Grundlagen: Rechnernetze und verteilte Systeme

Achte Woche: 11./15. Juni 2018

IPv6, NDP, SLAAC, Distanz-Vektor-Routing

Leo Glavinić
netze@eo.gl
eo.gl/netze

Inhalt

1. Distanz-Vektor-Routing (Mehr Tabellen!)

2. NDP, IPv6 (Mehr Protokollheader!)

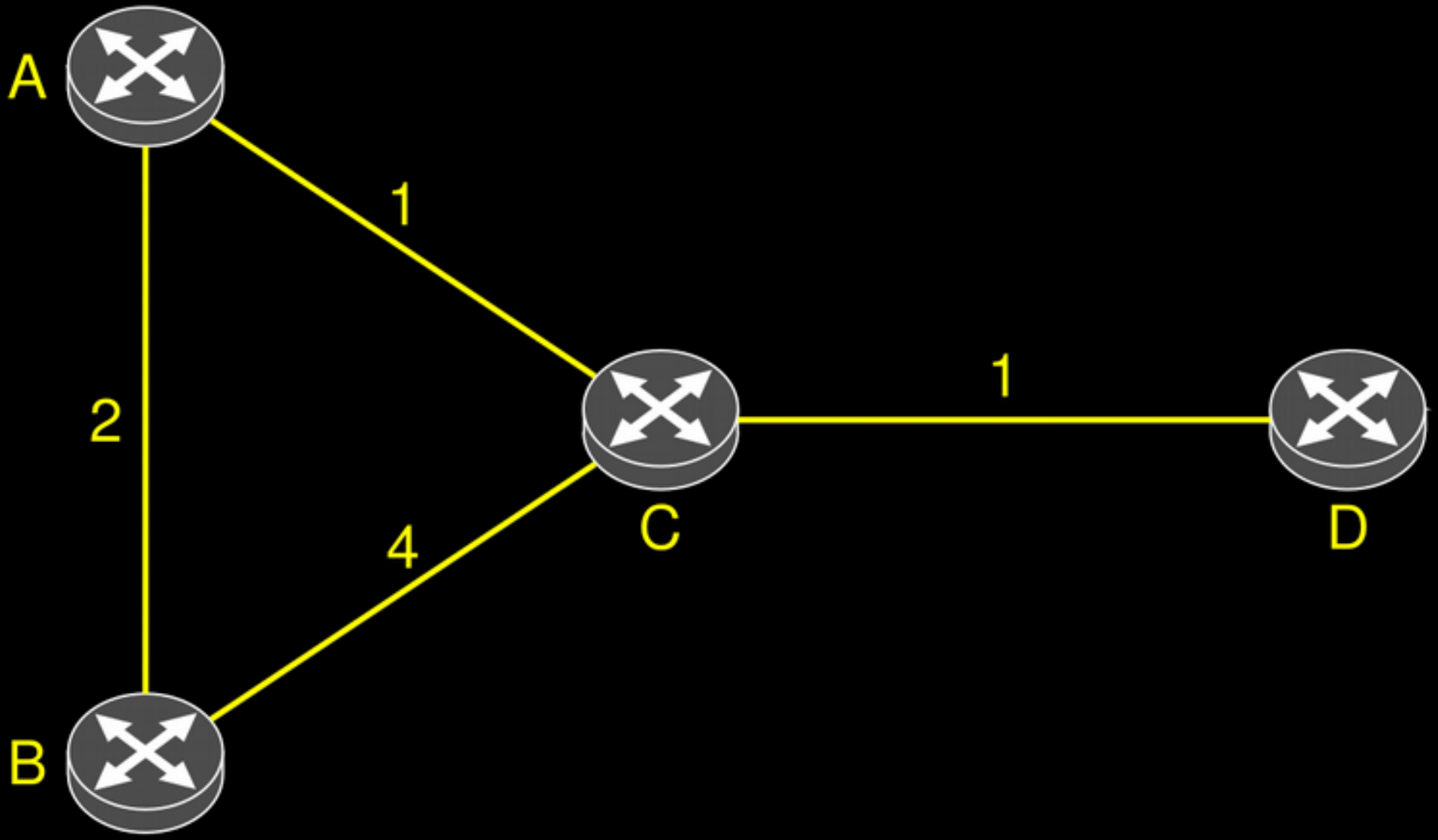
2. Distanz-Vektor-Routing

Vier Router, angegebene Link-(Zeit-, Weg-, ...)kosten per Kante

Notation der Routingtabellen: $[(x_A, y_A), ..., (x_D, y_D)]$ mit Kosten als x und Next Hop als y

Am Anfang: leere Routingtabellen (/ , /); Router erreicht sich selbst natürlich mit Kosten 0

2. Distanz-Vektor-Routing



2. Distanz-Vektor-Routing

Gegenseitiger Austausch der Vektoren mit direkten Nachbarn

Bei Fund einer kürzeren Route: Aktualisierung des eigenen Vektors

Gleichzeitige Updates zwecks Vereinfachung dieser Aufgabe

2. Distanz-Vektor-Routing

a. Routingtabellen nach jeweiligen Updates bis zum konvergenten Zustand

Schritt	Router A	Router B	Router C	Router D
0	[(0,A) (/,/) (/,/) (/,/)]	[(/,/) (0,B) (/,/) (/,/)]	[(/,/) (/,/) (0,C) (/,/)]	[(/,/) (/,/) (/,/) (0,D)]

2. Distanz-Vektor-Routing

a. Routingtabellen nach jeweiligen Updates bis zum konvergenten Zustand

Schritt	Router A	Router B	Router C	Router D
0	[(0,A) (/,/) (/,/) (/,/)]	[(/,/) (0,B) (/,/) (/,/)]	[(/,/) (/,/) (0,C) (/,/)]	[(/,/) (/,/) (/,/) (0,D)]
1	[(0,A) (2,B) (1,C) (/,/)]	[(2,A) (0,B) (4,C) (/,/)]	[(1,A) (4,B) (0,C) (1,D)]	[(/,/) (/,/) (1,C) (0,D)]
2	[(0,A) (2,B) (1,C) (2,C)]	[(2,A) (0,B) (3,A) (5,C)]	[(1,A) (3,A) (0,C) (1,D)]	[(2,C) (5,C) (1,C) (0,D)]
3	[(0,A) (2,B) (1,C) (2,C)]	[(2,A) (0,B) (3,A) (4,A)]	[(1,A) (3,A) (0,C) (1,D)]	[(2,C) (4,C) (1,C) (0,D)]

2. Distanz-Vektor-Routing

b. Graphenalgorithmus

Bellman-Ford-Algorithmus (dezentrale/verteilte Implementierung)

2. Distanz-Vektor-Routing

c. Verbindung zwischen C und D entfällt, entsprechende Pfadkosten werden auf unendlich gesetzt; was passiert weiterhin?

Schritt	Router A	Router B	Router C	Router D
4	[(0,A) (2,B) (1,C) (2,C)]	[(2,A) (0,B) (3,A) (4,A)]	[(1,A) (3,A) (0,C) (/,/)]	[(/,/) (/,/) (/,/) (0,D)]
5	[(0,A) (2,B) (1,C) (6,B)]	[(2,A) (0,B) (3,A) (4,A)]	[(1,A) (3,A) (0,C) (3,A)]	[(/,/) (/,/) (/,/) (0,D)]
6	[(0,A) (2,B) (1,C) (4,C)]	[(2,A) (0,B) (3,A) (7,C)]	[(1,A) (3,A) (0,C) (7,A)]	[(/,/) (/,/) (/,/) (0,D)]
7	[(0,A) (2,B) (1,C) (8,C)]	[(2,A) (0,B) (3,A) (6,A)]	[(1,A) (3,A) (0,C) (5,A)]	[(/,/) (/,/) (/,/) (0,D)]
8	[(0,A) (2,B) (1,C) (6,C)]	[(2,A) (0,B) (3,A) (9,C)]	[(1,A) (3,A) (0,C) (9,A)]	[(/,/) (/,/) (/,/) (0,D)]
9	[(0,A) (2,B) (1,C) (10,C)]	[(2,A) (0,B) (3,A) (8,A)]	[(1,A) (3,A) (0,C) (7,A)]	[(/,/) (/,/) (/,/) (0,D)]
10	[(0,A) (2,B) (1,C) (8,C)]	[(2,A) (0,B) (3,A) (11,C)]	[(1,A) (3,A) (0,C) (11,A)]	[(/,/) (/,/) (/,/) (0,D)]
⋮	⋮	⋮	⋮	⋮

→ Count to Infinity

2. Distanz-Vektor-Routing

d. Maßnahmen gegen Count to Infinity

Split Horizon: „Bewerbe eine Route nicht über das Interface, über das sie ursprünglich gelernt wurde“; hier: A und B senden keine Updates zur Route nach D an C; keine Lösung bei ringförmiger Topologie

2. Distanz-Vektor-Routing

d. Maßnahmen gegen Count to Infinity

Triggered Update: sofortige Sendung von Updates, sobald Änderungen der Topologie erkannt werden (statt periodischer Abstände); keine Lösung des CtI-Problems, aber Beschleunigung; Nachteil: kurzzeitig viel Datenverkehr durch Updates; von den meisten Routingprotokollen unterstützt

2. Distanz-Vektor-Routing

d. Maßnahmen gegen Count to Infinity

Path Vector: „Router merken sich, woher das Update kam“, Information wird in Updates inkludiert → Möglichkeit zur Wiederherstellung des gesamten Pfades (statt nur Next Hop); entdeckt ein Router sich selbst darin (Schleife), wird Update verworfen; Einsatz in Routingprotokoll BGP

1. NDP und IPv6-Fragmentierung

siehe Blatt 6, Aufgabe 2 für ARP und IPv4

SLAAC (Stateless Address Autoconfiguration): automatische Zuweisung von Link-Local-IPv6-Adressen auf Basis von MAC-Adressen

1. Präfix fe80::
2. erste drei Oktette der MAC-Adresse (OUI) mit invertiertem siebtem bit (von „links“)
3. Padding mit ff:fe
4. letzte drei Oktette der MAC-Adresse

1. NDP und IPv6-Fragmentierung

Ähnliche Generierung von Global-Unique-IPv6-Adressen: Präfix des jeweiligen Routers (durch Router Advertisements erkannt) statt fe80::

1. NDP und IPv6-Fragmentierung

NDP (Neighbour Discovery Protocol): Teil von ICMPv6, genutzt u.a. zur Adressauflösung

Analog zu ARP: Neighbour Solicitation → Neighbour Advertisement

Neigbour Solicitation wird an Solicited-Node-Adresse geschickt (Multicast); Präfix ff02::1:ff00:0/104 und letzte 24 bit der ursprünglichen IPv6-Adresse

Mapping auf MAC: erste zwei Oktette auf 33:33, Rest entspricht letzten vier Oktetten der Multicast-IPv6

1. NDP und IPv6-Fragmentierung

1. NDP und IPv6-Fragmentierung

a. Link-Local-Adressen aller Interfaces

PC1, af:fe:14:af:fe:20 → fe80::adfe:14ff:feaf:fe20
R1.eth0, af:fe:14:af:fe:21 → fe80::adfe:14ff:feaf:fe21
R1.eth1, af:fe:14:af:fe:22 → fe80::adfe:14ff:feaf:fe22
R2.eth1, af:fe:14:af:fe:23 → fe80::adfe:14ff:feaf:fe23
R2.eth0, af:fe:14:af:fe:24 → fe80::adfe:14ff:feaf:fe24
PC2, af:fe:14:af:fe:25 → fe80::adfe:14ff:feaf:fe25

1. NDP und IPv6-Fragmentierung

Router seien jeweils mit Präfices 2001:db8:1::/64 bzw. 2001:db8:2::/64 konfiguriert

b. Global-Unique-Adressen von PC1 und PC2

af:fe:14:af:fe:20 → 2001:db8:1:0:adfe:14ff:feaf:fe20
af:fe:14:af:fe:25 → 2001:db8:2:0:adfe:14ff:feaf:fe25

1. NDP und IPv6-Fragmentierung

1400 B Nutzdaten; MTU zwischen Routern sei 1280 B, in den lokalen Netzwerken 1500 B

c. Ort der Fragmentierung

Direkt an PC1, weil in IPv6 Router nicht fragmentieren

PC1 sendet zunächst ganzes Paket an R1 und erhält „Packet too big"-ICMPv6-Nachricht zurück

1. NDP und IPv6-Fragmentierung

d. Mindestanzahl der Fragmente

MTU: maximale Größe des gesamten L3-Paketes (=der L2-Payload)

Bei Fragmentierung: ein IPv6-Header (40 B) und Fragment Header (8 B) pro Paket

$$N = \left\lceil \frac{1400\,\text{B}}{1280\,\text{B} - 40\,\text{B} - 8\,\text{B}} \right\rceil = 2$$

1. NDP und IPv6-Fragmentierung

e. Größe der Schicht-3-SDU für jedes Fragment

Fragment Offset im Header ist in Vielfachen von 8 B gegeben → zwangsläufige Teilbarkeit der Nutzdatenlänge in B durch acht

Maximale Nutzdaten: 1280 B-40 B-8 B=1232 B (ist durch acht teilbar)

Erstes Fragment: 1232 B Payload; Zweites Fragment: 168 B Payload

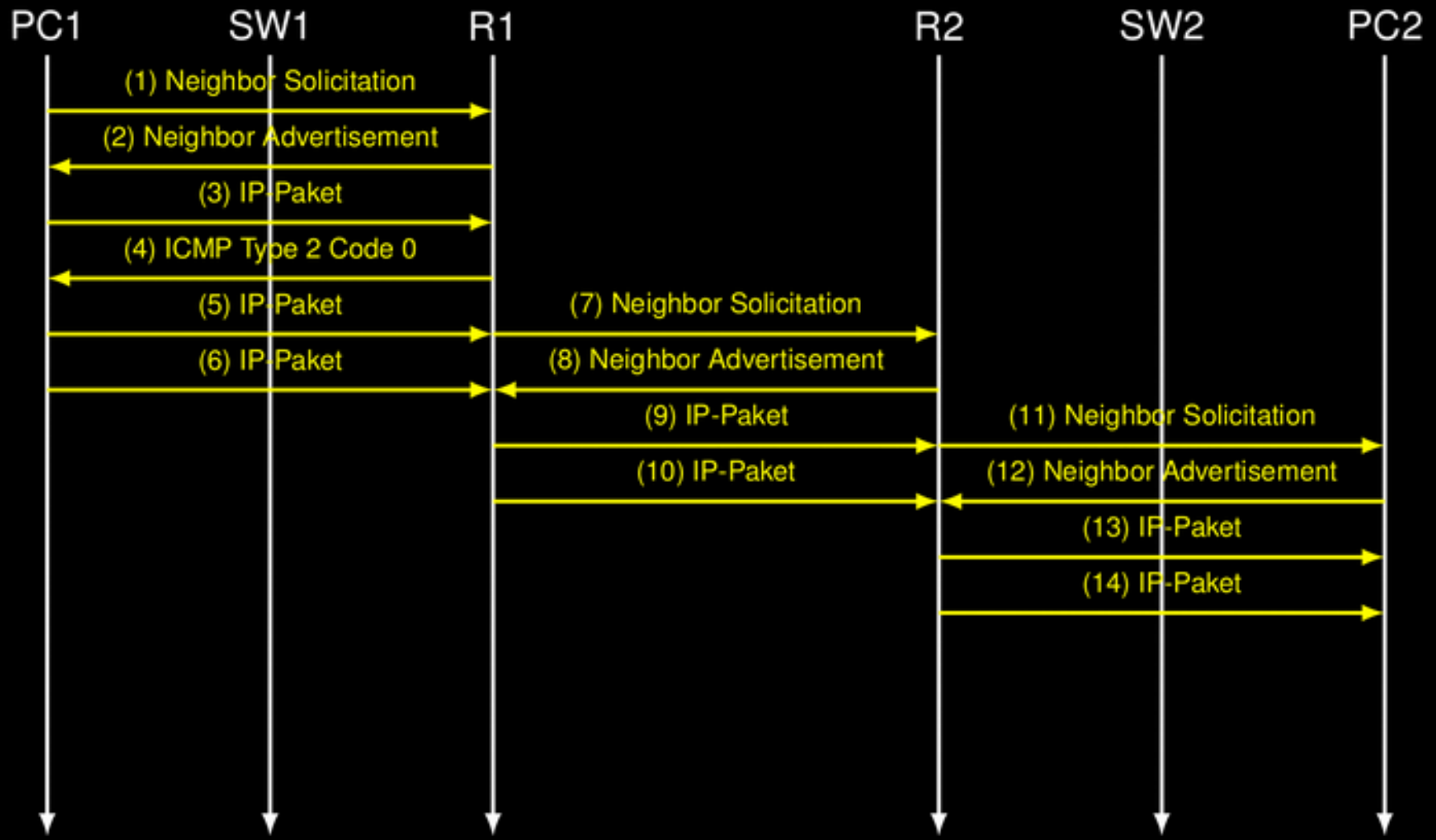
1. NDP und IPv6-Fragmentierung

f. Ort der Reassemblierung

PC2, weil Fragmente unabhängig voneinander geroutet werden

1. NDP und IPv6-Fragmentierung

g. Weg-Zeit-Diagramm aller Rahmen



1. NDP und IPv6-Fragmentierung

h. Destination-MAC-Adresse des ersten Rahmens

Solicited-Node-Adresse (siehe vorherige Folien)

→ ff02::1:ffaf:fe21 (IPv6)
→ Multicast-MAC-Adresse 33:33:ff:af:fe:21

1. NDP und IPv6-Fragmentierung

g. Header! (siehe Vordrucke)