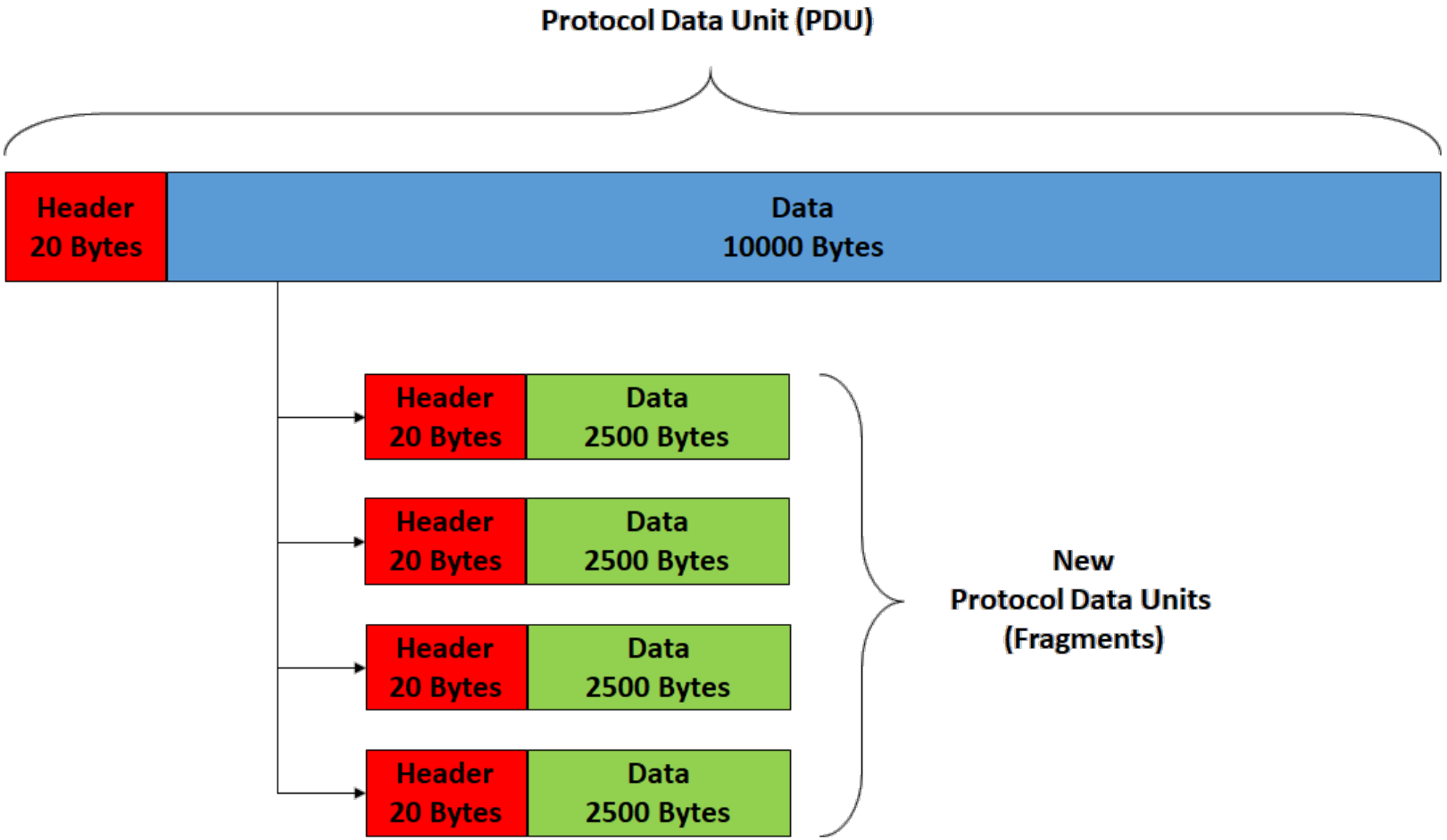


Das IP-Datagramm wird in sogenannte Fragmente zerlegt.
Fragmente sind wieder IP-Datagramme.

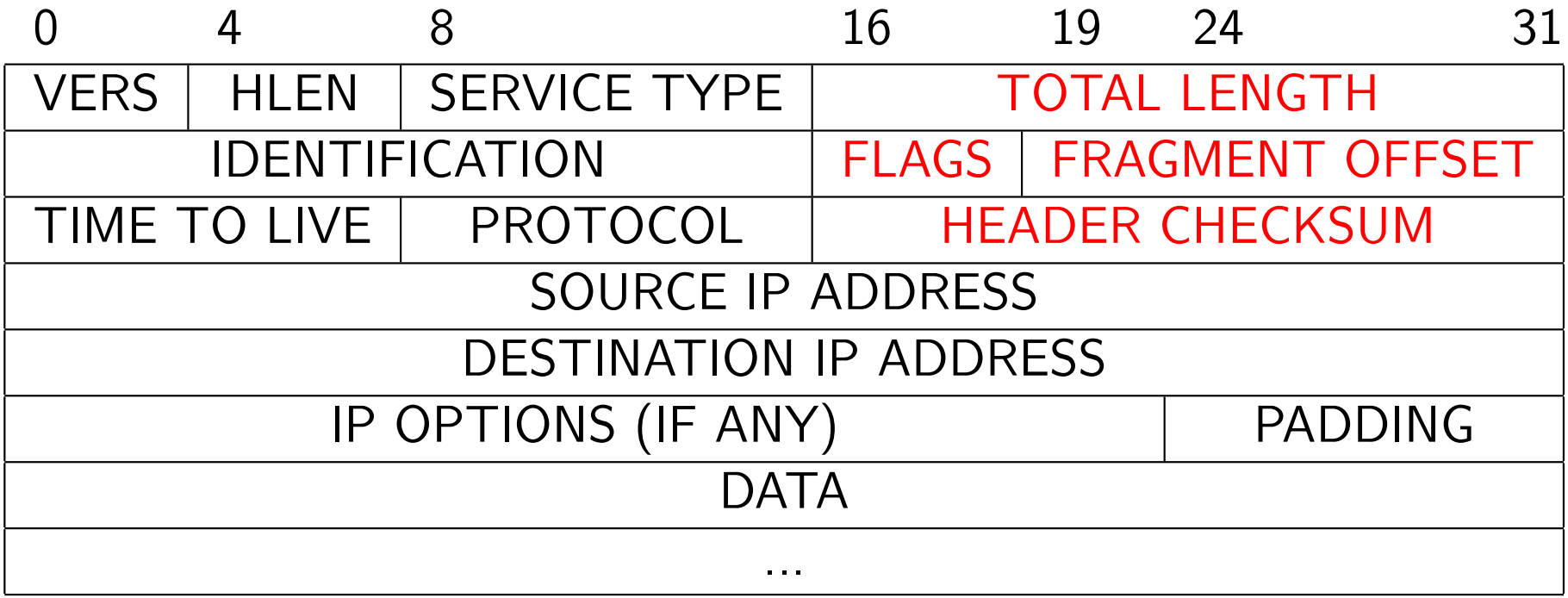
IDENTIFIKATION:= eindeutige Datagrammnummer $< 2^{16}$

$\Longrightarrow$ Die IDENTIFIKATION wird in jeden Fragmentheader kopiert.

Beispiel: Übergang in ein Netzwerk mit MTU = 2520.



Quelle: englische Wikipedia-Seite zu IP-Fragmentierung



Die Fragment-Header unterscheiden sich in:
FRAGMENT-OFFSET (Vielfaches von 8 Bytes), TOTAL LENGTH, FLAGS, HEADER CHECKSUM

Reassembly (Zusammensetzen) beim Empfänger:

1. Jedes empfangene Fragment wird an die Position gemäß dem angegebenen Fragment Offset kopiert
2. Der Fragment Offset wird als Vielfaches von 8 Bytes angegeben.
3. Alle Fragmente, bis auf das letzte Fragment haben das More-Fragment-Bit (MF-Bit) gesetzt.
4. Wurde das letzte Datagramm empfangen (MF-Bit = 0) und ist die Nachricht komplett, kann sie an die höhere Protokollschicht (TCP oder UDP) übergeben werden.

Beispiel:

Datagramm hat Payload von 3000 Bytes, MTU = 1500 Bytes.
minimaler IP-Header: 20 Bytes $\Longrightarrow$ maximal 1480 Bytes

mögliche Fragmentierung:

1. Datagramm: 1200 Bytes

...	Total Length	ID	Flags	Fragment Offset	....	Data
...	1220	12345	001	0	....	Data_1

2. Datagramm: 1200 Bytes

...	Total Length	ID	Flags	Fragment Offset	....	Data
...	1220	12345	001	1200/8=150	....	Data_2

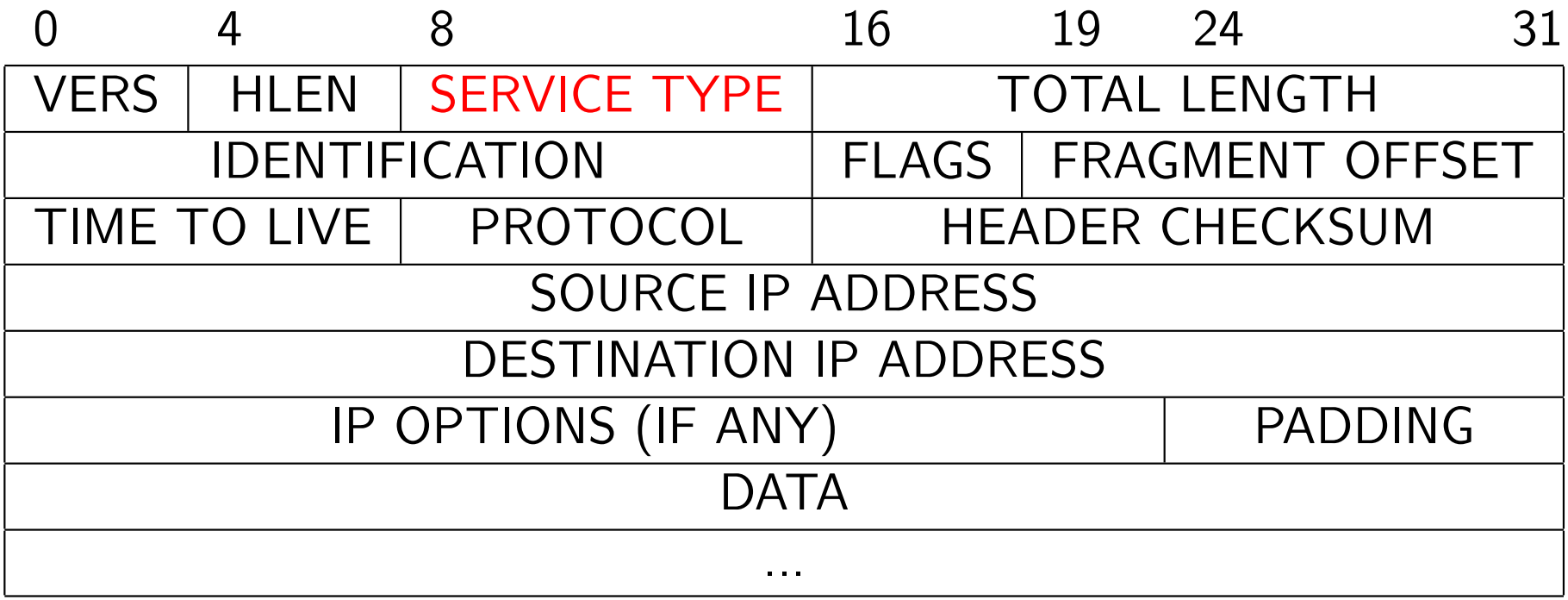
3. Datagramm: 600 Bytes

...	Total Length	ID	Flags	Fragment Offset	....	Data
...	620	12345	000	2400/8=300	....	Data_3

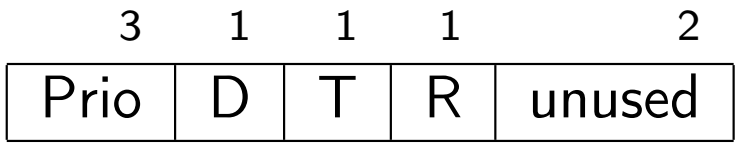
Bemerkung:

- Fragmentierung wird heutzutage eher kritisch gesehen, da es im Kontext von Netzwerkabsicherung mittels Firewalls Probleme bereitet
  (siehe Vorlesung Sicherheit in Rechnernnetzen)
- IPv6 fordert (mandatory) eine MTU von 1280 Bytes.
  $\Longrightarrow$ Verschickt der Absender nur Pakete mit maximal 1280 Bytes, ist keine Fragmentierung erforderlich.

Weiteres Flag-Bit:
Do-not-fragment-Bit: für Testzwecke, um z.B. die MTU eines Netzes herauszubekommen
Falls ein Vermittlungsrechner ein Datagramm fragmentieren muß, aber das Do-not-fragment-Bit gesetzt ist, wird das Datagramm gelöscht und eine ICMP-Nachricht an den Absender geschickt.



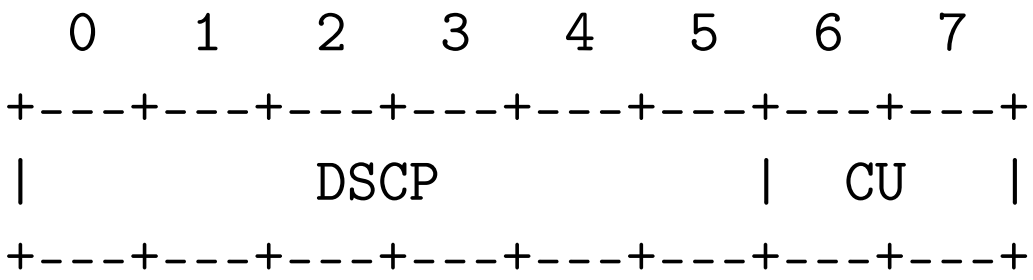
SERVICETYPE: IP wurde von Beginn an mit Unterstützung für Quality of Service entworfen:



D-Bit: low delay
T-Bit: high throughput
R-Bit: high reliability

1998 wurde das Servicetype-Feld umdefiniert:

- RFC 2474, 1998: *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*,



DSCP: differentiated services codepoint
CU: currently unused

Backward Compatability:
xxx000uu $\Longrightarrow$ xxx wird als Priorität interpretiert

aber: Vermittlungsrechner/Router dürfen Servicetype-Feld ignorieren!

IP Routing-Algorithm

gegeben: Wegwahltabelle, Datagram

1.) Bestimme die Ziel-IP-Adresse des Datagramms D und den zugehörigen Netzwerk-Präfix N

2.) **IF** N ist ein direkt angebundenes Netzwerk
Then schicke das Datagramm auf dem zugehörigen Interface raus
ELSE **IF** Wegwahltabelle besitzt einen Eintrag für Ziel-IP
THEN schicke Datagramm zum nächsten Router gemäß Eintrag
ELSE **IF** Wegwahltabelle besitzt einen Eintrag für Netzwerk N
THEN schicke das Datagramm zum nächsten Router gemäß Eintrag
ELSE IF Wegwahltabelle enthält Default-Eintrag
THEN schicke das Datagramm zum Default-Router gemäß Eintrag
ELSE Routing-Error!

Mit welcher Ziel-MAC-Adresse (Schicht 2) wird das Paket verschickt?

Address Resolution Protokoll (ARP)

- RFC 826, 1982, David C. Plummer: An Ethernet Address Resolution Protocol - or - Converting Network Protocol Addresses to 48 Bit Ethernet Address for Transmission on Ethernet Hardware
- IP-Adresse $\mapsto$ MAC-Adresse von Schicht 2
- ARP merkt sich seine gelernten Einträge eine gewisse Zeit lang im ARP Cache.

Wie lernt man Einträge? $\Longrightarrow$ Address Resolution

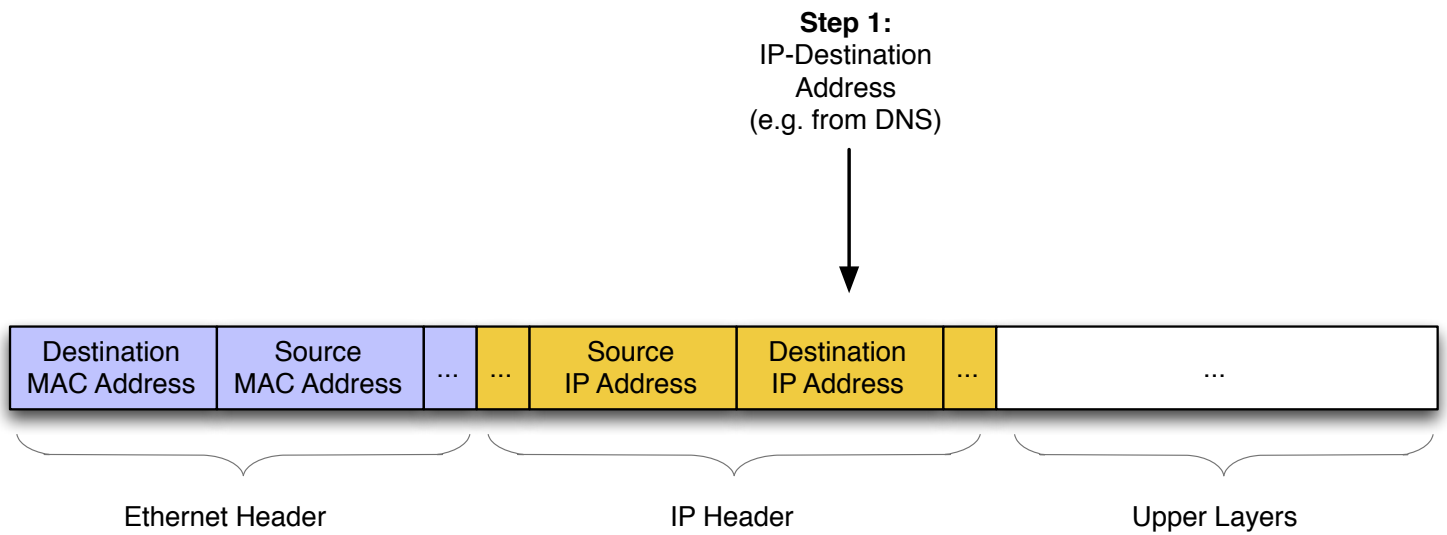
1. ARP Request: Maschine A schickt einen Broadcast „Wer hat IP-Adresse IP_B?“
   Der ARP Request enthält die MAC Adresse MAC_A und die zugehörige IP-Adresse IP_A von A.
2. Alle Empfänger aktualisieren ihren ARP Cache mit (IP_A, MAC_A).
3. Maschine B antwortet mit (IP_B, MAC_B).

IP Packet Processing - Outgoing Packet

What happens when the Kernel sends an IP-Packet?

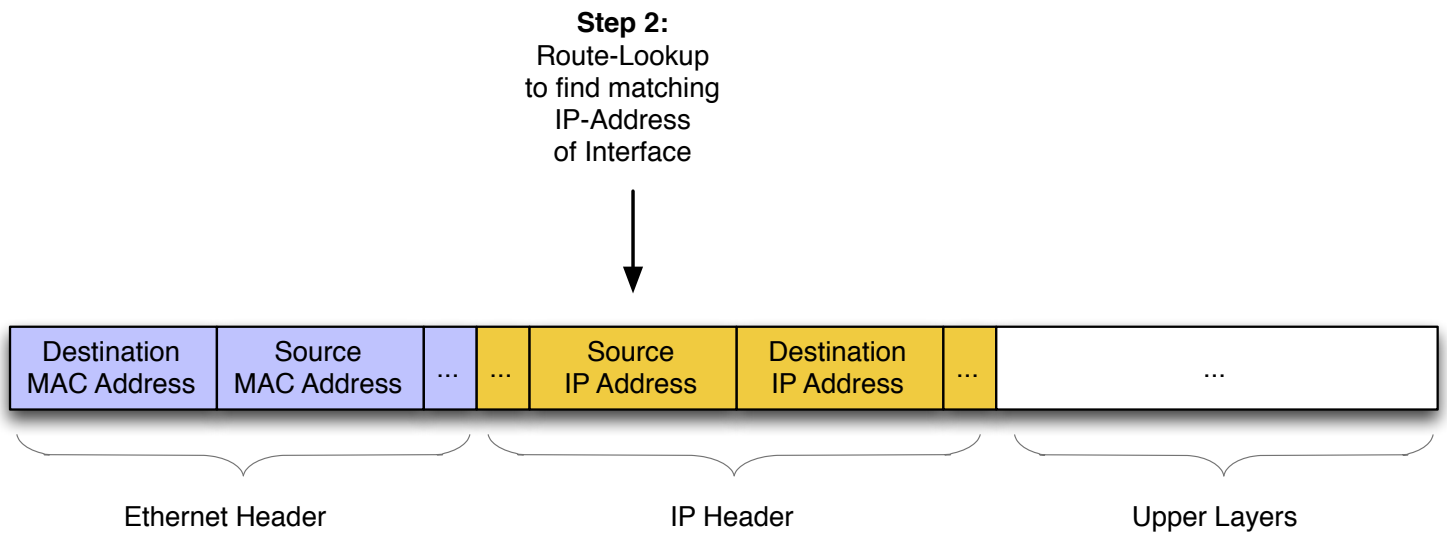
Step 1: **Fill in Destination IP-Address**

- The application supplies the Destination IP-Address (numeric Address or Hostname)
- This usually requires a DNS-Lookup, in order to map a Hostname to an IP-Address



Step 2: **Fill in Source IP-Address**

- If a host has only one configured interface/address, the IP-Address of this interface is used.
- If a host has multiple addresses the kernel performs a Route-Lookup over the routing table to determine the closest matching route to the destination. The IP-Address of the *source* interface of this route becomes the Source IP-Address.

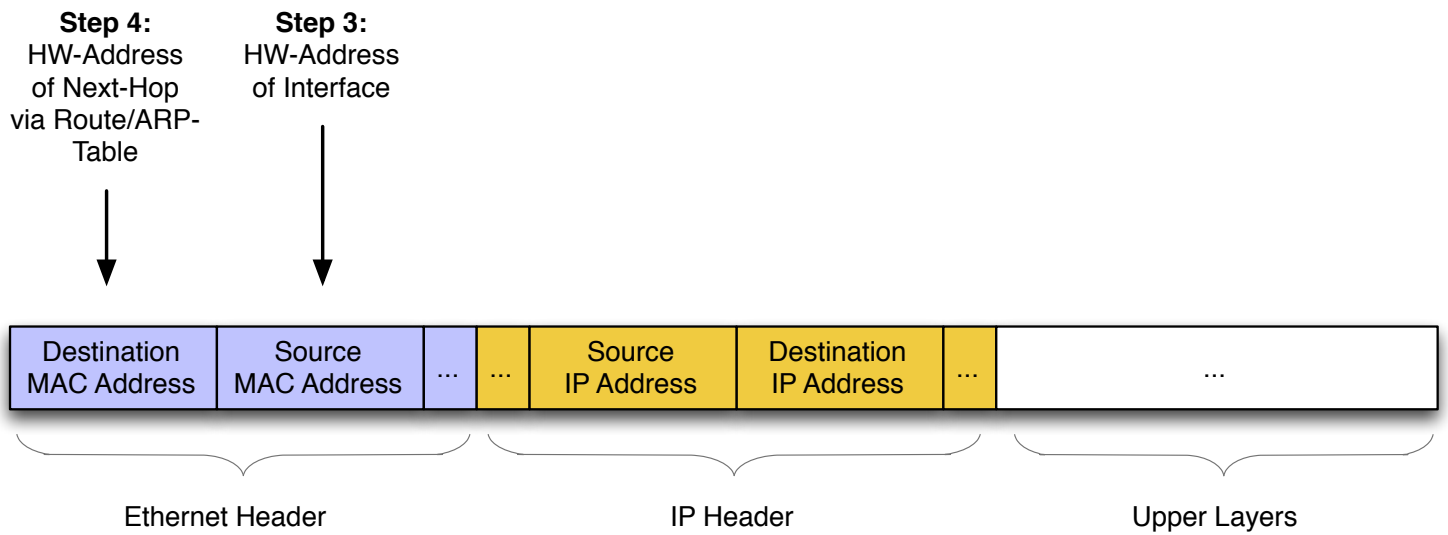


Step 3: **Fill in Source HW-Address**

- The Hardware(MAC)-Address of the Source Interface is used.

Step 4: **Fill in Destination HW-Address**

- The Network Address of the IP-Destination is compared with the local IP-Address (Subnet-Mask) in order to determine if the Destination is **local or remote**.



Local Destination: Packet can be send directly to the Host $\Longrightarrow$ Destination-MAC = MAC_{Host}

Remote Destination: Packet is send to the Next-Hop-Router $\Longrightarrow$ Destination-MAC = MAC_{Router}

Interaction IP and ARP:

- The ARP-Cache is consulted to find the MAC-Address of the L2-Destination Host or Router
- If a corresponding entry is found, it is used as Destination MAC-Address.
- Otherwise an ARP-Request is send as a Broadcast to the local LAN asking the holder of the IP-Address to send back their MAC-Address

What can go wrong?

- Wrong Subnet Mask $\rightarrow$ Leads to wrong L2-Next Hop decision.
- Wrong Gateway Address $\rightarrow$ Packets are not forwarded correctly.
- Outdated ARP-Cache entry $\rightarrow$ Packets are not forwarded correctly.
- No entry in the Routing table $\rightarrow$ Packet will be dropped.
- Wrong entries in the Host routing table $\rightarrow$ Source IP-Address and Next Hop Information may be wrong.
- ...

Einträge aus `/etc/route.conf`, Suse 7.3

```
192.168.0.0     0.0.0.0          255.255.255.0     eth1
141.89.59.0     0.0.0.0          255.255.255.0     eth0
default         141.89.59.254    0.0.0.0           eth0
```

1. **Spalte:** Ziel-Netz
2. **Spalte:** IP vom nächsten Router/Gateway,
   Adresse 0.0.0.0 bedeutet, daß der Zielrechner direkt angeschlossen ist (im LAN)
3. **Spalte:** Netzmaske
4. **Spalte:** ausgehende Interface, auf dem das Paket weitergereicht werden soll

Internet Control Message Protocol (ICMP)

ICMP legt Regeln zur Kommunikation zwischen verschiedenen IP–Instanzen fest zum Austausch von

- Fehlernachrichten (z.B. `destination unreachable`, `route change request`)
- Statusmeldungen (z.B. `echo request`)

ICMP ist Teil von IP. ICMP–Nachrichten werden als IP-Nachricht verschickt.

Wie erkennt IP, dass es sich um ein ICMP-Paket handelt?
protocol $= 1$ im IP-Header $\Longrightarrow$ ICMP–Nachricht

Beispiel: > ping www.google.com

```
schnor@petzi: > ping www.google.com
PING www.google.com (142.250.186.68) 56(84) bytes of data.
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=1 ttl=111 time=28.3 ms
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=2 ttl=111 time=22.5 ms
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=3 ttl=111 time=22.8 ms
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=4 ttl=111 time=22.7 ms
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=5 ttl=111 time=22.8 ms
64 bytes from fra24s05-in-f4.1e100.net (142.250.186.68): icmp_seq=6 ttl=111 time=22.6 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5010ms
rtt min/avg/max/mdev = 22.578/23.654/28.338/2.103 ms
```

`ping` ist ein Kommandozeilen-Tool, das eine Folge von ICMP-Nachrichten vom Typ `Echo Request` schickt und ICMP `Echo Reply` Nachrichten als Antwort bekommt.

$\Longrightarrow$ Verschiedene ICMP-Nachrichten werden über das ICMP Type Feld spezifiziert.

ICMP Type Field	Message
0	Echo Reply
1-2	unassigned
3	Destination Unreachable
4	Source Quench
5	Redirect (change a route)
8	Echo Request
9	Router Advertisement
10	Router Solicitation
11	Time Exceeded for a Datagram
12	Parameter Problem on a Datagram
13	Timestamp Request
14	Timestamp Reply

vollständige Liste: ICMP Parameterlister der IANA

Beispiel:

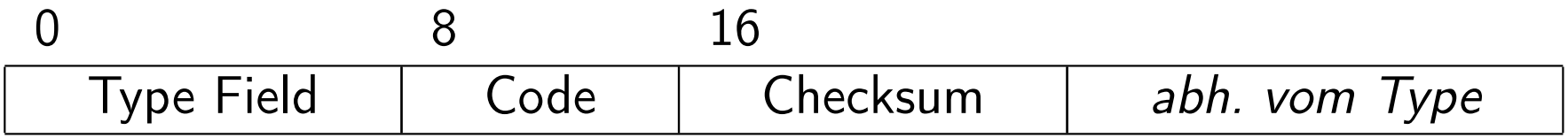
```
traceroute - print the route packets trace to network host
traceroute6
```

`traceroute` *tracks the route packets taken from an IP network on their way to a given host. It utilizes the IP protocol's time to live (TTL) field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to the host.*

Das ICMP Type Feld definiert

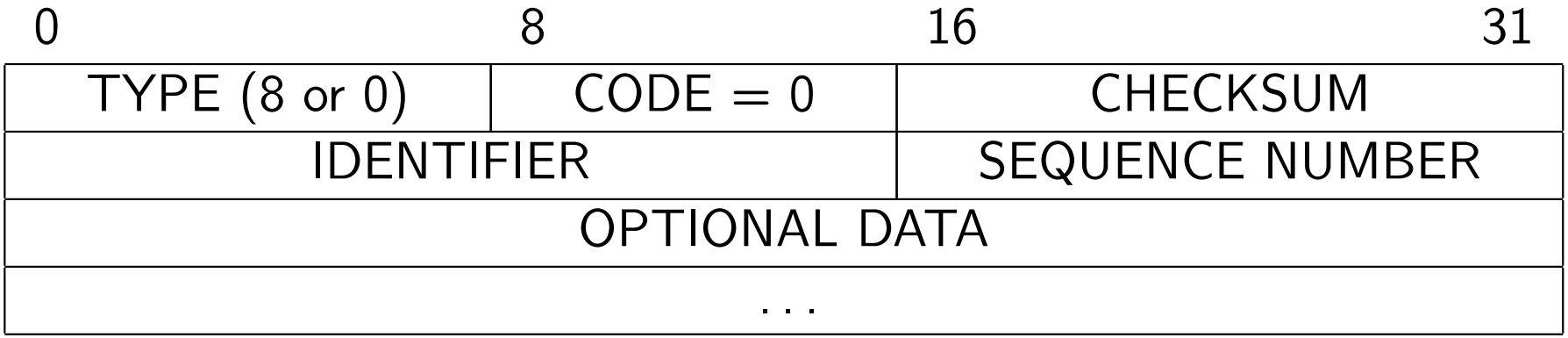
1. Bedeutung der ICMP-Nachricht
2. Format der ICMP-Nachricht

ICMP-Header:



CHECKSUM: wird analog wie bei IP gebildet

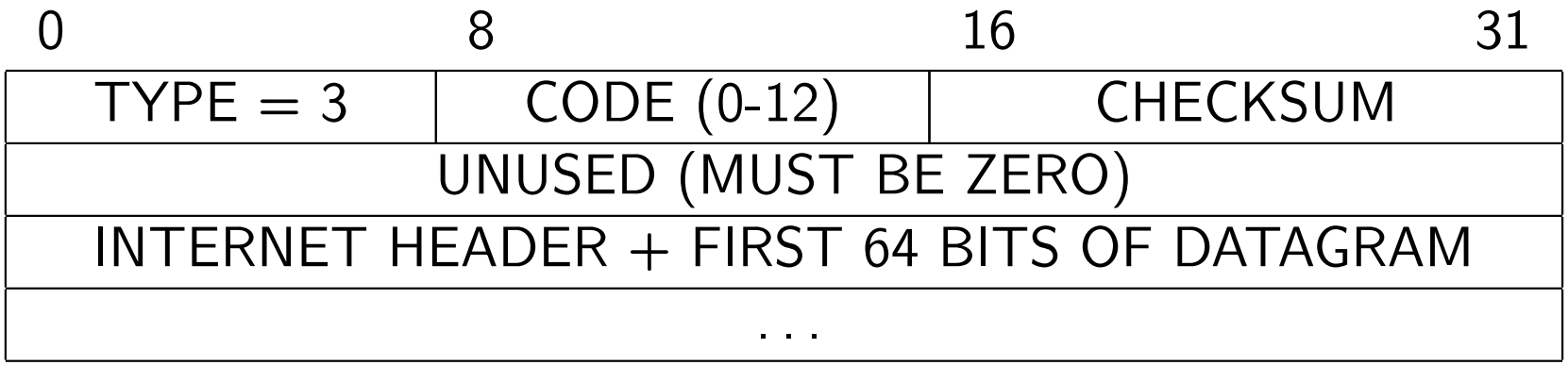
Beispiel: `Echo Request` und `Echo Reply` Format



IDENTIFIER, SEQUENCE NUMBER: benutzt Sender um Anfragen und Antworten einander zuordnen zu können
OPTIONAL DATA: In der Reply-Nachricht werden exakt die Daten aus der Request-Nachricht wiederholt (variable Länge)

Beispiel: ICMP-Nachricht Destination Unreachable

Wenn ein Router ein IP Datagram nicht weiterrouten kann, schickt er eine Destination Unreachable-Nachricht an den Absender zurück.



Das CODE-Feld beschreibt das Problem genauer.

Code Value	Meaning
0	Network unreachable
1	Host unreachable
2	Protocol unreachable
3	Port unreachable
4	Fragmentation needed and DF set
5	Source route failed
6	Destination network unknown
7	Destination host unknown
8	Source host isolated
9	Communication with destination network administratively prohibited
10	Communication with destination host administratively prohibited
11	Network unreachable for type of service
12	Host unreachable for type of service

Was Sie jetzt wissen sollten ...

- Ist Fluten ein statisches oder adaptives Wegwahlverfahren?
- Warum ist es ausreichen, wenn in einer Wegwahltabelle anstatt kompletter Wege nur das zugehörige ausgehende Interface gespeichert wird?
- Was ist ein Autonomes System? Wieso braucht man Interior und Exterior Gateway Protocols?
- Was versteckt sich hinter BGP?
- Was ist die MTU von Ethernet?
- Was sind die Aufgaben von IP?
- Was versteht man unter privaten IP-Adressen? Was ist eine Loopback-Adresse?
- Wie groß ist der minimale IP-Header?
- Wie funktioniert der IP Routing-Algorithmus?
- Wozu braucht man ARP?
- Wozu dient ICMP?