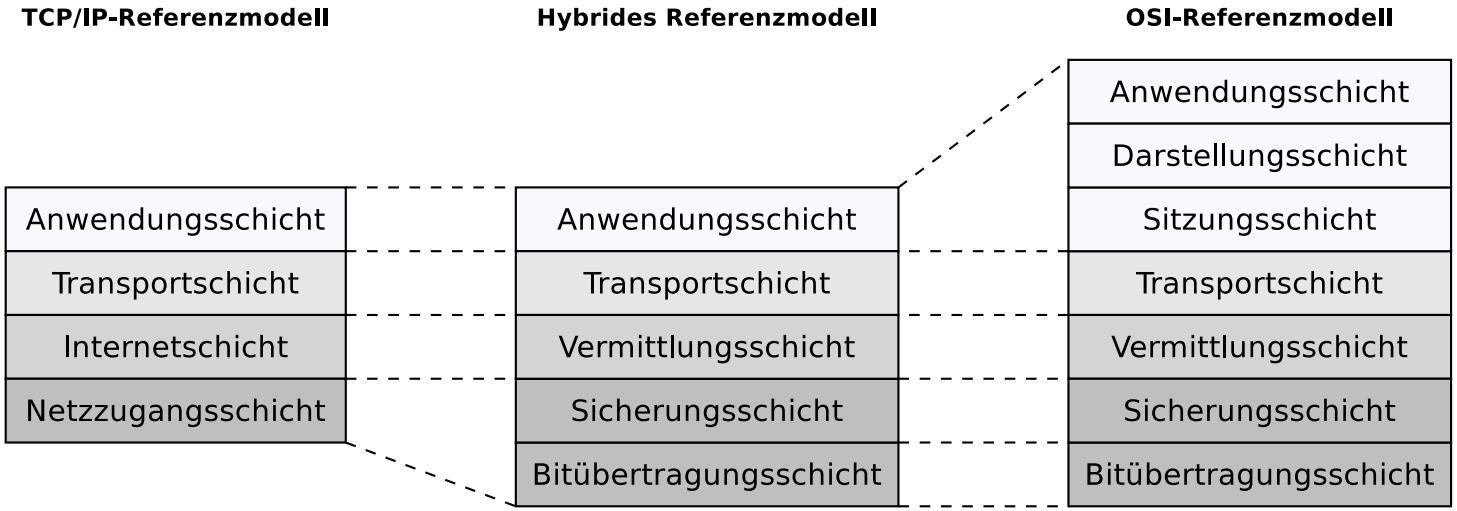
7. Foliensatz Computernetze

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences
(1971–2014: Fachhochschule Frankfurt am Main)
Fachbereich Informatik und Ingenieurwissenschaften
christianbaun@fb2.fra-uas.de

Vermittlungsschicht

- Aufgaben der Vermittlungsschicht (Network Layer):
  - Sender: Segmente der Transportschicht in Pakete unterteilen
  - Empfänger: Pakete in den Rahmen der Sicherungsschicht erkennen
  - Logische Adressen (IP-Adressen) bereitstellen
  - Routing: Ermittlung des besten Weges
  - Forwarding: Weiterleitung der Pakete zwischen logischen Netzen, also über physische Übertragungsabschnitte hinweg



Übungsblatt 4 wiederholt die für die Lernziele relevanten Inhalte dieses Foliensatzes

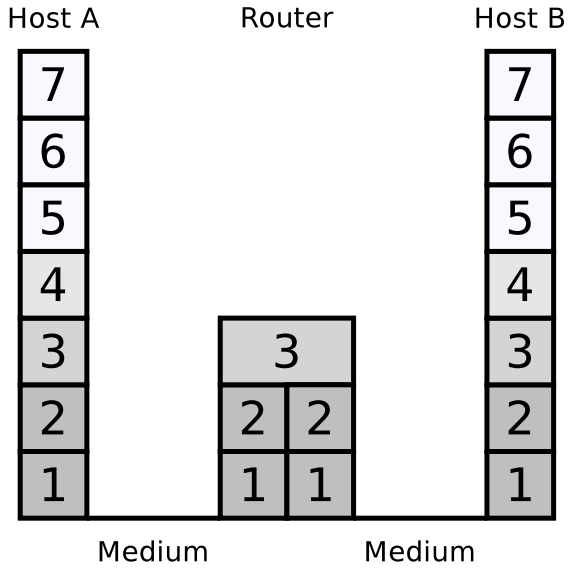
- Geräte: Router, Layer-3-Switch (Router ohne WAN-Schnittstelle)
- Protokolle: IPv4, IPv6, ICMP, IPX/SPX, DECnet

Lernziele dieses Foliensatzes

- Vermittlungsschicht (Teil 1)
  - Geräte der Vermittlungsschicht
    - Router
    - Auswirkungen auf die Kollisionsdomäne
    - Broadcast-Domäne (Rundsendedomäne)
  - Adressierung in der Vermittlungsschicht
    - Aufbau von IP-Adressen
    - Netzklassen, Netzwerkteil und Geräteteil, Subnetze und Netzmaske
    - Private IP-Adressen
    - Aufbau von IP-Paketen
    - Fragmentieren von IP-Paketen
  - Diagnose und Fehlermeldungen mit ICMP

Router, Layer-3-Switch und Gateway

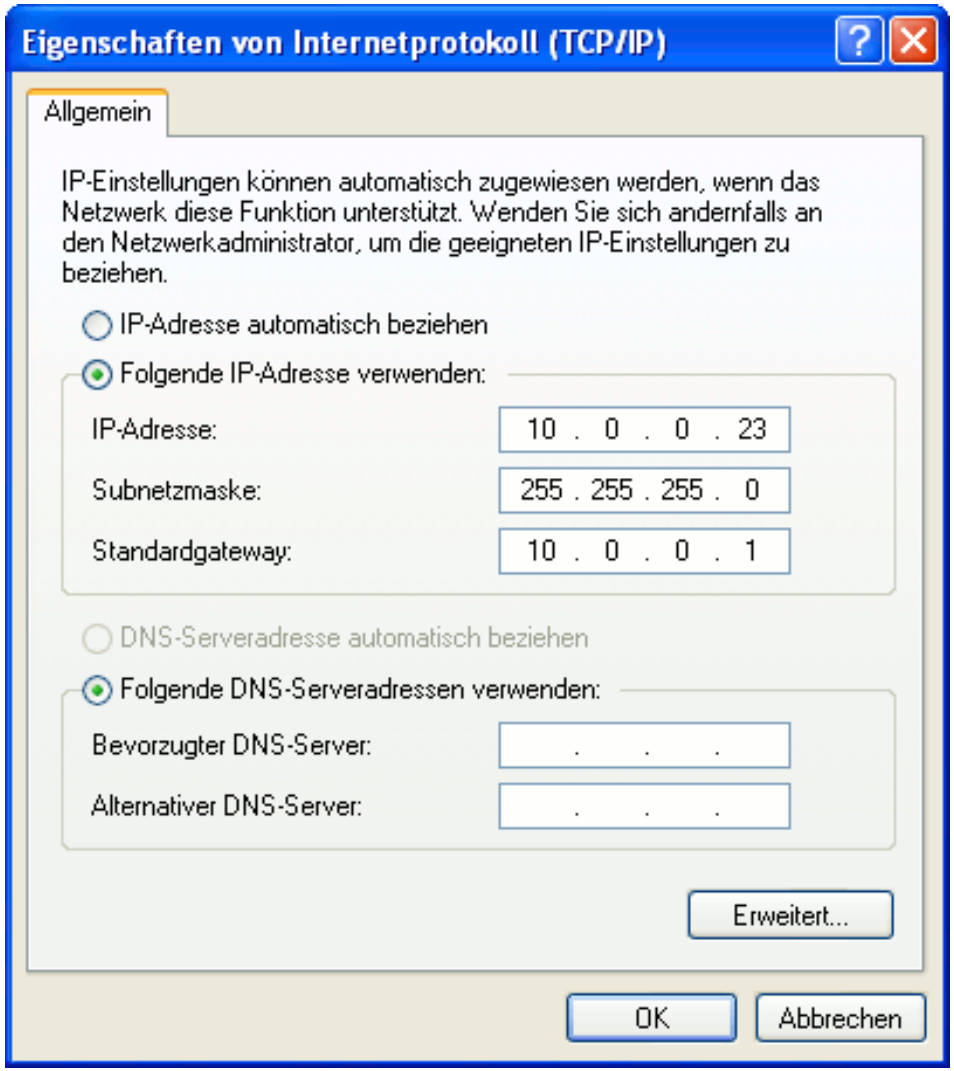
- **Router** leiten Datenpakete zwischen Netzen mit eigenen logischen Adressbereichen weiter
  - Besitzen genau wie Hubs und Switche mehrere Schnittstellen
  - Ermöglichen die Verbindung des lokalen Netzes (LAN) mit einem WAN (z.B. via DSL oder 3G/4G Mobilfunk)
- **Layer-3-Switche** sind Router ohne WAN-Schnittstelle
- **Gateways** sind Protokollumsetzer
  - Ermöglichen Kommunikation zwischen Netzen, die auf unterschiedlichen Protokollen basieren
  - Gateways, die auf der Vermittlungsschicht arbeiten, heißen auch **Mehrprotokoll-Router** oder **Multiprotokoll-Router**



Die beiden unteren Bilder zeigen einen Linksys WRT54GL Wireless-G Wireless Router mit einem WAN-Port und einem 4-Port Switch

Gateways (1/2)

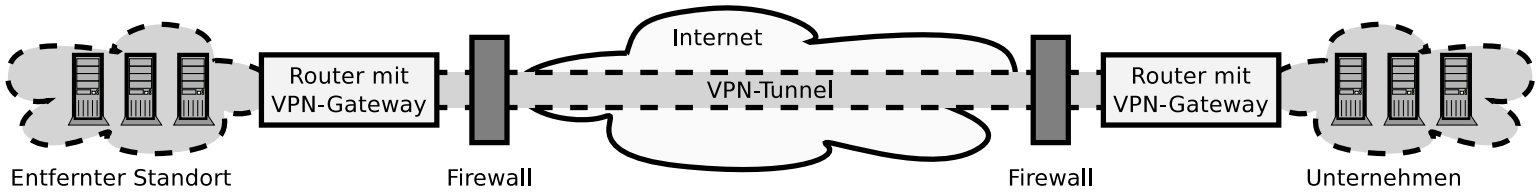
- Moderne Computernetze arbeiten fast ausschließlich mit dem Internet Protocol (IP)
  - Darum ist eine Protokollumsetzung auf der Vermittlungsschicht heute meist nicht nötig
- In früheren Zeiten wurde bei der Konfiguration eines Endgeräts der Gateway als **Default Gateway** eintragen
  - Heute trägt man in diesem Feld den Router ein, weil man keinen Gateway mehr braucht
    - Der Begriff **Default Router** wäre heute also eigentlich passender



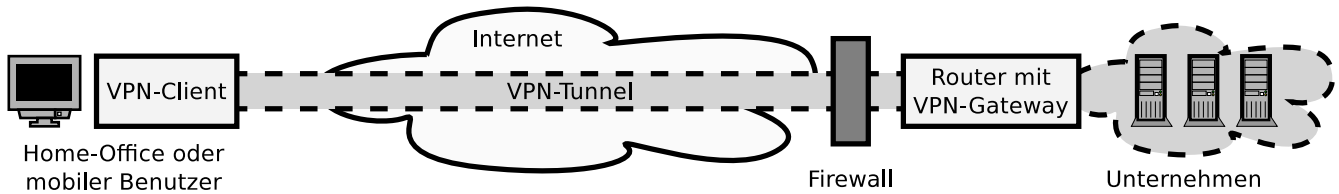
Gateways (2/2)

- Auch VPN-Gateways (Virtual Private Network) können auf der Vermittlungsschicht arbeiten
  - Sie ermöglichen über unsichere öffentliche Netze den sicheren Zugriff auf entfernte geschützte Netze (z.B. Hochschul-/Firmennetze)
  - Dienste (z.B. Email), die nur innerhalb des geschützten Netzes zur Verfügung stehen, werden über eine getunnelte Verbindung genutzt

Site-to-Site VPN

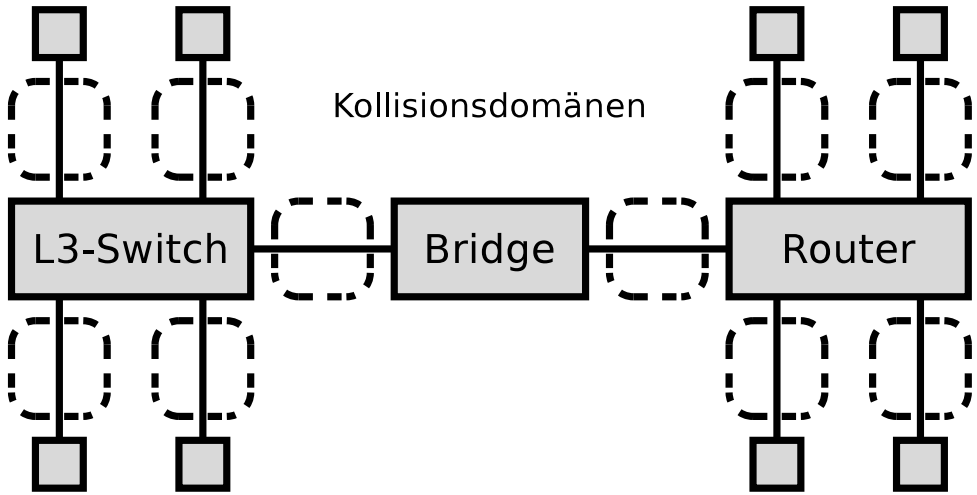


Remote Access VPN bzw. End-to-Site VPN



Kollisionsdomäne – Router und Layer-3-Switches

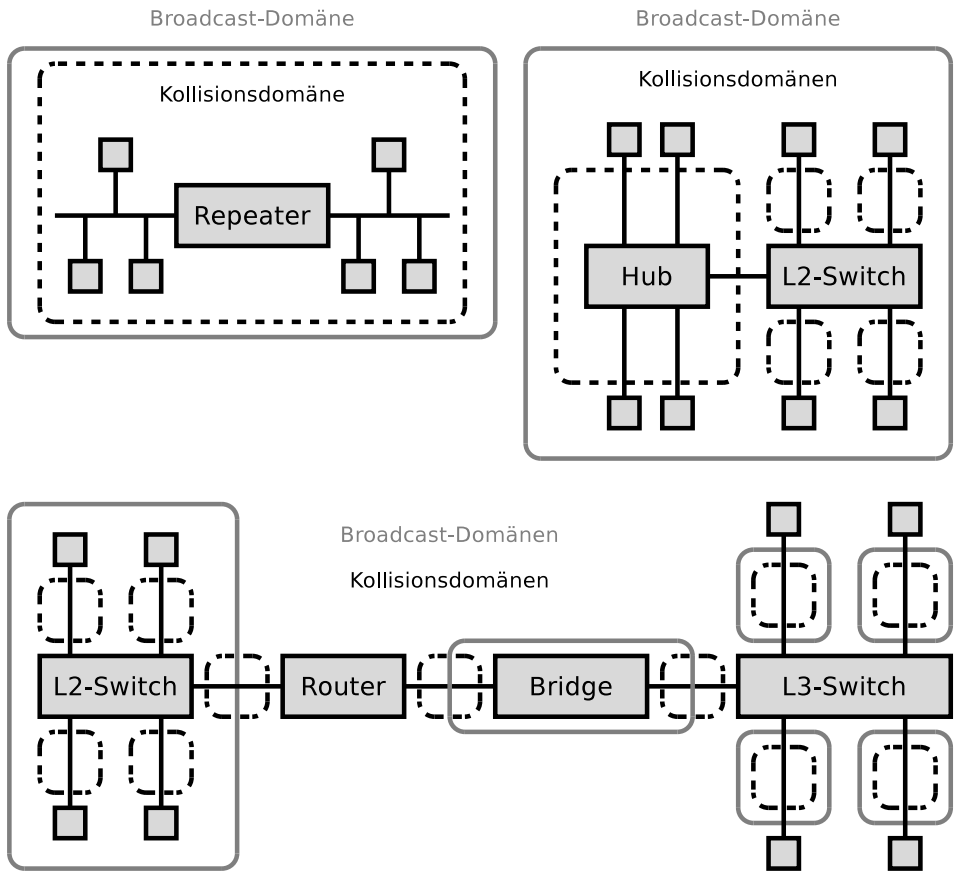
- Router und Layer-3-Switches teilen die Kollisionsdomäne
  - Genau wie Bridges und Layer-2-Switches



- Die Geräte aus Schicht 1 (**Repeater**, **Hubs**) unterbrechen die Kollisionsdomäne nicht
- Die Geräte aus Schicht 2 und 3 (**Bridges**, **Layer-2-Switche**, **Router**, **Layer-3-Switche**) unterbrechen die Kollisionsdomäne

Broadcast-Domäne – Rundsendedomäne (1/2)

- Logischer Teil eines Computernetzes, bei dem ein Broadcast alle Netzwerkgeräte, die zu diesem Teil gehören, erreicht
  - Geräte aus Schicht 3 (**Router**, **Layer-3-Switche**) teilen die Broadcast-Domäne
  - Geräte aus Schicht 1 und 2 (**Repeater**, **Hubs**, **Bridges**, **Layer-2-Switche**) unterbrechen sie nicht
    - Sie arbeiten aus Sicht logischer Netze transparent

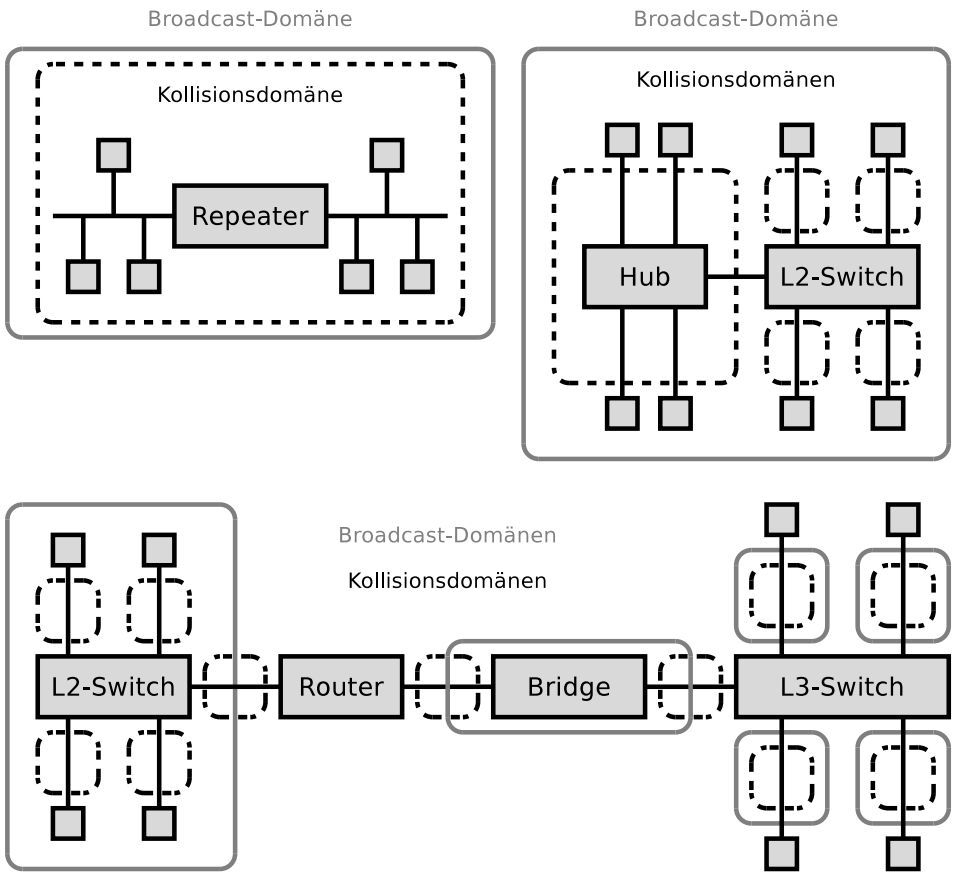


Der Begriff Broadcast-Domäne. . .

bezieht sich immer auf die Vermittlungsschicht und nie auf die Sicherungsschicht (obwohl es auch auf der Sicherungsschicht Broadcasts gibt)

Broadcast-Domäne – Rundsendedomäne (2/2)

- Broadcast-Domänen bestehen aus einer oder mehreren Kollisionsdomänen
- Router arbeiten auf der Vermittlungsschicht (Schicht 3)
  - Das heißt, an jedem Port eines Routers hängt ein anderes IP-Netz
    - Das ist wichtig, wenn man die Anzahl der nötigen Subnetze berechnen will
  - Man kann mehrere Hubs, Switche, Repeater oder Bridges in einem IP-Subnetz betreiben
    - Man kann aber nicht ein IP-Subnetz an mehreren Ports eines Routers betreiben

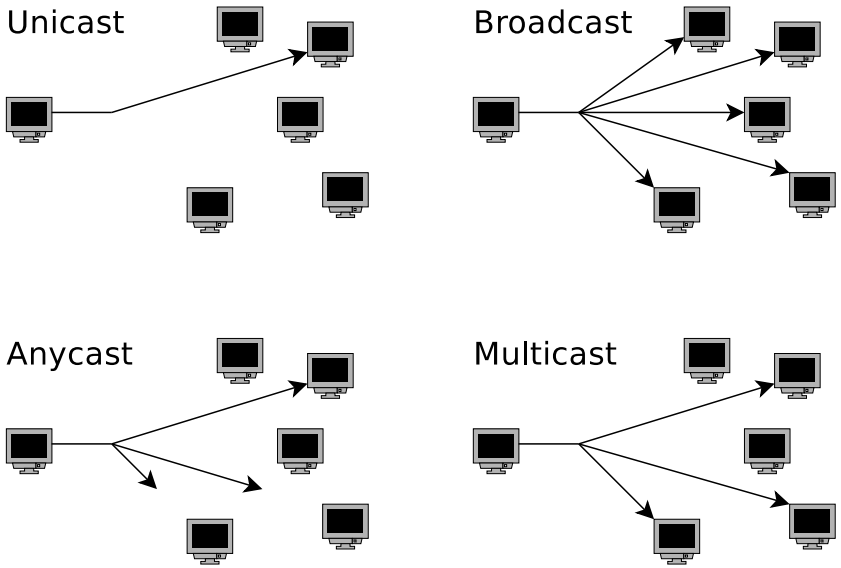


Adressierung in der Vermittlungsschicht (1/2)

- Ausschließlich physische Adressierung via MAC-Adressen ist in Computernetzen mit eventuell globalen Ausmaßen nicht sinnvoll
  - Grund: Wartbarkeit
- Es sind **logische Adressen** nötig, die von der konkreten Hardware unabhängig sind
  - Mit logischer Adressierung wird die Teilnehmersicht für Menschen (logische Adressen) von der internen Sicht für Rechner und Software (physische Adressen) getrennt

Adressierung in der Vermittlungsschicht (2/2)

- Jedes IP-Paket enthält eine Empfängeradresse
  - Den Aufbau von IP-Adressen definiert das Internet Protocol (IP)



- Eine IP-Adresse kann einen einzelnen Empfänger (**Unicast**) oder eine Gruppe von Empfängern bezeichnen (**Multicast** oder **Broadcast**)
- Einem Netzwerkgerät können auch mehrere IP-Adressen zugeordnet sein

- Bei **Anycast** erreicht man über eine Adresse einen einzelnen Empfänger aus einer Gruppe
  - Es antwortet der Empfänger, der über die kürzeste Route erreichbar ist

Multicast verwenden zum Beispiel die Routing-Protokolle RIPv2 und OSPF (siehe Foliensatz 8) und das Network Time Protocol – NTP (siehe Foliensatz 10) zur Synchronisierung von Uhren
Anycast verwenden zum Beispiel einigen Root-Nameserver im Domain Name System (siehe Foliensatz 10)

Aufbau von IP-Adressen

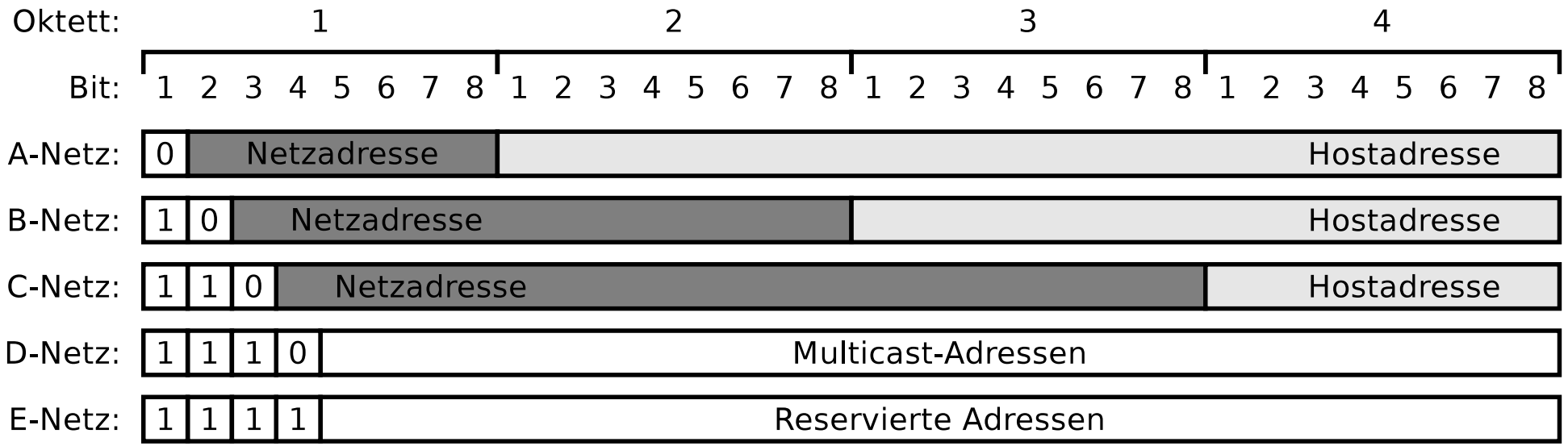
- IPv4-Adressen sind 32 Bits (4 Bytes) lang
  - Daher können $2^{32} = 4.294.967.296$ Adressen dargestellt werden

Adressraum = Menge aller gültigen Netzadressen

- Üblich ist die Darstellung in der sogenannten **Dotted decimal notation**
  - Die 4 Oktette werden als vier durch Punkte voneinander getrennte ganze Zahlen in Dezimaldarstellung im Bereich von 0 bis 255 geschrieben
    Beispiel: 141.52.166.25

Netzklassen, Netzwerkteil und Geräteteil

- Ursprünglich wurden IPv4-Adressen in Klassen von A bis C eingeteilt
  - Es existierten auch die Klassen D und E für spezielle Aufgaben
- Die 32 Bits einer IPv4-Adresse bestehen aus den beiden Feldern:
  - **Netzadresse** (Network Identifier bzw. Netzwerk-ID)
  - **Hostadresse** (Host Identifier bzw. Host-ID)
  - Klasse A: 7 Bits für Netzadresse und 24 Bits für Hostadresse
  - Klasse B: 14 Bits für Netzadresse und 16 Bits für Hostadresse
  - Klasse C: 21 Bits für Netzadresse und 8 Bits für Hostadresse



Netzklassen (1/2)

- Die Präfixe legen die Netzklassen und ihre Adressbereiche fest

Klasse	Präfix	Adressbereich	Netzteil	Hostteil
A	0	0.0.0.0 - 127.255.255.255	7 Bits	24 Bits
B	10	128.0.0.0 - 191.255.255.255	14 Bits	16 Bits
C	110	192.0.0.0 - 223.255.255.255	21 Bits	8 Bits
D	1110	224.0.0.0 - 239.255.255.255	—	—
E	1111	240.0.0.0 - 255.255.255.255	—	—

- $2^7 = 128$ Klasse A-Netze mit mit jeweils maximal $2^{24} = 16.777.216$ Hostadressen
- $2^{14} = 16.384$ Klasse B-Netze mit jeweils maximal $2^{16} = 65.536$ Hostadressen
- $2^{21} = 2.097.152$ Klasse C-Netze mit jeweils maximal $2^8 = 256$ Hostadressen
- Klasse D enthält Multicast-Adressen (zum Beispiel für IPTV)
- Klasse E ist für zukünftige (?!) Verwendungen und Experimente reserviert

Warum wird der Klasse E-Adressraum von IPv4 nicht verwendet?

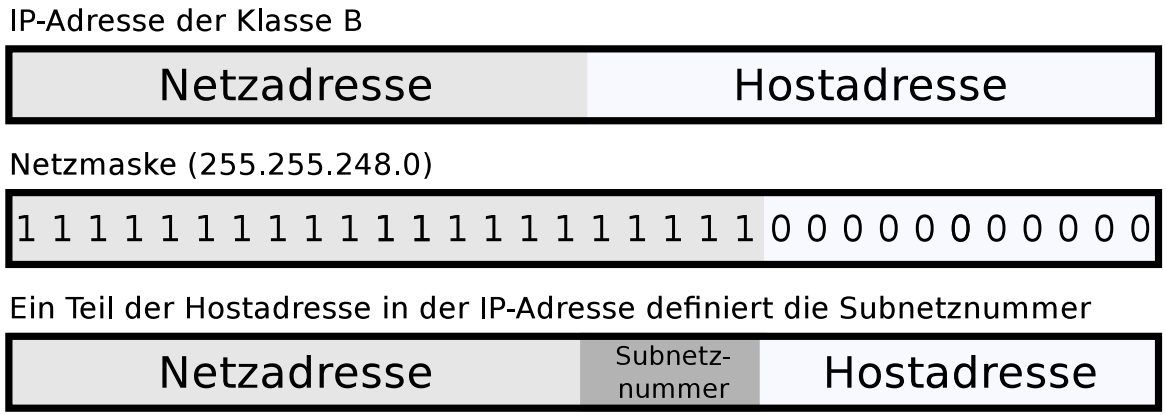
„The class E space has 268 million addresses and would give us in the order of 18 months worth of IPv4 address use. However, many TCP/IP stacks, such as the one in Windows, do not accept addresses from class E space and will not even communicate with correspondents holding those addresses. It is probably too late now to change this behavior on the installed base before the address space would be needed.“

Quelle: `http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_10-3/103_addr-cons.html`

Netzklassen (2/2)

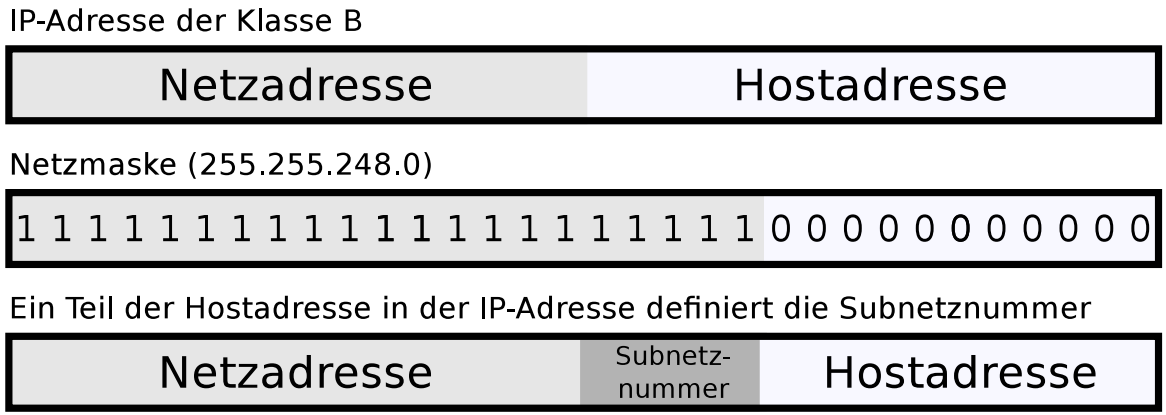
- Praktisch relevant sind nur die Klassen A, B und C
- Ursprünglich war beabsichtigt, durch die Netzadresse physische Netze eindeutig zu identifizieren
  - Dieses Vorgehen bringt aber Nachteile mit sich
- **Nachteile der Netzklassen:**
  - Sie können nicht dynamisch an Veränderungen angepasst werden
  - Sie verschwenden viele Adressen
    - Ein Klasse C-Netz mit 2 Geräten verschwendet 253 Adressen
    - Bei Klasse C-Netzen kann der Adressraum rasch knapp werden
    - Ein Klasse B-Netz mit 256 Geräten verschwendet $>$ 64.000 Adressen
    - Es gibt es nur 128 Klasse A-Netze
    - Migration vieler Geräte in eine andere Netzklasse ist aufwändig
- Lösung: Unterteilung logischer Netze in **Teilnetze** (**Subnetze**)
  - 1993: Einführung des klassenlosen Routings – **Classless Interdomain Routing** (CIDR)

Netzmaske (1/2)



- Um Subnetze zu bilden, ist eine **(Sub-)Netzmaske** nötig
  - Alle Knoten in einem Netzwerk bekommen eine Netzmaske zugewiesen
    - Länge: 32 Bits (4 Bytes)
    - Mit ihr wird die Anzahl der Subnetze und Hosts festgelegt
- Die Netzmaske unterteilt die Hostadresse der IP-Adresse in **Subnetznummer** und **Hostadresse**
  - Die Netznummer bleibt unverändert
  - Die Netzmaske fügt eine weitere Hierarchieebene in die IP-Adresse ein

Netzmaske (2/2)



- Aufbau der Netzmaske:
  - Einsen kennzeichnen den (Sub-)Netz-Nummernteil eines Adressraumes
  - Nullen kennzeichnen den Teil des Adressraumes, der für die Hostadressen zur Verfügung steht
- Um z.B. ein Klasse B-Netz in 20 Subnetze aufzuteilen, sind 5 Bits nötig
  - Jedes Subnetz braucht nämlich seine eigene Subnetznummer und diese muss binär dargestellt werden
  - Werden 5 Bits für die Darstellung der Subnetznummern verwendet, bleiben noch 11 Bits für den Hostteil

Schreibweise des Classless Interdomain Routing (CIDR)

- Seit Einführung des **CIDR** 1993 werden IP-Adressbereiche in der Notation `Anfangsadresse/Netzbits` vergeben
  - Die Netzbits sind die Anzahl der Einsen in der Netzmaske
- Die Tabelle zeigt die möglichen Aufteilungen eines Klasse C-Netzes in Subnetze

Netzbits	/24	/25	/26	/27	/28	/29	/30	/31	/32
Netzmaske	0	128	192	224	240	248	252	254	255
Subnetzbits	0	1	2	3	4	5	6	7	8
Subnetze	1	2	4	8	16	32	64	128	256
Hostbits	8	7	6	5	4	3	2	1	0
Hostadressen	256	128	64	32	16	8	4	2	—
Hosts	254	126	62	30	14	6	2	0	—

Nicht alle Adressen können/sollen verwendet werden

Netzbits	/24	/25	/26	/27	/28	/29	/30	/31	/32
Netzmaske	0	128	192	224	240	248	252	254	255
Subnetzbits	0	1	2	3	4	5	6	7	8
Subnetze	1	2	4	8	16	32	64	128	256
Hostbits	8	7	6	5	4	3	2	1	0
Hostadressen	256	128	64	32	16	8	4	2	—
Hosts	254	126	62	30	14	6	2	0	—

2 Hostadressen können nicht an Knoten vergeben werden, weil jedes (Sub-)-Netzwerk benötigt...

- eine Adresse (**Netzdeskriptor**) für das Netz selbst (alle Bits im Hostteil = 0)
- eine Broadcast-Adresse, um alle Knoten im Netz zu adressieren (alle Bits im Hostteil = 1)

2 Subnetznummern sollen nicht verwendet werden

- Die Subnetznummern, die ausschließlich aus Nullen und ausschließlich aus Einsen bestehen, sollen nicht verwendet werden $\Longrightarrow$ diese Regel ist veraltet, wird aber häufig angewendet
- Moderne Router und Netzwerksoftware haben kein Problem damit, wenn alle möglichen Subnetznummern für existierende Subnetze vergeben werden

Bestimmung der nötigen Bits für Subnetze

Netzbits	/24	/25	/26	/27	/28	/29	/30	/31	/32
Netzmaske	0	128	192	224	240	248	252	254	255
Subnetzbits	0	1	2	3	4	5	6	7	8
Subnetze	1	2	4	8	16	32	64	128	256
Hostbits	8	7	6	5	4	3	2	1	0
Hostadressen	256	128	64	32	16	8	4	2	—
Hosts	254	126	62	30	14	6	2	0	—

- Anhand der Tabelle ist es einfach, die nötigen Bits für Subnetze zu bestimmen
- Beispiel: Ein Klasse C-Netz soll in 5 Subnetze mit jeweils maximal 25 Hosts aufgeteilt werden
  - Jedes Subnetz benötigt eine Subnetznummer
  - Für 5 Subnetze sind 3 Subnetzbits nötig
  - Mit Hilfe der restlichen 5 Bits im Hostteil können in jedem Subnetz bis zu $32 - 2 = 30$ Hosts adressiert werden
  - Somit ist die Schrägstrichdarstellung /27 geeignet

Rechenbeispiel zu Subnetzen

- Beispiel: 172.21.240.90/27 ist eine Klasse B-Adresse ($\Longrightarrow$ siehe Präfix)
  - /27 = Anzahl der Einsen in der Netzmaske
- **IP-Adresse `AND` Netzmaske = Subnetzadresse**

```
1 AND 1 = 1,  1 AND 0 = 0,  0 AND 1 = 0,  0 AND 0 = 0
```

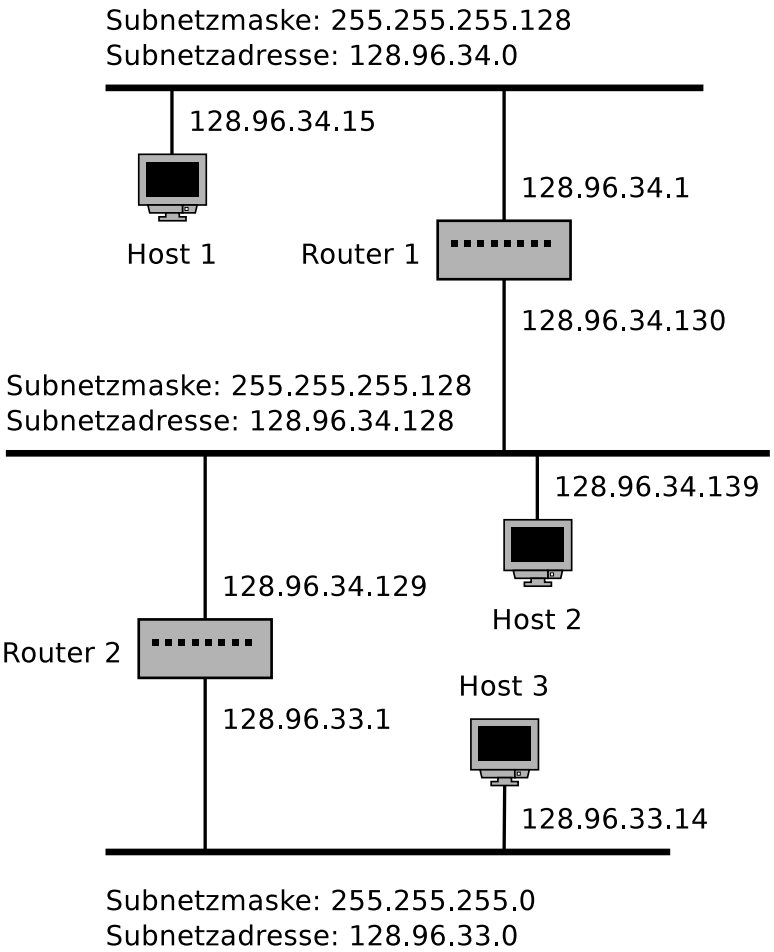
IP-Adresse	172.21.240.90	10101100 00010101 11110000 01011010
Netzmaske	255.255.255.224	11111111 11111111 11111111 11100000
Subnetzadresse	172.21.240.64	~~10101100 00010101~~ 11110000 010~~00000~~
Subnetznummer	1922	~~10101100 00010101~~ 11110000 010~~00000~~

- **IP-Adresse `AND` (`NOT` Netzmaske) = Hostadresse**

IP-Adresse	172.21.240.90	10101100 00010101 11110000 01011010
Netzmaske	255.255.255.224	11111111 11111111 11111111 11100000
negierte Netzmaske	000.000.000.31	~~00000000 00000000 00000000 000~~11111
Hostadresse	26	~~00000000 00000000 00000000 000~~11010

- /27 und Klasse B-Präfix $\Longrightarrow$ 11 Bits für die Subnetznummer
  - Es verbleiben 5 Bits und damit $2^5 = 32$ Adressen für den Hostteil
    - Davon sind 30 Hostadressen für Netzwerkgeräte verfügbar

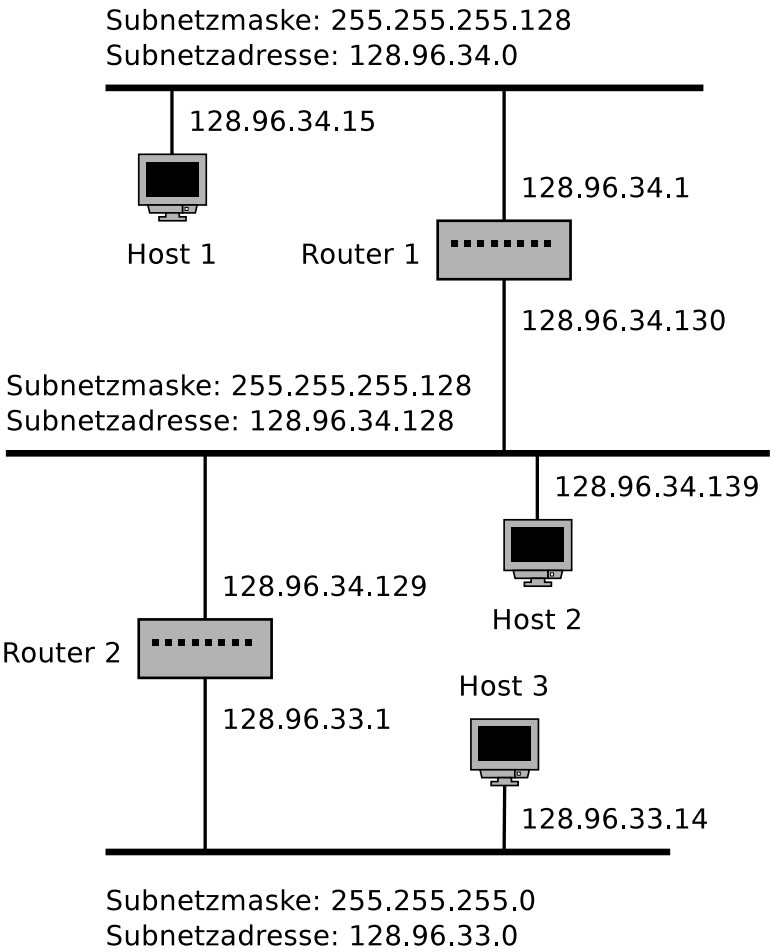
Beispiel (1/4)



Quelle: Computernetzwerke. Peterson und Davie. dpunkt (2000)

- Alle Hosts im gleichen Subnetz haben die gleiche Subnetzmaske
- IP `AND` Subnetzmaske = Subnetzadresse
- Will ein Host ein Paket versenden, führt er ein `AND` zwischen der eigenen Subnetzmaske und der IP des Ziels durch
  - Stimmt das Ergebnis mit der Subnetzadresse des Senders überein, weiß er, dass das Ziel im gleichen Subnetz liegt
  - Ist das Ergebnis nicht gleich, muss das Paket an einen Router gesendet werden, der es an ein anderes Subnetz weiterleitet

Beispiel (2/4)



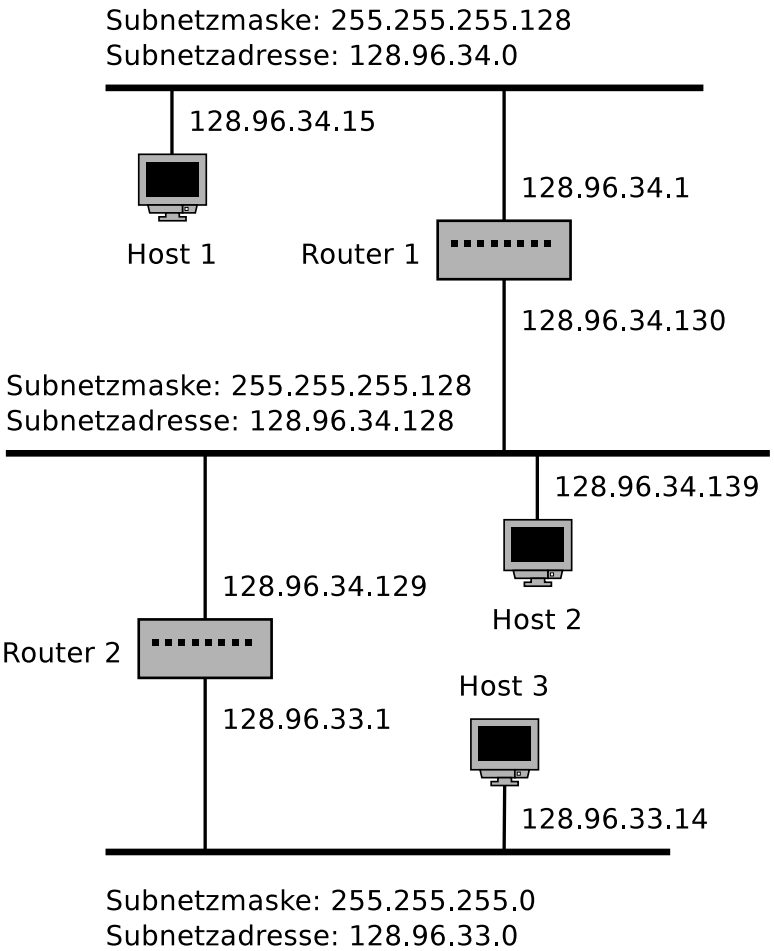
Quelle: Computernetzwerke. Peterson und Davie. dpunkt (2000)

- Beispiel: Host 1 sendet ein Paket an Host 2 (128.96.34.139)
- Host 1 berechnet Subnetzmaske (255.255.255.128) `AND` Zieladresse (128.96.34.139) und erhält 128.96.34.128
- Das ist nicht die Subnetzadresse von Host 1 $\Longrightarrow$ Host 2 ist in einem anderem Subnetz
- Host 1 übermittelt das Paket an seinen Standard-Router (128.96.34.1)
- Einträge in der Routing-Tabelle von Router 1

Subnetzadresse	Subnetzmaske	Nächster Hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- Routing-Protokolle/Algorithmen ($\Longrightarrow$ siehe Foliensatz 8) erstellen und pflegen die Einträge in den Routing-Tabellen der Router

Beispiel (3/4)



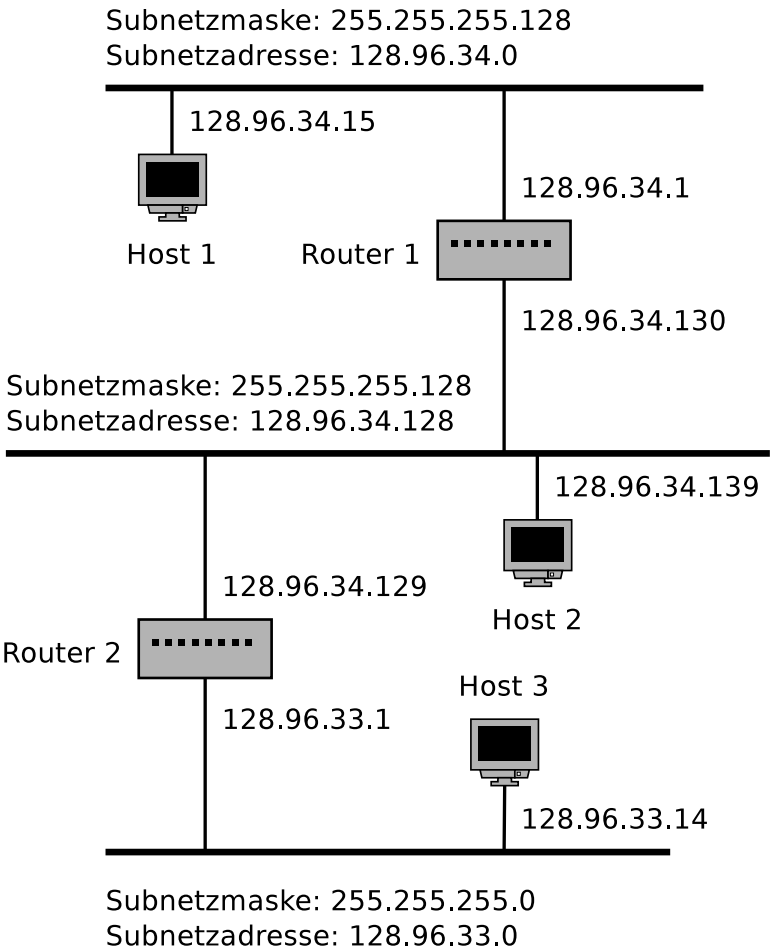
Quelle: Computernetzwerke. Peterson und Davie. dpunkt (2000)

- Einträge in der Routing-Tabelle von Router 1

Subnetzadresse	Subnetzmaske	Nächster Hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- Der Router führt ein `AND` zwischen der Zieladresse und der Subnetzmaske jedes Eintrags durch
- Stimmt das Ergebnis mit der Subnetzadresse des Eintrage überein, leitet der Router das Paket an den Router oder Port weiter
- Router 1 berechnet für die 1. Zeile: Host 2 (128.96.34.139) `AND` Subnetzmaske (255.255.255.128) ist 128.96.34.128
- Das stimmt nicht mit der Subnetzadresse (128.96.34.0) überein

Beispiel (4/4)



Quelle: Computernetzwerke. Peterson und Davie. dpunkt (2000)

- Einträge in der Routing-Tabelle von Router 1

Subnetzadresse	Subnetzmaske	Nächster Hop
128.96.34.0	255.255.255.128	Port 0
128.96.34.128	255.255.255.128	Port 1
128.96.33.0	255.255.255.0	Router 2

- Router 1 berechnet für die 2. Zeile: Host 2 (128.96.34.139) AND Subnetzmaske (255.255.255.128) ist 128.96.34.128
- Das stimmt mit der Subnetzadresse in der Routing-Tabelle überein
  $\Longrightarrow$ Der 2. Tabelleneintrag ist ein Treffer
- Router 1 sendet das Paket über Port 1 an Host 2, weil der Port mit dem gleichen Netzwerk wie Host 2 verbunden ist

Wo kommen die Einträge in den Weiterleitungstabellen her?

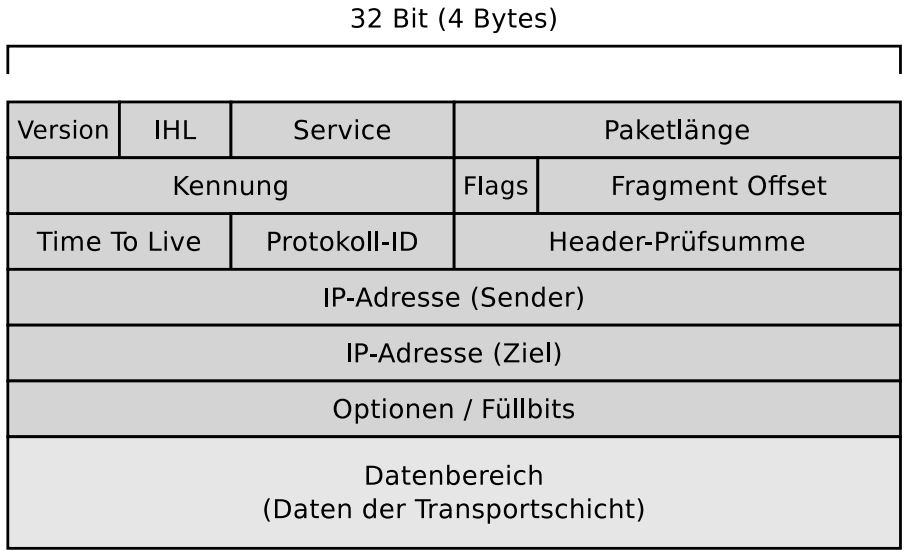
Durch **Wegbestimmung** (**Routing**) werden die Weiterleitungstabellen mit **Routing-Protokollen** erstellt
$\Longrightarrow$ siehe Foliensatz 8

Private Netze – Private IP-Adressen

- Auch im privaten LAN müssen IP-Adressen vergeben werden
  - Diese sollten nicht mit real existierenden Internetangeboten kollidieren
- Dafür existieren Adressbereiche mit privaten IP-Adressen
  - Diese Adressbereiche werden im Internet **nicht geroutet**

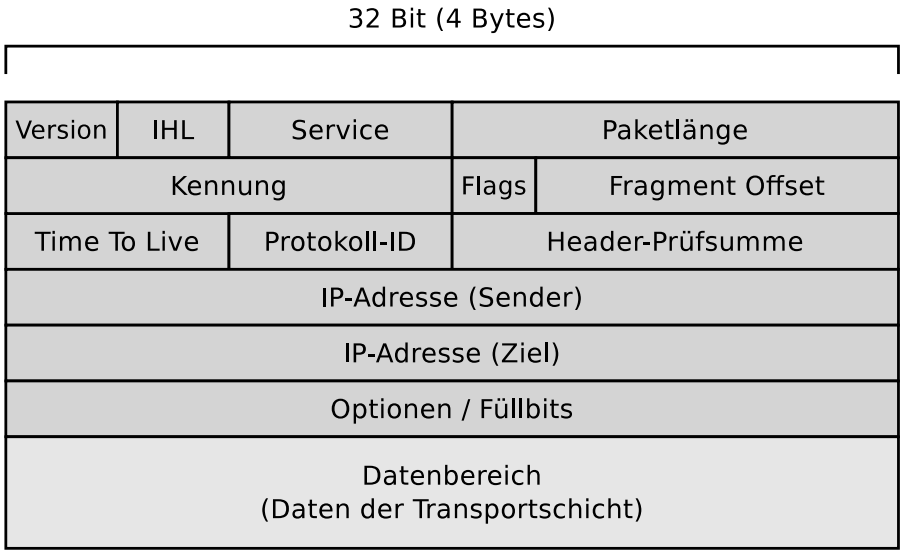
Adressbereich:	10.0.0.0 bis 10.255.255.255
CIDR-Notation:	10.0.0.0/8
Anzahl Adressen:	$2^{24} = 16.777.216$
Netzklasse:	Klasse A. 1 privates Netz mit 16.777.216 Adressen
Adressbereich:	172.16.0.0 bis 172.31.255.255
CIDR-Notation:	172.16.0.0/12
Anzahl Adressen:	$2^{20} = 1.048.576$
Netzklasse:	Klasse B. 16 private Netze mit jeweils 65.536 Adressen
Adressbereich:	192.168.0.0 bis 192.168.255.255
CIDR-Notation:	192.168.0.0/16
Anzahl Adressen:	$2^{16} = 65.536$
Netzklasse:	Klasse C. 256 private Netze mit jeweils 256 Adressen

Aufbau von IPv4-Paketen (1/6)



- **Version** (4 Bits)
  - Version $= 4 \Longrightarrow$ IPv4
  - Version $= 6 \Longrightarrow$ IPv6
- IP **H**eader **L**ength (4 Bits)
  - Länge des IP-Headers in Vielfachen von 4 Bytes
    - Beispiel: IHL $= 5 \Longrightarrow 5 * 4$ Bytes $= 20$ Bytes
  - Zeigt an, wo die Nutzdaten beginnen
- **Service** (8 Bits)
  - Hiermit ist eine Priorisierung von IP-Paketen möglich (Quality of Service)
  - Das Feld wurde mehrfach verändert (RFC 791, RFC 2474, RFC 3168)

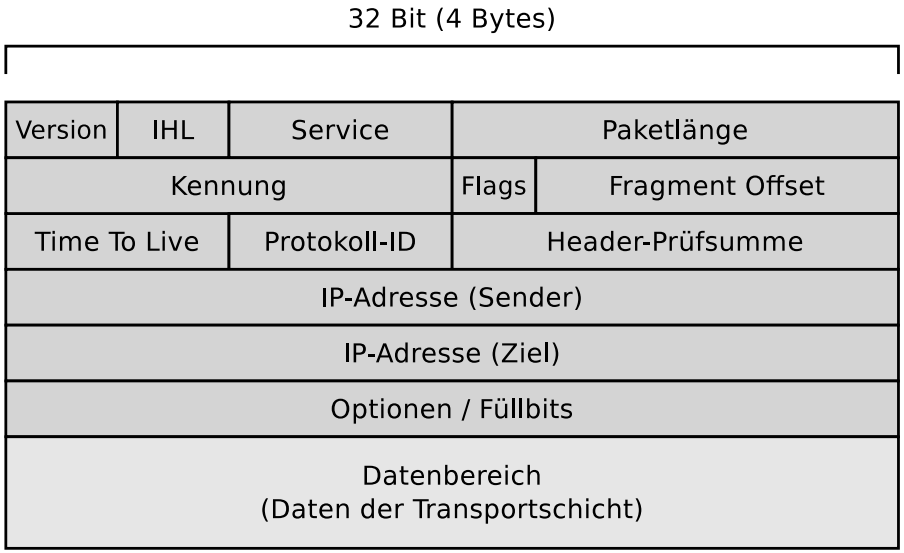
Aufbau von IPv4-Paketen (2/6)



- **Paketlänge** (16 Bits)
  - Länge des IP-Pakets (inkl. Header) in Bytes
  - Das Feld ist 16 Bits groß $\Longrightarrow$ max. Paketlänge in IPv4: 65.535 Bytes

Aufbau von IPv4-Paketen (3/6)

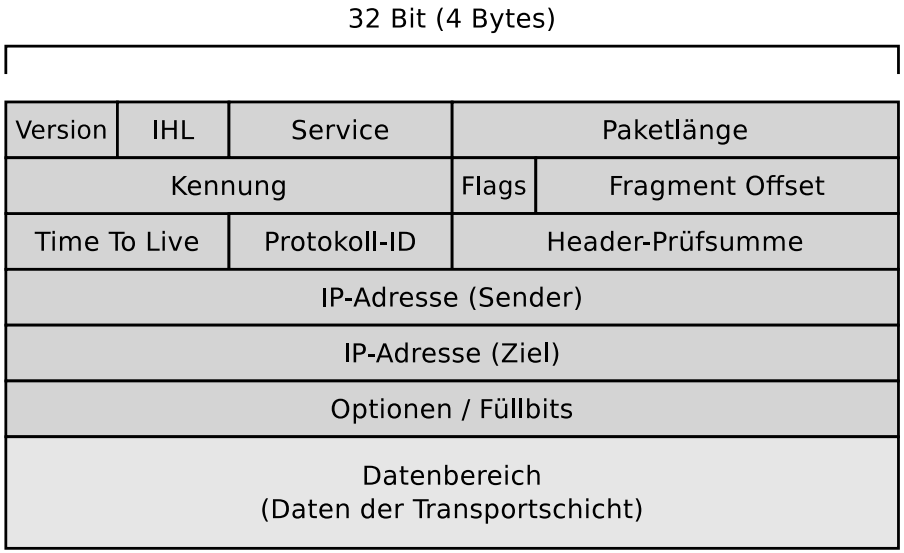
- Die Datenfelder **Kennung**, **Flags** und **Fragment Offset** steuern das Zusammensetzen fragmentierter IP-Pakete
- **Kennung** (16 Bits)
  - Eindeutige Kennung des IP-Pakets



- **Flags** (3 Bits)
  - Hier gibt der Sender an, ob das Paket fragmentiert werden darf und der Empfänger erfährt, ob noch weitere Fragmente folgen
- **Fragment Offset** (13 Bits)
  - Enthält eine Nummer, die bei fragmentierten Paketen besagt, ab welcher Position innerhalb des unfragmentierten Paketes das Fragment anfängt

Mehr Informationen über das Fragmentieren von IP-Paketen befinden sich auf den Folien 33 + 34

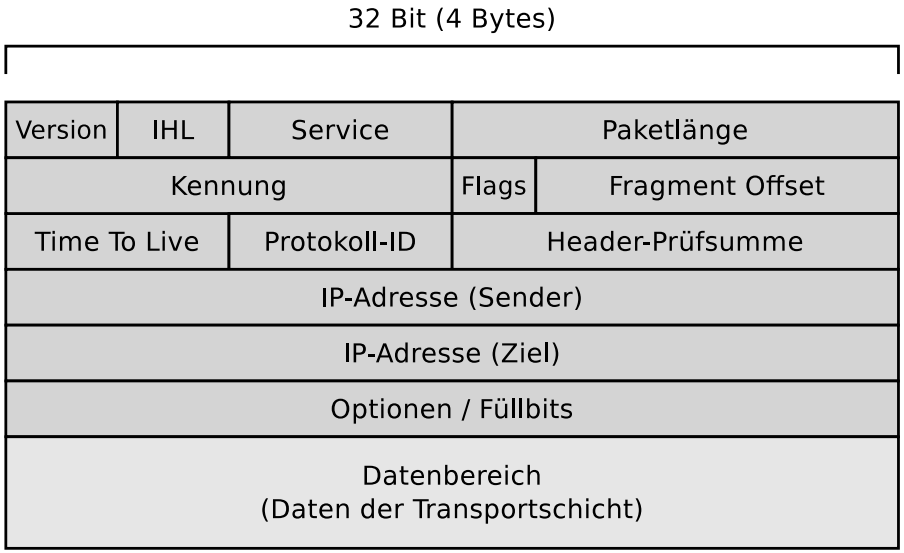
Aufbau von IPv4-Paketen (4/6)



- **Time To Live** (8 Bits)
  - Enthält die maximalen Hops
    - Jeder Router auf dem Weg zum Ziel verringert den Wert um eins
  - Das verhindert, das unzustellbare IP-Pakete endlos im Netz umherirren (kreisen)

Aufbau von IPv4-Paketen (5/6)

- **Protokoll-ID** (8 Bits)
  - Nummer des übergeordneten Protokolls in der Transportschicht
  - TCP-Segment $\Longrightarrow$ 6
  - UDP-Segment $\Longrightarrow$ 17
  - ICMP-Nachricht $\Longrightarrow$ 1
  - OSPF-Nachricht $\Longrightarrow$ 89

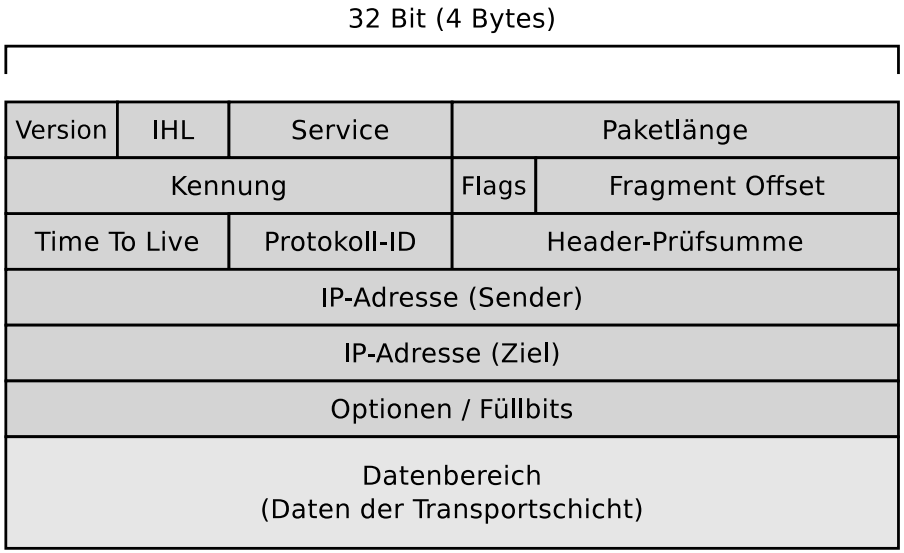


- Jedes IPv4-Paket enthält auch ein Feld für eine 16 Bits große Prüfsumme über die Daten des Headers
  - Weil sich bei jedem Router auf dem Weg zum Ziel der Inhalt des Datenfelds **Time To Live** ändert, müsste jeder Router die Prüfsumme überprüfen, neu berechnen und in den Header einsetzen

Router ignorieren die Prüfsumme üblicherweise, um die Pakete schneller weiterleiten zu können

Darum enthalten IPv6-Pakete auch kein Datenfeld für die Prüfsumme

Aufbau von IPv4-Paketen (6/6)



- **IP-Adresse (Sender)** (32 Bits) enthält die Adresse des Senders und das Datenfeld **IP-Adresse (Ziel)** die Adresse des Ziels
- **Optionen / Füllbits** kann Zusatzinformationen wie einen Zeitstempel enthalten
  - Dieses letzte Feld vor dem Datenbereich mit den Nutzdaten wird gegebenenfalls mit Füllbits (Nullen) aufgefüllt, weil es wie der vollständige Header auch ein Vielfaches von 32 Bits groß sein muss
- Der abschließende Datenbereich enthält die Daten der Transportschicht

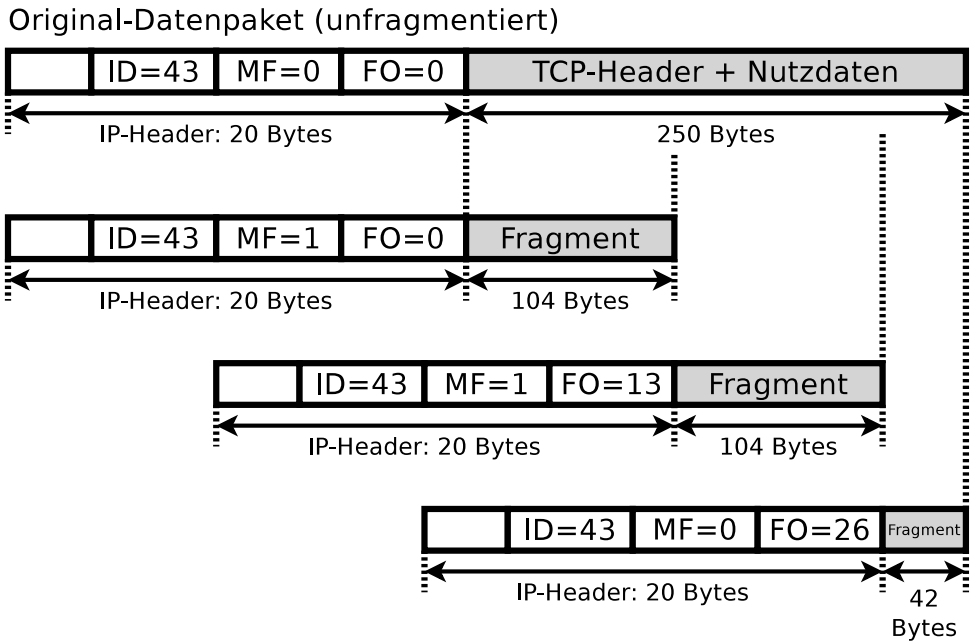
Fragmentieren (1/2)

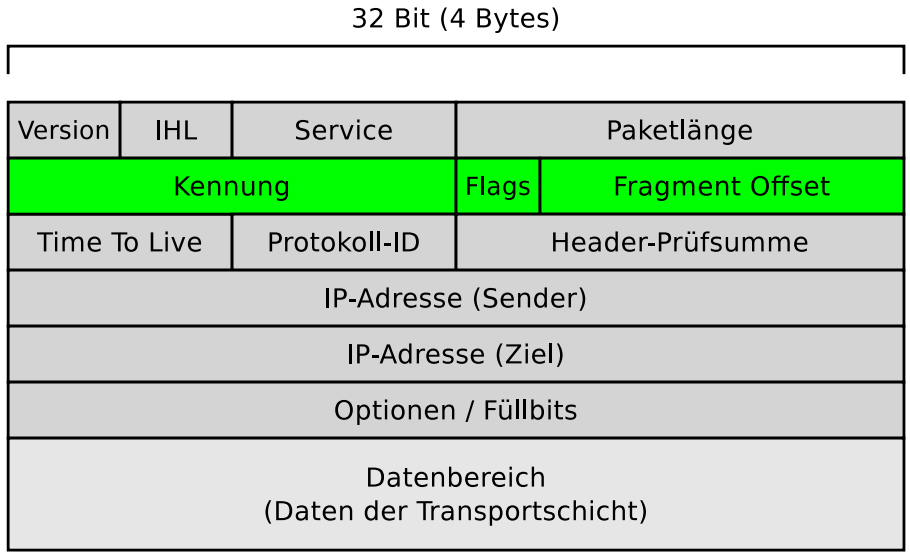
- Das Zerlegen (und Zusammensetzen) von IP-Paketen in kleinere Pakete (**Fragmente**) heißt **Fragmentieren**
  - Wird in der Regel von Routern durchgeführt
    - Fragmentieren kann aber auch der Sender durchführen
- Grund für Fragmentieren:
  - Die maximale Paketlänge hängt von der Vernetzungstechnologie ab
- Die **Maximum Transmission Unit** (MTU) gibt an, wie viele Nutzdaten ein Rahmen haben darf, also wie groß ein Paket sein darf
  - MTU von Ethernet: meist 1.500 Bytes
    - Bei Gigabit Ethernet gibt es auch *Jumboframes* mit bis zu 9.000 Bytes
  - MTU von WLAN (IEEE 802.11): 2.312 Bytes
  - MTU von Token Ring mit 4 Mbit/s (IEEE 802.5): 4.464 Bytes
  - MTU von Token Ring mit 16 Mbit/s: 17.914 Bytes
  - MTU von PPPoE (z.B. DSL): $\leq$ 1.492 Bytes
  - MTU von ISDN: 576 Bytes
  - MTU von FDDI: 4.352 Bytes

Fragmentieren (2/2)

- In IP-Paketen gibt es ein Flag, mit dem das Fragmentieren untersagt werden kann
  - Müsste ein Router ein Paket fragmentieren, weil es für die Weiterleitung zu groß ist, aber die Fragmentierung ist im Paket untersagt, verwirft der Router das Paket, da er es nicht weiterleiten kann
- Netzwerkgeräte, die nicht alle Fragmente eines IP-Pakets innerhalb einer bestimmten Zeitspanne (wenige Sekunden) erhalten, verwerfen alle empfangenen Fragmente
- Router können IP-Pakete in kleinere Fragmente unterteilen, wenn die MTU es nötig macht und es in den Paketen nicht untersagt ist
  - **Kein Router kann aber Fragmente eines Pakets zu einem größeren Fragment zusammenfügen**
    - Nur der Empfänger kann Fragmente zusammenfügen

Beispiel zur Fragmentierung (1/2)





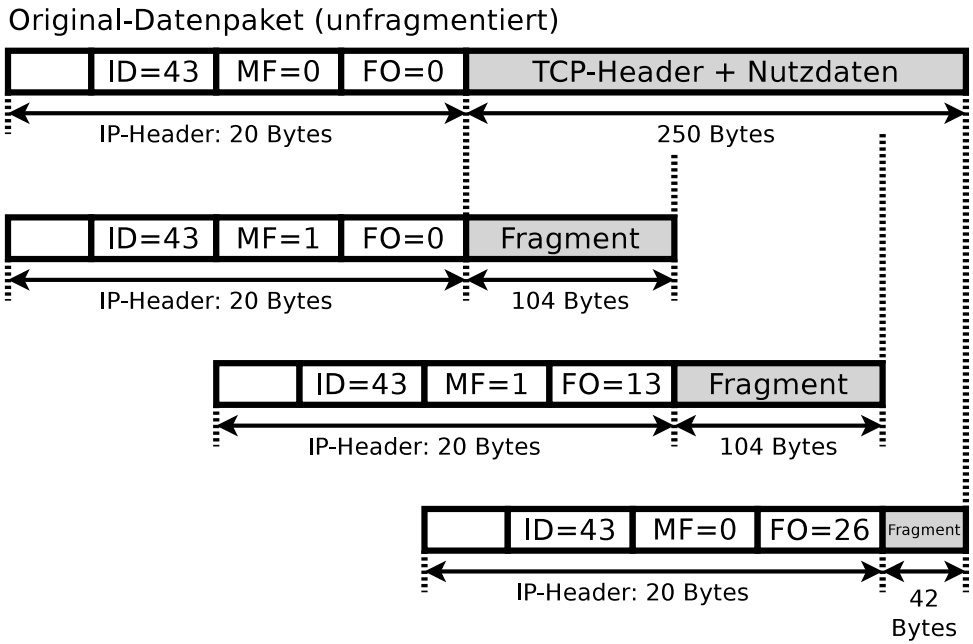
Quelle

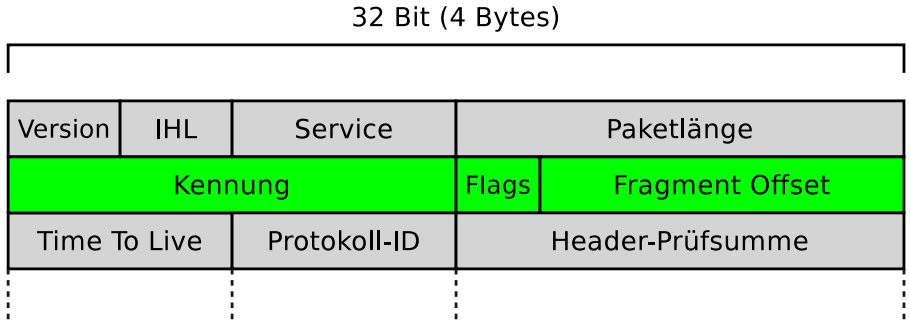
`http://www.netzmafia.de/skripten/netze/netz8.html`

- Ein 250 Bytes langes TCP-Segment wird via IP versandt
- Maximale Paketlänge: 124 Bytes
- Länge der IP-Header: 20 Bytes
- Paket-ID: 43

- Fragment Offset wird in 8-Byte-Schritten gezählt. **Die Nutzdaten in Fragmenten (außer beim letzten Fragment!) müssen also ein Vielfaches von 8 Bytes sein**
- Da alle Fragmente demselben Paket angehören, wird die ID für alle Fragmente beibehalten

Beispiel zur Fragmentierung (2/2)





- Im 1. Fragment ist `FO=0`
- MF-Bit=1 $\Longrightarrow$ mehr Fragmente folgen
- Im 2. Fragment ist `FO=13` ($104/8 = 13$) $\Longrightarrow$ Position des Fragments im unfragmentierten Paket
- MF-Bit=1 $\Longrightarrow$ mehr Fragmente folgen

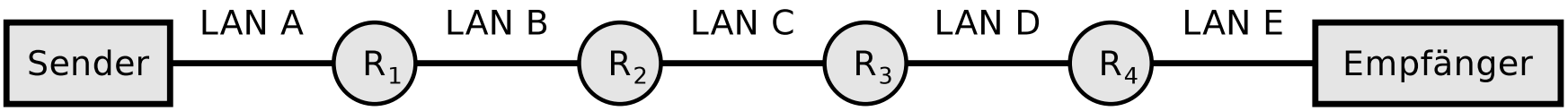
Quelle

`http://www.netzmafia.de/skripten/netze/netz8.html`

- Im 3. Fragment hat das MF-Bit den Wert 0, denn es ist das letzte Fragment von Paket 43
- `FO=26`, da schon $8 * 26 = 208$ Bytes Daten übertragen wurden

Weiteres Beispiel zur Fragmentierung (1/2)

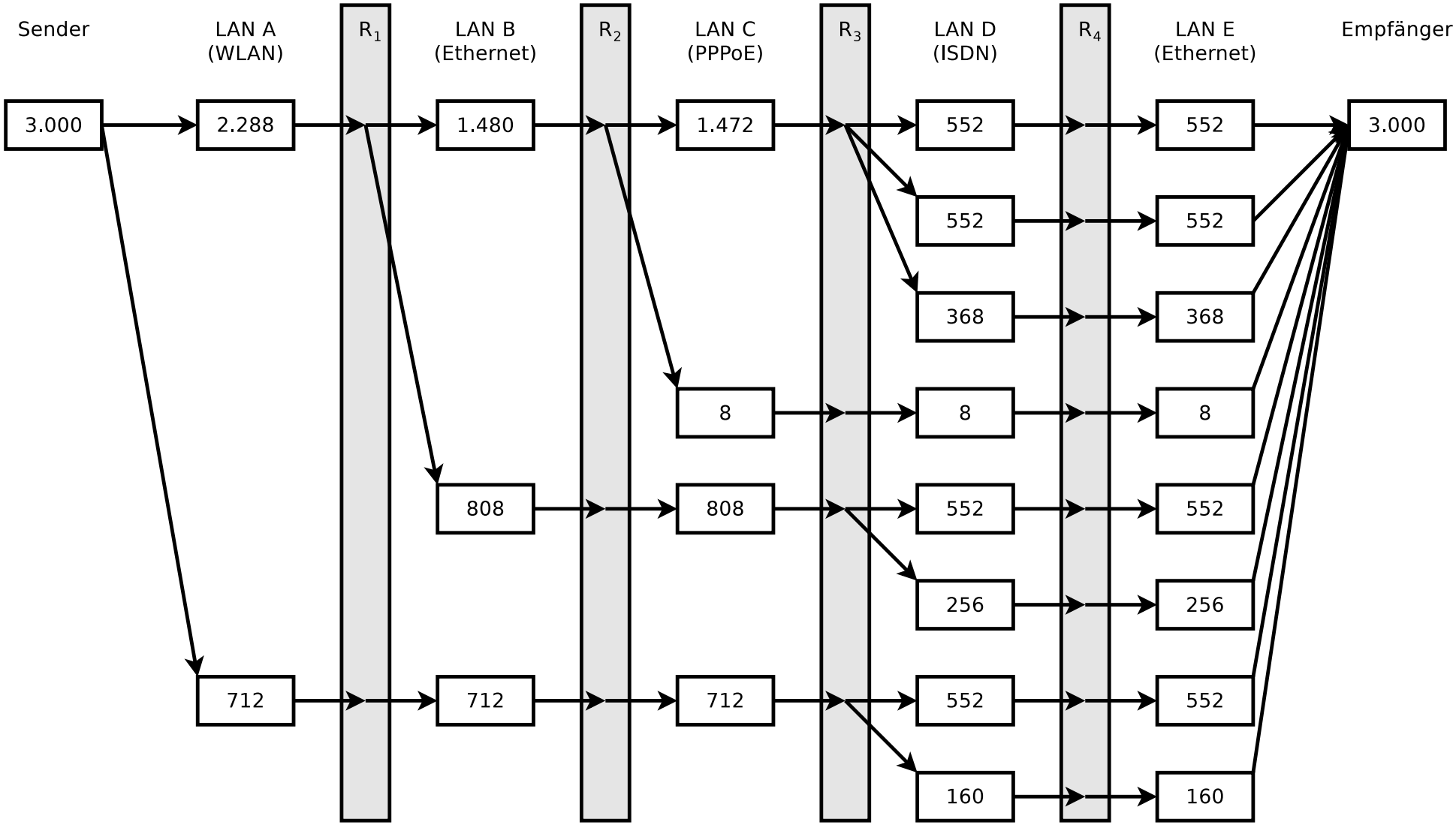
- 3.000 Bytes Nutzdaten soll via IP-Protokoll übertragen werden
- Die entstehenden Pakete müssen fragmentiert werden, weil sie über mehrere physische Netzwerke transportiert werden, deren MTU $<$ 3.000 Bytes ist



	LAN A	LAN B	LAN C	LAN D	LAN E
Vernetzungstechnologie	WLAN	Ethernet	PPPoE	ISDN	Ethernet
MTU [Bytes]	2.312	1.500	1.492	576	1.500
IP-Header [Bytes]	20	20	20	20	20
Max. Nutzdaten [Bytes] theoretisch	2.292	1.480	1.472	556	1.480
Vielfaches von 8	nein	ja	ja	nein	ja
Max. Nutzdaten [Bytes] in der Praxis	2.288	1.480	1.472	552	1.480

- Zeigen Sie grafisch wie das Paket fragmentiert wird und wie viele Bytes Nutzdaten jedes Fragment enthält

Weiteres Beispiel zur Fragmentierung (2/2)



Status von IPv4

ZEIT ONLINE | INTERNET

INTERNET PROTOKOLL

Bye, bye IPv4

Die letzten Adressblöcke des alten Internet Protokolls Version vier sind vergeben. Die Umstellung auf IPv6, die seit Jahren nicht vorankommt, wird nun beginnen müssen.

VON: Monika Ermert | 2.2.2011 - 16:36 Uhr

Im Netz hat eine neue Zeitrechnung begonnen: In der Nacht zum Dienstag hat die Internet Assigned Numbers Authority (IANA) die letzten freien IPv4-Adressen verteilt. Wer künftig IP-Adressen an Nutzer vergeben möchte, sei es für Mobiltelefone, PCs oder internetfähige Autos, muss sich mit der nächsten Generation von "Rufnummern" befassen, mit der Internet-Protokoll Version 6 – IPv6.

Das Internet-Protokoll ist Teil der komplexen Struktur, die notwendig ist, damit Computer miteinander Daten austauschen können. Es sorgt darin für die korrekte Vermittlung der transportierten Informationen. IPv4 nutzt Adressen mit einer Länge von 32 Bit, was die Zahl der insgesamt verfügbaren IPs auf 4.294.967.296 oder 4,2 Milliarden Stück beschränkte.

Das klingt viel. Aber bei 6,5 Milliarden Menschen weltweit und angesichts des Trends, mehr und mehr Geräte internetfähig zu machen, ist seit Jahren klar, dass die IPv4-Adressen knapp werden. Netzanbieter nutzten daher dynamische Adressen, vergaben also keine festen für jedes einzelne Gerät. Doch auch diese Technik ist begrenzt, weswegen seit vielen Jahren an einem neuen Internet-Protokoll gearbeitet wurde.

IPv6 basiert auf längeren Nummern und bietet damit für die Zukunft die nicht mehr so richtig vorstellbare Zahl von 340 Sextillionen eindeutiger Internetadressen. Jedes Sandkorn könnte damit künftig eine IP-Adresse bekommen.

Bis heute allerdings kam die technische Umstellung nur langsam voran. Nun sind jedoch die letzten freien IPv4-Blöcke an den für Asien zuständigen regionalen IP-Adressverwalter vergeben worden. Bis diese an die einzelnen Netzbetreiber und deren Kunden verteilt sind, wird es noch eine Weile dauern. Außerdem bekommt jede der weltweit fünf Verwaltungen in den kommenden Tagen noch eine Reserve von 16 Millionen IPv4-Adressen, doch der Zeitraum ist absehbar.

Diagnose und Fehlermeldungen mit ICMP

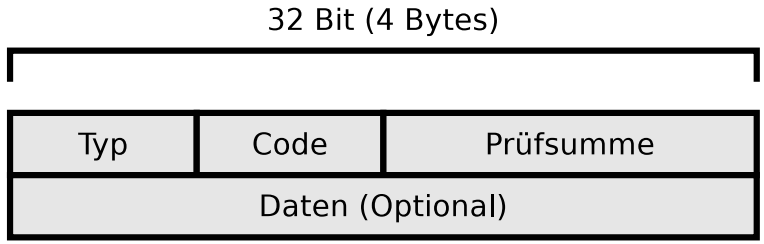
- Das **Internet Control Message Protocol** (ICMP) ermöglicht den Austausch von...
  - Diagnosemeldungen
  - Steuernachrichten
  - Fehlermeldungen
- ICMP ist ein Bestandteil (*Partnerprotokoll*) von IPv4
  - Es wird aber wie ein eigenständiges Protokoll behandelt

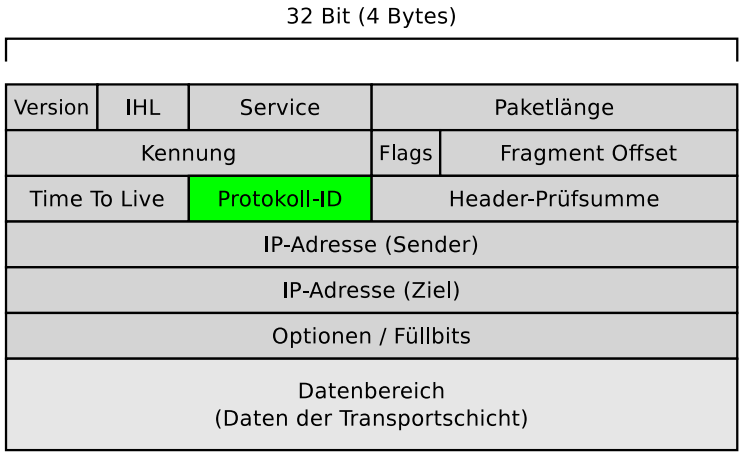
Für IPv6 existiert mit ICMPv6 ein ähnliches Protokoll

- Alle Router und Endgeräte können mit ICMP umgehen
- Typische Situationen, wo ICMP zum Einsatz kommt:
  - Ein Router verwirft ein IP-Paket, weil er nicht weiß, wie er es weiterleiten kann
  - Nur ein Fragment eines IP-Pakets kommt am Ziel an
  - Das Ziel eines IP-Pakets ist unerreichbar, weil die Time To Live (TTL) abgelaufen ist

ICMP

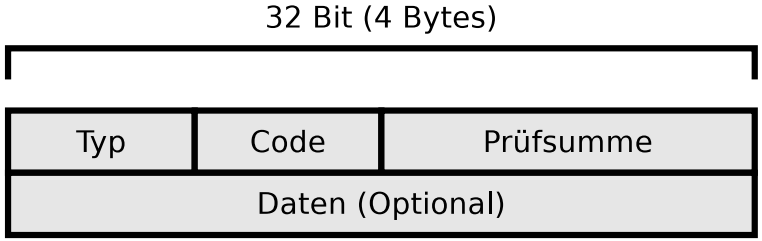
- Eine Anwendung, die ICMP-Pakete versendet, ist das Programm `ping`
- ICMP definiert verschiedene Informationsnachrichten, die ein Router zurücksenden kann





- ICMP-Nachrichten werden im Nutzdatenteil von IPv4-Paketen übertragen
  - Im Header des IPv4-Pakets steht dann im Datenfeld **Protokoll-ID** der Wert 1
  - Bei ICMPv6 ist die Protokoll-ID 58
- Kann ein ICMP-Paket nicht zugestellt werden, wird nichts unternommen

ICMP-Nachrichten

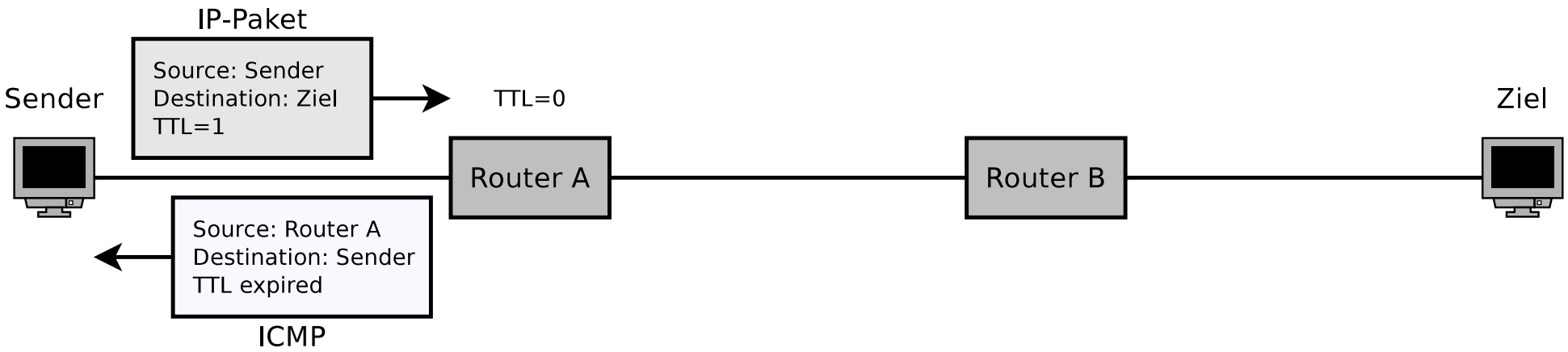


- Das Datenfeld **Typ** im ICMP-Header gibt den Nachrichtentyp an
- **Code** spezifiziert die Art der Nachricht innerhalb eines Nachrichtentyps
- Die Tabelle enthält einige Nachrichtentyp-Code-Kombinationen

Typ	Typname	Code	Bedeutung
0	Echo-Antwort	0	Echo-Antwort (Antwort auf `ping`)
3	Ziel nicht erreichbar	0	Netz unerreichbar
		1	Ziel unerreichbar
		2	Protokoll nicht verfügbar
		3	Port nicht verfügbar
		4	Fragmentierung nötig, aber im IP-Paket untersagt
		13	Firewall des Ziels blockt IP-Paket
5	Umleitung (Redirect)	0	Router informiert über eine bessere Route (IP des ersten Hops) zum Zielnetz
		1	Router informiert über eine bessere Route (IP des ersten Hops) zum Ziel-Host
8	Echo-Anfrage	0	Echo-Anfrage (`ping`)
11	Zeitlimit überschritten	0	TTL (Time To Live) abgelaufen
		1	Zeitlimit während der Defragmentierung überschritten

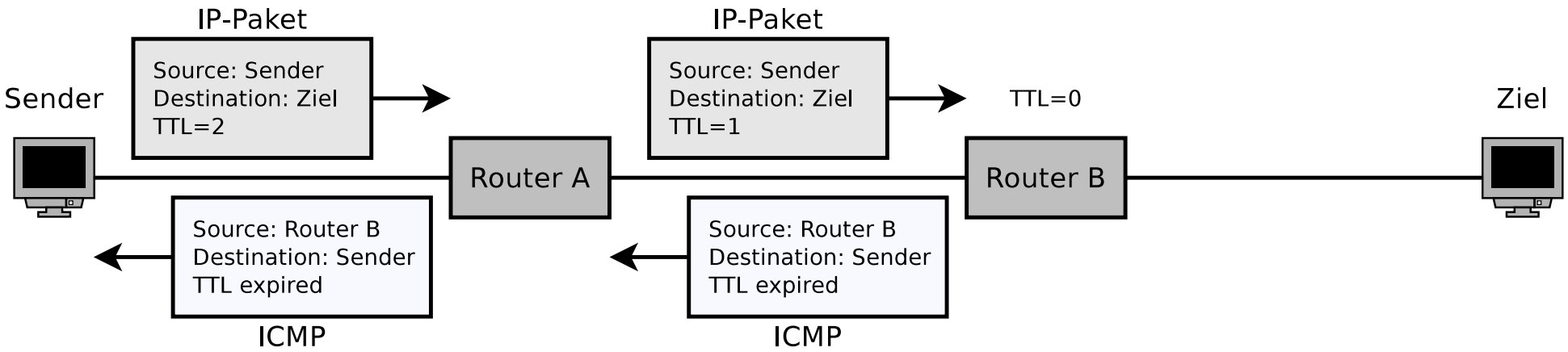
Das ICMP-Protokoll enthält noch viele weitere Nachrichtentyp-Code-Kombinationen (siehe RFC 792), aber die meisten wurden in der Praxis selten oder nie verwendet und gelten als veraltet (siehe RFC 6633 und RFC 6918)

Anwendungsbeispiel für ICMP: `traceroute` (1/3)



- Ein weiteres Anwendungsbeispiel für ICMP: Das Werkzeug `traceroute`
- `traceroute` ermittelt, über welche Router Datenpakete bis zum Ziel vermittelt werden
  - Es misst auch die Rundlaufzeit = Round Trip Time (RTT) vom Sender zu jedem Router
- Der Sender schickt ein IP-Paket an den Empfänger mit `TTL=1`
- Router A empfängt das IP-Paket, setzt `TTL=0`, verwirft das IP-Paket und sendet eine ICMP-Nachricht vom Nachrichtentyp 11 und Code 0 an den Sender

Anwendungsbeispiel für ICMP: `traceroute` (2/3)

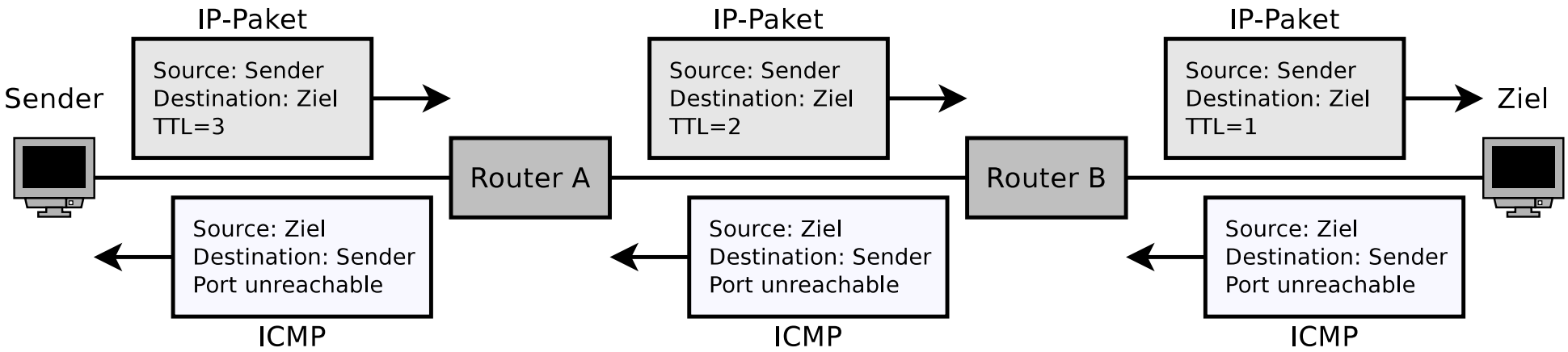


- Daraufhin schickt der Sender ein IP-Paket an den Empfänger mit `TTL=2`
- Das IP-Paket wird von Router A weitergeleitet
  - Dabei wird auch der Wert von `TTL` dekrementiert
- Router B empfängt das IP-Paket, setzt `TTL=0`, verwirft das IP-Paket und sendet eine ICMP-Nachricht vom Nachrichtentyp `11` und Code `0` an den Sender

Achtung! Es gibt verschiedene Implementierungen von `traceroute`

`tracert` unter Windows verwendet standardmäßig ICMP aber `traceroute` unter Linux und Mac OS X verwendet standardmäßig UDP. Der Einsatz von ICMP kann aber via Kommandozeilenparameter `-I` erzwungen werden. Alternativ ist auch TCP möglich.

Anwendungsbeispiel für ICMP: `traceroute` (3/3)



- Sobald der Wert von `TTL` groß genug ist, dass der Empfänger erreicht wird, sendet dieser eine ICMP-Nachricht vom Nachrichtentyp 3 und Code 3 an den Sender
- So kann der Sender via ICMP den Weg zum Empfänger nachvollziehen

```
$ traceroute -q 1 wikipedia.de
traceroute to wikipedia.de (134.119.24.29), 30 hops max, 60 byte packets
 1  fritz.box (10.0.0.1) 1.834 ms
 2  p3e9bf6a1.dip0.t-ipconnect.de (62.155.246.161) 8.975 ms
 3  217.5.109.50 (217.5.109.50) 9.804 ms
 4  ae0.cr-polaris.fra1.bb.godaddy.com (80.157.204.146) 9.095 ms
 5  ae0.fra10-cr-antares.bb.gdinf.net (87.230.115.1) 11.711 ms
 6  ae2.cgn1-cr-nashira.bb.gdinf.net (87.230.114.4) 13.878 ms
 7  ae0.100.sr-jake.cgn1.dcnet-emea.godaddy.com (87.230.114.222) 13.551 ms
 8  wikipedia.de (134.119.24.29) 15.150 ms
```

Das Beispiel sendet pro Hop nur ein Paket (`-q 1`). Darum enthält die Ausgabe pro Hop nur einen Round-Trip-Time-Wert (RTT). Standardmäßig sendet `traceroute` drei Pakete pro Hop, was die Anzeige von drei RTT-Werten zur Folge hat. Starke Schwankungen der RTT sind ein Indikator für Netzwerkprobleme oder Ressourcenengpässe

IPv6-Adressen und Netze

- IPv6-Adressen bestehen aus 128 Bits (16 Bytes)
  - Daher können 2^{128}, also $\approx 3{,}4 * 10^{38}$ Adressen dargestellt werden
- Einführung ist wegen des begrenzten Adressraums von IPv4 sinnvoll
- Problem: Dezimaldarstellung ist unübersichtlich
  - Aus diesem Grund stellt man IPv6-Adressen hexadezimal dar
    - Je 4 Bits werden als eine hexadezimale Zahl dargestellt
  - Je 4 Hexadezimalzahlen werden zu Blöcken gruppiert
  - Die Blöcke werden durch Doppelpunkte getrennt
    Beispiel: `2001:0db8:85a3:08d3:1319:8a2e:0370:7344`

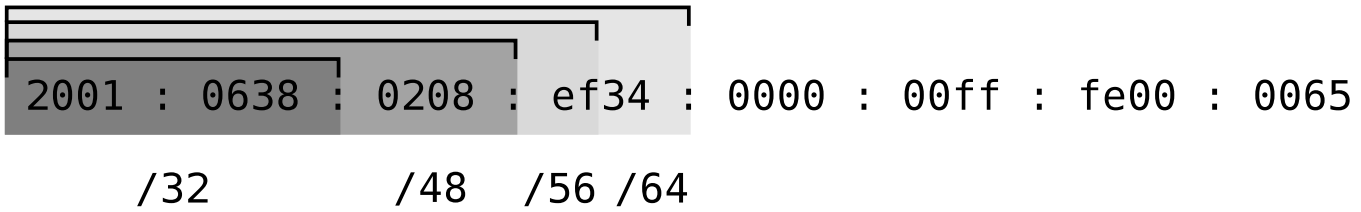
- Die letzten 4 Bytes (32 Bits) einer IPv6-Adresse dürfen auch in dezimaler Notation geschrieben werden
- Das ist sinnvoll, um den IPv4-Adressraum in den IPv6-Adressraum einzubetten
  $\Longrightarrow$ siehe Folie 74

Addresses only for documentation

`2001:db8::/32` $\Longrightarrow$ Addresses only for documentation purposes

Subnetze in IPv6

- (Sub-)Netzmasken gibt es bei IPv6 nicht
  - Die Unterteilung von Adressbereichen in Subnetze geschieht durch die Angabe der Präfixlänge
- IPv6-Netze werden in CIDR-Notation angegeben
  - Die Adresse eines einzelnen Geräts hat manchmal ein angehängtes `/128`
  - Ein Beispiel ist die Loopback-Adresse von IPv6: `::1/128`
    - Alle Bits – außer das letzte Bit – haben den Wert `0`
      (Bei IPv4 ist die Loopback-Adresse: `127.0.0.1`)
  - Internetprovider (ISP) oder Betreiber großer Netze bekommen die ersten 32 oder 48 Bits von einer Regional Internet Registry (RIR) zugewiesen
    - Diesen Adressraum teilt der Provider oder Netzbetreiber in Subnetze auf
    - **Endkunden bekommen meist ein `/64`- oder sogar `/56`-Netz zugeteilt**



- Bekommt ein Endkunde ein `/56`-Netz zugeteilt, sind die 8 Bits zwischen dem Präfix und der Interface Identifier das **Subnet Präfix**

IPv6-Adressen vereinfachen

- Regeln zur Vereinfachung (RFC 5952):
  - Führende Nullen innerhalb eines Blocks dürfen ausgelassen werden
  - Aufeinanderfolgende Blöcke, deren Wert 0 (bzw. 0000) ist, dürfen **innerhalb einer IPv6-Adresse genau 1x** ausgelassen werden
    - Das Auslassen wird durch 2 aufeinander folgende Doppelpunkte angezeigt
  - Gibt es mehrere Gruppen aus Null-Blöcken, ist es empfehlenswert die Gruppe mit den meisten Null-Blöcken zu kürzen
- Beispiele:
  - Die IPv6-Adresse von `j.root-servers.net` ist:
    `2001:0503:0c27:0000:0000:0000:0002:0030`
    $\Longrightarrow$ `2001:503:c27::2:30`

Schreibweise von IPv6-Adressen (URLs)

- IPv6-Adressen werden in eckigen Klammern eingeschlossen
- Portnummern werden außerhalb der Klammern angehängt
  `http://[2001:500:1::803f:235]:8080/`
- Das verhindert, das die Portnummer als Teil der IPv6-Adresse interpretiert wird

Struktur von IPv6-Adressen und Netzen

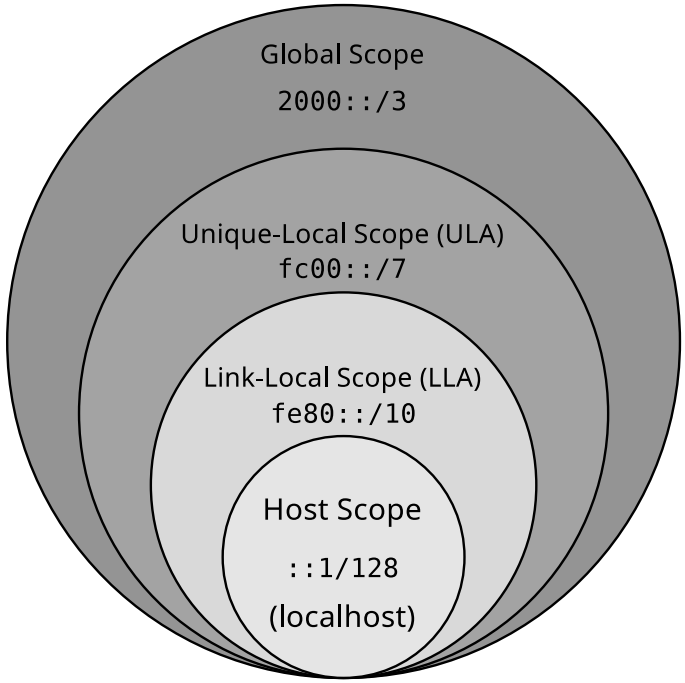
- IPv6-Adressen bestehen aus 2 Teilen

64 Bits	64 Bits
Network Prefix	Interface Identifier
2001:638:208:ef34	:0:ff:fe00:65

1. **Präfix** (Network Prefix)
   - Kennzeichnet das Netz
2. **Interface Identifier** (Interface-ID)
   - Kennzeichnet eine Netzwerkgerät in einem Netz
     - Die Konfiguration bzw. Zuweisung der Interface-ID ist auf verschiedene Arten möglich (siehe Foliensatz 57)

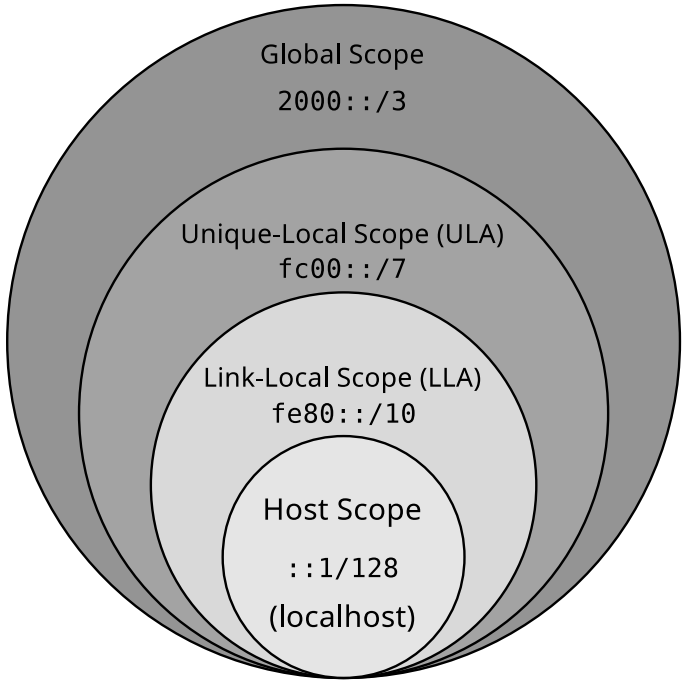
Gültigkeitsbereiche – Scopes (1/4)

- IPv6 unterscheidet nicht nur zwischen privaten und öffentlichen Adressen (wie IPv4), sondern auch mehrere Gültigkeitsbereiche (sog. *Scopes*)
- Jede IPv6-Adresse hat einen sogenannten Scope
- Der Gültigkeitsbereich ist der Teil eines Netzes, in dem die zugehörige Adresse als gültig betrachtet und weitergeleitet wird
- **Host Scope**: Loopback-Adresse
  - Die Loopback-Adresse ist `::1/128` $\Longrightarrow$ `0:0:0:0:0:0:0:1/128`

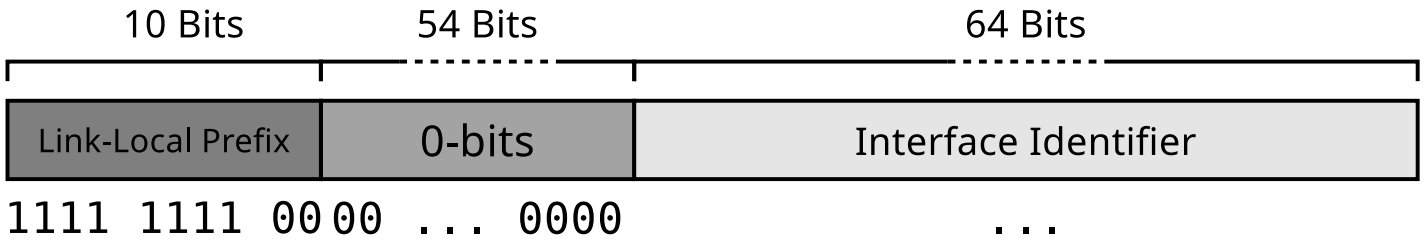


Gültigkeitsbereiche – Scopes (2/4)

- **Link-Local Scope**: Link-Local (Unicast) Addresses (LLA)
  - Jede Netzwerkschnittstelle benötigt zu jeder Zeit eine Link-Local-Adresse
  - Link-Local-Adressen `fe80::/10` sind nur im lokalen Netz gültig
  - Router leiten Pakete mit diesen Adressen nicht weiter

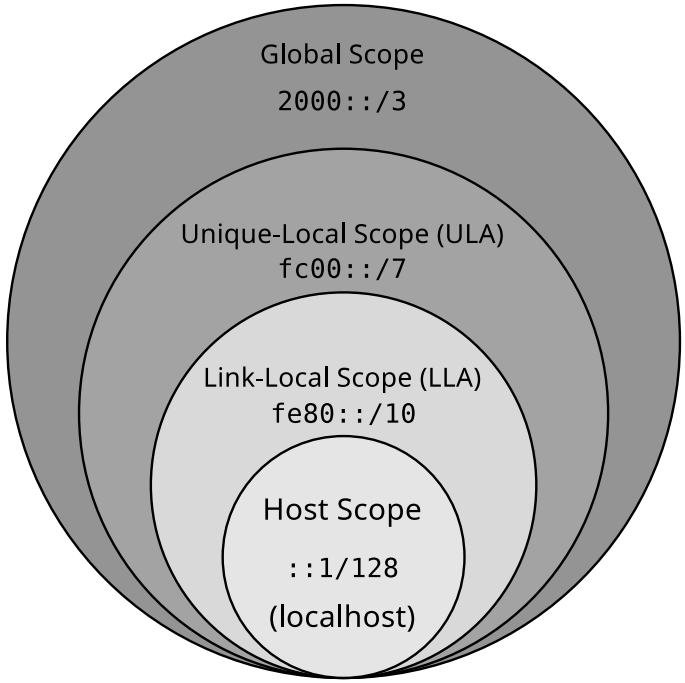


- Sie werden für die Kommunikation innerhalb des lokalen Netzes (z.B. WLAN oder Ethernet) benötigt und dienen u.a. dazu, eine global gültige IPv6-Adresse zu generieren oder über DHCPv6 zu beziehen



Gültigkeitsbereiche – Scopes (3/4)

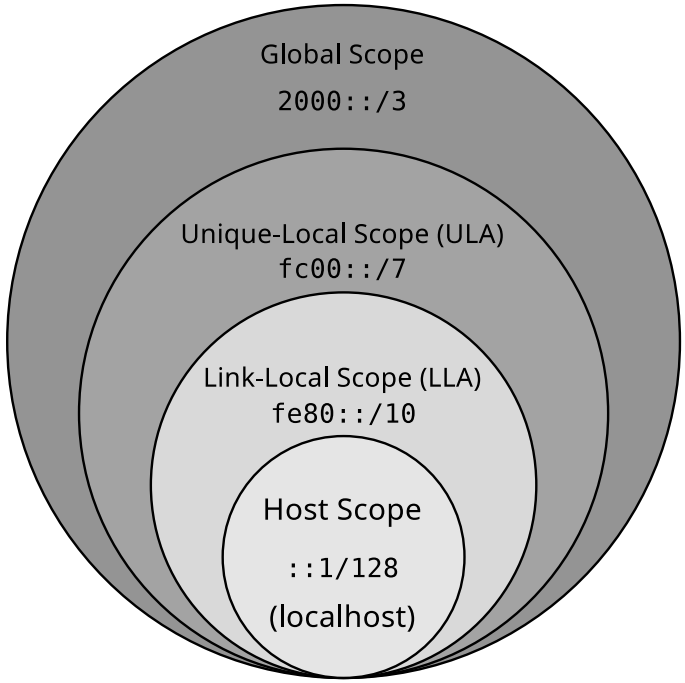
- **Unique-Local Scope**: Unique Local Addresses (ULA)
  - Router sollen die Adressen `fc00::/7` ($\Longrightarrow$ `fc00...` bis `fdff...`) nicht außerhalb des lokalen Verwaltungsbereichs weiterleiten
  - *Private Adressen* zur lokalen Kommunikation innerhalb eines Verwaltungsbereichs (Organisation oder Standort)



- `fc...` $\Longrightarrow$ zugewiesene eindeutige ULA
  - Auf das Präfix `fc` folgt eine 40 Bits lange zugewiesene (eindeutige) Site-ID und eine 16 Bits lange Subnetz-ID
  - Weltweit gültige, eindeutige, von einem Provider vergebene Adressen
- `fd...` $\Longrightarrow$ lokal generierte ULA
  - Auf das Präfix `fd` folgt eine 40 Bits lange selbständig generierte (wahrscheinlich eindeutige) Site-ID und eine 16 Bits lange Subnetz-ID

Gültigkeitsbereiche – Scopes (4/4)

- **Global Scope**: Global Unicast Addresses
  - Router leiten die Adressen `2000::/3` ($\Longrightarrow$ `2000...` bis `3fff...`) weiter



Site-Local Scope

Dieser scope (siehe RFC 1884) definiert Adressen (`fec0::/10`), die innerhalb des Netzwerks einer Organisation (*inside a site*), gültig sind, gilt seit 2004 als veraltet – *obsolet* (siehe RFC 3879).
Der Scope wurde 2005 (siehe RFC 4193) durch Unique Local Addresses (ULA) ersetzt

Unique Local Addresses – ULA (RFC 4193)

- Die Tabelle zeigt das Adressierungsschema für statisches IPv6 RFC 4193

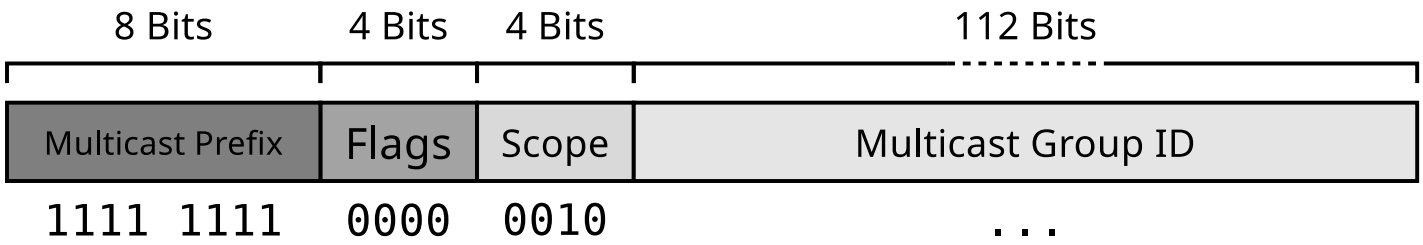
Präfix/Länge	**Global-ID**	**Subnetz-ID**	**Interface-ID**
fd00::/8	40 Bits	16 Bits	64 Bits
fd00::/8	12:3456:789a	0001	0000:0000:0000:0001

Resultierende IPv6-Adresse: fd12:3456:789a:0001:0000:0000:0000:0001
Vereinfachte IPv6-Adresse: fd12:3456:789a:1::1

- ULAs werden dort eingesetzt, wo eine Netz-ID (Netzwerk-Präfix) von einem Provider bereitgestellt wird
- Wird nur für lokalen Umgebungen (lokaler Verwaltungsbereich) genutzt
  - Diese Adressen werden nicht in das globale Internet geroutet
    - Bei lokal generierten ULAs sind Adresskonflikte möglich (aber sehr unwahrscheinlich)
  - ULAs sind analog zu privaten Adressen in IPv4
    - Sie werden innerhalb der Verwaltungsdomäne (Standort oder Organisation) verwendet

IPv6-Multicast-Adressen (1/2)

- **IPv6 definiert keine Broadcast-Adressen**
  - **Multicast-Adressen emulieren die Broadcast-Funktionalität**



- Bei Multicast-Adressen haben die ersten 8 Bits den Wert `11111111`
  - Somit haben sie das **Multicast-Präfix** `ff::/8`
- Auf das Präfix folgen 4 Bits **Flags** (1-Bit-Felder) und 4 Bits für den **Scope** (Gültigkeitsbereich)

Die Flags sind im Rahmen dieser Vorlesung irrelevant und haben in der Praxis meist den Wert `0`

- In der Praxis relevante Gültigkeitsbereiche (Scopes) sind die Werte `1` und `2` ($\Longrightarrow$ siehe nächste Folie)

IPv6-Multicast-Adressen (2/2)

Präfix	**Scope**	**Bedeutung**
`ff01`	interface-local	Pakete an diese Adresse verlassen die Schnittstelle nicht $\Longrightarrow$ **Loopback**
`ff02`	link-local	Pakete werden von Routern nicht weitergeleitet $\Longrightarrow$ bleiben im Subnetz

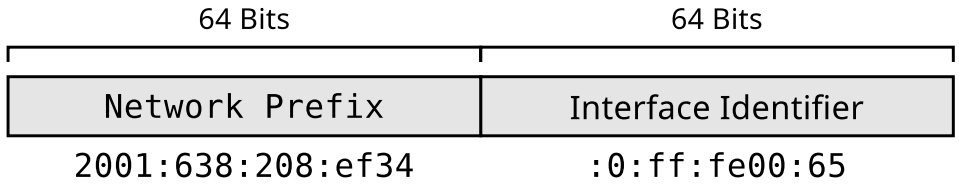
Der veraltete (siehe RFC 3879) Gültigkeitsbereich `ff05` (site-local) definiert, dass ein Paket nicht von Border-Routern weitergeleitet wird $\Longrightarrow$ es das Netz einer Organisation nicht verlässt

- Auf den Gültigkeitsbereich folgt die Multicast-Gruppen-ID
- Die Tabelle enthält einige gängige Multicast-Adressen

IPv6-Adressen	**Scope**	**Bedeutung (Adressen...)**
`ff01::1`	interface-local	alle Knoten im lokalen Netz
`ff01::2`	interface-local	alle lokalen Router
`ff02::1`	link-local	alle Knoten im lokalen Netz $\Longrightarrow$ emuliert **Broadcast**
`ff02::2`	link-local	alle lokalen Router
`ff02:1:2`	link-local	all lokale DHCPv6-Server
`ff02::9`	link-local	alle lokalen Router, die das Routing-Protokoll RIP einsetzen
`ff02::5`	link-local	alle lokalen Router, die das Routing-Protokoll OSPF einsetzen
`ff02::6`	link-local	alle lokalen designierten Router, die OSPF einsetzen
`FF02::F`	link-local	alle Geräte, die UPnP (Universal Plug and Play) einsetzen

Konfiguration der Schnittstellenkennung (Interface-ID)

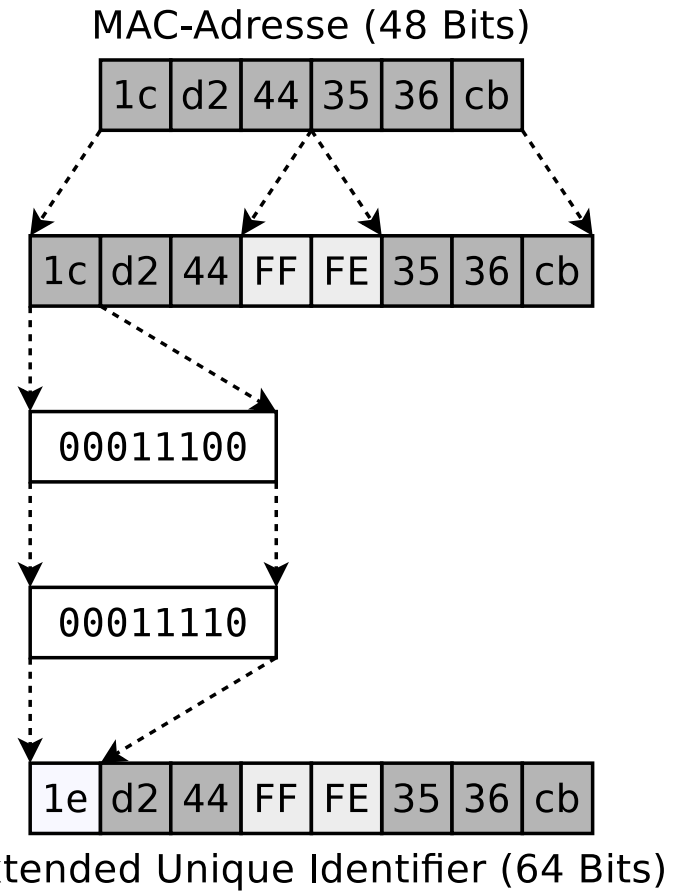
- Die **Interface-ID** kann auf verschiedene Arten konfiguriert werden



1. **Statische manuelle Adressierung**
   - Interface ID manuell festlegen – möglich, aber unpraktisch
2. **Zustandslose automatische Adresskonfiguration** (RFC 4862)
   - Interface-ID (64 Bits) aus der MAC-Adresse (48 Bits) berechnen
     $\Longrightarrow$ Die Interface-ID heißt dann **Extended Unique Identifier** (EUI)
   - Optionale Erweiterung: **Stable Privacy Addresses** (RFC 7217)
     - Dauerhafte Interface-ID mit Hilfe eines zufälligen geheimen Schlüssels berechnen (ohne Verwendung der MAC-Adresse), um Anonymität zu gewährleisten
   - Optionale Erweiterung: **Privacy Extension** (RFC 4941)
     - Regelmäßige Berechnung einer neuen Interface-ID unter Verwendung einer Zufallszahl (ohne Verwendung der MAC-Adresse), für noch mehr Anonymität
3. **Netzwerkkonfiguration über DHCPv6 einstellen** (RFC 8415)
   - Einzige Möglichkeit zur Adresskonfiguration, die **zustandsbehaftet** (*stateful*) funktioniert

Stateless Address Autoconfiguration – SLAAC (RFC 4862)

- Automatische zustandslose IPv6-Adress-generierung aus der MAC-Adresse
- MAC in eine Host-ID umwandeln
  1. Die MAC-Adresse wird halbiert
     - 1.Teil bildet die ersten 24 Bits
     - 2.Teil bildet die letzten 24 Bits der modifizierten EUI-64-Adresse
  2. Bitmuster der 16 Bits in der Mitte der EUI-64-Adresse: `1111 1111 1111 1110` (hex: `FFFE`)
  3. Abschließend das siebte Bits invertieren
- Nachteil: Einfache Rückgewinnung der MAC-Adresse ($\Longrightarrow$ Datenschutzbedenken)



Router $\Longrightarrow$ Advertisement Daemon (`radvd`)

Für die automatische Vergabe von Netzwerk-Präfixen benötigt der Router einen `radvd` zur Verwaltung von Netzwerk-Präfixen im Netzwerk. Ohne `radvd` wird das Link-Local-Präfix `fe80::/64` zugewiesen

Das Invertieren des 7. Bits bei EUI-64 hat mit dem Universal/Local-Bit (U/L) bei MAC-Adressen zu tun. Hat das Bit den Wert 0, ist die MAC-Adresse global eindeutig. Hat es den Wert 1, wird sie lokal administriert (was bei SLAAC der Fall ist) und ist nicht global eindeutig

Stateless Address Autoconfiguration – Schritte

- In der Praxis arbeitet SLAAC in 5 Schritten
  1. Client erzeugt eine modifizierte EUI-64 Host-ID aus seiner eigenen MAC-Adresse
  2. Client erzeugt eine Link-Local-Adresse (LLA) mit Präfix `fe80::/64`

Durch die Verkettung beider Teile entsteht eine 128-Bit-IPv6-Adresse´, die im lokalen Netz gültig ist

  3. Client sendet zur Erkennung doppelter Adressen **Duplicate Address Detection** (DAD) eine **Neighbor Solicitation** (NS)-Nachricht mit seiner eigenen Link-Local-Adresse über Multicast im Link-Local-Netz und wartet, ob eine **Neighbor Advertisement** (NA)-als Antwort eintrifft (siehe Folie 63)
  4. Client sendet eine **Router Solicitation** (RS) Nachricht an den Router (siehe Folie 66)
  5. Router sendet eine **Router Advertisement** (RA)-Nachricht an den Client und übergibt das global gültige Netzwerk-Präfix (siehe Folie 65)

IPv6 Neighbor Discovery Protocol

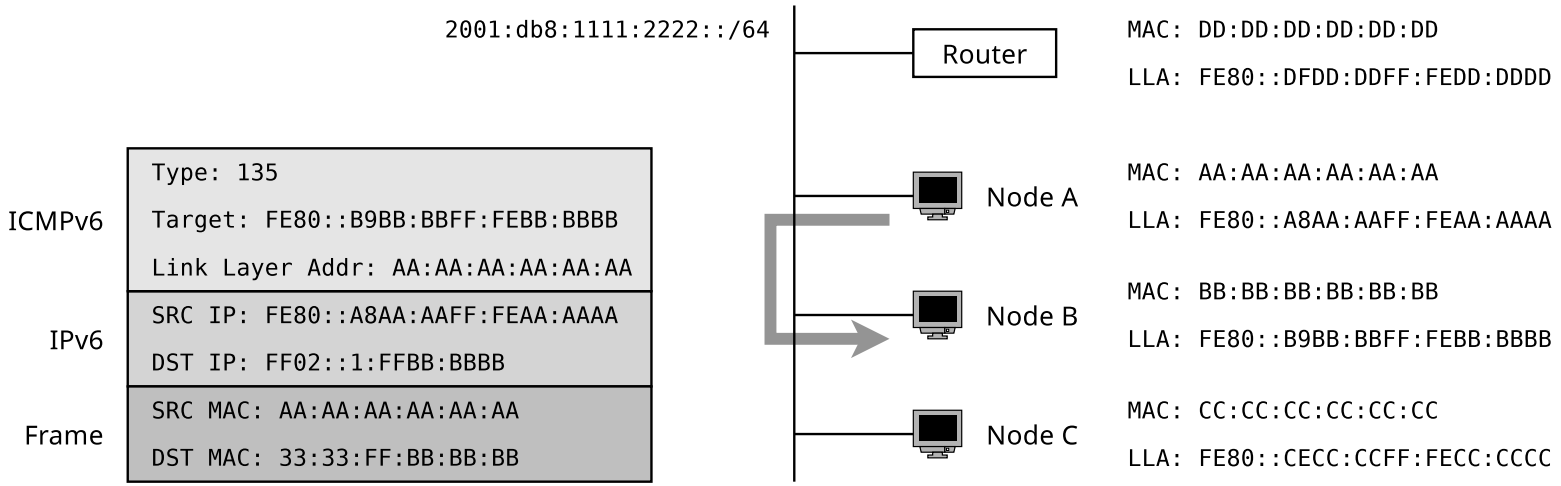
- IPv6 implementiert keine Broadcast-Adressen und es gibt keine zum Address Resolution Protocol (ARP) vergleichbare Lösung
  - Allerdings ist die Auflösung von IPs in MAC-Adressen auch hier nötig
- **Das Neighbor Discovery Protocol (NDP) löst MACs aus IPv6-Adressen auf und nutzt dafür Multicast-Adressen**

In IPv6 beschreibt der Begriff **Neighbor** Knoten, die sich im gleichen Netzwerk der Sicherungsschicht (Data Link Layer) befinden

- NDP-Nachrichten werden als Nutzdaten in ICMPv6-Nachrichten ausgetauscht
- NDP implementiert 5 Arten von Nachrichten
  - **Router Solicitation** (ICMPv6 Typ 133)
  - **Router Advertisement** (ICMPv6 Typ 134)
  - **Neighbor Solicitation** (ICMPv6 Typ 135)
  - **Neighbor Advertisement** (ICMPv6 Typ 136)
  - **Redirect Message** (ICMPv6 Typ 137)

Mit der **Redirect Message** informiert ein Router über eine bessere Route (anderer First Hop $\Longrightarrow$ anderer lokaler Router) für ein Ziel. Dieser Nachrichtentyp wird in dieser Vorlesung nicht weiter behandelt

Neighbor Solicitation – NS (1/2)

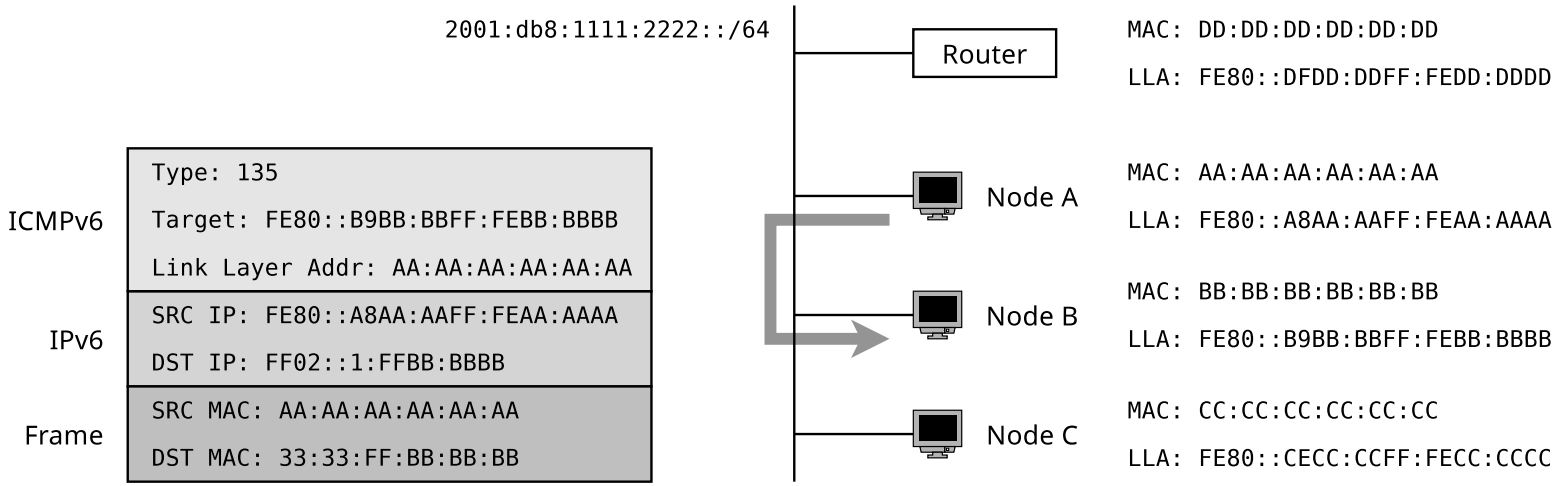


Die Nachricht Neighbor Solicitation (NS) ist die IPv6-Alternative zu einer ARP-Anfrage bei der Nutzung von IPv4

- Anforderung der MAC-Adresse eines Nachbarn
  - In der Abbildung fordert Knoten A die MAC-Adresse von Knoten B an
- Ziel-IP-Adresse im IPv6-Paket = **Solicited-node multicast address**
  - Jeder Knoten tritt einer Multicast-Gruppe für jede konfigurierte IPv6-Adresse bei
    - Die Multicast-Gruppe hat die Adresse `FF02::1:FFXX:XXXX`

`XX:XXXX` steht für die letzten 6 hexadezimalen Zeichen der Link-Local (Unicast) Adresse (LLA)

Neighbor Solicitation – NS (2/2)



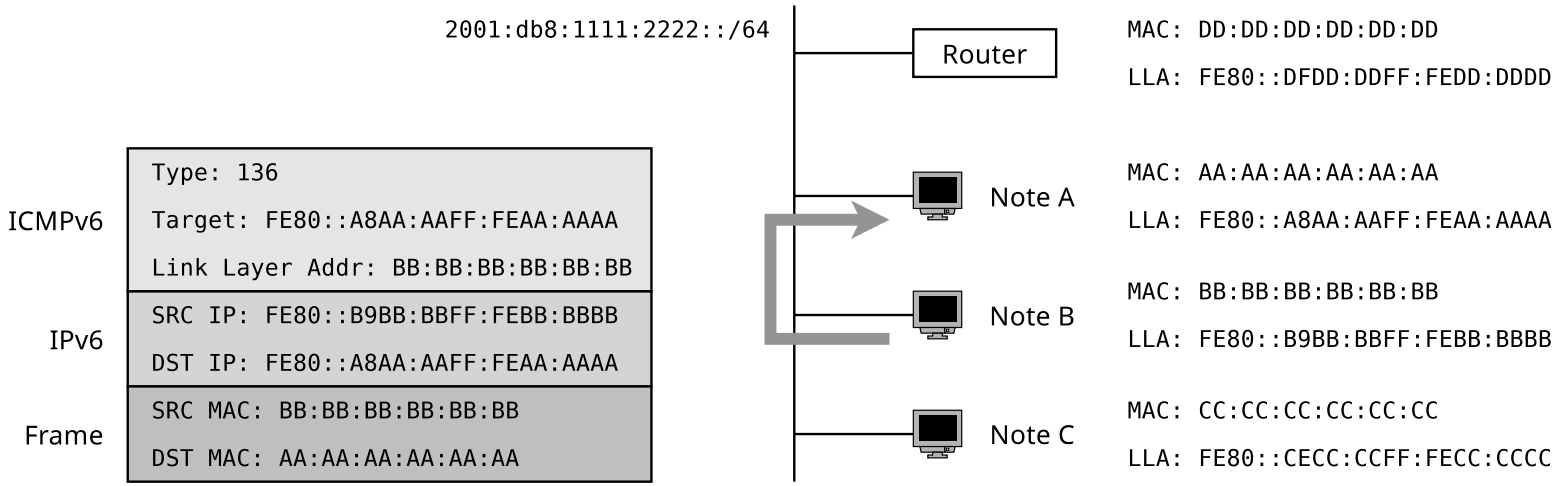
- Ziel-MAC-Adresse im Rahmen = **Multicast-MAC-Adresse**
  - Die Multicast-MAC-Adresse ist `33:33.XX:XX.XX:XX`

`XX:XX:XX:XX` steht für die letzten 8 hexadezimalen Zeichen der Multicast-Adresse des angefragten Knotens

Erkennung doppelter Adressen mit Neighbor Solicitation

- Die Nachricht Neighbor Solicitation (NS) wird auch zur Erkennung von Adressduplikaten – **Duplicate Address Detection (DAD)** verwendet
  - Wenn ein Knoten eine vorläufige (*tentative*) IPv6-Adresse für sich selbst generiert, muss er prüfen, dass kein anderer Knoten im Netz diese Adresse bereits verwendet
- Der Knoten sendet eine Nachricht Neighbor Solicitation (NS) an die Adresse, die er selbst verwenden möchte
  - Absenderadresse ist die unspezifische Adresse (`::` $\Longrightarrow$ 128 Nullbits)
  - Wenn ein Knoten im lokalen Netz diese IP-Adresse bereits verwendet, handelt es sich um ein Duplikat
    - Der Knoten antwortet mit einer Nachricht Neighbor Advertisement (NA) an die link-lokale Multicast-Adresse `FF02::1` (jeder Knoten im lokalen Netzwerk erhält diese Nachricht)
  - Der Knoten, der die Nachricht Neighbor Solicitation (NS) gesendet hat, muss eine neue Adresse erzeugen und die Duplicate Address Detection erneut durchführen
  - Wenn für einige Zeit keine Nachricht Neighbor Advertisement (NA) empfangen wird, kann die Adresse verwendet werden ($\Longrightarrow$ kein Duplikat)

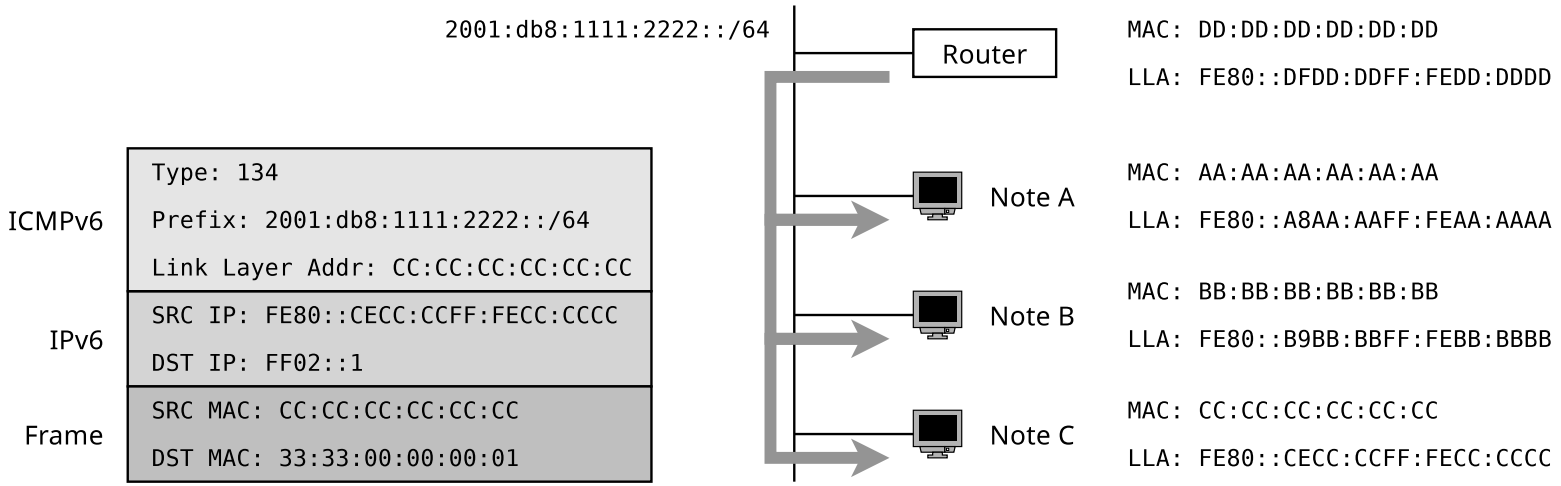
Neighbor Advertisement – NA



Die Nachricht Neighbor Advertisement (NA) ist die IPv6-Alternative zu einer ARP-Antwort bei der Nutzung von IPv4

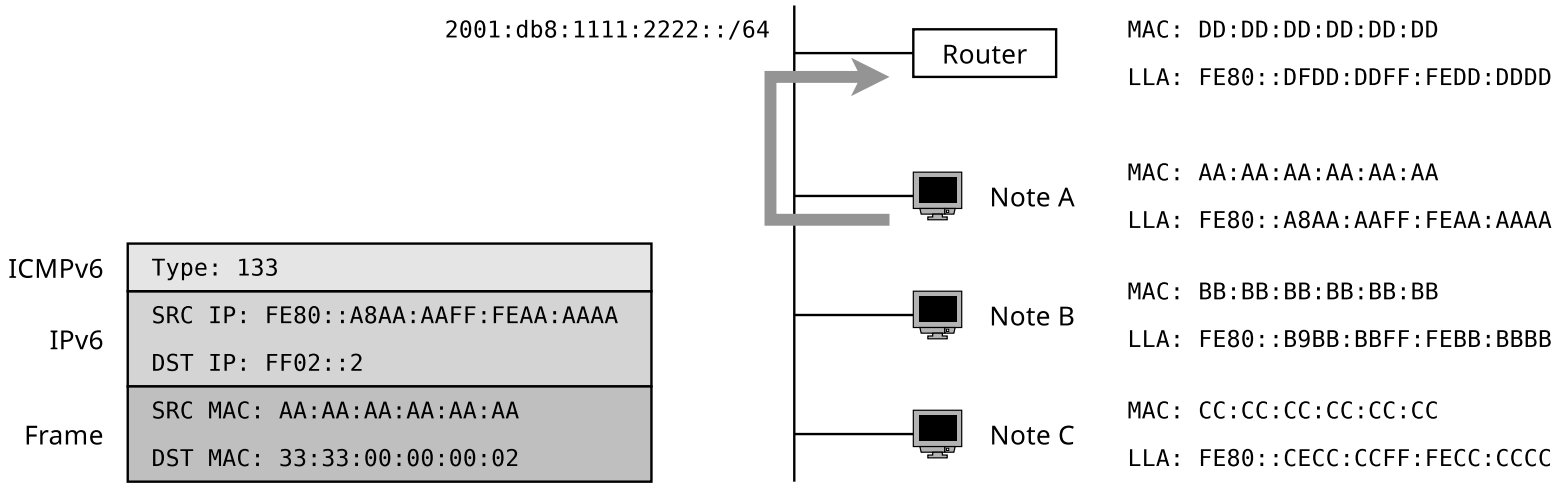
- Antwort auf eine Neighbor Solicitation (NS) Nachricht
- Neighbor Advertisement ist eine Unicast-Nachricht
  - Hier werden keine Multicast-Adressen verwendet

Router Advertisement – RA



- Router senden in regelmäßigen Abständen (die Zeit kann eingestellt werden) RA-Nachrichten in angeschlossene Netze
  - Damit informieren sie über ihre Anwesenheit, das Netzwerk-Präfix, die Präfix-Länge und u.a. die MTU
  - Zieladresse im IPv6-Paket ist die link-lokale Multicast-Adresse `FF02::1` um alle Knoten im lokalen Netz zu erreichen
- Die RA-Nachricht enthält auch das Flag **managed**
  - Ist es gesetzt, soll der Client die Adresse nicht zustandslos selbst festlegen, sondern von einem DHCPv6-Server anfordern (zustandsbehaftet)

Router Solicitation – RS



- Wenn ein Knoten nicht auf eingehende RA-Nachrichten (Router Advertisement) warten möchte, kann er diese anfordern, indem er RS-Nachrichten sendet
  - Zieladresse im IPv6-Paket ist die link-lokale Multicast-Adresse `FF02::2` um alle Router im lokalen Netz zu erreichen
  - In der Abbildung fordert Knoten A von jedem lokalen Router die RS-Nachricht an

SLAAC-Erweiterung: Stable Privacy (RFC 7217) – (1/3)

- **Optionale Erweiterung von SLAAC** (Stateless Address Autoconfiguration)
- Definiert die Adresserzeugung ohne Verwendung einer MAC-Adresse
  - Ein zufälliger geheimer Schlüssel wird erstellt und für die Generierung der Interface-ID verwendet
    - Der geheime Schlüssel ist eine 128-Bit lange hexadezimale Zeichenfolge, die aussieht wie eine IPv6-Adresse

Speicherort des geheimen Schlüssels in Linux und erforderlicher Kernel-Parameter

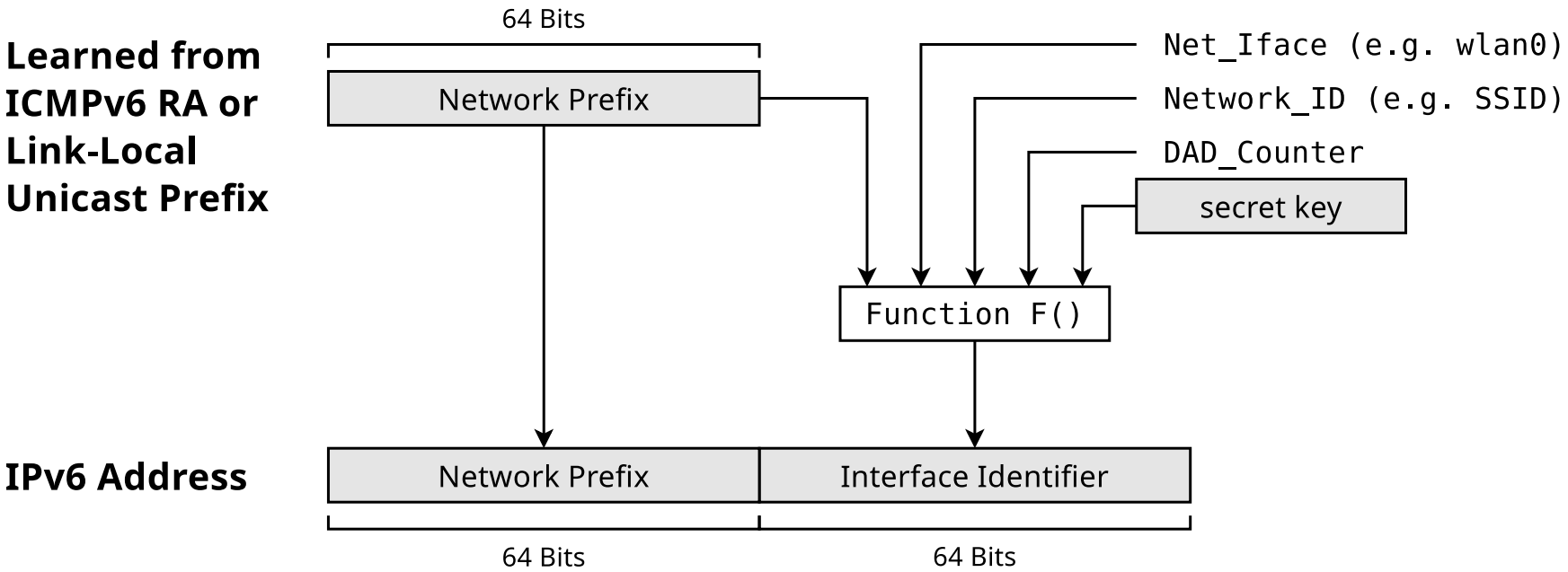
Der stabile geheime Schlüssel ist in der Datei `/proc/sys/net/ipv6/conf/eth0/stable_secret` gespeichert und wird durch Setzen des Kernel-Parameters `addr_gen_mode=3` erzeugt

Beispiel für einen geheimen Schlüssel

```
$ cat /proc/sys/net/ipv6/conf/eth0/stable_secret
c8c8:036d:9312:71e2:eadc:7c9f:0535:649a
```

- Vorteile:
  - Verbesserte Sicherheit, da keine MAC-Adresse für die Erzeugung verwendet wird
    - Die MAC-Adresse des Knoten wird nicht preisgegeben $\Longrightarrow$ Anonymität
  - Stabile Adresse für den Knoten
    - Einmal generiert, ändert sich die Interface-ID nicht (bis zum Neustart)

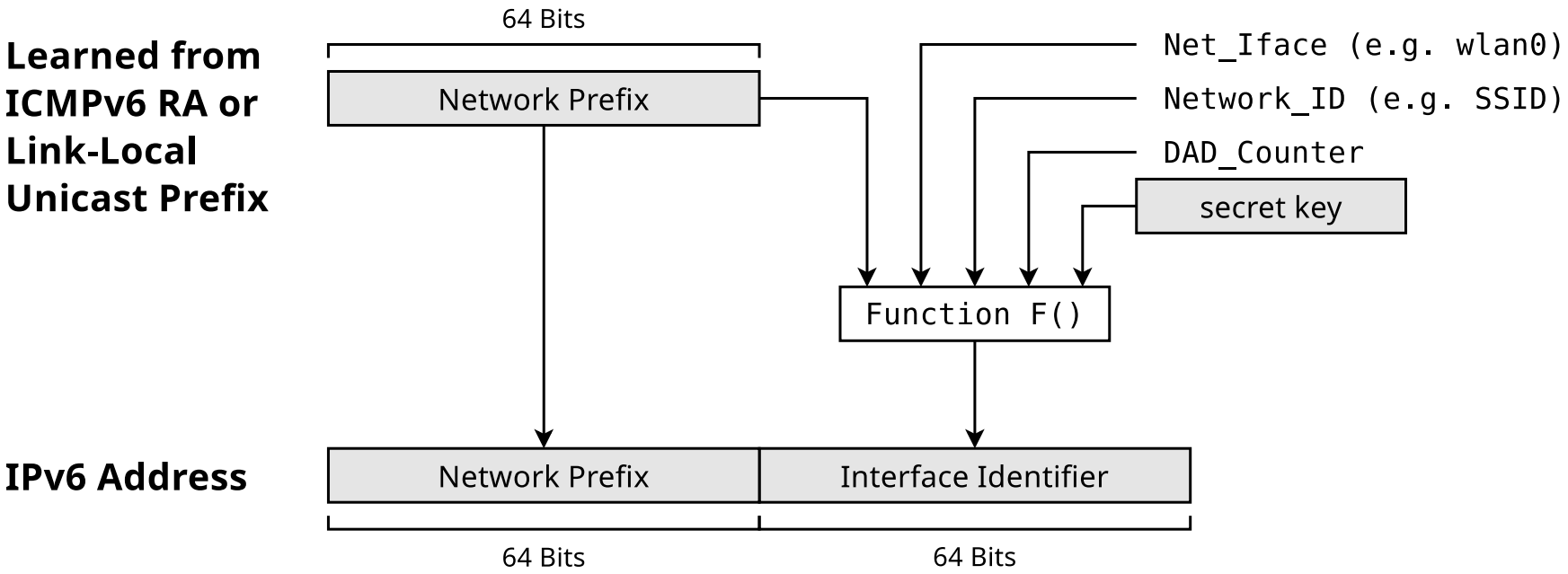
SLAAC-Erweiterung: Stable Privacy (RFC 7217) – (2/3)



- Pseudozufällige Funktion zur Erzeugung der Schnittstellen-ID:
  `F(Prefix, Net_Iface, Network_ID, DAD_Counter, key)`

- `Prefix`: Aus einer ICMPv6 Router Advertisement (RA) Nachricht oder dem Link-Local Unicast Präfix entnommen
- `Net_Iface`: Der Netzwerkschnittstelle zugeordnete ID (z. B. `wlan0`)
- `Network_ID`: Dem Netzwerk zugeordnete ID. z. B. der WLAN Service Set Identifier (SSID)
- `DAD_Counter`: Zum Auflösen von DAD-Konflikten (Duplicate Address Detection). Der Anfangswert ist `0`. Er wird für jede neue Adresse, die konfiguriert wird, um 1 erhöht.
- `key`: 128 Bits langer geheimer Schlüssel

SLAAC-Erweiterung: Stable Privacy (RFC 7217) – (3/3)

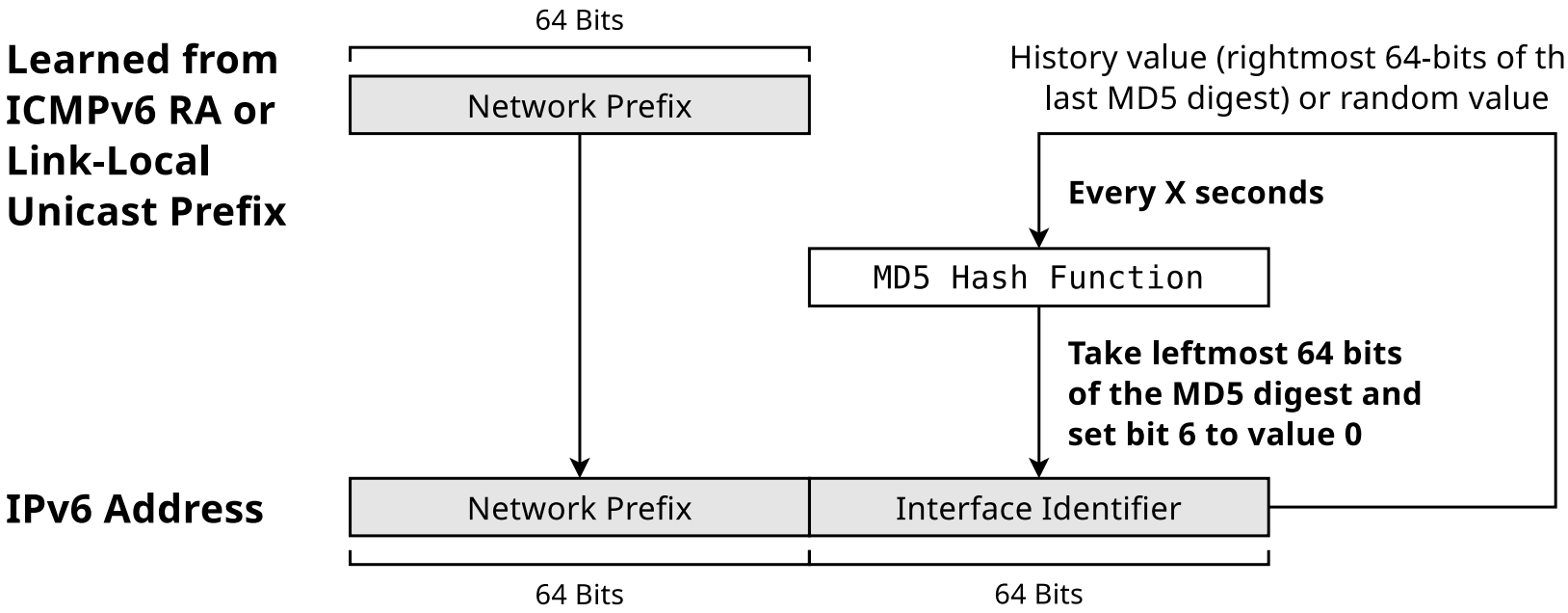


- SHA-1 und SHA-256 sind zwei mögliche Optionen für `F()`
  - Aber nicht `MD5` (siehe RFC 6151)

Beispiel für eine erzeugte Adresse mit Stable Privacy

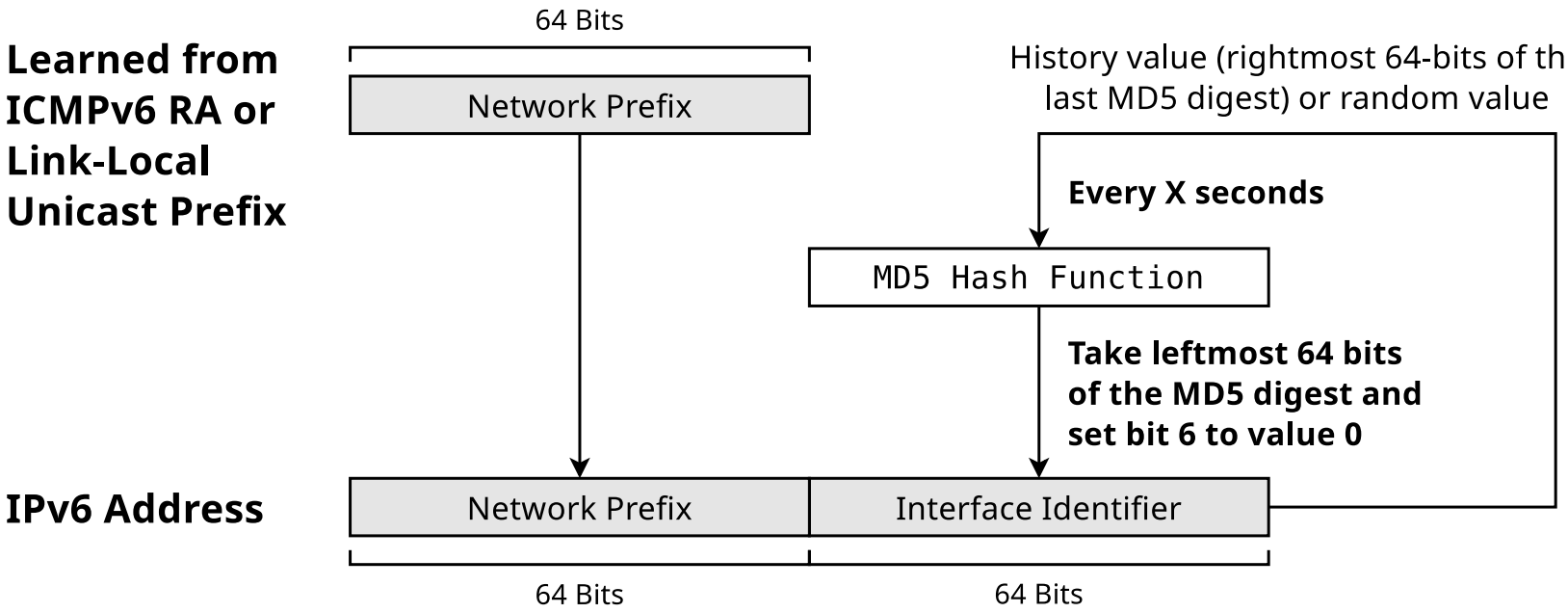
```
MAC: 86:3a:ea:8a:a7:d9
stable-privacy    -> inet6 fe80::6f6d:80e:ab6c:65a0/64
link local EUI-64 -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

SLAAC-Erweiterung: Privacy Extension (RFC 4941) – (1/2)



- **Optionale Erweiterung von SLAAC** (Stateless Address Autoconfiguration)
- Definiert die Adressgenerierung mit einer Zufallszahl
  - Die Interface-ID wird nur vorübergehend verwendet
    - Eine neue Interface-ID wird in regelmäßigen Zeitabständen erzeugt
    - Alte Interface-IDs bleiben für bestehende Verbindungen gültig
  - Vorteil: Es wird keine MAC-Adresse verwendet
    $\Longrightarrow$ Verbesserte Sicherheit + Anonymität
    - Die MAC-Adresse des Knotens wird nicht angezeigt
  - Nachteil: Die Adresse verfällt $\Longrightarrow$ sie ist nicht stabil

SLAAC-Erweiterung: Privacy Extension (RFC 4941) – (2/2)



Beispiel für eine zufällig erzeugte Adresse mit Privacy Extension

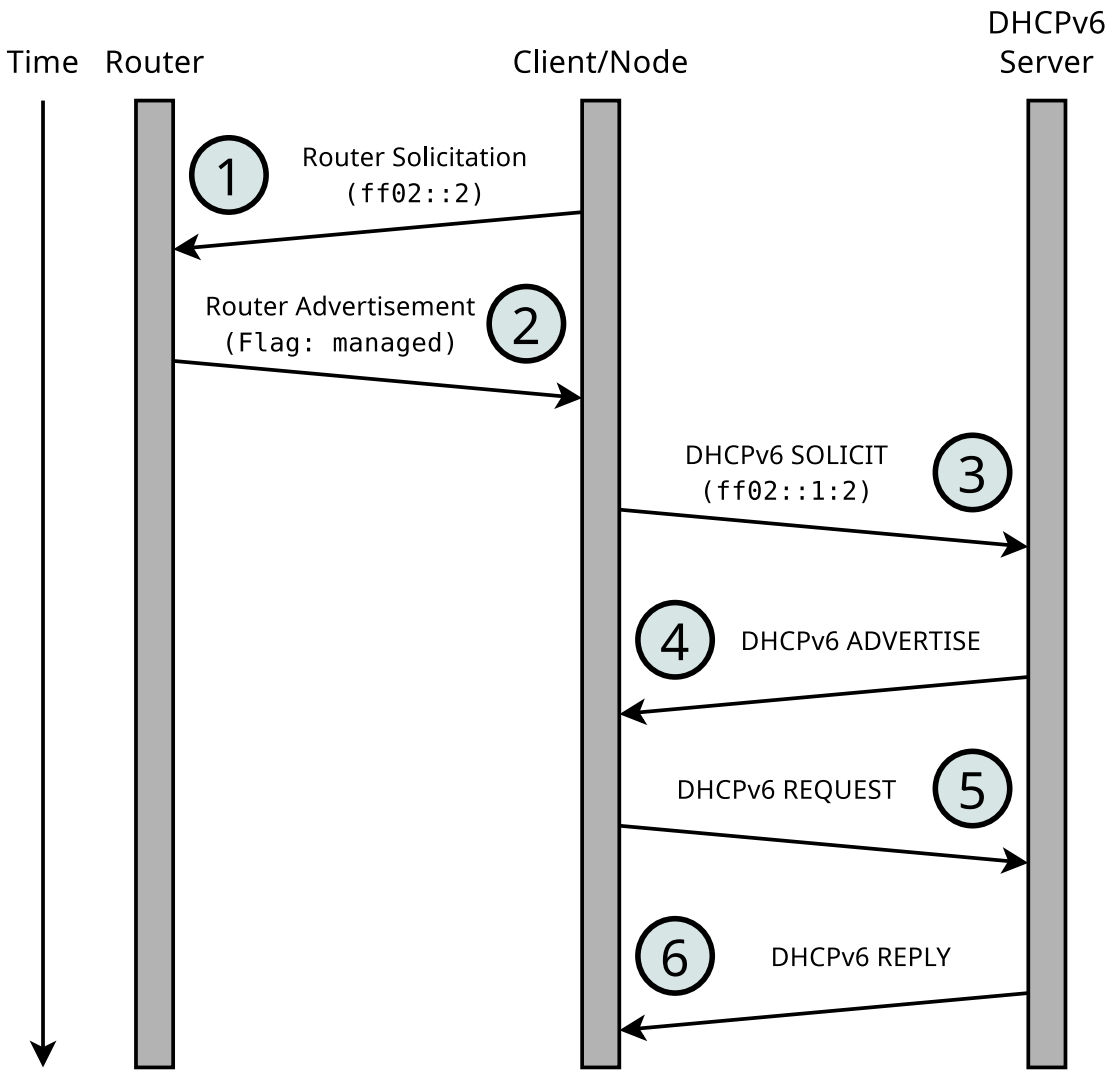
```
MAC: 86:3a:ea:8a:a7:d9
privacy-extension -> inet6 fd12::8992:3c03:d6e2:ed72/64
link local -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

Zufällige erzeugte Interface-ID

Die oben gezeigte Adresse wird zufällig und temporär generiert und kann nicht zu irgendwelchen Merkmalen des Knotens zurückverfolgt werden

DHCPv6 (RFC 8415) – (1/2)

1. Der Knoten fordert mit einer RS-Nachrich an die Multicast-Adresse `ff02::2` (alle Router) ein Präfix für eine global gültige Adresse an
2. Der Router antwortet mit einer RA-Nachricht, in der das Flag `managed` gesetzt ist
3. Der Knoten sendet eine Nachricht `DHCPv6 SOLICIT` an die Multicast-Adresse `ff02::1:2` (alle DHCPv6-Server)
4. Alle DHCPv6-Server in Reichweite antworten mit einer Nachricht `DHCPv6 ADVERTISE`, die eine Netzwerkkonfiguration enthält (DNS-Server, NTP-Server, ein Präfix für die global gültige Adresse,. . .)



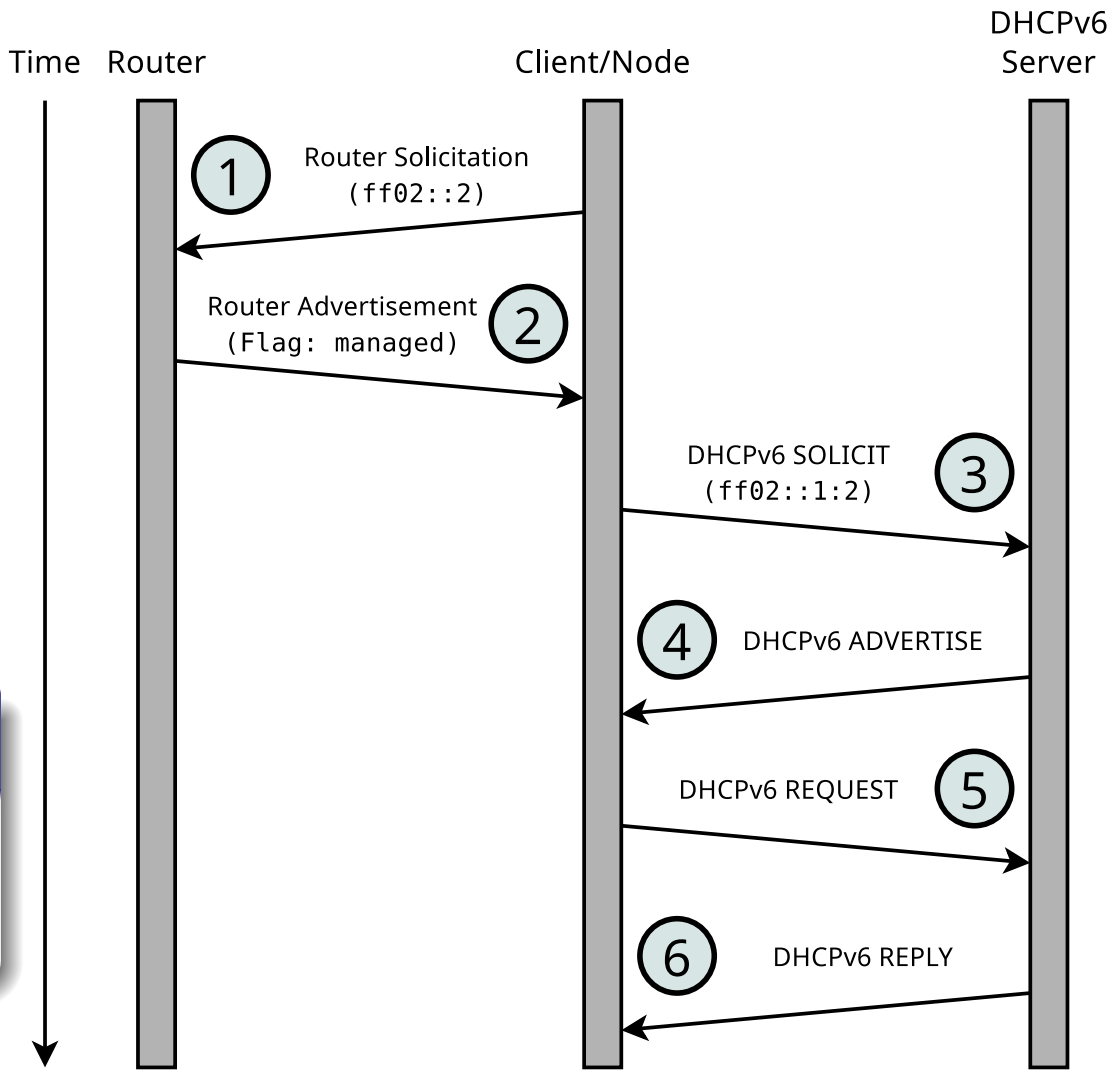
DHCPv6 ist die einzige **zustandsbehaftete** Möglichkeit der IPv6-Adresskonfiguration

DHCPv6 (RFC 8415) – (2/2)

5. Der Knoten wählt ein Konfigurationsangebot aus und fordert es mit einer Nachricht `DHCPv6 REQUEST` an
6. Der DHCPv6-Server markiert die IP in seinem Adresspool mit der Client-ID als zugewiesen und quittiert die Anfrage mit einer Nachricht `DHCPv6 REPLY`

DHCPv6 und DHCP (für IPv4) sind beides Protokolle der Anwendungsschicht

DHCPv6 verwendet UDP über die Ports 547 (Server oder Relay Agent) und 546 (Client)



IPv4-Adressen in IPv6-Netze einbetten (*IPv4 mapped*)

- Eine global geroutete (Unicast) IPv4-Adresse kann als IPv6-Adresse dargestellt und somit in den IPv6-Adressraum integriert werden
  - Diese Vorgehensweise heißt in der Literatur *IPv4 mapped*
- Dafür erhält die IPv4-Adresse einen 96 Bits langen Präfix:
  `0:0:0:0:0:FFFF::/96`

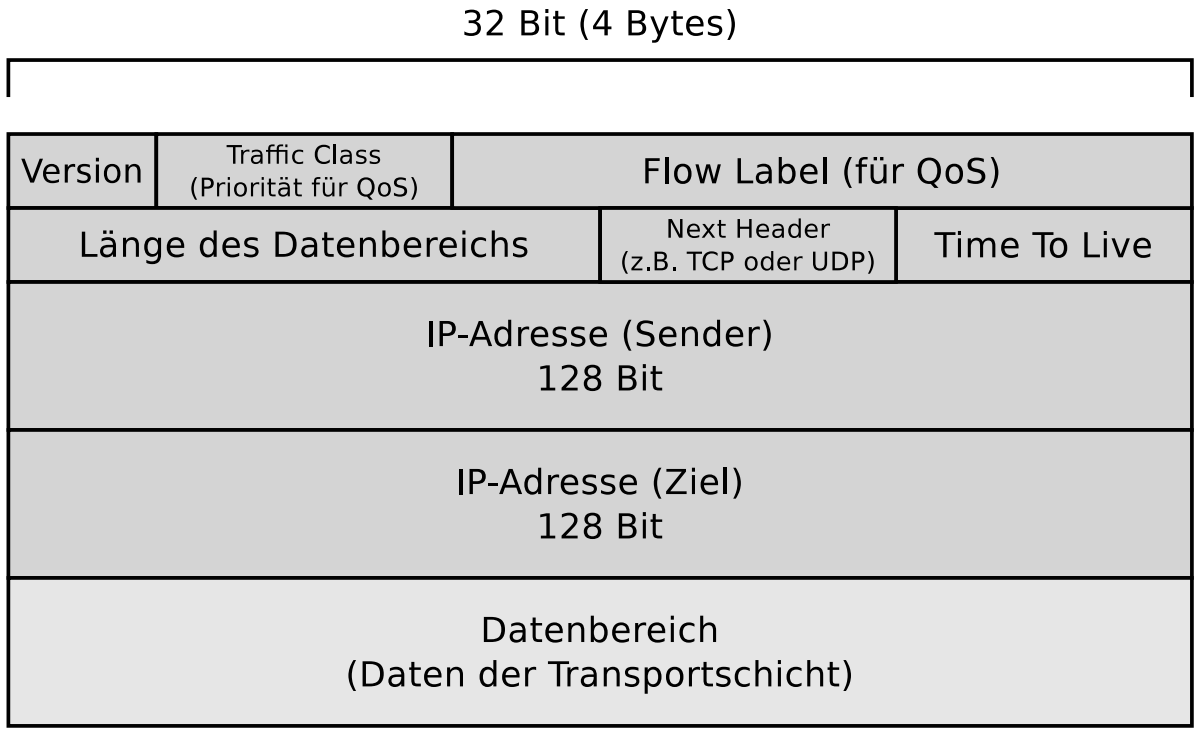
80 Bits					16 Bits	32 Bits
0000	0000	0000	0000	0000	FFFF	IPv4-Adresse

- Die IPv4-Adresse darf in hexadezimaler oder in dezimaler Schreibweise dargestellt sein
- Beispiel

IPv4-Adresse:	`131.246.107.35`
IPv6-Adresse:	`0:0:0:0:0:FFFF:83F6:6B23`
Kurzschreibweisen:	`::FFFF:83F6:6B23`
	`::FFFF:131.246.107.35`

Aufbau von IPv6-Paketen

- Der Header von IPv6-Paketen hat eine feste Länge (320 Bits $\Longrightarrow$ 40 Bytes)



- Im Feld **Next Header** kann auf einen Erweiterungs-Kopfdatenbereich (Extension Header) oder das Protokoll der Transportschicht (z.B. TCP = Typ 6 oder UDP = Typ 17) verwiesen werden

Konzept: Vereinfachte (reduzierte) Paketstruktur und gleichzeitig können zusätzliche (neue) Funktionen durch eine Kette von Erweiterungs-Kopfdaten (*Extension Headers*) hinzugefügt werden

Das Thema Extension Header (siehe RFC 2460 und RFC 4303) wird in dieser Vorlesung nicht behandelt