Rechnernetze I

SoSe 2024

Roland Wismüller
Universität Siegen
roland.wismueller@uni-siegen.de
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: 13. Februar 2024

Rechnernetze I

SoSe 2024

5 Internetworking

Inhalt

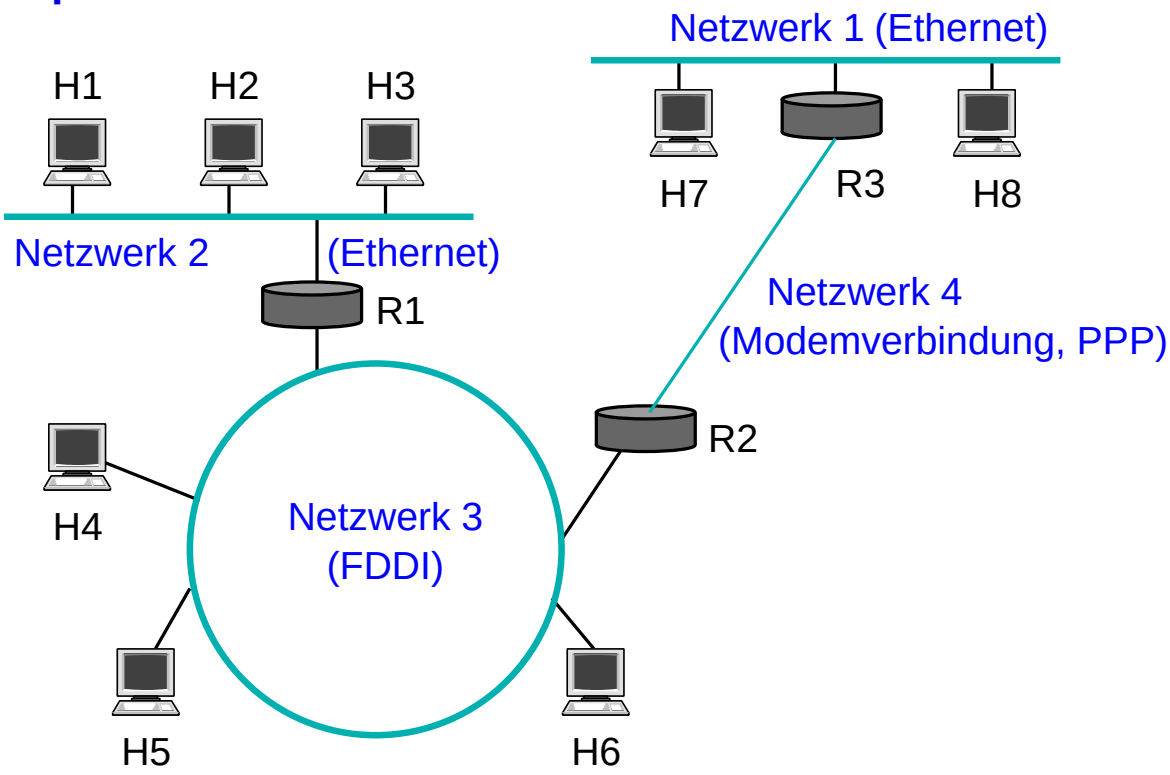
- IP
  - Grundlagen, Adressierung und Weiterleitung, Aufbau eines IP-Pakets, Fragmentierung / Reassembly
- ICMP
- Adreßübersetzung: APR, NDP
- Automatische IP-Konfiguration: DHCP
- NAT
- Tunneling
- Übergang von IPv4 auf IPv6

- Peterson, Kap. 4.1
- CCNA, Kap. 6, 7, 8, 5.3

5 Internetworking ...

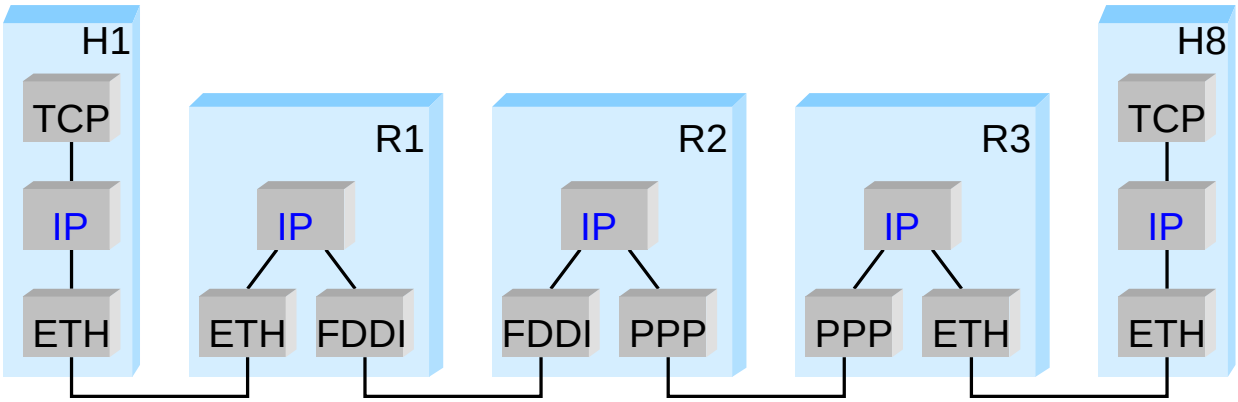
- Was ist ein Internetwork?
  - Zusammenschluß von einzelnen (physischen) Netzen über Router zu einem logischen Netz
    - Netz von Netzen
- Was macht ein Internetwork aus?
  - **Heterogenität**
    - Verbindung unterschiedlichster Netzwerktypen (auch zukünftiger!)
  - **Skalierung**
    - Integration von sehr vielen Rechnern und Netzen
    - Internet: $\geq$ 1 Milliarde Rechner
- Internetwork $\neq$ Internet

Beispiel für ein Internetwork



5.1 IP: Grundlagen

IP (*Internet Protocol*) als Internetwork-Protokoll



- Auf jedem Rechner und jedem Router läuft IP
  - IP kann auf unterschiedlichsten Netztechnologien aufsetzen

Anmerkungen zu Folie 171:

Beachten Sie, daß bei der Weiterleitung eines IP-Pakets (im Beispiel von H1 nach H8) **zwei** verschiedene Adressen notwendig sind. Wenn das Paket z.B. von H1 an R1 weitergegeben wird, ist das IP-Paket selbst an H8 adressiert, der Ethernet-Frame, in dem sich das Paket befindet, jedoch an R1.

Erst wenn R3 das Paket an H8 weitergibt, ist auch der Ethernet-Frame, in dem sich das Paket befindet, an H8 adressiert.

5.1 IP: Grundlagen ...

★★★

IP Dienstmodell (was bietet IP?)

- Adressierungsschema
- Datagramm-Zustellung

- Um Heterogenität und Skalierbarkeit zu unterstützen: kleinster gemeinsamer Nenner
  - IP bietet nur das, was mit jeder Netzwerktechnologie realisiert werden kann
    - „*run over everything*“
  - „*Best Effort*“-Modell:
    - IP „bemüht sich“, gibt aber keinerlei Garantien
    - Verlust, Duplikate, Vertauschung von Paketen möglich
    - höhere Schichten bieten bessere Dienste

IP-Versionen: IPv4 und IPv6

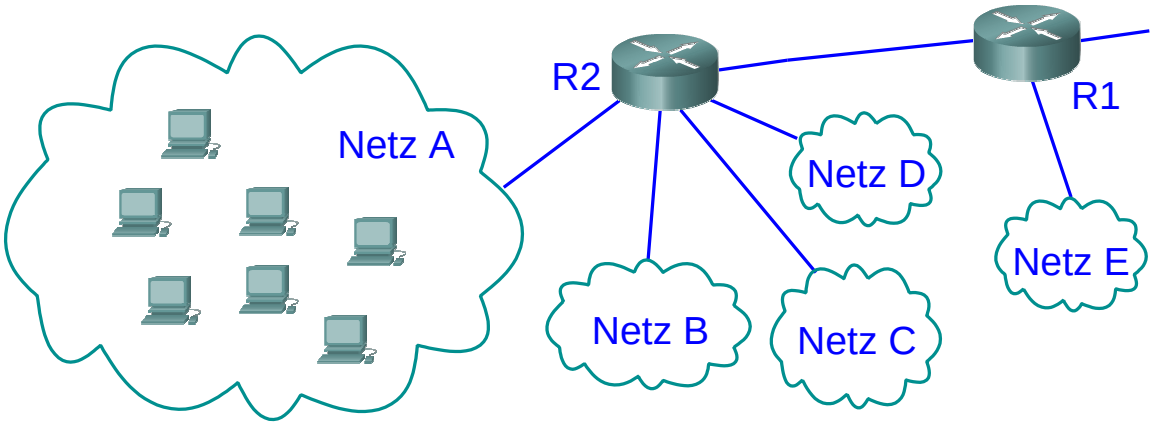
- IPv4 ist seit 1980 standardisiert
- Motivation für IPv6: Wachstum des Internets
  - größerer Adreßraum für IP-Adressen notwendig
- Arbeit an IPv6 seit ca. 1991
  - längere IP-Adressen ⇒ neuer IP-Header ⇒ Anpassung aller IP-Software
  - daher: auch andere Probleme von IPv4 adressiert, u.a.
    - Unterstützung von Dienstgüte-Garantien
    - Sicherheit
    - automatische Konfiguration
    - erweitertes Routing (z.B. mobile Hosts)

5.2 IP: Adressierung und Weiterleitung

(Animierte Folie)

Ziel: Effiziente Weiterleitung von IP-Paketen

- Typische Situation:



- Router müssen Pakete nur in das richtige Netz weiterleiten
  - Weiterleitungstabellen sollten nur Information über Netze enthalten, nicht über einzelne Hosts
  - Router muss aus IP-Adresse zugehöriges Netz bestimmen

Anmerkungen zu Folie 174:

Router R2 sollte in seiner Weiterleitungstabelle nur *einen* Eintrag für das gesamte Netz A haben müssen, statt für jeden einzelnen Host in diesem Netz. Analog hat dann auch R1 nur einen Eintrag für das Netz A.

In der Praxis kann das Netz A (z.B. ein großes Firmennetz) intern in meherere kleinere Netze (z.B. für die einzelnen Abteilungen) unterteilt sein, die über einen Router (R3 in Animationsschitt 2) verbunden sind. Man spricht dabei von *Subnetting*. Idealerweise sollten die Router ausserhalb von Netz A davon nichts wissen müssen, d.h., R2 und R1 haben nach wie vor nur einen Eintrag für das gesamte Netz A, während R3 natürlich für jedes Subnetz einen Weiterleitungstabellen-Eintrag besitzen muß.

Wenn man noch etwas weiter denkt, könnten für R1 alle Netze, die über R2 erreichbar sind (also die Netze A, B, C, D) zu einem einzigen, größeren Netz ABCD zusammengefasst werden, so daß R1 (und alle anderen Router, die diese Netze erreichen können) nur noch einen einzigen Weiterleitungstabellen-Eintrag für diese vier Netze benötigen (Animationsschritt 3). An dieser Stelle spricht man von *Supernetting*.

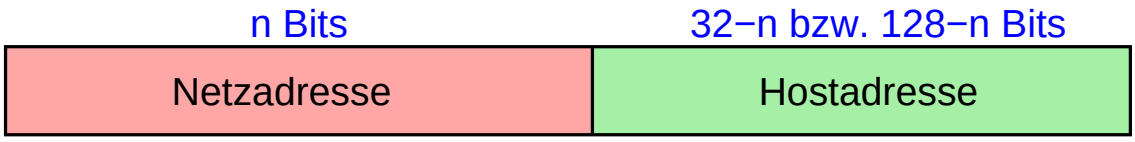
174-1

5.2.1 Adressierung in IP

★★★

Aufgaben bei der Adressierung

- Identifikation von Hosts
  - durch numerische Adresse (IPv4: 32 Bit, IPv6: 128 Bit)
  - hierarchischer Aufbau:



- Identifikation von Netzen
  - durch Netzadresse und Präfixlänge n
  - in IPv4 ursprünglich:
    - n geht aus der Adresse eindeutig hervor (Adressklassen)
  - heute in IPv4 und IPv6: explizite Angabe von n (klassenlose Adressierung, CIDR)

Anmerkungen zu Folie 175:

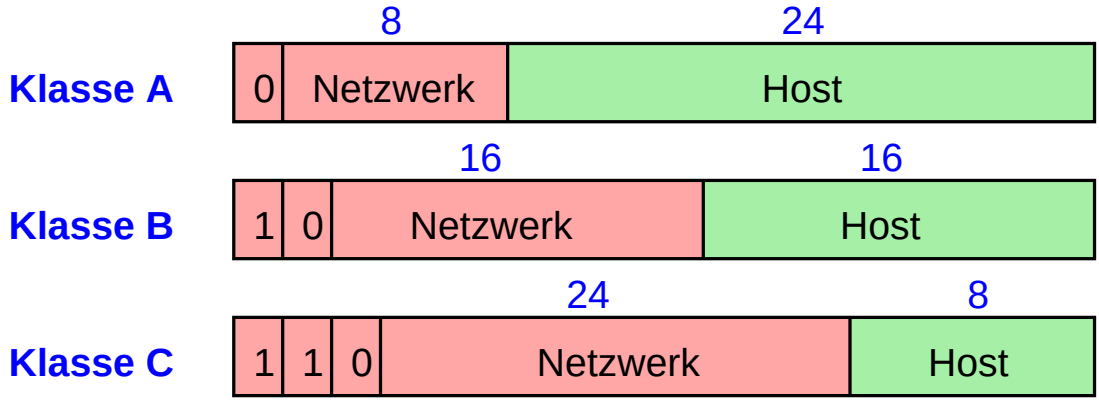
CIDR ist die Abkürzung für *Classless InterDomain Routing*. Die Motivation für die Einführung von CIDR in IPv4 und die Auswirkungen auf die IP-Weiterleitung bzw. das Routing sind im RFC 4632 beschrieben.

Netzadressen werden der Einfachheit halber wie normale IP-Adressen notiert und gespeichert. Per Konvention werden dabei alle Bits, die normalerweise die Hostadresse angeben, auf 0 gesetzt.

5.2.1 Adressierung in IP ...

IPv4 Adressen

- Adressklassen:



- Schreibweise: byteweise dezimal, durch Punkt getrennt
  - z.B. 131.159.31.17
- Bei klassenloser Adressierung ggf. mit Angabe der Präfixlänge
  - z.B. 131.159.31.17/16 oder 131.159.30.0/24

Anmerkungen zu Folie 176:

- IPv4-Adressen werden byteweise dezimal aufgeschrieben, wobei die einzelnen Byte-Werte durch Punkte getrennt werden. Z.B.: 131.159.32.17
- Die klassenbasierte Adressierung erlaubt 128 Netzwerke der Klasse A mit jeweils max. 2^{24}-2 Hosts, 16384 Klasse-B-Netze mit jeweils max. 65534 Hosts sowie 2^{21} Klasse-C-Netze mit max. 254 Hosts.
- In jedem Netz sind immer zwei spezielle Werte für den Hostteil reserviert:
  - Sind alle Bits im Hostteil einer Zieladresse auf 1 gesetzt, z.B. in der Adresse 131.159.255.255 (Klasse B), so bedeutet dies einen Broadcast im angegebenen Netz (hier 131.159.0.0). Die Adresse 255.255.255.255 als Zieladresse bewirkt immer einen Broadcast im lokalen Netz.
  - Die Kombination „alle Bits auf 0“ ist immer reserviert, um die Netzadresse eindeutig bezeichnen zu können (z.B. 131.159.0.0)
- Ab 1993 wurde die klassenlose Adressierung eingeführt, bei der die Grenze zwischen Netzwerk- und Hostteil beliebig gesetzt werden kann. Da die Länge des Netzwerkteils damit nicht mehr eindeutig aus der Adresse hervorgeht, wird sie (z.B. bei Netzadressen oder auch bei Hostadressen, wenn das Netzwerk mit aus der Angabe hervorgehen soll) explizit mit angegeben, z.B. 131.159.0.0/23

176-1

5.2.1 Adressierung in IP ...

★

IPv6 Adressen

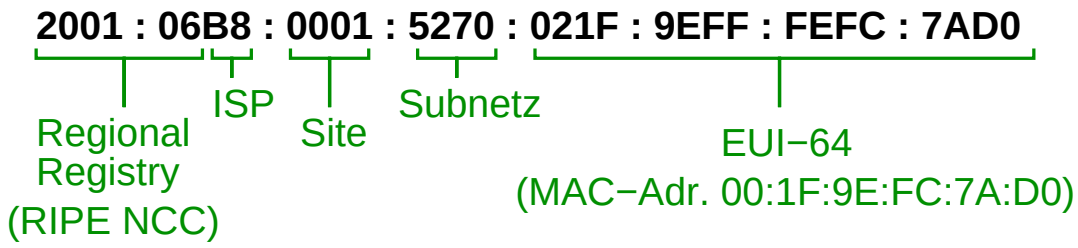
- Typische Struktur:

48	16	64
Global Routing Prefix	Subnet ID	Interface ID

- Schreibweise:
  - 16-Bit-Teile hexadezimal, getrennt durch ':'
    - z.B. 47CD:0000:0000:0000:0000:1234:A456:0124
  - Kurzform
    - ohne führende Nullen, eine Nullfolge durch '::' ersetzt
    - z.B. 47CD::1234:A456:124
- Ggf. mit Angabe der Präfixlänge, z.B. 2000::/3

Adreßvergabe in IPv6

- Routing-Präfix: hierarchisch, i.W. nach Regionen
  - IANA vergibt Präfixe an *Regional Registries*, diese vergeben längere Präfixe an Provider, diese noch längere an Kunden
  - ermöglicht Aggregation von Routing-Information
- Interface-ID: manuell vergeben oder EUI-64 (∼ MAC-Adresse)
- Beispielstruktur einer IPv6-Adresse:



Anmerkungen zu Folie 178:

- Die IANA (*Internet Assigned Numbers Authority*) ist zuständig für die globale Koordinierung von Internet-Ressourcen, z.B. IP-Adressen. Sie vergibt Adressbereiche an die *Regional Registries*, z.B. ARIN für Nordamerika und RIPE NCC für Europa.
- Die im Beispiel gezeigte Struktur wurde im Wesentlichen in den Internet-Standards RFC 2073 und RFC 2374 so festgelegt. Später wurde im RFC 3587 diese Festlegung de facto in weiten Teilen rückgängig gemacht, da Implementierungen die Struktur nicht ausnutzen sollten. Geblieben ist die auf Folie 177 gezeigte Einteilung in „Global Routing Prefix“, „Subnet ID“ und „Interface ID“, wobei der Standard die Längen nicht verbindlich vorschreibt. Für IPv6-Adressen, die nicht mit 000 (binär) beginnen, ist die Länge der Interface-ID aber auf 64-Bit festgelegt (siehe dazu auch RFC 3513).
- EUI-64 ist ein von der IEEE standardisiertes Adressformat, das das EUI-48 Format für MAC-Adressen erweitert.
- Im Beispiel zu EUI-64 ist die MAC-Adresse 00:1F:9E:FC:7A:D0. Diese wird durch Einschieben von FFFE in der Mitte auf 64-Bit erweitert. Zusätzlich muss noch das „universal/local“ (U/L) Bit, das ist das Bit mit der Wertigkeit 2 im ersten Byte invertiert werden (zur Begründung: siehe Abschnitt 2.5.1 in RFC 4291), aus 00:1F wird daher 021F.

Gültigkeitsbereiche von IP-Adressen

- **Global**
  - weltweit eindeutig
  - Pakete mit diesen Adressen werden im globalen Internet weitergeleitet
- **Link Local**
  - nur innerhalb eines physischen LANs eindeutig
  - Router leiten Pakete mit diesen Adressen nicht weiter
- **Private** (IPv4) bzw. **Unique Local** (IPv6)
  - nur innerhalb eines privaten Netzes eindeutig
    - z.B. eine Organisation
  - Pakete mit diesen Adressen werden nicht im globalen Internet weitergeleitet

Anmerkungen zu Folie 179:

- Da link-lokale Adressen nur innerhalb eines physischen LANs gültig sind, können Schnittstellen mit diesen Adressen nicht aus anderen Netzen aus angesprochen werden. Eine mögliche Anwendung sind z.B. Netzwerkdrucker, die typischerweise nur aus dem lokalen Netz heraus erreichber sein sollen.
- Da private IP-Adressen nicht weltweit eindeutig sind, kann ein privates IP-Netz nicht ohne Zusatzmaßnahmen (NAT, siehe 5.8) an das öffentliche Internet angeschlossen werden.
- Bei *Unique Local* Adressen beinhaltet das *Global Routing Prefix* eine 40-Bit lange Zufallszahl. Dadurch wird erreicht, daß trotz unkoordinierter und dezentraler Adreßvergabe die Netzwerkadresse sehr wahrscheinlich auch global eindeutig ist. Dadurch kann man einige Probleme privater Netze vermeiden (z.B., wenn zwei private Netze verbunden werden sollen). Siehe RFC 4193.

Arten von IP-Adressen

- **Unicast**: Adresse für genau eine Netzwerk-Schnittstelle
  - aber: in IPv6 hat eine Schnittstelle i.d.R. mehrere Adressen
- **Multicast**: Adresse für eine Gruppe von Empfängern
- **Broadcast** (nur IPv4): alle Schnittstellen innerhalb eines Netzes
  - Adresse 255.255.255.255: Broadcast im lokalen Netz
- **Anycast**: nächstgelegene Schnittstelle aus einer Menge
  - mehrere Hosts mit identischer Adresse und gleicher Funktion
  - Router leiten Pakete zum nächstgelegenen Host weiter
  - Anwendung z.B. für DNS-Server

Spezielle Adreß-Bereiche in IP

Bedeutung	IPv4	IPv6
Global Unicast	0.0.0.0 - 223.255.255.255	2000::/3
Link Local Unicast	169.254.0.0/16	FE80::/10
Unique Local Unicast		FC00::/7
Private Adressen	10.0.0.0/8 172.16.0.0/12 192.168.0.0/16	
Multicast	224.0.0.0/4	FF00::/8
Loopback	127.0.0.0/8	::1

Anmerkungen zu Folie 181:

- Die Adressbereiche wurden soweit möglich durch eine Adresse und die Präfixlänge angegeben, z.B.:
  - 169.254.0.0/16 = 169.254.0.0 - 169.254.255.255
  - 10.0.0.0/8 = 10.0.0.0 - 10.255.255.255
  - 172.16.0.0/12 = 172.16.0.0 - 172.31.255.255
  - 224.0.0.0/4 = 224.0.0.0 - 239.255.255.255
- Neben den in der Tabelle genannten gibt es noch etliche weitere Adressbereiche, die für spezielle Zwecke reserviert sind, u.a.:
  - 0.0.0.0/8 – U.a. für Default-Route in Routingtabellen
  - 192.0.2.0/24 – TEST-NET: nur für Beispiele und Dokumentation
- In IPv4 wird typischerweise die Adresse 127.0.0.1 als Loopback-Adresse verwendet. Eine Loopback-Adresse verweist immer auf den eigenen Host zurück.
- Die Loopback-Adresse ::1 in IPv6 lautet ausgeschrieben: 0000:0000:0000:0000:0000:0000:0000:0001
- Für Anycast gibt es keinen eigenen Adreß-Bereich. Stattdessen verwendet man normale Global Unicast Adressen, die in diesem Fall an mehr als eine Schnittstelle zugewiesen werden.

181-1

5.2.2 IP-Weiterleitung

★★

Grundlagen

- Jedes IP-Datagramm enthält IP-Adresse des Ziels
  - Netzadresse kennzeichnet das physisches Netz des Ziels
- Hosts mit gleicher Netzadresse kommunizieren direkt über ihr lokales Netz
- Jedes physische Netz, das Teil des Internets ist, ist mit mindestens einem **Router** verbunden
  - Router hat mehrere Netzwerk-Schnittstellen
    - jede Schnittstelle hat ihre eigene IP-Adresse
    - ***Gateway***: Schnittstelle eines Routers im lokalen Netz
- Aufgabe bei der Weiterleitung:
  - an welche Schnittstelle muß ein IP-Paket mit gegebener Zieladresse weitergeleitet werden?

★★★

Routing-Tabelle (Weiterleitungstabelle)

- Weiterleitung wird durch Routing-Tabelle gesteuert
  - in Routern und auch in normalen Hosts
- Prinzipieller Aufbau der Tabelle:

Netzadresse$_1$	*Präfixlänge*$_1$	*Next Hop*$_1$
Netzadresse$_2$	*Präfixlänge*$_2$	*Next Hop*$_2$
...	...	...

- Netzadresse und Präfixlänge zusammen identifizieren ein Netz
  - bei IPv4 statt Präfixlänge ggf. auch Subnetzmaske (☞ **S. 187**)
- *Next Hop* = Router bzw. Schnittstelle, an den/die das Paket weitergegeben werden soll, falls Ziel im angegebenen Netz liegt

Anmerkungen zu Folie 183:

- In Tabelleneinträgen für lokale Netze, an die der Router/Host direkt angeschlossen ist, spezifiziert *Next Hop* die Schnittstelle, über die dieses lokale Netz erreichbar ist.

  In der Weiterleitungstabelle eines **Hosts** findet sich dabei immer ein Eintrag für das lokale Netz des Hosts. Dieser sorgt dafür, daß IP-Pakete, die an ein Ziel im lokalen Netz adressiert sind, über die entsprechende Schnittstelle direkt an das Ziel versendet werden und nicht an einen Router.
- In Tabelleneinträgen für Netze, an die der Router/Host nicht direkt angeschlossen ist, spezifiziert *Next Hop* typischerweise die IP-Adresse des Routers, an die Pakete für dieses Netz weitergeleitet werden sollen.

★★★

Vorgehensweise bei der Weiterleitung

- Algorithmus:
  - suche Eintrag i mit größter *Präfixlänge*$_i$, für den gilt:
    - Zieladresse und *Netzadresse*$_i$ stimmen in den ersten *Präfixlänge*$_i$ Bits überein
    - d.h. Zieladresse liegt in dem durch *Netzadresse*$_i$ und *Präfixlänge*$_i$ gegebenen Netz
  - falls Eintrag gefunden: Weiterleiten an *NextHop*$_i$
  - sonst: Verwerfen des Pakets
- Typisch: zusätzlicher Tabellen-Eintrag für Default-Route

0.0.0.0	*0*	*Next Hop*$_{\text{default}}$

  - dieser Eintrag „paßt“ auf jede Zieladresse

Anmerkungen zu Folie 184:

Da jede Adresse in den ersten 0 Bits mit der Adresse 0.0.0.0 (IPv4) bzw. ::0 (IPv6) übereinstimmt, passt der Eintrag für die Default-Route auf jede Ziel-IP-Adresse. Da der Eintrag aber die kürzest kürzest mögliche Präfix-Länge hat (nämlich 0), wird er nur dann verwendet, wenn es keinen anderen passenden Eintrag in der Weiterleitungstabelle gibt.

★★★

Beispiel

	Netzadresse	Subnetzmaske	Next Hop
0	0.0.0.0	0.0.0.0	10.0.0.1
1	141.99.0.0	255.255.0.0	10.1.0.1
2	141.99.179.0	255.255.255.0	Interface 1
3	141.99.128.0	255.255.192.0	10.3.4.1

- IP-Paket mit Ziel 141.99.178.6
  - 0: 141.99.178.6 AND 0.0.0.0 = **0.0.0.0** $(n = 0)$
  - 1: 141.99.178.6 AND 255.255.0.0 = **141.99.0.0** $(n = 16)$
  - 2: 141.99.178.6 AND 255.255.255.0 = **141.99.178.0**
  - 3: 141.99.178.6 AND 255.255.192.0 = **141.99.128.0** $(n = 18)$
- Weiterleitung an Router 10.3.4.1!

Anmerkungen zu Folie 185:

Der Router bestimmt für jeden Eintrag die UND-Verknüpfung der Ziel-IP-Adresse mit der jeweiligen Subnetzmaske. Stimmt das Ergebnis mit der Netzadresse des Eintrags überein, haben wir einen Treffer.

Am Ende wird der Treffer verwendet, der die größte Präfixlänge (oder gleichbedeutend, die numerische größte Subnetzmaske) besitzt.

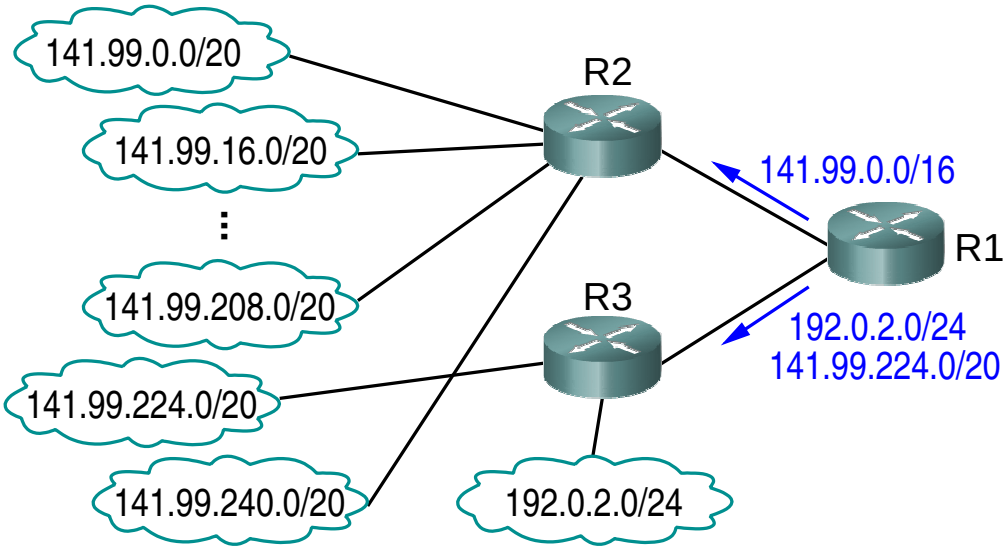
Zum Eintrag 3:

- 178 = 10110010_2, 192 = 11000000_2.
- Damit 178 AND 192 = 10110010_2 AND 11000000_2 = 10000000_2 = 128.
- Die anderen Bytes sind problemlos, da für alle x:
  - x AND 255 = x
  - x AND 0 = 0

(Animierte Folie) ★

Motivation für die Suche nach dem längsten Präfix

- Erlaubt überlappende Einträge in der Routing-Tabelle
- Damit auch Zusammenfassung „nicht zusammenhängender“ Adreßbereiche möglich:



5.2.3 Subnetting

★★★

Bildung von Subnetzen

- Motivation: Unterteilung eines großen Netzes (z.B. Firmennetz) in meherere kleinere Netze (z.B. Abteilungsnetze)
  - nach „aussen“ hin ist nur das Gesamtnetz sichtbar
- IPv4: ein Teil der Host-Bits wird für die Subnetz-ID „geborgt“

Netzwerk−Adresse	Host−Adresse	Klasse−B−Adresse (/16)
11111111 11111111 11111111	00000000	Subnetz−Maske (255.255.255.0 bzw. /24)
Netzwerk−Adresse	Subnetz−ID / Host−Adr.	Adresse mit Subnetz

  - i.a. werden Subnetze unterschiedlicher Größe erzeugt
  - Subnetzmaske legt Präfixlänge für das Subnetz fest
- IPv6: Subnetz-ID in eigenem 16-Bit-Feld (☞ **S. 177**)

Anmerkungen zu Folie 187:

Man kann bei **IPv4** mehrere Arten des *Subnettings* unterscheiden:

- Beim einfachen *Subnetting* besitzen alle Subnetze eines Netzes dieselbe Subnetzmaske. Wenn die Subnetzmaske k Bits für die Subnetz-ID bereitstellt und m Bits für die Hostadresse verbleiben, wird das Netz also in 2^k Subnetze der Größe 2^m-2 unterteilt (zwei Werte sind für die (Sub-)Netz- und die Broadcast-Adresse reserviert).
- Beim hierarchischen Subnetting kann man ein Subnetz durch eine „längere“ Subnetzmaske weiter unterteilen. Ein Subnetz mit Subnetzmaske 255.255.255.0 kann z.B. mit einer Subnetzmaske 255.255.255.128 in zwei Sub-Subnetze unterteilt werden.
- Beim VLSM (*Variable Length Subnet Mask*) *Subnetting* wird ein Netz durch Subnetzmasken unterschiedlicher Länge in verschieden große Subnetze unterteilt.

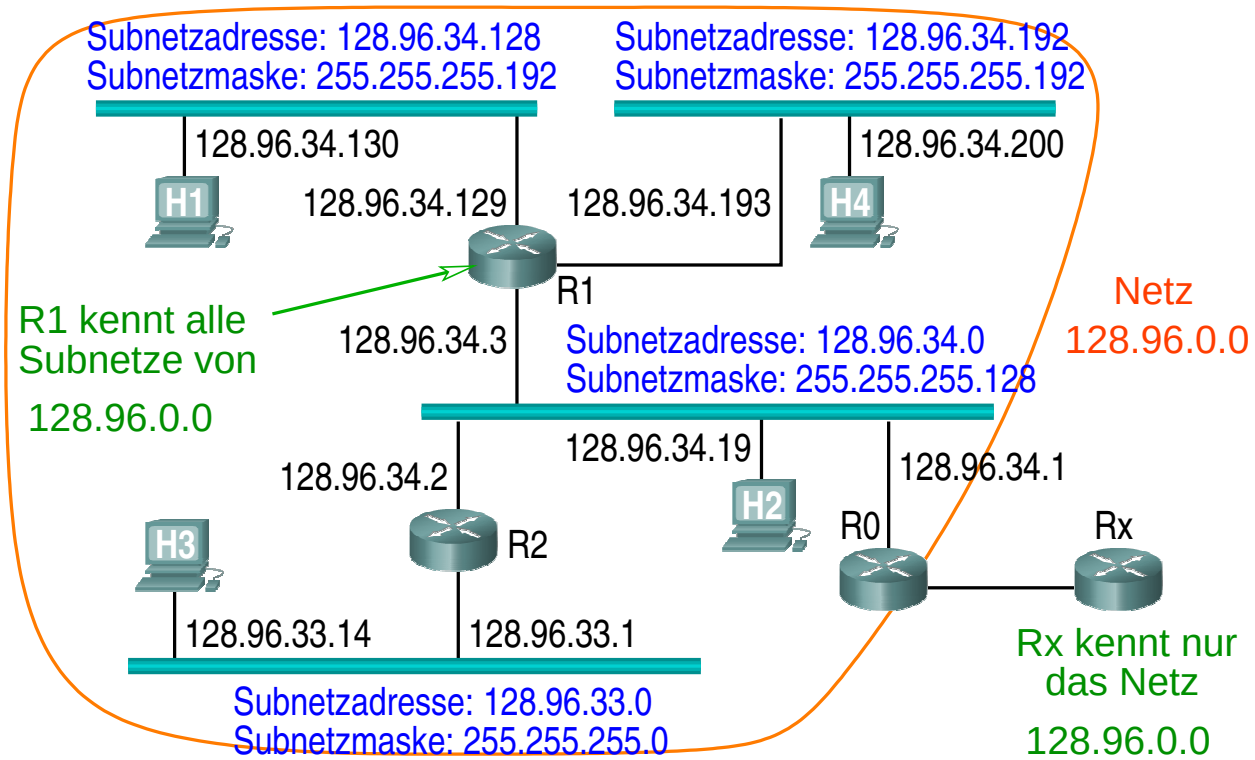
Im Endeffekt wird ein Subnetz aber immer wie auf Folie 175 beschrieben durch seine (Sub-)Netzadresse und die Präfixlänge identifiziert, wobei die Präfixlänge direkt aus der Subnetzmaske hervorgeht (Zahl der 1-Bits in der binären Subnetzmaske). Häufig wird daher auch bei IPv4 statt der (Sub-)Netzmaske einfach die Präfix-Länge angegeben. Z.B. wird ein Subnetz 128.96.34.0 mit Subnetzmaske 255.255.255.128 auch als 128.96.34.0/25 beschrieben.

Bei **IPv6** ist das Subnetting wesentlich einfacher, da man sich um die Größe des Subnetzes keine Gedanken machen muss (jedes Subnetz kann aufgrund der 64 Host-Bits über 10^{19} Hosts aufnehmen), so daß man die Subnetze einfach nur durchnumerieren muß. Die aufgrund der 16 Bit großen Subnetz-ID maximal mögliche Zahl von 65536 Subnetzen reicht in der Praxis ebenfalls immer aus.

(Animierte Folie)

★

Ein Netz mit Subnetzen (IPv4):



★★★

Bestimmung der (Sub-)Netzzugehörigkeit

- Wie bestimmt ein Host bzw. Router, ob eine IP-Adresse xyz in einem gegebenen (Sub-)Netz liegt?
- Gegeben: (Sub-)Netzadresse und Präfixlänge n
  - stimmen xyz und Netzadresse in den ersten n Bits überein?
  - (vgl. Weiterleitungsalgorithmus auf S. 184)
- Gegeben: (Sub-)Netzadresse und (Sub-)Netzmaske
  - gilt xyz AND Netzmaske $=$ Netzadresse ?
- Beispiel: 128.96.34.19 AND 255.255.255.128 $=$ 128.96.34.0

Erstellung von Subnetzen in IPv4

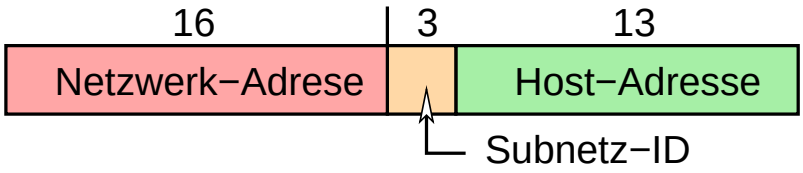
- Einschränkungen durch Realisierung:
  - Zahl der Hostadressen in einem Subnetz ist immer eine Zweierpotenz
    - bei Präfixlänge l des Subnetzes also 2^{32-l}
    - zwei Hostadressen sind reserviert:
      - $0...0_2$: Adresse des Netzwerks selbst
      - $1...1_2$: Broadcastadresse
    - damit $2^{32-l} - 2$ Hosts möglich
  - Subnetz mit 2^k Hostadressen kann nur an einer durch 2^k teilbaren Adresse beginnen
    - z.B. Adressbereich 141.99.179.64 - 141.99.179.191 ist nicht möglich
- Im folgenden: ursprüngliche Präfixlänge sei n

Mögliche Vorgehensweisen

- Einfaches Subnetting
  - feste Anzahl von k Bits des (bisherigen) Hostanteils ($32 - n$ Bits) wird für die Subnetz-ID „geborgt“
  - identische Subnetzmaske für alle Subnetze
  - ergibt 2^k Subnetze mit identischer Größe $2^{32-n-k} - 2$
- Hierarchisches Subnetting
  - bei Bedarf werden einzelne Subnetze weiter unterteilt
  - dabei entstehen längere Subnetzmasken
- VLSM (*Variable Length Subnet Mask*)
  - Subnetze werden von Anfang an entsprechend ihrer Größe durch Subnetzmasken unterschiedlicher Länge realisiert

Beispiel: Einfaches Subnetting

- Im Netzwerk 141.99.0.0/16 sollen 6 Subnetze realisiert werden
- Für 6 Subnetze müssen von den 16 Host-Bits 3 Bits „geborgt“ werden ($6 \leq 2^3$):

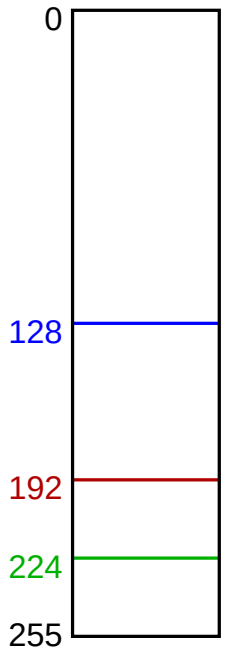


- Es entstehen 8 Subnetze für jeweils $2^{13} - 2 = 8190$ Hosts
  - 141.99.0.0 - 141.99.31.255; 141.99.32.0 - 141.99.63.255; ...; ...; 141.99.224.0 - 141.99.255.255
  - Subnetzmaske 255.255.224.0, Präfixlänge: 19
- Nachteil:
  - Subnetze unterschiedlicher Größe sind nicht möglich

(Animierte Folie)

Beispiel: Hierarchisches Subnetting

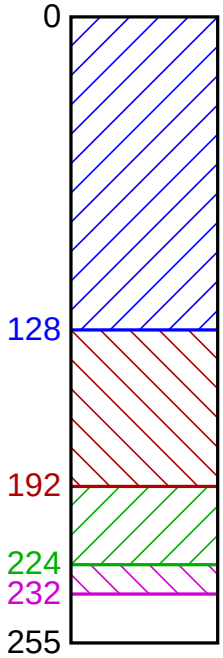
- Im Netzwerk 1.2.3.0/24 sollen 4 Subnetze realisiert werden, die 100, 50, 25 und 5 Hosts aufnehmen können
- Durch Borgen eines Bits entstehen 2 Subnetze:
  - 1.2.3.0 - 1.2.3.127; 1.2.3.128 - 1.2.3.255
  - Netzmaske 255.255.255.128, Präfixlänge: 25
- Weitere Unterteilung des zweiten Subnetzes:
  - 1.2.3.128 - 1.2.3.191; 1.2.3.192 - 1.2.3.255
  - Netzmaske 255.255.255.192, Präfixlänge: 26
- Nochmal Unterteilung des zweiten Subnetzes:
  - 1.2.3.192 - 1.2.3.223; 1.2.3.224 - 1.2.3.255
  - Netzmaske 255.255.255.224, Präfixlänge: 27



(Animierte Folie)

Beispiel: VLSM

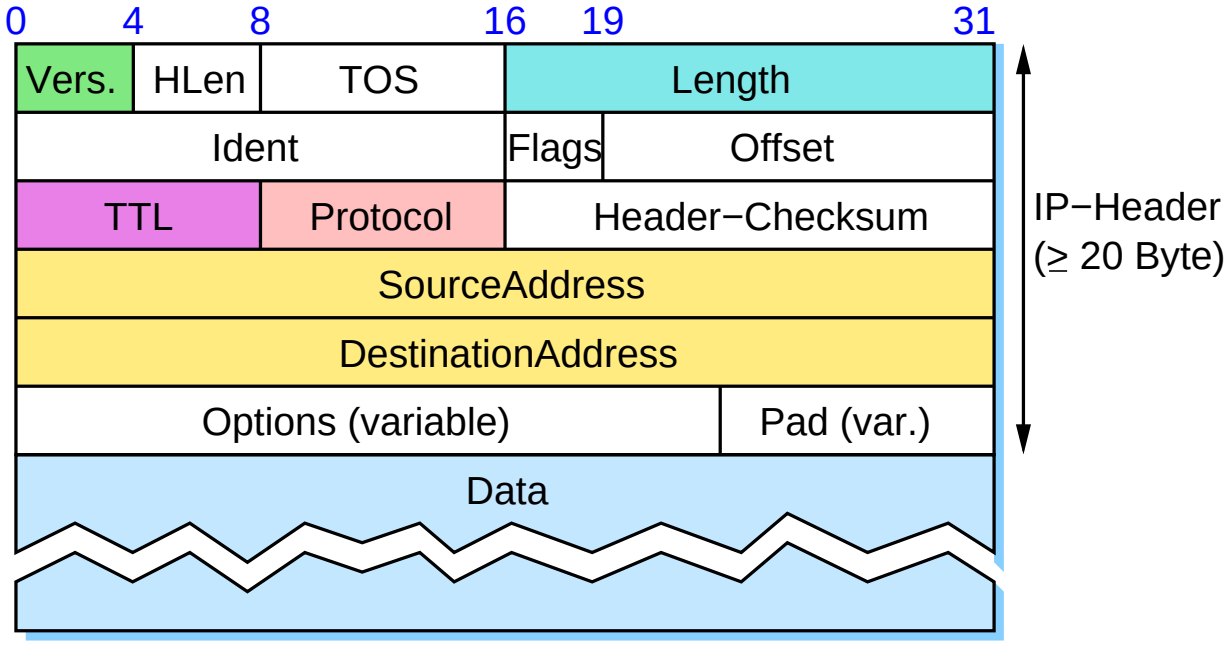
- Im Netzwerk 1.2.3.0/24 sollen 4 Subnetze realisiert werden, die 100, 50, 25 und 5 Hosts aufnehmen können
- Teile möglichst kleine Subnetze in **absteigender** Reihenfolge der Größe zu:
  - 100 Hosts: Größe $128 = 2^7$, P.länge $32 - 7 = 25$
    - Netzadr. 1.2.3.0, N.maske 255.255.255.128
  - 50 Hosts: Größe $64 = 2^6$, P.länge $32 - 6 = 26$
    - Netzadr. 1.2.3.128, N.maske 255.255.255.192
  - 25 Hosts: Größe $32 = 2^5$, P.länge $32 - 5 = 27$
    - Netzadr. 1.2.3.192, N.maske 255.255.255.224
  - 5 Hosts: Größe $8 = 2^3$, P.länge $32 - 3 = 29$
    - Netzadr. 1.2.3.224, N.maske 255.255.255.248



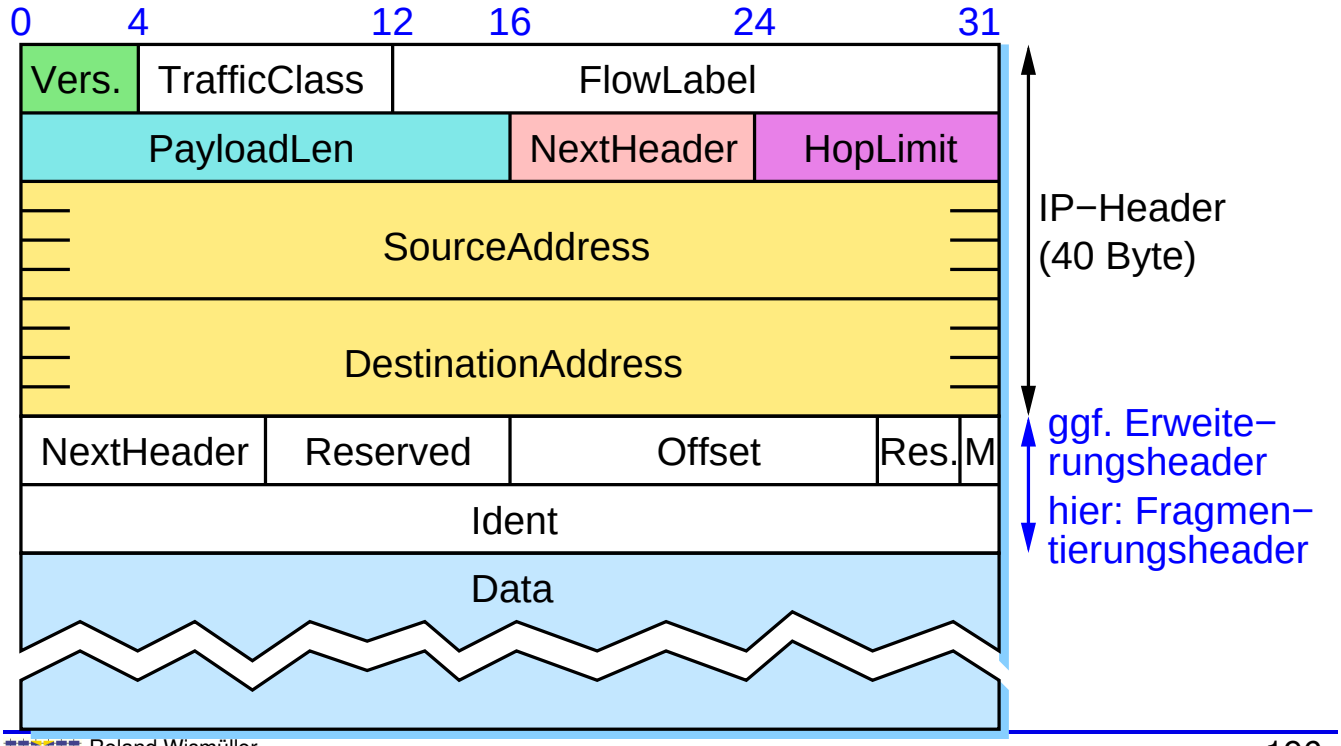
5.3 Aufbau eines IP-Pakets

★

IPv4



IPv6



Anmerkungen zu Folie 196:

- Optionale Felder des IP-Headers sind bei IPv6 nicht mehr im IP-Header selbst gespeichert, sondern werden in Erweiterungsheader ausgelagert, die bei Bedarf auf den eigentlichen IP-Header folgen. Das Feld **NextHeader** gibt jeweils an, welcher Header als nächstes folgt, z.B. ein (weiterer) IP-Erweiterungsheader oder ein TCP-Header, falls der Datenteil des Pakets ein TCP-Segment enthält.
- Für die verschiedenen Erweiterungsheader ist eine verbindliche Reihenfolge festgelegt, um die Bearbeitung beim Empfänger zu vereinfachen.
- Erweiterungsheader sind (derzeit) definiert für:
  - Optionen für Teilstrecken
    - Datagramme mit $>$ 64 KB (für Hochleistungsrechner)
  - Optionen für Zielrechner
  - Routing-Optionen
    - Angabe eines Bereichs, über den Paket geleitet werden soll
  - Fragmentierung (☞ **5.4**)
  - Authentifizierung / Verschlüsselung (☞ **RN_II, Secure IP**)
  - Optionen für Mobilität (☞ **RN_II, Mobile IP**)

Bedeutung der wichtigsten Felder

- **HLen**: Länge des Headers in 32-Bit Worten
- **TOS** / **TrafficClass** / **FlowLabel**: für *Quality of Service*
- **Length**: Gesamtlänge des Pakets inkl. IP-Header in Bytes
  PayloadLen: Länge des Pakets ohne Basis-IP-Header
  - maximal 65535 Bytes
- **Ident** / **Flags (M)** / **Offset**: für Fragmentierung / Reassembly
- **TTL** / **HopLimit**: zur Erkennung endlos kreisender Pakete
  - wird von jedem Router heruntergezählt, bei 0 wird das Paket verworfen
- **Protocol** / **NextHeader**: kennzeichnet das im Datenteil versendete Protokoll (z.B. TCP, UDP; für Demultiplexing)
  oder (bei IPv6) ggf. den Typ des folgenden Erweiterungsheaders

Anmerkungen zu Folie 197:

- Das Feld **Version** (Vers.) enthält die IP-Version (4 oder 6).
- Mit **TOS** bzw. **TrafficClass** kann ein IP-Paket einer bestimmten Verkehrsklasse zugeordnet werden, das **FlowLabel** ordnet das Paket einem bestimmten Datenfluß zu. Beide Angaben werden für *Quality of Service*-Mechanismen verwendet (☞ **RN_II**).
- Das Feld **Flags** bei IPv4 enthält neben dem M-Flag (*more fragments*), das für die Fragmentierung wichtig ist, zwei weitere Flags: eines ist reserviert und hat immer den Wert 0, das andere (DF, *don't fragment*) verbietet, daß das Paket von Routern weiter fragmentiert wird. Bei IPv6 ist dieses Bit nicht notwendig, da eine Weiterfragmentierung eines Pakets prinzipiell verboten ist (☞ **5.4**).
- **TTL** bedeutet eigentlich *Time-To-Live*. Das Feld beinhaltet aber auch bei IPv4 keine Zeitangabe, sondern einen Zähler und wurde daher bei IPv6 umbenannt.
- **Header-Checksum** ist eine einfache Prüfsumme über den IPv4-Header, die wegen der praktisch in allen Schicht-2-Protokollen vorhandenen CRC-Felder überflüssig ist.

5.4 IP: Fragmentierung/Reassembly

Problem für IP

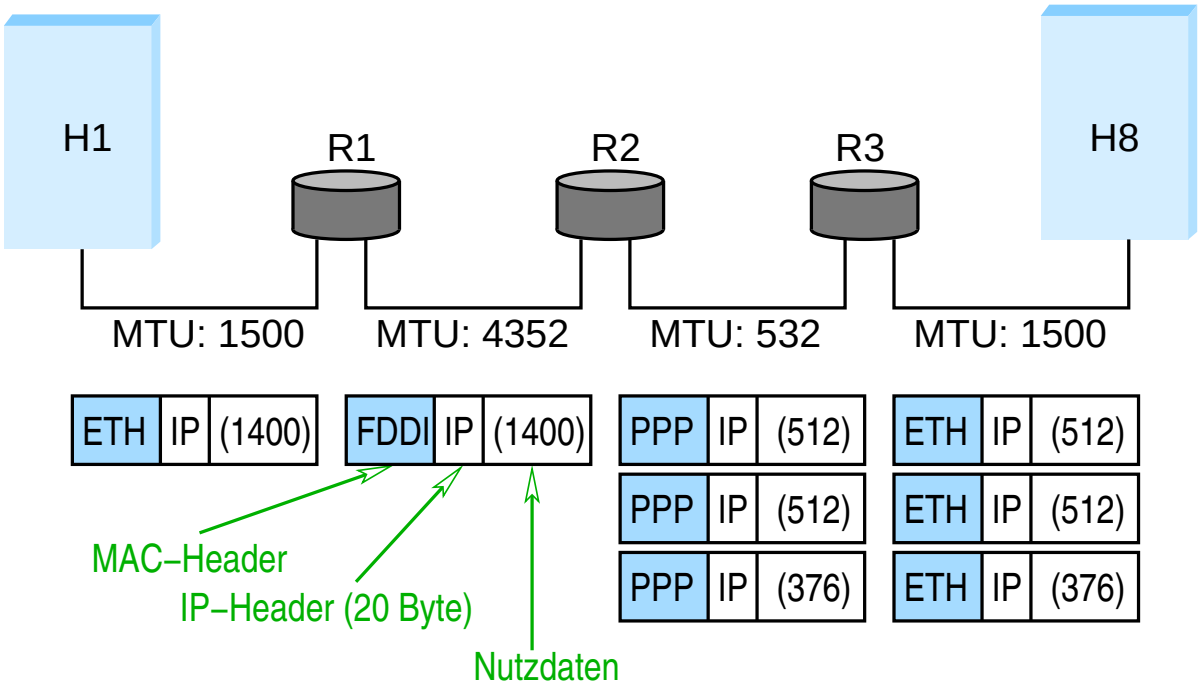
- Jedes lokale Netzwerk definiert eine (unterschiedliche) maximale Framegröße (**MTU**: ***Maximum Transmission Unit***)

Netzwerk	MTU [Byte]
Ethernet	1500
FDDI	4352
4 Mbits/sec Token-Ring (IEEE 802.5)	4464
16 Mbits/sec Token-Ring (IBM)	17914

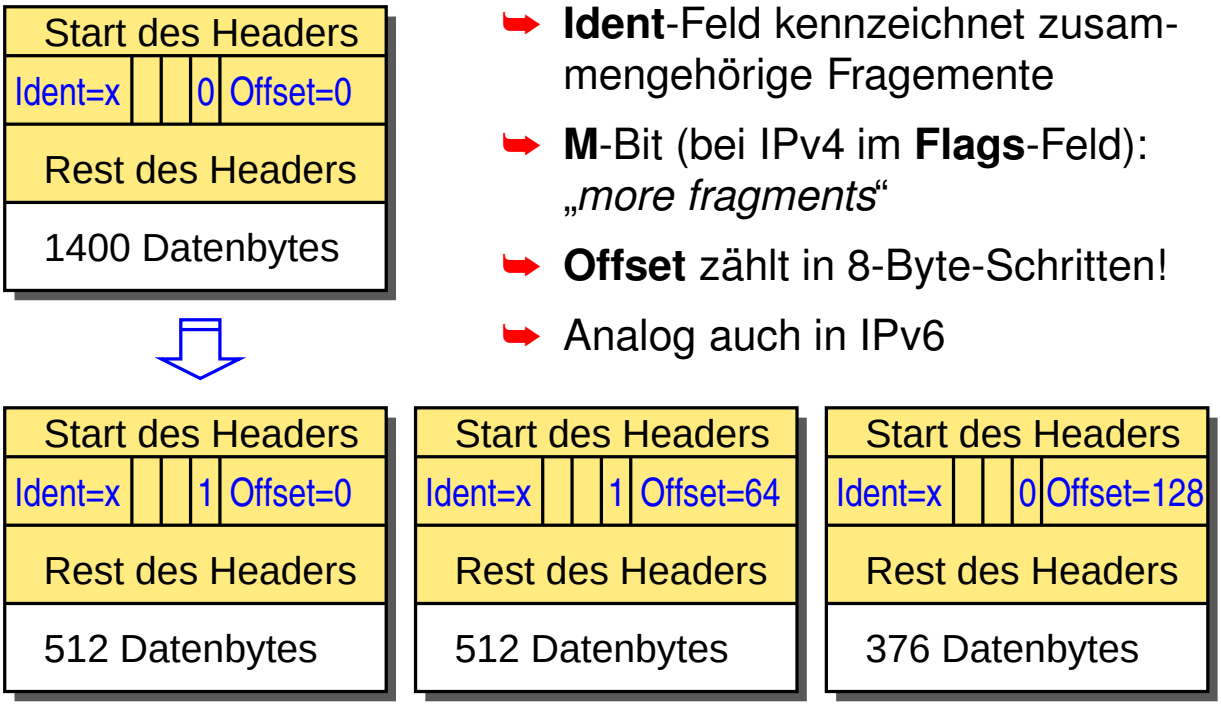
- Alternativen für IP:
  - max. Paketgröße = minimale MTU (welcher Netze??)
  - Möglichkeit der Fragmentierung von IP-Paketen

5.4 IP: Fragmentierung/Reassembly ...

Beispiel zur Fragmentierung (IPv4)



Fragmentierung im Detail (IPv4)



- **Ident**-Feld kennzeichnet zusammengehörige Fragemente
- **M**-Bit (bei IPv4 im **Flags**-Feld): „*more fragments*“
- **Offset** zählt in 8-Byte-Schritten!
- Analog auch in IPv6

Anmerkungen zu Folie 200:

Bei IPv6 wird die Fragmentierungsinformation in einem eigenen Erweiterungs-Header gespeichert:

0	8	16	29	31
NextHeader	Reserved	**Offset**	Res.	**M**
Ident				

★★

Fragmentierung im Detail ...

- Fragmentierung geschieht bei Bedarf im Sender bzw. bei IPv4 auch in den Routern
- Jedes Fragment ist ein eigenständiges IP-Datagramm
  - IPv4: ein Fragment kann ggf. nochmals fragmentiert werden
- Empfänger baut aller Fragmente wieder zusammen
  - falls ein Fragment nicht ankommt, werden alle anderen zugehörigen Fragmente verworfen
- Bei IPv6 und meist auch bei IPv4: „*Path MTU Discovery*“
  - Fragmentierung im Router verboten (IPv4: **DF**-Flag im Header) ⇒ ggf. Fehlermeldung an Sender (über ICMP, siehe später)
  - Sender kann minimale MTU ermitteln

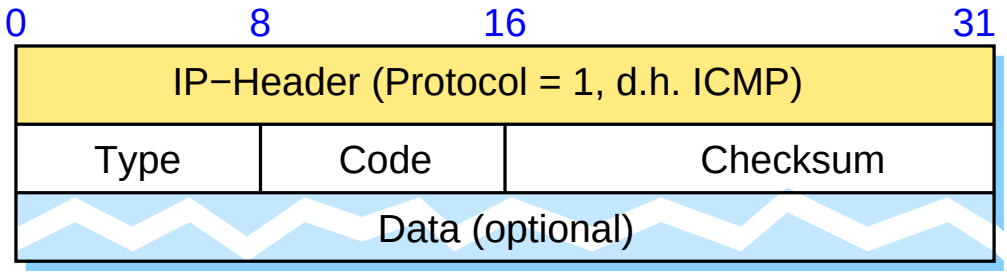
5.5 ICMP

★★

ICMP: *Internet Control Message Protocol*

- Datagramme für Fehler- und Verwaltungsmeldungen:
  - Ziel nicht erreichbar
  - Reassembly fehlgeschlagen
  - Fragmentierung nicht erlaubt, aber erforderlich
  - TTL wurde 0
  - *Redirect*: besserer Router für das Ziel
  - *Echo Request / Reply*: z.B. für `ping` und `traceroute`
  - *Router Solicitation / Advertisement* (nur IPv6): Suche nach / Bekanntgabe von lokalen Routern
  - *Neighbor Solicitation / Advertisement* (nur IPv6): Adreßübersetzung (siehe später)
  - ...

Aufbau eines ICMP-Pakets (siehe auch Wireshark-Aufzeichung)

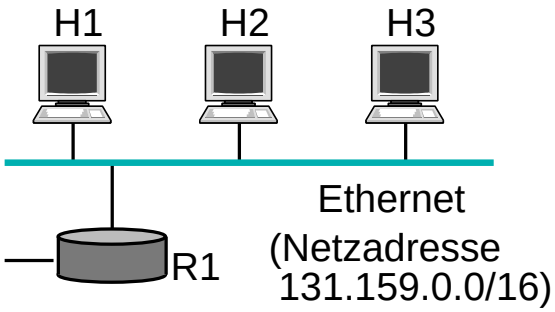


- *Echo* / *Echo Reply* (Typ = 8 / 0):
  - Data: Identifikator + Sequenznummer
- Ziel nicht erreichbar (Typ = 3):
  - Code: z.B. 0 ≙ Netz, 1 ≙ Host, 3 ≙ Port nicht erreichbar
  - Data: IP-Header + erste 64 Datenbytes des unzustellbaren Pakets

5.6 Adressübersetzung

Motivation

- IP-Weiterleitung bringt ein Paket in das richtige LAN
- Wie funktioniert IP-Kommunikation **innerhalb** eines LANs?
- Beispiel:
  - R1 empfängt Paket für Rechner 131.159.32.12
  - R1 muß Paket über Ethernet an diesen Rechner weiterleiten
  - Woher weiß R1 die MAC-Adresse des Rechners mit IP-Adresse 131.159.32.12?



- Anmerkung: dasselbe Problem tritt auch auf, wenn z.B. H1 ein Paket an IP-Adresse 131.159.32.12 senden will

Motivation ...

- Problem: Umsetzung von IP-Adressen auf MAC-Adressen im lokalen Netz
  - allgemein: auf Sicherungsschicht-Adresse
- Lösungs-Alternativen:
  - MAC-Adresse in IP-Adresse kodieren?
    - bei IPv4 nicht realisierbar (Ethernet: 48 Bit MAC-Adresse!)
  - Manuell verwaltete Tabellen?
    - Verwaltungsaufwand!
  - Automatisches (dynamisches) Erstellen der Tabellen!

★★★

ARP: *Address Resolution Protocol* (IPv4)

- Annahme: ein Rechner H will ein Paket an IP-Adresse *xyz* senden, *xyz* ist im lokalen Netz
- H sucht in seinem ARP-Cache nach der zu *xyz* gehörigen MAC-Adresse
- Falls gefunden: Paket an diese MAC-Adresse senden
- Sonst:
  - H sendet Anfrage (*ARP-Request*) per Broadcast in das LAN: „wer hat IP-Adresse *xyz*?“
  - Der betroffene Rechner sendet Antwort (*ARP Reply*) mit seiner IP- und MAC-Adresse zurück
  - H trägt Zuordnung in sein ARP-Cache ein
    - automatische Löschung nach bestimmter Zeit ohne Nutzung

Anmerkungen zu Folie 206:

Jede APR-Nachricht beinhaltet vier Adressfelder (siehe RFC 826):

- *Source Hardware Address* (SHA): die MAC-Adresse des Senders (bei einer ARP-Antwort steht hier die gesuchte MAC-Adresse),
- *Source Protocol Address* (SPA): die IP-Adresse des Senders,
- *Target Hardware Address* (THA): die MAC-Adresse des Empfängers (bei einer ARP-Anfrage mit 00:00:00:00:00:00 belegt),
- *Target Protocol Address* (TPA): die IP-Adresse des Empfängers.

RFC 826 spezifiziert folgendes Verhalten beim Empfang eines ARP-Pakets (unabnhängig davon, ob es sich um eine ARP-Anfrage oder ARP-Antwort handelt):

- wenn SPA im ARP-Cache steht, wird die MAC-Adresse im ARP-Cache auf SHA aktualisiert,
- wenn der Empfänger die IP-Adresse in TPA besitzt, trägt er die Zuordnung (SPA, SHA) in seinen ARP-Cache ein,

In gängigen Betriebssystemen wird offenbar teilweise ein anderes Verhalten implementiert.

206-1

5.6 Adressübersetzung ...

Aufbau eines ARP-Pakets (siehe auch Wireshark-Aufzeichung)

0 … 8	8 … 16	16 … 31
Hardwareadresstyp		Protokolladresstyp
HA−Länge	PA−Länge	Operation
Sender−Hardware−Adresse (z.B. MAC−Adresse)		
Sender−HW−Ad. (MAC−A.)		Sender−Protokoll−Adr. (IP)
Sender−Protokoll−Adr. (IP)		Ziel−HW−Ad. (MAC−A.)
Ziel−Hardware−Adresse (z.B. MAC−Adresse)		
Ziel−Protokoll−Adressre (z.B. IP−Adresse)		

- ARP wird direkt über das Layer-2-Protokoll übertragen
- Typ und Länge der Sender- und Zieladressen sind frei wählbar
- Operation: *request* (1), *reply* (2)

Spezielle Verwendungen von ARP

- ARP *probe*
  - zur Prüfung von Konflikten nach Konfiguration der IP-Adresse
  - Host sendet ARP-Anfrage nach seiner gewählten IP-Adresse
- *Gratuitous* APR
  - Host sendet ARP-Anfrage mit seiner eigenen IP-Adresse als Sender- und Zieladresse
  - alle anderen Hosts aktualisieren ihren ARP-Cache
  - z.B. beim Booten oder bei Umschaltung auf Backup-Server
- *Proxy* ARP
  - Router sendet ARP-Antwort für Anfrage nach einem Host in einem andern Netz
  - anfragender Host schickt IP-Paket dann an Router

Anmerkungen zu Folie 208:

- Der RFC 5227 schreibt vor, daß beim ARP *probe* die Sender-IP-Adresse 0.0.0.0 sein muß, damit andere Hosts nicht eine (ggf. falsche) Zuordnung in Ihren ARP-Cache eintragen.
- Das *Gratuitous* APR Paket ist typischerweise das erste Paket, das ein Rechner nach dem Booten (d.h., sobald er eine IP-Adresse hat) versendet. Durch dieses Paket lernt dann übrigens auch ein Ethernet-Switch sofort, an welchem Port sich der Rechner befindet.
- Proxy-ARP kann z.B. eingesetzt werden, wenn ein Netz in mehrere Subnetze geteilt ist, die einzelnen Hosts dies aber nicht wissen (sollen). Der Router erkennt dann anhand der Ziel-IP-Adresse, dass diese in einem anderen Subnetz liegt und antwortet mit seiner eigenen MAC-Adresse.

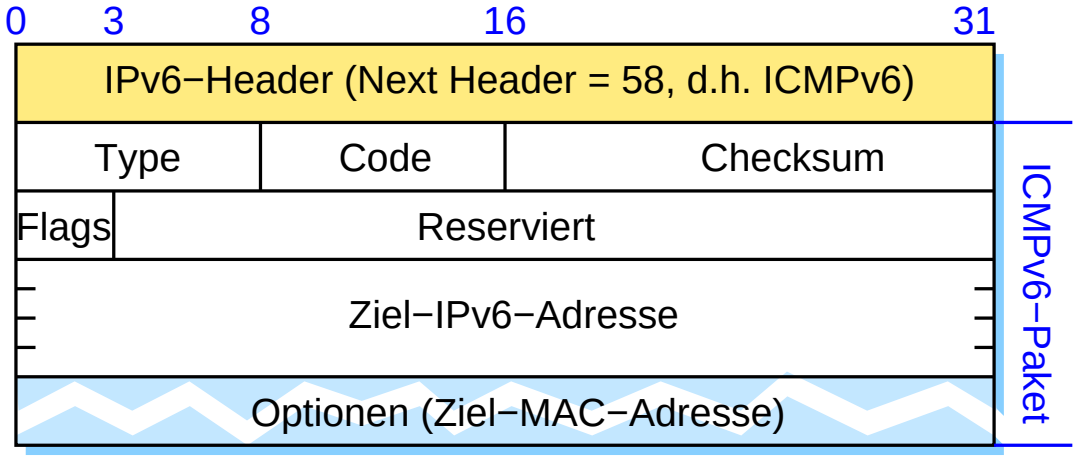
NDP: *Neighbor Discovery Protocol* (IPv6)

- NDP ist Teilprotokoll von ICMPv6
  - *Neighbor Solicitation/Advertisement*
- Funktionsweise der Adreßumsetzung analog zu ARP
- Unterschiede:
  - Nutzung von ICMP-Paketen statt ARP-Protokoll
    - Anfrage: *Neighbor Solicitation*
    - Antwort: *Neighbor Advertisement*
  - Anfrage nicht per Broadcast sondern per Multicast
    - an die zugehörige *Solicited Nodes* Multicast-Gruppe
    - Gruppe wird aus letzten 24 Bits der Ziel-IP-Adresse bestimmt

Anmerkungen zu Folie 209:

Die *Neighbor Solicitation* wird an die IPv6 Multicast-Adresse FF02:0:0:0:0:1:FFxx:xxxx gesendet, wobei xx:xxxx die letzten 24 Bits der Ziel-IP-Adresse sind. Diese IP-Multicast-Adresse wird fest auf die MAC-Multicast-Adresse 3333:FFxx:xxxx abgebildet, so daß die Anfrage als Ethernet-Frame mit dieser Zieladresse versendet wird. Dadurch wird erreicht, daß die Anfrage nicht von allen, sondern nur von einigen möglicherweise betroffenen Ethernet-Karten empfangen wird und entsprechend nicht von allen Rechnern bearbeitet werden muß.

Aufbau eines NDP-Pakets (siehe auch Wireshark-Aufzeichung)



- *Neighbor Solicitation* (Type = 135)
  - Multicast an ff02::1:ff00:0/104 + letzte 24 Bits der Zieladresse
- *Neighbor Advertisement* (Type = 136)
  - Multicast an alle IPv6 Hosts (ff02::1)
  - Options-Feld beinhaltet MAC-Adresse des Ziels

Anmerkungen zu Folie 210:

- Das Feld „Code“ ist bei *Neighbor Solicitation* und *Neighbor Advertisement* auf 0 gesetzt.
- Das Feld „Flags“ wird nur beim *Neighbor Advertisement* genutzt:
  - Das erste Bit (**R**), wenn die antwortende Station ein Router ist.
  - Das zweite Bit (**S**) zeigt an,daß das *Neighbor Advertisement* die Antwort auf eine ***Unicast***-*Neighbor-Solicitation* ist.
  - Das dritte Bit (**O**) wird gesetzt, wenn der Empfänger den Eintrag im *Neighbor Cache* aktualisieren soll.

★★

DHCP: *Dynamic Host Configuration Protocol*

- Automatisiert u.a. die Vergabe von IP-Adressen (IPv4 und IPv6)
- Vorgehensweise:
  1. Rechner sendet Broadcast-Anfrage (DHCPDISCOVER)
     - Verbreitung nur im lokalen Netz
  2. DHCP-Server sendet DHCPOFFER: Angebot für IP-Adresse
     - Zuordnung statisch oder dynamisch, als Schlüssel dient MAC-Adresse des Rechners
     - ggf. auch weitere Optionen (Hostname, DNS-Server, ...)
  3. Rechner fordert angebotene Adresse vom DHCP-Server an (DHCPREQUEST)
  4. DHCP-Server bestätigt (DHCPACK)
- IP-Adresse wird nur für eine bestimmte Zeit „gemietet“
  - periodische Wiederholung der Schritte 3 und 4

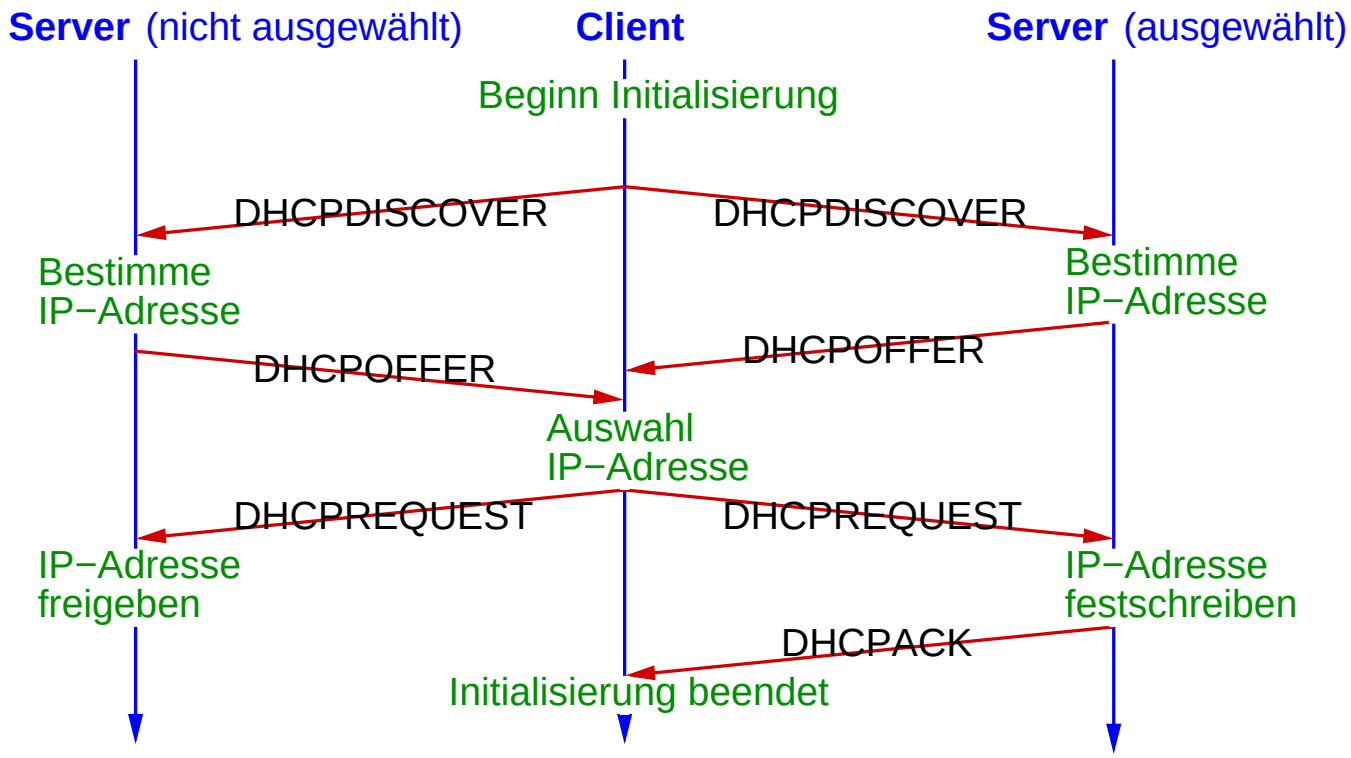
Anmerkungen zu Folie 211:

Das DHCP-Protokoll unterstützt, daß es mehrere DHCP-Server in einem Netz geben kann. DHCPDISCOVER wird von allen Servern empfangen, die auch alle antworten. Der Client kann sich dann eine Antwort aussuchen. Der DHCPREQUEST wird ebenfalls als Broadcast versendet. Er enthält zusätlich die IP-Adresse des Servers, dessen DHCPOFFER der Client annehmen will. Der DHCPREQUEST zeigt damit allen anderen Servern an, daß der Client deren Angebot nicht annimmt.

Zusätzlich zum DHCPREQUEST gibt es auch noch DHCPINFORM Pakete, mit denen ein Client, der schon eine (statisch konfigurierte) IP-Adresse hat, weitere Konfigurationsparameter vom DHCP-Server anfordern kann.

DHCP sieht auch vor, daß ein sog. DHCP-Relay Anfragen in ein anderes Netz weitergeben kann (d.h., der DHCP-Server kann sich dann auch in einem anderen Netz befinden).

Ablauf



Aufbau eines DHCP-Pakets (siehe auch Wireshark-Aufzeichung)

- Felder u.a. für:
  - Pakettyp (*request*, *reply*)
  - angefragte Client-IP-Adresse
    - im DHCPREQUEST
  - angebotene Client-IP-Adresse
    - in DHCPOFFER und DHCPACK
  - Hardware-Adresse des Clients
    - i.a. MAC-Adresse
  - Name einer Boot-Datei
    - wird vom Server an Client per SFTP übertragen
  - Optionen
    - z.B. Anfrage / Zuweisung von Subnetzmaske, Hostname, Default-Gateway, DNS-Server, NTP-Server

Zustandslose IPv6 Autokonfiguration

- Vorgehensweise eines Hosts:
  1. bilde *link-local* Adresse
     - Präfix FE80::/10 und z.B. EUI-64 oder Zufallszahl
  2. prüfe Adresse mit NDP: *Neighbor Solicitation*
     - falls keine Antwort: Adresse im LAN eindeutig
  3. warte auf *Router Advertisement*
     - sende ggf. explizit *Router Solicitation* wg. Wartezeit
  4. *Router Advertisement* teilt mit:
     - Adresse und/oder andere Optionen über DHCP holen?
     - ggf. globales Routing-Präfix und Präfixlänge
  5. bilde Adresse aus Präfix und z.B. EUI-64 oder Zufallszahl
  6. prüfe Adresse mit NDP: *Neighbor Solicitation*
- Adresse ist i.d.R. nur begrenzte Zeit gültig

Anmerkungen zu Folie 214:

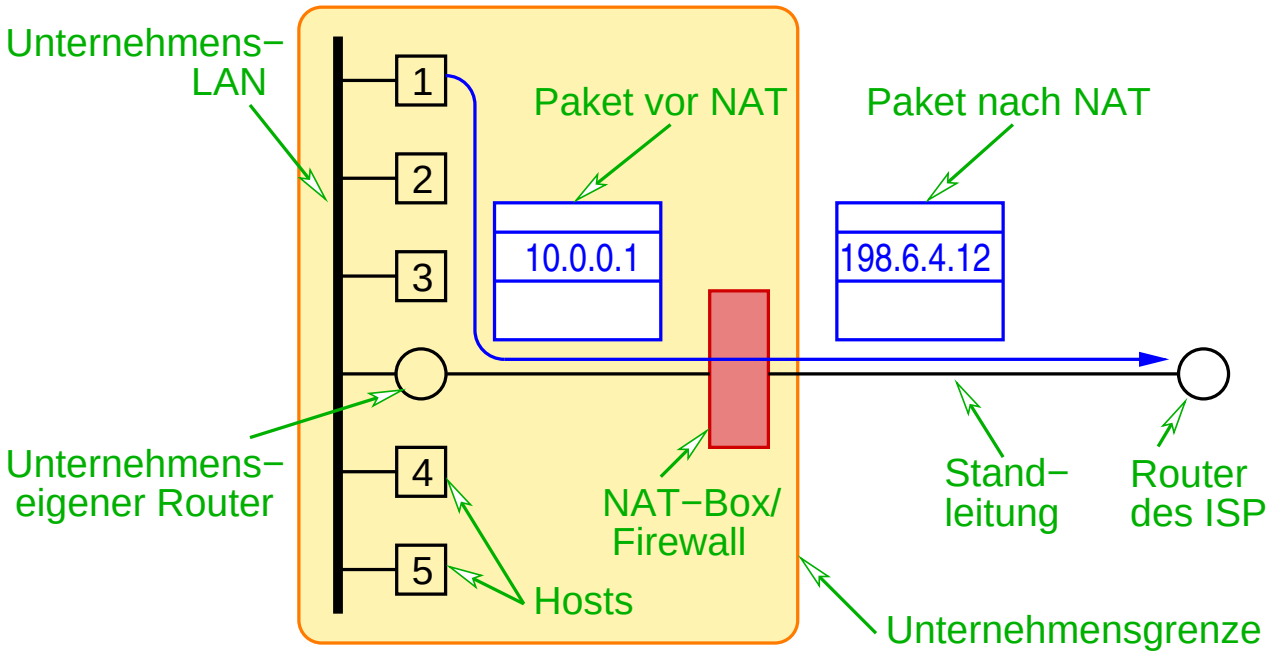
- Die Vorgehensweise bei der zustandslosen IPv6 Autokonfiguration (Stateless Address Autoconfiguration, SLAAC) ist in RFC 4862, Abschnitt 4 gut beschrieben.
- Bei der Prüfung auf Eindeutigkeit der gewählten Adresse wird das *Neighbor Solicitation* Paket an die gewählte Adresse gesendet. Falls es einen anderen Host mit dieser Adresse gibt, wird er mit seiner MAC-Adresse in einem *Neighbor Advertisement* Paket antworten. Zusätzlich muß der Host auch prüfen, ob er ein *Neighbor Solicitation* Paket an die von ihm gewählte Adresse empfängt. Das kann passieren, wenn ein anderer Host gleichzeitig dieselbe Adresse konfiguriert.
- Falls im lokalen Netz kein Router vorhanden ist, wird die Autokonfiguration mit Schritt 3 beendet. Dann hat der Host (genauer: das zu betreffende Interface) nur eine *link-local* Adresse, was in diesem Fall aber ausreicht.
- Der Schritt 5 findet nur wie angegeben statt, wenn das *Router Advertisement* mitteilt, daß kein DHCP benutzt werden soll. Ansonsten wird statt oder zusätzlich zu Schritt 5 noch eine DHCP-Konfiguration wie auf S. 211 durchgeführt.

★★

NAT, speziell NAPT (*Network Address Port Translation*)

- Ziel: Einsparung von IP-Adressen
  - mehrere Rechner eines Netzes werden auf dieselbe, nach außen sichtbare IP-Adresse gemultiplext
- Prinzip:
  - jeder Rechner des Netzes bekommt **private** IP-Adresse
  - drei reservierte Adreßbereiche:
    - 10.0.0.0/8 (10.0.0.0 - 10.255.255.255)
    - 172.16.0.0/12 (172.16.0.0 - 172.31.255.255)
    - 192.168.0.0/16 (192.168.0.0 - 192.168.255.255)
  - das gesamte Netz erhält **eine** „echte“ IP-Adresse
  - ausgehende Pakete durchqueren NAT-Box
    - NAT-Box ersetzt private IP-Adresse durch echte

Prinzipielle Funktionsweise von NAT



★★

Problem: Zustellung von Antwort-Paketen!?

- Alle eingehende Pakete sind an dieselbe Adresse gerichtet
  - woher weiß die NAT-Box, an wen das Paket gehen soll?
- Lösung (schmutzig!):
  - praktisch der gesamte IP-Verkehr ist UDP oder TCP
  - UDP- und TCP-Header enthalten Felder zur Adressierung von Quell- und Zielprozess (Quell-/Ziel-Port)
  - NAT-Box verwendet Quell-Port-Feld des UDP/TCP-Headers zum Speichern eines Demultiplex-Schlüssels
    - bei ausgehendem Paket: speichere Quell-IP-Adresse und Quell-Port ($\Rightarrow$ Schlüssel)
    - ersetze IP-Adresse, ersetze Quell-Port durch Schlüssel
    - Antwortpaket enthält Schlüssel als Ziel-Port
    - ersetze Ziel-Adresse/Port durch gespeicherte Adresse/Port

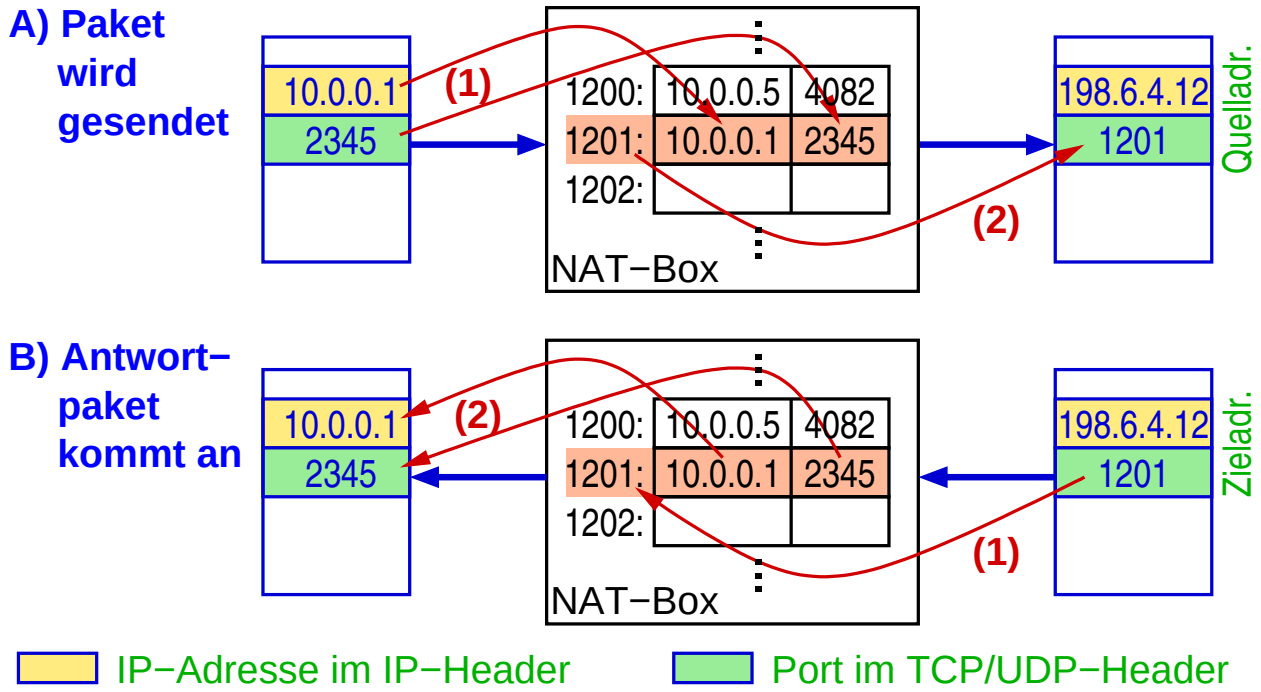
Anmerkungen zu Folie 217:

Was in diesem Abschnitt beschrieben (und in den meisten NAT-Boxen eingesetzt) wird, ist NA**P**T (*Network Address **Port** Translation*). Dabei wird ein ganzes Netz auf eine einzige IP-Adresse abgebildet.

Prinzipiell kann auch reines NAT genutzt werden, wobei dem Netz dann aber eine **Menge** von öffentlichen IP-Adressen zugewiesen wird. Die NAT-Box ersetzt dann die private IP-Adresse in ausgehenden Paketen durch eine dieser öffentlichen Adressen. Antwort-Pakete können dann zugestellt werden, wenn im lokalen Netz (in einem Zeitraum) höchstens soviele Rechner aktiv sind, wie öffentliche IP-Adressen vorhanden sind. Die NAT-Box hält dazu einen Cache mit den Zuordnungen zwischen privater und öffentlicher IP-Adresse mit einer begrenzten Zahl an Einträgen, wobei jeder Eintrag nach einer gewissen Zeit der Inaktivität gelöscht wird. Die öffentliche IP-Adresse wird damit wieder frei für andere Rechner des Netzes.

★★

Genaue Funktionsweise von NAT



★

Kritik:

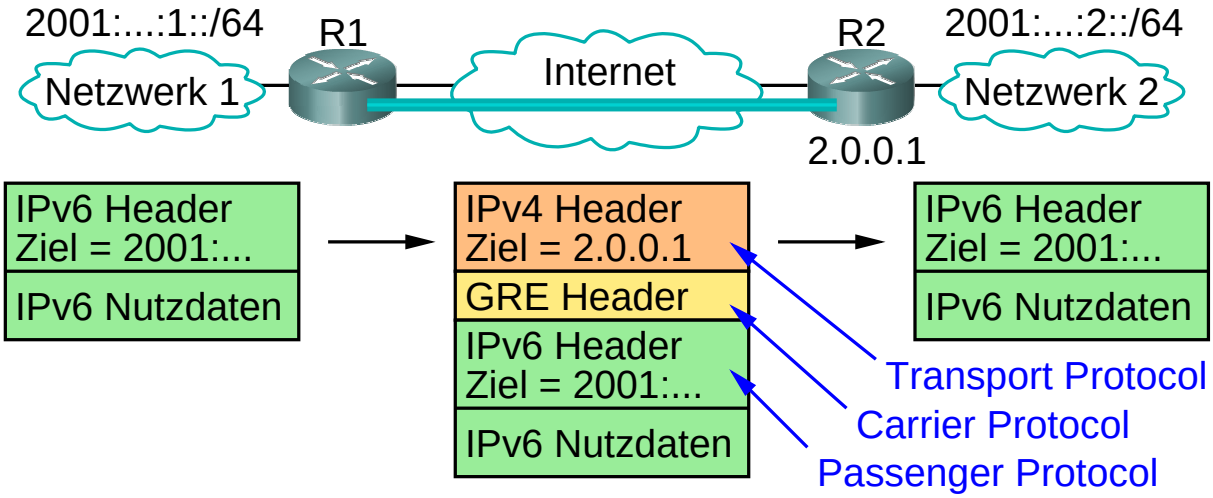
- IP-Adressen nicht mehr weltweit eindeutig
- Macht aus Internet quasi verbindungsorientiertes Netz
  - Ausfall der NAT-Box: Zerstörung aller Kommunikationsbeziehungen (TCP und UDP)!
- NAT (Schicht 3) macht Annahmen über Schicht 4!
  - neue TCP/UDP-Version, andere Protokolle über IP !?
- Nutzdaten können IP-Adressen enthalten (z.B. FTP, H.323)

Vorteile:

- Einsparung von IP-Adressen
- Sicherheit: Hosts sind von außen nicht erreichbar, Struktur des lokalen Netzes nach außen verborgen

5.9 Tunneling

- Übertragung eines Protokolls über ein Protokoll derselben oder einer höheren OSI-Schicht
- z.B. Übertragung von IPv6-Paketen über IPv4



5.9 Tunneling ...

- Aufgaben des *Carrier*-Protokolls:
  - z.B. Multiplexing, Reihenfolgeerhaltung, Authentifizierung, ...
  - ggf. kann das Carrier-Protokoll auch fehlen
- Einsatz von Tunneln:
  - Kommunikation zwischen Routern mit speziellen Fähigkeiten
    - z.B. Multicast, IPv6
  - Kopplung von nicht-IP Netzen (z.B. Ethernet) über das Internet
  - VPNs: Authentifizierung und Verschlüsselung im Tunnel
  - „Durchtunneln“ von Firewalls
    - oft HTTP als Transportprotokoll

Anmerkungen zu Folie 221:

- GRE (*Generic Routing Encapsulation*) ist ein einfaches, unsicheres Tunnel-Protokoll, das von CISCO entwickelt wurde.
- Bei dem von Microsoft verwendeten Teredo-Tunneling zur Übertragung von IPv6-Paketen über IPv4-Netze übernimmt UDP die Aufgabe des Carrier-Protokolls.

  Die Verwendung von UDP ermöglicht dabei die Übertragung der Pakete durch NAT-Boxen hindurch.

5.10 Übergang von IPv4 auf IPv6

Verschiedene Ansätze

- Dual-Stack-Ansatz
  - Netzknoten implementieren sowohl IPv6 als auch IPv4
  - Fähigkeiten des Zielknotens über DNS zu ermitteln
    - IPv6-Adresse für IPv6-fähigen Knoten, IPv4-Adresse sonst
  - Router müssen ggf. IP-Header der Pakete anpassen
    - wenn zwischenliegende Router nur IPv4 unterstützen
    - Informationsverlust möglich (z.B. FlowLabel nur in IPv6)
- Tunneling
  - Aufbau eines IPv4-Tunnels zwischen IPv6-Knoten
  - d.h. IPv6-Pakete werden in IPv4-Pakete eingepackt
- Adreßübersetzung analog zu NAT (6to4)

5.10 Übergang von IPv4 auf IPv6 ...

Stand

- Praktisch alle neuen Features von IPv6 inzwischen auch in IPv4 verfügbar
  - Hauptmotivation: größerer Adreßbereich
- Moderne Betriebssysteme beinhalten IPv4 und IPv6
  - z.B. Linux, Windows (ab XP)
- Bis Juni 2006: Internet-Testbett 6bone
- IPv6 inzwischen im „Produktionseinsatz“
  - einige Dienste nur noch mit IPv6 angeboten
- Zeitrahmen für die Ablösung von IPv4 aber unklar

5.11 Zusammenfassung

- Internetwork: Netz von Netzen
- IP-Protokoll: Best-Effort, run over everything
  - Hierarchische Adressen: Routing nur zwischen Netzen
  - Bessere Skalierbarkeit durch CIDR und Subnetze
- Hilfsprotokolle: ICMP, ARP, NDP, DHCP
- NAT: Umsetzung privater auf globale IP-Aressen

Nächste Lektion:

- Routing