INSTITUT FÜR
INFORMATIK
DER LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LS Prof. Kranzlmüller

Praktikum Rechnernetze

Kapitel 3: Autonome Systeme

MNM
TEAM
Munich Network Management Team

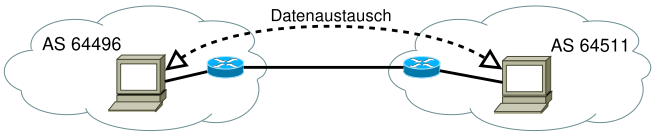
3 Das Internet und Autonome Systeme

Inhaltsverzeichnis

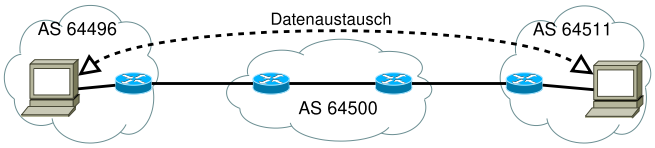
Das Internet ist ein Verbundnetz, bestehend aus den Netzen vieler verschiedener Betreiber (Unternehmen, Bildungseinrichtungen, Militär, usw), die jeweils eigene, administrativ unabhängige Netze betreiben. Das Verbinden zweier Netze verschiedener Netzbetreiber geschieht meist aus gegenseitigem Nutzen (Rechner in dem einen Netz können mit Rechnern im anderen Netz kommunizieren), oder wirtschaftlichem Interesse. In diesem Zusammenhang nennt man die unabhänigigen administrativen Domänen eines Betreibers *Autonome Systeme* (AS). Ein AS ist genehmigungspflichtig (bei der Internet Assigned Numbers Authority, IANA) und trägt eine weltweit eindeutige *AS-Nummer* – diese stellt eine Art Adresse der administrativen Domäne dar.

Grundsätzlich wird unterschieden, ob es sich bei der Verbindung zweier Netze bzw. AS um eine sogenannte *Peering*-Beziehung handelt, oder um eine *Transit*-Beziehung. Abbildung 3.1(a) zeigt AS-Peering, das den Netzteilnehmern in den Netzen der Peering-Partner Kommunikation *in das Netz* des anderen Peering-Partners erlaubt. Transit bedeutet die Erlaubnis, Nachrichten *durch das Netz* eines Betreibers zu vermitteln, auch wenn sie an Endsysteme außerhalb dessen Netz adressiert sind: Das Netz, durch das vermittelt wird dient also als „Durchfahrtstraße". In Abbildung 3.1(b) benötigen die beiden äußeren AS ein Transitabkommen mit dem AS 64500, damit ein Datenaustausch zwischen den beiden Endgeräten erfolgen kann.

Analog zu privaten IP-Adressen wurden AS-Nummern für private Nutzung (z.B. im Testbetrieb) reserviert, sowie zur Nutzung in Dokumentation:



(a) Peering-Beziehung zwei benachbarter Autonomer Systeme



(b) Transit-Beziehungen zwischen benachbarten Autonomen Systemen

Abb. 3.1: Zwei Arten von Beziehungen zwischen Autonomen Systemen erlauben den Datenaustausch zwischen zwei Rechnern

```
64512-65534 Designated for private use (Allocated to the IANA)
64496-64511 Reserved for use in documentation and sample code  [RFC5398]
```

Die Versuche im Praktikum nutzen diese privaten AS-Nummern.

3.1 Vermittlungsschicht (OSI-Schicht 3)

Aufbauend auf der Sicherungsschicht (siehe Kapitel 2.2), die dafür zuständig ist Rahmen durch ein LAN zu transportieren, werden mit den Funktionen der Vermittlungsschicht Nachrichten von der Quelle, durch das Transportnetz, bis zum endgültigen Ziel übertragen. Dabei können Teilnetze durchquert werden, die beliebige Schicht 2 Implementierungen einsetzen. Dadurch wird es ermöglicht, Daten zwischen Endpunkten in heterogenen Netzen auszutauschen. Die Komponenten der Schichten 1, 2 und 3 bilden zusammen das Transportsystem.

Da die Vermittlungsschicht Datagramme bis zum endgültigen Ziel überträgt, müssen alle Endpunkte über alle Teilnetze hinweg eindeutig adressierbar sein. Mit dem geplanten Einsatzzweck viele Schicht 2 Netze beinhalten zu können, muss bei der Entwicklung von Schicht 3 Protokollen davon ausgegangen werden, dass man mit sehr vielen Teilnehmern zurecht kommen muss, im Gegensatz zu einem einzelnen Schicht 2 Teilnetz.

Zwei Hauptaufgaben der Schicht 3 sind also eine globale (netzweite), eindeutige Adressierung aller möglicher Endpunkte und die Verschattung der zugrunde liegenden Übertragungstechniken. Protokolle höherer Schichten müssen sich dadurch nicht mehr mit Anzahl, Art und Topologie der Komponenten des Transportnetzes befassen.

3.1.1 Fragmentierung

Oft wird die maximale Länge einer Schicht-N-PDU durch die darunter liegende Schicht beschränkt. Auf dem Weg zwischen Sender und Empfänger durchquert ein Datagramm (Schicht-3-PDU) häufig verschiedene Subnetze bzw. LANs, deren Schicht-2-PDUs unterschiedliche maximale Längen für Nutzdaten zulassen. Ein Datagramm gegebener Größe muss daher in kleinere *Fragmente* aufgeteilt werden, jedesmal wenn ein Subnetz passiert werden soll, dessen MTU (engl. Maximum Transmission Unit) kleiner ist, als die Größe des Datagramms. Ethernet (ohne Erweiterungen) erlaubt z.B. bis zu 1500 Bytes Nutzdaten pro Rahmen.

Fragmentierung von IP-Paketen

Wird ein IP-Paket fragmentiert, so sind die entstehenden Fragmente ebenfalls IP-Pakete. Im IP-Header eines jeden Fragments bestimmt das *Fragment Offset*-Feld die Position der Daten (in Einheiten von 64-Bit) im ursprünglichen IP-Paket. Das *more*-Flag zeigt an, dass auf ein Fragment ein weiteres Fragment folgt. Bis auf das letzte Fragment hat jedes Fragment das more-Flag im Header gesetzt. Mit Hilfe des Fragment Offset-Felds und dem more-Bit kann ein Empfänger der Fragmente diese wieder zusammensetzen und ggf. das ursprüngliche IP-Paket weiterleiten.

3.1.2 Tunneling

Tunneling ist eine weitere Methode um ein logisches Netz in einem vorhandenen Netz zu implementieren. Eine ähnliche Technik haben Sie mit VLANs im vorherigen Abschnitt kennengelernt. Im Gegensatz zu VLANs, die eine logische Topologie mit mehreren Endpunkten auf einen physischen Aufbau aufbringen, zielen Tunnel darauf ab, eine direkte (logische) Verbindung zwischen zwei Endpunkten zu erstellen. Dabei wird das Netz (oder auch mehrere Netze) zwischen den Endpunkten zu einer virtuellen Leitung abstrahiert.

Durch diese Technik können Teilnetze transparent zusammengeschaltet werden. Sie wird häufig eingesetzt, um entfernte Standorte in ein internes Firmennetz zu integrieren oder mobilen Endgeräten mit Internetzugang Zugriff auf die internen Ressourcen zu ermöglichen.

Abbildung 3.2 zeigt ein IP-Paket, das vom Rechner „Sender" an „Empfänger" gesendet wird. Bis zum ersten Router A wird das IP-Paket als Nutzdaten, beispielsweise innerhalb eines Ethernet-Rahmens, transportiert. Router A behandelt das IP-Paket wie Nutzdaten einer höheren Schicht: er erstellt ein äußeres IP-Paket, schreibt unser ursprüngliches Paket in das Nutzdatenfeld des äußeren IP-Pakets und vermittelt das äußere Paket weiter auf dem Pfad zu Router B. Dieser stellt das ursprüngliche IP-Paket dem Empfänger zu. Das Netz zwischen den Routern A und B wird dabei aus Sicht unseres (inneren) IP-Pakets zu einer direkten Verbindung – einem Tunnel – abstrahiert.

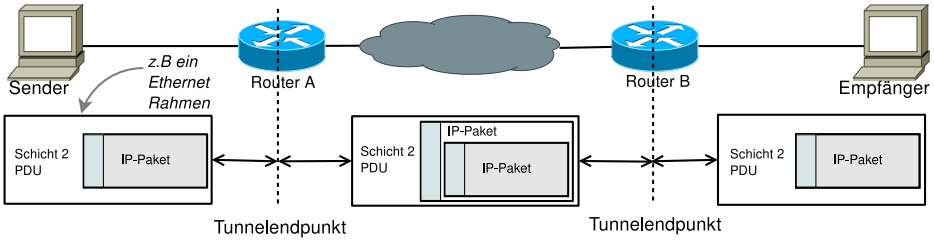


Abb. 3.2: Ein IP-Paket wird für den Transit zwischen den Routern A und B in ein weiteres IP-Paket gekapselt

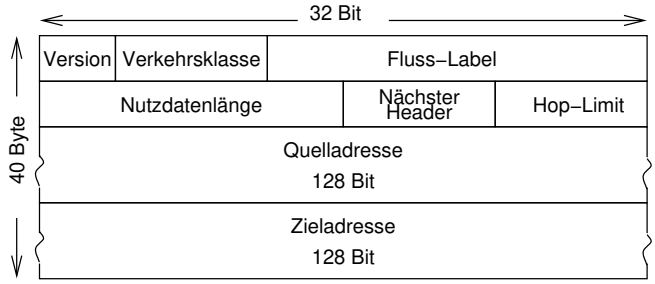


Abb. 3.3: Aufbau des IPv6 Headers

3.2 Internet Protocol Version 6

Seit der Einführung von IPv4 in den 1980er Jahren haben sich die Anforderungen an das Protokoll verändert. Dies spiegelt sich in nachträglichen Änderungen und Anpassungen wieder. Zum Beispiel wurde die Bildung von Subnetzen Jahre nach der Veröffentlichung von [RFC 791] (IPv4) mit [RFC 1519] (CIDR) verändert. Durch das stetige Wachstum der Anzahl an Endgeräte im Internet und die hierarchische Aufteilung von IP-Adressen beinhaltet der 32-Bit Adressraum von IPv4 nicht genug Adressen für den Bedarf im Internet. Der 128-Bit Adressraum von IPv6 ist ein wesentlicher Grund für dessen Entwicklung als designierter Nachfolger von IPv4.

Als eigenständige Neuentwicklung konnte IPv6 ([RFC 2460]) von den Erfahrungen mit IPv4 profitieren. So wurde IPv6 mit vielen Funktionen und Eigenschaften entwickelt, mit der Absicht Probleme von IPv4 zu lösen und die Erweiterung des Protokolls zu erleichtern. Einige der wichtigsten Eigenschaften von IPv6 sind:

128-Bit Adressen IPv6-Adressen sind 128-Bit lang und werden hierarchisch in Subnetze aufgeteilt. Eine IPv6-Adresse setzt sich aus einer 64-Bit langen Subnetz-ID und einer 64-Bit langen Host-ID zusammen. Die größe des Adressraums für Host-IDs erlaubt es (weltweit) eindeutige Host-IDs zu vergeben. Dafür sieht die ursprüngliche IPv6-Spezifikation einen entsprechenden Algorithmus zur automatischen Generierung von Host-IDs vor. Das Ziel war die eindeutige Identifizierung

eines Hosts unabhängig von der Subnetz-ID und damit von dem Teilnetz indem sich ein Host befindet zu machen.

Besonders für mobile Endgeräte ist dies ein relevanter Aspekt, denn diese können häufig von einem (Funk-)Netz in ein anderes wechseln. Unter Anderem ist es hier für die Abrechnung (siehe Abschnitt „Abrechnungs-Management") wichtig mobile Endgeräte über Subnetze hinweg eindeutig identifizieren zu können. Heute ist man von dem Gedanken der eindeutigen Adressierung zum Schutz der Privatsphäre der Benutzer abgekommen, allerdings ist die technische Möglichkeit nach wie vor gegeben.

Feste Header-Länge und Header-Extensions Für die Vermittlung einer Nachricht ist es wichtig die Länge des IP-Headers zu kennen, da dieser eventuell relevante Informationen für die Vermittlung auf Schicht 3 enthält. ☺ Der IPv6-Ansatz sieht einen Header fester Länge vor, der durch *Header-Extensions* um zusätzliche Informationen erweitert werden kann. Zu diesem Zweck ist im IPv6-Header das Feld „Next-Header" vorgesehen (siehe Abbildung 3.3). Dies ermöglicht es den IPv6-Header nur bei Bedarf um Funktionen zu erweitern. Außerdem kann IPv6 zu einem späteren Zeitpunkt leicht durch das Definieren neuer Header-Extensions erweitert werden. Das erste Feld einer jeden Header-Extension ist ein Next-Header-Feld. Auf diese Art und Weise können Header-Extensions kombiniert werden.

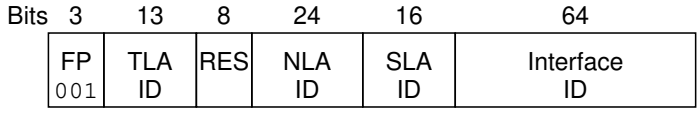
IPsec wurde ursprünglich als [RFC 2401] eingeführt, um bereits auf der Ebene der Vermittlungsschicht Daten verifizieren und verschlüsseln zu können. IPsec wurde komplett in IPv6 übernommen und kann mittels Header-Extensions eingesetzt werden.

Autoconfiguration Unter dem Stichwort *Autoconfiguration* werden mehrere Funktionen und Eigenschaften von IPv6 zugefasst. Ein prominenter Vertreter aus dieser Kategorie ist das automatische Generieren der Host-ID, zum Beispiel (aber nicht unbedingt!) aus der MAC-Adresse. Des weiteren implementiert IPv6 Gültigkeitsbereiche (engl. Scopes) von IP-Adressen. Die hier relevanten Scopes sind „link-local" und „global". IP-Adressen deren Scope link-local ist werden nicht geroutet. Aus einem im Standard definierten Präfix und der automatisch generierten Host-ID erhält jede Schnittstelle, auf der IPv6 eingesetzt wird, eine eindeutige link-local IP-Adresse.

3.2.1 Adressen

IPv6-Adressen werden als Folge von 16 Bit langen Segmenten notiert. Jedes der acht Segmente wird als Hexadezimalzahl aufgeschrieben, wobei je zwei Segmente (Oktettenpaare) durch einen Doppelpunkt ':' getrennt werden, zum Beispiel die Adresse `1080:0:0:0:8:800:200C:417A`.

Aufgrund des großen Adressraums kommen oft Bereiche einer Adresse vor, in denen mehrere aufeinanderfolgende Oktettenpaare den Wert null haben. Zugunsten einer verkürzten Schreibweise darf (höchstens einmal) eine Folge von Nullen zu zwei aufeinander



Feld	Name
FP	Format Prefix
TLA ID	Top-Level Aggregation Identifier
RES	reserviert
NLA ID	Next-Level Aggregation Identifier
SLA ID	Site-Level Identifier

Abb. 3.4: Struktur einer Global Unicast IPv6-Adresse

folgenden Doppelpunkten '::' verkürzt werden. Die Anzahl der Null-Oktette, die so repräsentiert werden kann offensichtlich aus der festen Länge einer Adresse und den ausdrücklich angegebenen Oktetten abgeleitet werden, z.B:

nicht verkürzt	`1080:0:0:0:8:800:200C:417A`
verkürzt	`1080::8:800:200C:417A`

Aufbau einer Adresse Die Abkehr von Netzklassen, wie sie ursprünglich in IPv4 verwendet wurden, eröffent neue Möglichkeiten für die Strukturierung von IP-Adressen, insbesondere für die Subnetz-ID. Die Struktur von IPv6-Adressen ist dahingehend gestaltet, den organisatorischen Gegebenheiten im Internet gerecht zu werden: es werden seitens verschiedener Organisationen Kern- und Transitnetze betrieben, sowie Zugangsnetze und lokale Netze. Dies wird abgebildet auf eine dreistufige Adresshierarchie. Abbildung 3.4 zeigt die Aufteilung einer *Aggregatable Global Unicast Address* in Segmente, die aus der Adresshierarchie abgeleitet werden. Die ersten 48 Bit der Adresse bezeichnen die **Public Topology** (insbesondere die Felder TLA und NLA), zum Beispiel Transitnetze. Die zweite Hierarchietufe ist die **Site Topology** (SLA-Feld), die standortbezogene Adressierung bezweckt. Die dritte Hierarchiestufe wird durch den **Interface Identifier** repräsentiert. Diese entspricht am ehesten der aus IPv4 bekannten Host-ID.

Spezielle Adressen Auch bei IPv6 sind bestimmte Adressen bzw. Präfixe mit besonderer Bedeutung belegt, unter anderem gibt es Festlegungen für eine nicht definierte Adresse für eine Schnittstelle, einer loopback-Adresse und privaten Adressen.

nicht definiert	`0:0:0:0:0:0:0:0` (bzw. '`::0`')
loopback	`0:0:0:0:0:0:0:1` (bzw. '`::1`')
private Adressen	Private Adressen sind solche, die nicht im Internet vermittelt werden. Dazu gehören bei IPv4 beispielsweise die Adressen des 10.*-Netzes. Bei IPv6 gehören dazu auch die sogenannten *link-local*-Adressen, mit dem 10-Bit-Präfix `1111 1110 10` bzw. das Subnetz `FE80::0/10`.

3.2.2 Hilfsprotokolle

Auch für IPv6 benötigt man bestimmte Funktionen, die in Hilfsprotokollen implementiert werden. Die bekannten Funktionen der (IPv4-)Hilfsprotokolle ARP und ICMP sind in ICMPv6 ([RFC 4443]) konsolidiert. Die Neuauflage war notwendig, da IPv6 subtile konzeptionelle Unterschiede zu IPv4 aufweist. Zum Beispiel geht IPv6 von Punkt-zu-Punkt Verbindungen auf Schicht 2 aus, während ARP für den Betrieb in einem Broadcast-Netz erforderlich ist. Infolgedessen ist z.B. die Auflösung IP-Adresse zu MAC-Adresse bei IPv6 deutlich komplexer.

Des weiteren implementiert ICMPv6 Funktionen aus dem Bereich Autoconfiguration, der erst mit IPv6 eingeführt wurde. Hier gibt es Überschneidungen mit den Funktionen von DHCP. Deshalb muss auch das DHCP-Protokoll angepasst werden um mit IPv6 zusammen zu arbeiten. Analog zu ICMP heißt die IPv6 Variante des DHCP-Protokolls DHCPv6 ([RFC 3315]). Zum Beispiel beinhaltet ICMPv6 „Router Advertisements", mit deren Hilfe Client-Rechner automatisch eine globale (geroutete) IPv6-Adresse und eine Default-Route erhalten. Dies sind typische Eigenschaften, die bei IPv4 nur über DHCP zugewiesen werden (können).

3.2.3 Koexistenz von IPv4 und IPv6

Seit geraumer Zeit wird die Umstellung des (gesamten) Internet auf die Version 6 von IP beschworen; tatsächlich findet diese Umstellung jedoch nur dort statt, wo ein akuter Bedarf an freien IP-Adressen (z.B. in asiatischen Ländern) besteht, oder bestimmte Eigenschaften von IPv6 zwingend erforderlich werden. Daher ist für die nächste Zeit eine Koexistenz der beiden Protokollversionen abzusehen.

Um eine Kompatibilität zwischen IP-Netzen und -Hosts zu gewähren, obwohl diese mit verschiedenen IP-Versionen arbeiten, existieren mehrere prinzipielle Ansätze:

- Dual-Stack: Knoten besitzen sowohl IPv4- als auch IPv6-Implementierung.
- Tunneling: Kapselung von Paketen im tunnelnden Protokoll, d.h. IPv6-Verkehr über IPv4-Netz bzw. IPv4-Verkehr über IPv6-Netz.
  Herausforderungen: Management des Tunnels (Erstellung, Abbau), Umgang mit Fragmenten

- Übersetzung von IPv4/IPv6-PDU in die jeweils andere Protokollversion. Diese Übersetzung kann in Übergangsknoten/Router geschehen, bzw. in Programmbibliotheken (APIs zur Netzprogrammierung) des Betriebssystems.

3.3 Routing-Verfahren

In Netzen mit vielen Routern und einer großen Anzahl von Verbindungen in andere Netze ist es sehr aufwändig Routing-Tabellen manuell zu verwalten. Zur Erleichterung setzt man *Routing-Protokolle* ein, die Router den Austausch von Informationen ermöglicht, insbesondere Routen. Indem Router „Wissen" über erreichbare Ziele austauschen können Routing-Tabellen automatisch erstellt und aktualisiert werden.

Adaptive Routing-Verfahren zeichnen sich dadurch aus, dass sie sich dem Zustand des Netzes anpassen. Durch den Einsatz von Routing-Protokollen erhält ein Router ereignis- oder zeitgesteuert aktuelle Informationen über die Erreichbarkeit entfernter Netzsegmente. Router nutzen diese Informationen um die eigene Routing-Tabelle zu ergänzen oder zu modifizieren. Dadurch „lernen" Router Pfade in entfernte Netzsegmente und können auf Veränderungen in der Topologie reagieren. Unter den adaptiven Routing-Verfahren wird zwischen *zentralen* und *verteilten* Verfahren unterschieden. Bei zentralen Verfahren erhalten Router Informationen von fest definierten Stationen, während bei verteilten Routing-Verfahren die Router als gleichgestellte Einheiten (engl. peers) interagieren. Die im folgenden dargestellten Verfahren sind *verteilte* Verfahren und werden daher auf jedem Router ausgeführt. Heute verwenden die meisten Routing-Protokolle das adaptive *Distanz-Vektor Verfahren* oder das adaptive *Link-State Verfahren*.

3.3.1 Optimale Pfade

Die Hauptfunktion von Routing-Protokollen besteht darin, dass jeder Router Information über die ihm bekannten Subnetze anderen Routern mitteilt, so dass Rechner aus unterschiedlichen Subnetzen interagieren können. Dabei ist eine wichtige Herausforderung die Ermittlung eines *optimalen Pfades* zu jedem Subnetz. Mit einer Kostenfunktion werden die Transitkosten für jede Verbindung zwischen zwei benachbarten Routern festgelegt. Auf dem resultierenden Graphen mit gewichteten Kanten (wobei die Knoten des Graphen Router im Netz sind und die Kanten Leitungen zwischen den Routern) sucht ein Planungsalgorithmus einen optimalen Pfad zu jedem Subnetz.

Kostenfunktionen dienen dazu den Planungsalgorithmus zu beeinflussen. Eine Nachricht die sich im Transitnetz befindet belegt Ressourcen in Routern und Switches. Da diese Ressourcen meist sehr begrenzt sind, wird die Kostenfunktion häufig so gewählt, dass eine Nachricht so schnell wie möglich das Transitnetz verlässt. Eine Nachricht verlässt das Transitnetz entweder durch direkte Zustellung an den Adressaten oder indem sie in ein Netz eines anderen Betreibers weitergeleitet wird. Beispiele für Kostenfunktionen sind „Anzahl Hops", d.h. die Anzahl der Zwischenschritte, oder die summierte Latenz der Leitungen auf einem Pfad.

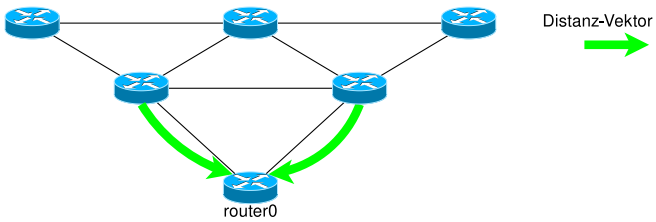


Abb. 3.5: Beim Distanz-Vektor Verfahren erhält router0 nur von den direkt benachbarten Routern Informationen

3.3.2 Distanz-Vektor Verfahren

Beim Distanz-Vektor Verfahren (DV) erstellt jeder Router eine Liste aller erreichbaren Ziele und die zugehörigen Kosten für den Transit. Indem benachbarte Router ihre Listen austauschen, „lernen" sie Pfade zu allen Zielen, die über ihre Nachbarn erreicht werden können. Der zugrunde liegende Algorithmus für das Distanz-Vektor Verfahren ist der Bellman-Ford oder Ford-Fulkerson Algorithmus.

Ein Router kennt und interagiert nur mit seinen direkten Nachbarn. Ein Ziel wird entweder direkt, oder über einen der Nachbarn erreicht. Die für das Verfahren benötigten Informationen über erreichbare Ziele erhält ein Router aus seiner Routing-Tabelle. Die Kosten werden über eine Kostenfunktion ermittelt. Ein einfaches Beispiel einer Kostenfunktion für das Distanz-Vektor Verfahren ist die Anzahl der Zwischenschritte (oder Entfernung, engl. distance), mit der ein Rechner im Zielnetz erreicht werden kann. Direkt erreichbare Netze erhalten einen Wert von 0 für die „Kosten" der Erreichbarkeit, da keine Zwischenschritte benötigt werden. Die Kosten der indirekt erreichbaren Ziele entspricht der länge des Pfades (Anzahl vermittelnder Router) zum Zielnetz.

Die vollständige Liste mit *Routing-Informationen* wird an alle direkten Nachbarn geschickt. Empfängt ein Router Routing-Informationen, vergleicht er deren Inhalt mit der eigenen Routing-Tabelle und den darin gespeicherten Kosten. Ziele, die noch keinen Eintrag in der eigenen Routing-Tabelle haben, werden ergänzt. Da die Informationen von einem direkten Nachbarn stammen, werden die Kosten (Distanz) um eins erhöht, bevor ein Eintrag in die eigene Routing-Tabelle eingefügt wird. Existiert bereits ein Eintrag in der Routing-Tabelle entscheidet der Router anhand der Kosten ob der Eintrag in der Routing-Tabelle ersetzt oder beibehalten wird. Bei dieser Entscheidung wird stets der Pfad mit weniger Zwischenschritten bevorzugt. Abbildung 3.5 zeigt ein Netz aus Routern, die ein Distanz-Vektor Verfahren einsetzen. In diesem Beispiel schicken nur die beiden direkt benachbarten Router Informationen an router0.

3.3.3 Link-State Verfahren

Ein Router, der das Link-State Verfahren einsetzt, berechnet optimale Pfade zu jedem Ziel, z.B. mit dem Dijkstra Algorithmus. Dazu ist es notwendig, dass jeder Router die gesamte Topologie kennt.

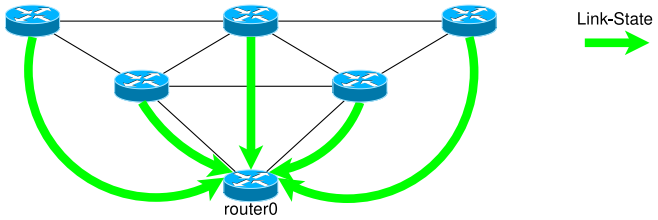


Abb. 3.6: Beim Link-State Verfahren erhält router0 von allen teilnehmenden Routern Informationen

Teilnehmende Router erstellen eine Liste mit allen direkt erreichbaren Zielen und mit allen benachbarten Routern. Mit einer Kostenfunktion werden die Kosten für den Transit zu benachbarten Routern bestimmt. Ein Beispiel für eine Kostenfunktion ist die *Round Trip Time* zwischen den Routern. Für die Vermittlung zu den Zielen werden keine Kosten festgelegt. Die gesammelten Informationen werden an alle anderen Router verteilt. Die hierzu ausgetauschten Nachrichten werden *link state advertisements* (LSA) genannt. Aus den LSAs erstellt jeder Router einen Graphen mit gewichteten Kanten und berechnet für sich die günstigsten Pfade zu allen Zielen.

Abbildung 3.6 zeigt ein Netz aus Routern, die ein Link-State Verfahren einsetzen. In diesem Beispiel schicken alle teilnehmenden Router Informationen an router0. Link-State Verfahren benötigen im Vergleich mit Distanz-Vektor Verfahren deutlich mehr Interaktion zwischen den Routern und Ressourcen zur Berechnung der optimalen Pfade. Mit diesen Nachteilen erhält man die Vorteile, dass Pfadberechnungen mit allen Informationen und unabhängig voneinander erfolgen. Dadurch sind mit Link-State Verfahren berechnete Pfade immer optimal, während beim Distanz-Vektor Verfahren Pfade iterativ, durch den Austausch von Routing-Informationen, zu optimalen Pfaden *konvergieren*.

3.3.4 Inter-AS-Routing

Zwischen Autonomen Systemen werden Wegewahlverfahren eingesetzt, die politische oder wirtschaftliche Interessen der Betreiber berücksichtigen. Standardmäßig wird dabei das Border Gateway Protocol ([RFC 4271]) eingesetzt, das mittels eines sogenannten Pfadvektor-Verfahrens (ähnlich dem Distanz-Vektor-Verfahren) Datenverkehr zwischen autonomen Systemen leitet.

Routing-Protokolle, die zwischen autonomen Systemen eingesetzt werden nennt man EGP (Exterior Gateway Protocol, BGP ist ein wichtiger Vertreter für EGP); sie ermitteln Pfade zwischen sogenannten Border Routern. Border Router unterscheiden sich dahingehend von „normalen" Routern, dass Border Router über mindestens eine logische Verbindung in ein anderes AS verfügen. Andere Router sind lediglich für die Vermittlung innerhalb eines AS zuständig.

Kosten-„Metriken" bei EGP sind meist monetär, z.B. EUR/Datenvolumen oder aber durch Regeln mit vertraglichem Hintergrund vorgegeben. Routing-Protokolle, die innerhalb

AS benutzt werden, nennt man IGP (Interior Gateway Protocol, z.B. RIP, OSPF). Die Border Router geben Informationen, die sie durch ein EGP gelernt haben mittels IGP an die inneren Router weiter. Ob und welche Routen wirklich zwischen EGP und IGP ausgetauscht werden bleibt der Konfiguration der Router überlassen.

Durch die Unterscheidung zwischen EGP und IGP entsteht eine Routing-Hierarchie. Während IGPs meist darauf ausgerichtet sind kürzeste Wege zu wählen, berücksichtigen EGPs andere, nicht-technische Aspekte.

3.3.5 FRRouting Suite

FRRouting (FRR) ist ein GPL-lizenziertes Software-Paket, das Implementierungen verschiedener Routingprotokolle, wie z.B. RIP, OSPF und BGP bereitstellt. Die Kernkomponente ist der *zebra*-Dienst, der als Schnittstelle zum Betriebssystem dient. Für jedes in FRR enthaltene Routing-Protokoll existiert ein eigenes Programm (daemon) etwa `ripd` für RIP, `ospfd` für OSPF usw.

Jeder Daemon bzw. Dienst kann über eine Kommandozeile (genannt „vty") administriert werden, die sich an der Bedienung anderer Routing-Programme orientiert. Damit nicht jeder Dienst separat verwaltet werden muss, existiert zusätzlich das Werkzeug "vtysh", über das alle Prozesse gemeinsam administriert werden können.

In der Datei /etc/frr/daemons konfiguriert man welche Dienste beim Start von FRR aktiviert werden.

```
zebra=yes
bgpd=no
ospfd=yes
ospf6d=no
ripd=no
ripngd=no
```

Um FRR und die damit verknüpften Dienste zu starten oder anzuhalten verwendet man den Befehl:

```
# /etc/init.d/frr <AKTION>
```

```
FRR starten:                          FRR anhalten:
# /etc/init.d/frr start               # /etc/init.d/frr stop
```

Eine ausführliche Dokumentation der FRRouting Suite, über die darin enthaltenen Dienste und deren Konfiguration, finden Sie unter [WWW 18].

vtysh

Nach dem Aufruf von `vtysh` befindet man sich auf der gemeinsamen Konsole aller aktivierten Routing-Dienste. Dort kann man sich jederzeit durch die Eingabe von `?` eine Liste der möglichen Befehle anzeigen lassen:

```
vtysh# ?
  enable   Turn on privileged mode command
  exit     Exit current mode and down to previous mode
  list     Print command list
```

```
  ping     Send echo messages
  quit     Exit current mode and down to previous mode
  show     Show running system information
  ...
```

Dies ist insbesondere bei teilweise eingegebenen Befehlen hilfreich. Wenn man bereits einen Teil des Befehls eingegeben hat und mit einem `?` (Fragezeichen) abschließt, werden alle weiteren Optionen angezeigt, die statt dem `?` stehen können:

```
vtysh# show ip ?
  ...
  bgp      BGP information
  ospf     OSPF information
  rip      Show RIP routes
  ...
```

Es gibt verschiedene VTY Modi. Der *View-Modus* erlaubt nur lesenden Zugriff auf die Konfiguration des Routers, während der *Enable-Modus* sowohl lesenden als auch schreibenden Zugriff erlaubt. Nach dem Aufruf von `vtysh` befindet man sich im View-Modus. Mit dem Kommando `enable` wechselt man in den Enable-Modus. Mit `disable` kehrt man zurück in den View-Modus. Möchten Sie die Konfiguration anpassen, so müssen Sie zunächst in den *Configuration Mode* wechseln.

In den *Configuration Mode* gelangen Sie durch den Aufruf von `configure terminal`. Nun stehen Ihnen weitere Befehle zur Verfügung. Beispielsweise können Sie jetzt mit `interface eth1` in den Konfigurationsmodus für das Interface eth1 wechseln und dort wiederum z.B. die IP-Adresse ändern. Mit dem Befehl `exit` verlassen Sie den aktuellen Modus und kehren zum vorherigen zurück. Die Wirkung von Befehlen, die eine Funktion aktivieren oder deaktivieren, können Sie durch Voranstellen von `no` umkehren.

Die gesamte Abfolge um z.B. die IP-Adresse von Interface eth1 zu ändern sieht so aus:

```
(1) router1# vtysh
(2) router1-vtysh# enable
(3) router1-vtysh# configure terminal
(4) router1-vtysh(config)# interface eth1
(5) router1-vtysh(config-if)# ip address 10.0.0.1/8
(6) router1-vtysh(config-if)# exit
(7) router1-vtysh(config)# exit
(8) router1-vtysh# exit
```

Erklärung: (1) `vtysh` starten, (2) Enable Mode aktivieren (nur nötig, falls noch nicht aktiviert), (3) Konfigurationsmodus aktivieren, (4) Konfigurationsmodus von Interface eth1 aufrufen, (5) IP-Adresse ändern, (6) Interface-Konfigurationsmodus verlassen, (7) Konfigurationsmodus deaktivierten, (8) `vtysh` beenden

Neben der interaktiven Konfiguration über `vtysh` können sämtliche Einstellungen auch in Konfigurationsdateien verwaltet werden. Diese befinden sich standardmäßig im Verzeichnis `/etc/frr/` und werden mit `<daemonname>.conf` benannt, also z.B. `ripd.conf`. Wenn Sie FRRouting über `vtysh` konfiguriert haben, können Sie mit dem Befehl `write file` die entsprechenden Konfigurationsdateien automatisch erstellen lassen.

ripd

Die meisten Befehle zur Konfiguration des `ripd` erreichen Sie vom Konfigurationsmodus aus durch den Befehl `router rip`. Als minimale Konfiguration wird nur eine Angabe benötigt, auf welchen Interfaces RIP aktiviert werden soll. Dies geschieht durch den Befehl:
`network <IFNAME>|<IP-PREFIX>`
Bei diesem Befehl geben Sie entweder ein Interface (z.B. `eth1`) oder ein IP-Prefix (z.B. `10.6.0.0/16`) an. Letzteres bezieht alle Interfaces mit ein, die sich im angegebenen Adressbereich befinden.

ospfd

Die Basiskonfiguration des `ospfd` verläuft analog zum `ripd`. Die Befehle lauten hier `router ospf` und `network <IP-PREFIX> area <AREA>`. Bei `<AREA>` reicht es für unsere Zwecke, wenn Sie `0` angeben.

bgpd

Um den `bgpd` zu administrieren, geben Sie im Konfigurationsmodus den Befehl `router bgp <AS-NUMMER>` ein, wobei Sie als AS-Nummer die Nummer Ihres autonomen Systems ersetzen. Nun können Sie mit dem Befehl `neighbor <IP-ADRESSE> remote-as <AS-NUMMER>` Border-Router anderer autonomer Systeme als Nachbarn hinzufügen. Zusätzlich sollten Sie mit dem Befehl `network <IP-PREFIX>` angeben, für welches Netz Ihr Router als Border-Router fungiert. (Der `network` Befehl hat hier also eine andere Semantik!) Mit `neighbor <IP-ADRESSE> weight <GEWICHT>` können Sie einem Nachbarn ein Gewicht zuordnen, wobei höher gewichtete Nachbarn beim Routing bevorzugt werden.

3.4 Aufgaben

A300 Fragmentierung und IP-Tunneling (Theorie)

i) In wie viele Fragmente wird ein IP-Paket mit Größe 9000 Byte zerlegt, um über ein Ethernet mit MTU = 1500 Byte übertragen zu werden? Geben Sie eine vollständige Rechnung an!

ii) Nennen Sie drei Gründe dafür, dass Netzbetreiber IP-Fragmentierung in ihren Netzen verbieten. Erläutern Sie diese Gründe!

iii) Wie wird in modernen TCP/IP-Implementierungen dafür gesorgt, dass Fragmentierung in der Regel nicht erforderlich ist?

A301 IPv6 (Theorie)

i) Erklären Sie anhand eines Beispiels auf einer Ihrer VMs wie link-local Adressen aus der MAC-Adresse abgeleitet werden!

ii) Wie implementiert IPv6 „Broadcasts"?

iii) Welches sind die privaten Adressbereiche in IPv6, analog zu 10.0.0.0/8, 172.16.0.0/12 und 192.168.0.0/16 in IPv4?

iv) Für besondere Zwecke, außer für den privaten Gebrauch, sind noch weitere Bereiche reserviert. Wie teilt sich der IPv6 Addressraum auf? *Hinweis:* IANA, ignorieren Sie die vom IETF reservierten Bereiche

A302 Fragmentierung und Tunneling

i) Legen Sie den Pfad (pc1, router1, router4, router3, router2, pc2) an, so dass Nachrichten zwischen pc1 und pc2 vermittelt werden und zeigen Sie mittels `traceroute`, dass die Nachrichten entlang dieses Pfades vermittelt werden!

ii) Bestimmen Sie mittels `ip` die MTU der genutzten Schnittstellen! Erstellen Sie für die Ausarbeitung eine Tabelle mit den Spalten: Rechnername, Schnittstelle, IP-Adresse, MTU.

iii) Beschränken Sie nun mittels `ip` die MTU von router3.eth3 auf 1000 Bytes und die MTU von router4.eth3 auf 1100 Bytes!

Beschreiben Sie das Verhalten der Rechner, wenn Sie IP-Nachrichten mit 1200 Bytes Nutzdaten von pc1 an pc2 schicken! Nutzen sie `ping` um Nachrichten dieser Größe zu erzeugen. Achten Sie darauf, dass `ping` das *Don't Fragment* Flag setzt.

iv) Wiederholen Sie den `ping`-Versuch ohne gesetztes *Don't Fragment* Flag. Wie verändert sich der Datenfluss im Vergleich zum vorherigen Versuch?

v) Weisen Sie dem Interface router3.eth3 eine beliebige IPv6 Adresse zu. Ist der Vorgang erfolgreich? Wenn nein, was ist die Ursache? Ziehen sie hierzu auch die beschriebenen Rahmenbedingungen aus [RFC 2460] in Betracht.

vi) Konfigurieren Sie einen IP-Tunnel zwischen router1 und router2 und konfigurieren Sie die Systeme so, dass sämtliche IP-Nachrichten zwischen pc1 und pc2 durch den Tunnel übertragen werden!

Hinweis: Benutzen Sie einen GRE-Tunnel, den Sie mit dem Programm `ip` einrichten. Hintergrundinformation zu GRE (Generic Routing Encapsulation) lesen Sie bitte in [RFC 2784] vor Durchführung des Versuchs nach; zur praktischen Nutzung finden Sie die Dokumentation in der Manpage `ip-tunnel (8)`.

vii) Zeigen Sie mit `traceroute`, dass alle Nachrichten durch den Tunnel übertragen werden!

viii) Vergleichen Sie die von `traceroute` angegebenen Zwischenstationen aus den vorangegangenen Versuchen. Nennen Sie alle Unterschiede!

ix) (Freiwillig) Fangen Sie mit Hilfe von `tcpdump` ein Paket ab, das den Tunnel gerade „betritt" bzw. „verläßt" und analysieren Sie die gekapselte Headerstruktur. Welche Unterschiede zwischen den Header-Daten des inneren versus denen des äußeren IP-Paketes stellen Sie fest?

Hinweis: Beim Arbeiten mit `tcpdump` kann es nötig sein den Detailgrad der Ausgabe (`-v`) oder die Mitschnitt-Größe der Pakete (`-s`) anzupassen. Für eine komfortablere Auswertung kann man zunächst die Rohdaten in eine Datei umleiten (`-w`). In diesem Fall erzeugt `tcpdump` keine Ausgaben mehr. Die Datei mit Rohdaten können Sie anschließend mit Wireshark auswerten.

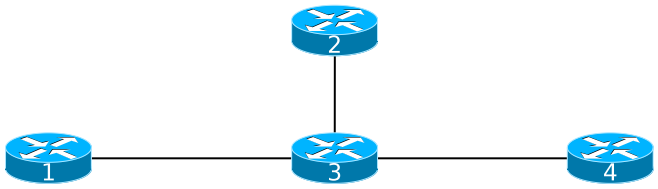


Abb. 3.7: Router 1 bis 4 mit kreisfreier Sterntopolgie

A303 IPv6

i) Entfernen Sie alle IPv4-Adressen von den Schnittstellen eth{1,2,3,4} auf allen Routern!

ii) Bilden Sie das Netz aus Abblidung 3.7 nach! Verwenden Sie DHCPv6-PD (prefix delegation) um ihr Netz hierarchisch aufzuteilen mit router3 als Zentrum. (*Hinweis:* `/etc/config/{network,dhcp}`, siehe OpenWrt-Dokumentation[1]) Beachten Sie, dass ...

   1) Sie IPv6-Adressen aus den Bereichen 2001:db8:⟨Gruppennummer⟩::/48 und fc00:dead:beef:⟨Gruppennummer⟩::/52 verwenden! (Die ersten 48 Bits sind gesetzt, Bits 48 bis 52 ersetzen Sie durch Ihre Gruppennummer, die restlichen Bits stehen zu Ihrer freien Verfügung.)
   2) Die Router IPv6-Pakete zwischen ihren Schnittstellen vermitteln!
   3) `ping`-Anfragen zwischen allen Routern vermittelt werden! Fügen Sie die Routing-Tabellen Ihrer Router in Ihre Ausarbeitung ein.

iii) Konfigurieren Sie Ihre Router so, dass die PCs automatisch IPv6-Adressen mit Ihrem Präfix erhalten. Erläutern Sie anhand von geeigneten Aufzeichnungen den Vorgang wie pc1 und pc2 eine globale IPv6-Adresse erhalten! Betrachten Sie auch die Kommunikation zwischen router3 und router{1,2}.

iv) Passen Sie die Routing-Konfigurationen von router1, router2 und router3 an, so dass IPv6-Pakete zwischen pc1 und pc2 vermittelt werden. Sind manuelle Änderungen daran notwendig? Zeigen Sie mittels `traceroute6` (8) den Pfad von IPv6-Paketen zwischen pc1 und pc2!

v) Konfigurieren Sie einen ip4ip6-Tunnel zwischen router1 und router2, so dass IPv4-Pakete von pc1 an pc2 (und zurück!) per IPv6 zwischen den Routern vermittelt werden! Zeigen Sie, dass Ihre Konfiguration funktioniert und IPv4-Pakete tatsächlich in IPv6 gekapselt werden.

[1] `https://openwrt.org/docs/guide-user/network/ipv6/configuration`

A304 Distanz-Vektor Routing mit RIP

Installieren Sie die FRRouting-Suite auf router1 bis 4. (*Hinweis:* Auf Debian: frr, auf OpenWrt: frr frr-zebra frr-watchfrr frr-staticd frr-vtysh frr-ripd frr-ospfd frr-bgpd)

i) Vernetzen Sie die Router wie folgt: router1–router2–router4–router3, deaktivieren Sie alle anderen Verbindungen zwischen den Routern! Vergeben Sie passende Subnetze und IP-Adressen aus Ihrem Gruppennetz 10.⟨*Gruppennummer*⟩.0.0/16, legen Sie keine manuellen Routen an!

ii) Erstellen Sie die Konfigurationsdatei `/etc/frr/ripd.conf`! Die Syntax entnehmen Sie der Dokumentation [2]. *Hinweis:* Auf Debian können Sie auch Beispielkonfigurationen in `/usr/share/doc/frr/examples` finden. Zeilen, die mit `!` anfangen, sind Kommentare.

iii) Passen Sie die Datei `/etc/frr/daemons` an und starten Sie die Dienste `zebra` und `ripd` nacheinander auf Ihren Routern! (`/etc/init.d/frr start`) *Hinweis:* starten Sie zuerst beide Dienste auf router1, dann auf router2, usw. usf.)

iv) Beobachten Sie auf router1.eth2 die ein- und ausgehenden RIP-PDUs! Zeigen Sie die Zwischenschritte in denen router1 sein Wissen über die Infrastruktur vervollständigt!

v) Machen Sie sich mit `vtysh` vertraut. Benutzen Sie `vtysh` um sich die Routing-Tabelle für router1 anzeigen zu lassen! Welche Bedeutung hat das Feld „From"?

vi) Aktivieren Sie die Verbindung router1-router3 und zeigen Sie die Zwischenschritte, bis sich die Routingtabellen von router2 und router4 an den neuen Zustand angepasst haben!

vii) Deaktivieren Sie router1.eth1, um einen Verbindungsausfall zu simulieren! Beobachten Sie das Verhalten der Router!

   a) Welches Verhalten der Router ist nach [RFC 1058] zu erwarten, um das über router1.eth1 erreichbare Subnetz aus den Routing-Tabellen zu entfernen?

   b) Welches Verhalten der Router ist tatsächlich zu beobachten? Welche Technik wird eingesetzt um das nun nicht mehr erreichbare Subnetz aus den Routing-Tabellen zu entfernen? Zeigen Sie die entsprechenden PDUs der Router!

[2] `https://docs.frrouting.org/en/latest/ripd.html`

A305 Link-State Routing mit OSPF

Im Folgenden wiederholen Sie den Versuch von oben. Anstatt RIP wird diesmal OSPF eingesetzt.

i) Stellen Sie den Zustand aus A304, i) wieder her. Deaktivieren Sie FRR auf allen Routern!

ii) Erstellen Sie die Konfigurationsdatei `/etc/frr/ospfd.conf`! Die Syntax entnehmen Sie der Dokumentation [3].

iii) Passen Sie die Datei `/etc/frr/daemons` an und starten Sie `zebra` und `ospfd` nacheinander! *Hinweis:* zuerst beide Dienste auf router1, dann router2, usw. usf.

iv) Beobachten Sie die ein- und ausgehenden OSPF-PDUs auf router1.eth2! Beschreiben Sie die Zwischenschritte, in denen router1 sein Wissen über die Infrastruktur vervollständigt!

v) Aktivieren Sie die Verbindung router1-router3. Beschreiben Sie die Zwischenschritte, bis sich die Routingtabellen von router2 und router4 an den neuen Zustand angepasst haben! Belegen Sie Ihre Erklärung mit entsprechenden Programmausgaben!

vi) Deaktivieren Sie router1.eth1, um einen Verbindungsausfall zu simulieren! Erläutern Sie den Unterschied zwischen RIP und OSPF in der Art und Weise wie der Verbindungsausfall an andere Router propagiert wird!

[3] `https://docs.frrouting.org/en/latest/ospfd.html`

A306 Autonome Systeme und BGP

Neben den Interior Gateway Protokollen wollen wir uns BGP als Exterior Gateway Protokoll ansehen. Behalten Sie Ihre OSPF-Konfiguration aus der vorherigen Aufgabe bei.

i) Stellen Sie sicher, dass Sie Adressen aus Ihrem Subnet (10.⟨*Gruppe*⟩.0.0/16) verwenden. Ihre Infrastruktur bildet das Autonome System mit der Nummer 65000 + ⟨*Gruppe*⟩.

ii) Für diese Aufgabe werden vom Lehrstuhl zwei weitere Maschinen bereitgestellt, Gruppe11 (Odd, AS 65011) und Gruppe12 (Even, AS 65012). Auf diesen Maschinen ist BGP bereits konfiguriert. Abhängig von ihrer Gruppennummer können Sie allerdings nur mit einer der Maschinen direkt kommunizieren. Ihr Ziel ist es mittels BGP eine Route in das Subnet der anderen Maschine zu bekommen.

   Stellen Sie mit Hilfe eines GRE-Tunnels eine Verbindung zu der Maschine her. Verwenden Sie als Endpunkte die Adresse von router1 an eth0 und die globale IP-Adresse der Zielmaschine. Achten Sie darauf, dass eine TTL > 3 verwendet wird. *Hinweis:* Sie können auf die selbe Weise eine Verbindung zu anderen Gruppen herstellen.

iii) Konfigurieren Sie Router1 als Border Router Ihres Autonomen Systems. Vermitteln Sie mit dem Border Router am anderen Ende des Tunnels. Dieser hat die IP-Adresse 10.11.0.⟨*Ihre Gruppe*⟩ bzw. 10.12.0.⟨*Ihre Gruppe*⟩. *Hinweis:* Definieren Sie Host-Routen.

iv) Passen Sie die OSPF-Konfiguration von Ihren Routern so an, dass die über BGP gelernten Routen auch den anderen Routern im jeweiligen Netz mitgeteilt werden. *Hinweis:* Sie können dazu entweder mit `vtysh` arbeiten, oder die Datei `/etc/frr/ospfd.conf` anpassen. Die Befehle hierzu finden sich in der Dokumentation zur FRR-Suite.

v) Zeigen Sie mit `traceroute` den Weg einer Nachricht von Ihrem PC3 zu 10.11.2.100 und 10.12.2.100. Über welche AS und welche Router wird die Nachricht vermittelt?

vi) Angenommen, der Lehrstuhl verlangt hohe Gebühren für den Transit Ihrer Nachrichten. Konfigurieren Sie Ihren Router1 so, dass über dieses AS nur vermittelt wird, wenn keine andere Verbindung zur Verfügung steht.
   Optional: Weisen Sie nach, das Ihre Konfiguration funktioniert, indem Sie mit den andern Gruppen kommunizieren und den Transit über diese abwickeln.

vii) Konfigurieren Sie nun Router4 anstatt Router1 als BGP-Router Ihres AS, wobei die externen AS weiterhin an Router1 angeschlossen bleiben.
   Erläutern Sie anhand Ihrer Beobachtungen, inwiefern sich die beiden Szenarien unterscheiden.
   Theorie: Nennen Sie mindestens 2 Gründe, die für ein solches Szenario sprechen würden.

◇

Literatur

[RFC 1058] HEDRICK, C.L.: *Routing Information Protocol*. RFC 1058 (Historic), Juni 1988, http://www.ietf.org/rfc/rfc1058.txt . Updated by RFCs 1388, 1723.

[RFC 1519] FULLER, V., T. LI, J. YU und K. VARADHAN: *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*. RFC 1519 (Proposed Standard), September 1993, http://www.ietf.org/rfc/rfc1519.txt . Obsoleted by RFC 4632.

[RFC 1881] IAB und IESG: *IPv6 Address Allocation Management*. RFC 1881 (Informational), Dezember 1995, http://www.ietf.org/rfc/rfc1881.txt .

[RFC 2401] KENT, S. und R. ATKINSON: *Security Architecture for the Internet Protocol*. RFC 2401 (Proposed Standard), November 1998, http://www.ietf.org/rfc/rfc2401.txt . Obsoleted by RFC 4301, updated by RFC 3168.

[RFC 2460] DEERING, S. und R. HINDEN: *Internet Protocol, Version 6 (IPv6) Specification*. RFC 2460 (Draft Standard), Dezember 1998, http://www.ietf.org/rfc/rfc2460.txt . Updated by RFC 5095.

[RFC 2784] FARINACCI, D., T. LI, S. HANKS, D. MEYER und P. TRAINA: *Generic Routing Encapsulation (GRE)*. RFC 2784 (Proposed Standard), März 2000, http://www.ietf.org/rfc/rfc2784.txt . Updated by RFC 2890.

[RFC 3315] DROMS, R., J. BOUND, B. VOLZ, T. LEMON, C. PERKINS und M. CARNEY: *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*. RFC 3315 (Proposed Standard), Juli 2003, http://www.ietf.org/rfc/rfc3315.txt . Updated by RFCs 4361, 5494.

[RFC 4271] REKHTER, Y., T. LI und S. HARES: *A Border Gateway Protocol 4 (BGP-4)*. RFC 4271 (Draft Standard), Januar 2006, http://www.ietf.org/rfc/rfc4271.txt .

[RFC 4361] LEMON, T. und B. SOMMERFELD: *Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4)*. RFC 4361 (Proposed Standard), Februar 2006, http://www.ietf.org/rfc/rfc4361.txt . Updated by RFC 5494.

[RFC 4443] CONTA, A., S. DEERING und M. GUPTA: *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*. RFC 4443 (Draft Standard), März 2006, http://www.ietf.org/rfc/rfc4443.txt . Updated by RFC 4884.

[RFC 4884] BONICA, R., D. GAN, D. TAPPAN und C. PIGNATARO: *Extended ICMP to Support Multi-Part Messages*. RFC 4884 (Proposed Standard), April 2007, http://www.ietf.org/rfc/rfc4884.txt .

[RFC 5095] ABLEY, J., P. SAVOLA und G. NEVILLE-NEIL: *Deprecation of Type 0 Routing Headers in IPv6*. RFC 5095 (Proposed Standard), Dezember 2007, http://www.ietf.org/rfc/rfc5095.txt .

[RFC 5398] HUSTON, G.: *Autonomous System (AS) Number Reservation for Documentation Use*. RFC 5398 (Informational), Dezember 2008, http://www.ietf.org/rfc/rfc5398.txt .

[RFC 5494] ARKKO, J. und C. PIGNATARO: *IANA Allocation Guidelines for the Address Resolution Protocol (ARP)*. RFC 5494 (Proposed Standard), April 2009, http://www.ietf.org/rfc/rfc5494.txt .

[RFC 791] POSTEL, J.: *Internet Protocol*. RFC 791 (Standard), September 1981, http://www.ietf.org/rfc/rfc791.txt . Updated by RFC 1349.

[WWW 18] KUNIHIRO ISHIGURO, ET AL.: *FRRouting Documentation*, Dezember 2018, http://docs.frrouting.org/en/latest/ .