LUND UNIVERSITY

EITF45
Internet Routing

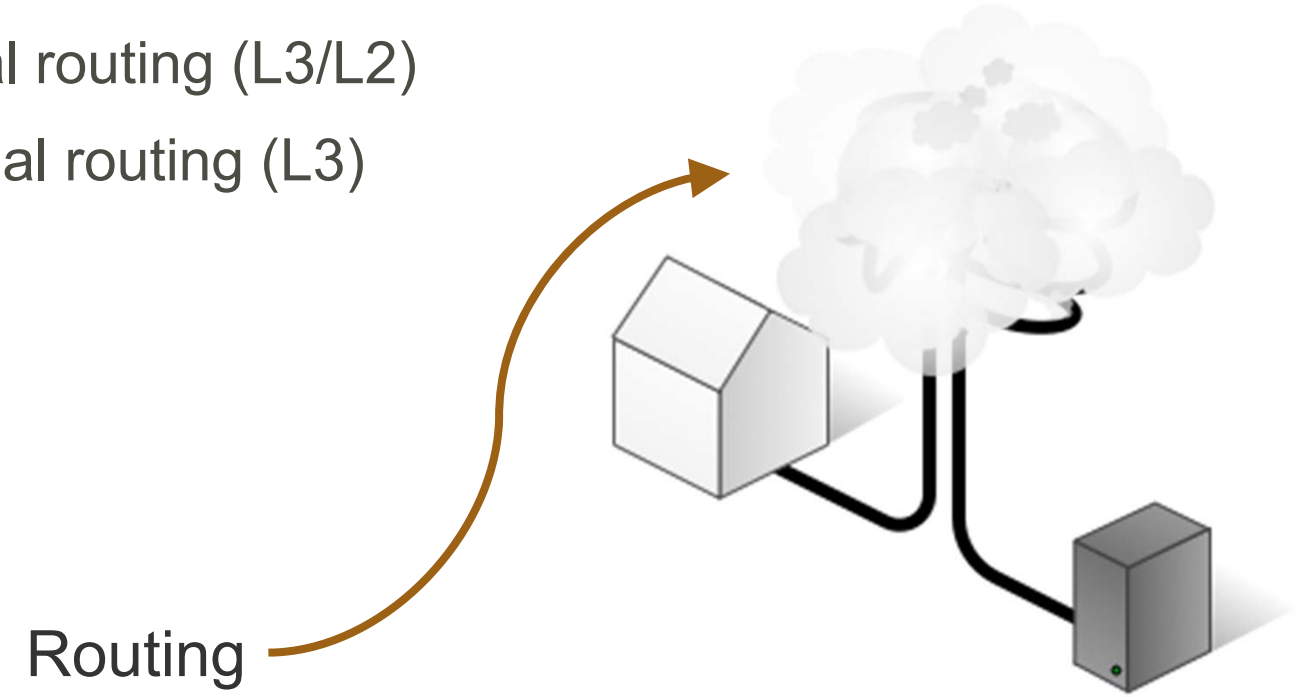
JENS ANDERSSON (BILDBIDRAG WILLIAM TÄRNEBERG)

Läsanvisning

- Kihl & Andersson:
  - Särtryck Kap 8 (finns på Canvas), kap 9.3 – 9.4
- Stallings:
  - Kap 19.1 & 19.2
- Forouzan 5th ed
  - Kap 20.1 – 20.3, 21.1 – 21.2

Agenda

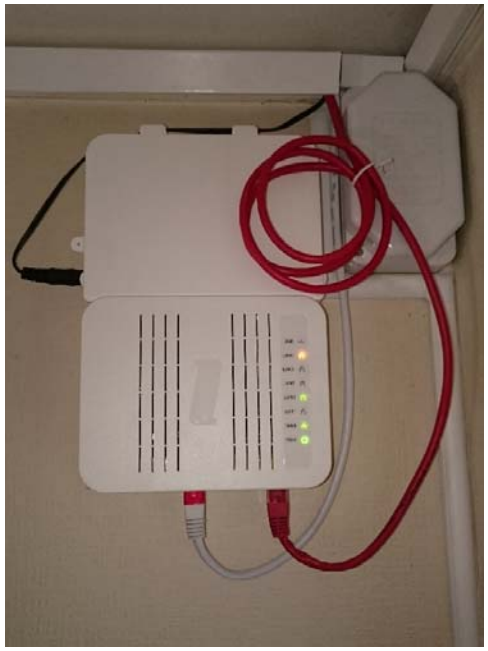
- Internet
- Lokal routing (L3/L2)
- Global routing (L3)



LUND UNIVERSITY

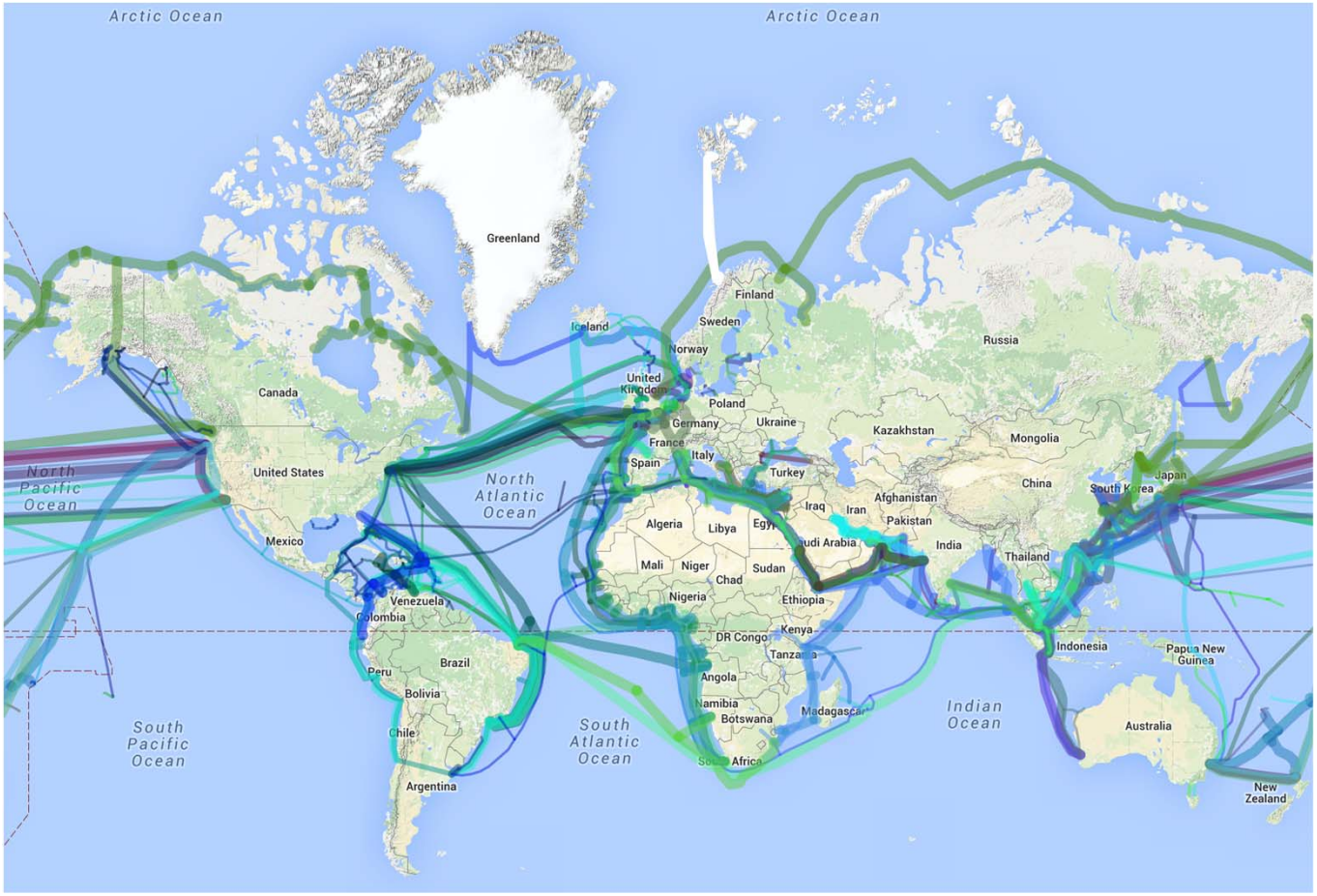
Fysiskt Internet: Hemmanätet

- Brandvägg
- Router
- WLAN
- Fil-server
- NAS
- Tjänsteserver



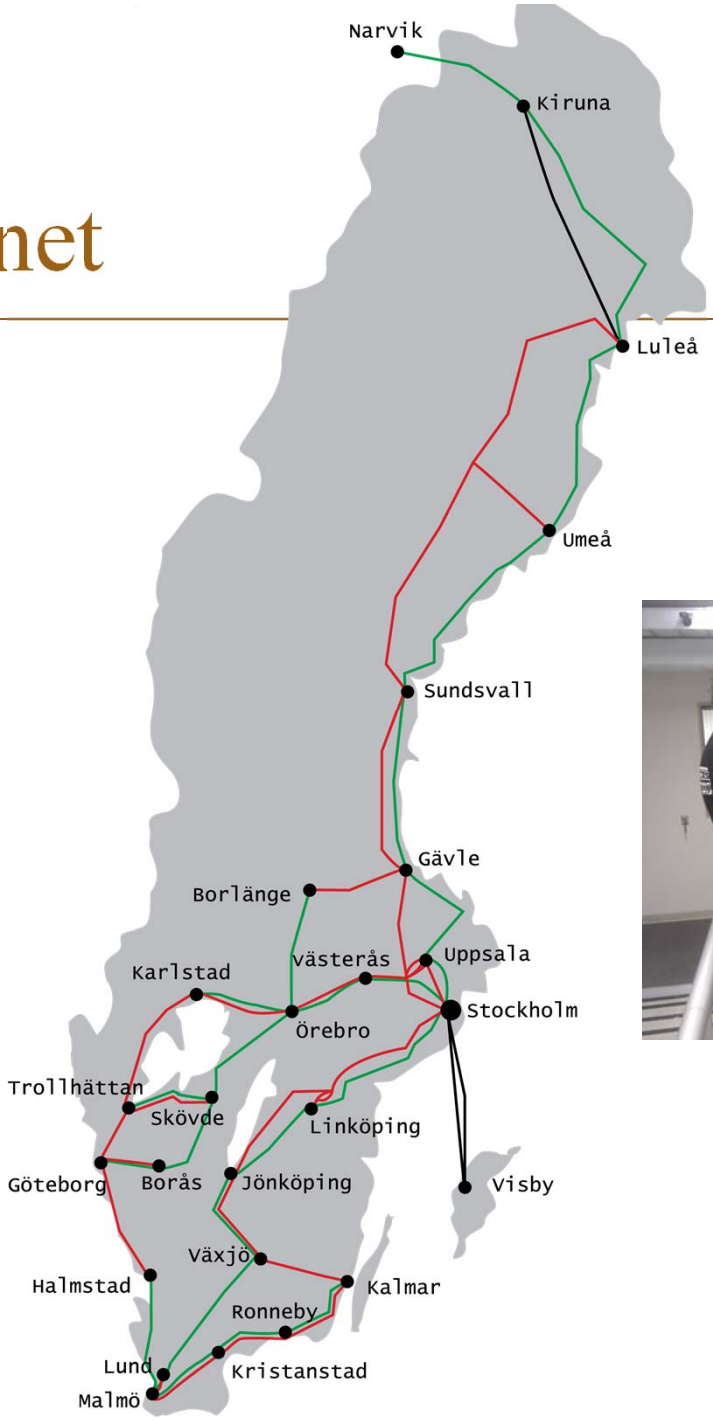
LUND UNIVERSITY

Fysiskt Internet – Globalt



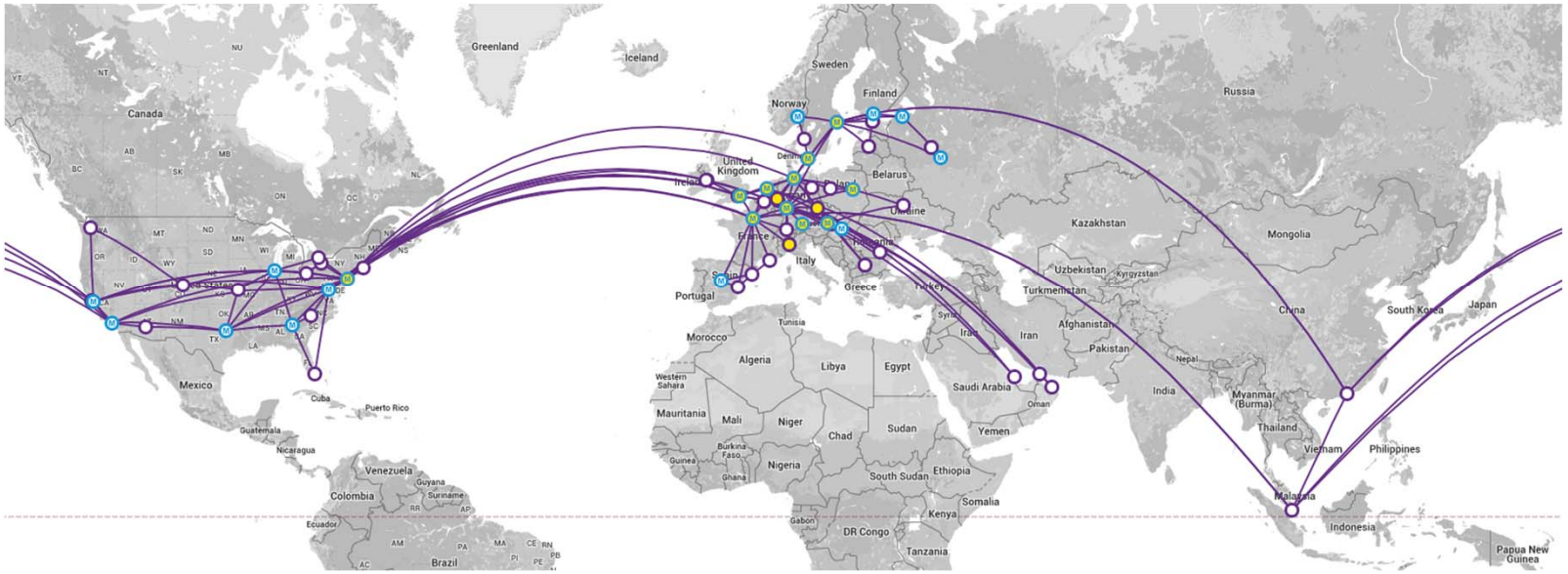
LUND UNIVERSITY

Backbone - Sunet



LUND UNIVERSITY

TeliaSoneras carrier network

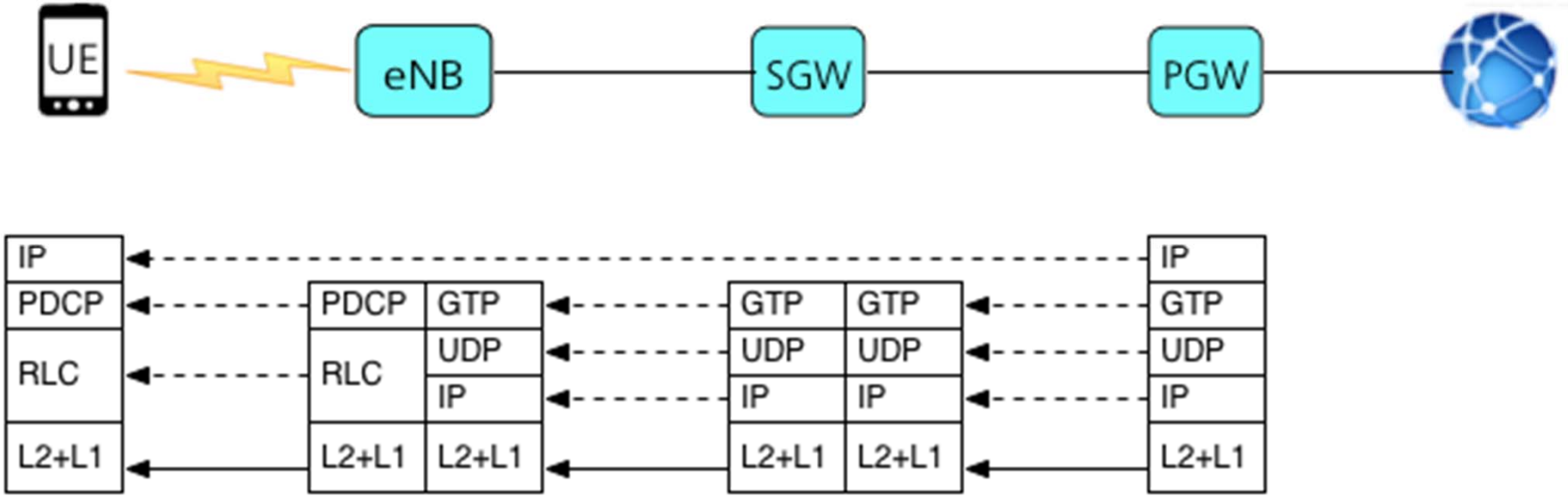


LUND UNIVERSITY

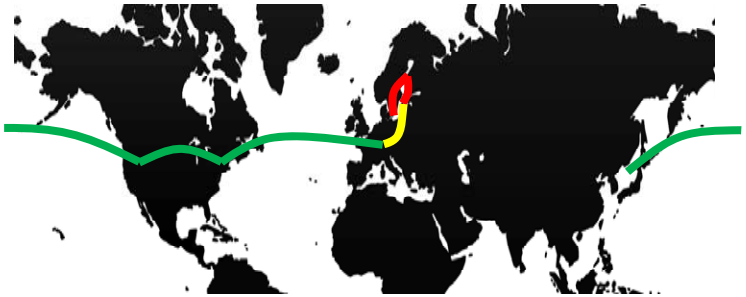
Virtuella Internet

LUND UNIVERSITY

Virtuellt Internet i LTE



Inter Domain Routing = Policy Routing



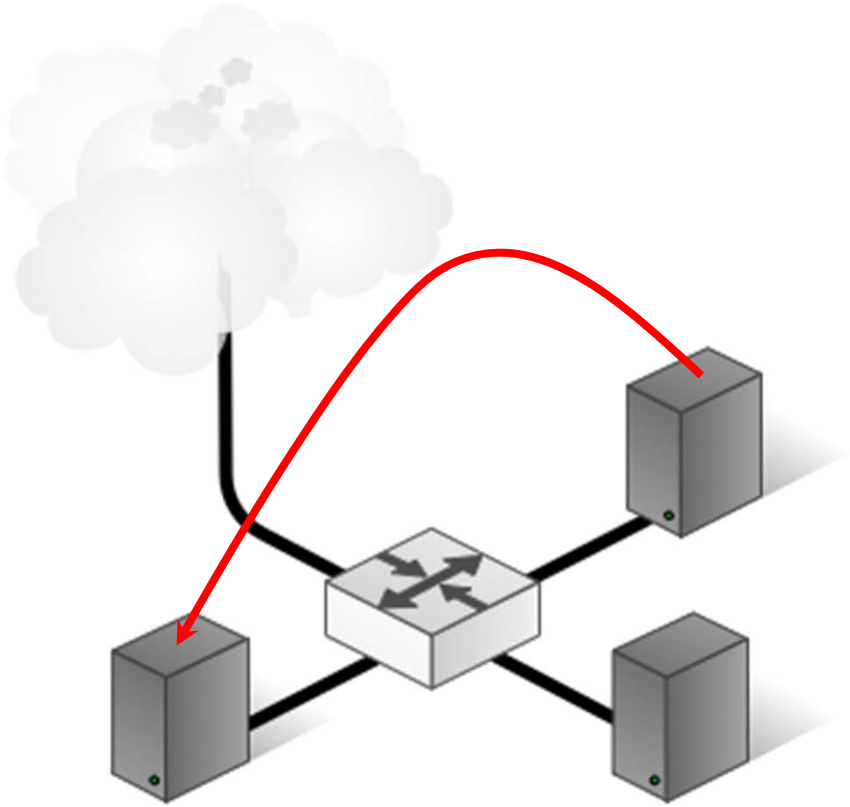
- Internet är ett nät av nät, inte ett nät
- Avtal, policy viktigare än konnektivitet
  - Vem har vi trafikutbyte med
  - Vem får bära vår trafik

Mer om Inter Domain Routing i ETSF10 Internetprotokoll

LUND UNIVERSITY

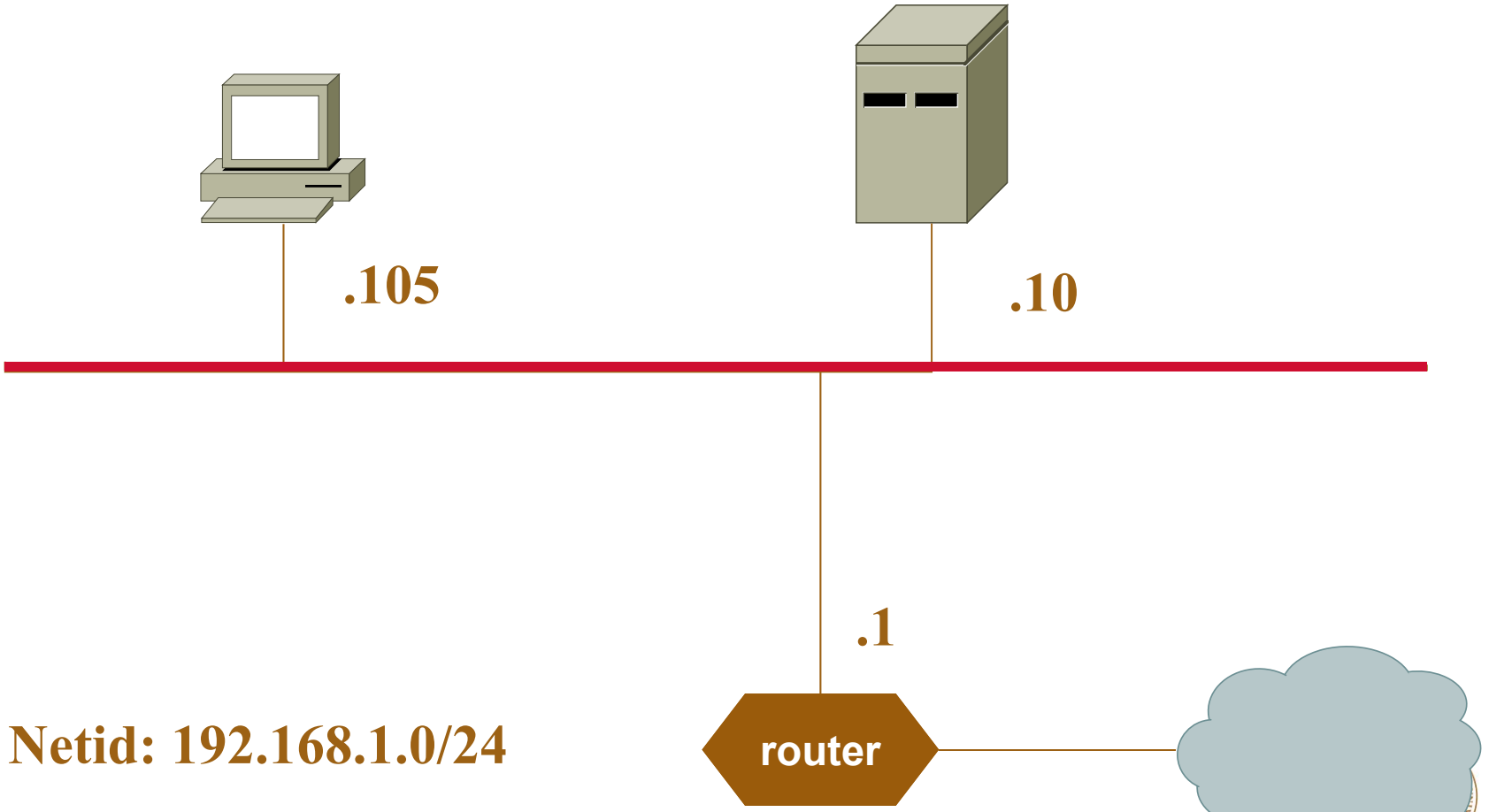
Lokal Routing

Lokal routing



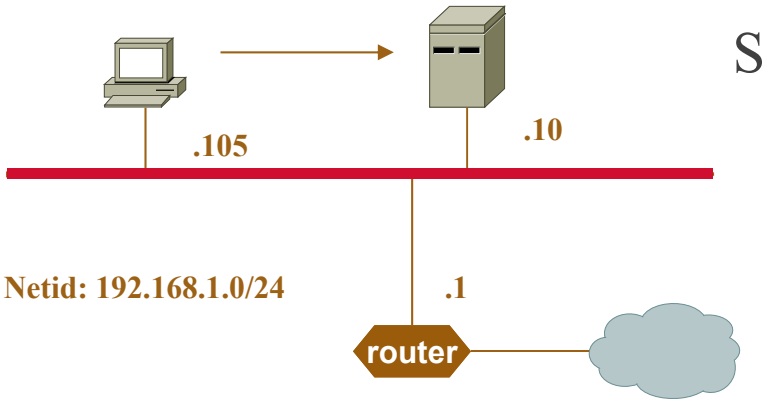
LUND UNIVERSITY

Lokal Routing & ARP (1)



LUND UNIVERSITY

Lokal Routing & ARP (2)



Sänd datagram till 192.168.1.10!

Är destinationen på samma nät?

Sändaren jämför egen nät-id med desitnationens nät-id.

i detta fall JA

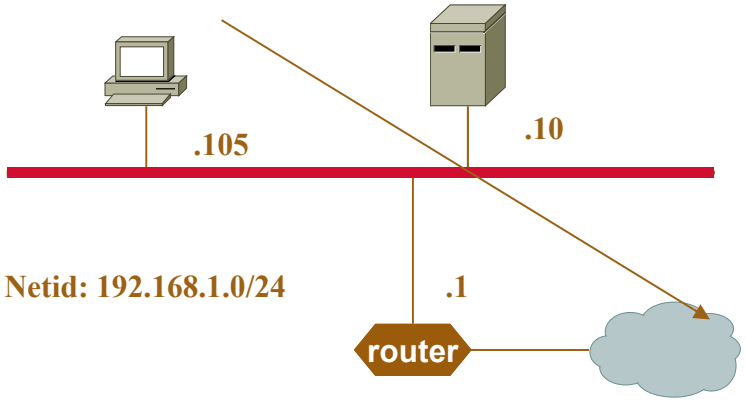
Är destinationens MAC-adressen i ARP-cache?

om JA använd den

om nej använd ARP för destinationen

LUND UNIVERSITY

Lokal Routing & ARP (3)



Sänd datagram till 10.0.100.35!

Är destinationen på samma nät?

Sändaren jämför egen nät-id med desitnationens nät-id.

i detta fall NEJ

Är def. gateway MAC-adressen känd och i ARP-cache?

om JA använd den

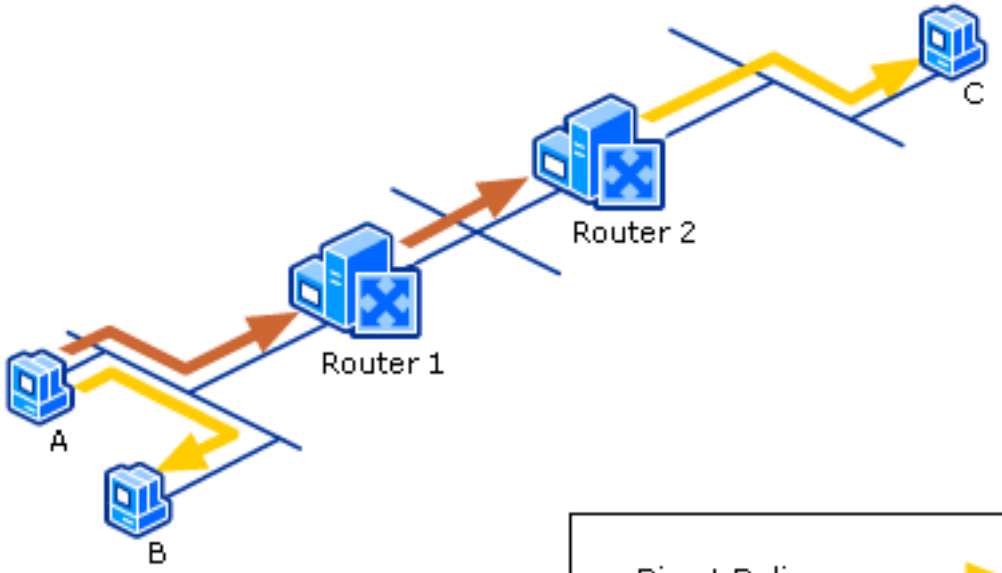
om NEJ använd ARP för def. gateway

LUND UNIVERSITY

Lokal routing och IPv6

- Fungerar analogt med IPv4 och ARP
- ARP ersätts med Neighbour Discovery Protocol i ICMPv6

ARP på alla länkar

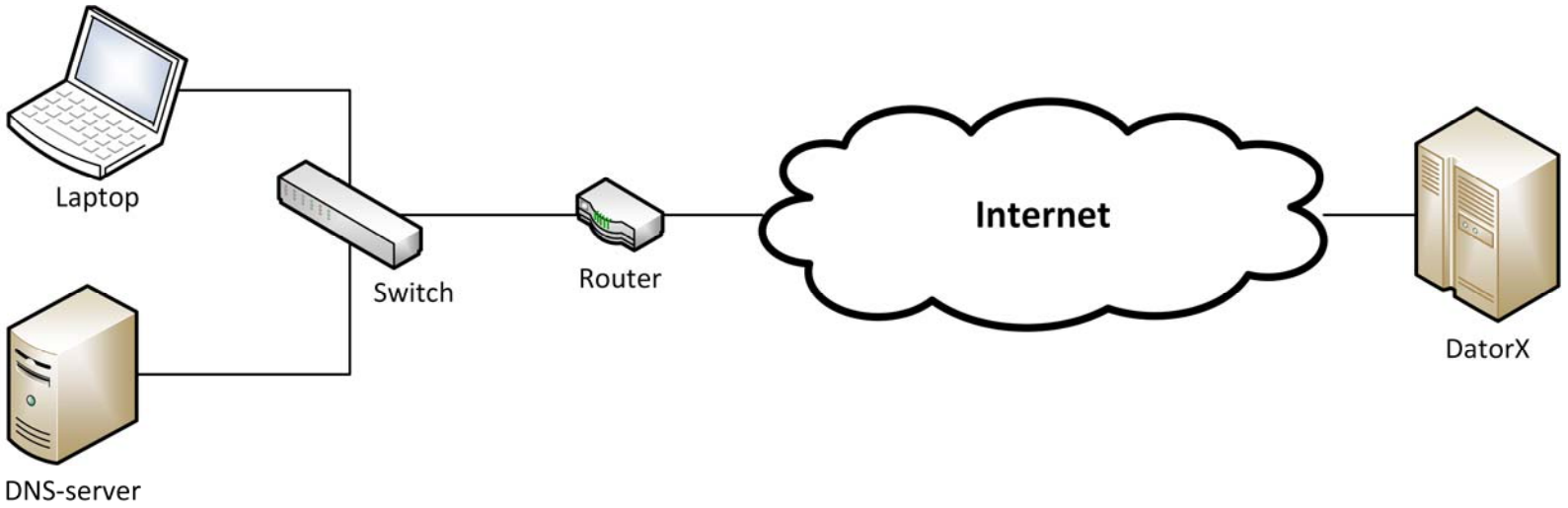


Från dator A till dator C

1. Dator A: ARP request (broadcast) “Router1”
2. Router 1: ARP reply med MAC
3. Router 1: ARP request (broadcast) “Router2”
4. Router 2: ARP reply med MAC
5. Router 2: ARP request (broadcast) “Dator C”
6. Dator C: ARP reply med MAC

LUND UNIVERSITY

Tentafråga

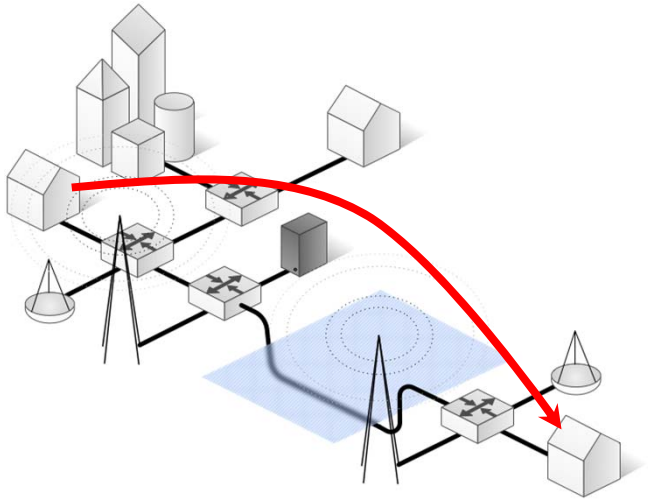


Laptop ska skicka ett IP-paket till DatorX. Vilken IP-address frågas efter i laptoppens ARP request?

LUND UNIVERSITY

Global routing

Global routing

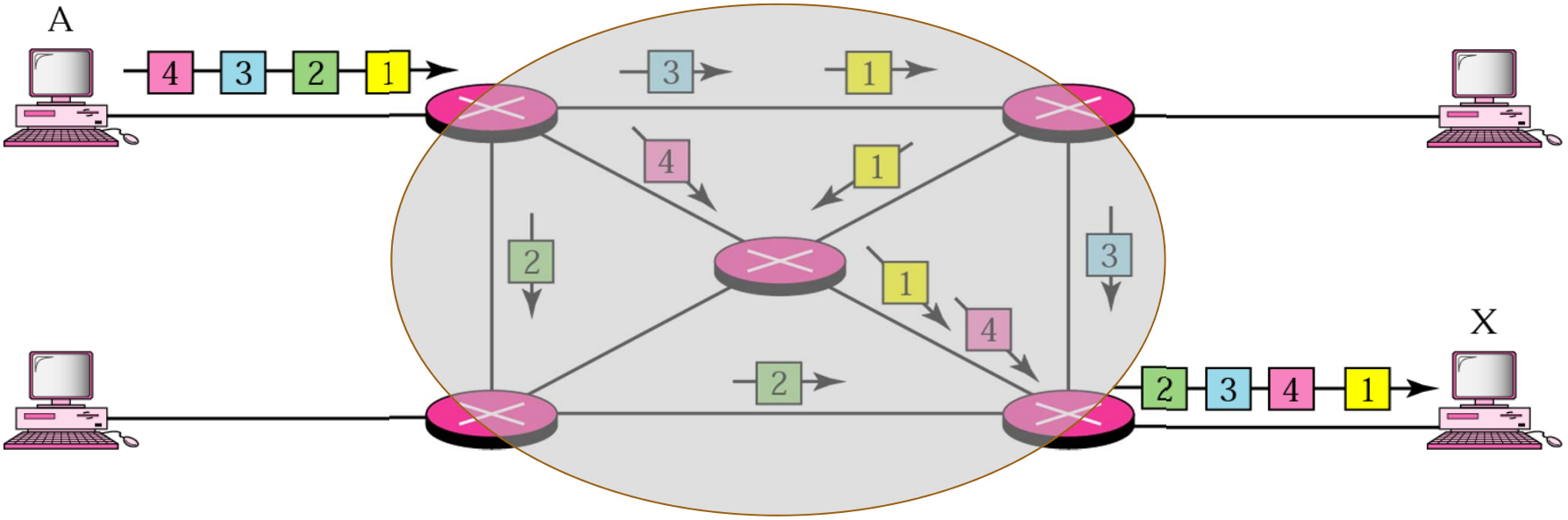


Problem att lösa:

- Enskilda noder skickar vidare till nästa nod
- Varje nod måste ha **kunskap om hela(?) nätet** (destinationer)
- Bestämma **bästa väg** från nod till destinationen
- Bestämma ***Next Hop*** på bästa väg

LUND UNIVERSITY

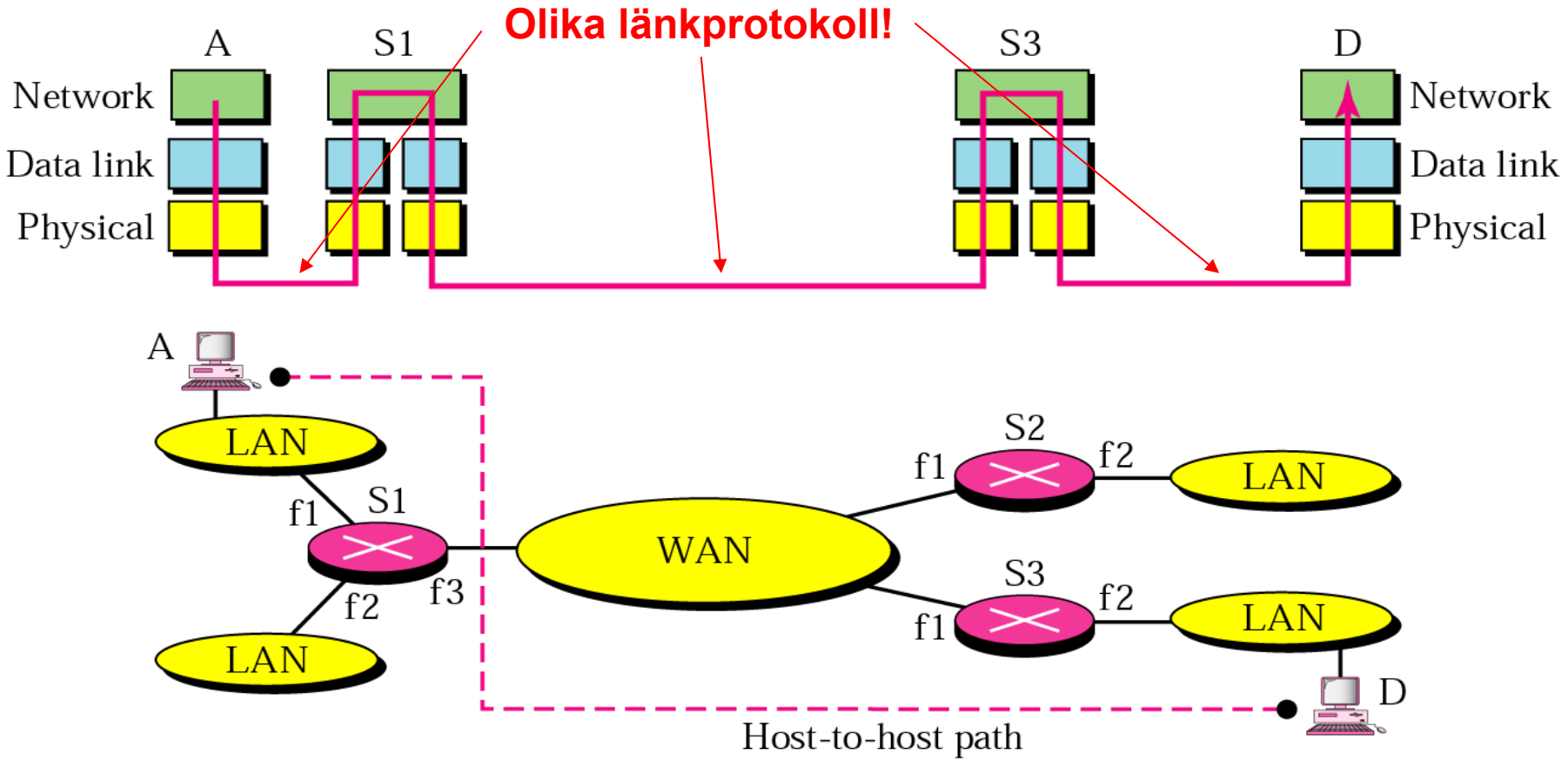
Datagram och nätet



- Paketen tar olika vägar
  - kan komma fram i oordning
- Dator A behöver inte veta vilken väg paketet kommer att ta eller om det kommer fram (gäller IP-lagret)

LUND UNIVERSITY

Nätverkslagret /Lager 3



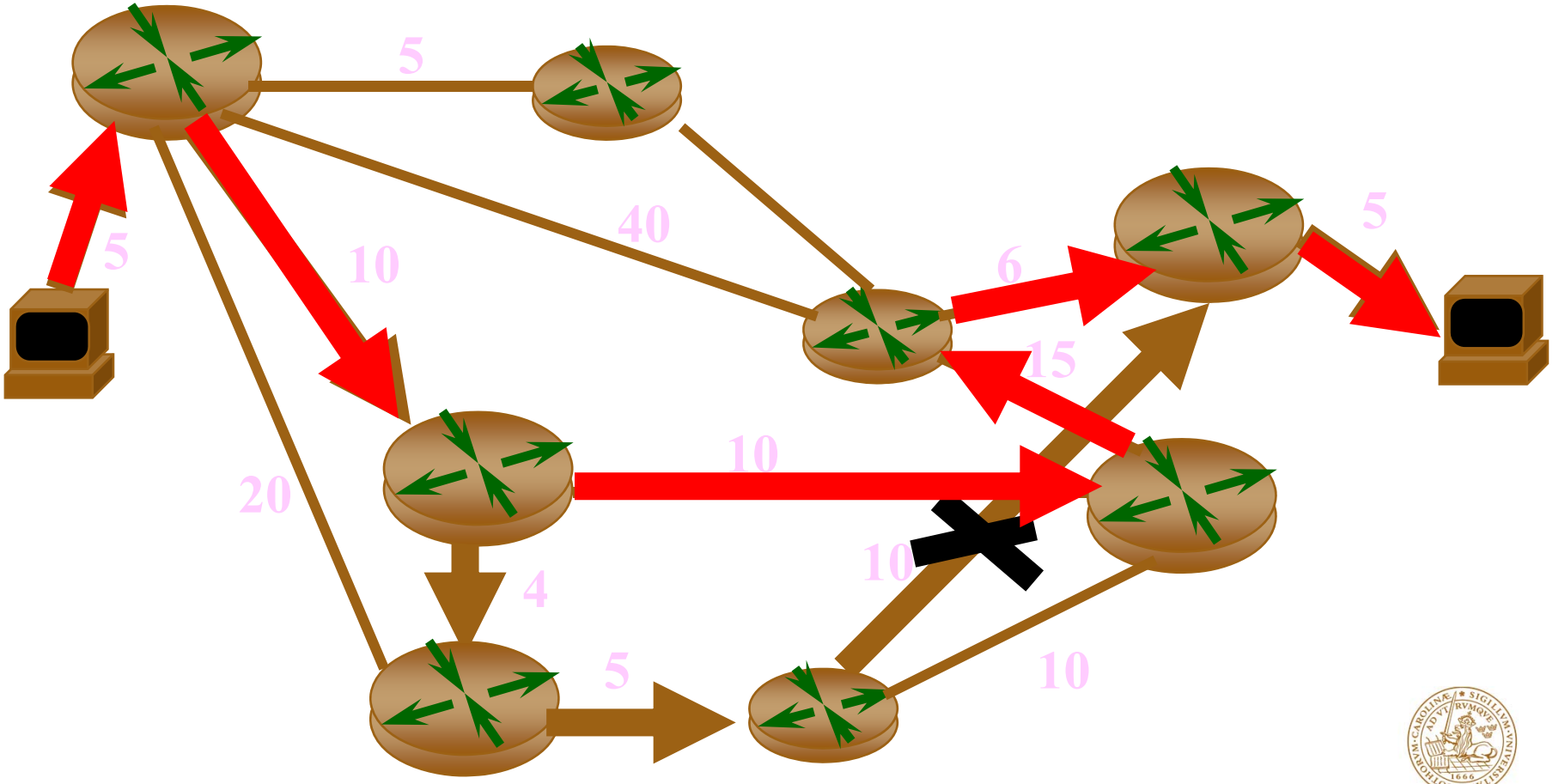
Routern

- En router förmedlar paket mellan nätverk baserat på nätverkslagrets adresser
- Routing-beslut fattas utifrån nät-identitet (net id), inte värd-identitet (*host id*)
- En router gör ”intelligenta” beslut om bästa väg för paketets vidare leverans mot slutdestinationen

Routern och nät-id

- Routern lär sej hur paket ska skickas bästa väg mot destinationens nät
- Routern arbetar med nät-identiteter/nät-addresser
- Destinationens värdadress är bara intressant för den sista routern på vägen

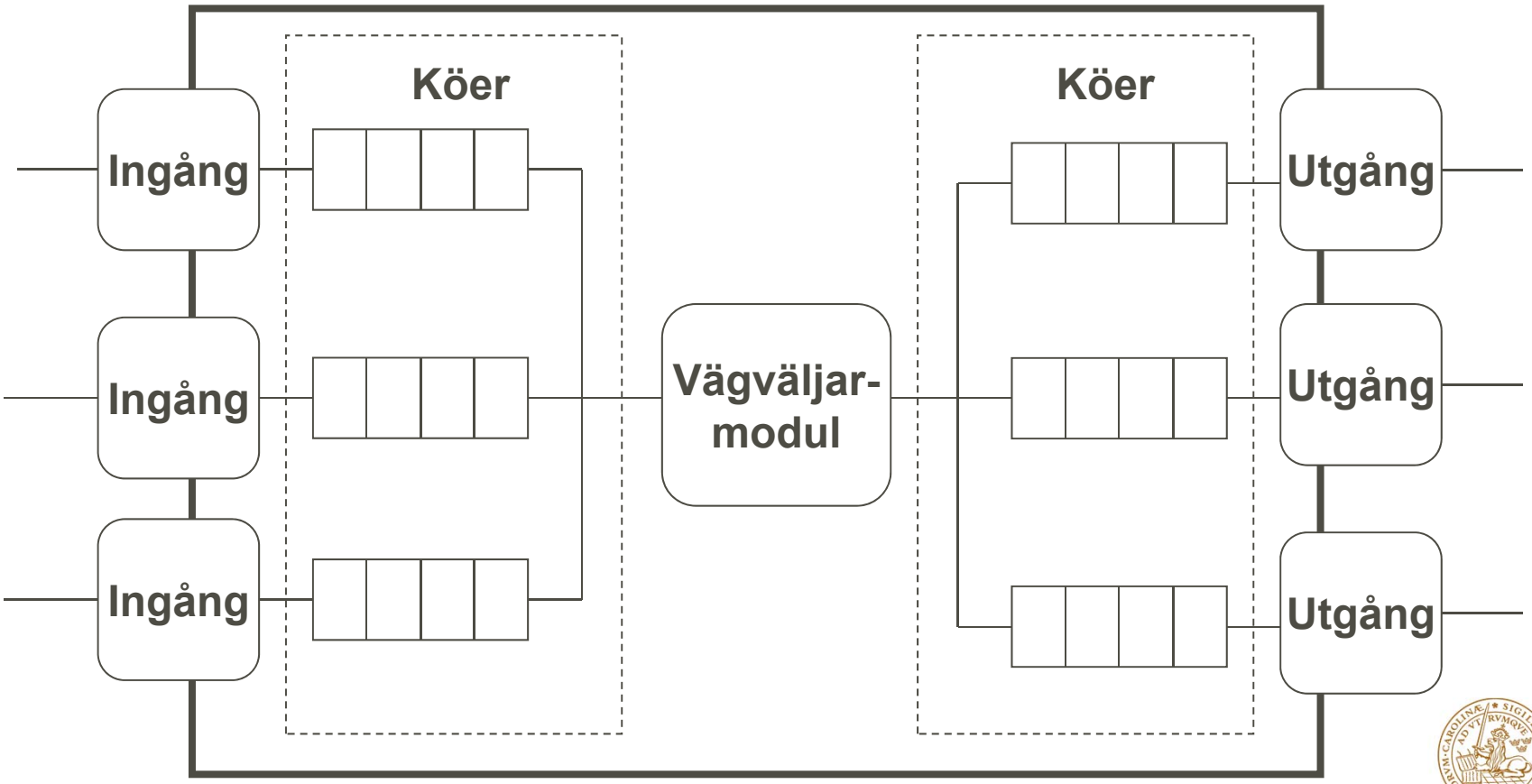
Uppgift: Välj bästa väg!



I alla lägen!

LUND UNIVERSITY

Routern, schematiskt



LUND UNIVERSITY

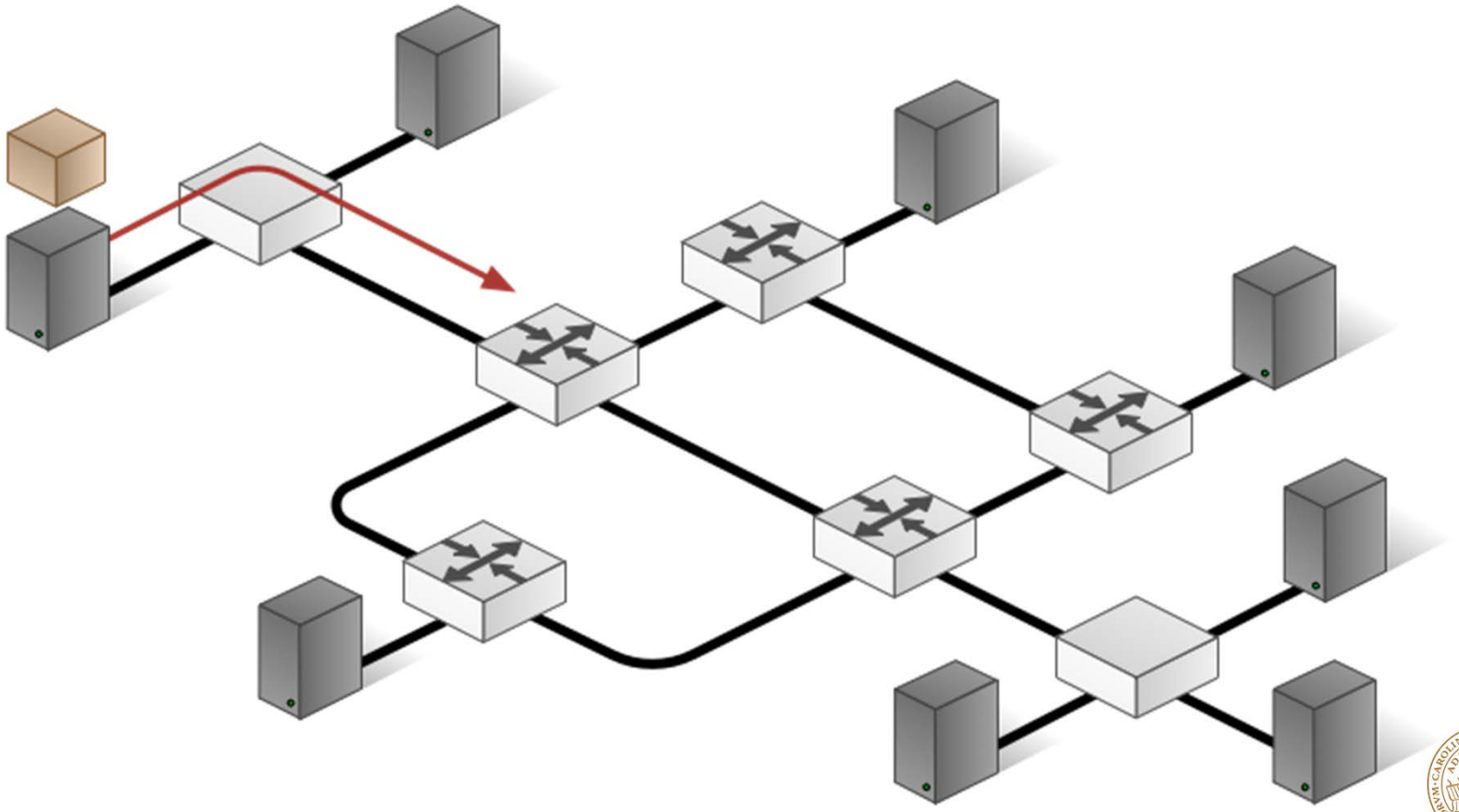
Routingprinciper

- Ingen “intelligens”
- Centraliserad
- Distribuerad

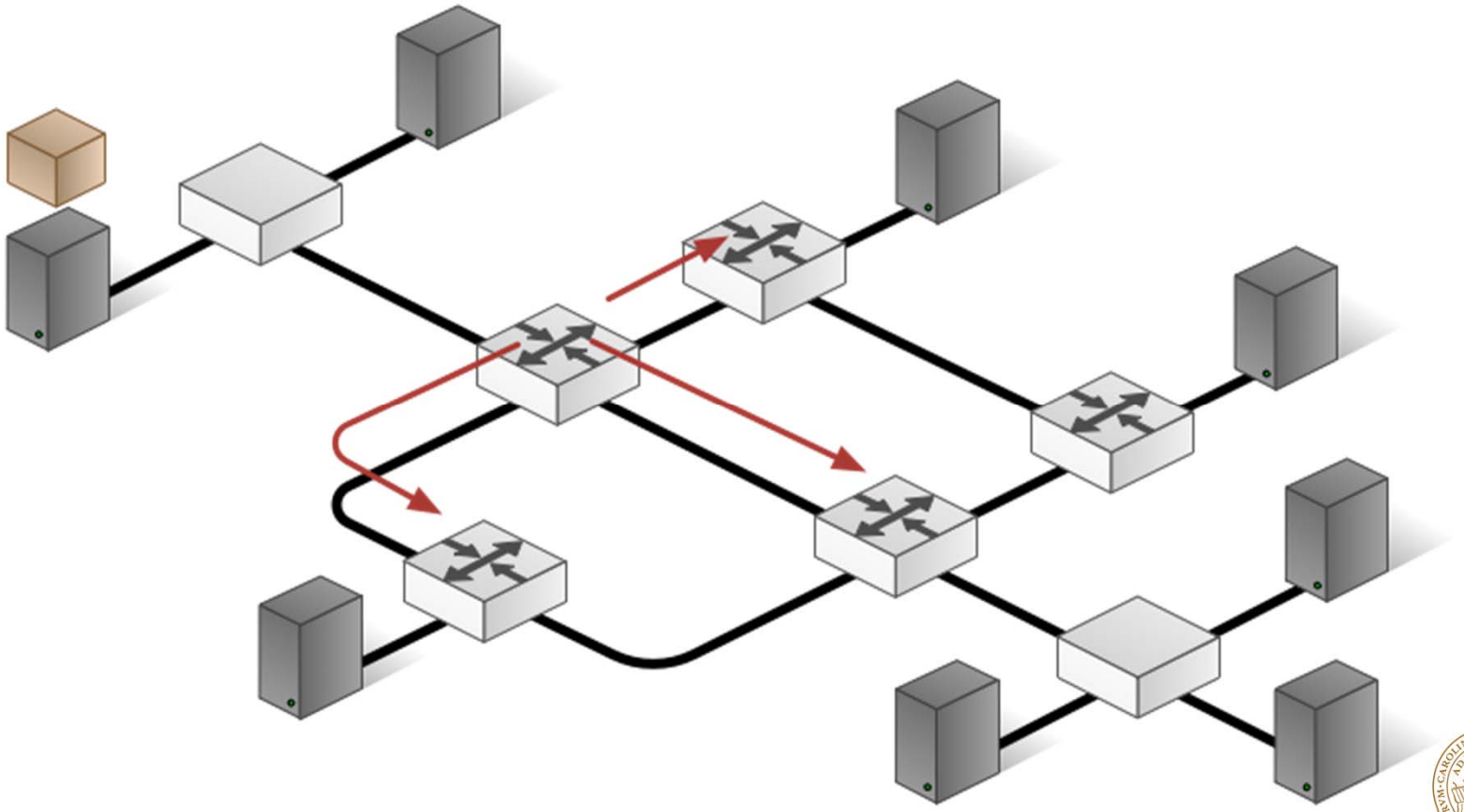
Ingen “intelligens”: Flooding

- Skicka ut all paket/datagram
  - På alla portar/interface/linkar
  - Utom ingress-porten/interfacet/linken

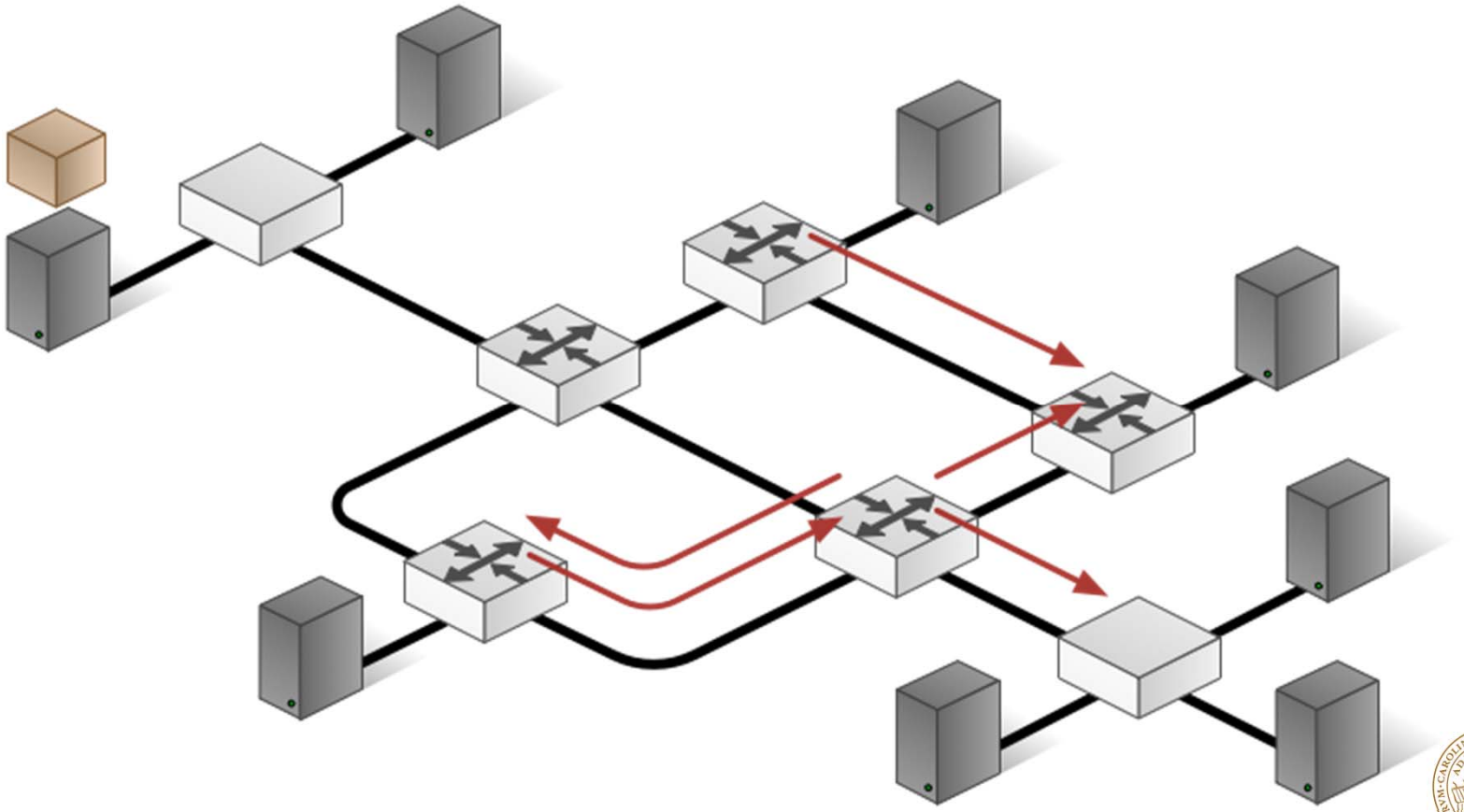
Flooding



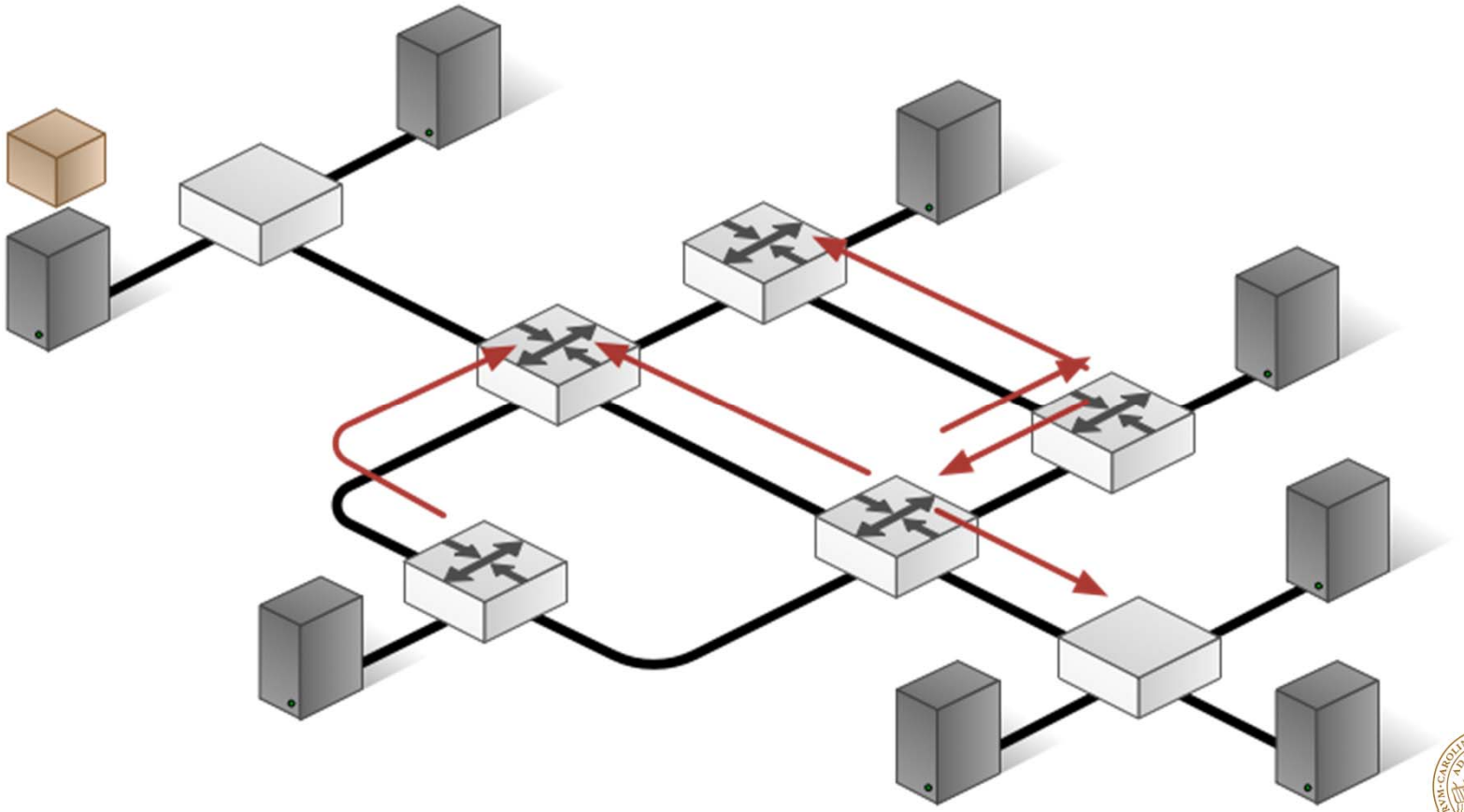
Flooding



Flooding



Flooding



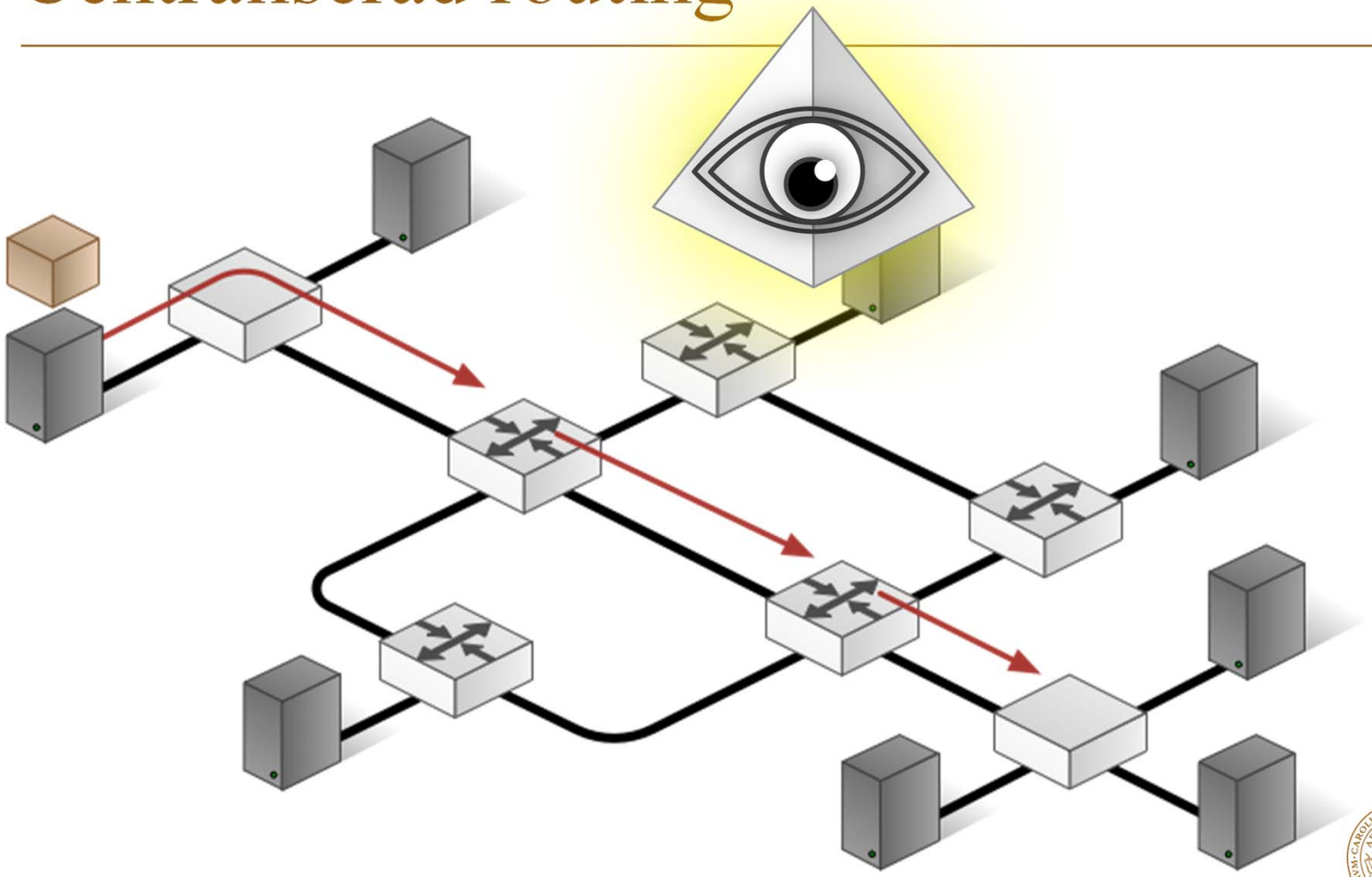
Ingen “intelligens”: Flooding

- Skicka ut all paket/datagram
  - På alla portar/interface/linkar
  - Utom ingress-porten/interfacet/linken
- Problem?
  - Paket som loopar
  - Onödig trafik
  - Två lösningar
    - »TTL-räknare
    - »Kom ihåg vilka paket som redan hanterats

Centraliserad routing

- Separera *control plane* och *data plane*
- Databas och algoritm centralt
  - Noderna i nätet uppdaterar den centrala funktionen
- Paketförmedlingen distribuerad
  - självklart! eller?
- *Software Defined Networks*

Centraliserad routing



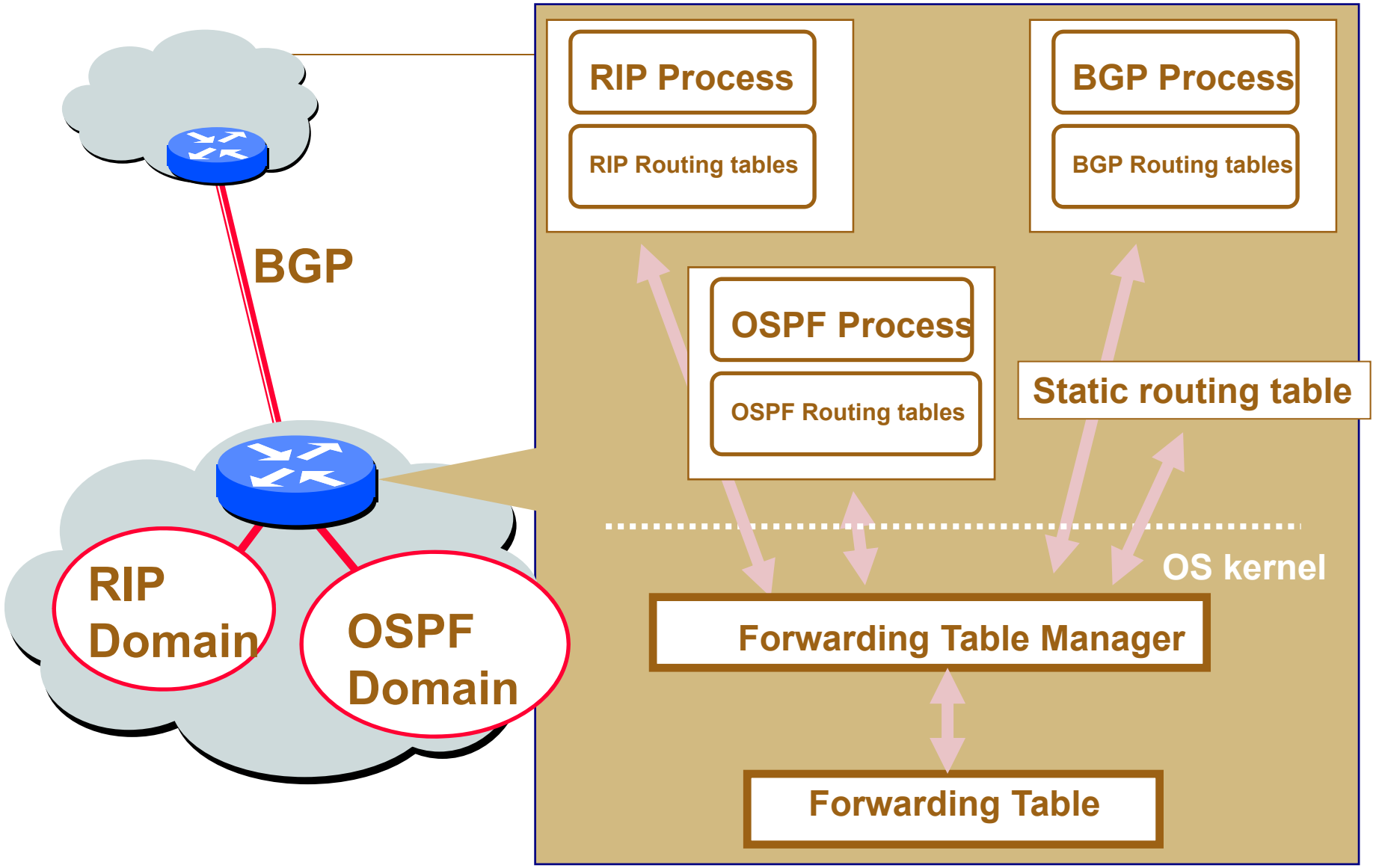
LUND UNIVERSITY

Distribuerad routing

- Routingprocessen (*control plane* och *data plane)* distriburad till alla routrar
- Två metoder
  - **Distance Vector**
    - » Varje nods information om bästa vägar distribueras till nodens grannar
    - » Bästa väg e-2-e fås fram genom jämförelse med alla möjliga *next hop*
    - » Enkelt, låga krav på processor och minne
  - **Link State**
    - » Information om lokal om topologi flödas (*flooding*) till alla noder
    - » Bästa väg e-2-e till alla noder beräknas lokalt i varje nod (trädbyggnad)
    - » Komplicerat med krav på processorkraft och minne

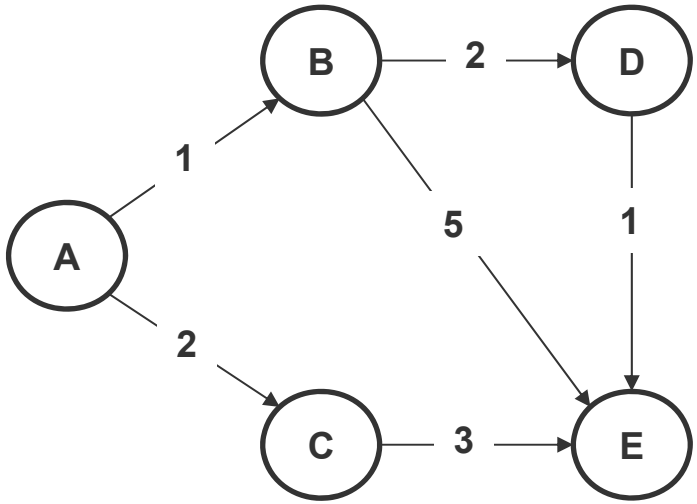
LUND UNIVERSITY

Routing Tables and Forwarding Table



Grafteori

Grafteori



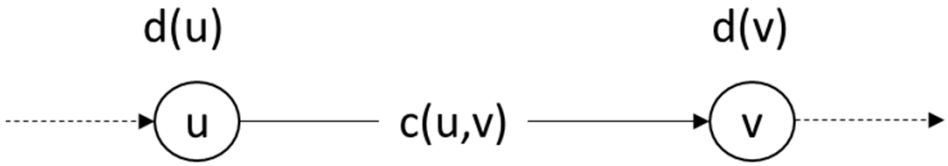
- En graf består av noder (*vertices*) och bågar/kanter (*edges*)
- Bågar sammanbinder noder
- Viktad (*weighted)* graf: bågar har kostnader
- Riktad (*directed*) graf: bågarna kan har riktning
- Trädbyggnad:
  I en arbiträr graf finna ett träd med bästa vägar från en nod till varje annan nod
- Bäst = lägst kostnad

LUND UNIVERSITY

Trädbyggnad: två principer

- Routing = hitta bästa väg från källan till alla destinatoren i nätet -> Trädbyggnad
- Två algoritmer för trädbyggnad
  - Bellman-Ford
  - Shortest Path First (SPF, Dijkstras algoritm)

Bellman-Ford



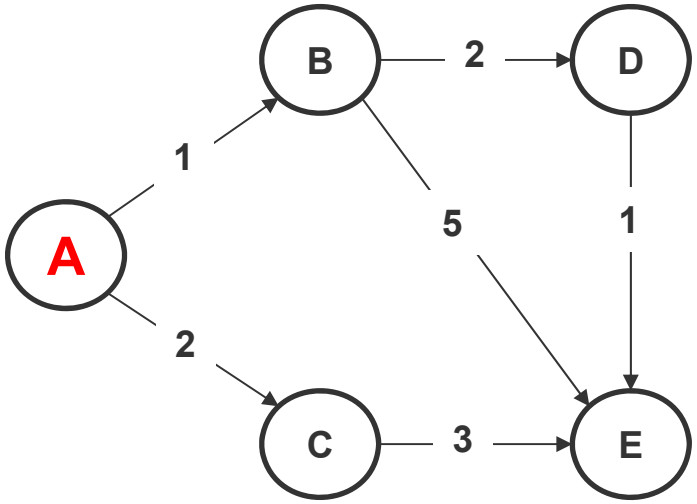
- Iterativ metod
- För varje iteration
  - För varje båge (u,v)

```
if d(u) + c(u,v) < d(v) then d(v) = d(u) + c(u,v)
```

- d(u) = aktuell kostnad (*distance*) för nod u
- c(u,v) = bågen (u,v) kostnad (*cost*)

Bellman-Fords algoritm

```
for i= 1 to number of vertices - 1
   for all edges (u,v) i edge list
      if d(u) + c(u,v) < d(v) then d(v) = d(u) + c(u,v)
```



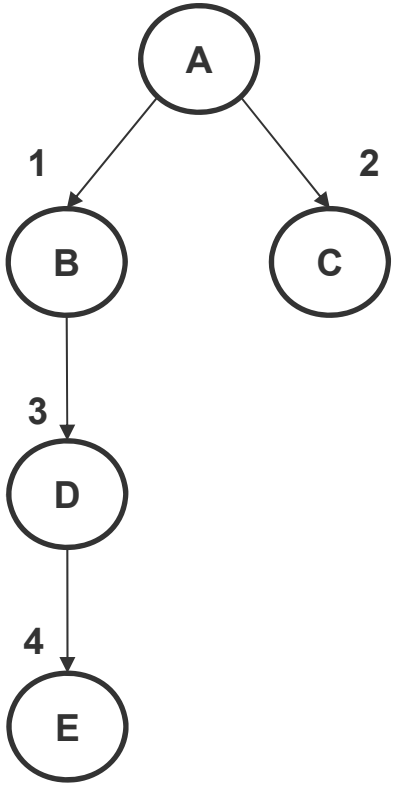
Exempel

Iteration	A	B	C	D	E
0	0	∞	∞	∞	∞
1	0	1	2	∞	∞
2	0	1	2	3	5
3	0	1	2	3	4

Iteration	A	B	C	D	E
0	0	∞	∞	∞	∞
1	0	1(A)	2(A)	∞	∞
2	0	1(A)	2(A)	3(AB)	5(AC)
3	0	1(A)	2(A)	3(AB)	4(ABD)

LUND UNIVERSITY

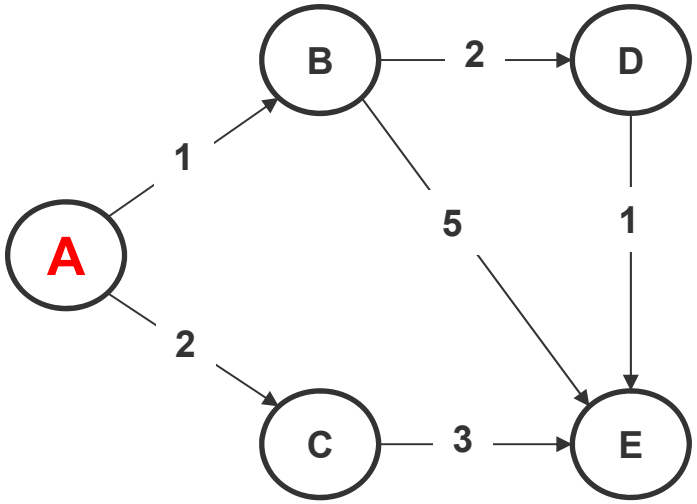
Slutligt träd



LUND UNIVERSITY

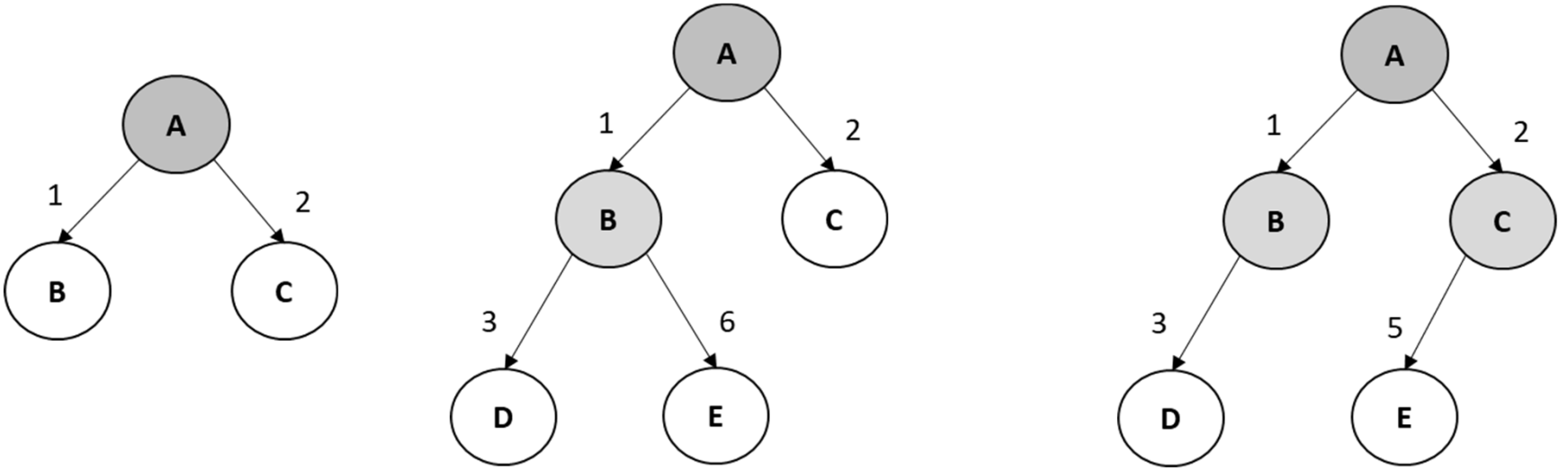
Shortest Path First (Dijkstra)

1. Identify the root (the node itself)
2. Attach all neighbor nodes temporarily
3. Make link and node with least cumulative cost permanent
4. Choose this node
5. Repeat 2 and 3 until all nodes are permanent



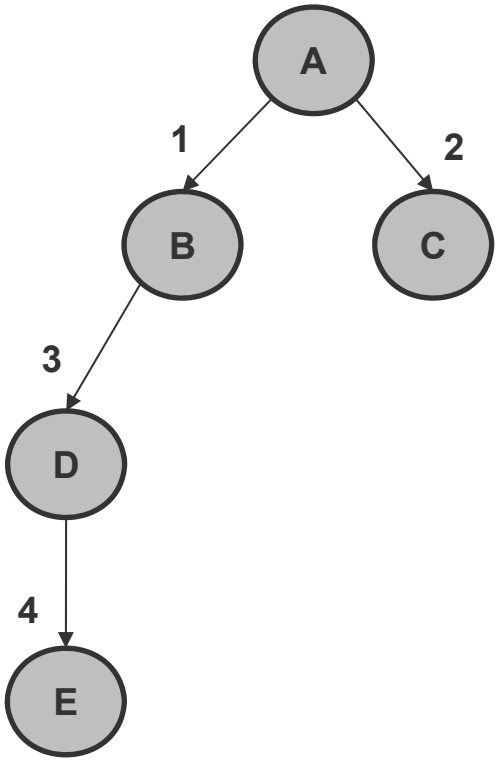
LUND UNIVERSITY

SPF: tre iterationer



LUND UNIVERSITY

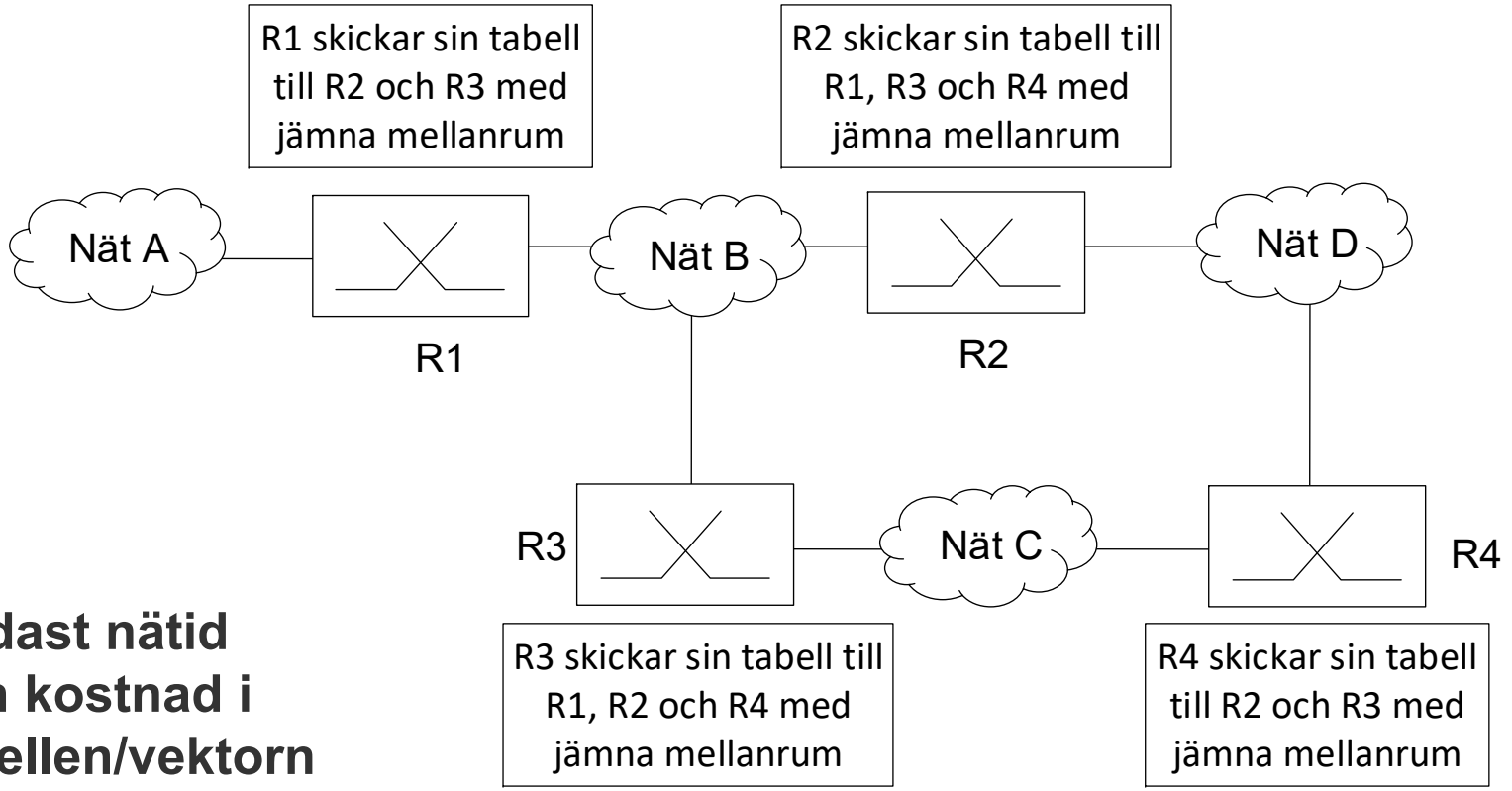
Slutligt träd



LUND UNIVERSITY

Distance Vector

Distance Vector: Princip



Endast nätid och kostnad i tabellen/vektorn

LUND UNIVERSITY

Distance vector: princip

- Alla kända bästa vägar **skickas till grannar**
  - Periodiskt
  - Vid varje förändring
- Routingtabeller **uppdateras** vid
  - Info om nya noder
  - Ändrad kostnad eller vägar/*paths*
- ”Global kunskap sprids lokalt”

En distance vector

Allmänna fallet

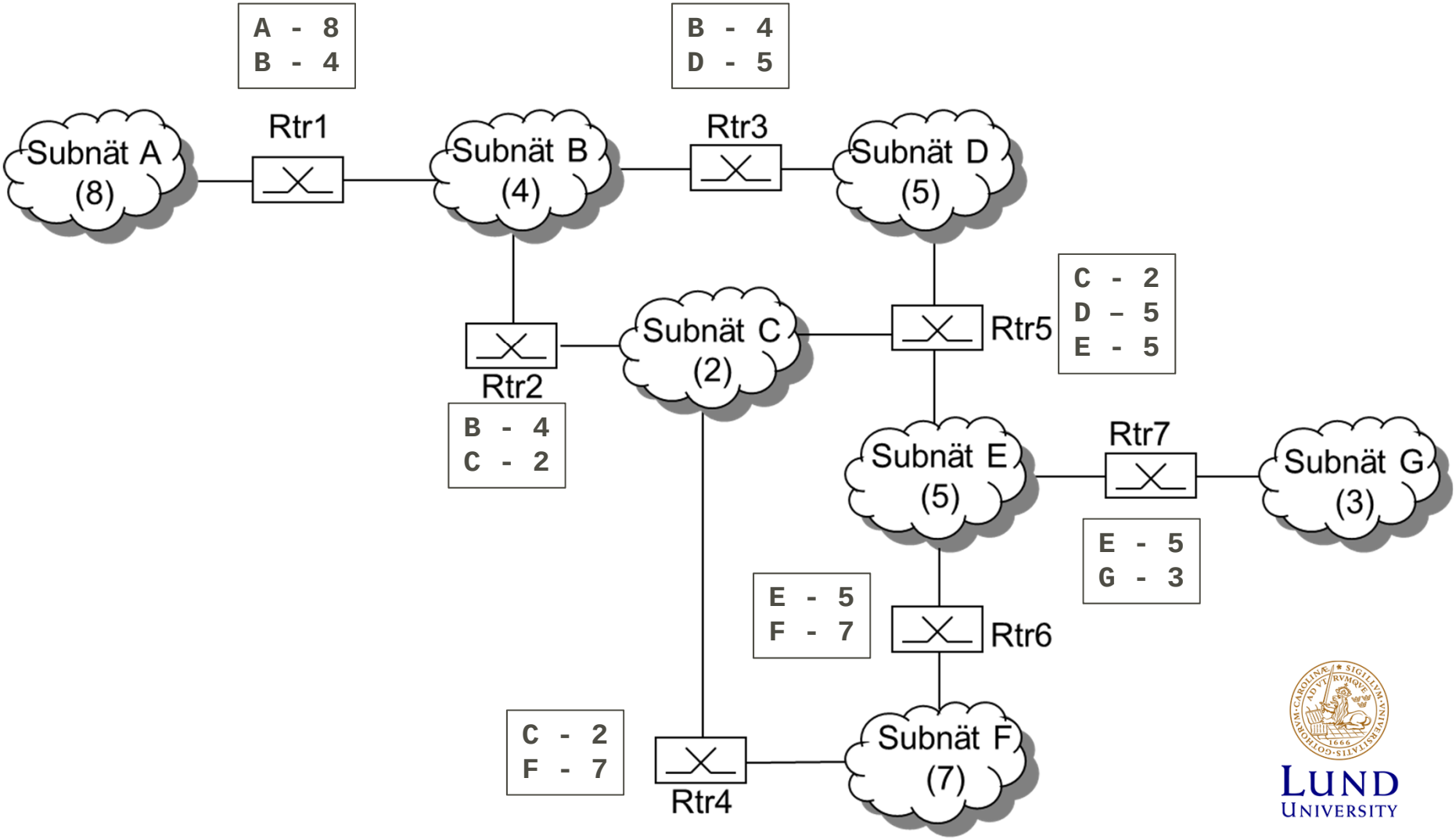
Nod	Kostnad
-	-
-	-
-	-
-	-
-	-

För routing

Destination	Kostnad
-	-
-	-
-	-
-	-
-	-

Next Hop = den som skickar vektorn

Exempel



LUND UNIVERSITY

Uppdatering av routingtabell

Rtr3

Ursprunglig		
B	-	4
D	-	5

Från Rtr5		+5
C	2	+5 = 7
D	5	+5 = 10
E	5	+5 = 10

Uppdaterad		
B	–	4
C	rtr5	7
D	–	1
E	rtr5	10

Rtr1

Ursprunglig		
A	-	8
B	-	4

Från Rtr2		+4
B	4	+4 = 8
C	2	+4 = 6

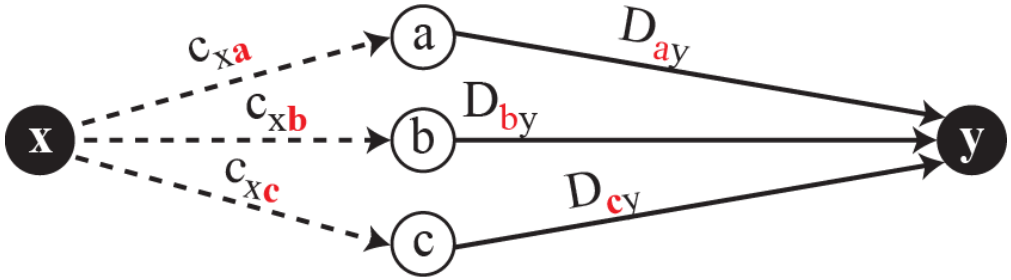
Uppdaterad		
A	-	8
B	–	4
C	rtr2	6

LUND UNIVERSITY

Bellman-Fords algoritm (anpassad)

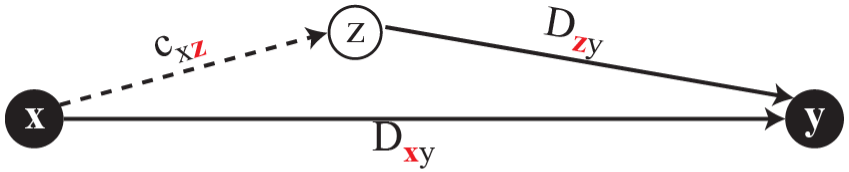
```
(1)          if (advertised destination not in table) then
                    update table
(2)          else
(2.a)        if (advertised next-hop = next-hop in table) then
                    replace entry
(2.b)        else
(2.b.i)      if (advertised hop count < hop count in table) then
                    replace entry
(2.b.ii)     else
                    do nothing
```

Bellman-Fords algoritm



a. General case with three intermediate nodes

$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}) \ldots\}$$



b. Updating a path with a new route

$$D_{xy} = min\{D_{xy}, (c_{xz} + D_{zy})\}$$

Not! D_{xy} kan ändras utan att nod z tillkommit!

LUND UNIVERSITY

Tentafråga

NetID	Cost	Next Hop
A	3	rtrX
B	1	---
C	4	rtrY

Uppdatering från rtrY

NetID	Cost
A	1
C	2
D	4

Hur ser den nya routingtabellen ut efter uppdatering från rtrY?

Distance Vector, funderingar

- Periodiska uppdateringar!?
  - Hur hitta grannar?
  - Hur upptäcka att en granne försvinner?
- Problem med länkar och noder (bortom grannar) som försvinner.
  - Finns inget naturligt sätt att säga ”avbrott”

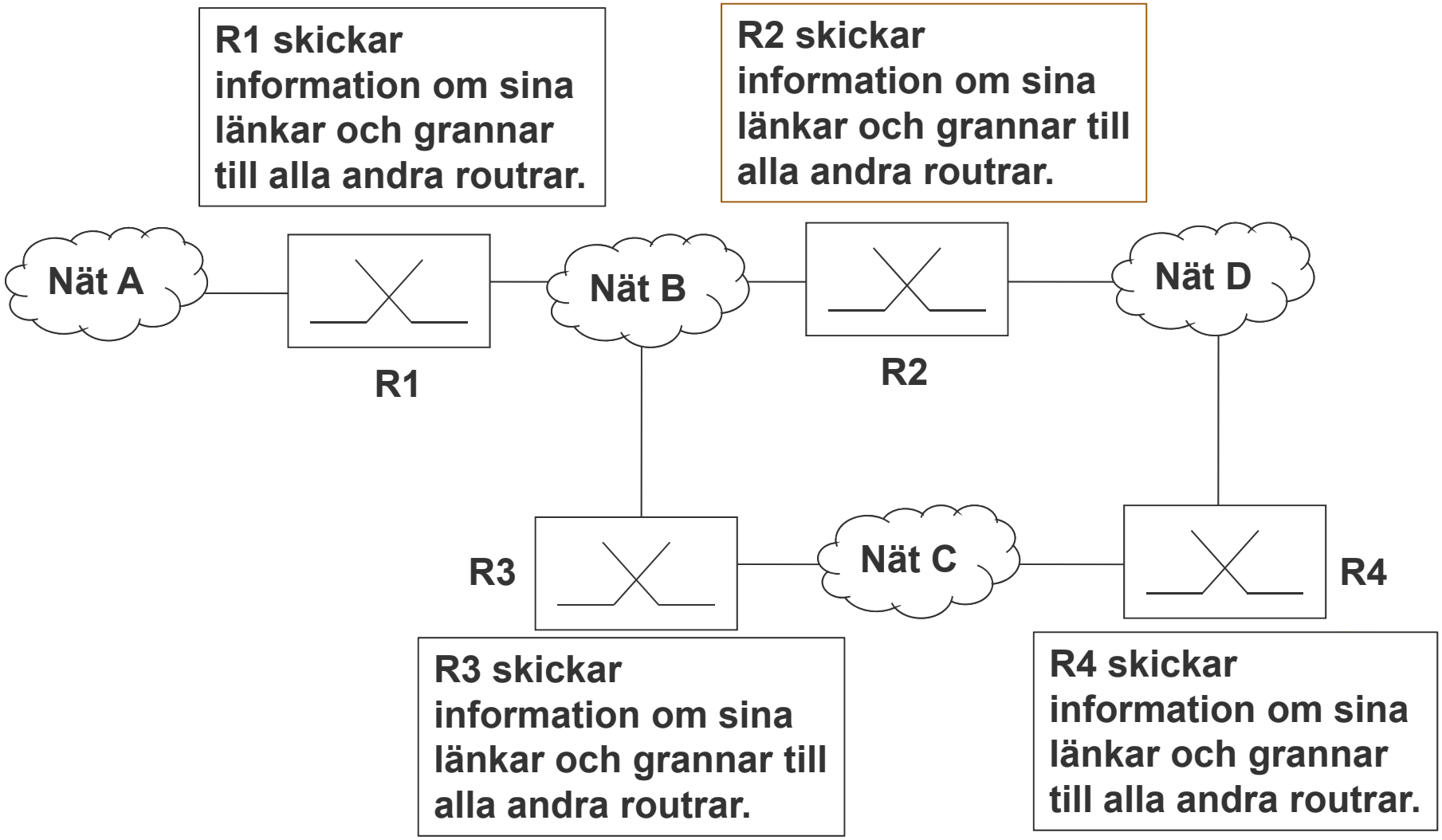
Mer i ETSF10 Internetprotokoll

Link State

Link state: princip

- **Lokal topologi** info **flödas** globalt (LSA)
  - Vid lokal förändring
  - Periodiskt (i praktiken mycket sällan, typ varje halvtimme)
- Skapa databas i varje node med alla kända link states
- Uppdatera routing-tabell när ny information läggs in i databasen (Shortest Path First)
- ”Lokal kunskap sprids globalt”

Link State: princip



LSA (*Link State Advertisement*)

Advertiser	Network ID	Cost	Neighbour

Link State Databas, exempel

Advertiser	Network ID	Cost	Neighbour
Rtr 1	Net A	8	---
Rtr 1	Net B	4	Rtr 2
Rtr 1	Net B	4	Rtr 3
Rtr 2	Net B	4	Rtr 1
Rtr 2	Net C	2	Rtr 4
Rtr 2	Net C	2	Rtr 5
Rtr 3	Net B	4	Rtr 2
Rtr 3	Net B	4	Rtr 2
Rtr 3	Net D	10	Rtr 5
…..	…..	…..	…..

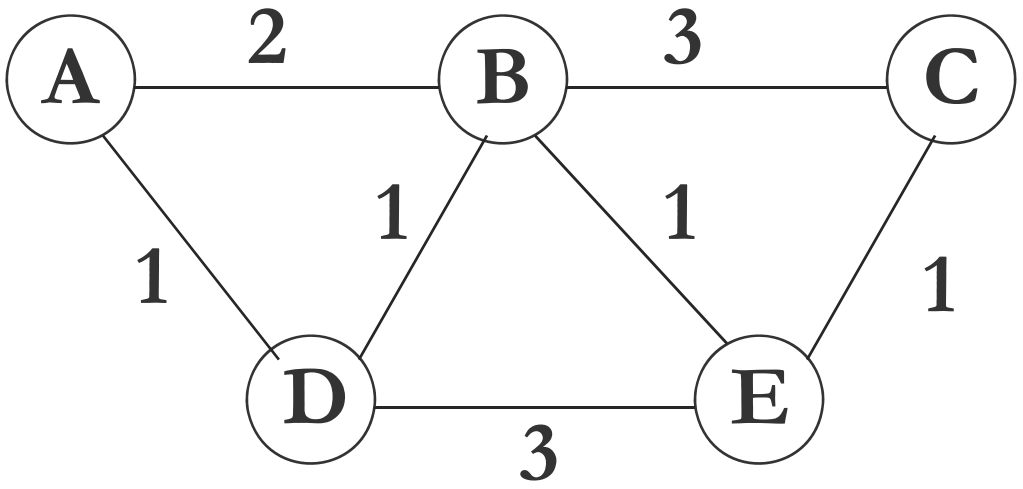
LUND UNIVERSITY

Dijkstras algoritm: Shortest Path First

1. Identify the root (the node itself)
2. Attach all neighbor nodes temporarily
3. Make link and node with least cumulative cost permanent
4. Choose this node
5. Repeat 2 and 3 until all nodes are permanent

LUND UNIVERSITY

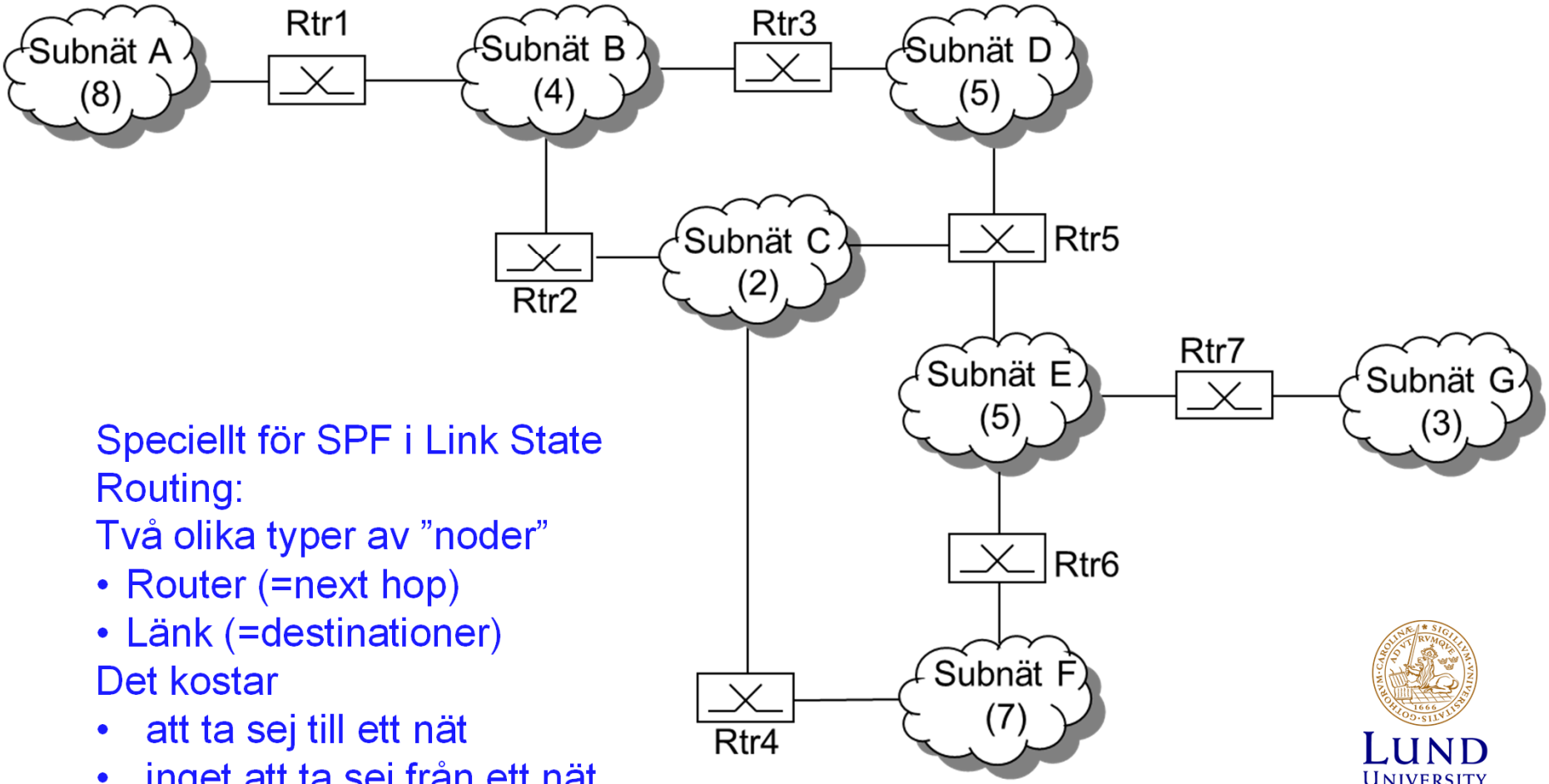
Tentafråga



Rita ett Shortest Path Fast-träd med Dijkstars algoritm utgående från nod A.

LUND UNIVERSITY

Link State: ett exempel



Speciellt för SPF i Link State Routing:

Två olika typer av ”noder”

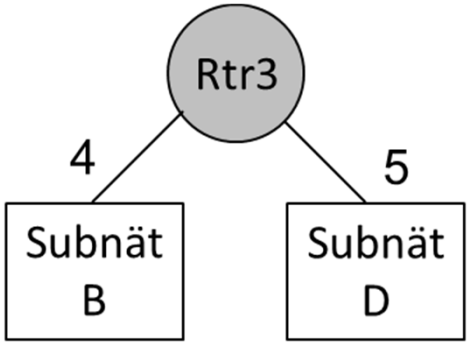
- Router (=next hop)
- Länk (=destinationer)

Det kostar

- att ta sej till ett nät
- inget att ta sej från ett nät

LUND UNIVERSITY

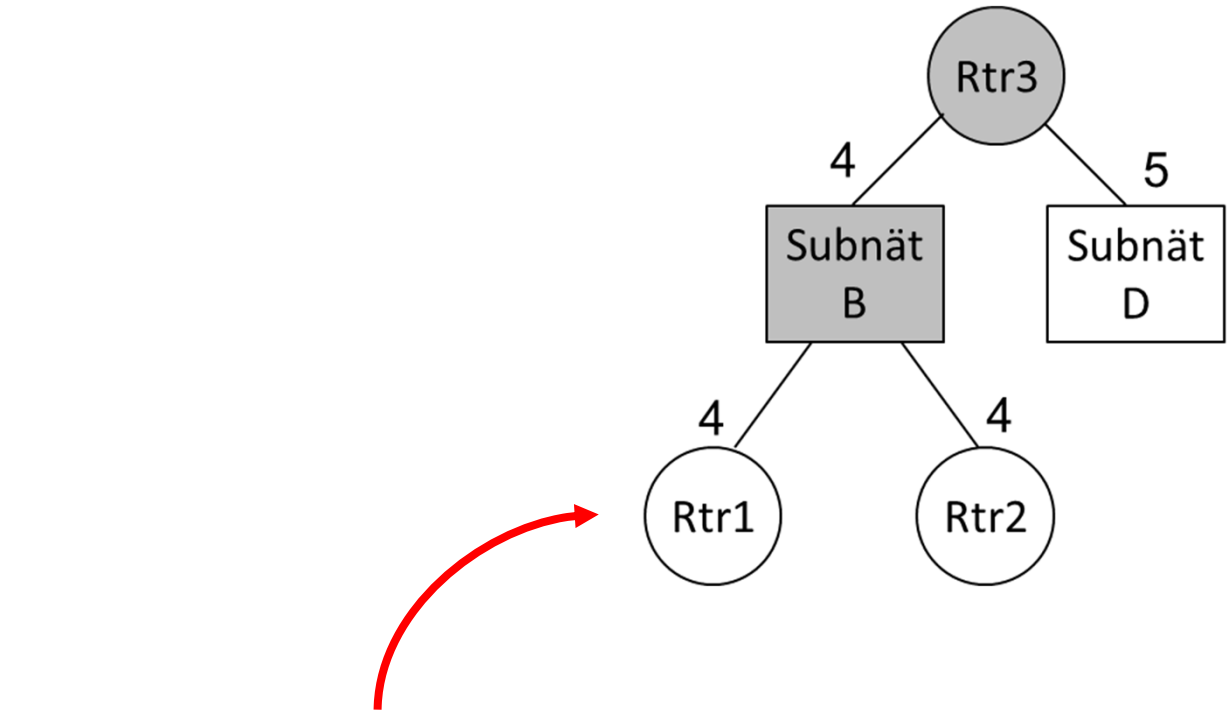
SFP Rtr 3: steg 1



“Billigast” -> Permanent

LUND UNIVERSITY

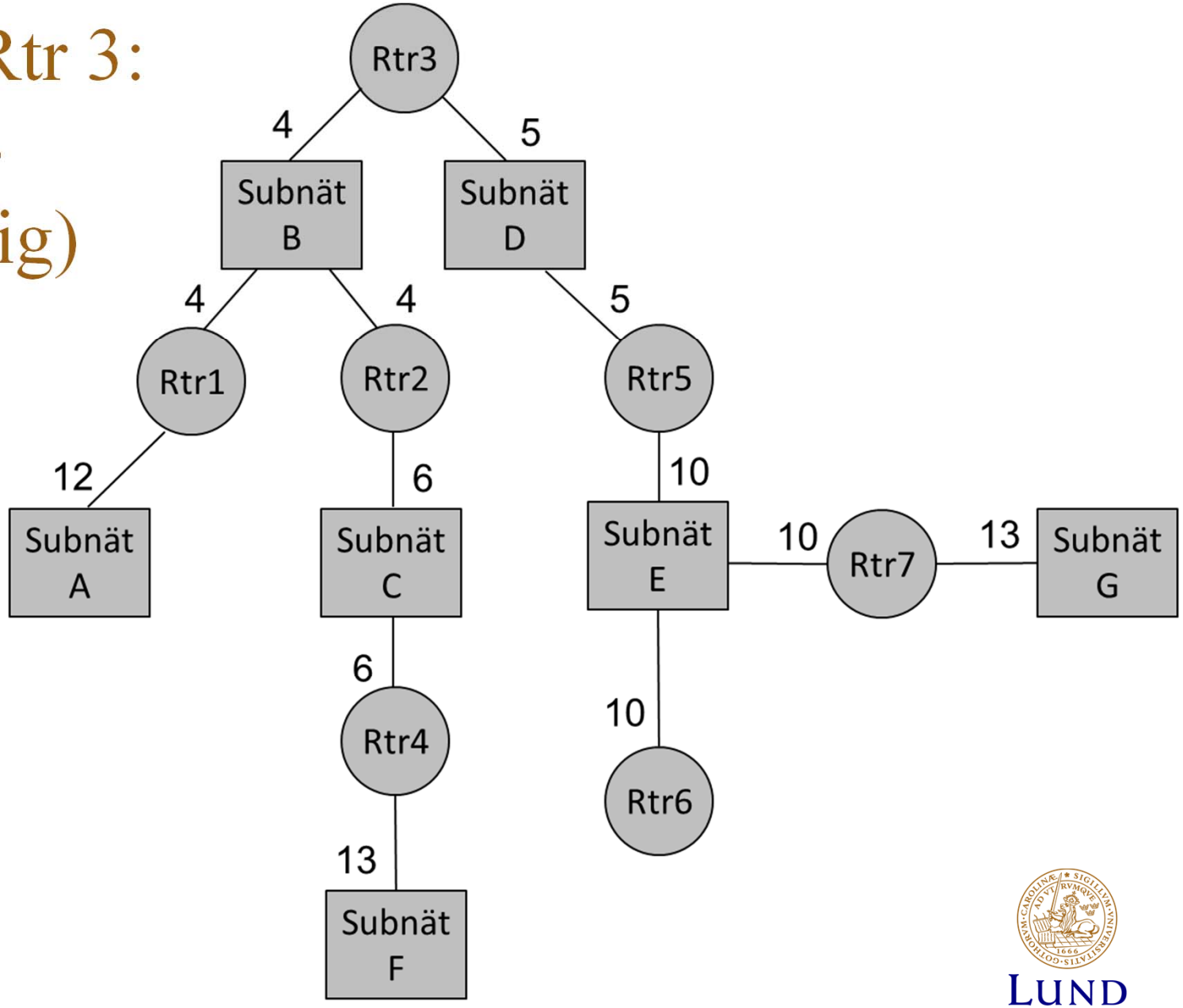
SFP Rtr 3: steg 2



“Billigast” -> Permanent

LUND UNIVERSITY

SFP Rtr 3: steg 4 (Slutlig)



LUND UNIVERSITY

Routingtabell för rtr3

Network ID	Next-hop	Cost
Net A	Rtr1	12
Net B	-	4
Net C	Rtr2	6
Net D	-	5
Net E	Rtr2	10
Net F	Rtr2	13
Net G	Rtr2	13

LUND UNIVERSITY

Link State, funderingar

- Problem med länkar och noder som försvinner?
- Hur hitta grannar?
- Hur upptäcka att en granne försvinner?
- Periodiska uppdateringar!?

Mer i ETSF10 Internetprotokoll

Tentafråga

NetID	Cost	Next Hop	Interface
10.0.10.0	2	17.16.12.254	rl0
192.168.100.0	1	d.c.	rl2
1.0.0.0	4	23.45.103.2	rl3
0.0.0.0	3	17.16.12.254	rl0

IP-paket med följande destination tas emot av router med denna routing-/forwarding-tabell. Vilken är nästa mottagare av paketen?

1. 1.0.0.10
2. 192.168.100.100
3. 130.235.200.53

LUND
UNIVERSITY