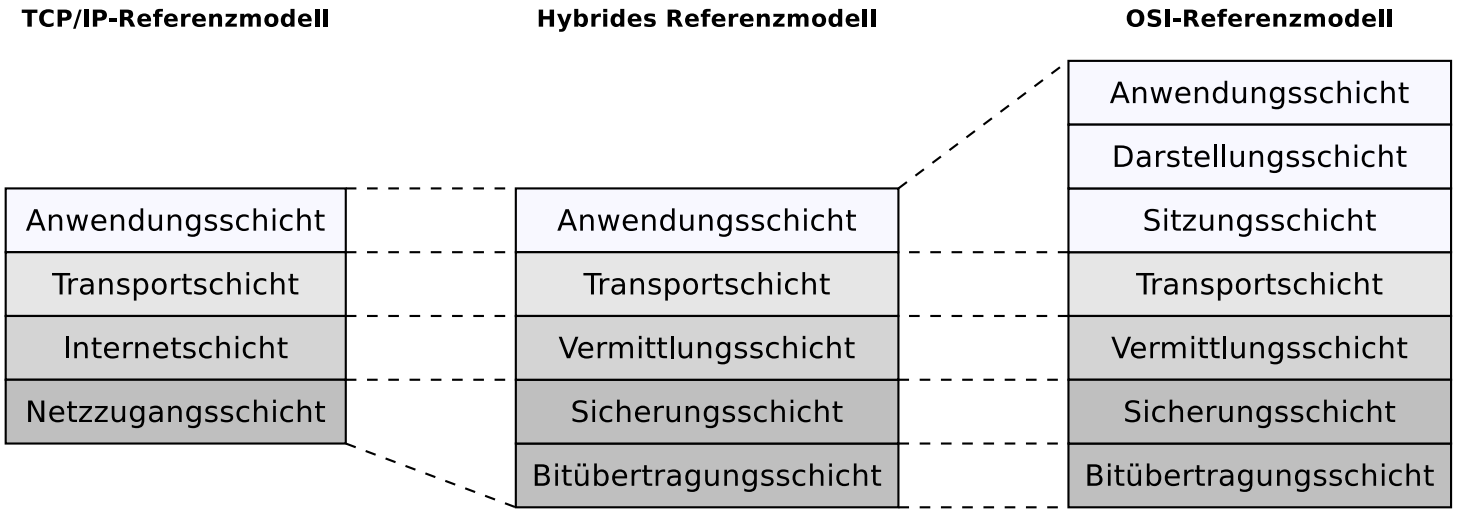
8. Foliensatz Computernetze

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



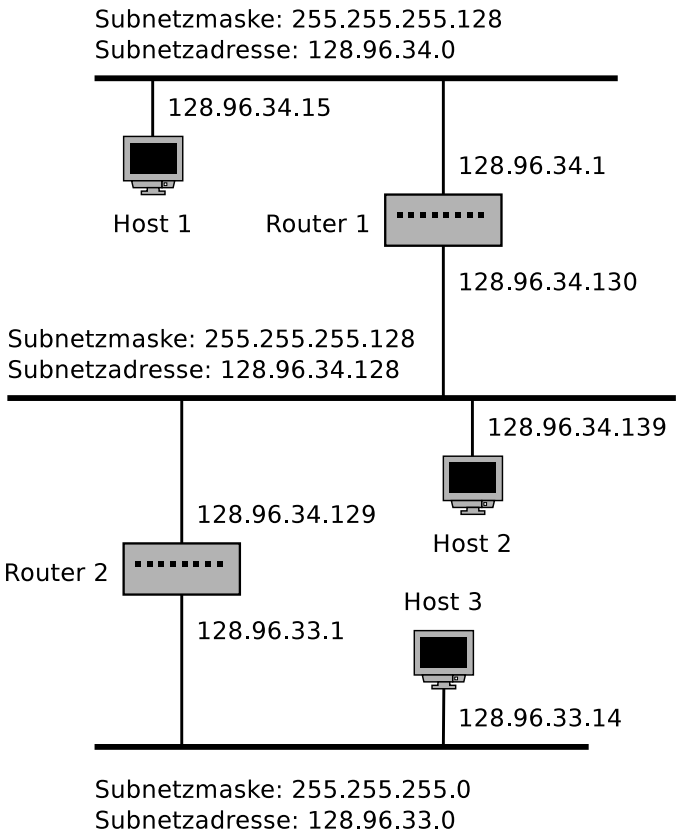
Übungsblatt 4 wiederholt die für die Lernziele relevanten Inhalte dieses Foliensatzes

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

Lernziele dieses Foliensatzes

- Vermittlungsschicht (Teil 2)
  - Weiterleitung und Wegbestimmung
    - Distanzvektor-Routing-Protokolle
    - Link-State-Routing-Protokolle
  - Diagnose und Fehlermeldungen mit ICMP
  - Netzübergreifende Kommunikation $\Longrightarrow$ Internetworking (Zusammenfassung)
  - Network Address Translation (NAT)

Weiterleitung



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

- Primäre Aufgabe der Router: **Weiterleitung (Forwarding)** der IP-Pakete
  - Um diese Aufgabe zu erfüllen, müssen die Router für jedes eintreffende Paket die korrekte Schnittstelle (Port) ermitteln
- Jeder Router verwaltet eine lokale **Routing-Tabelle** (Weiterleitungstabelle)
  - Die Routing-Tabelle enthält...
    - die dem Router bekannten **logischen Netze**
    - die Information, welches logische Netz über welchen **Port** erreichbar ist

Siehe Beispiel zur Adressierung in der Vermittlungsschicht in Foliensatz 7

- Ein Router muss die IP-Pakete also nur in die Richtung versenden, die die Routing-Tabelle vorgibt

Wegbestimmung (Routing)

- Die **Wegbestimmung** (**Routing**) ist der Prozess, bei dem die **Weiterleitungstabellen** mit Hilfe von **Routing-Protokollen** erstellt werden
  - Die Weiterleitungstabellen sind nötig, damit der die Bestimmung des besten Weges, also zu den niedrigsten Kosten, zum Ziel möglich ist
  - Diese Routing-Protokolle werden zwischen den Routern ausgeführt
  - Routing-Protokolle basieren auf **verteilten Algorithmen**
    - Grund: Skalierbarkeit
- 2 Hauptklassen von Routing-Protokollen existieren:
  - **Distanzvektor-Routing-Protokolle** (verwenden den Bellman-Ford-Algorithmus)
    - Beispiel: **Routing Information Protocol** (RIP)
  - **Link-State-Routing-Protokolle** (verwenden den Dijkstra-Algorithmus)
    - Beispiel: **Open Shortest Path First** (OSPF)

Das Border Gateway Protocol (BGP) implementiert Pfad-Vektor-Routing

Pfad-Vektor-Routing hat Ähnlichkeiten mit Distanzvektor-Routing „BGP ist eine Form von Distanzvektor-Protokoll“
Quelle: Computernetzwerke. *Andrew S. Tanenbaum, David J. Wetherall*. 5. Auflage. Pearson (2012). P.548

Einsatzbereiche – Autonome Systeme (AS)

- Router sind in **autonomen Systemen** (AS) organisiert
  - Jedes AS besteht aus einer Gruppe von logischen Netzen, die...
    - das Internet Protocol verwenden
    - von der gleichen Organisation (z.B. einem Internet Service Provider, einem Unternehmen oder einer Universität) betrieben und verwaltet werden
    - das gleiche Routing-Protokoll verwenden
  - Die miteinander verbundenen AS bilden in ihrer Gesamtheit das Internet
- Jedes AS hat eine eindeutige **Autonomous System Number** (ASN)
  - Die Verwaltung der ASNs übernimmt die Internet Assigned Numbers Authority (IANA)
  - Die Verteilung der ASNs übernehmen die Regional Internet Registries
    - Für Europa: RIPE NCC: `http://www.ripe.net`

Eine ASN kann ein 16-Bit-Integer-Wert (alter Standard) oder ein 32-Bit-Integer-Wert (neuer Standard sein)

Karte des Internets – Stand: 2011

Quelle: Rajan Sodhi. PEER 1 Hosting Blog. 1. März 2011

`http://www.peer1.com/blog/2011/03/map-of-the-internet-2011`

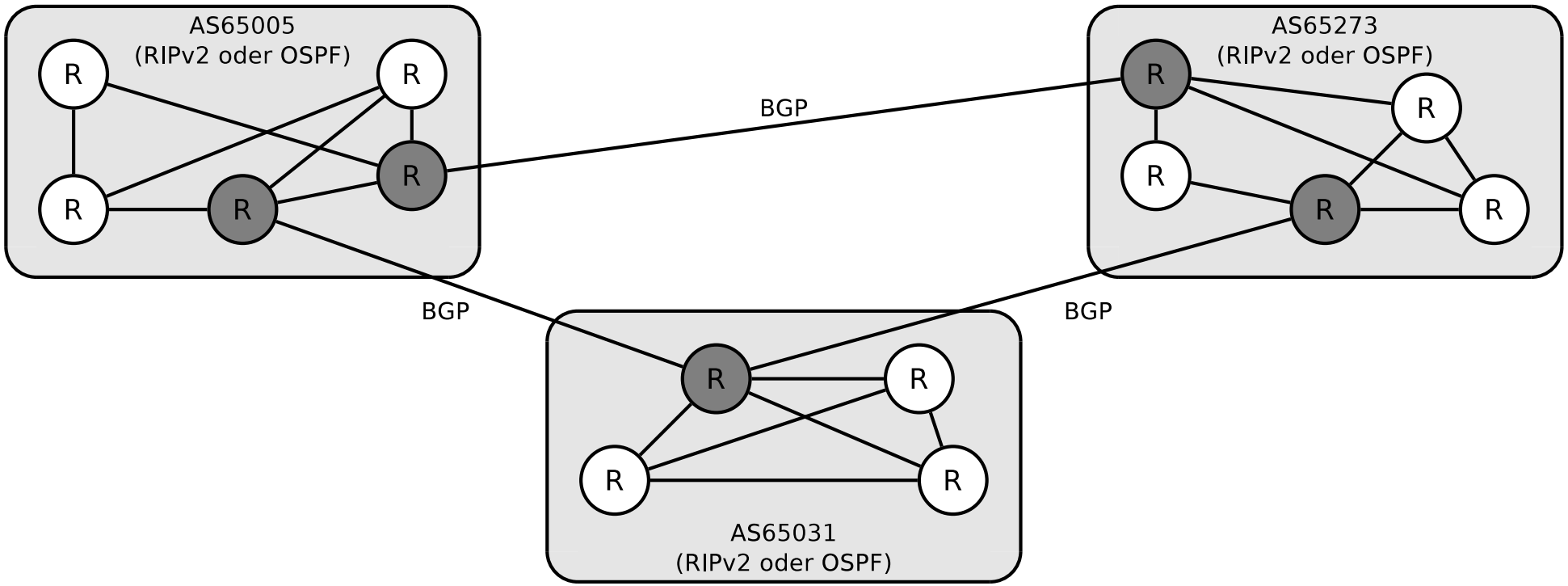
- Zeigt die Verbindungen zwischen den Autonomen Systemen
- Interaktive Version ist verfügbar

Erklärung des Autors

„Each autonomous system is a network operated by a single organization, and has routing connections to some number of neighboring autonomous systems. The image depicts a graph of 19,869 autonomous system nodes, joined by 44,344 connections. The sizing and layout of the autonomous systems are based on their eigenvector centrality, which is a measure of how central to the network each autonomous system is: an autonomous system is central if it is connected to other autonomous systems that are central. This is the same graph-theoretical concept that forms the basis of Google's PageRank algorithm.“

Intra-AS-Routing und Inter-AS-Routing

- Für das Routing innerhalb AS, das sogenannte **Intra-AS-Routing**, sind die Betreiber der AS selbst verantwortlich
  - Geeignete Protokolle für Intra-AS-Routing sind u.a. RIP und OSPF
- Für das Routing zwischen den AS, das sogenannte **Inter-AS-Routing**, wird das BGP verwendet



Konvergenz und Konvergenzzeit

- **Konvergenz**
  - Dieser Zustand ist erreicht, wenn nach einer Änderung der Netzwerk-Topologie, alle Router wieder eine einheitliche Sicht auf das Netzwerk haben
  - Ab diesem Zeitpunkt sind die Einträge in den lokalen Routing-Tabellen der Router dahingehend angepasst, dass die Änderung der Topologie berücksichtigt ist
- **Konvergenzzeit**
  - Zeitspanne, die ein Routing-Protokoll benötigt, um nach einer Änderung der Topologie die Einträge in den lokalen Routing-Tabellen anzupassen

Distanzvektor-Routing-Protokolle (1/3)

- Verwenden den *Bellman-Ford-Algorithmus*
- Beispiel: **Routing Information Protocol** (RIP)
  - Ermöglicht Routing innerhalb autonomer Systeme (**Intra-AS-Routing**)
- Arbeitsweise von RIP:
  - Jeder Router sendet während der Initialisierung über alle seine Ports via Broadcast eine RIP-Anfrage (**RIP Request**)
    - Der neue Router fordert damit alle benachbarten (erreichbaren) Router auf, ihre Routing-Tabellen zu übermitteln
  - Mit den Routing-Informationen der eintreffenden RIP-Antworten (**RIP Response**) füllen der Router seine lokale Routing-Tabelle mit Einträgen

Quelle: Ethernet. Jörg Rech. Heise (2008)

- RIPng (*RIP next generation*) unterstützt IPv6

RFC 2080 (1997)

Distanzvektor-Routing-Protokolle (2/3)

- Arbeitsweise von RIP (Fortsetzung):
  - Alle 30 Sekunden sendet jeder Router seine Routing-Tabelle, die in diesem Kontext auch **Kostenvektor** heißt, über das verbindungslose Transportprotokoll UDP an seine direkten Nachbarn
    - Diese regelmäßige Nachricht heißt **Advertisement**
  - Empfängt ein Router einen Kostenvektor, überprüft er, ob Einträge darin besser sind, als die bislang in der Routing-Tabelle gespeicherten
    - Enthält der empfangene Vektor günstigere Wege, aktualisiert der Router die entsprechenden Einträge in seiner lokalen Routing-Tabelle

Protokoll-Overhead von RIP

- RIP verursacht weniger Protokoll-Overhead als andere Routing-Protokolle wie z.B. OSPF
- Bei Verwendung von RIP wird das Netzwerk nicht mit Routing-Informationen geflutet
- Nachteilig dabei ist, das die Konvergenzzeit länger ist

Distanzvektor-Routing-Protokolle (3/3)

- Arbeitsweise von RIP (Fortsetzung):
  - Zusätzlich zur periodischen Aktualisierungsnachricht sendet ein Router seinen Kostenvektor an die direkten Nachbarn, wenn er eine Metrik in seiner Routing-Tabelle verändert hat ($\Longrightarrow$ **Triggered Updates**)
  - Die Wegkosten zum Zielnetz hängen bei IP ausschließlich von der Anzahl der Router ab, die auf dem Weg passiert werden müssen
    - Die Anzahl der Router wird in **Hops** angegeben
  - Jeder Router erhöht den Wert der Hops um 1
  - Bei RIP kennt jeder Router nur den Inhalt seiner eigenen Routing-Tabelle
    - Die einzelnen Router haben keinen Überblick über das vollständige Netzwerk

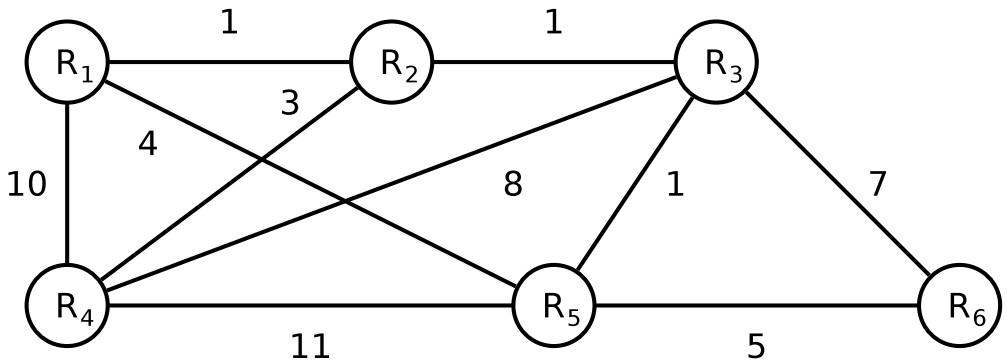
- Weil kein Router einen Überblick über das komplette Netzwerk hat, implementiert das Protokoll einen **verteilten Algorithmus**
  - Nur so erreicht man eine gute Skalierbarkeit

Distanzvektor-Routing-Protokoll – Beispiel (1/5)

Ziel	Hop	Metrik
R_1	R_1	0
R_2	?	∞
R_3	?	∞
R_4	?	∞
R_5	?	∞
R_6	?	∞

Ziel	Hop	Metrik
R_1	?	∞
R_2	R_2	0
R_3	?	∞
R_4	?	∞
R_5	?	∞
R_6	?	∞

Ziel	Hop	Metrik
R_1	?	∞
R_2	?	∞
R_3	R_3	0
R_4	?	∞
R_5	?	∞
R_6	?	∞



Ziel	Hop	Metrik
R_1	?	∞
R_2	?	∞
R_3	?	∞
R_4	R_4	0
R_5	?	∞
R_6	?	∞

Ziel	Hop	Metrik
R_1	?	∞
R_2	?	∞
R_3	?	∞
R_4	?	∞
R_5	R_5	0
R_6	?	∞

Ziel	Hop	Metrik
R_1	?	∞
R_2	?	∞
R_3	?	∞
R_4	?	∞
R_5	?	∞
R_6	R_6	0

- Initialisierung der Tabellen mit $\text{Hop}_{ij} \longleftarrow$? und $\text{Metrik}_{ij} \longleftarrow \infty$ für $i \neq j$ sowie $\text{Hop}_{ij} \longleftarrow R_i$ und $\text{Metrik}_{ij} \longleftarrow 0$ für $i = j$
- Für jeden direkten Nachbarn R_j von R_i wird eingetragen: $\text{Hop}_{ij} \longleftarrow R_j$ und $\text{Metrik}_{ij} \longleftarrow \text{Distanz}(R_i, R_j)$
- Die Distanz wird auf den Wert 1 gesetzt, wenn als Metrik der Hopcount verwendet wird
- Jeder direkte Nachbar R_j von R_i sendet seine Routing-Tabelle an R_i
- Für einen Tabelleneintrag zu R_k wird überprüft, ob $\text{Metrik}_{ij} + \text{Metrik}_{jk} < \text{Metrik}_{ik}$
- Wenn das gilt, erfolgen diese Zuweisungen: $\text{Hop}_{ik} \longleftarrow R_j$ und $\text{Metrik}_{ik} \longleftarrow \text{Metrik}_{ij} + \text{Metrik}_{jk}$

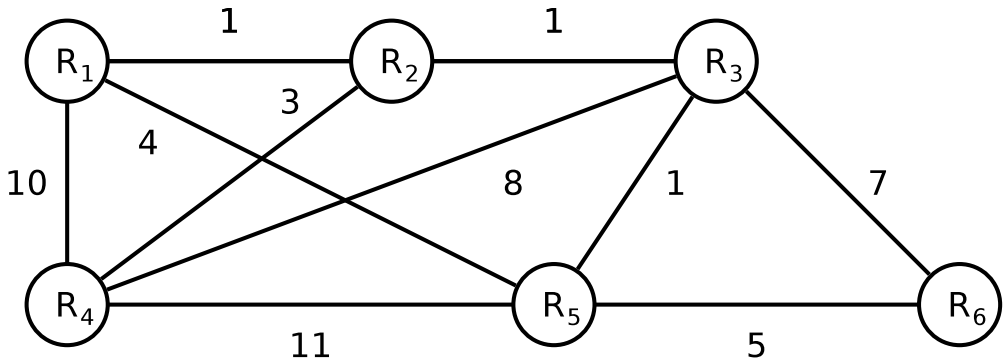
Quelle: Jörg Roth. **Prüfungstrainer Rechnernetze: Aufgaben und Lösungen**. Vieweg (2010)

Distanzvektor-Routing-Protokoll – Beispiel (2/5)

Ziel	Hop	Metrik
R_1	R_1	0
R_2	R_2	1
R_3	?	∞
R_4	R_4	10
R_5	R_5	4
R_6	?	∞

Ziel	Hop	Metrik
R_1	R_1	1
R_2	R_2	0
R_3	R_3	1
R_4	R_4	3
R_5	?	∞
R_6	?	∞

Ziel	Hop	Metrik
R_1	?	∞
R_2	R_2	1
R_3	R_3	0
R_4	R_4	8
R_5	R_5	1
R_6	R_6	7



Ziel	Hop	Metrik
R_1	R_1	10
R_2	R_2	3
R_3	R_3	8
R_4	R_4	0
R_5	R_5	11
R_6	?	∞

Ziel	Hop	Metrik
R_1	R_1	4
R_2	?	∞
R_3	R_3	1
R_4	R_4	11
R_5	R_5	0
R_6	R_6	5

Ziel	Hop	Metrik
R_1	?	∞
R_2	?	∞
R_3	R_3	7
R_4	?	∞
R_5	R_5	5
R_6	R_6	0

- Distanzen zu den direkten Nachbarn eingetragen

Das Beispiel hat einen großen Nachteil

- Das Beispiel zeigt sehr gut den Verlauf des Bellman-Ford-Algorithmus
- Das Beispiel ist aber komplexer als die Realität bei IP
- Die Wegkosten zum Zielnetz hängen bei IP ausschließlich von der Anzahl der Router ab, die auf dem Weg passiert werden müssen
- Wegkosten für bestimmte Verbindungen, wie im Beispiel gezeigt, gibt es bei IP nicht

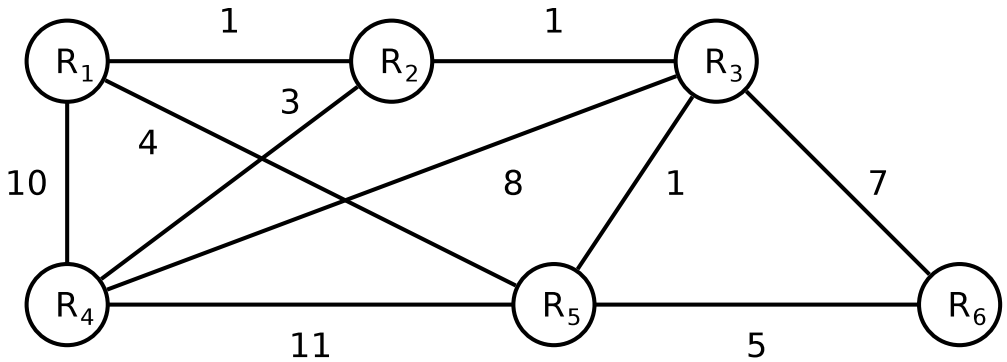
Quelle: Jörg Roth. **Prüfungstrainer Rechnernetze: Aufgaben und Lösungen**. Vieweg (2010)

Distanzvektor-Routing-Protokoll – Beispiel (3/5)

Ziel	Hop	Metrik
R_1	R_1	0
R_2	R_2	1
R_3	R_2	2
R_4	R_2	4
R_5	R_5	4
R_6	R_5	9

Ziel	Hop	Metrik
R_1	R_1	1
R_2	R_2	0
R_3	R_3	1
R_4	R_4	3
R_5	R_3	2
R_6	R_3	8

Ziel	Hop	Metrik
R_1	R_2	2
R_2	R_2	1
R_3	R_3	0
R_4	R_2	4
R_5	R_5	1
R_6	R_5	6



Ziel	Hop	Metrik
R_1	R_2	4
R_2	R_2	3
R_3	R_2	4
R_4	R_4	0
R_5	R_3	9
R_6	R_3	15

Ziel	Hop	Metrik
R_1	R_1	4
R_2	R_3	2
R_3	R_3	1
R_4	R_3	9
R_5	R_5	0
R_6	R_6	5

Ziel	Hop	Metrik
R_1	R_5	9
R_2	R_3	8
R_3	R_5	6
R_4	R_3	15
R_5	R_5	5
R_6	R_6	0

- Jeden Eintrag in den Routing-Tabelle mit den Tabellen der direkten Nachbarn vergleichen und gegebenenfalls anpassen

Das Beispiel hat einen großen Nachteil

- Das Beispiel zeigt sehr gut den Verlauf des Bellman-Ford-Algorithmus
- Das Beispiel ist aber komplexer als die Realität bei IP
- Die Wegkosten zum Zielnetz hängen bei IP ausschließlich von der Anzahl der Router ab, die auf dem Weg passiert werden müssen
- Wegkosten für bestimmte Verbindungen, wie im Beispiel gezeigt, gibt es bei IP nicht

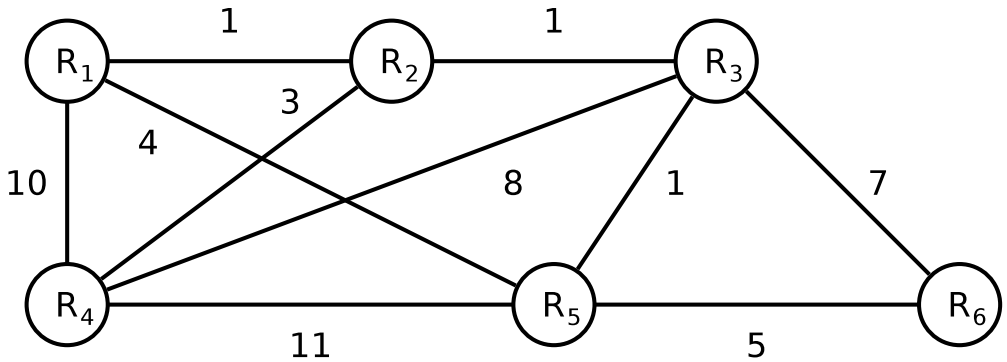
Quelle: Jörg Roth. **Prüfungstrainer Rechnernetze: Aufgaben und Lösungen**. Vieweg (2010)

Distanzvektor-Routing-Protokoll – Beispiel (4/5)

Ziel	Hop	Metrik
R_1	R_1	0
R_2	R_2	1
R_3	R_2	2
R_4	R_2	4
R_5	R_2	3
R_6	R_5	9

Ziel	Hop	Metrik
R_1	R_1	1
R_2	R_2	0
R_3	R_3	1
R_4	R_4	3
R_5	R_3	2
R_6	R_3	7

Ziel	Hop	Metrik
R_1	R_1	2
R_2	R_2	1
R_3	R_3	0
R_4	R_2	4
R_5	R_5	1
R_6	R_5	6



Ziel	Hop	Metrik
R_1	R_2	4
R_2	R_2	3
R_3	R_2	4
R_4	R_4	0
R_5	R_2	5
R_6	R_2	11

Ziel	Hop	Metrik
R_1	R_3	3
R_2	R_3	2
R_3	R_3	1
R_4	R_3	5
R_5	R_5	0
R_6	R_6	5

Ziel	Hop	Metrik
R_1	R_5	9
R_2	R_5	7
R_3	R_5	6
R_4	R_2	11
R_5	R_5	5
R_6	R_6	0

- Jeden Eintrag in den Routing-Tabelle mit den Tabellen der direkten Nachbarn vergleichen und gegebenenfalls anpassen

Das Beispiel hat einen großen Nachteil

- Das Beispiel zeigt sehr gut den Verlauf des Bellman-Ford-Algorithmus
- Das Beispiel ist aber komplexer als die Realität bei IP
- Die Wegkosten zum Zielnetz hängen bei IP ausschließlich von der Anzahl der Router ab, die auf dem Weg passiert werden müssen
- Wegkosten für bestimmte Verbindungen, wie im Beispiel gezeigt, gibt es bei IP nicht

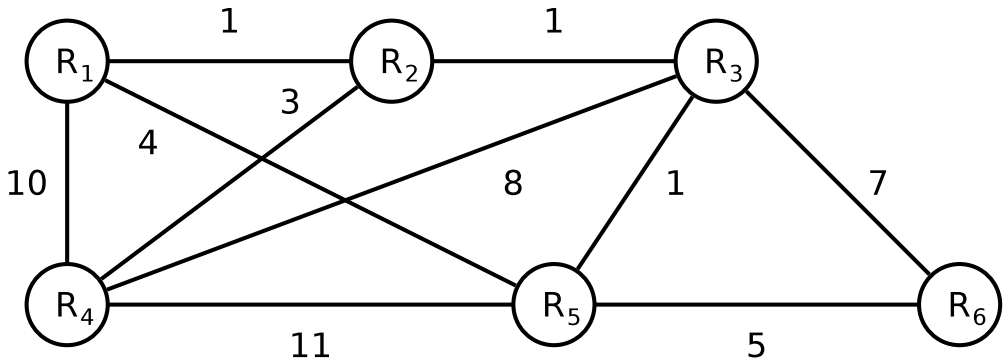
Quelle: Jörg Roth. **Prüfungstrainer Rechnernetze: Aufgaben und Lösungen**. Vieweg (2010)

Distanzvektor-Routing-Protokoll – Beispiel (5/5)

Ziel	Hop	Metrik
R_1	R_1	0
R_2	R_2	1
R_3	R_2	2
R_4	R_2	4
R_5	R_2	3
R_6	R_2	8

Ziel	Hop	Metrik
R_1	R_1	1
R_2	R_2	0
R_3	R_3	1
R_4	R_4	3
R_5	R_3	2
R_6	R_3	7

Ziel	Hop	Metrik
R_1	R_1	2
R_2	R_2	1
R_3	R_3	0
R_4	R_2	4
R_5	R_5	1
R_6	R_5	6



Ziel	Hop	Metrik
R_1	R_2	4
R_2	R_2	3
R_3	R_2	4
R_4	R_4	0
R_5	R_2	5
R_6	R_2	10

Ziel	Hop	Metrik
R_1	R_3	3
R_2	R_3	2
R_3	R_3	1
R_4	R_3	5
R_5	R_5	0
R_6	R_6	5

Ziel	Hop	Metrik
R_1	R_5	8
R_2	R_5	7
R_3	R_5	6
R_4	R_5	10
R_5	R_5	5
R_6	R_6	0

- Jeden Eintrag in den Routing-Tabelle mit den Tabellen der direkten Nachbarn vergleichen und gegebenenfalls anpassen

Das Beispiel hat einen großen Nachteil

- Das Beispiel zeigt sehr gut den Verlauf des Bellman-Ford-Algorithmus
- Das Beispiel ist aber komplexer als die Realität bei IP
- Die Wegkosten zum Zielnetz hängen bei IP ausschließlich von der Anzahl der Router ab, die auf dem Weg passiert werden müssen
- Wegkosten für bestimmte Verbindungen, wie im Beispiel gezeigt, gibt es bei IP nicht

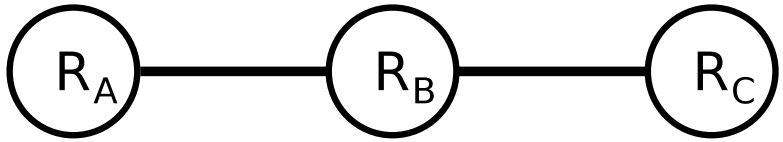
Quelle: Jörg Roth. **Prüfungstrainer Rechnernetze: Aufgaben und Lösungen**. Vieweg (2010)

Maximale Metrik

- RIP hat eine maximale Metrik
- Die **Metrik** (= **Kosten**) sind der Aufwand, um ein Netz zu erreichen
- Beim Protokoll IP wird dazu **ausschließlich der Hopcount** verwendet
  - Dieser bezeichnet die Anzahl der Router, die entlang eines Pfades bis zum Zielnetz durchlaufen werden müssen
- Die Unerreichbarkeit eines Netzwerks gibt RIP mit Hopcount 16 ($= \infty$ Kosten) an
  - RIP erlaubt also nur Computernetze mit einer maximalen Länge von 15 Routern

Count-to-Infinity Problem (1/2)

- Nachteil des Algorithmus von RIP:
  - **Schlechte Nachrichten verbreiten sich nur langsam**
- Beispiel:



- Nach und nach verbreiten sich die Entfernungswerte zu R_A
  - Die Tabelle enthält die gespeicherte Entfernung zu Router R_A in den Routing-Tabellen von R_A, R_B und R_C

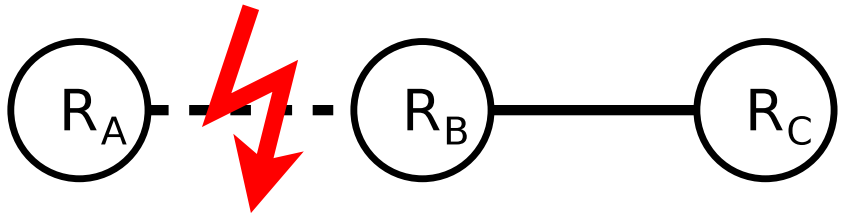
A	**B**	**C**	
0	∞	∞	Initialer Eintrag
0	1	∞	Nach Advertisement-Runde 1
0	1	2	Nach Advertisement-Runde 2
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Count-to-Infinity Problem (2/2)

A	B	C	
0	1	2	Initialer Eintrag
0	3	2	Nach Advertisement-Runde 1
0	3	4	Nach Advertisement-Runde 2
0	5	4	Nach Advertisement-Runde 3
0	5	6	Nach Advertisement-Runde 4
0	7	6	Nach Advertisement-Runde 5
0	7	8	Nach Advertisement-Runde 6
0	9	8	Nach Advertisement-Runde 7
0	9	10	Nach Advertisement-Runde 8
0	11	10	Nach Advertisement-Runde 9
0	11	12	Nach Advertisement-Runde 10
0	13	12	Nach Advertisement-Runde 11
0	13	14	Nach Advertisement-Runde 12
0	15	14	Nach Advertisement-Runde 13
0	15	∞	Nach Advertisement-Runde 14
0	∞	∞	Nach Advertisement-Runde 15

$\Longrightarrow$ **Count-to-Infinity**

- Szenario: Die Verbindung zu R_A fällt aus



- Bei Advertisement-Runde 1 hört R_B nichts mehr von R_A und hält den Pfad über R_C nun am besten um R_A zu erreichen
- Bei Advertisement-Runde 2 erfährt R_C das sein Nachbar R_B eine Route mit der Länge 3 zu R_A hat und passt darum seine lokal gespeicherte Entfernung zu R_A auf 4 an
- ...

Split Horizon

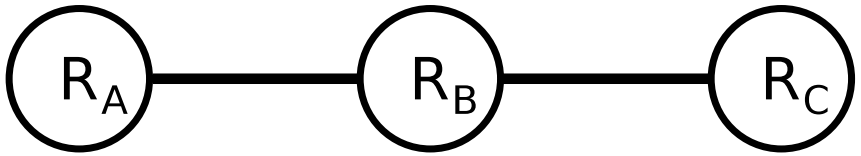
- Wegen des Problems Count-to-Infinity geht viel Zeit verloren, bis die Unerreichbarkeit einer Route erkannt wird

Advertisement-Nachrichten werden alle 30 s ausgetauscht. Ohne Triggered Updates kann es darum bis zu $15 * 30\,\text{s} = 7{:}30$ Minuten dauern, bis ein Netzausfall zwischen 2 Routern erkannt wird und die betroffenen Routen in den Routing-Tabellen als nicht erreichbar markiert sind

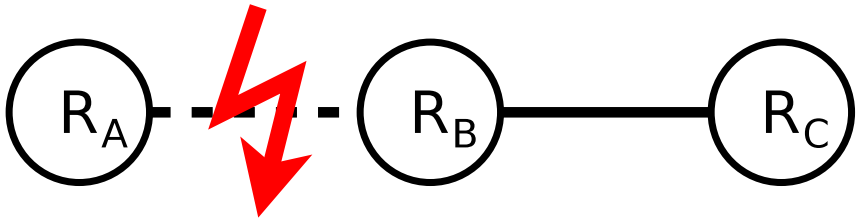
- Lösung **in einigen Anwendungsfällen**: Split Horizon
  - Es verhindert Routing-Schleifen zwischen **2 Routern**
- Eine Routing-Informationen darf nicht über den Port veröffentlicht werden, über den sie empfangen wurde
  - Das verhindert, das ein Router eine Route zurück an den Router übermittelt, von dem er sie gelernt hat
- Um Split Horizon zu ermöglichen, muss in der Routing-Tabelle für jedes Zielnetz nicht nur die Anzahl der Hops und die Adresse des nächsten Routers (nächster Hop) gespeichert werden, sondern auch die Information, von welchem Router die Informationen gelernt wurde

Beispiel zu Split Horizon

- R_c weiß von R_B, das R_A über ihn erreichbar ist



- Szenario: R_A ist nicht zu erreichen

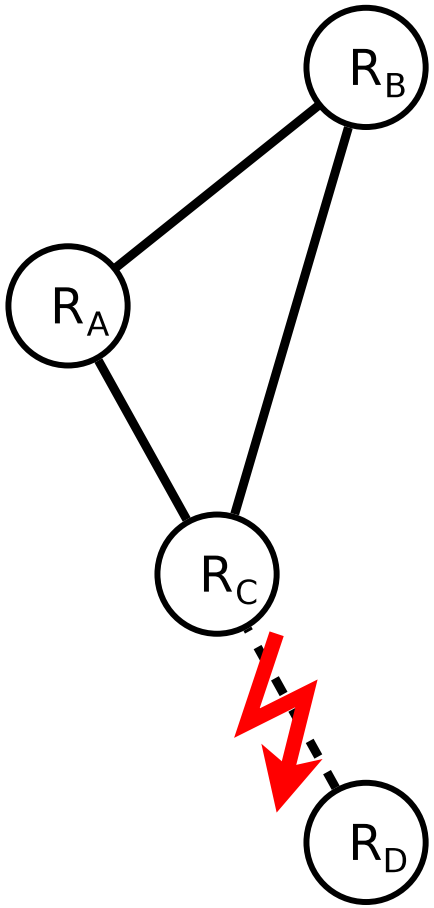


- Auswirkung von Split Horizon:
  - R_B sendet beim nächsten Update an R_C, dass R_A nicht erreichbar ist
  - R_C passt seine Routing-Tabelle an und sendet weder jetzt noch in Zukunft Routing-Informationen für R_A an R_B

Problem: Split Horizon versagt in vielen Fällen

Ein Beispiel wo Split Horizon versagt

- Ist die Verbindung zwischen R_C und R_D gestört, markiert R_C in seiner Routing-Tabelle R_D als nicht erreichbar

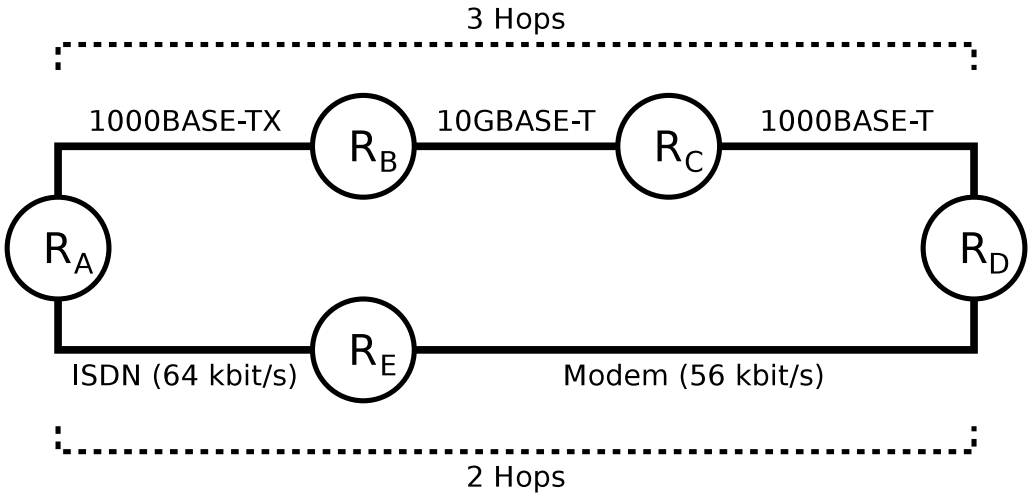


- R_C informiert R_A und R_B das R_D nicht erreichbar ist
- Kommt die Advertisement-Nachricht zuerst bei R_A an, glaubt dieser die beste Route zu R_D ist via R_B
- R_A teilt R_B mit, dass R_D nicht erreichbar ist und informiert R_C das er R_D mit 3 Hops erreicht
- R_C glaubt, dass er R_D via R_A mit 4 Hops erreicht und informiert R_B das er eine Route zu R_D hat
- R_B informiert R_A das er R_D mit 5 Hops erreicht
- R_A informiert R_C das er R_D mit 6 Hops erreicht
- ...

$\Longrightarrow$ **Count-to-Infinity**

Fazit zu RIP

- RIPv1 (RFC 1058) wurde zu einem Zeitpunkt entwickelt und etabliert, als die Computernetze noch relativ klein waren
  - RIPv1 unterstützt nur Netzklassen und keine Subnetze
- Als RIPv1 entwickelt wurde, existierten innerhalb eines Netzes nur selten verschiedene Übertragungsmedien mit deutlichen Unterschieden bzgl. Verbindungsqualität und Übertragungsrate



- Die Metrik Hopcount führt heute häufig zu **Routen, deren Verlauf nicht optimal** ist, weil alle Netzabschnitte **gleich stark gewichtet** werden

- RIPv2 (RFC 2453) unterstützt Subnetze und kann zwischen internen und externen Routen unterscheiden (siehe Folie 26)

Aufbau von RIPv1-Nachrichten (*Advertisements*)

32 Bit (4 Bytes)

Kommando	RIP-Version	00000000	00000000
Adressfamilie (von Netz 1)		00000000	00000000
IP-Adresse (von Netz 1)			
00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
Metrik (Hops bis zu Netz 1)			
Adressfamilie (von Netz 2)		00000000	00000000
IP-Adresse (von Netz 2)			
00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
Metrik (Hops bis zu Netz 2)			
⋮			
Adressfamilie (von Netz 25)		00000000	00000000
IP-Adresse (von Netz 25)			
00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
Metrik (Hops bis zu Netz 25)			

- Enthält das Feld **Kommando** den Wert 1, handelt es sich um eine RIP-Anfrage (*RIP request*)
  - Damit wird ein Router aufgefordert seine Routing-Tabelle zu übermitteln
- Enthält das Feld **Kommando** den Wert 2, handelt es sich um eine RIP-Antwort (*RIP response*)
  - Damit übermittelt ein Router seine Routing-Tabelle
- Bei IP-Netzen hat das Feld **Adressfamilie** den Wert 2

RFC 1058 (1988)

Aufbau von RIPv2-Nachrichten (*Advertisements*)

32 Bit (4 Bytes)

Kommando	RIP-Version	00000000	00000000
Adressfamilie (von Netz 1)		Route Tag	
IP-Adresse (von Netz 1)			
Subnetzmaske (von Netz 1)			
Nächster Hop (bis zu Netz 1)			
Metrik (Hops bis zu Netz 1)			
Adressfamilie (von Netz 2)		Route Tag	
IP-Adresse (von Netz 2)			
Subnetzmaske (von Netz 2)			
Nächster Hop (bis zu Netz 2)			
Metrik (Hops bis zu Netz 2)			
⋮			
Adressfamilie (von Netz 25)		Route Tag	
IP-Adresse (von Netz 25)			
Subnetzmaske (von Netz 25)			
Nächster Hop (bis zu Netz 25)			
Metrik (Hops bis zu Netz 25)			

- Im Feld **Route Tag** ist festgelegt, ob es eine interne oder externe Route ist

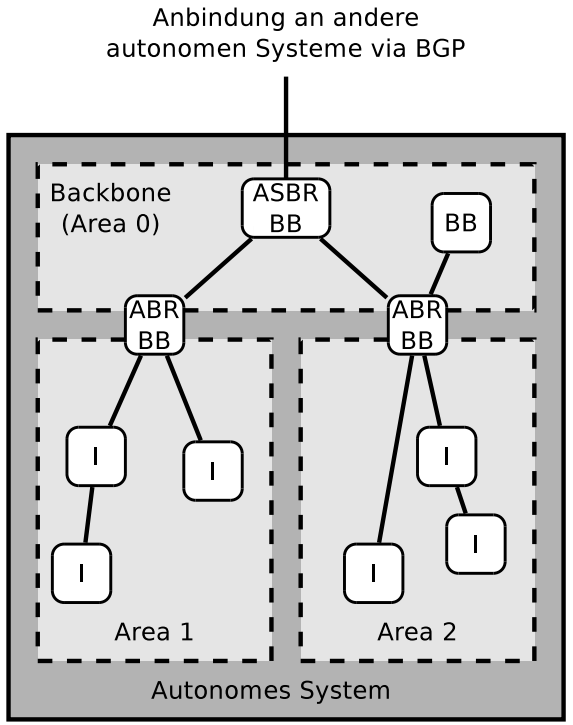
RFC 2453 (1998)

Link-State-Routing-Protokolle

- Verwenden den **Dijkstra-Algorithmus** (**Shortest Path First**)
  - **Berechnet den kürzesten Weg zwischen einem Startknoten und allen anderen Knoten in einem kantengewichteten Graphen**
- Beispiel: **Open Shortest Path First** (OSPF)
  - Ermöglicht Routing innerhalb autonomer Systeme (**Intra-AS-Routing**)
  - OSPF-Nachrichten werden direkt, also ohne ein Protokoll der Transportschicht, im Nutzdatenteil von IPv4-Paketen übertragen
    - Im Header des IPv4-Pakets steht im Datenfeld **Protokoll-ID** der Wert 89
  - Die Arbeitsweise von OSPF ist im Vergleich zu RIP kompliziert
    - Eine detaillierte Beschreibung enthält RFC 2328

Routing-Hierarchie mit OSPF

- Großer Unterschied gegenüber RIP:
  - Mit OSPF können Routing-Hierarchien gebildet werden
- Dafür werden autonome Systeme in Bereiche (**Areas**) unterteilt
  - Jede Area besteht aus einer Gruppe von Routern
  - Jede Area ist für die übrigen Areas des autonomen Systems unsichtbar
  - Jeder Router kann verschiedenen Areas zugeordnet sein
- Ein Vorteil, der sich aus Routing-Hierarchien ergibt:
  - Bessere Skalierbarkeit



ASBR = Autonomous System Boundary Router
ABR = Area Border Router
BB = Backbone Router
I = Internal Router

Gute Quellen zum Thema OSPF

Ethernet, *Jörg Rech*, Heise (2008)
Computernetzwerke, *James F. Kurose, Keith W. Ross*, Pearson (2008)
TCP/IP, *Gerhard Lienemann, Dirk Larisch*, Heise (2011)

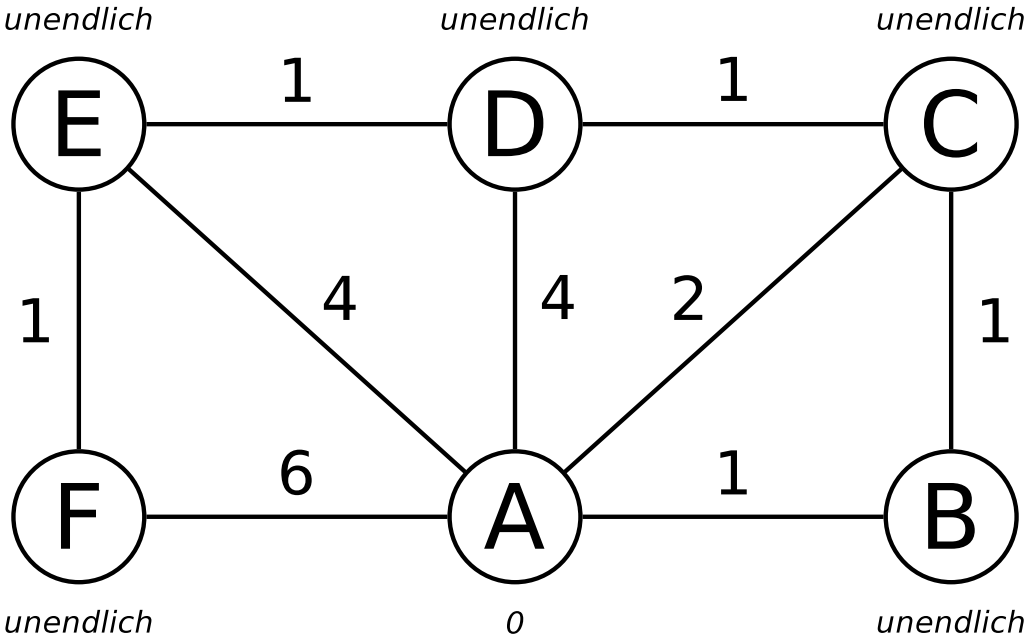
OSPF ist deutlich komplexer als RIP und wird im Rahmen dieser Vorlesung nicht vertieft

Dijkstra-Algorithmus

- Berechnung des kürzesten Weges zwischen einem Startknoten und allen anderen Knoten in einem kantengewichteten Graphen
  - Kantengewichte dürfen nicht negativ sein
- Schritte:
  1. Weise allen Knoten die Eigenschaften **Distanz** und **Vorgänger** zu
     - Initialisiere die Distanz im Startknoten mit 0 und in allen anderen Knoten mit ∞
  2. Solange es noch nicht besuchte Knoten gibt, wähle darunter denjenigen mit minimaler Distanz aus
     - Speichere, dass dieser Knoten schon besucht wurde
     - Berechne für alle noch nicht besuchten Nachbarknoten die Summe der Kantengewichte über den aktuellen Knoten
     - Ist dieser Wert für einen Knoten kleiner als die dort gespeicherte Distanz, aktualisiere diese und setze den aktuellen Knoten als Vorgänger

Ist man nur am Weg zu einem bestimmten Knoten interessiert, kann man in Schritt 2 abbrechen, wenn der gesuchte Knoten der aktive ist

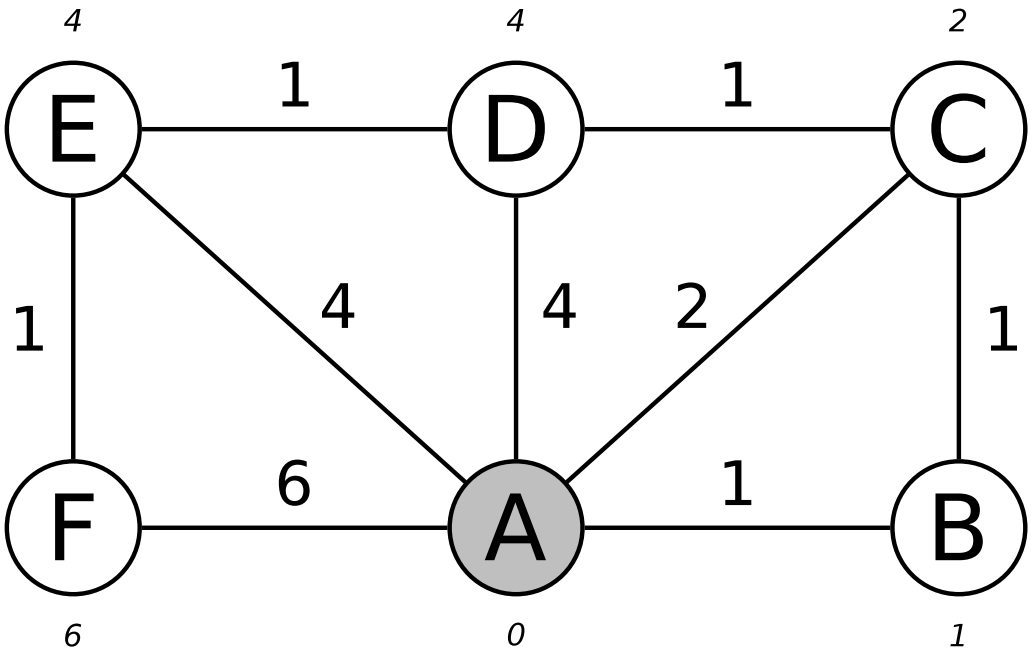
Dijkstra-Algorithmus – Beispiel (1/7)



Distanzwerte	
$d_A = 0$	
$d_B = \infty$	
$d_C = \infty$	
$d_D = \infty$	
$d_E = \infty$	
$d_F = \infty$	

- Schritt 1: Initialisiere mit 0 und ∞
  - Sei A der Startknoten
  - A hat die minimale Distanz
- Besuchte Knoten $= \{\}$
- Kürzeste Pfade $= \{\}$

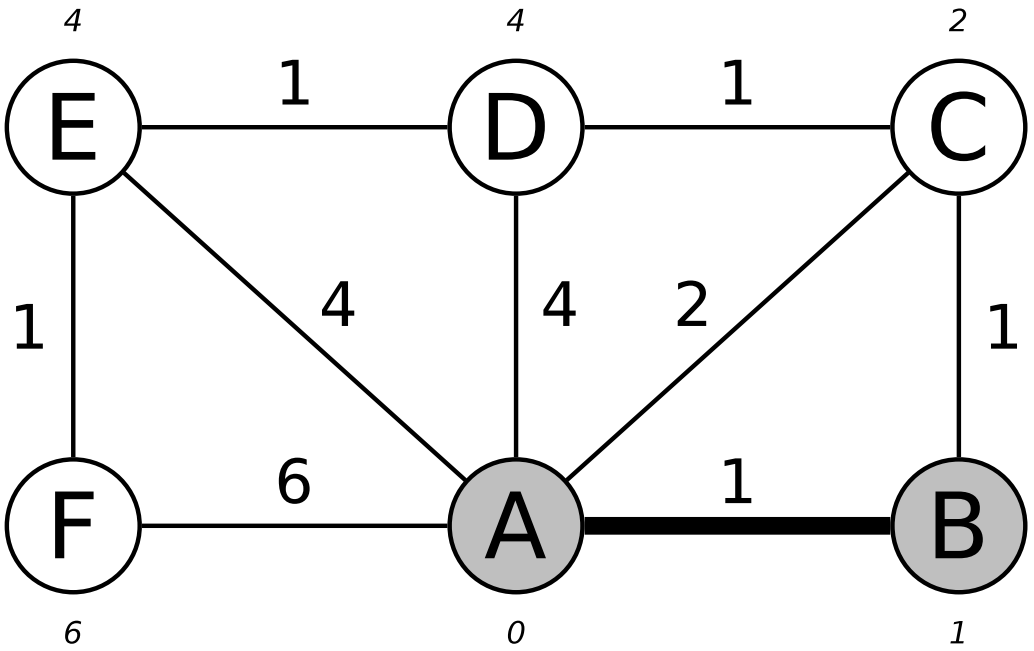
Dijkstra-Algorithmus – Beispiel (2/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	$\longleftarrow$ minimale Distanz
$d_C = 2$	
$d_D = 4$	
$d_E = 4$	
$d_F = 6$	

- Schritt 2: Summe der Kantengewichte berechnen
  - B hat die minimale Distanz
- Besuchte Knoten $= \{A\}$
- Kürzeste Pfade $= \{A\}$

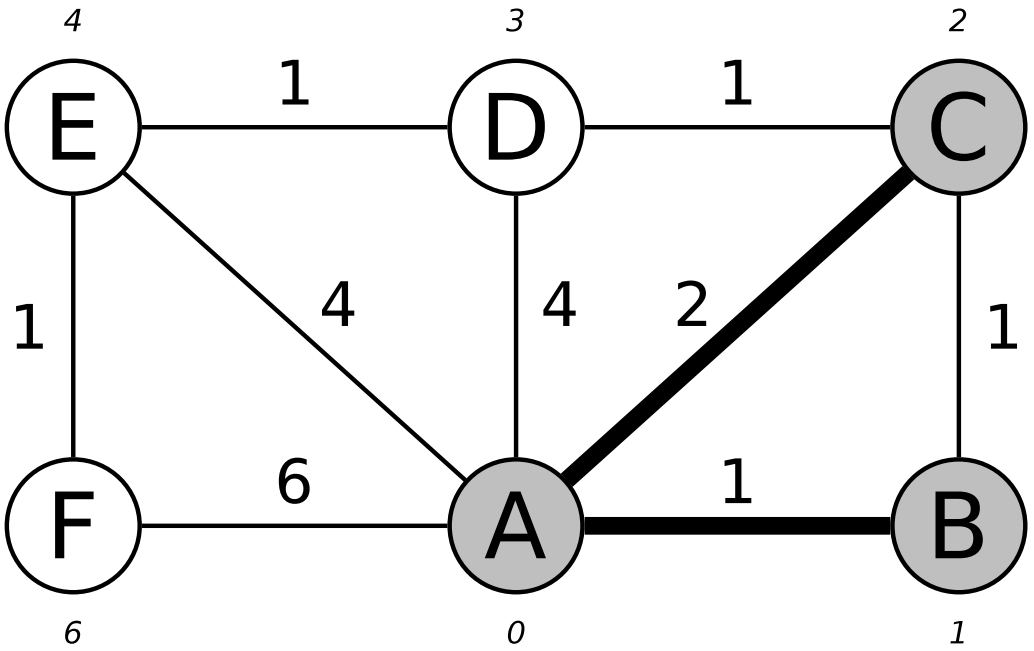
Dijkstra-Algorithmus – Beispiel (3/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	besucht
$d_C = 2$	$\longleftarrow$ minimale Distanz
$d_D = 4$	
$d_E = 4$	
$d_F = 6$	

- Schritt 3: Knoten B besuchen
  - Keine Veränderung zu C
  - C hat die minimale Distanz
- Besuchte Knoten = {A, B}
- Kürzeste Pfade = {A, A$\longrightarrow$B}

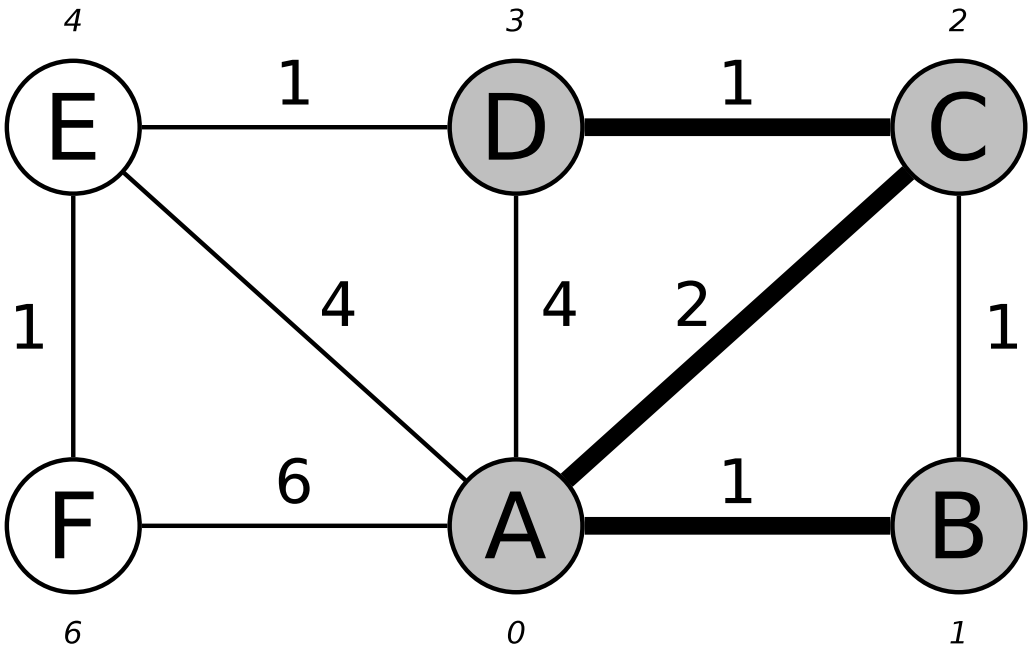
Dijkstra-Algorithmus – Beispiel (4/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	besucht
$d_C = 2$	besucht
$d_D = \mathbf{3}$	$\longleftarrow$ minimale Distanz
$d_E = 4$	
$d_F = 6$	

- Schritt 4: Knoten C besuchen
  - Keine Veränderung zu B
  - Veränderung zu D (Weg über C ist kürzer als der direkte Weg)
  - D hat die minimale Distanz
- Besuchte Knoten = {A, B, C}
- Kürzeste Pfade = {A, A$\longrightarrow$B, A$\longrightarrow$C}

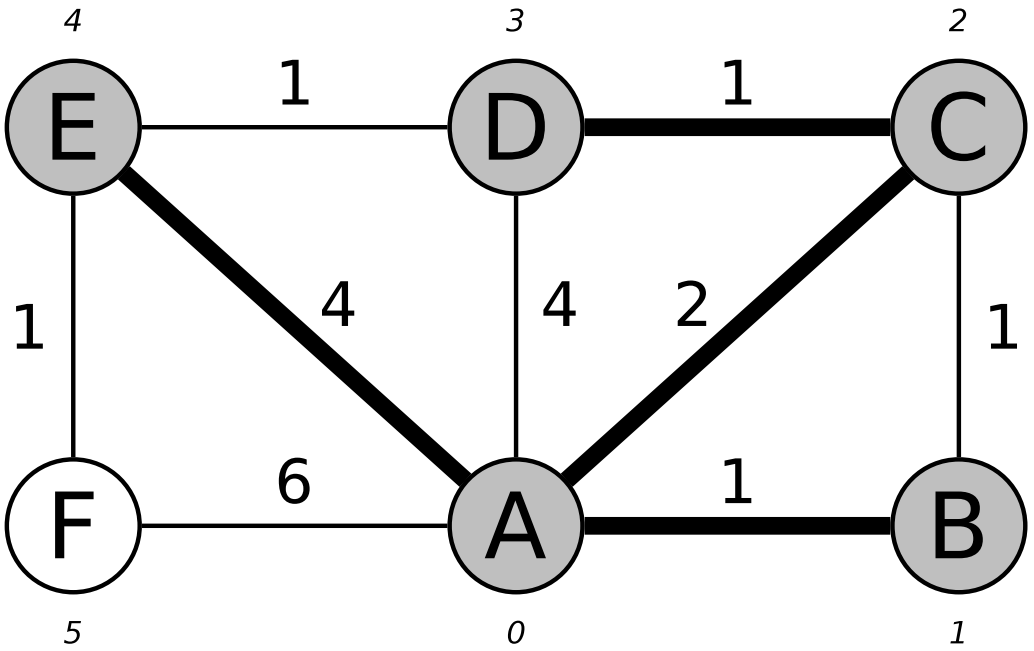
Dijkstra-Algorithmus – Beispiel (5/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	besucht
$d_C = 2$	besucht
$d_D = 3$	besucht
$d_E = 4$	$\longleftarrow$ minimale Distanz
$d_F = 6$	

- Schritt 5: Knoten D besuchen
  - Keine Veränderung zu C
  - Keine Veränderung zu E
  - E hat die minimale Distanz
- Besuchte Knoten = {A, B, C, D}
- Kürzeste Pfade = {A, A$\longrightarrow$B, A$\longrightarrow$C, C$\longrightarrow$D}

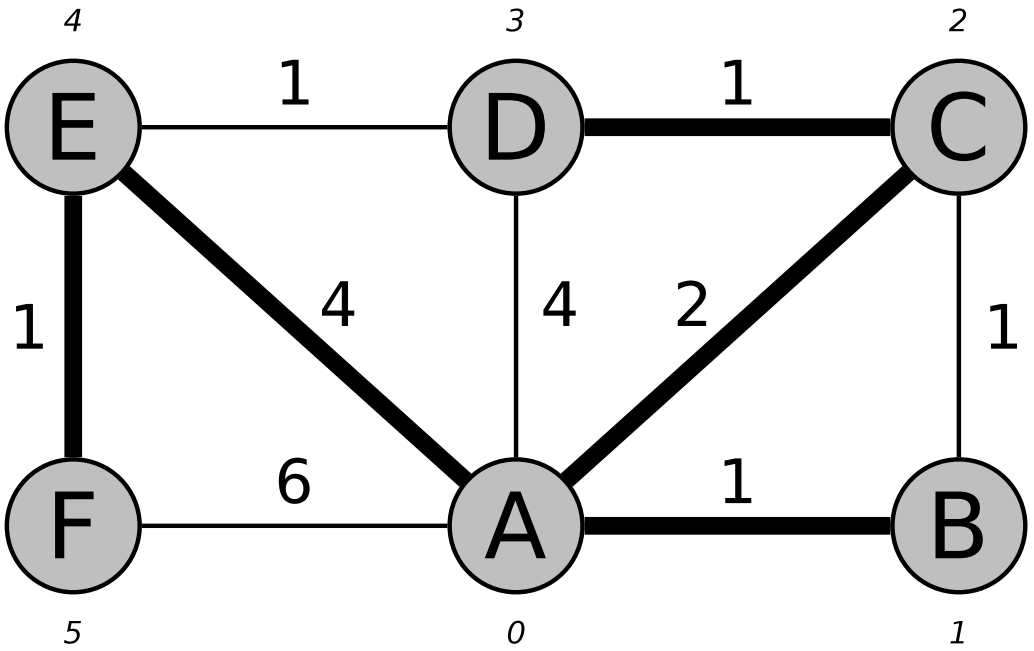
Dijkstra-Algorithmus – Beispiel (6/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	besucht
$d_C = 2$	besucht
$d_D = 3$	besucht
$d_E = 4$	besucht
$d_F = \mathbf{5}$	$\longleftarrow$ minimale Distanz

- Schritt 6: Knoten E besuchen
  - Keine Veränderung zu D
  - Veränderung zu F (Weg über E ist kürzer als der direkte Weg)
  - F hat die minimale Distanz
- Besuchte Knoten = {A, B, C, D, E}
- Kürzeste Pfade = {A, A$\longrightarrow$B, A$\longrightarrow$C, C$\longrightarrow$D, A$\longrightarrow$E}

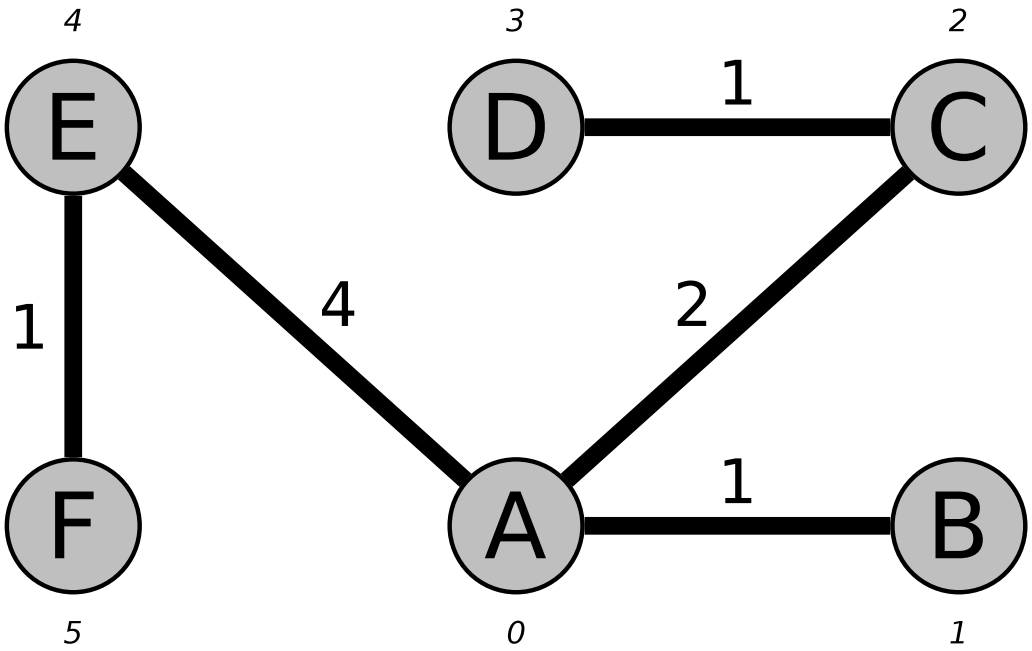
Dijkstra-Algorithmus – Beispiel (7/7)



Distanzwerte	
$d_A = 0$	besucht
$d_B = 1$	besucht
$d_C = 2$	besucht
$d_D = 3$	besucht
$d_E = 4$	besucht
$d_F = 5$	besucht

- Schritt 7: Knoten F besuchen
  - Keine Veränderung zu E
- Besuchte Knoten = {A, B, C, D, E, F}
- Kürzeste Pfade = {A, A$\longrightarrow$B, A$\longrightarrow$C, C$\longrightarrow$D, A$\longrightarrow$E, E$\longrightarrow$F}

Dijkstra-Algorithmus – Beispiel (Ergebnis)



- Ergebnis: Spannbaum mit den kürzesten Pfaden

Distanzvektor-Routing-Protokoll vs. Link-State-Routing-Protokolle

- Distanzvektor-Routing-Protokolle (*Bellman-Ford*)
  - Jeder Router kommuniziert nur mit seinen **direkten** Nachbarn
    - Vorteil: Geringe Belastung für das Netzwerk
    - Nachteil: Langsame Konvergenz, weil sich Aktualisierungen nur langsam *fortpflanzen*
  - **Kein Router hat Kenntnis über die komplette Netzwerk-Topologie**
  - Die Wegkosten (**Metrik**) zum Zielnetz hängen ausschließlich von der Anzahl der Router (**Hops**) ab, die auf dem Weg passiert werden müssen
- Link-State-Routing-Protokolle (*Dijkstra*)
  - **Alle** Router kommunizieren untereinander
    - Vorteil: Schnelle Konvergenz
    - Nachteil: Netzwerk wird geflutet $\Longrightarrow$ hohe Belastung für das Netzwerk
  - Jeder Router baut eine **komplexe Datenbank mit Topologie-Informationen** auf
  - Mit Areas werden **Routing-Hierarchien** gebildet
    - Das verbessert die Skalierbarkeit
  - Netzabschnitte werden mit **Pfadkosten** unterschiedlichen stark gewichtet

Diagnose und Fehlermeldungen mit ICMP

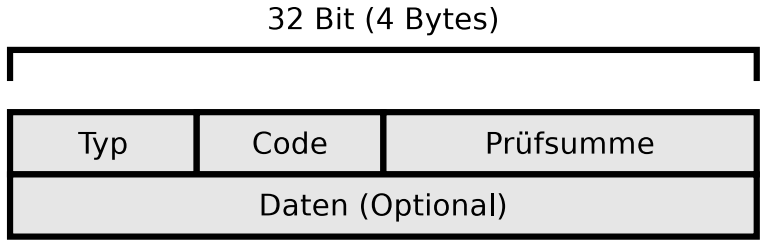
- Das **Internet Control Message Protocol** (ICMP) ermöglicht den Austausch von...
  - Diagnosemeldungen
  - Steuernachrichten
  - Fehlermeldungen
- ICMP ist ein Bestandteil (*Partnerprotokoll*) von IPv4
  - Es wird aber wie ein eigenständiges Protokoll behandelt

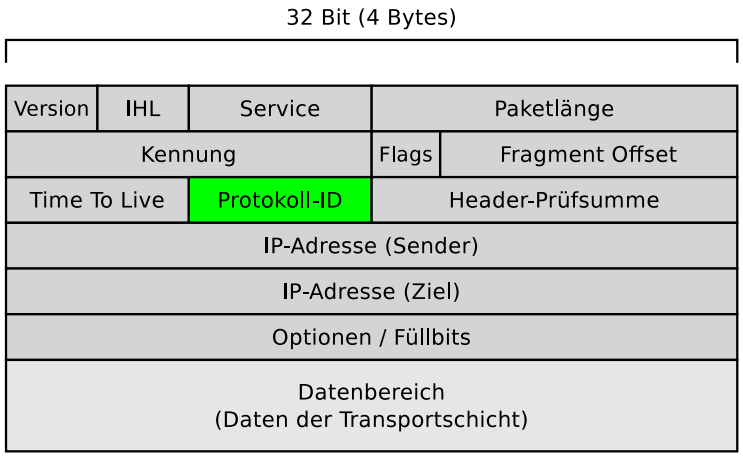
Für IPv6 existiert mit ICMPv6 ein ähnliches Protokoll

- 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

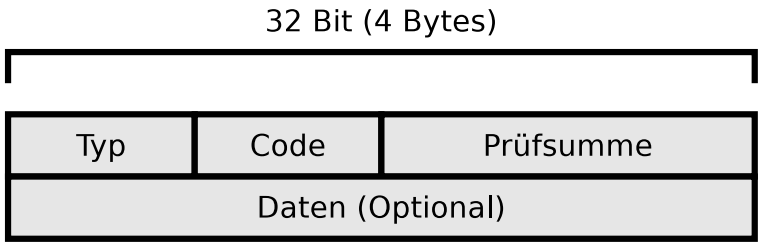
- Eine Anwendung, die ICMP-Pakete versendet, ist das Programm `ping`
- ICMP definiert verschiedene Informationsnachrichten, die ein Router zurücksenden kann





- 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

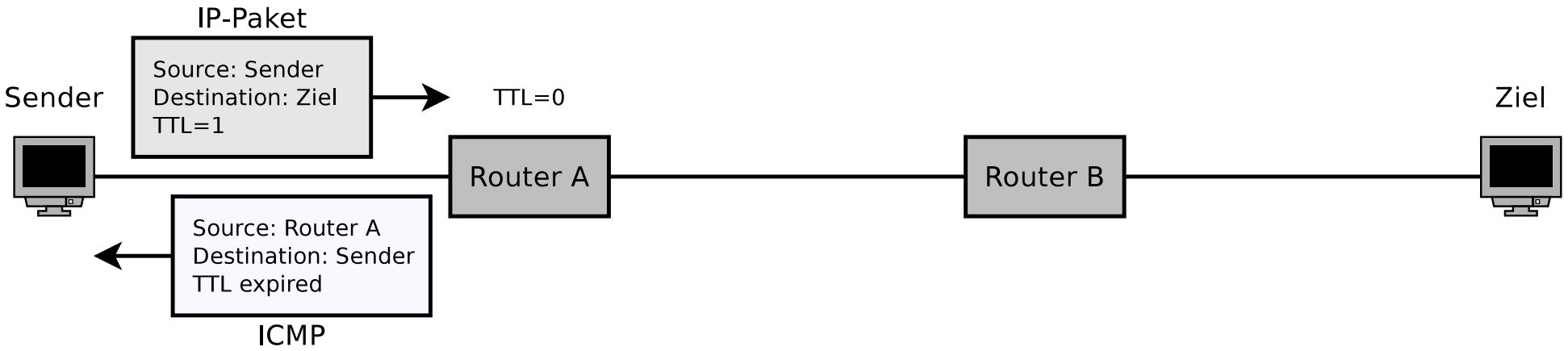
ICMP-Nachrichten



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

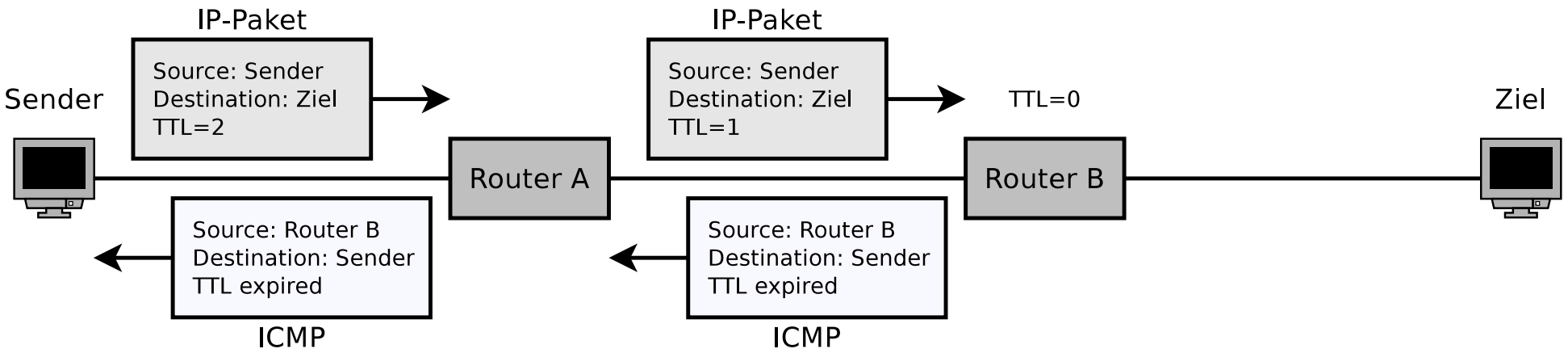
Typ	Typname	Code	Bedeutung
0	Echo-Antwort	0	Echo-Antwort (Antwort auf `ping`)
3	Ziel nicht erreichbar	0	Netz unerreichbar
		1	Ziel unerreichbar
		2	Protokoll nicht verfügbar
		3	Port nicht verfügbar
		4	Fragmentierung nötig, aber im IP-Paket untersagt
		13	Firewall des Ziels blockt IP-Paket
4	Sender verlangsamen	0	Empfangspuffer voll, IP-Paket verworfen (Überlastkontrolle)
8	Echo-Anfrage	0	Echo-Anfrage (`ping`)
11	Zeitlimit überschritten	0	TTL (Time To Live) abgelaufen
17	Address Mask Request	0	Anfrage nach der Anzahl der Bits in der Subnetzmaske
18	Address Mask Reply	0	Antwort auf Nachrichtentyp 17
		1	Zeitlimit während der Defragmentierung überschritten
30	Traceroute	0	Weg zum Ziel ermitteln

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



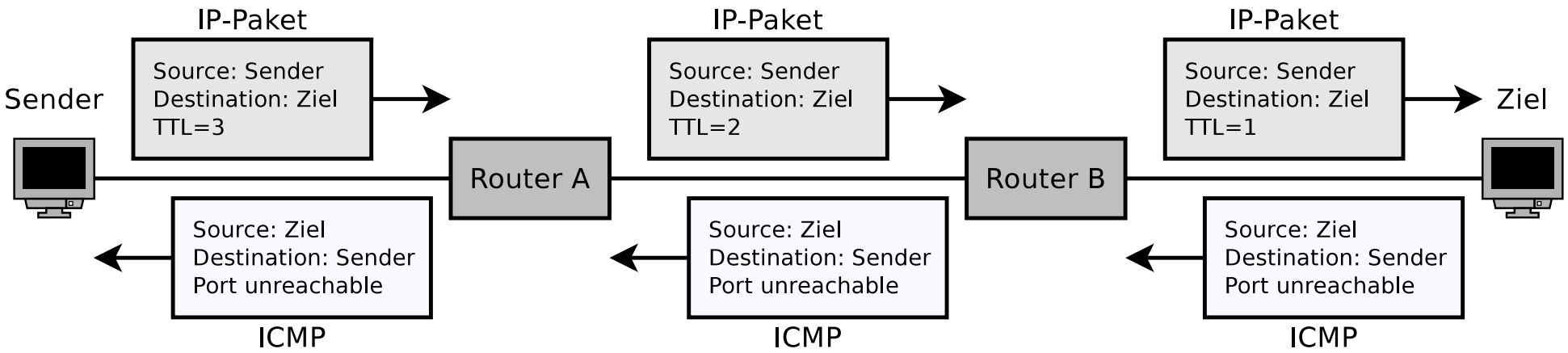
- Ein weiteres Anwendungsbeispiel für ICMP ist das Werkzeug `traceroute`
- `traceroute` ermittelt, über welche Router Datenpakete bis zum Ziel vermittelt werden
- Der Sender schickt ein IP-Paket an den Empfänger mit `TTL=1`
- Router A empfängt das IP-Paket, setzt `TTL=0`, verwirft das IP-Paket und sendet 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 und dabei wird auch der Wert von `TTL` dekrementiert
- Der zweite Router empfängt das IP-Paket, setzt TTL=0, verwirft das IP-Paket und sendet eine ICMP-Nachricht an den Sender
- 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

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

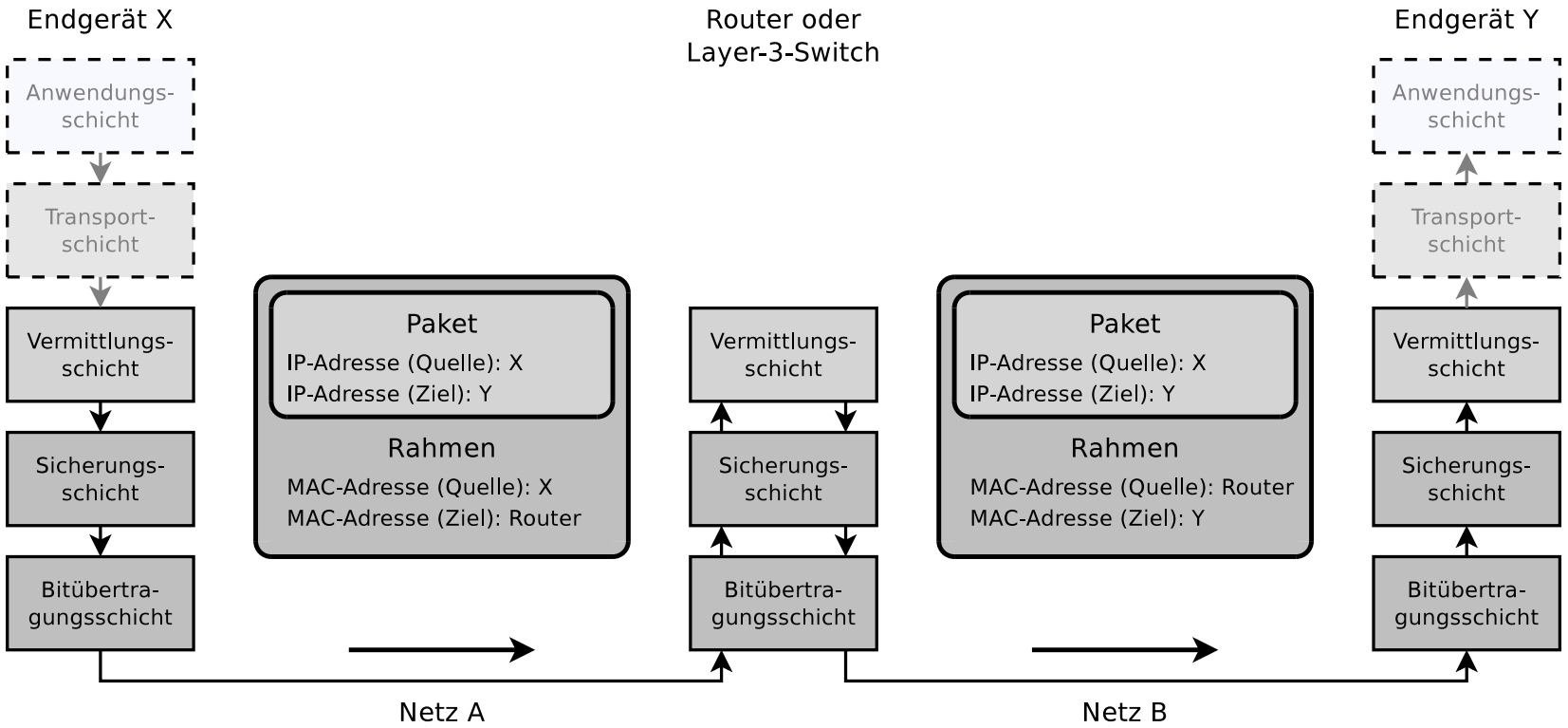


- Sobald der Wert von TTL groß genug ist, dass der Empfänger erreicht wird, sendet dieser eine ICMP-Nachricht vom Nachrichtentyp 3 und Code 3 an den Sender
- So kann der Sender via ICMP den Weg zum Empfänger nachvollziehen

```
$ 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
```

Netzübergreifende Kommunikation (1/6)

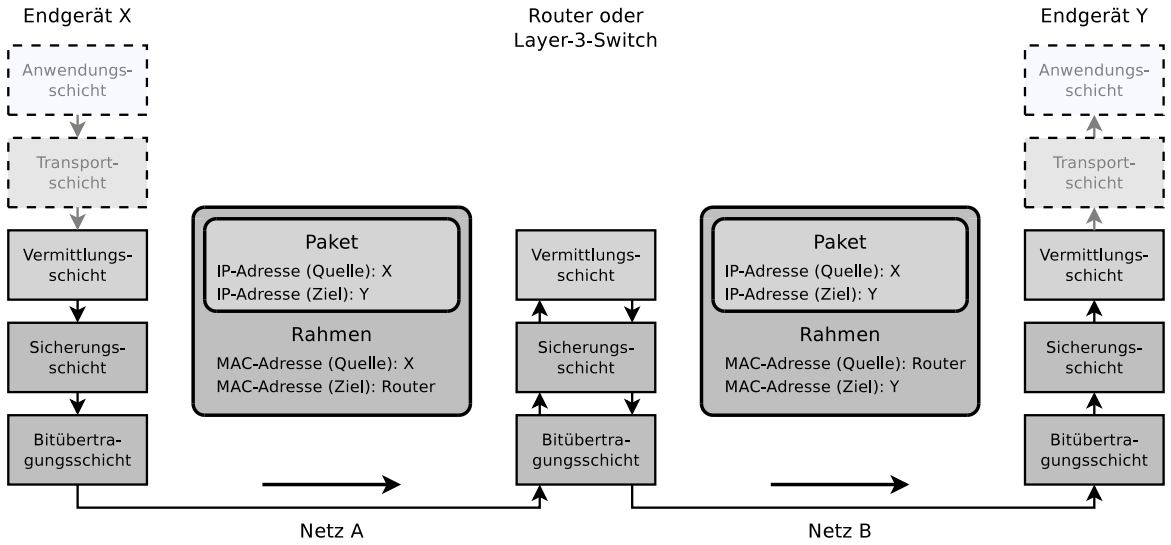
- **Internetworking** = Kommunikation zwischen Netzwerkgeräten mit Protokollen der Sicherungsschicht und Vermittlungsschicht über Netze, die auf unterschiedlichen Vernetzungstechnologien basieren können
- Denkbares Szenario für Internetworking



Netzübergreifende Kommunikation (2/6)

In diesem Szenario haben alle Kommunikationspartner öffentliche IP-Adressen

- X will ein IP-Paket an Y senden
  - Dafür muss X die **logische Adresse** (IP-Adresse) von Y kennen



Sie wissen bereits (aus Foliensatz 4)...

Für die Weiterleitung auf der Sicherungsschicht ist auch die **physische Adresse** (MAC-Adresse) nötig

- X berechnet die Subnetznummern ($\Longrightarrow$ Foliensatz 7) ...
  - Netzmaske_X `AND` IP-Adresse_X = Subnetznummer des eigenen Netzes
  - Netzmaske_X `AND` IP-Adresse_Y = Subnetznummer des Netzes von Y

Netzübergreifende Kommunikation (3/6)

- Identische Subnetznummern $\Longrightarrow$ X und Y sind im gleichen logischen Subnetz

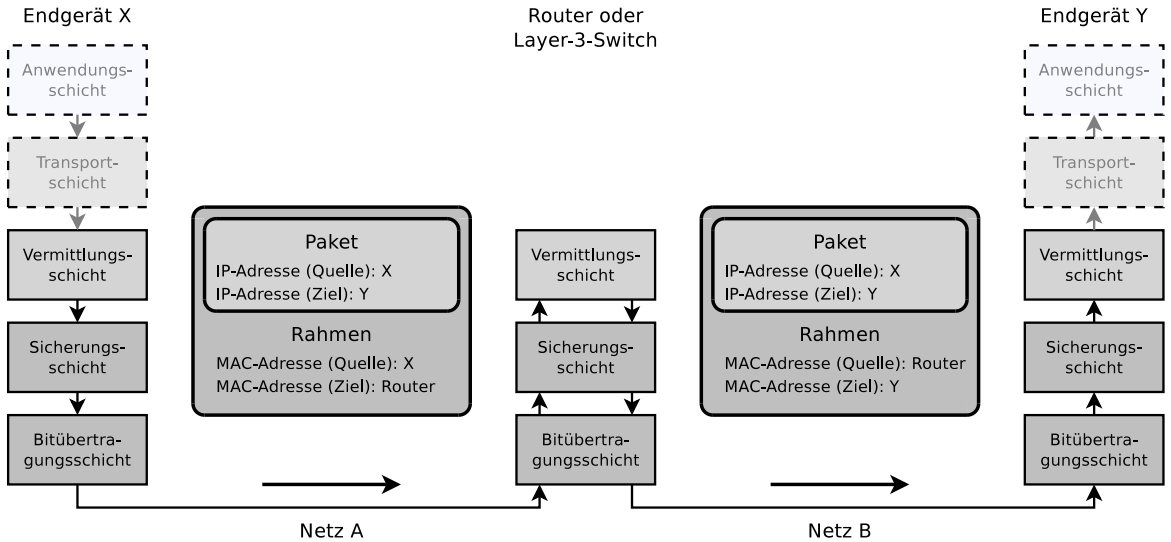
Sie wissen bereits (aus Foliensatz 7)...

Ein logisches Subnetz deckt mindestens ein physisches Netz ab und kann immer nur mit einer Schnittstelle eines Routers verbunden sein

- Unterschiedliche Subnetznummern $\Longrightarrow$ X und Y sind in verschiedenen logischen Subnetzen

Sie wissen bereits (aus Foliensatz 6)...

Befinden sich 2 Kommunikationspartner im gleichen logischen und physischen Netz, kann der Sender via Adressauflösung mit ARP die MAC-Adresse von Empfängers erfahren



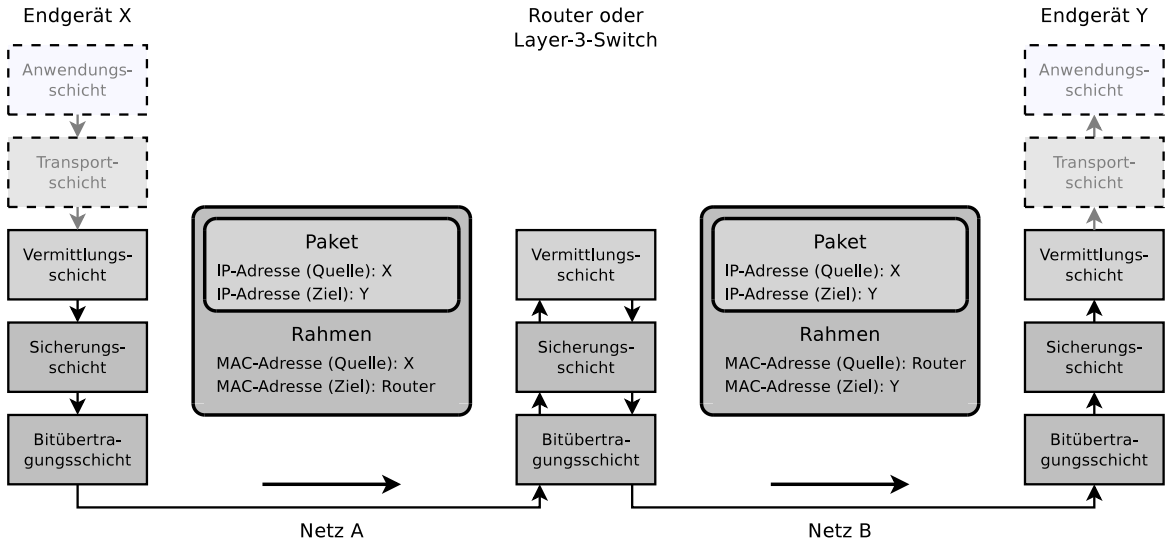
- **Hier handelt es sich um Kommunikation über logische und physische Netzgrenzen hinweg**

Netzübergreifende Kommunikation (4/6)

Sie wissen bereits (aus Foliensatz 6)...

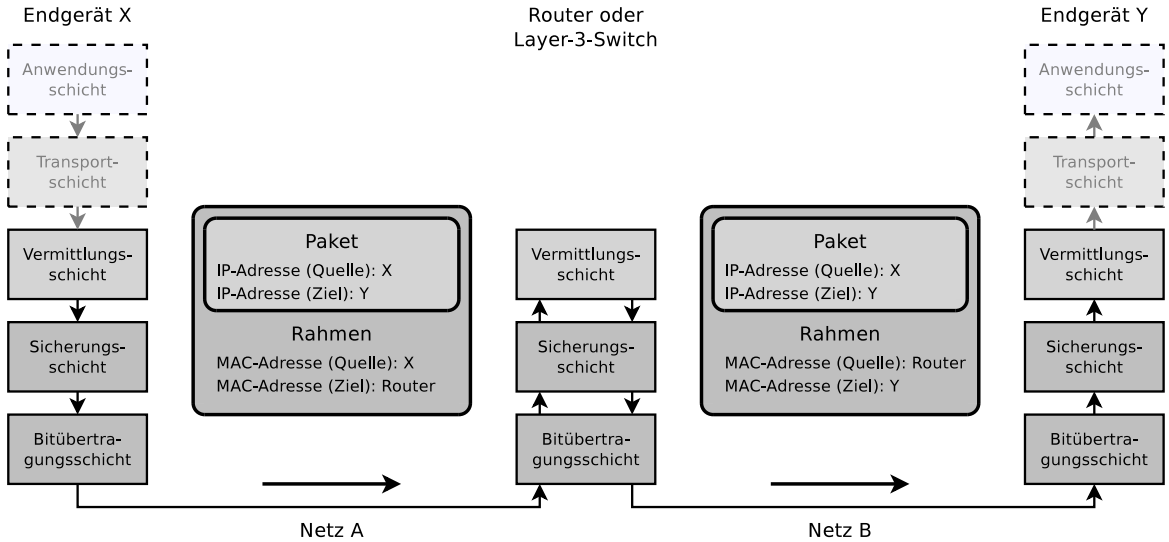
- ARP ist nur für die Auflösung der MAC-Adressen im lokalen physischen Netz zuständig
- Grund: ARP-Anfragen werden in Rahmen der Sicherungsschicht gesendet
- Das Feld mit der Zieladresse enthält die Broadcast-Adresse
- Solche Rahmen werden von Bridges und Switches nicht weitergeleitet
  $\Longrightarrow$ Darum ist mit ARP keine netzübergreifende Adressauflösung möglich

- Im Nutzdatenteil des Rahmens befindet sich das IP-Paket für Y mit der IP-Adresse von X als Quelle und der IP-Adresse von Y als Ziel



Netzübergreifende Kommunikation (5/6)

- Der Router empfängt das IP-Paket
  - Er ermittelt mit seiner lokalen Routing-Tabelle, die alle ihm bekannten logischen Netze enthält, die korrekte Schnittstelle für die Weiterleitung des Pakets
- Der Router ist über eine seiner Schnittstellen mit dem physischen Netz verbunden ist, über das auch Y erreichbar ist

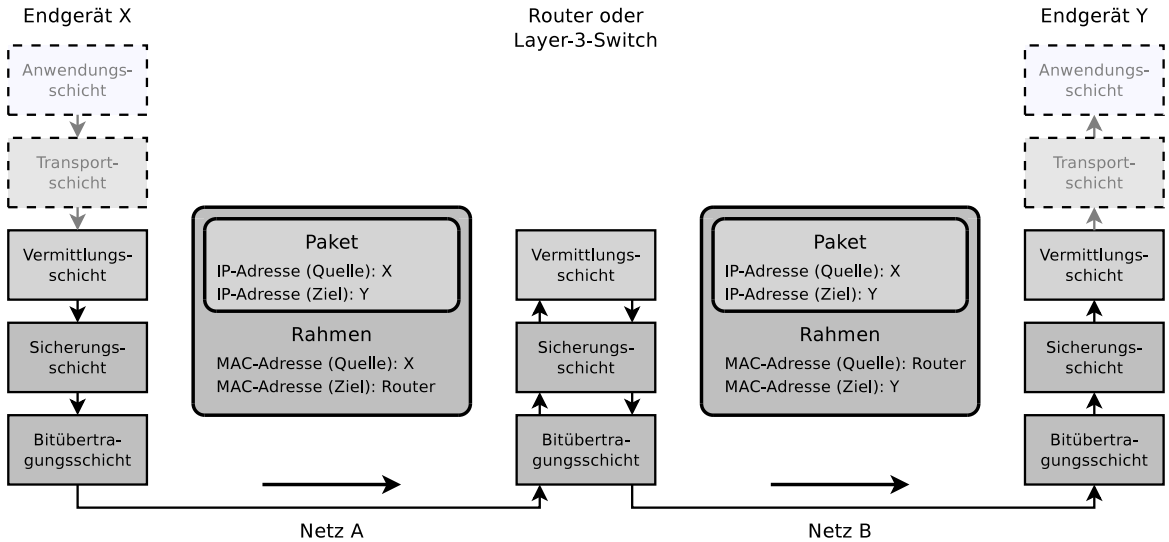


- Der Router ermittelt die MAC-Adresse von Y via Adressauflösung mit ARP
- Der Router verpackt das IP-Paket in einem Rahmen
  - Das Feld mit der Senderadresse enthält die MAC-Adresse des Routers
  - Das Feld mit der Zieladresse enthält die MAC-Adresse von Y

Netzübergreifende Kommunikation (6/6)

- Möglicherweise ist die maximale Paketlänge (*Maximum Transmission Unit*) von Netz B kleiner als die von Netz A
  - Dann kann es abhängig von der Größe des weiterzuleitenden IP-Pakets nötig sein, dass der Router das empfangene Paket in mehrere kleinere Pakete fragmentiert ($\Longrightarrow$ Foliensatz 7)

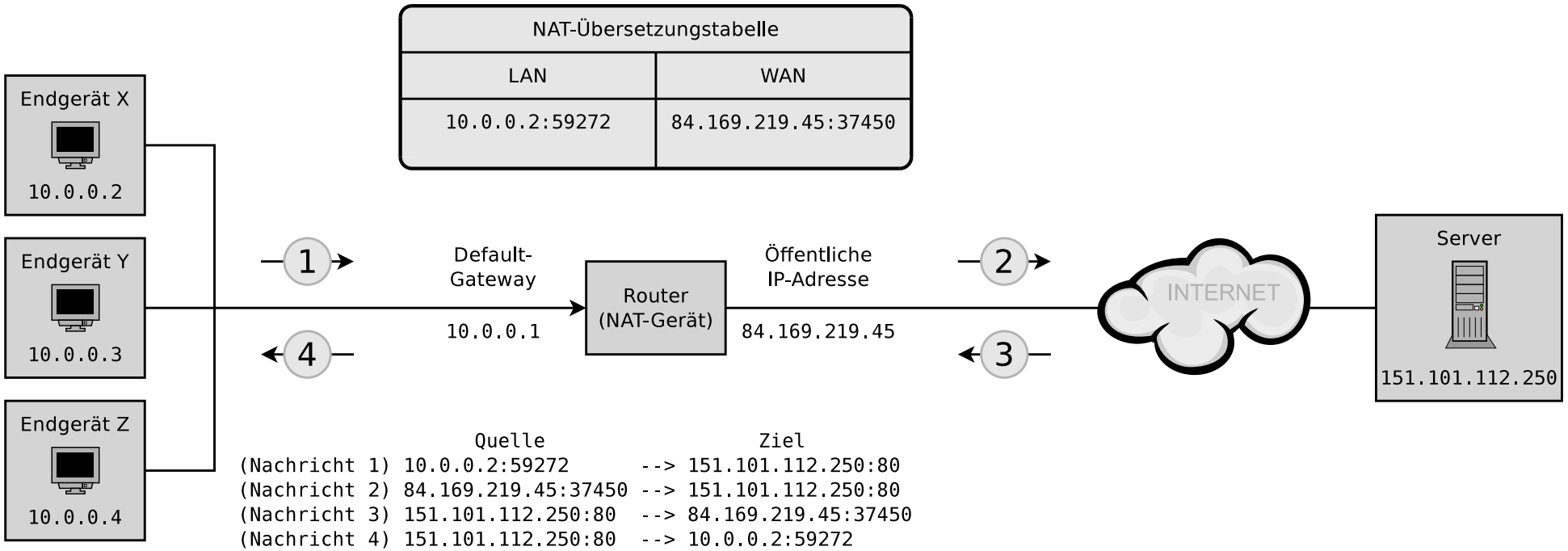
- Die IP-Adressen von Sender (X) und Empfänger (Y) im IP-Paket werden bei der Weiterleitung nicht verändert



Network Address Translation (1/5)

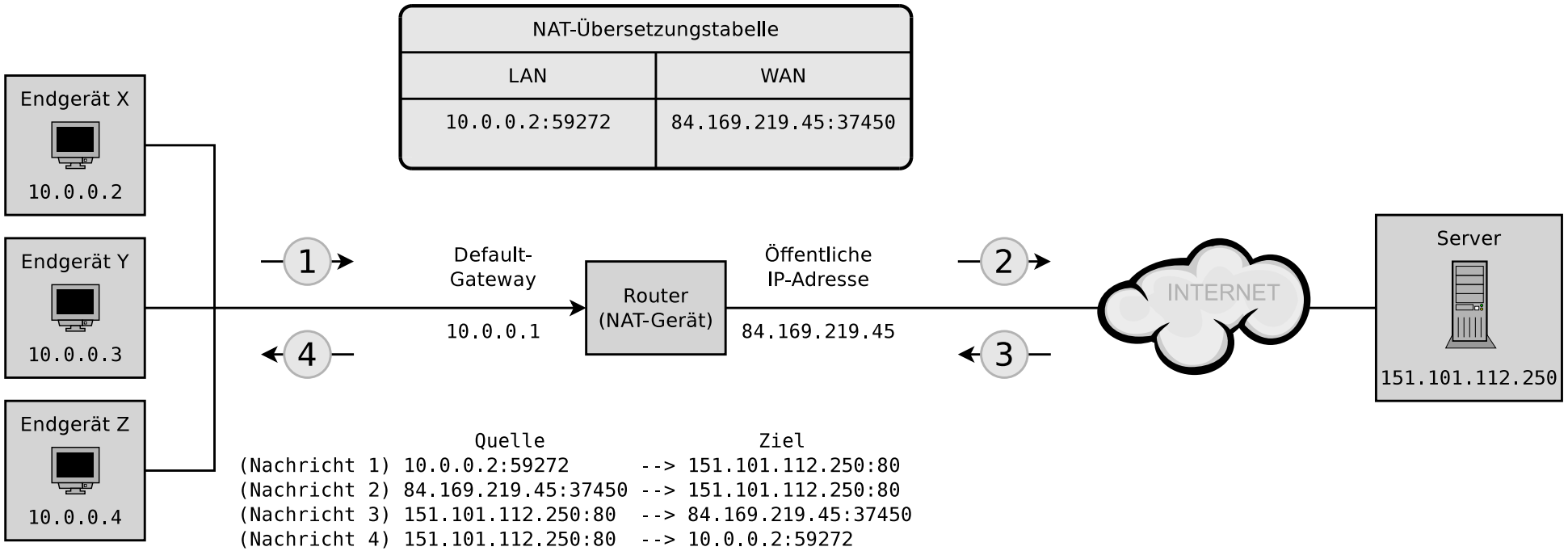
- Problem: Die allerwenigsten Haushalte, Unternehmen und Bildungs-/Forschungseinrichtungen haben genug öffentlich erreichbare IPv4-Adressen, um alle ihre Netzwerkgeräte mit eigenen IPs auszustatten
  - Darum verwenden lokale Netze meist einen privaten IPv4-Adressraum (siehe Foliensatz 7)
  - Problem: Wie können Netzwerkgeräte in privaten Netzen mit Netzwerkgeräten mit global erreichbaren Adressen kommunizieren?
  - Lösung: **Network Address Translation** (NAT)
    - Der lokale Router gibt sich selbst als Quelle derjenigen IP-Pakete aus, die er aus dem direkt verbundenen privaten Netz ins Internet weiterleitet
    - Zudem leitet er eintreffende Antworten zu den Teilnehmern im direkt verbundenen privaten Netz zu

Network Address Translation (2/5)



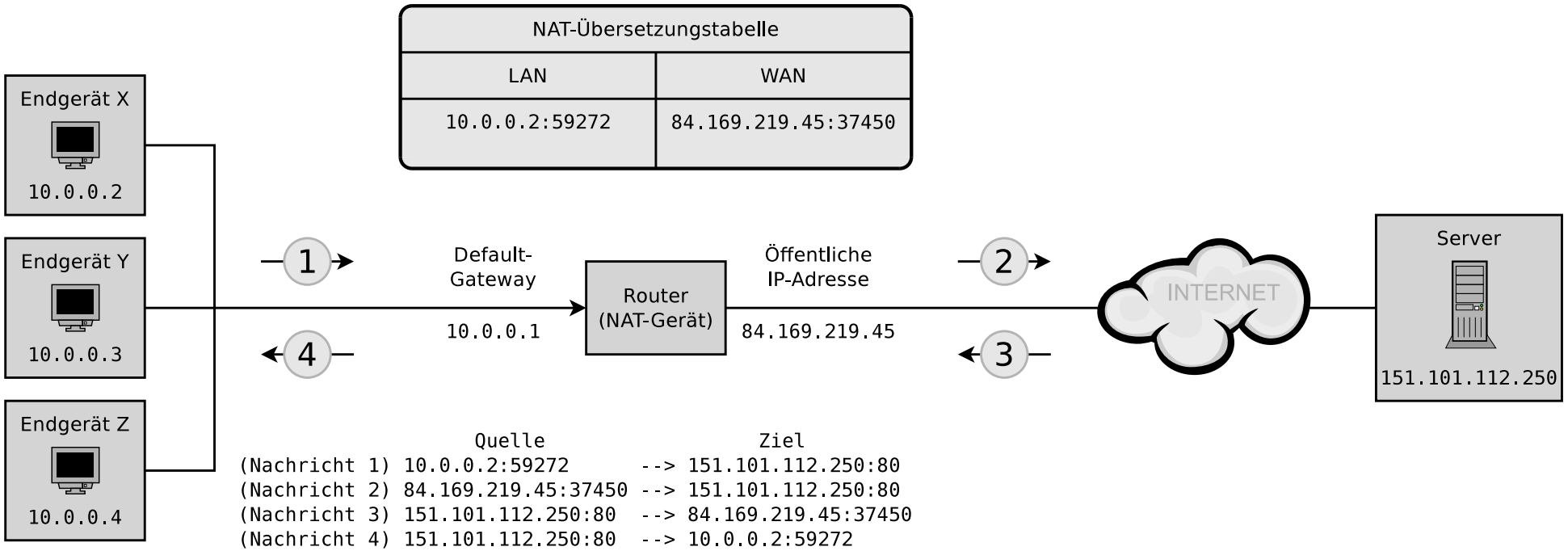
- Die Clients X, Y und Z befinden sich in einem Netz mit einem privaten IP-Adressbereich
- Nur der Router hat eine global erreichbare IP-Adresse
  - Er wirkt für die Außenwelt nicht wie ein Router, sondern wie ein Netzwerkgerät mit einer einzelnen öffentlich registrierten IP-Adresse

Network Address Translation (3/5)



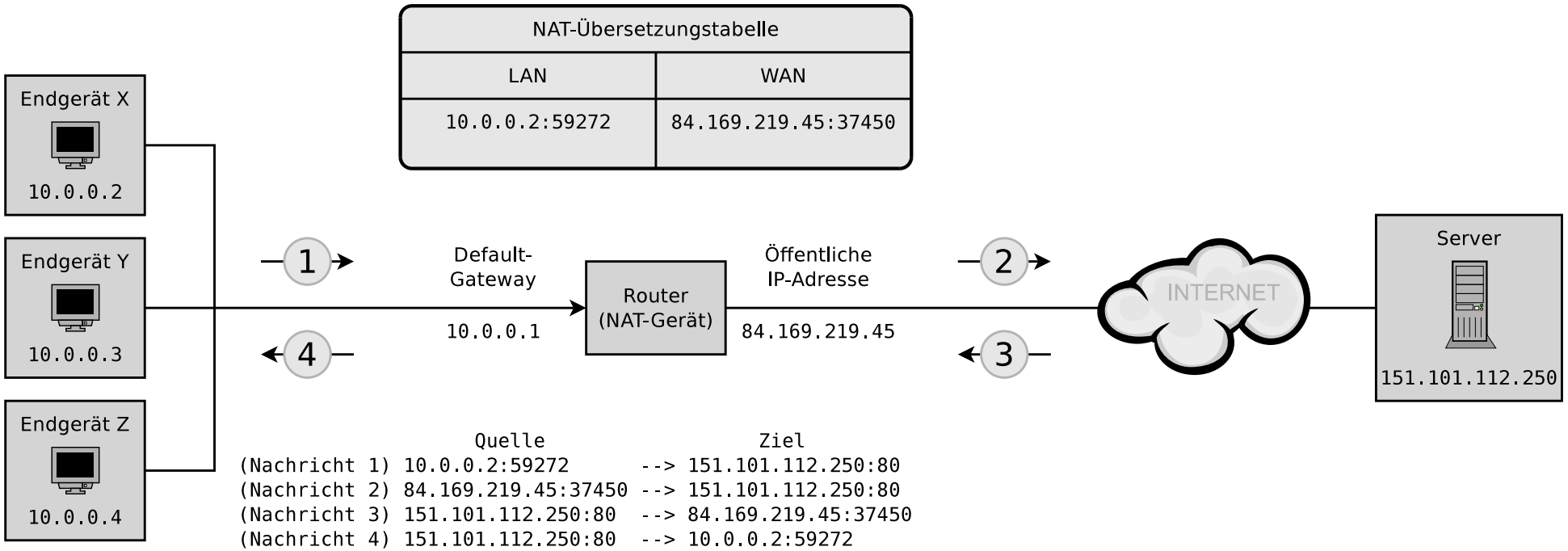
- Client X fordert eine Webseite vom Server an
  - Die Anfrage (Nachricht 1) enthält als Quelladressen die IP-Adresse und Portnummer von X und als Zieladressen die IP-Adresse und Portnummer des Servers
- Der Router ersetzt in der weitergeleiteten Anfrage (Nachricht 2) die IP und Portnummer des Clients durch seine eigenen Adressen

Network Address Translation (4/5)



- Die Zuordnungen zwischen den Ports des Routers und den zugehörigen Netzwerkgeräten im lokalen Netz speichert der Router in einer **NAT-Übersetzungstabelle** (*NAT Translation Table*)
- Die Antwort des Servers (Nachricht 3) ist an den Router adressiert
  - Dieser ersetzt die Adressinformationen entsprechend der Tabelle und leitet die Antwort an X weiter (Nachricht 4)

Network Address Translation (5/5)



- Bei IPv6 ist NAT unnötig, weil der Adressraum groß genug ist, um allen Netzwerkgeräten global erreichbare Adressen zuzuweisen
  - Ob das aus Gründen der Sicherheit allerdings ratsam ist, ist umstritten
    - NAT verbessert die Netzwerksicherheit, weil es die Topologie des lokalen Netzes vor der Außenwelt verbirgt
- NAT bei IPv6: **IPv6-to-IPv6 Network Address Translation** (NAT66)