

Routing

Routing

- * Berechnung der Routing-Tabelle für jeden Router
- * Dazu tauschen Router untereinander Kontrollnachrichten aus (= Routingprotokolle)

VS

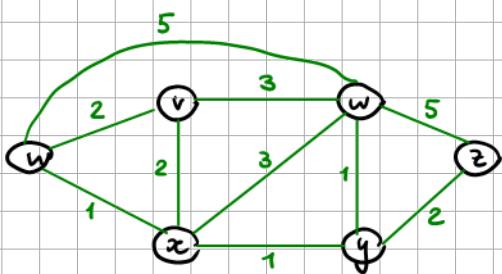
Forwarding

Weiterleitung von Paketen zur Zieladresse mit Longest Matching Präfix

Abstraktion des Internets als Graph

$$G = (N, E), \quad N - \text{Menge der Router } \{u, v, w, x, y, z\}$$

E - Menge der Links



$c(n, n')$ - Kosten
Kosten des Pfades
 $= c(u, v) + c(v, x) \dots$

Ziel: Berechnen des Pfades mit minimalen Kosten zwischen zwei Routern

Klassifikation: Routing

Zentral: wir wissen alle Wege

Zentral oder dezentral?

• Link State:

- Topologie Info werden geflutet, jeder Router kennt **komplette Topologie**
- Berechnung der kürzesten Wege: Dijkstra
- Bsp: **Open Shortest Path First (OSPF)**

• Distanz Vector:

- Jeder Router kennt nur direkte Nachbarn und nur ihre Kosten.
Nachbarn teilen per Routingnachricht mit, welche Knoten mit welchen Gesamtkosten sie erreichen können
- Berechnung: **async Bellman-Ford**
- Bsp: **Routing Information Protokoll (RIP)**

Statisch und dynamisch?

- **Statisch:** Manuelle Konfiguration der Forwarding Tabelle
- **Dynamisch:** Periodischer Austausch der Routinginformation, Änderungen werden automatisch erkannt

	Linkstate (LS)	Distanz Vector (DV)
Routingnachrichten	jeder Router fließt Infos über seine Links im ganzen Netz	jeder Router informiert seine Nachbarn welche Ziele und mit welchen Kosten er erreichen kann
Konvergenz	Berechnung auf jedem Router hat Komplexität $O(E \cdot V)$, falls binäre Heaps verwendet werden	Konvergenzgeschwindigkeit hängt von der Reihenfolge des Nachrichtenaustausches ab (Count-to-Infinity Problem)
Robustheit (was passiert, wenn ein Router "bösaig" ist)	<ul style="list-style-type: none"> Router kündigt falsche Linkkosten an Fehler begrenzt, da jeder Router seine eigene Tabelle berechnet 	<ul style="list-style-type: none"> Router kann falsche Pfadkosten ankündigen Fehler pflanzen sich fort, da Tabelle eines Routers Einfluss auf andere Router-Tabelle hat

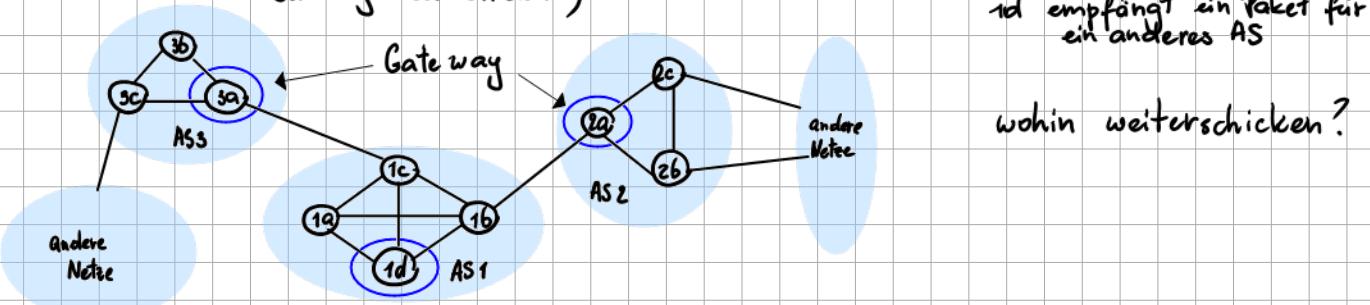
Hierarchisches Routing

Internet = Netz von Netzen

- Administrative Unabhängigkeit der einzelnen Netze (jedes Netz will Routing für sein Netz selbst kontrollieren)
- Skalierbarkeit:
 - nicht jeder Router muss alle Subnetze kennen
 - nicht jeder Router sieht alle Änderungen / Linkausfälle

Lösung: Hierarchisches Routing:

- gruppiere Router in "Autonome Systeme" (AS) (aka "domains")
(Bsp-le: Deutsche Telekom, Deutsches Forschungsnetz, ...)
- Intradomain Routing:** Routing für Ziele im gleichen AS bestimmt welche externen Ziele über welches Transfer-AS / Gateway erreichbar sind (AS 3 mit za und AS2 mit zc)
- Interdomain Routing:** Routing für Ziele in anderen ASen bestimmt, wie die Gateways zu den Nachbarnetzen aus dem lokalen Netz erreichbar sind (Router 1c informiert 1a, 1b, 1d wie man Gateway za erreicht)



Routing in der Praxis

Intradomain:

(auch Interior Gateway Protokol (IGP))

- RIP (Routing Information Protokol)
- OSPF (Open Shortest Path First)
- IGRP (Interior Gateway Routing Protokol)

Interdomain:

(Exterior Gateway Protokol (EGP))

- BGP (Border Gateway Protokol)

• Wichtig sind Routing Policies:

- jeder R kann lokal bestimmen, was er bevorzugt und welche R-Nachrichten er weiterleitet
- erlaubt es wirtschaftliche Aspekte zu berücksichtigen

Open Shortest Path First (OSPF)

- Link State:
- Infos über Nachbarrouter und -links werden durchs Netz geflutet
 - Jeder R lernt so die komplette Topologie
 - R-Berechnung mit Dijkstra

Router fluten Link State Advertisement Nachrichten an alle anderen Router im gesamten AS

- OSPF werden direkt über IP (nicht TCP, nicht UDP) gesendet
- enthalten Link State für jeden benachbarten Link

Weitere Merkmale:

- Authentisierung der OSPF Nachbarn
- Linkgewichte sind konfigurierbar (nicht zwingend 1!)
- Lastverteilung möglich falls mehrere Pfade mit gleichen Kosten
- Hierarchisches OSPF für große Netze

BGP (Border Gateway Protokoll)

• BGP Session:

- TCP zwischen zwei BGP-Routern
- jeder R teilt mit, welche Ziele (= IP-Präfixe) er kennt

Varianten:

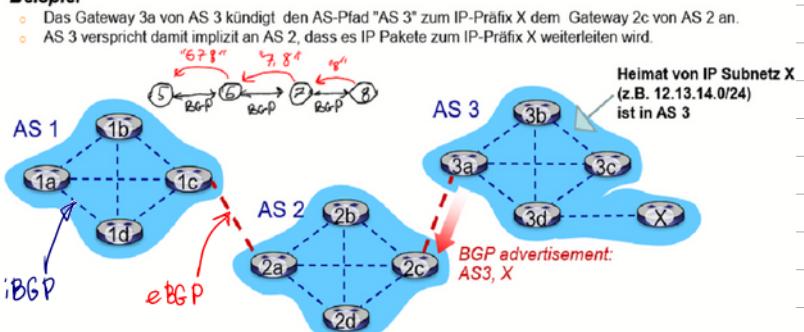
- iBGP (zwischen R-ten im gleichen AS)

- eBGP (zwischen R-ten in benachbarten ASen)

BGP Session

- 2 Router unterhalten TCP Verbindung zum Austausch von Routing Updates.
- Kündige Routen zu IP Präfixe an.
- BGP ist ein "Pfad Vector Protocol"

Beispiel



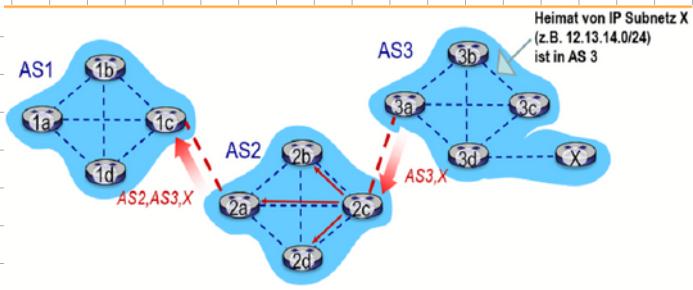
BGP Route besteht aus IP-Präfix & BGP Attribut

BGP Attribute:

- AS-PATH:** Liste von ASen, durch die das Prefix Advertised gelassen ist
- NEXT-HOP:** IP-Adresse des Gateways

Policy-Based Routing:

- BGP-R verwendet Import Policies um einen Pfad zu akzeptieren oder abzulehnen
- Bsp 1: Ignoriere Pfad durch AS Y
- Bsp 2: gib Routinginfo nicht an AS X weiter



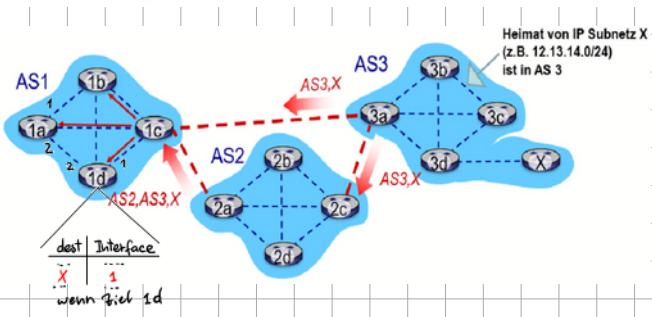
Router 2c empfängt Advertisement AS3,X über eBGP

Import

- Policies von AS 2 erlauben, dass Router 2c diesen Pfad akzeptiert und ihn (über iBGP) an alle anderen Router im AS 1 weitergibt.

Export

- Policies von AS 2 erlauben, dass Router 2a (über eBGP) Pfad AS2, AS3, X an das Gateway 1c von AS1 weitergibt.



ein Gateway Router kann mehrere Pfade zum gleichen Ziel IP Prifix X lernen

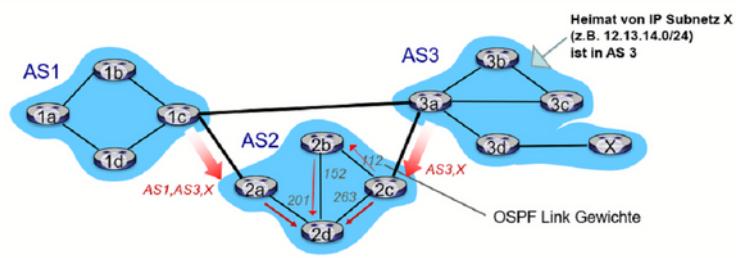
Aufgrund konfigurierter Policy (z.B.: wählt immer den kürzesten Weg) entscheidet sich der R 1c für Pfad AS3,X und kündigt nur diesen Pfad über iBGP intern im AS an

Die besten Routen werden nach folgenden Kriterien gewählt:

- 1) Local Preference (Zuweisung von Prio beim Import)
- 2) Kürzester AS-Pfad (Route, bei der man am wenigsten ASen durchqueren muss)
- 3) Route mit dem am schnellsten erreichbaren Next-Hop (= Gateway)

(Hot-Potato Routing)

Hot Potato Routing

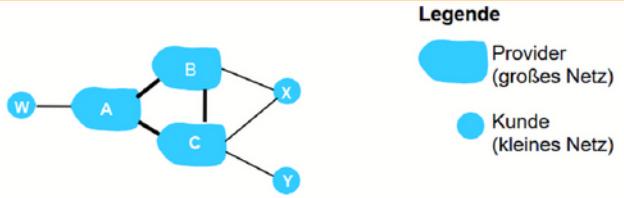


- 2d lernt über iBGP dass es X über 2a oder 2c erreichen kann.

Hot-Potato Routing

- "Jedes Netz möchte Pakete so schnell wie möglich aus eigenem AS/Netz loswerden."
- Wähle lokales Gateway mit den geringsten Intradomain-Kosten
- Hier: 2d wählt 2a, obwohl dann der AS-Pfad länger ist.

Inter-AS Routing: Policies



- A, B, C sind Provider und X, W, Y sind Kunden der Provider
 - Provider verlangen Geld abhängig von der Datenmenge zu den Kunden.
- Dual-Homing:**
 - Kunde kann mit 2 Providern verbunden sein (z.B. X)
- Soll X per Routingprotokoll B sagen, dass es C erreichen kann?
 - Nein → X möchte keinen Transitverkehr von B zu C weiterleiten

IPv6

- Ziele:
- Unterstützung viele viele Hosts
 - kleine kompakte Routingtabellen
 - Vereinfachung des Protokolls (schnellere Verarbeitung)
 - Flexibilität: Erlaubt zukünftige Erweiterungen
 - Migration und Koexistenz von IPv4 und IPv6 während des Übergangs
 - bessere Unterstützung von Multicasting, Mobilität, Quality of Service (QoS)

Doppelt so groß wie IPv4: Header 40 Byte

Diff. Server

- "Priorität" des Pakets oder Flows

Flow Label

- Pakete mit gleichem Label bilden eine Gruppe (= Flow)
- Sollten gleich behandelt werden.
- Sehr verwendet.

Next Header

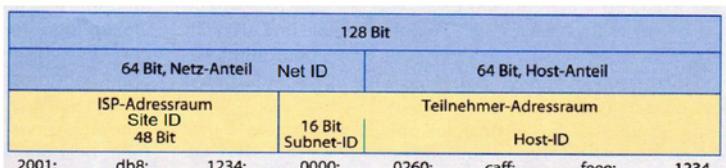
- Gibt an ob Extension Header folgt oder welches Transport Layer Protocol (TCP/UDP)
- Mehrere Extension Header möglich; jeder Header verweist auf den nächsten ("Kette")

Hop Limit = TTL

- Longest Prefix Matching wie bei IPv4!

- Netz- und Host-Anteil
 - Host-ID: Praktisch immer genau 64 Bit (rechter Teil)
 - Es gibt also keine /80 Subnetze

- Praxis:
 - Site-ID: ISP weist Privatkunden z.B. /48 or /56 IP Präfix zu.
 - Subnet-ID: Jeder hat 8 or 16 Bits (subnet-ID) um sein eigenes Netz in weitere Subnetze zu unterteilen.
 - Host-ID: 64 Bit.



Volle Schreibweise

- 128 Bit werden in 8 Blöcke zu je 16 Bit (4 Hexadezimalstellen) unterteilt
- Blöcke werden durch : getrennt
- Beispiel
 - 2001:0db8:85a3:08d3:1319:8a2e:0370:7344

$$2^{128} \rightarrow 128/8 = 16 \text{ Blöcke}$$

Den von vorne muss man nicht extra schreiben

Abgekürzte Schreibweise

- Führende Nullen können weggelassen werden
 - 2001:0db8:85a3:8d3:1319:8a2e:370:7344
- Nur einmal (!) dürfen ein oder mehr aufeinanderfolgende Blöcke mit dem Wert 0000 ausgelassen werden und durch :: ersetzt werden
 - 2001:0db8:0:0:0:1428:57ab wird zu 2001:db8:1428:57ab
- IPv4 Adressen können wie folgt geschrieben werden:
 - ::192.31.20.46

URL Notation von IPv6 Adressen mit eckigen Klammern

- [http://\[2001:0db8:85a3:08d3:1319:8a2e:0370:7344\]:8080/](http://[2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:8080/)

Adressbereiche

- ::1/128 Loopback
- 2000::/3 Global Unicast: Global erreichbare Adressen
- FE80::/10 Link-Local: Nur im lokalen Subnetz gültig

Unterschiede zu IPv4:

- keine Fragmentierung (Router informiert Sender per ICMPv6, dass Nachricht zu groß)
- jeder IPv6 Host verfügt automatisch über Link local IPv6 Adresse (z.B. abgeleitet von MAC und nur im lokalen LAN gültig)
- kein ARP (wird über ein anderes Protokoll (Nachbar Protokoll) implementiert)

Migration: Tunnelling

- Tunnell : IPv6 Datagramm wird in den Nutzdaten eines IPv4 Paketes transportiert falls IPv4 - Leitung passiert werden muss
- Dual - Horned : Die Geräte an den Tunnelenden müssen sowohl IPv4 als auch IPv6 sprechen

