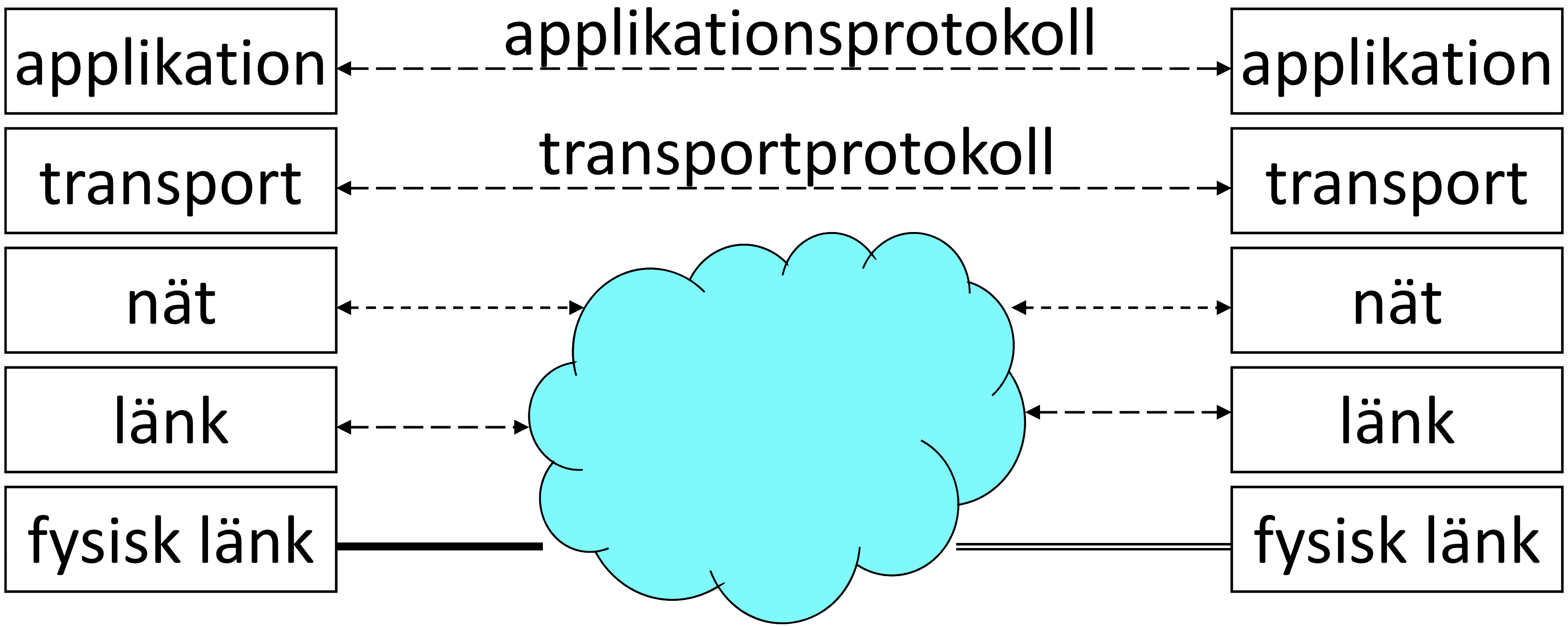
ETSF05 Internetprotokoll

PPP, NAT, IPv6, Go-Back-N Routing och Routingalgoritmer

Föreläsning 1

Jens Andersson

Nätmodellen



Sändarsidan

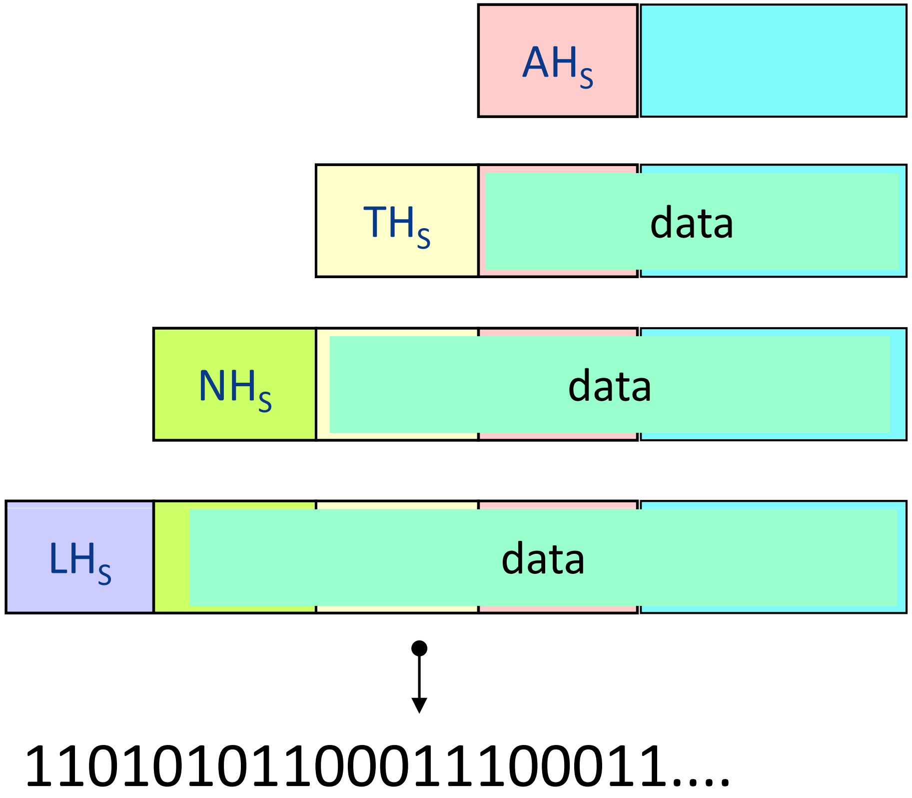
applikation

transport

nät

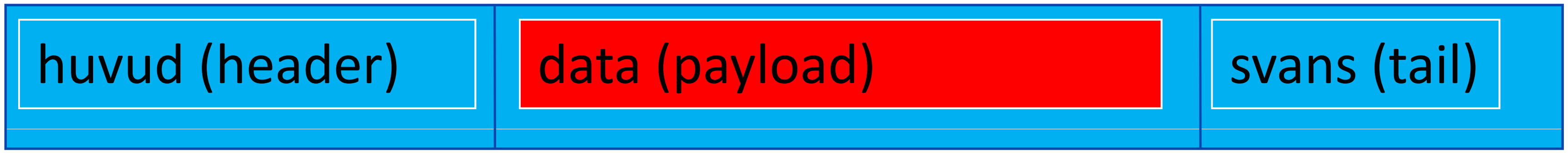
länk

fysisk länk



Datapaket

- När data skall skickas mellan två datorer delas den (oftast) först upp i mindre delar, så kallade paket.
- Ett paket består av upp till tre delar:
- huvud, data och svans



Huvud och svans innehåller kontrollinformation.

PPP

- Point-to-Point Protocol
- Bygger på HDLC
- Generellt
  - Kan användas för olika nätprotokoll
- Olika stödprotokoll
  - Autentisering
  - Länkkontroll (kontroll på länknivån)
  - Nätkontroll (kontroll på nätnivån)

Figure 11.32 *PPP frame format*

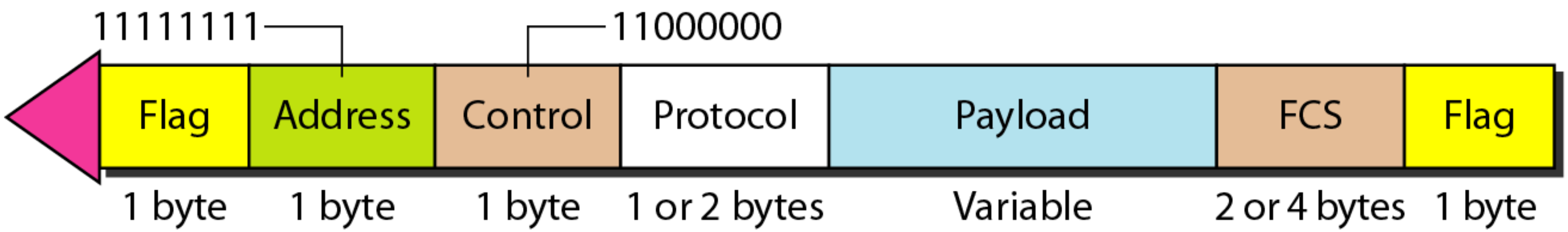


Figure 11.34 *Multiplexing in PPP*

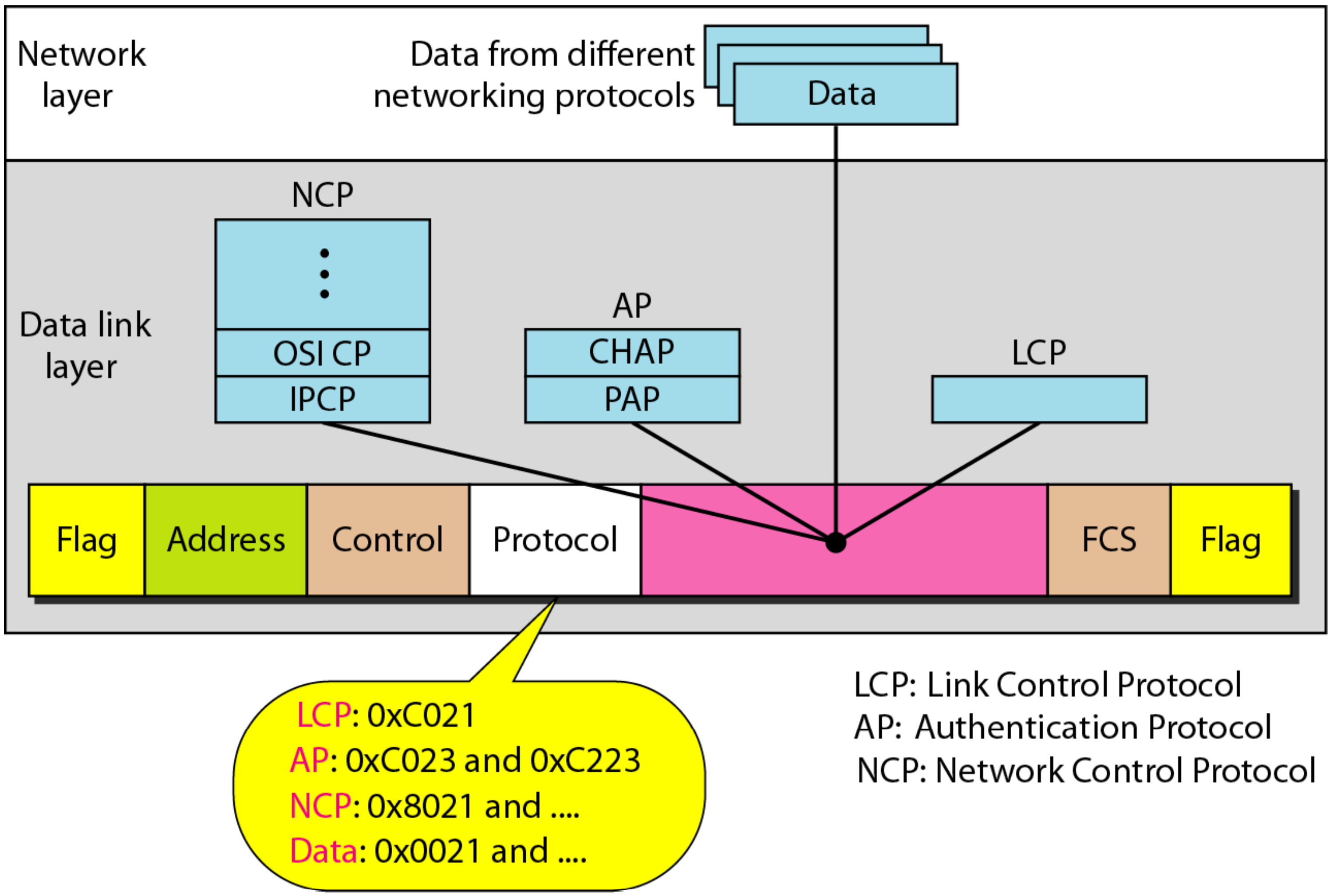


Figure 11.35 *LCP packet encapsulated in a frame*

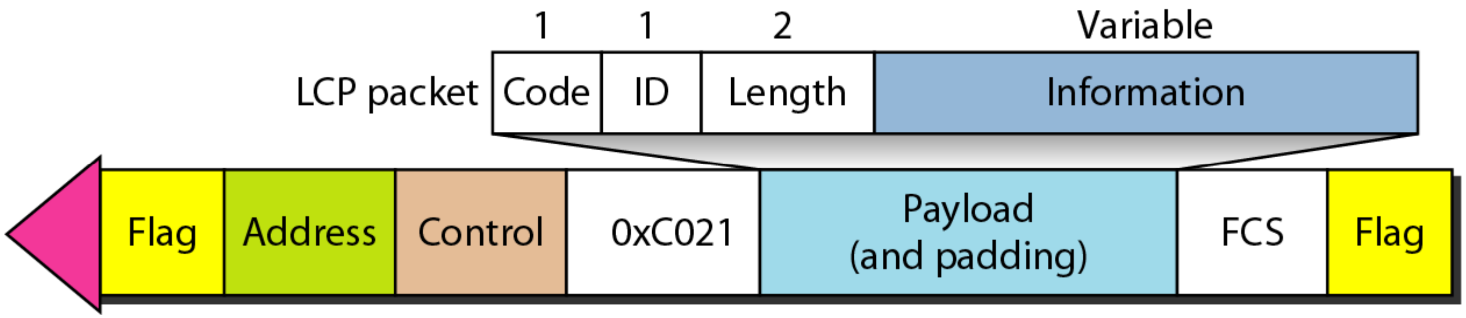


Table 11.2 *LCP packets*

Code	*Packet Type*	*Description*
0x01	Configure-request	Contains the list of proposed options and their values
0x02	Configure-ack	Accepts all options proposed
0x03	Configure-nak	Announces that some options are not acceptable
0x04	Configure-reject	Announces that some options are not recognized
0x05	Terminate-request	Request to shut down the line
0x06	Terminate-ack	Accept the shutdown request
0x07	Code-reject	Announces an unknown code
0x08	Protocol-reject	Announces an unknown protocol
0x09	Echo-request	A type of hello message to check if the other end is alive
0x0A	Echo-reply	The response to the echo-request message
0x0B	Discard-request	A request to discard the packet

Table 11.3 *Common options*

Option	*Default*
Maximum receive unit (payload field size)	1500
Authentication protocol	None
Protocol field compression	Off
Address and control field compression	Off

Figure 11.36 *PAP packets encapsulated in a PPP frame*

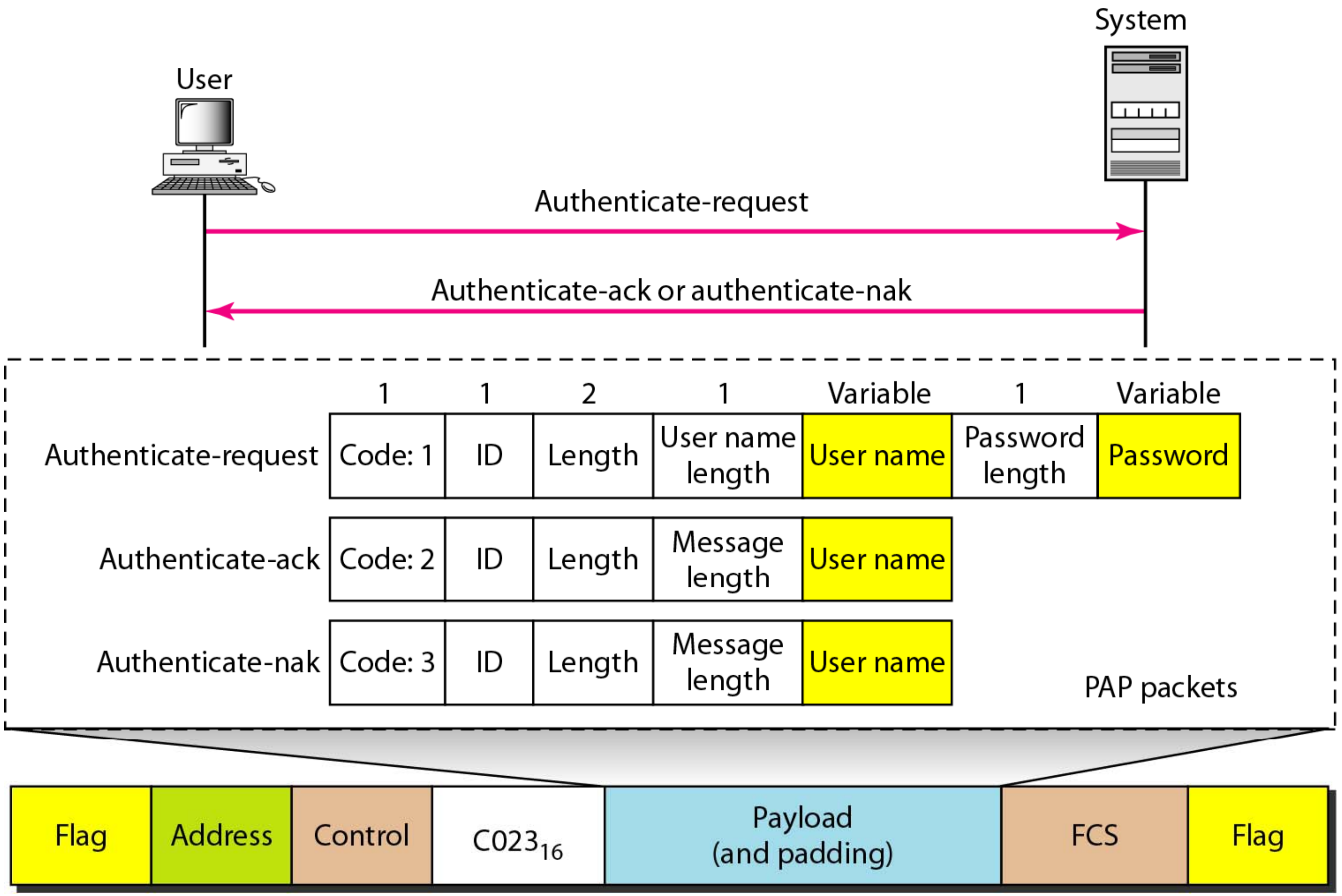


Figure 11.37 *CHAP packets encapsulated in a PPP frame*

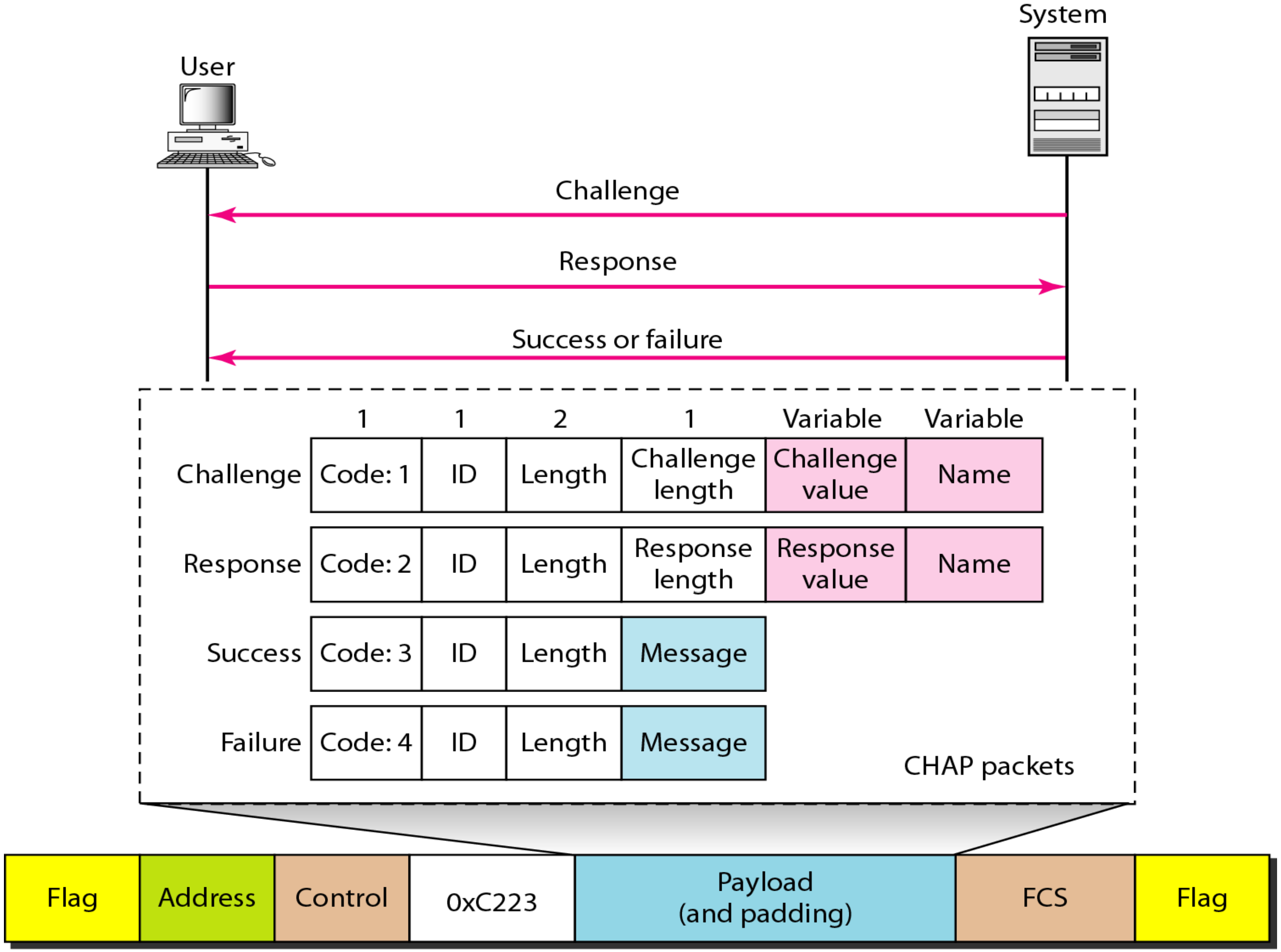


Figure 11.38 *IPCP packet encapsulated in PPP frame*

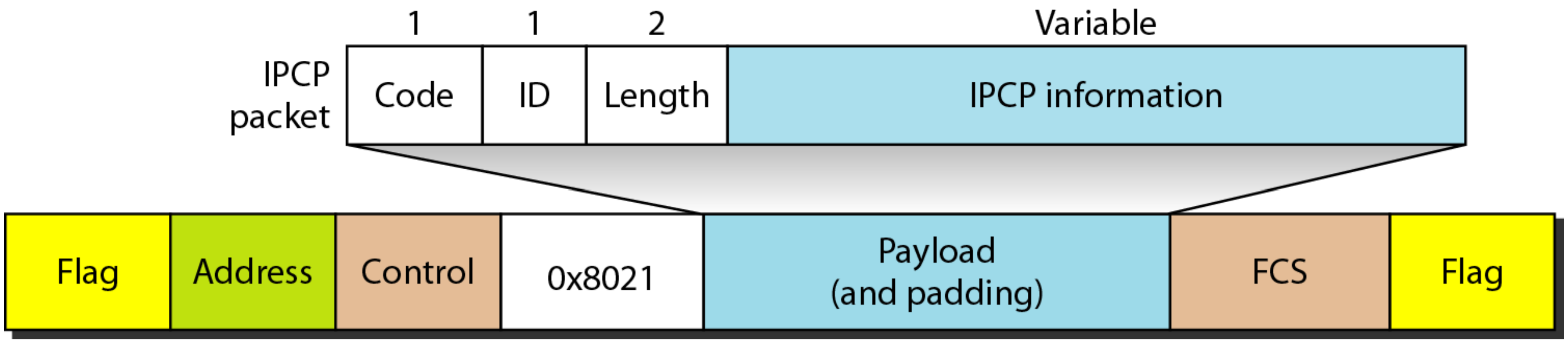


Table 11.4 *Code value for IPCP packets*

Code	*IPCP Packet*
0x01	Configure-request
0x02	Configure-ack
0x03	Configure-nak
0x04	Configure-reject
0x05	Terminate-request
0x06	Terminate-ack
0x07	Code-reject

Figure 11.39 *IP datagram encapsulated in a PPP frame*

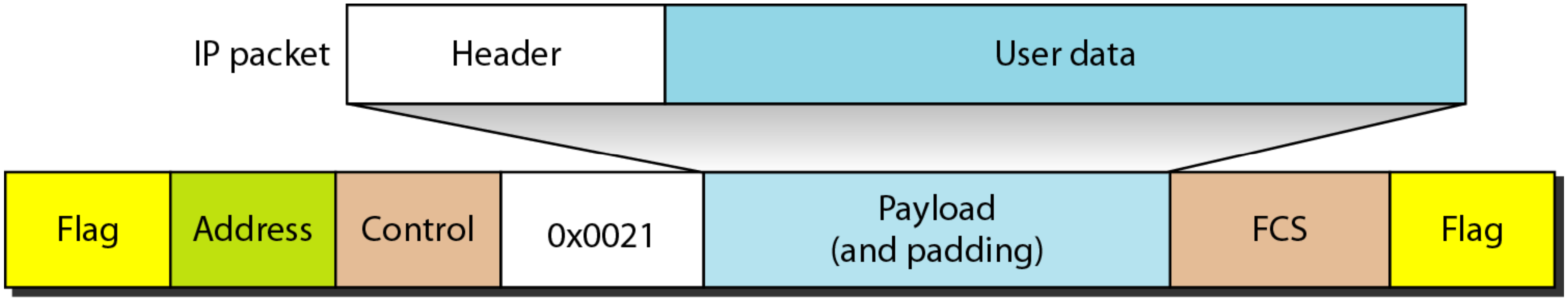


Figure 11.40 *Multilink PPP*

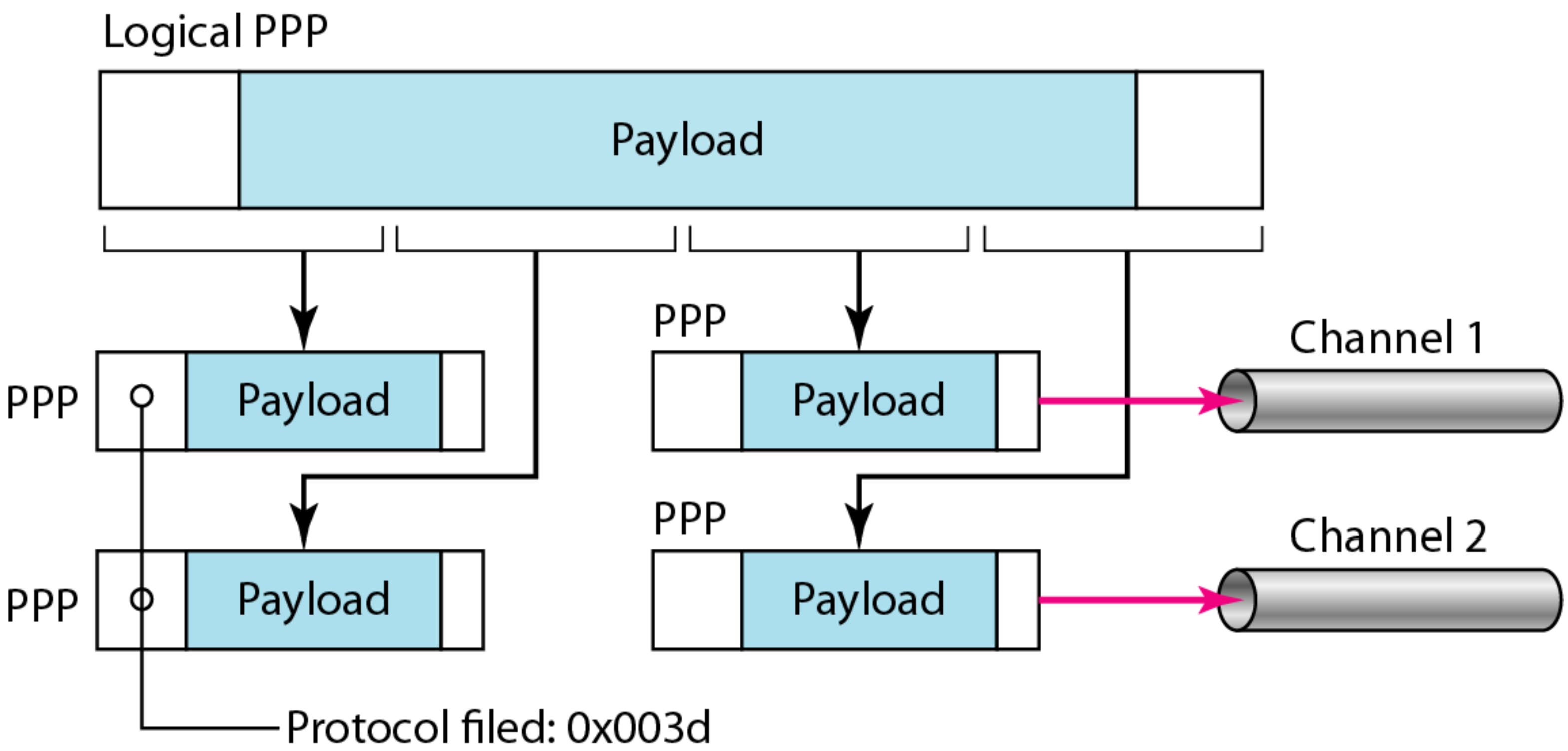


Figure 11.41 *An example*

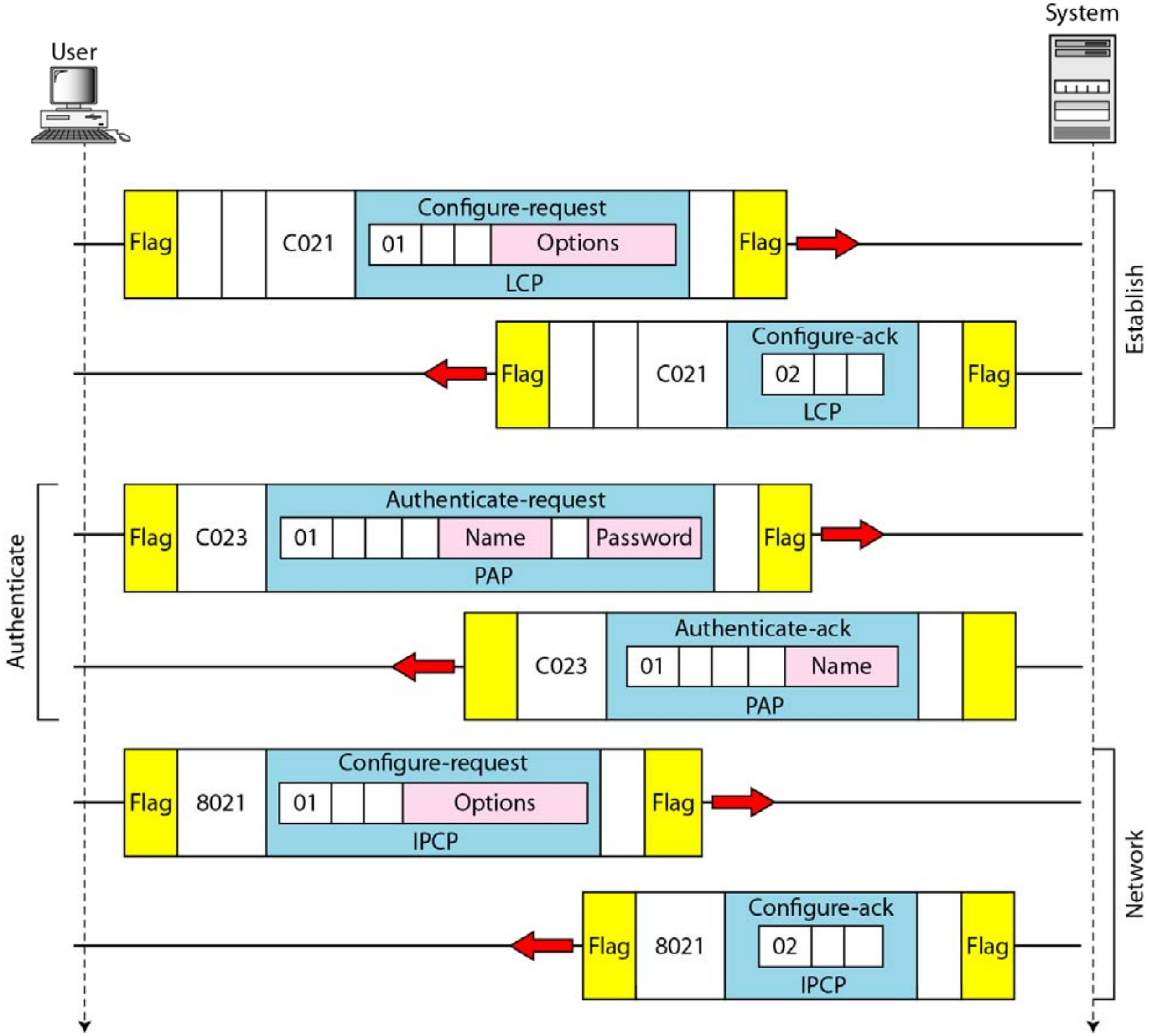


Figure 11.41 *An example (continued)*

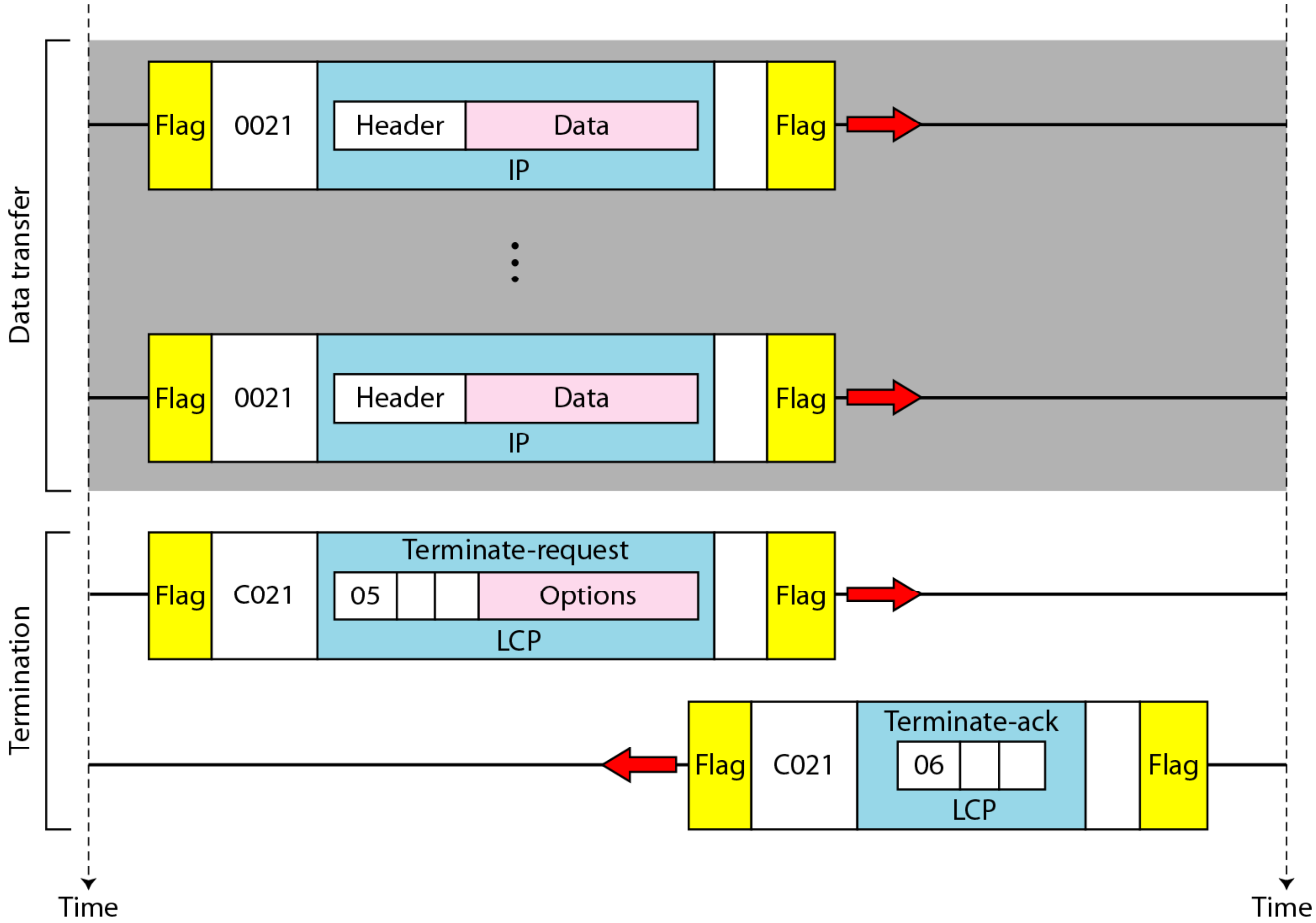
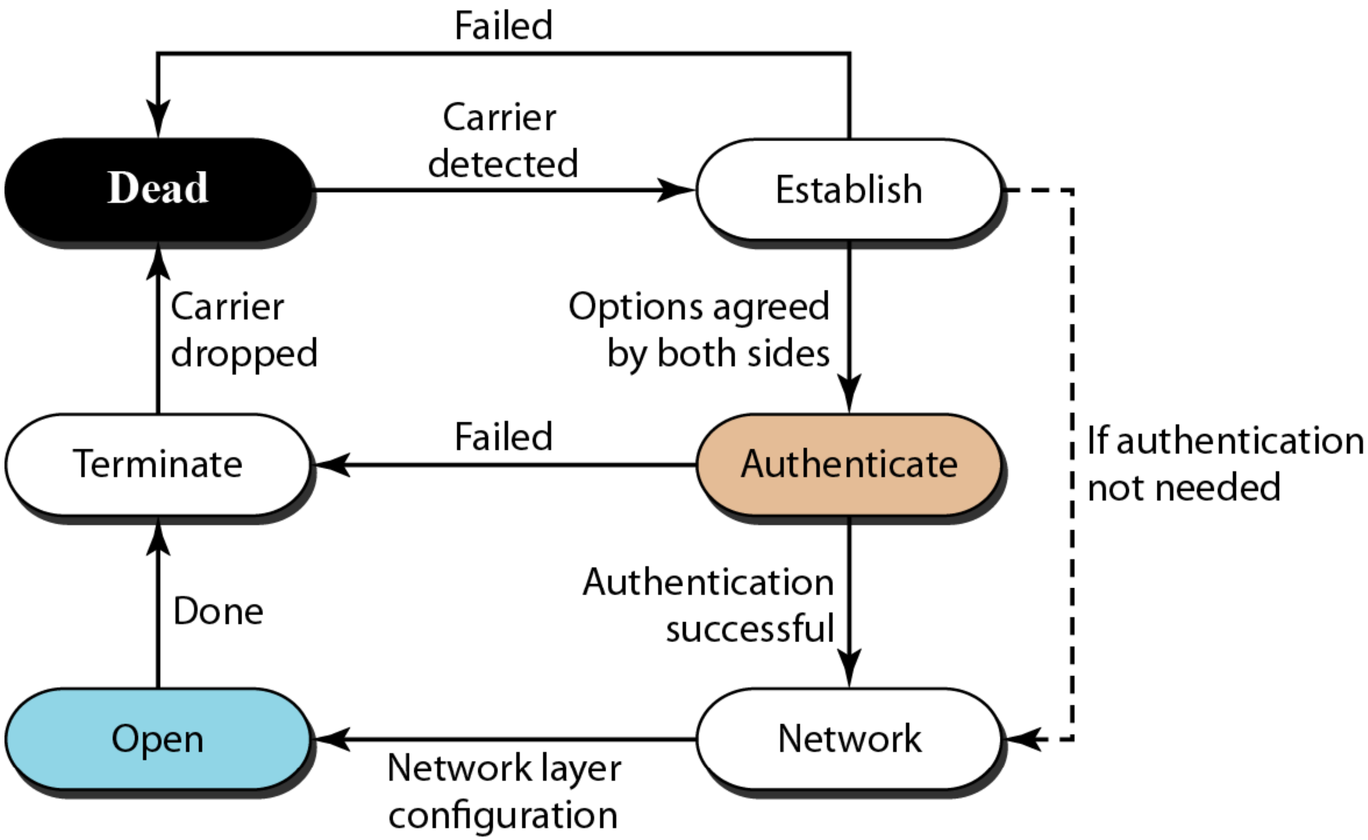


Figure 11.33 *Transition phases*



NAT

- Network Address Translation
- Räddar Internet från adressbrist
- Möjligt återanvända IP-adresser
- Ändrar IP-adresser/TCP&UDP-portar i paket

Table 19.3 *Addresses for private networks*

Range			*Total*
10.0.0.0	to	10.255.255.255	2^{24}
172.16.0.0	to	172.31.255.255	2^{20}
192.168.0.0	to	192.168.255.255	2^{16}

Figure 19.10 *A NAT implementation*

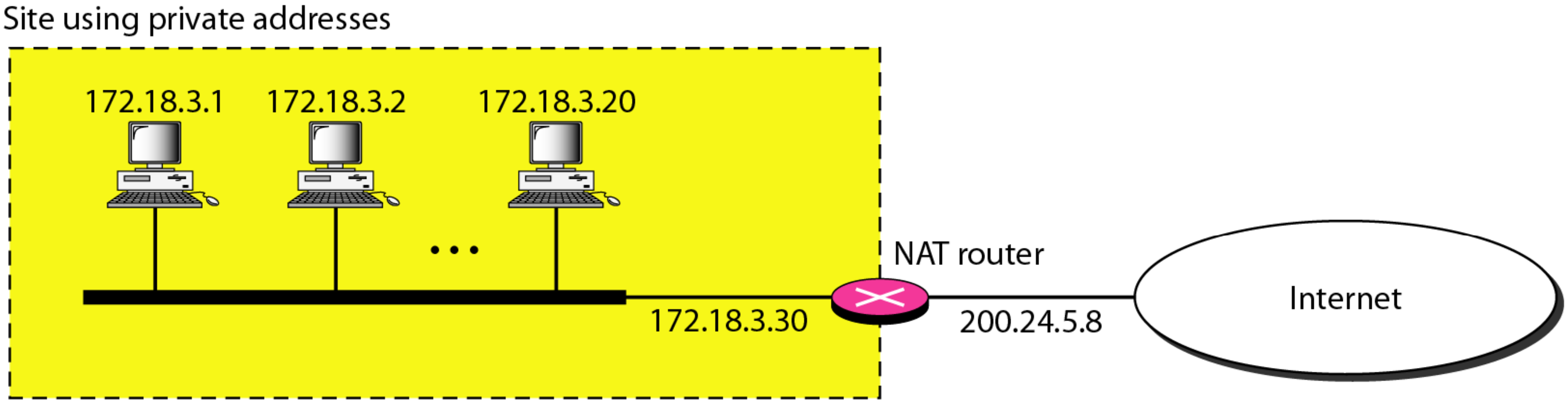


Figure 19.11 *Addresses in a NAT*

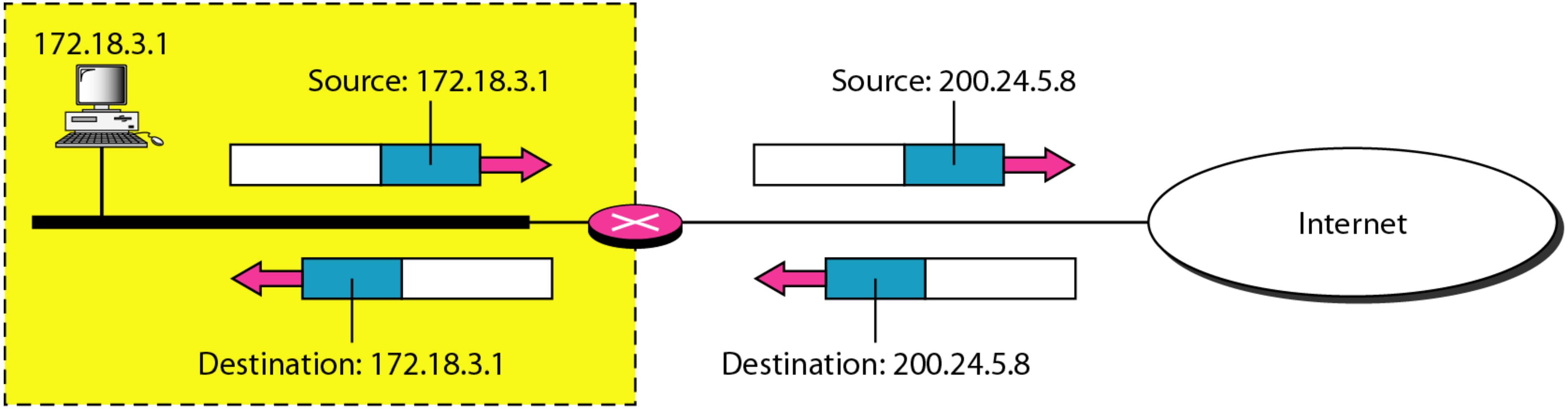


Figure 19.12 *NAT address translation*

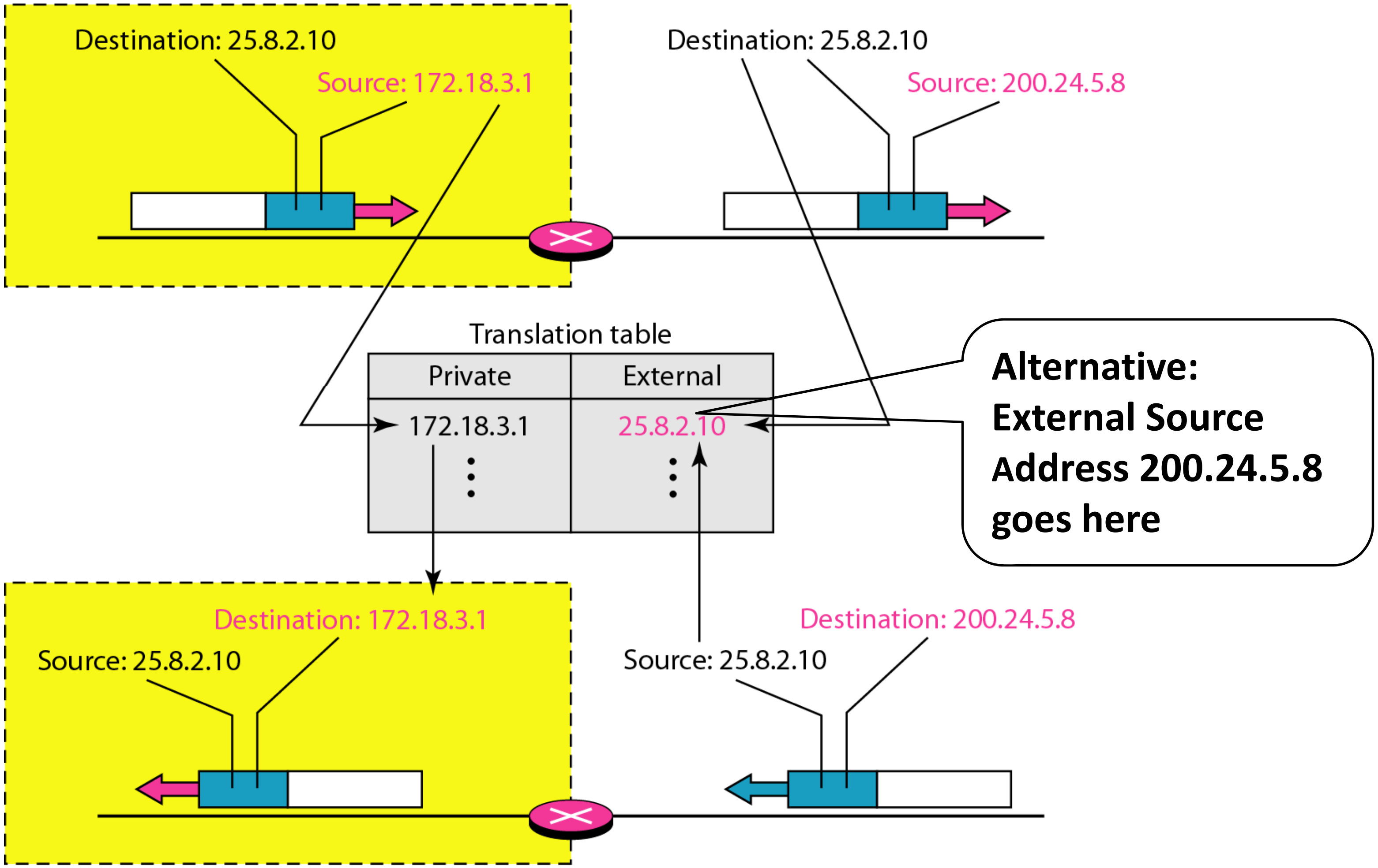


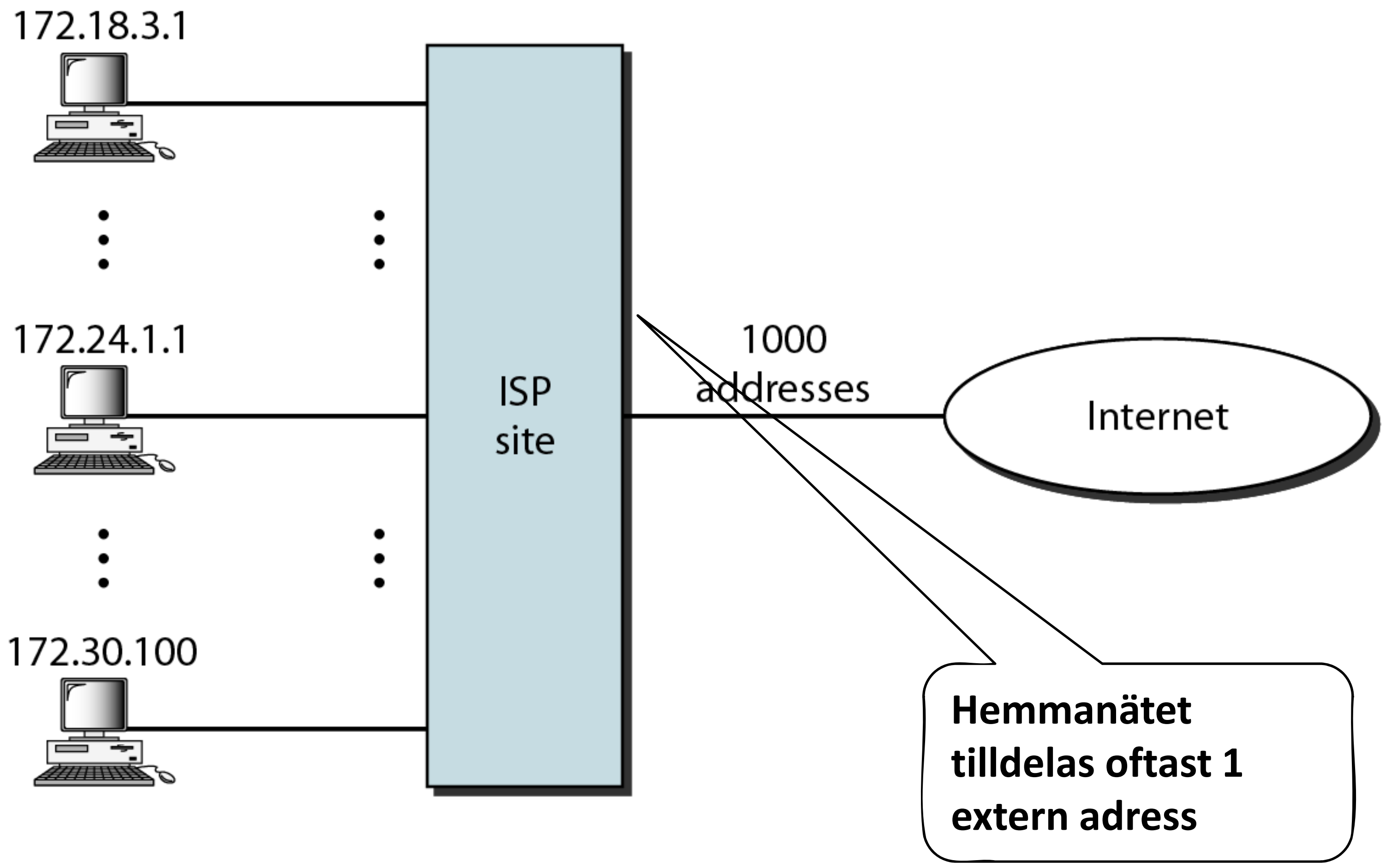
Table 19.4 *Five-column translation table*

Private Address	*Private Port*	*External Address*	*External Port*	*Transport Protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
. . .	. . .	. . .	. . .	. . .

Alternative: External source address 200.24.5.8 goes here

Alternative: External source port goes here

Figure 19.13 *An ISP and NAT*



IPv6

- IPv4 adresser tar slut
  - 2^{32} adresser
  - Orättvist fördelade i världen
  - Variabel header
- IPv6
  - 2^{128} adresser
  - Konceptet flöde (både TCP och UDP)
  - Fast headerstorlek
  - Kolla fragmentering redan vid källan

Figure 19.14 *IPv6 address in binary and hexadecimal colon notation*

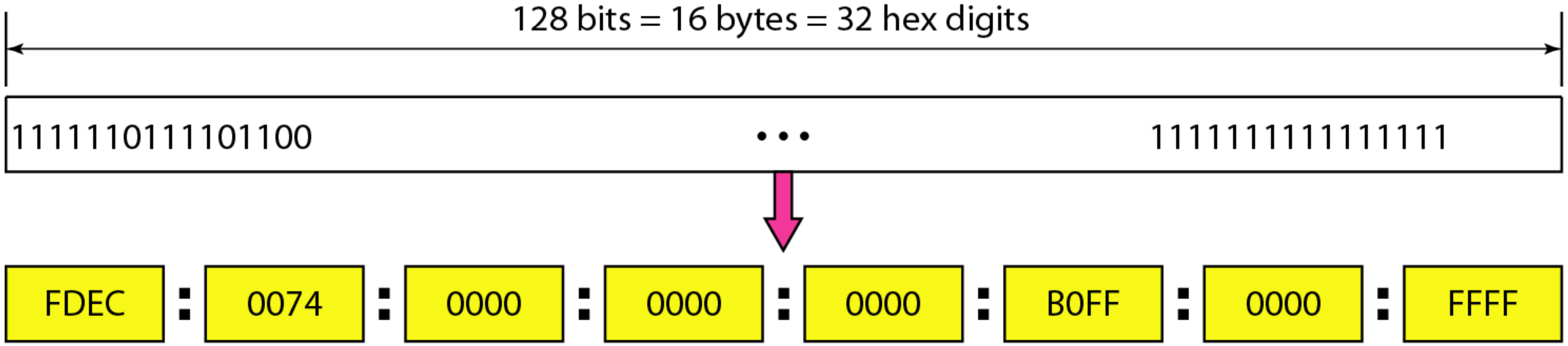


Figure 19.15 *Abbreviated IPv6 addresses*

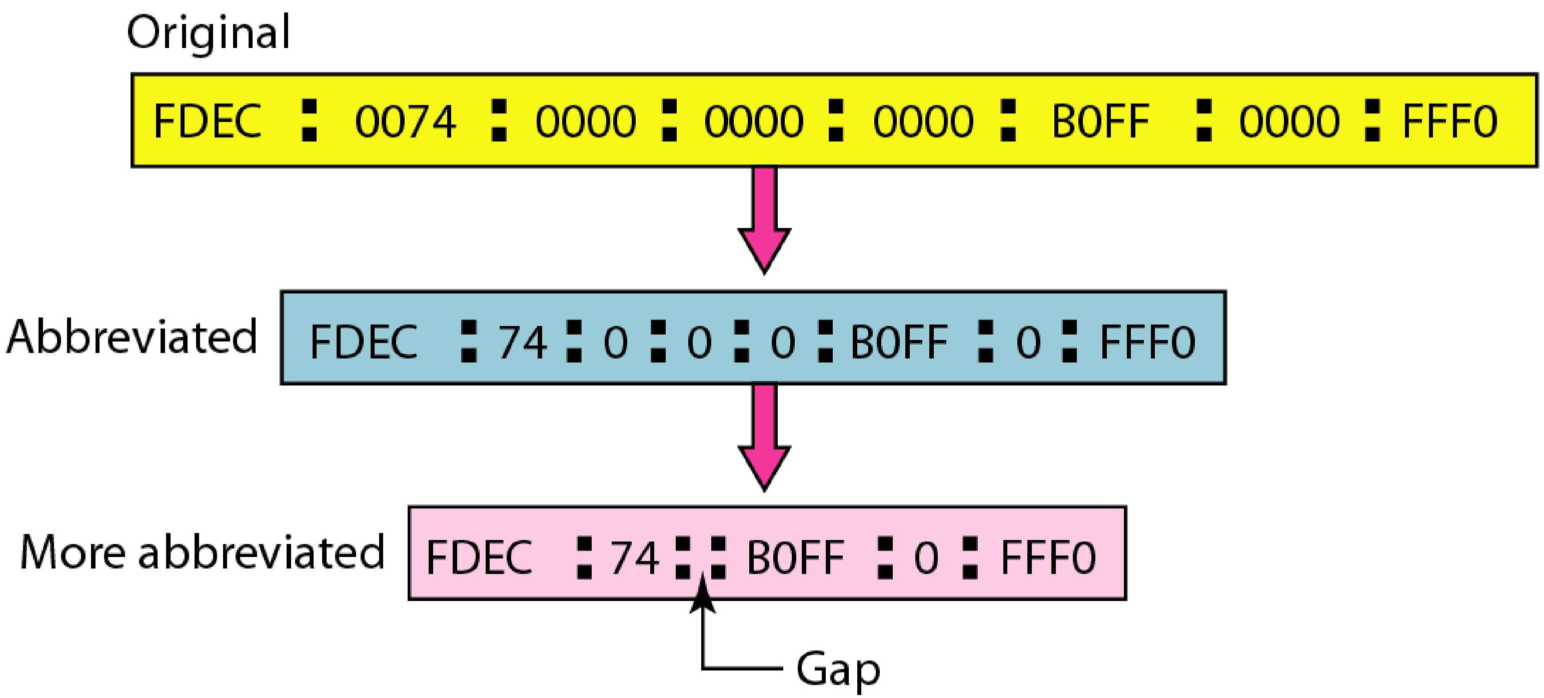


Table 19.5 *Type prefixes for IPv6 addresses*

Type Prefix	*Type*	*Fraction*
0000 0000	Reserved	1/256
0000 0001	Unassigned	1/256
0000 001	ISO network addresses	1/128
0000 010	IPX (Novell) network addresses	1/128
0000 011	Unassigned	1/128
0000 1	Unassigned	1/32
0001	Reserved	1/16
001	Reserved	1/8
010	**Provider-based unicast addresses**	**1/8**

Table 19.5 *Type prefixes for IPv6 addresses (continued)*

Type Prefix	*Type*	*Fraction*
011	Unassigned	1/8
100	Geographic-based unicast addresses	1/8
101	Unassigned	1/8
110	Unassigned	1/8
1110	Unassigned	1/16
1111 0	Unassigned	1/32
1111 10	Unassigned	1/64
1111 110	Unassigned	1/128
1111 1110 0	Unassigned	1/512
1111 1110 10	Link local addresses	1/1024
1111 1110 11	Site local addresses	1/1024
1111 1111	Multicast addresses	1/256

Figure 19.16 *Prefixes for provider-based unicast address*

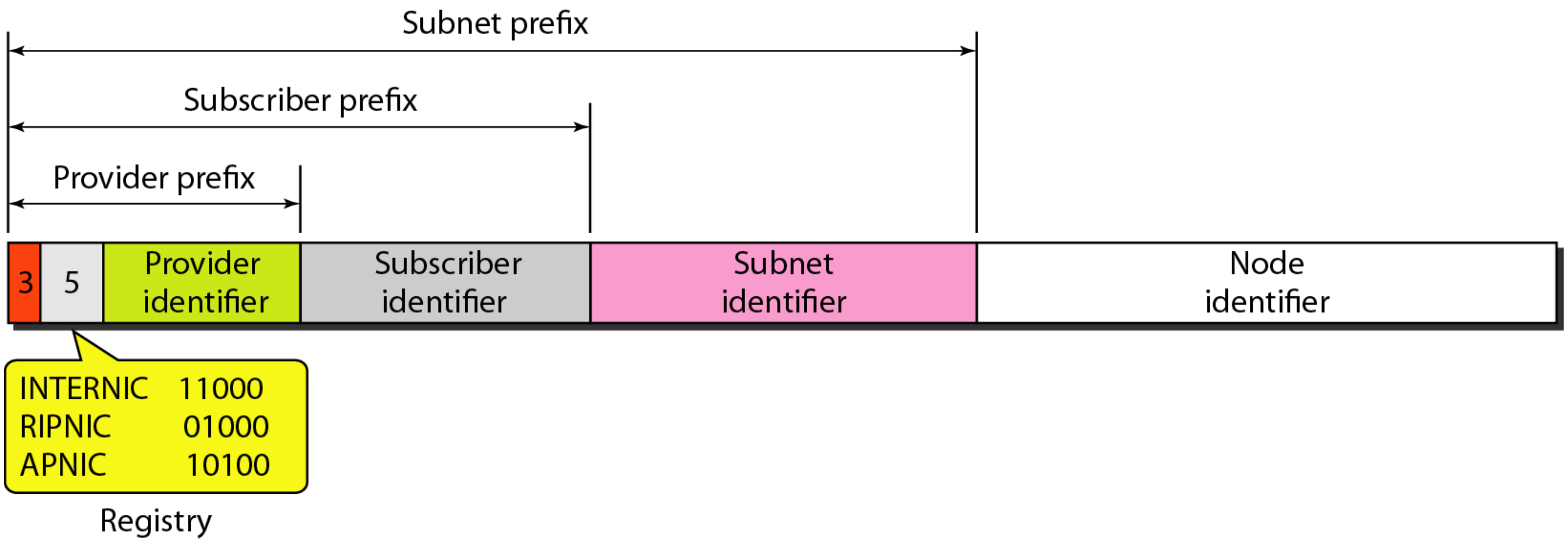


Figure 19.17 *Multicast address in IPv6*

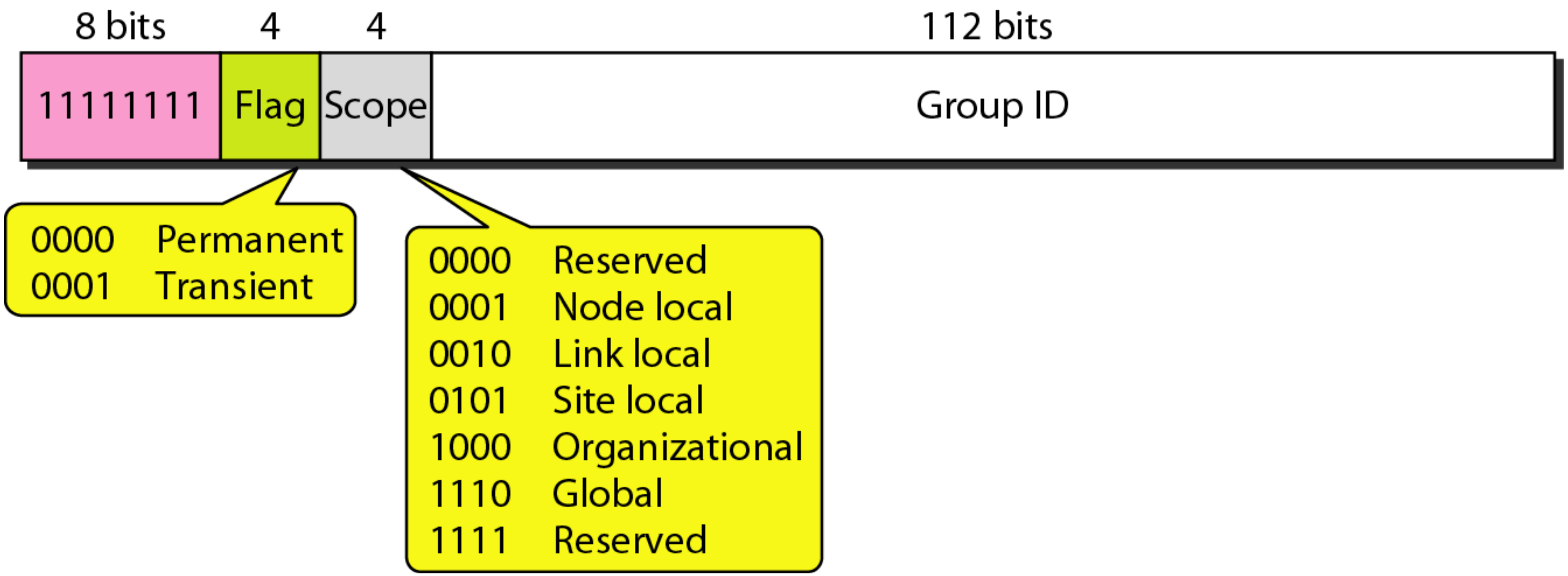


Figure 19.18 *Reserved addresses in IPv6*

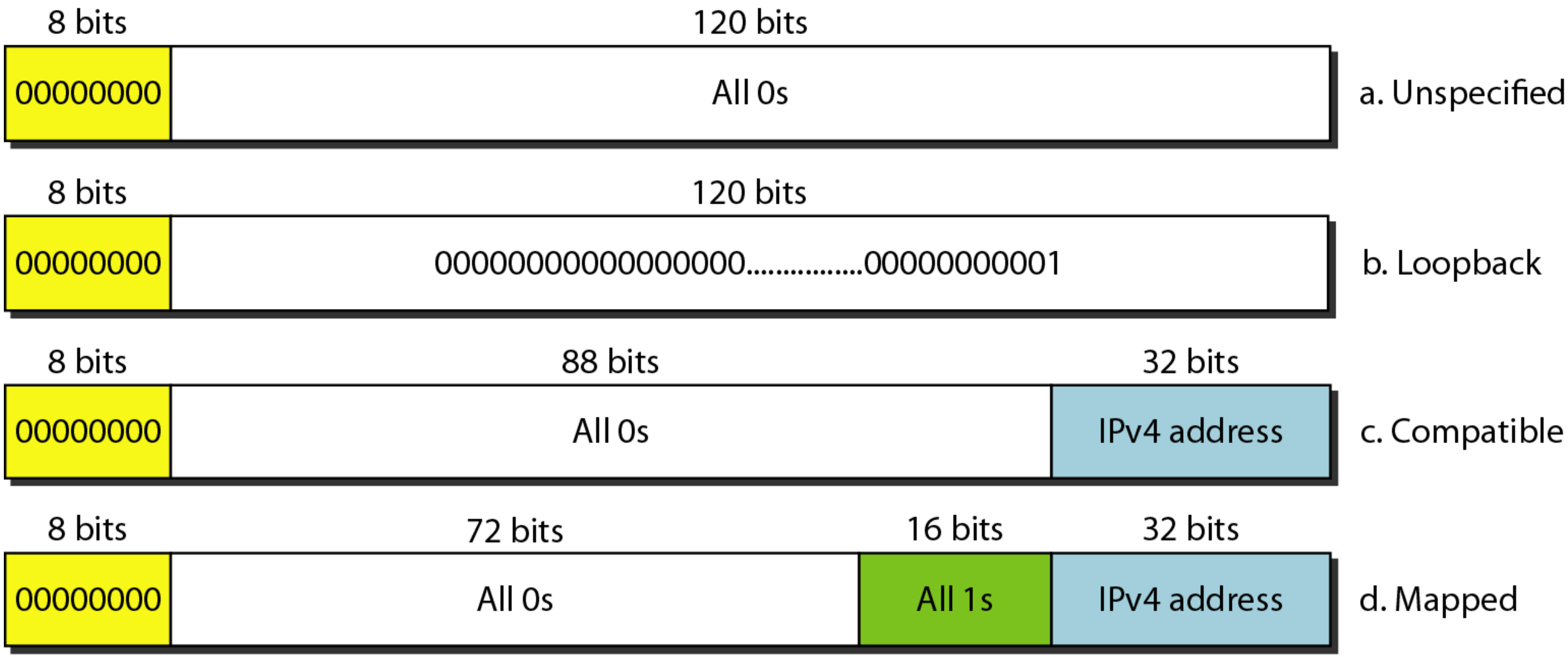
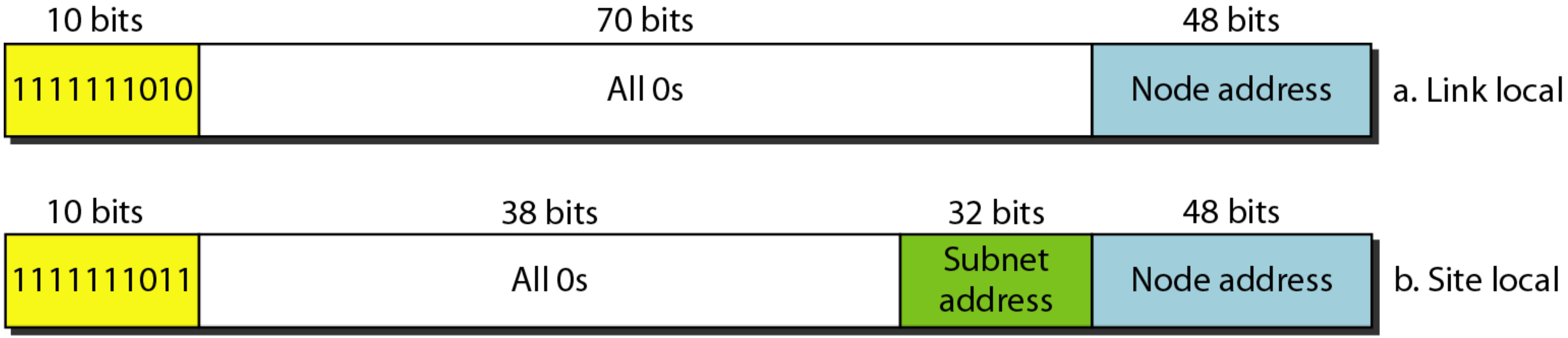


Figure 19.19 *Local addresses in IPv6*



Go-Back-N ARQ

- Repetion
- Behövs när vi diskuterar TCP i nästa lp

Figure 11.14 *Design of Go-Back-N ARQ*

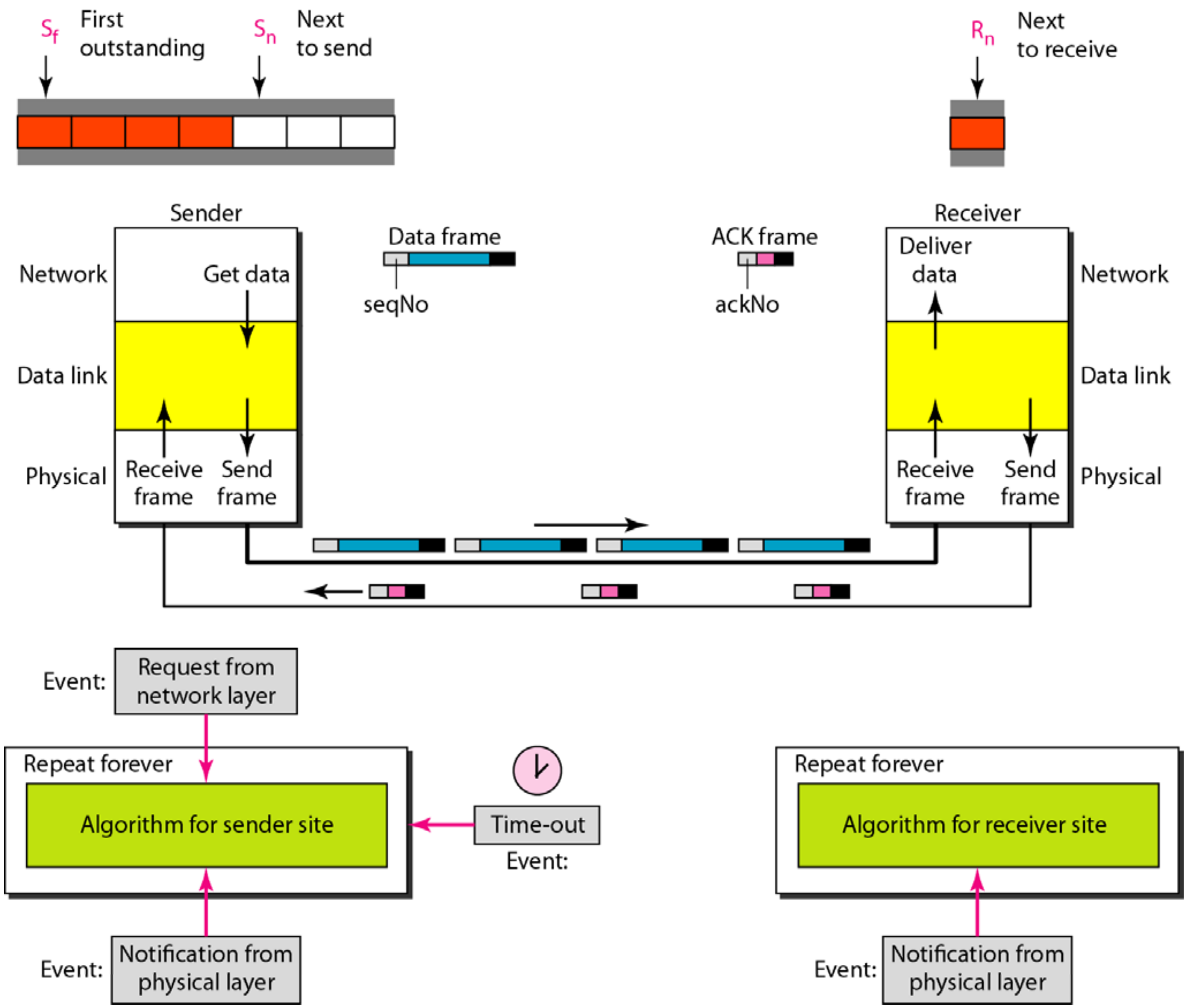
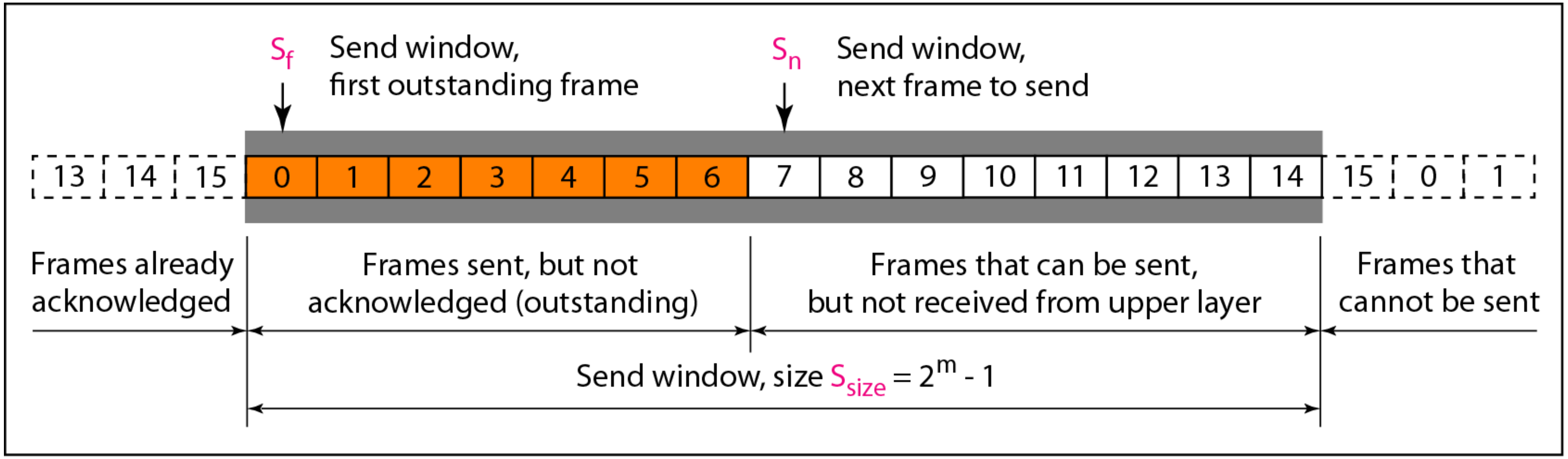
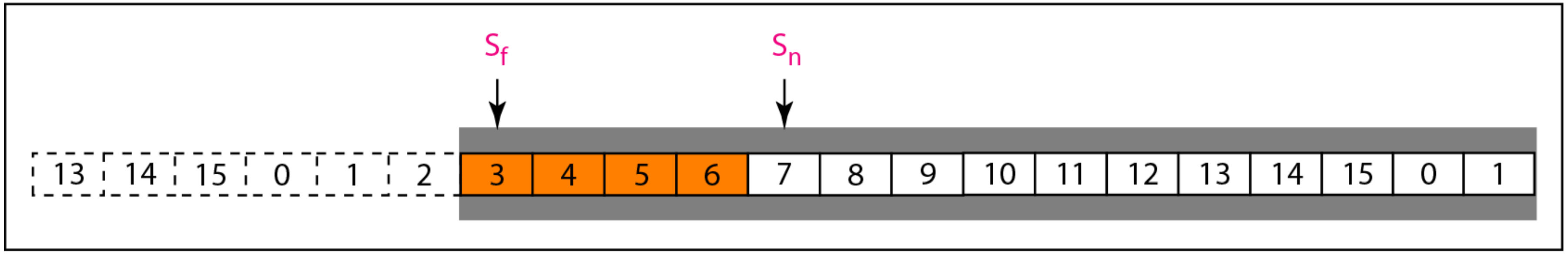


Figure 11.12 *Send window for Go-Back-N ARQ*

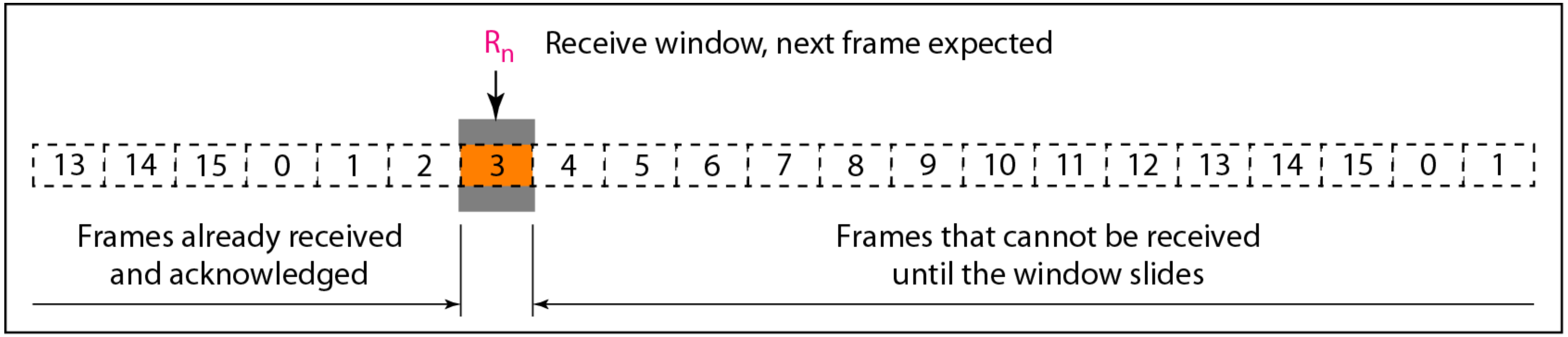


a. Send window before sliding

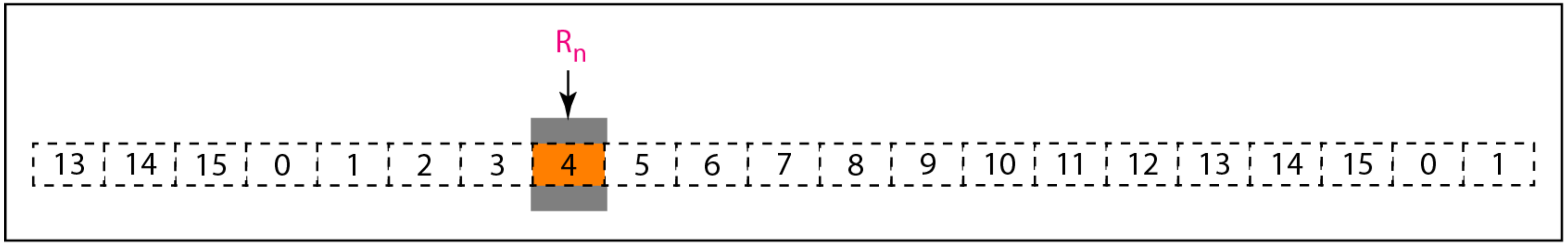


b. Send window after sliding

Figure 11.13 *Receive window for Go-Back-N ARQ*



a. Receive window



b. Window after sliding

Figure 11.16 *Flow diagram for Example 11.6*

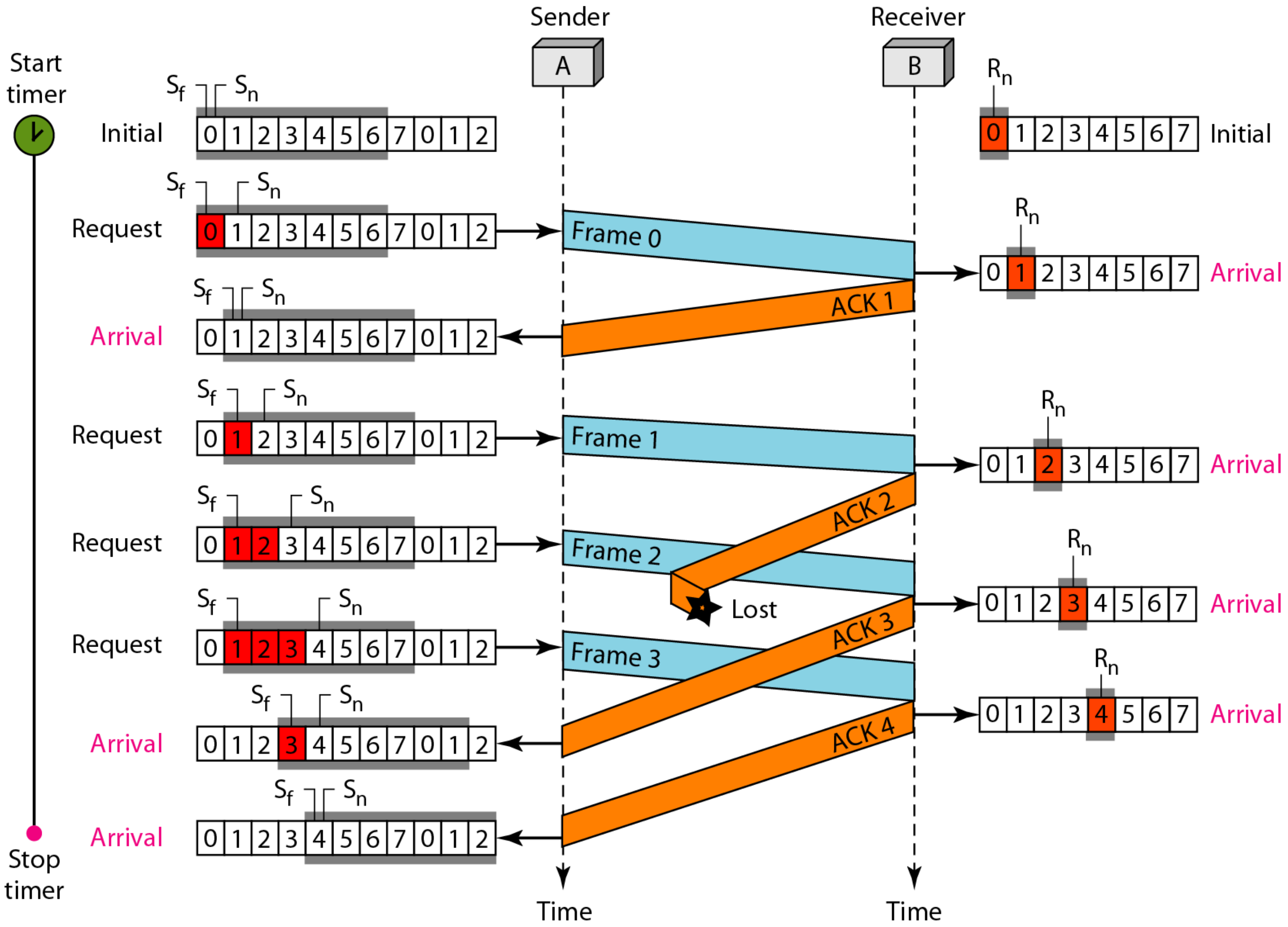


Figure 11.17 *Flow diagram for Example 11.7*

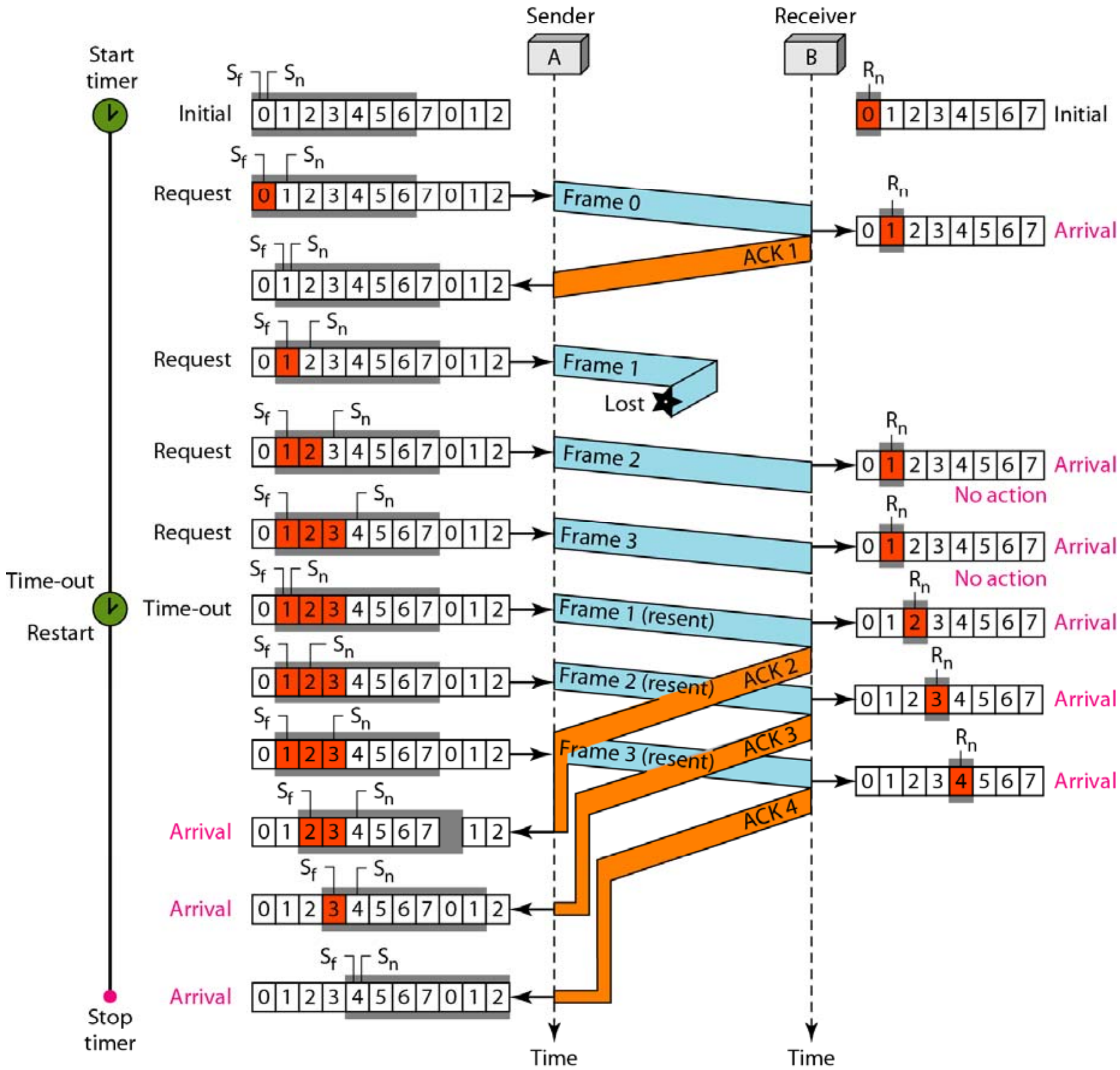
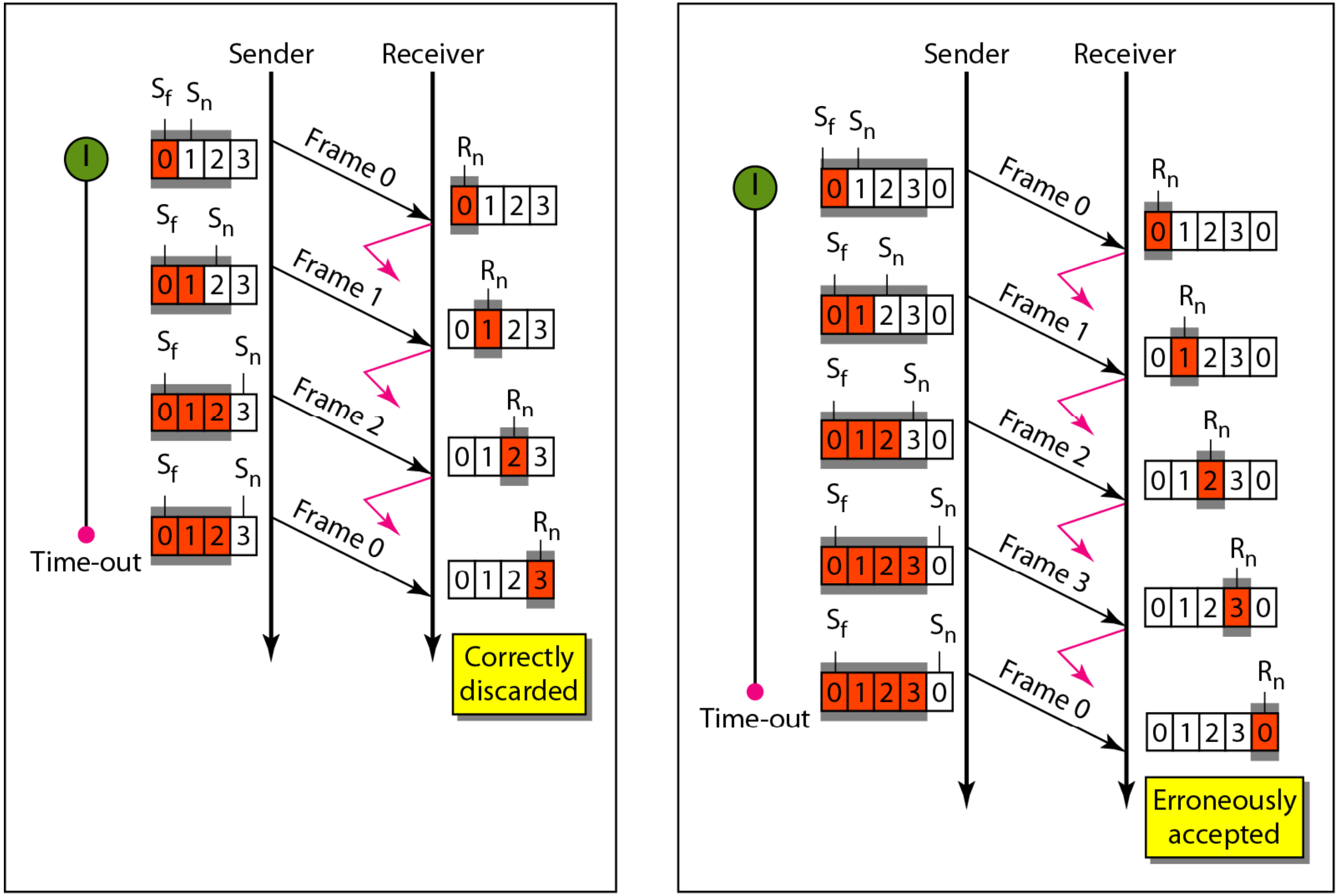


Figure 11.15 *Window size for Go-Back-N ARQ*



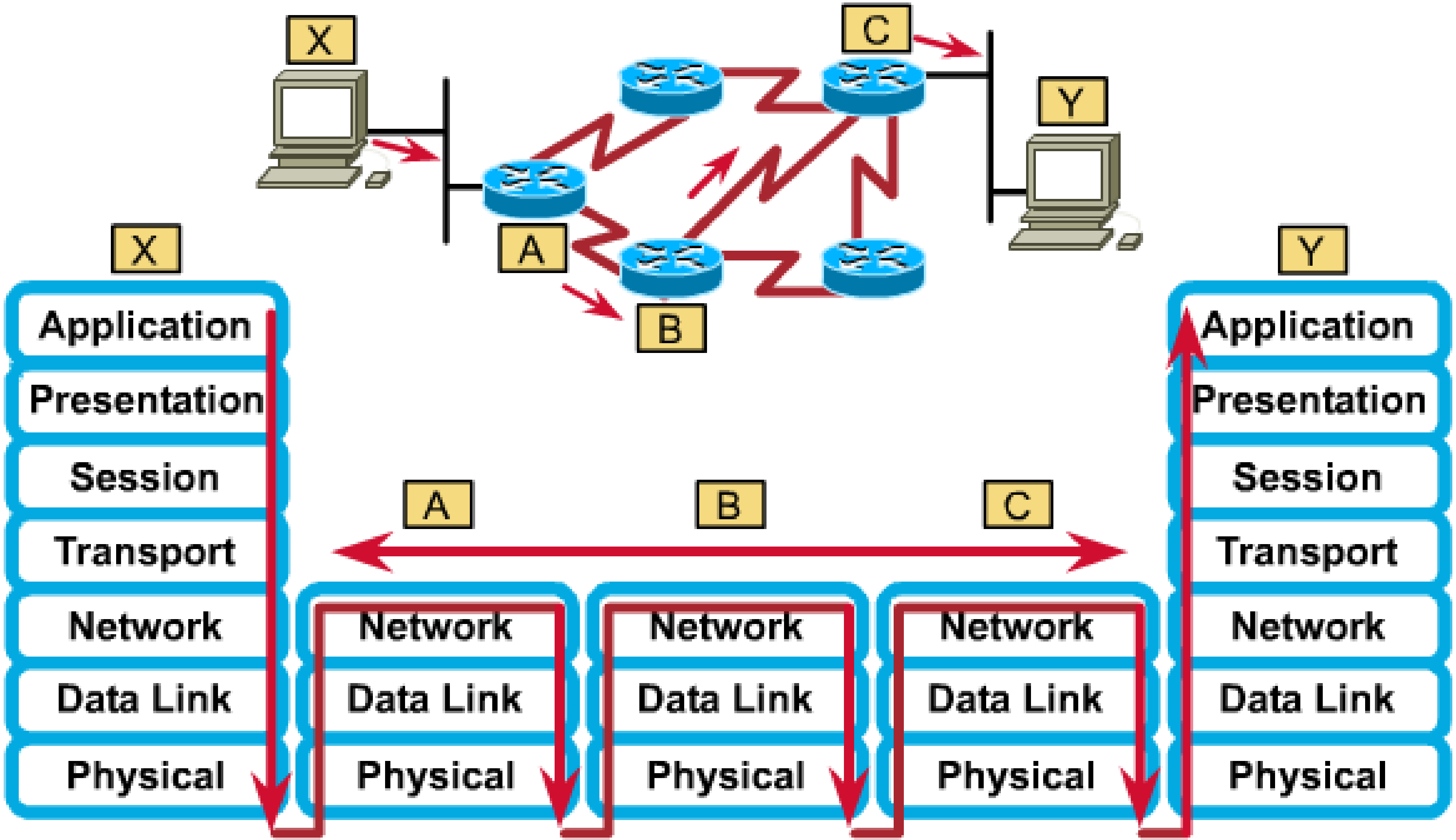
a. Window size < 2^m

b. Window size = 2^m

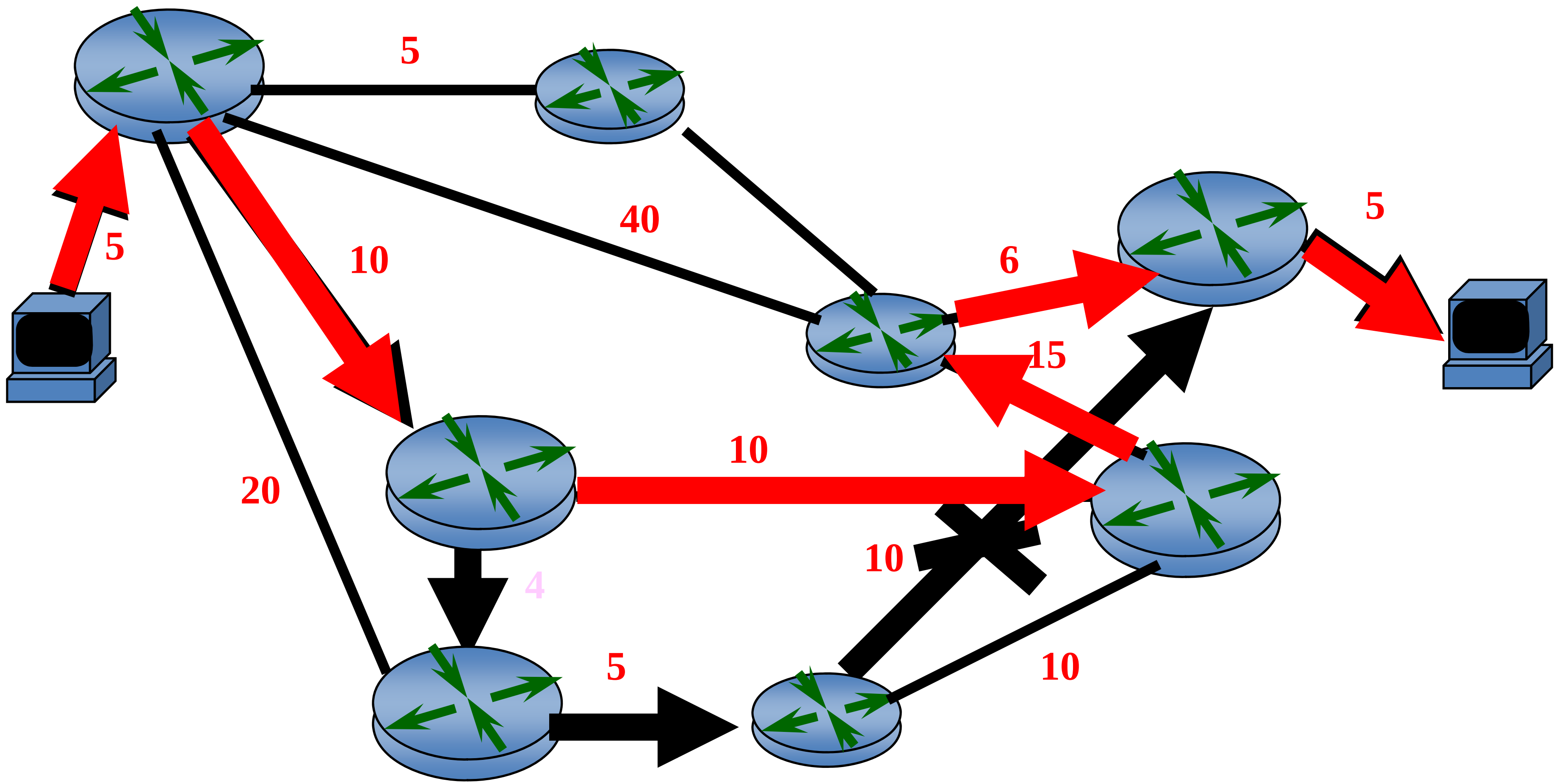
Routing och routingalgoritmer

- Distance Vector Routing
- Link State Routing

Network protocol operation



Task: Choosing an Optimal Path

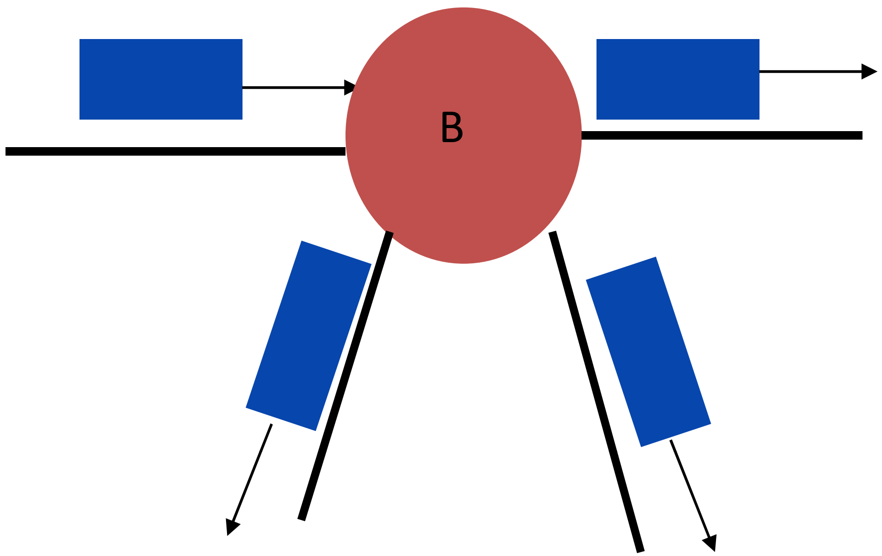


Routing Technologies

- No “intelligence”
- Centralized
- Distributed

Flooding

- Forward packet on all outgoing interfaces
  - Exclude incoming interface
- Hop-count prevents looping



Centralized routing technology

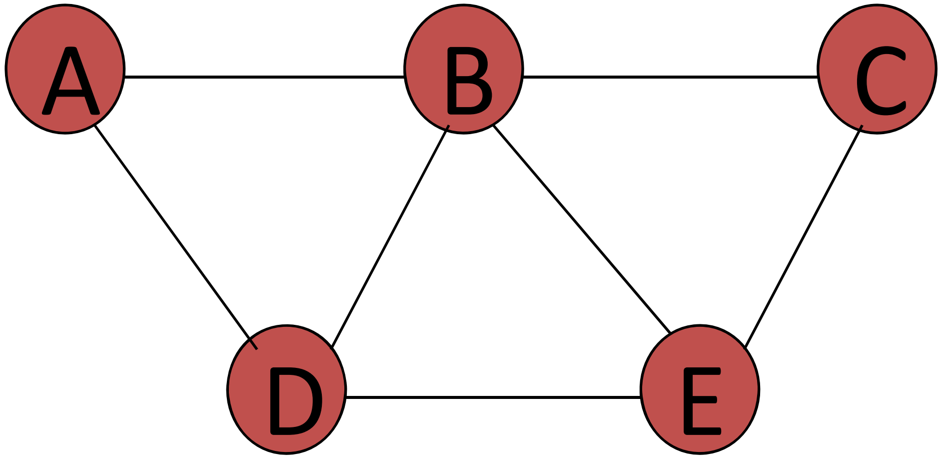
- Database and algorithm centralized
  - Network equipment informs central function
- Packet forwarding distributed
  - of course!?

Distributed routing technology

- Routing process distributed among all routers
- Two approaches
  - Distance Vector
    - Each nodes best-path information is distributed to its neighbours
    - Best end-to-end paths result from composition of all next-hop choices
    - Simple, low demands on processor and memory
  - Link State
    - Local topology information is flooded to all nodes
    - Best end-to-end paths are computed locally at each router
    - More complicated, high demands on processor and memory

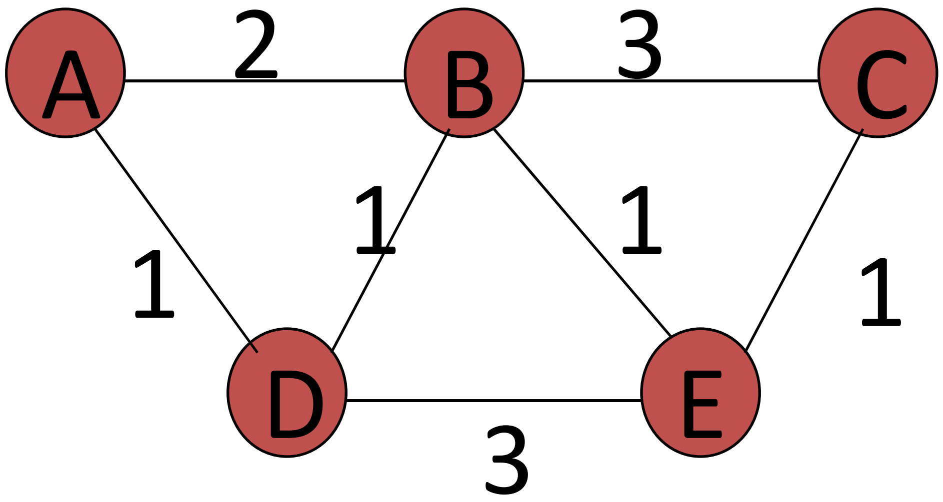
Least-hop path

- Path with least number of hops is best



Least-cost path

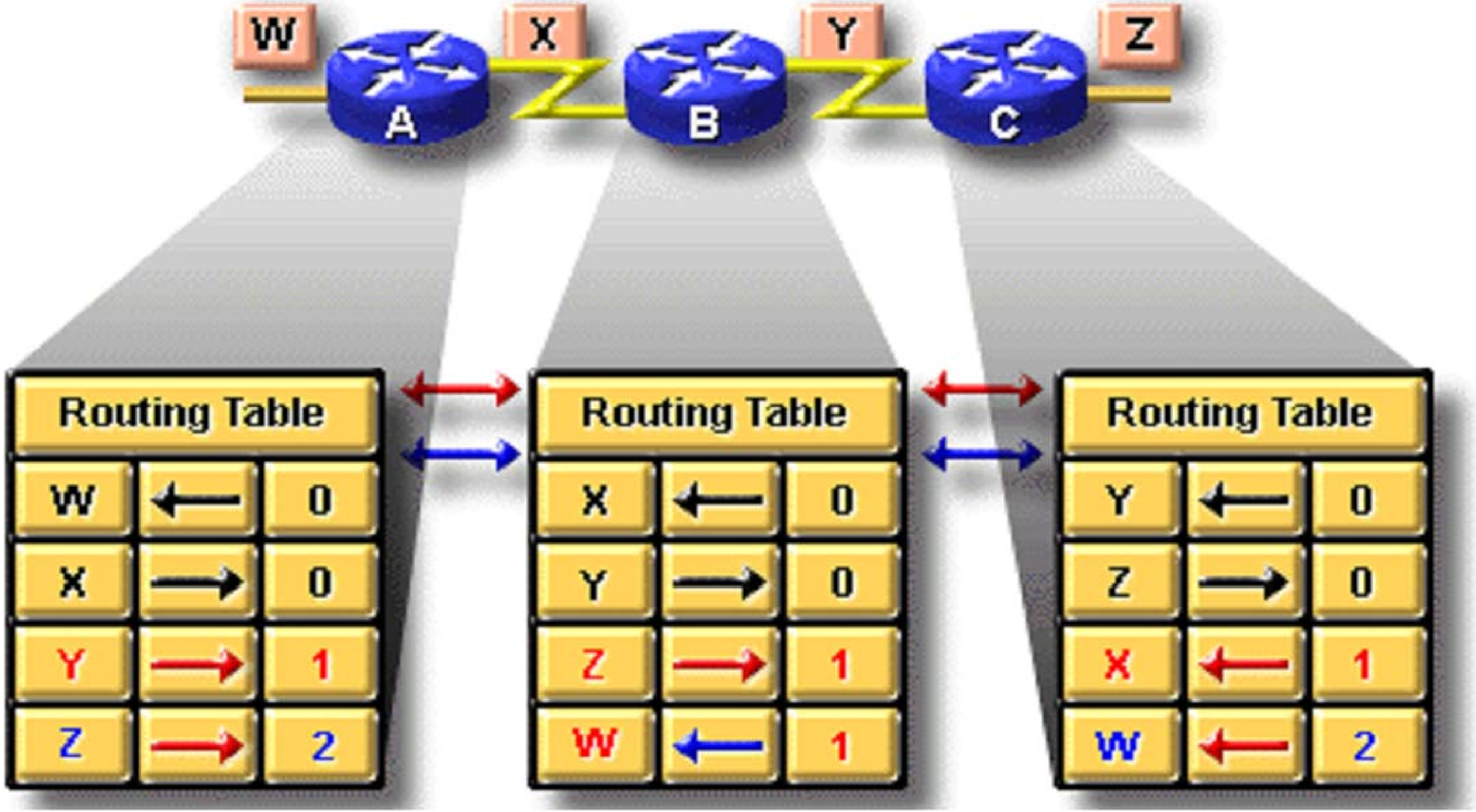
- Best path is path with least aggregated cost



The Link Metric

- Possible metrics
  - hop count
  - inverse of the link bandwidth
  - delay
  - dynamically calculated
  - administratively assigned
  - combination

Distance vector routing



Link state routing

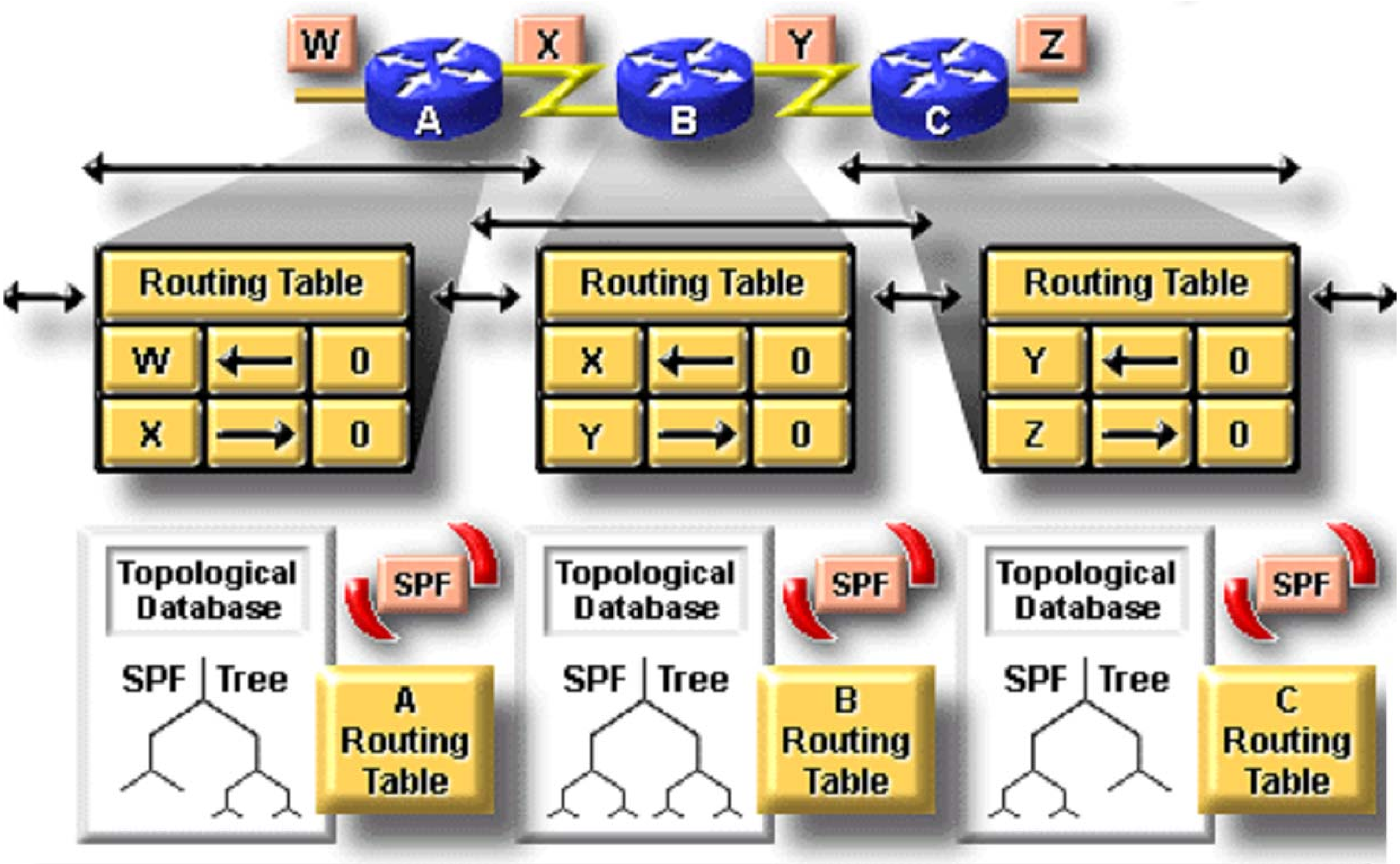


Figure 21-17

Example of an Internet

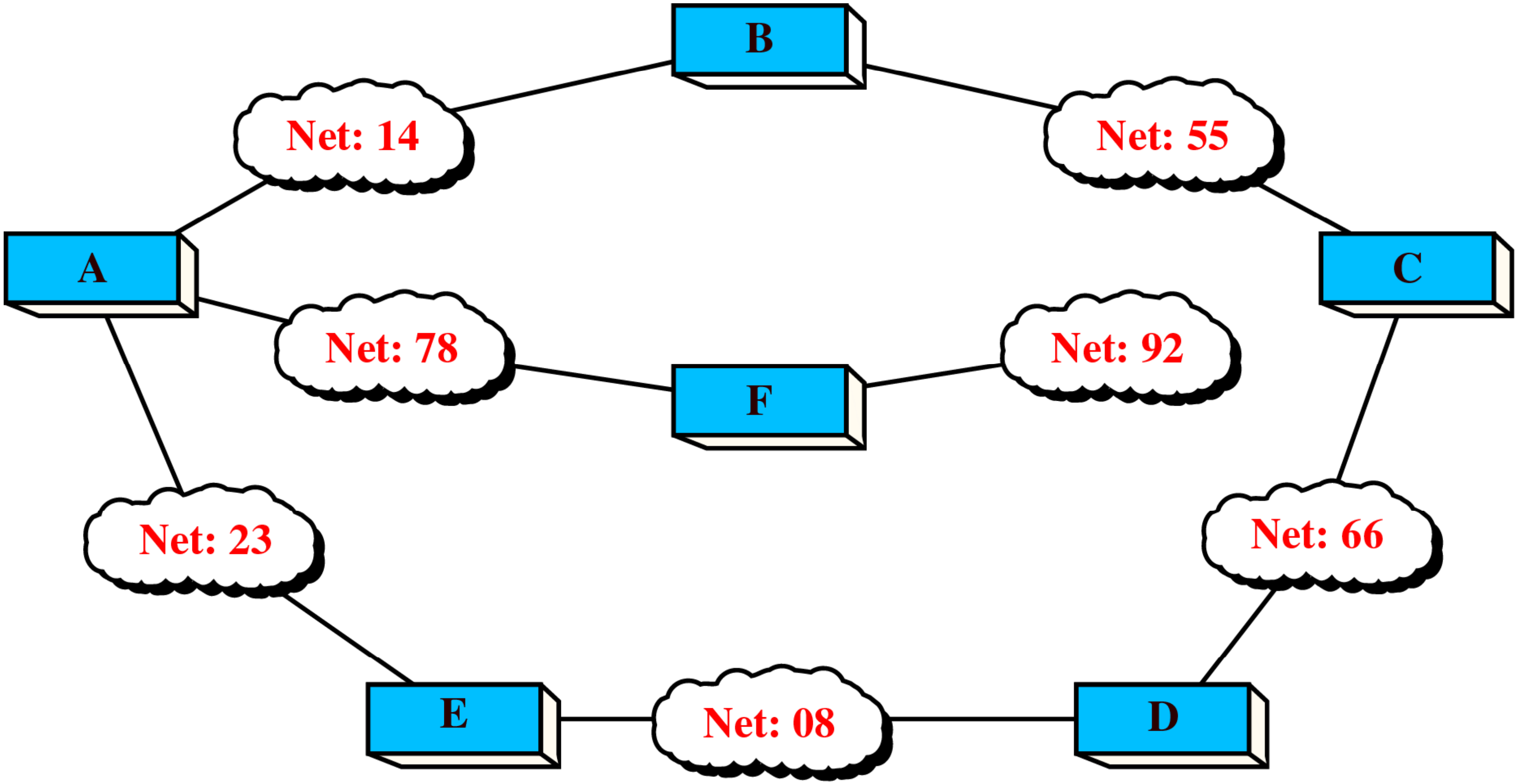


Figure 21-18

The Concept of Distance Vector Routing

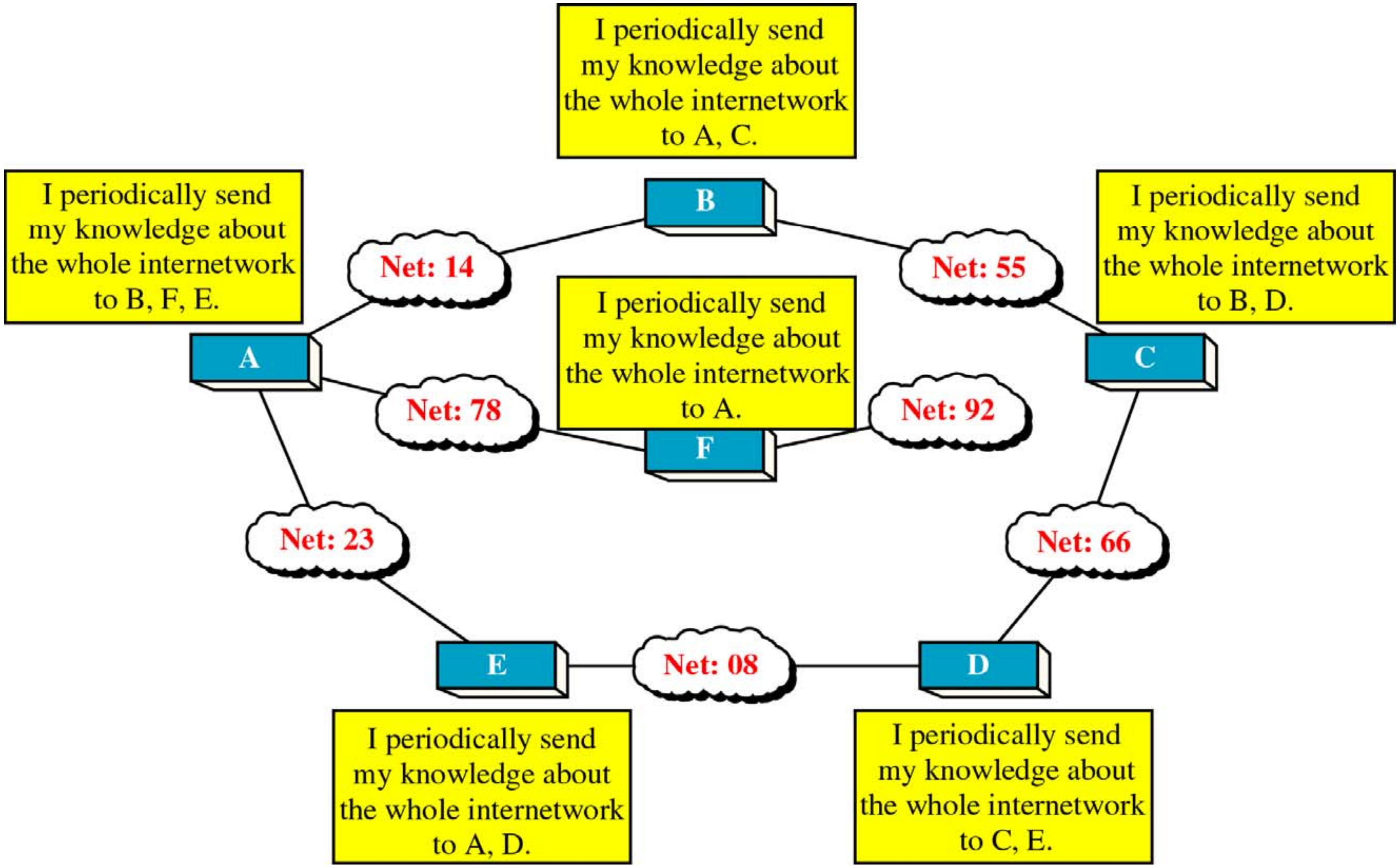


Figure 21-19

Distance Vector Routing Table

Network ID	Cost	Next Hop
. . . .	. . . .	. . . .
. . . .	. . . .	. . . .
. . . .	. . . .	. . . .
. . . .	. . . .	. . . .
. . . .	. . . .	. . . .

Figure 21-20

Routing Table Distribution

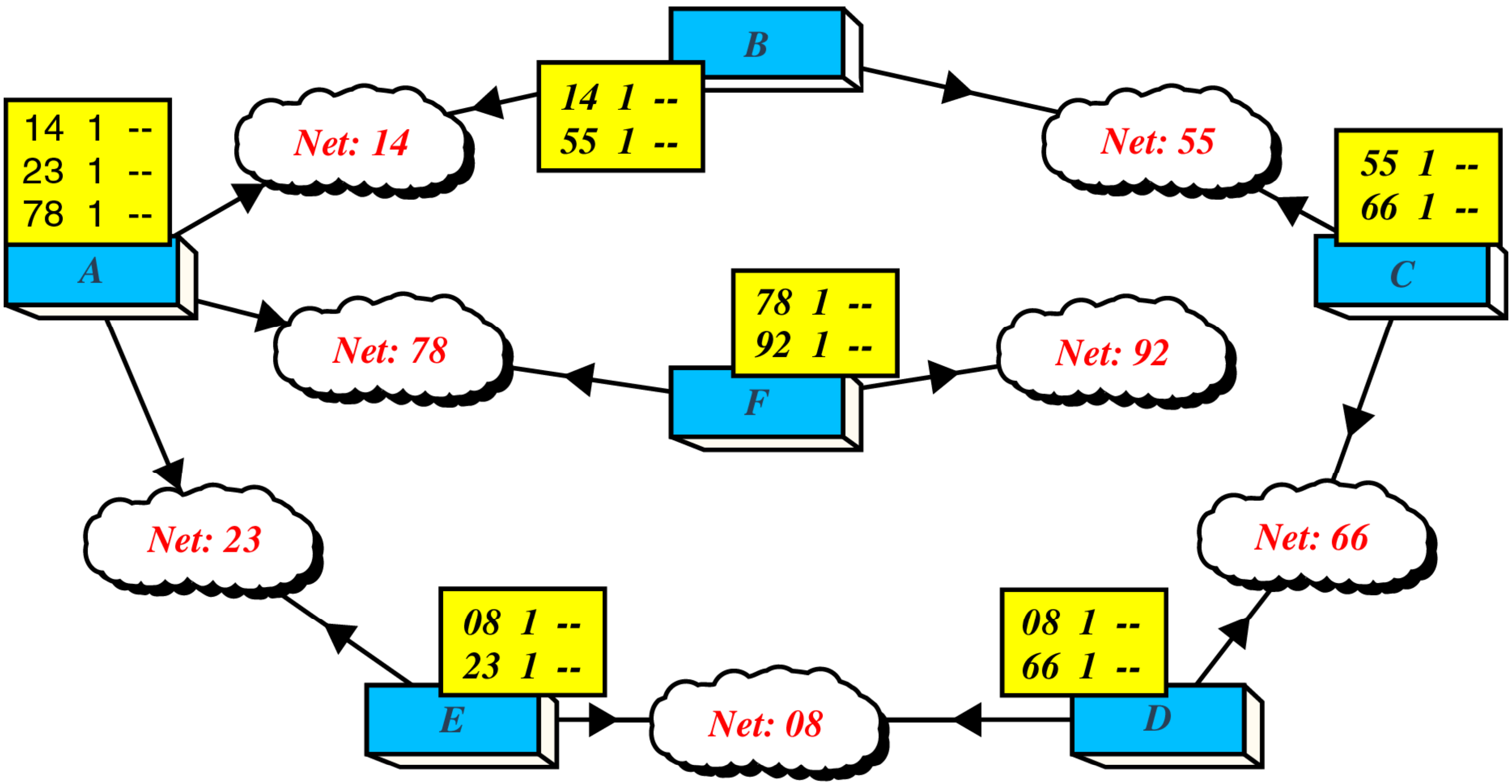


Figure 21-21

Updating Routing Table for Router A

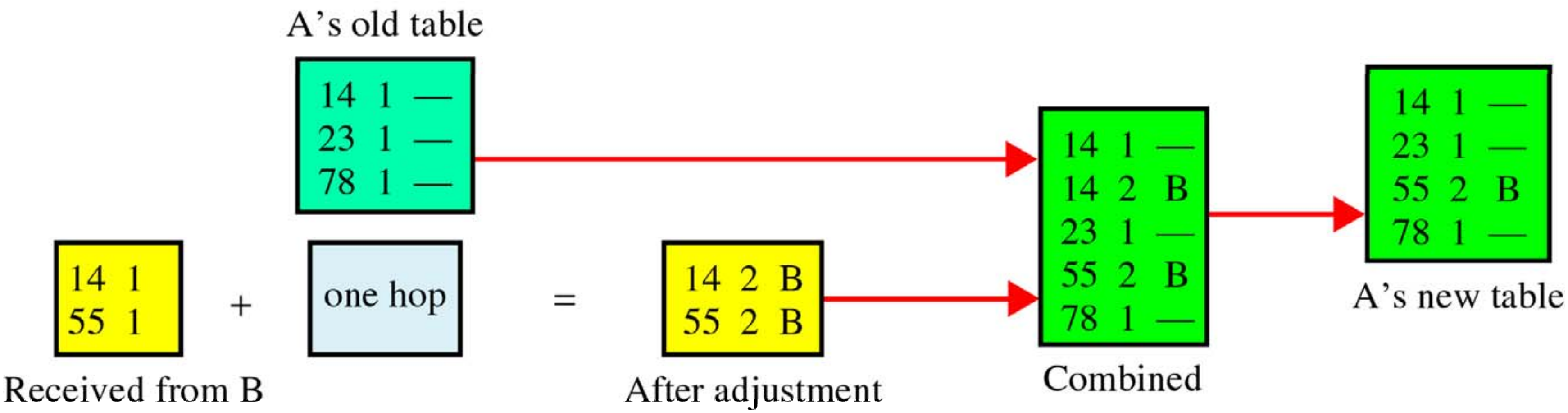
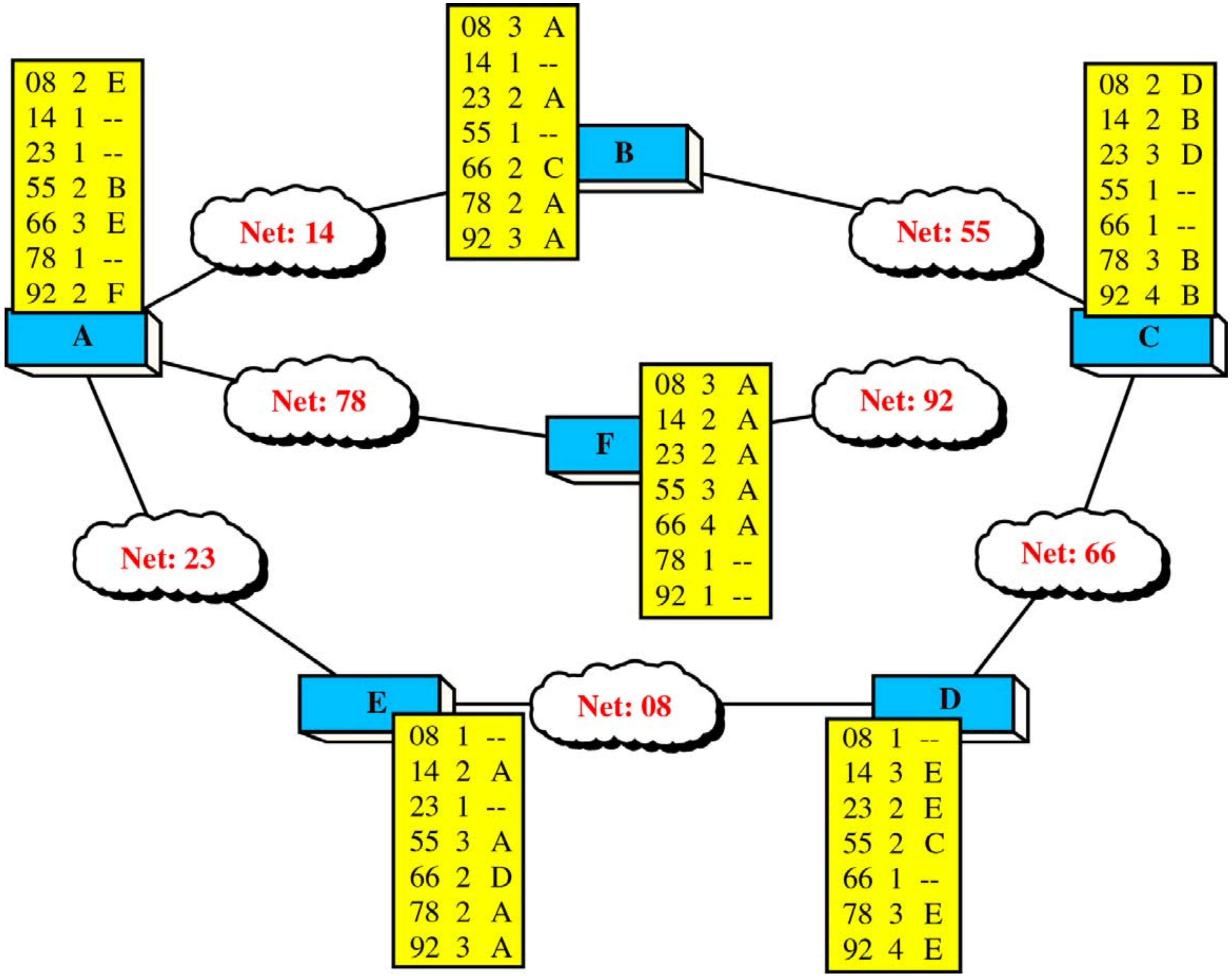


Figure 21-22

Final Routing Tables

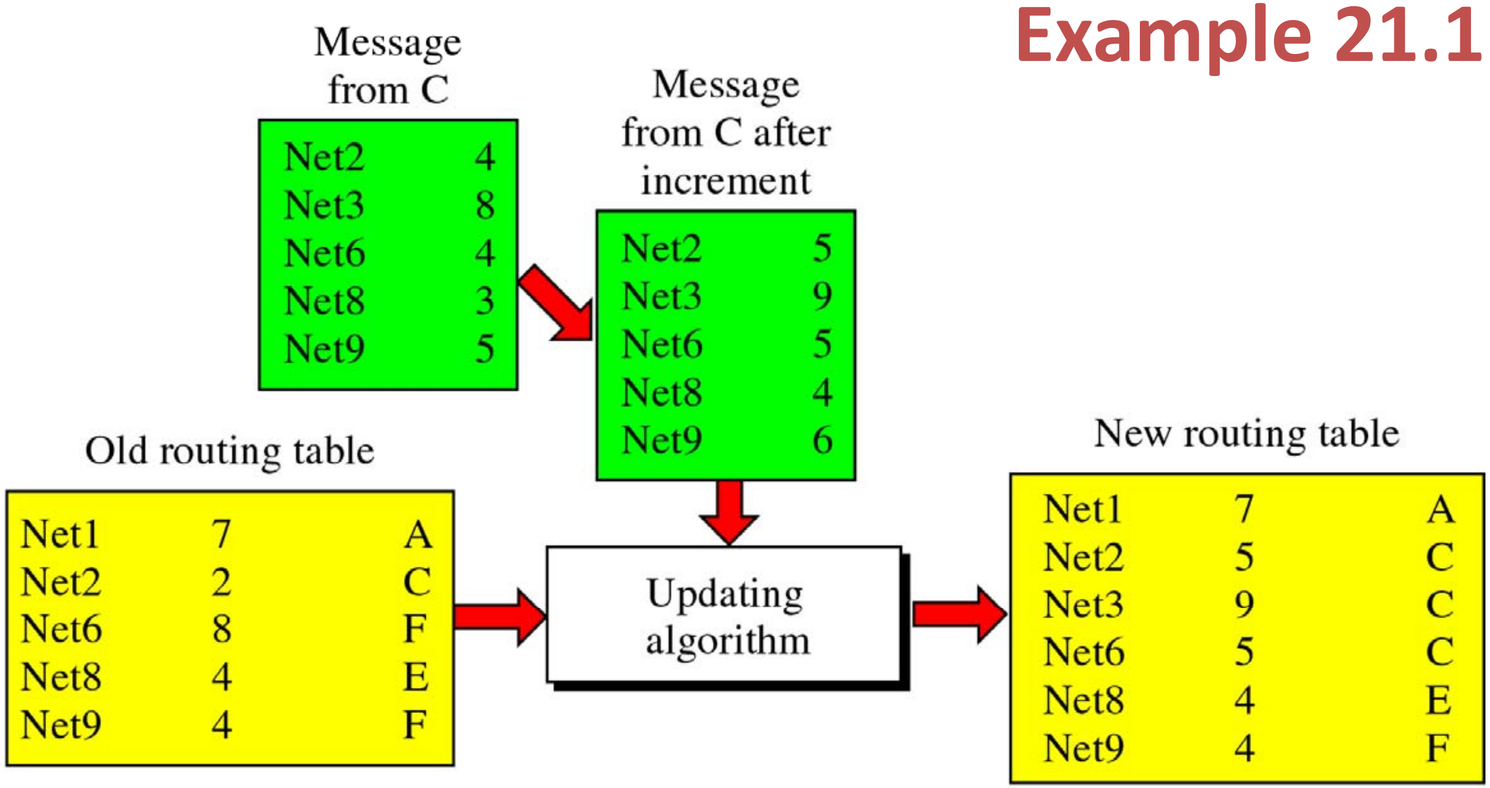


Updating Algorithm

(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

Figure 21-23

Example 21.1



Rules

Net2: Replace (Rule 2.a)
Net3: Add (Rule 1)
Net6: Replace (Rule 2.b.i)
Net8: No change (Rule 2.b.ii)
Net9: No change (Rule 2.b.ii)

Note that there is no news about Net1 in the advertised message, so none of the rules apply to this entry.

Figure 21-24

Concept of Link State Routing

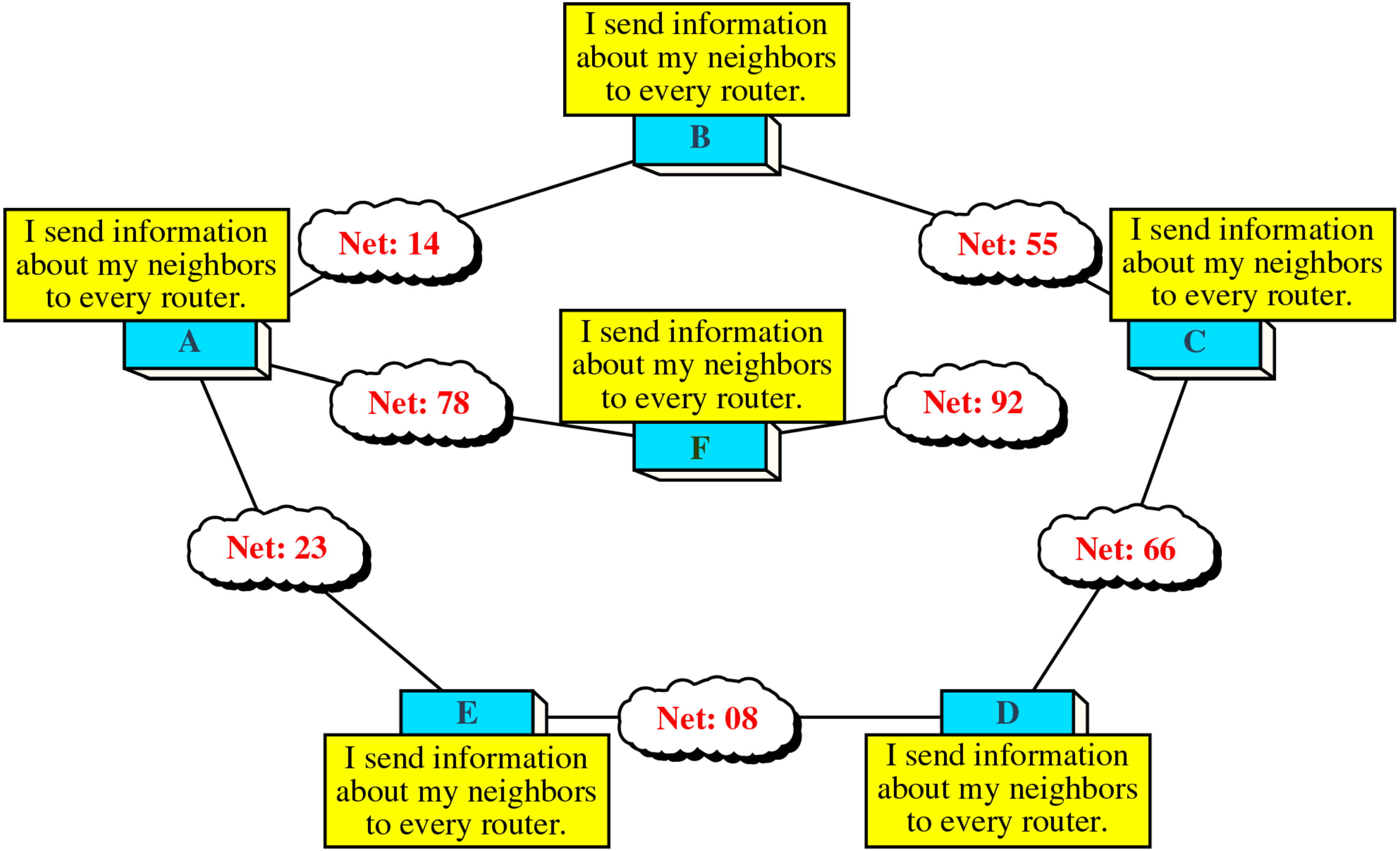


Figure 21-29

Cost in Link State Routing

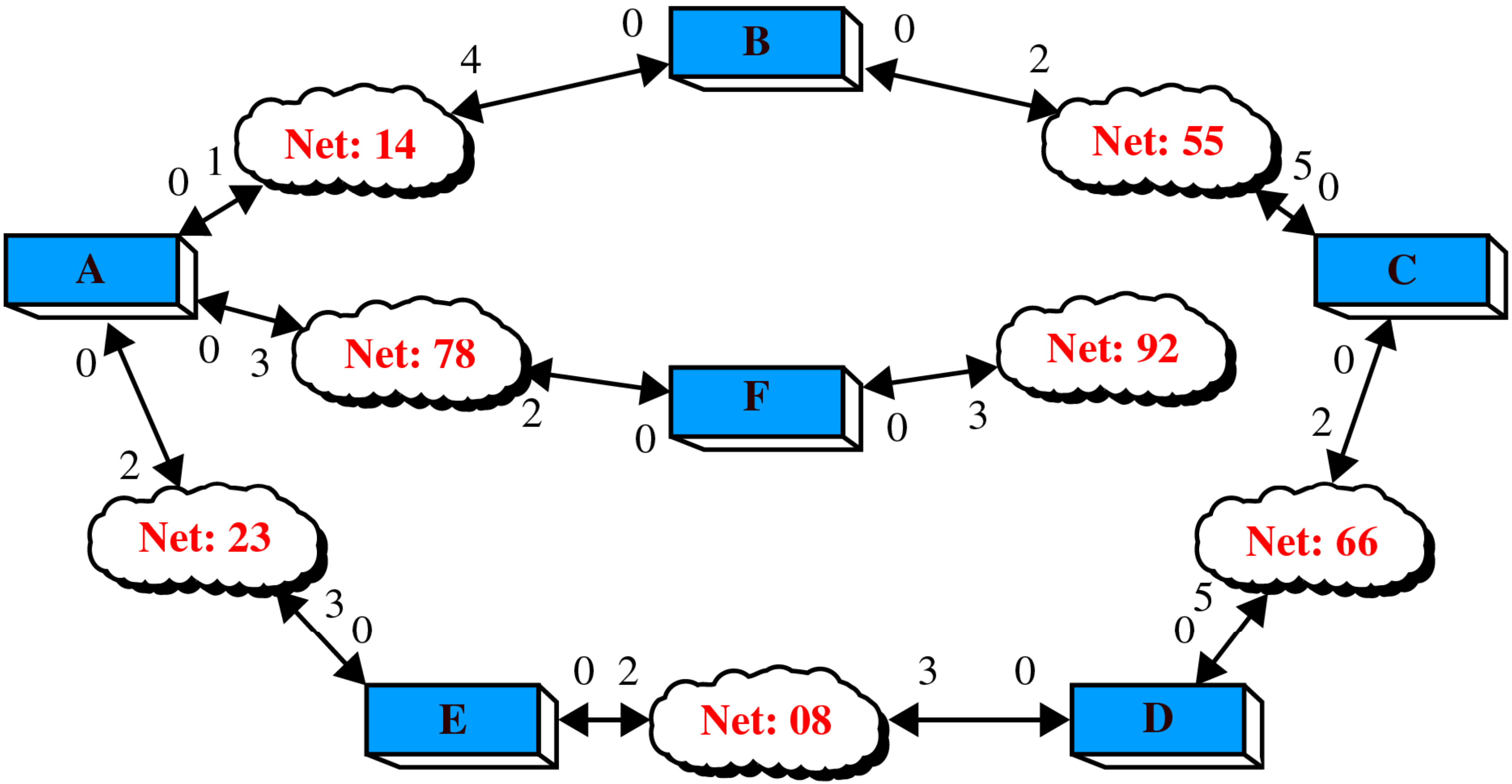


Figure 21-26

Link State Packet

Advertiser	Network	Cost	Neighbor
. . . .	. . . . . .	. . . .	. . . .
. . . .	. . . . . .	. . . .	. . . .
. . . .	. . . .	. . . .	. . . .

Figure 21-27

Flooding of A’s LSP

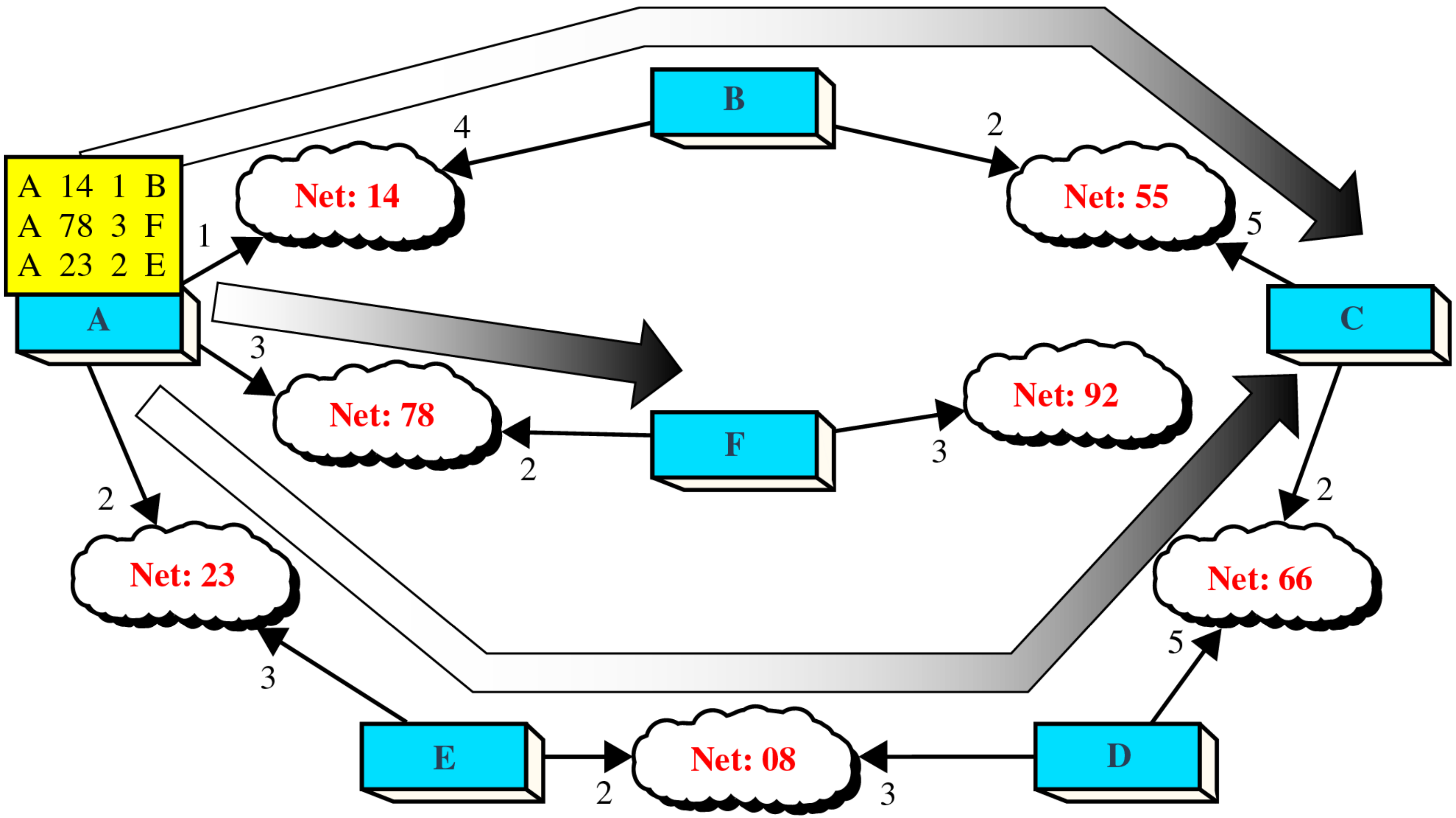


Figure 21-28

Link State Database

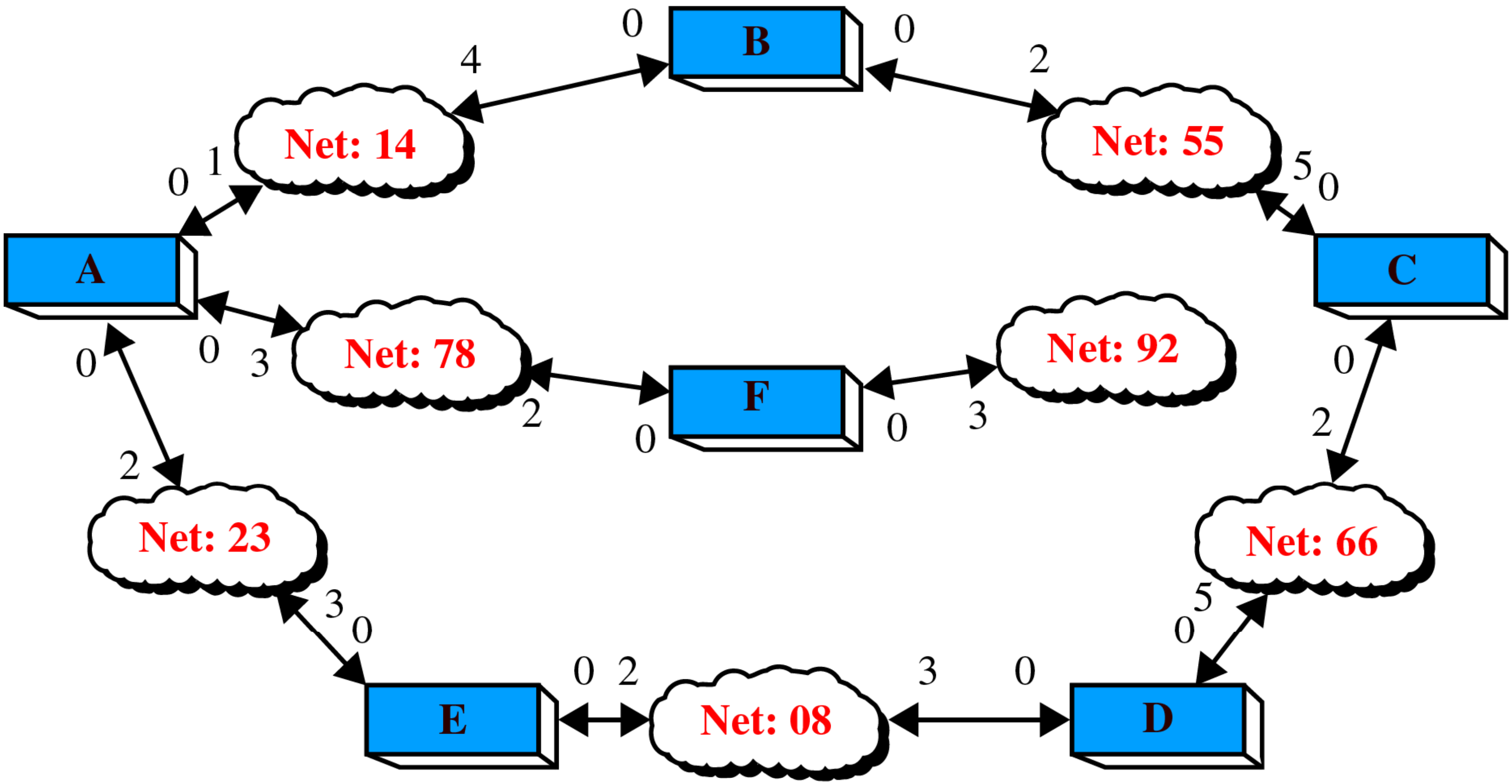
Advertiser	Network	Cost	Neighbor
A	14	1	B
A	78	3	F
A	23	2	E
B	14	4	A
B	55	2	C
C	55	5	B
C	66	2	D
D	66	5	C
D	08	3	E
E	23	3	A
E	08	2	D
F	78	2	A
F	92	3	—

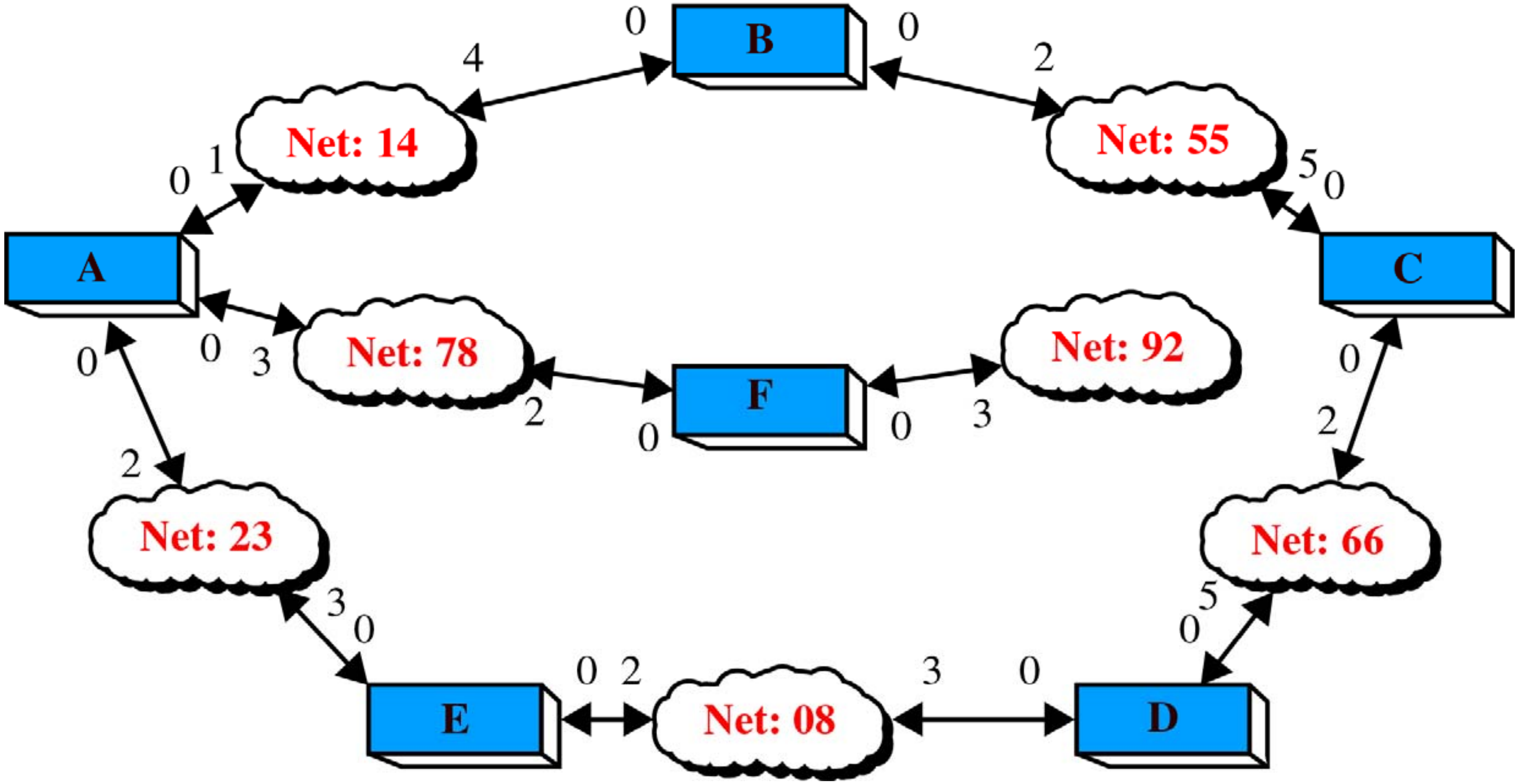
The Dijkstra Algorithm

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

Figure 21-29

Costs in the Dijkstra Algorithm

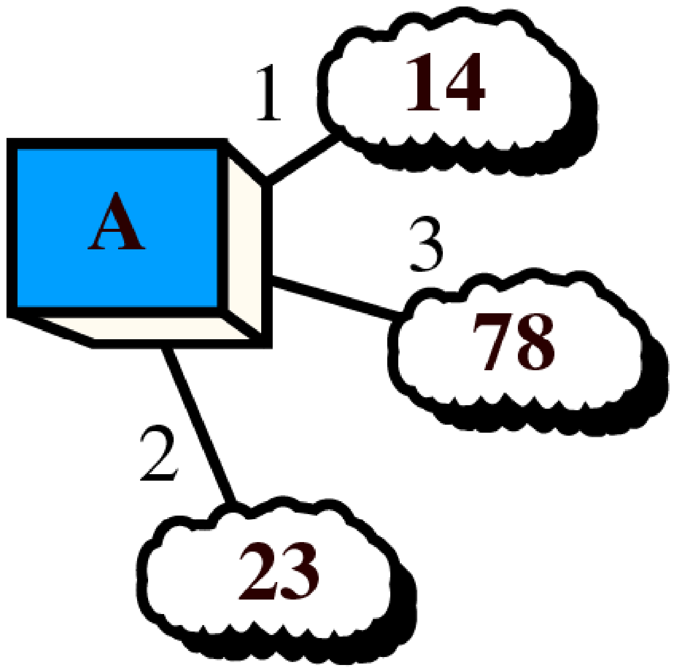




1. Identify the root (the node itself)
2. Attach all neighbour nodes temporarily
3. Make arc and node with least cumulative cost permanent
4. Choose this node
5. Repeat 2 and 3 until all nodes are permanent

Figure 21-30, Part I

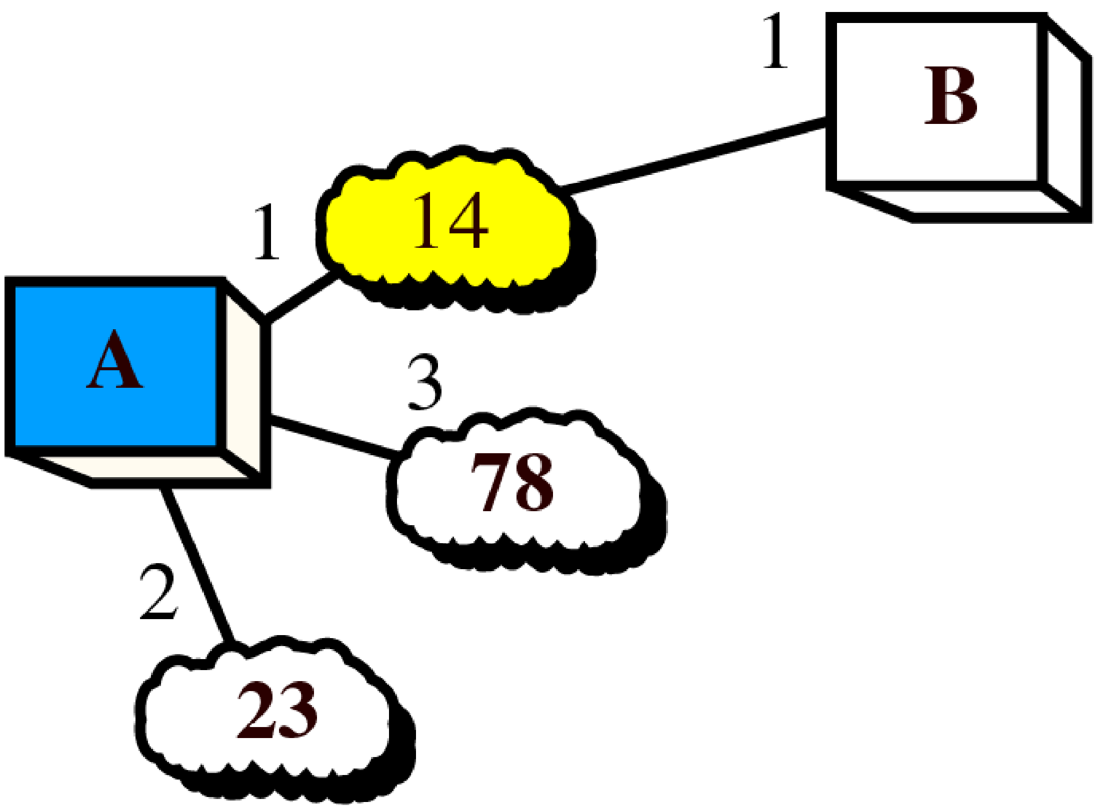
Shortest Path Calculation, Part I



Root is A, networks
14, 78, 23 added

Figure 21-30, Part II

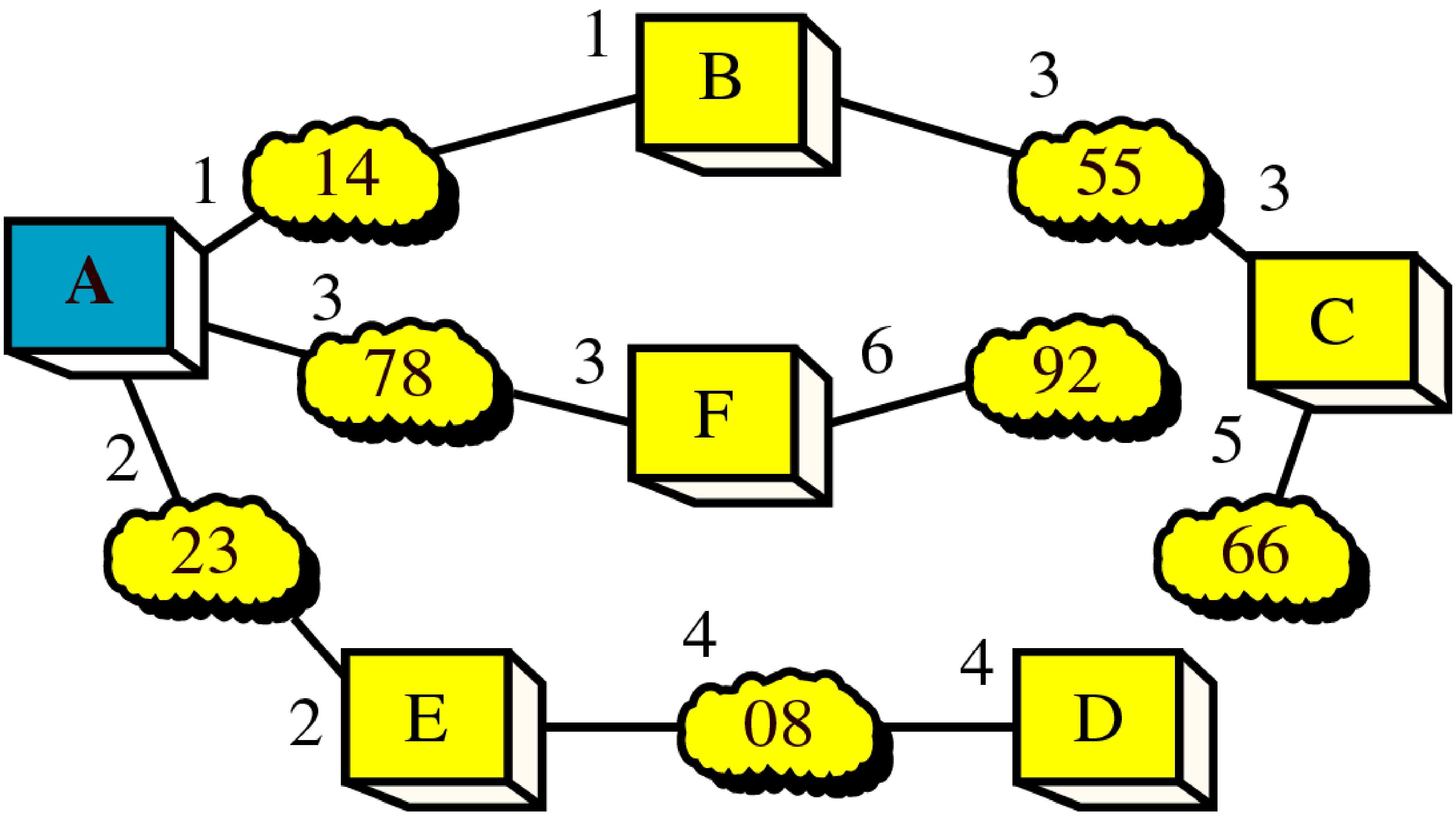
Shortest Path Calculation, Part II



14 permanent, B added

Figure 21-31, Part XIII

Shortest Path Calculation, Part XIII



92 permanent

Figure 21-32

Routing Table for Router A

Net	Cost	Next router
08	4	E
14	1	--
23	2	--
55	3	B
66	5	B
78	3	--
92	6	F