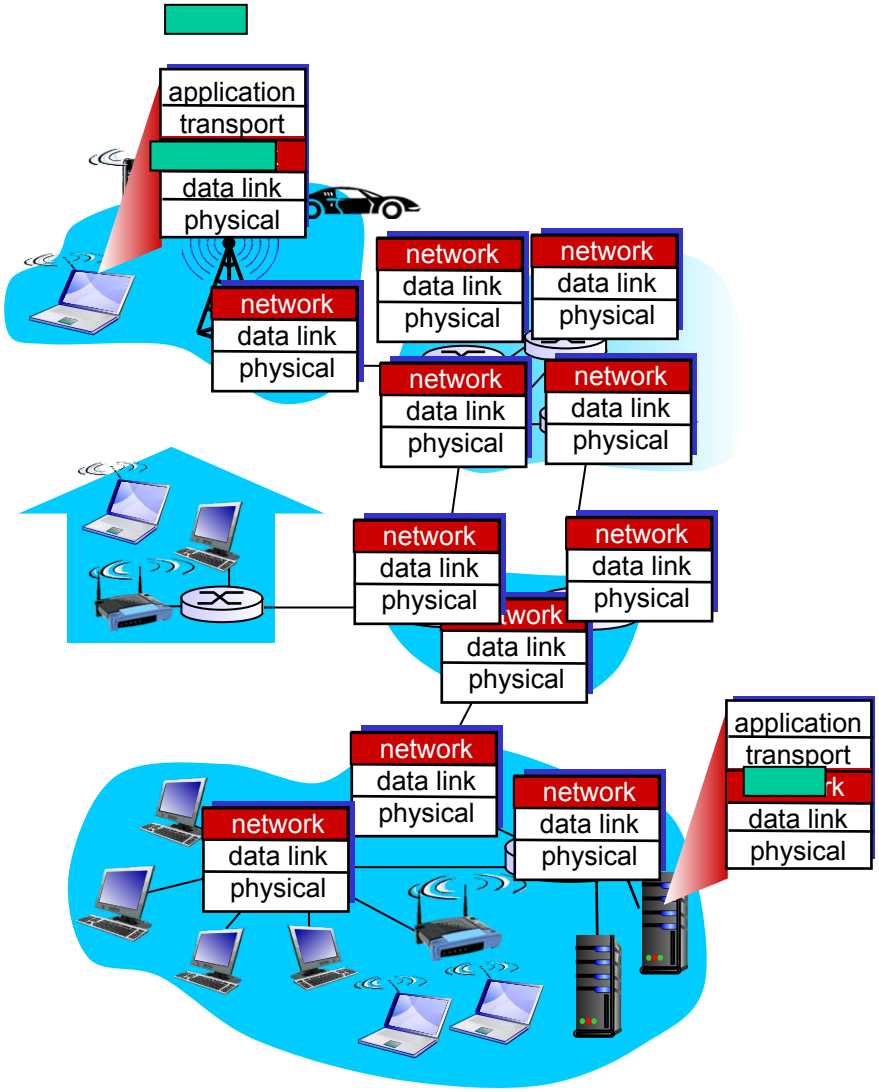
DCN

Unit-4
Network Layer

Dr. Vijeta Khare

Introduction: Network Layer



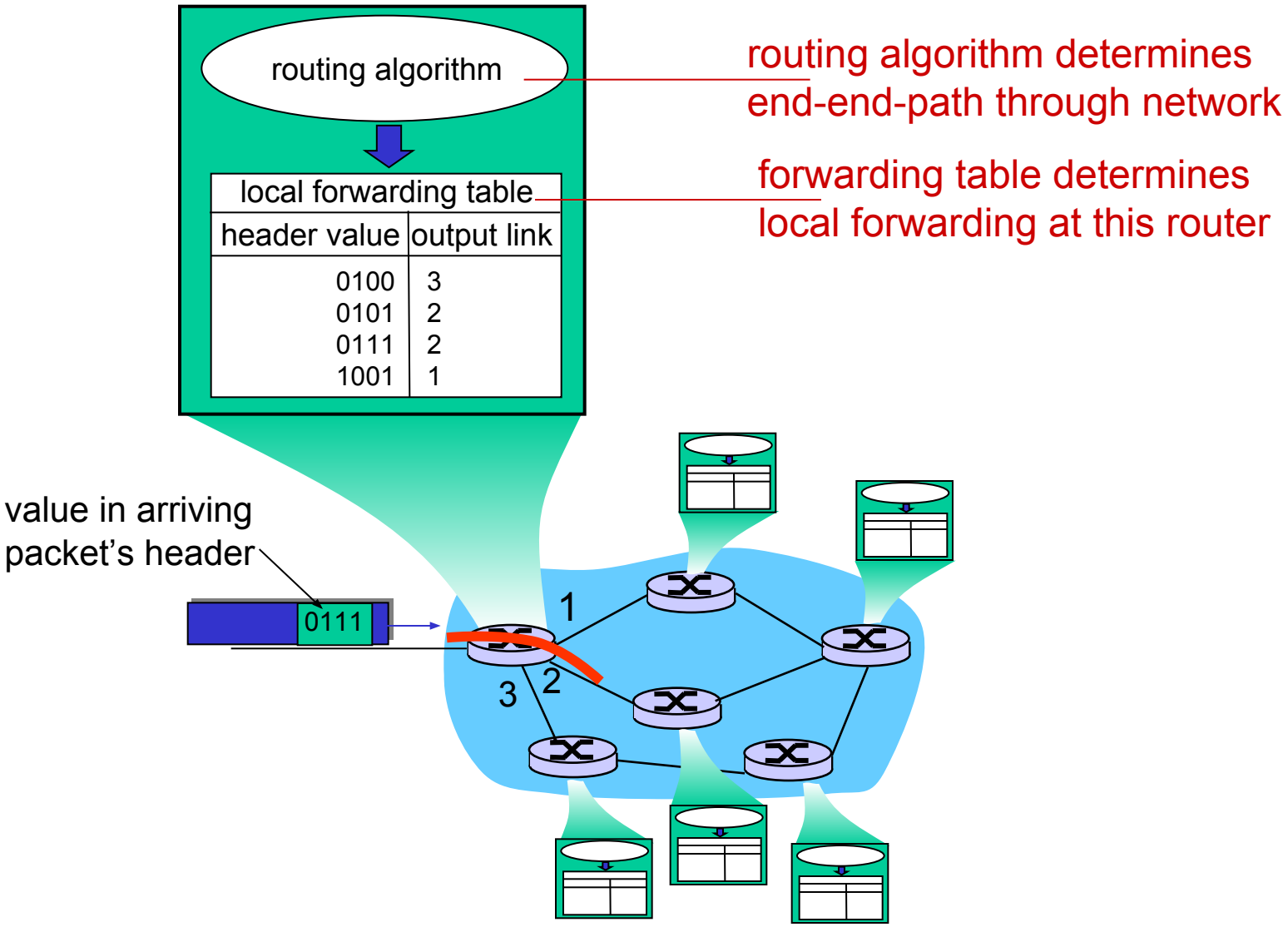
- To deliver segment from sending to receiving host/ router. (Host to host)
- On sending side, it encapsulates segments into datagrams.
- On receiving side, it delivers segments to transport layer.
- Network layer protocols in every host and router.
- Router examines header fields in all IP datagrams passing through it.

Key Function of Network Layer

- Role of the network layer is simple - to move packets from a sending host to a receiving host.
- Two important network layer functions can be identified:

1. **Forwarding**
   - ✔ When a packet arrives at a router's input link, the router must move the packet to the appropriate output link.
2. **Routing**
   - ✔ It's a process of selecting best paths in a network.
   - ✔ The network layer must determine the route or path taken by packets as they flow from a sender to a receiver.
   - ✔ The algorithms that calculate these paths are referred to as routing algorithms.

Routing and Forwarding



Network Service Model

- Services provided by network layer for individual datagrams.
- **Guaranteed delivery**
  - ✔ This service guarantees that the packet will eventually arrive at its destination.
- **Guaranteed delivery with bounded delay**
  - ✔ This service not only guarantees delivery of the packet, but delivery within a specified host-to-host delay bound.

Network Service Model – Cont…

- Services provided by network layer for a flow of datagrams.
- **In-order packet delivery**
  - ✔ This service guarantees that packets arrive at the destination in the order that they were sent.
- **Guaranteed minimal bandwidth**
  - ✔ This network-layer service emulates the behaviour of a transmission link of a specified bit rate (for example, 1 Mbps) between sending and receiving hosts.
  - ✔ As long as the sending host transmits bits at a rate below the specified bit rate, then no packet is lost.

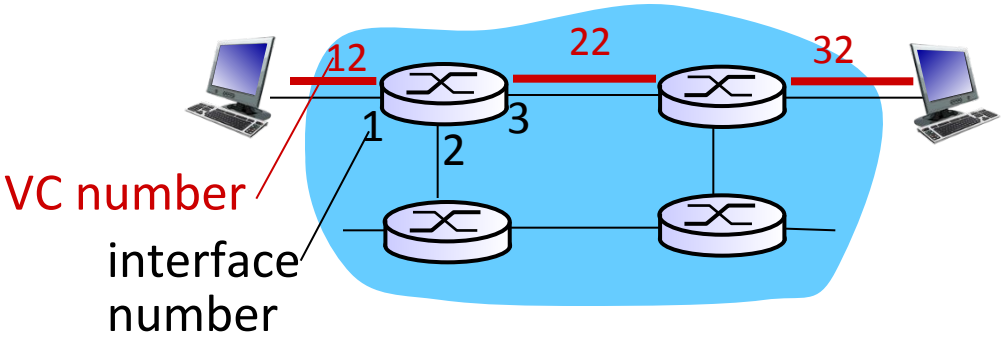
Network Service Model – Cont…

- **Guaranteed maximum jitter**
  - ✔ This service guarantees that the amount of time between the transmission of two successive packets at the sender is equal to the amount of time between their receipt at the receiver.
- **Security services**
  - ✔ Using a secret session key known only by a source and destination host, the network layer in the source host could encrypt the payloads of all datagrams being sent to the destination host.
  - ✔ The network layer in the destination host would then be responsible for decrypting the payloads.

Virtual Circuit Switching

- A VC consists of
  1. A path between the source and destination hosts
  2. VC numbers, one number for each link along the path
  3. Entries in the forwarding table in each router along the path
- A packet belonging to a virtual circuit will carry a VC number in its header.
- VC number can be changed on each link
  - ✔ New VC number comes from forwarding table

VC Forwarding Table



forwarding table in router:

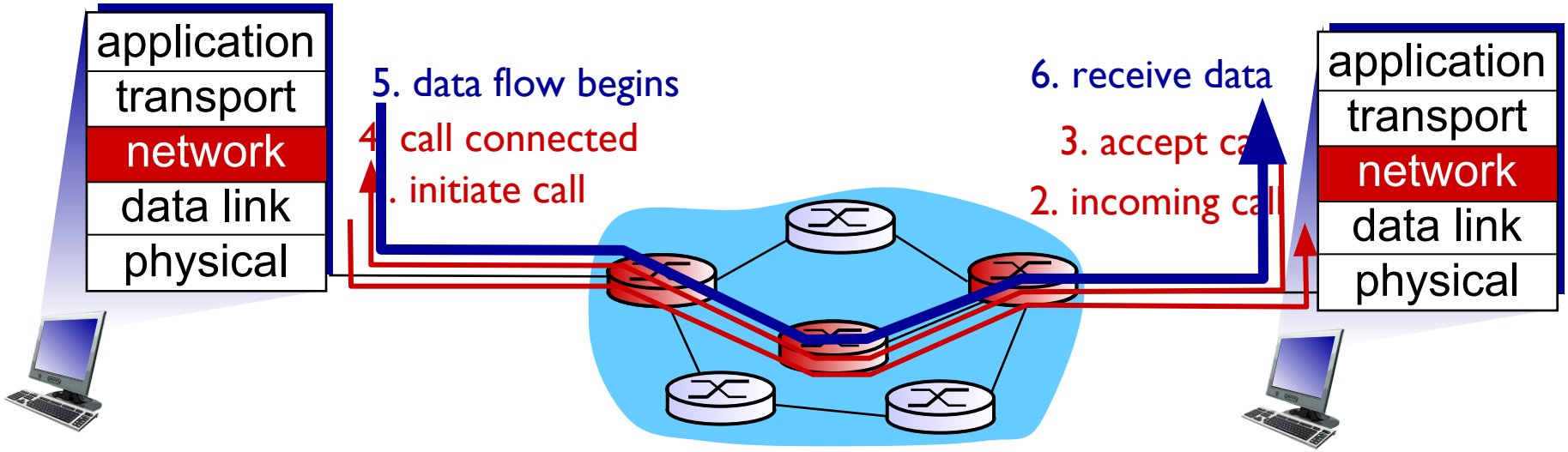
Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

VC routers maintain connection state information

Virtual Circuit Setup

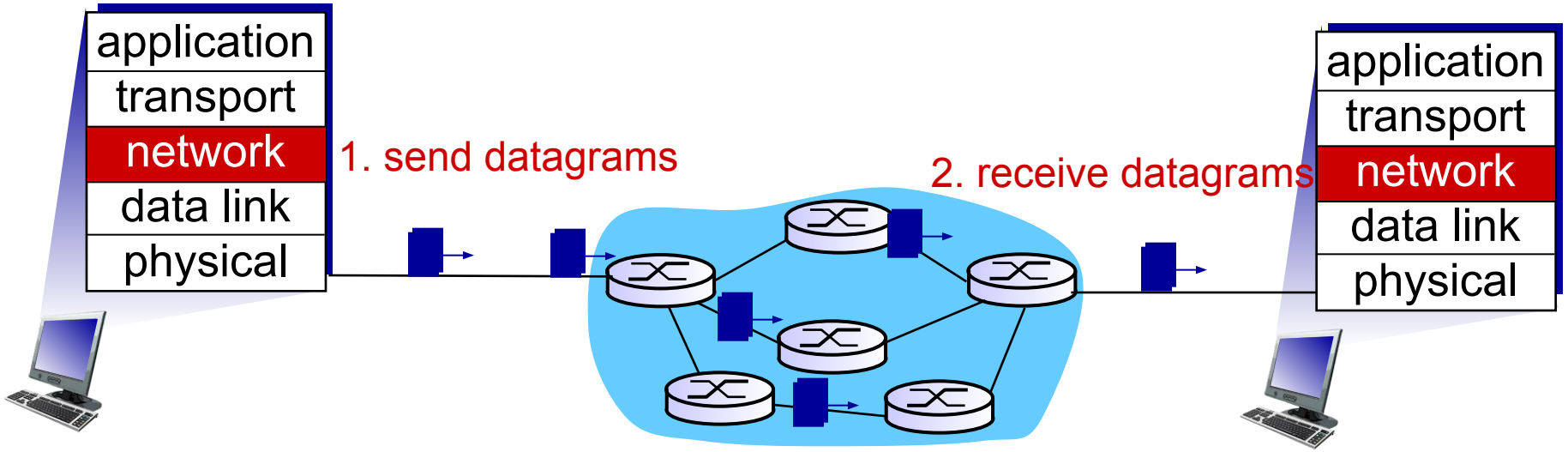
There are three identifiable phases in a virtual circuit:

1. **VC setup**
2. **Data transfer**
3. **VC teardown**



Datagram Network

- In connectionless service, packets are injected into the subnet individually and routed independently of each other.
- No advance setup is needed. The packets are frequently called datagrams and the subnet is called a **datagram subnet**.
- Only directly-connected lines can be used.



Datagram Network vs. Virtual Circuit Network

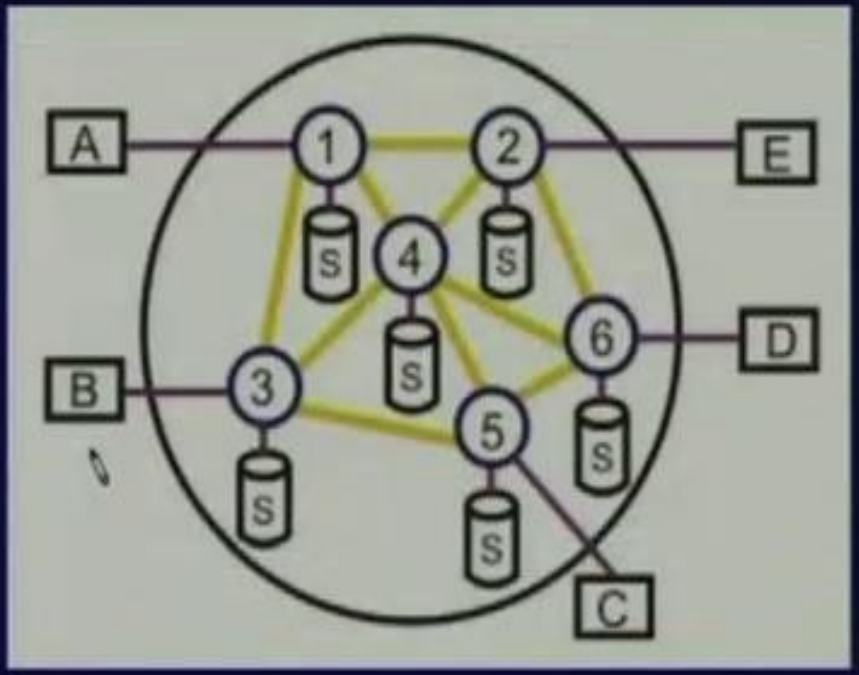
	Datagram	Virtual Circuit
Connection Setup	None	Required
Addressing	Packet contains full source and destination address	Each virtual circuit number entered to table on setup, used for routing.
State Information	None other than router table containing destination network	Route established at setup, all packets follow same route.
Effect of Router Failure	Only on packets lost during crash	All virtual circuits passing through failed router terminated.
Congestion Control	Difficult since all packets routed independently router resource requirements can vary.	Simple by pre-allocating enough buffers to each virtual circuit at setup, since maximum number of circuits fixed.

	Type of	subnet
Service to upper layer	**Datagram**	**Virtual Circuit**
Connectionless	UDP IP	UDP IP ATM
Connection-oriented	TCP IP	ATM AAL1 ATM

Routing Defined

In most of the situations, packets will require multiple hops to make journey towards the destination station.

- Routing is one of the most complex and crucial aspect of packet-switched network design.



Desirable properties

- **Correctness and simplicity**: Self-explanatory.
- **Robustness**: Ability of the network to deliver packets via some route even in the face of failures.
- **Stability**: The algorithm should converge to equilibrium fast in the face of changing conditions in the network.
- **Fairness and optimality**: obvious requirements, but conflicting.
- **Efficiency**: Minimum overhead

Routing algorithms

- Desirable properties
  - ✔ Correctness
  - ✔ Simplicity
  - ✔ Robustness: able to cope with
    - changes in topology, load
    - hardware and software failures
  - ✔ Stability
    - Converge to equilibrium
  - ✔ Fairness
  - ✔ Optimality

} all algorithms

} hard to achieve

} conflicting

Design Parameters

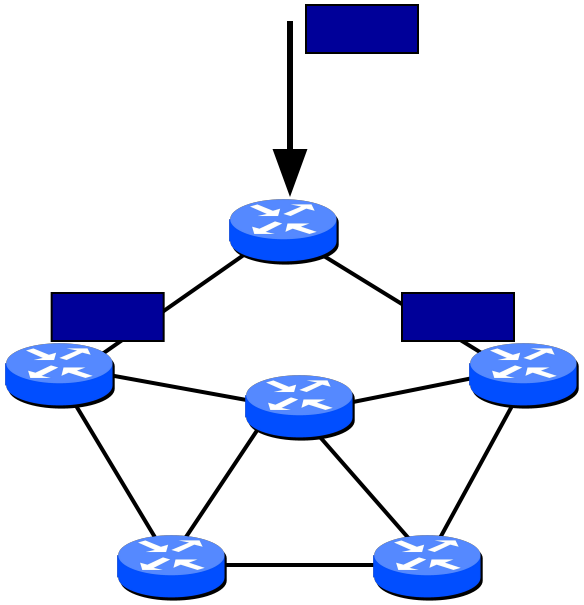
- **Performance Criteria:** Number of hops, Cost, Delay, Throughput
- **Decision Time:** Per packet (Datagram), per session (Virtual-circuit)
- **Decision Place:** Each node (distributed), Central node (centralized), Originated node (source)
- **Network Information Source:** None, Local, Adjacent node, Nodes along route, All nodes
- **Network Information Update Timing:** Continuous, Periodic, Major load change, Topology change

Routing Strategies

- A large number of routing strategies have evolved over the years.
- Important strategies are:
  - Fixed (Static) routing
  - Flooding
  - Random routing
  - Flow-based Routing
  - Adaptive/Dynamic routing

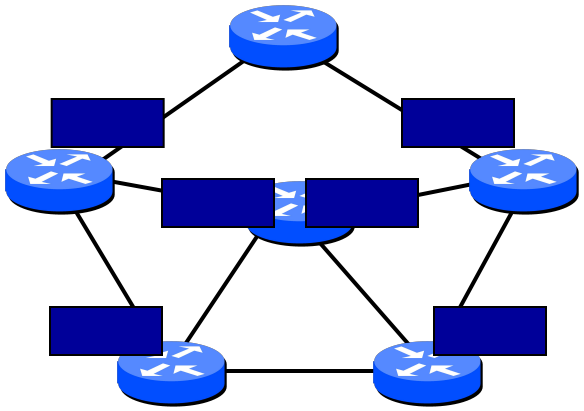
Routing: flooding

- Every packet is sent out on every outgoing line except the one it arrived at



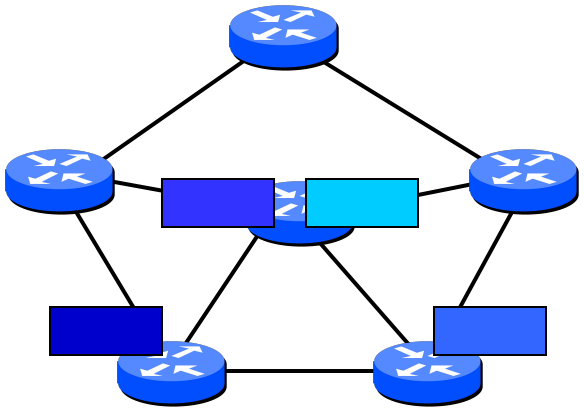
Routing: flooding

- Every packet is sent out on every outgoing line except the one it arrived at



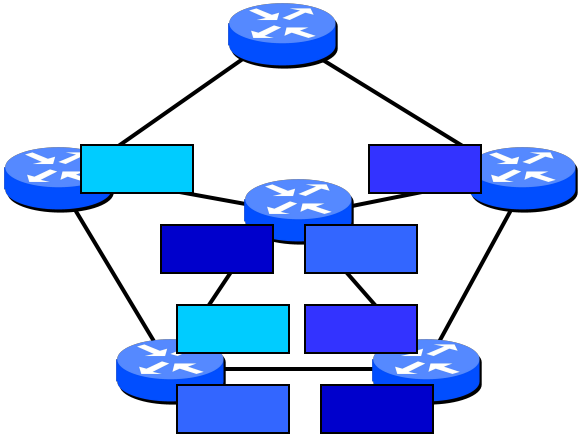
Routing: flooding

- Every packet is sent out on every outgoing line except the one it arrived at



Routing: flooding

- Every packet is sent out on every outgoing line except the one it arrived at



Routing: flooding

- Every packet is sent out on every outgoing line except the one it arrived at
- Duplicates!! How to limit?
  - ✔ Hop counter
    - Decrement in each router
    - Discard packet if counter is 0
    - Initialisation?
  - ✔ Sequence number in packet
    - Avoid sending the same packet a second time
    - Keep in each router per source a list of packets already seen
- *Useful?*

Routing: flooding

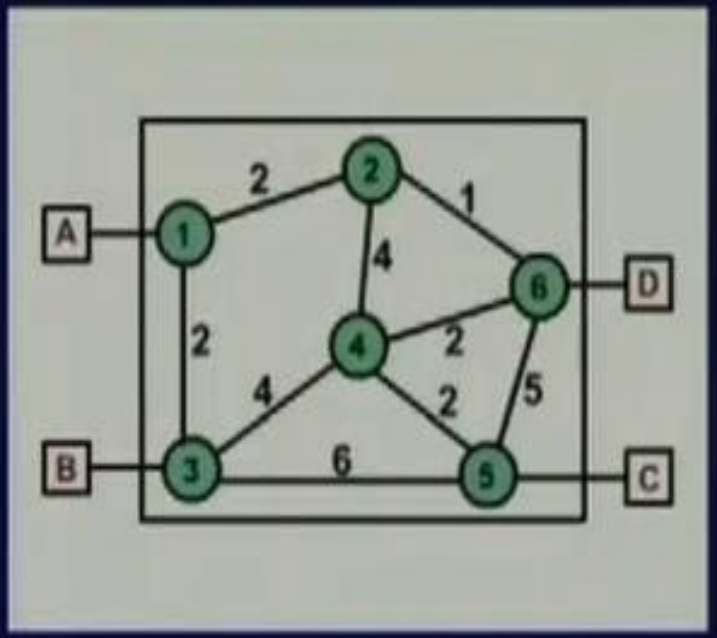
- Every packet is sent out on every outgoing line except the one it arrived at
- Sometimes useful
  - ✔ Robust algorithm: e.g. military applications
  - ✔ Broadcast
  - ✔ Comparison purposes: always shortest path
- Selective flooding
  - ✔ Use only those lines that are going approximately in right direction
  - ✔ Still working?

Fixed Routing

- A route is selected for each source-destination pair of nodes in the network.
- The routes are fixed; they may only change if there is a change in the topology of the network.
- How fixed routing may be implemented?

Fixed Routing: Example

- A central routing matrix is created based on least-cost path, which is stored at a network control center.
- The matrix shows, for each source-destination par of nodes, the identity of the next node on the route.



Central Routing Directory

From node \ To node	1	2	3	4	5	6
1	-	2	3	2	2	2
2	1	-	1	6	6	6
3	1	1	-	4	5	1
4	6	6	3	-	5	6
5	4	4	3	4	-	4
6	2	2	2	4	4	-

Fixed Routing: Example

- From the main routing matrix, routing tables to be used by each individual node can be developed

Node 1 Directory

Destination	Next Node
2	2
3	3
4	2
5	2
6	2

Node 2 Directory

Destination	Next Node
1	1
3	1
4	6
5	6
6	6

Node 3 Directory

Destination	Next Node
1	1
2	1
4	4
5	5
6	1

Node 4 Directory

Destination	Next Node
1	6
2	6
3	3
5	5
6	6

Node 5 Directory

Destination	Next Node
1	4
2	4
3	3
4	4
6	4

Node 6 Directory

Destination	Next Node
1	2
2	2
3	2
4	4
5	4

Least-cost Path

- A cost is associated with each link.
- The simplest criterion is to choose the minimum-hop route through the network.
  - Easily measured criterion.
- A generalization is least-cost routing.
- For any pair of attached stations, the least cost route through the network is looked for.
- For either case, several well-known algorithms exist to obtain the optimum path.
  - Dijkstra's algorithm
  - Bellman-Ford algorithm

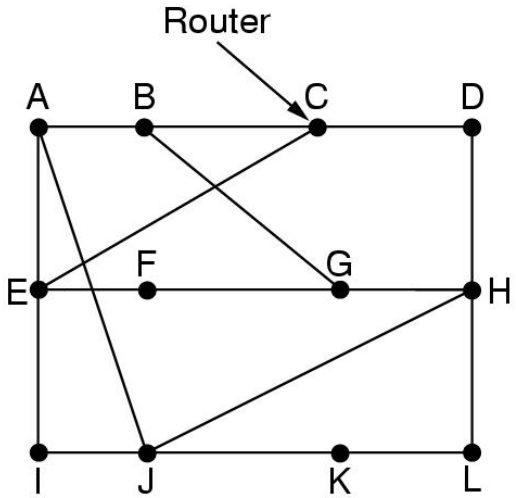
Routing: distance vector

- Adaptive algorithm
  - ✔ Exchange of info only with neighbours
- Data to be available in each router
  - ✔ Routing table: per destination
    - Distance
    - Outgoing line
  - ✔ Distance to all neighbours

Distance Vector Algorithm

- Distance-vector (DV) algorithm is iterative, asynchronous, and distributed.
- It is distributed in that each node receives some information from one or more of its directly attached neighbours, performs a calculation, and then distributes the results of its calculation back to its neighbours.
- It is iterative. so, process continues on until no more information is exchanged between neighbours.
- The algorithm is asynchronous. It does not require all of the nodes to operate with each other.

Routing: distance vector



(a)

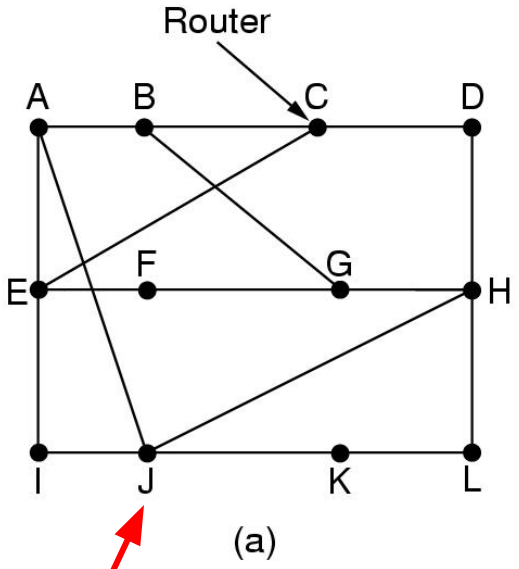
Routing table for A

To	cost	via
A	0	-
B	12	B
C	25	B
D	40	B
E	14	E
F	23	E
G	18	B
H	17	J
I	21	E
J	9	J
K	24	J
L	29	J

Routing: distance vector

- Algorithm
  - ✔ At each step within a router:
    - Get routing tables from neighbours
    - Compute distance to neighbours
    - Compute new routing table
  - ✔ Characteristics:
    - Iterative
    - Asynchronous
    - Distributed

Routing: distance vector



(a)

JA=8

JI=10

JH=12

JK=6

Routing tables from

New estimated delay from J

To	A	I	H	K	Line
G	18	31	6	31	18	H

JA delay is 8

JI delay is 10

JH delay is 12

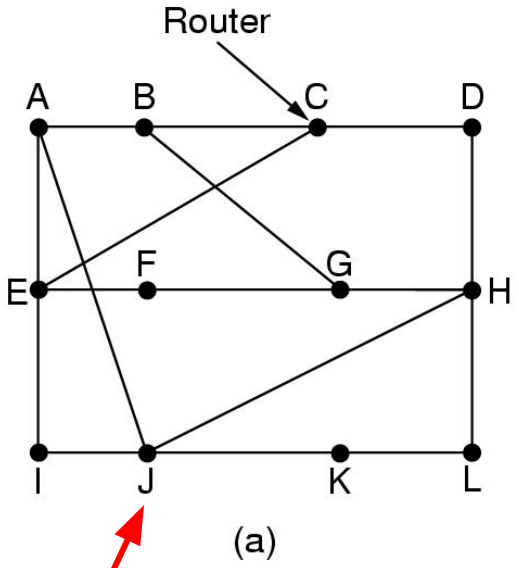
JK delay is 6

New routing table for J

Vectors received from J's four neighbors

(b)

Routing: distance vector



(a)

Better?

- Keep 4 tables (one for each neighbour)
- Use shortest path

To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

(b)

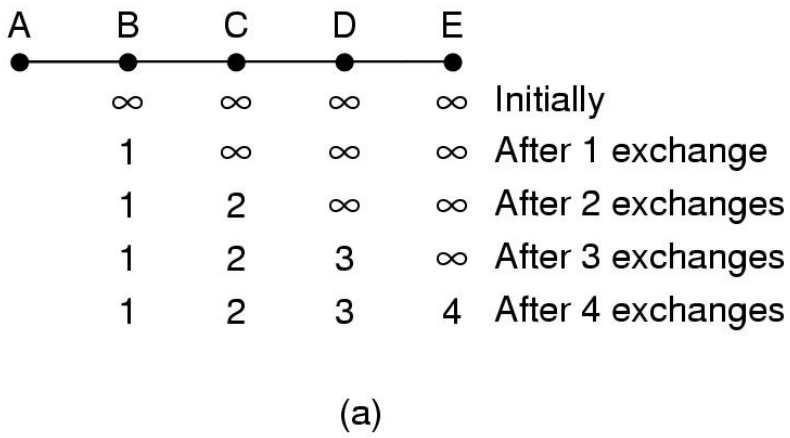
Routing: distance vector

- Distributed algorithm
  - ✔ Triggers:
    - Change in delay to neighbour
    - Receive new table from neighbour
  - ✔ Update local tables
  - ✔ If changed: forward routing tables to neighbours
- Asynchronous
  - ✔ Execution in lock step not required
- Iterative
  - ✔ Stops?

How fast are changes propagated?

- Good news?
- Bad news?

Routing: distance vector



(a)

Good news:

- A comes up again

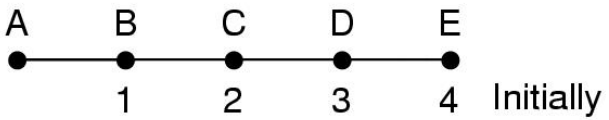
Only distances to A

Faster not possible!!!

Routing: distance vector

Bad news:

- A goes down



B receives:

- Distance ∞ from A
- Distance 2 from C

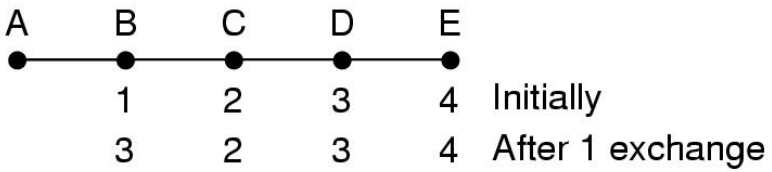
New distance from B to A: 3 via C

(b)

Routing: distance vector

Bad news:

- A goes down



LOOP!!!

C still

- believes its distance to A is 2
- routes via B

B routes its packets for A via C

(b)

Routing: distance vector

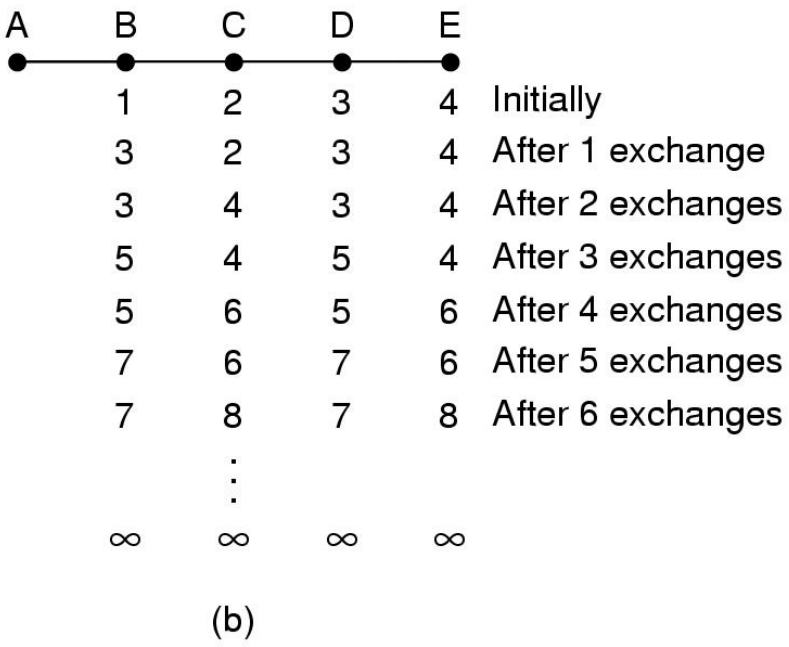
Bad news:

- A goes down

Loops!!

Slow!!

$\infty = 5?$



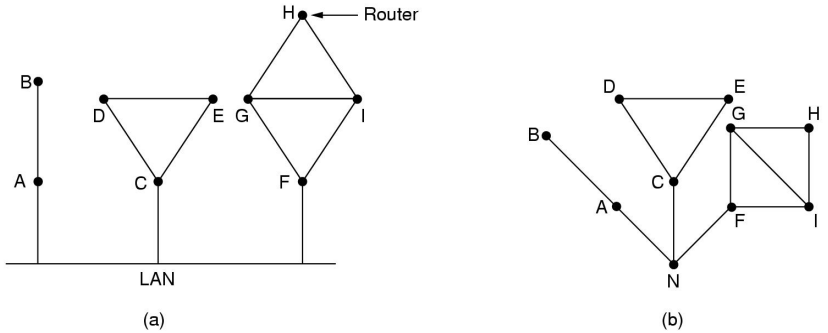
Routing: link state

Overview of algorithm:

- Each router must
  1. Discover its **neighbours** and **learn** their network addresses
  2. Measure the delay or **cost** to each of its neighbours
  3. Construct a **packet** with these distances
  4. Send this packet to **all** other routers
  5. Compute the **shortest path** to every other router through **Dijkstra's Algorithm**.
- Also known as **Dijkstra's Algorithm**.

Routing: link state

1. Learning about neighbours:
   - ✔ Upon boot of router
     - Send HELLO packet on each point-to-point line
     - Routers are supposed to send reply with a globally unique name
   - ✔ LAN model



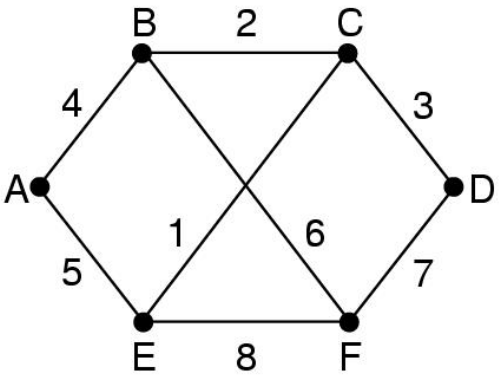
2. Measuring line cost
   - ✔ Measure round-trip delay of HELLO Packet and its reply
   - ✔ Take load into account? Arguments both ways:

Routing: link state

3. Building link state packets

- ✔ Packet containing:
  - Identity of sender
  - Sequence number + age
  - For each neighbour: name + distance

- When to build?
  - periodically
  - when significant events occur



(a)

Link State Packets

A	B	C	D	E	F
Seq.	Seq.	Seq.	Seq.	Seq.	Seq.
Age	Age	Age	Age	Age	Age
B 4	A 4	B 2	C 3	A 5	B 6
E 5	C 2	D 3	F 7	C 1	D 7
	F 6	E 1		F 8	E 8

(b)

Routing: link state

4. Distributing link state packets
   - Flooding +
   - Sequence number (in each packet) to limit duplicates
   - Age (Time to live) is to remove flooding problem

5. Computing new routes:
   - ✔ With a full set of link state packets, a router can:
     - Construct the entire subnet graph
     - Run Dijkstra's algorithm to compute the shortest path to each destination
   - ✔ Problems for large subnets
     - Memory to store data
     - Compute time

Link State Routing Algorithm

- It computes the least-cost path from one node (source node) to all other nodes in the network.
- Its iterative and after the k^{th} least-cost paths are known to k destination nodes.
- **Notation:**
  - ✔ **c(x,y):** link cost from node x to y; = ∞ if not direct neighbours
  - ✔ **D(v):** current value of cost of path from source to destination v
  - ✔ **p(v):** predecessor node along path from source to v
  - ✔ **N':** set of nodes whose least cost path definitively known

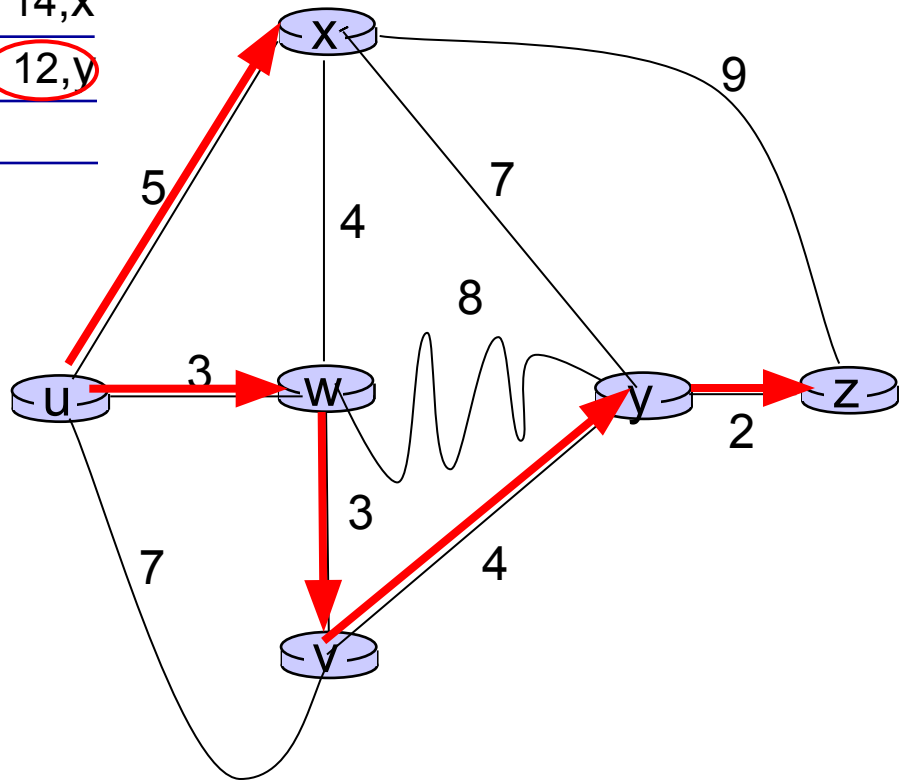
Dijkstra's Algorithm

```
1  Initialization:
2    N' = {u}
3    for all nodes v
4      if v adjacent to u
5          then D(v) = c(u,v)
6      else D(v) = ∞
7
8  Loop
9     find w not in N' such that D(w) is a minimum
10    add w to N'
11    update D(v) for all v adjacent to w and not in N' :
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* new cost to v is either old cost to v or known
14     shortest path cost to w plus cost from w to v */
15 until all nodes in N'
```

Dijkstra's Algorithm – Example:1

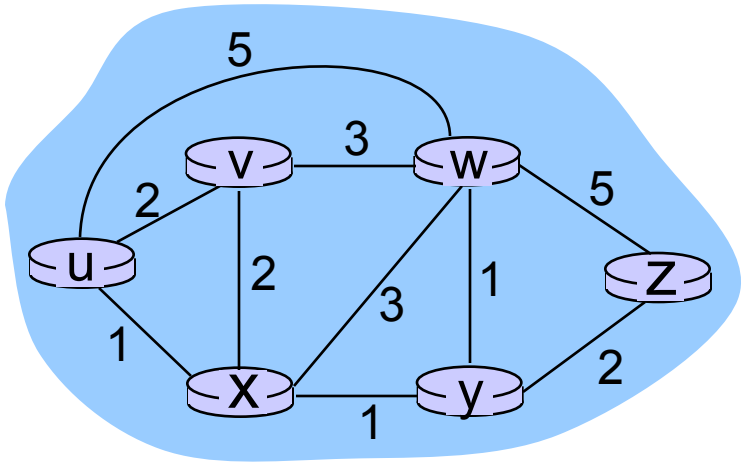
Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

```
1  Initialization:
2   N' = {u}
3   for all nodes v
4     if v adjacent to u
5        then D(v) = c(u,v)
6     else D(v) = ∞
7
8  Loop
9    find w not in N' such that D(w) is a minimum
10   add w to N'
11   update D(v) for all v adjacent to w and not in N' :
12     D(v) = min( D(v), D(w) + c(w,v) )
13   /* new cost to v is either old cost to v or known
14    shortest path cost to w plus cost from w to v */
15 until all nodes in N'
```



Dijkstra's Algorithm – Example:2

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Difference: LS and DV Routing Algorithm

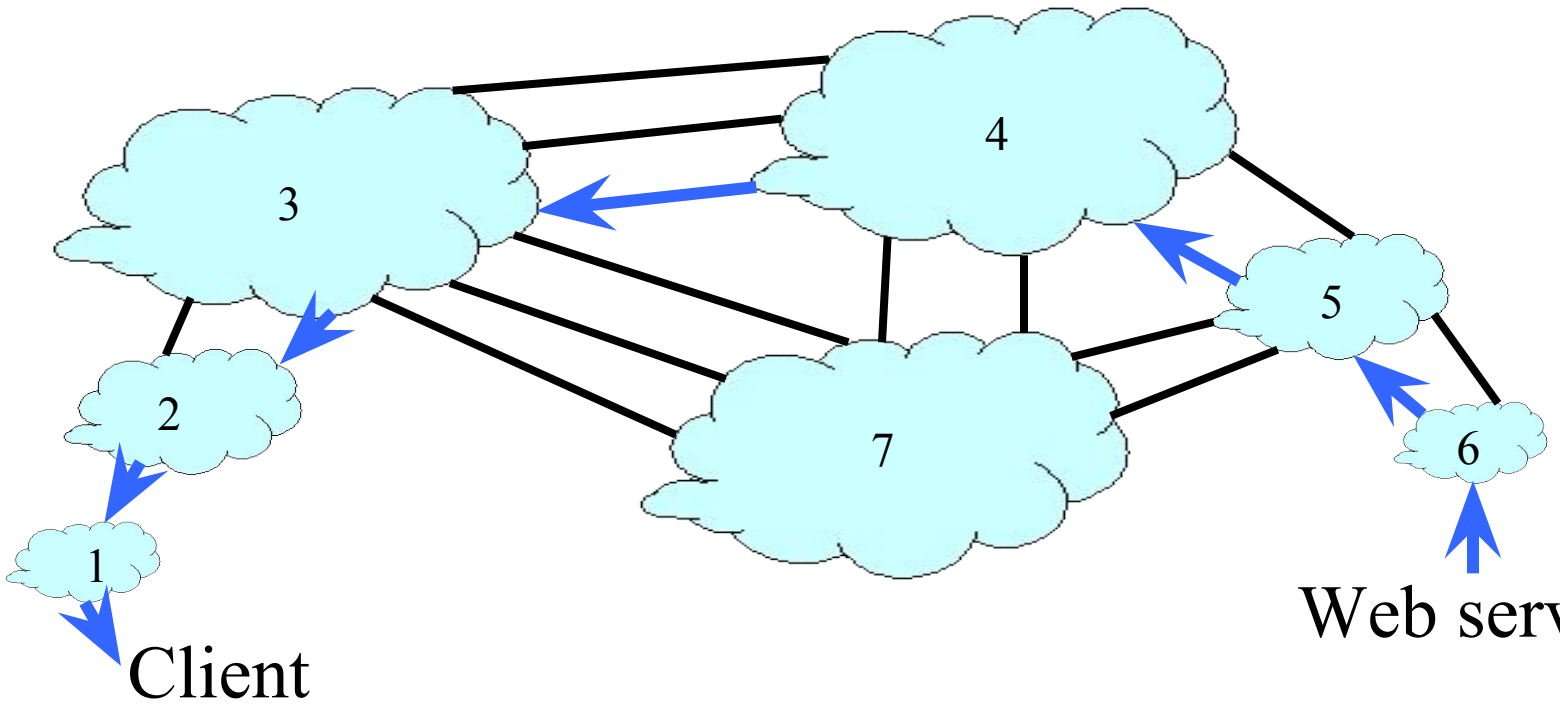
Distance Vector Protocol	Link State Protocol
Entire routing table is sent as an update	Updates are incremental & entire routing table is not sent as update
Distance vector protocol send periodic update at every 30 or 90 second	Updates are triggered not periodic
Update are broadcasted	Updates are multicasted
Updates are sent to directly connected neighbour only	Update are sent to entire network & to just directly connected neighbour
Routers don't have end to end visibility of entire network.	Routers have visibility of entire network of that area only.
It is prone to routing loops	No routing loops

Path Vector Routing

- Both LSR and DVR are intradomain routing algorithm, which works on single network
- Path vector is a inter network routing Algorithm.
- Internet is divided into Autonomous Systems
  - ✔ Distinct regions of administrative control
  - ✔ Routers/links managed by a single “institution”
  - ✔ Service provider, company, university, …
- Hierarchy of Autonomous Systems
  - ✔ Large, tier-1 provider with a nationwide backbone
  - ✔ Medium-sized regional provider with smaller backbone
  - ✔ Small network run by a single company or university
- Interaction between Autonomous Systems
  - ✔ Internal topology is not shared between ASes
  - ✔ … but, neighboring ASes interact to coordinate routing

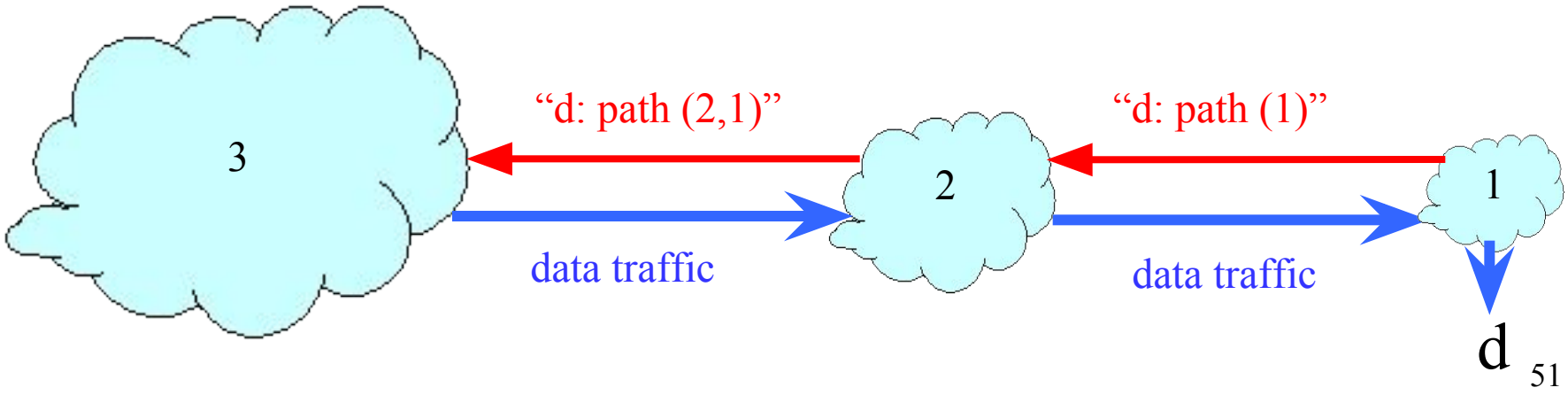
Interdomain Routing

- AS-level topology
  - ✔ Destinations are IP prefixes (e.g., 12.0.0.0/8)
  - ✔ Nodes are Autonomous Systems (ASes)
  - ✔ Edges are links and business relationships



Path-Vector Routing

- Extension of distance-vector routing
  - ✔ Support flexible routing policies
  - ✔ Avoid count-to-infinity problem
- Key idea: advertise the entire path
  - ✔ Distance vector: send *distance metric* per dest d
  - ✔ Path vector: send the *entire path* for each dest d

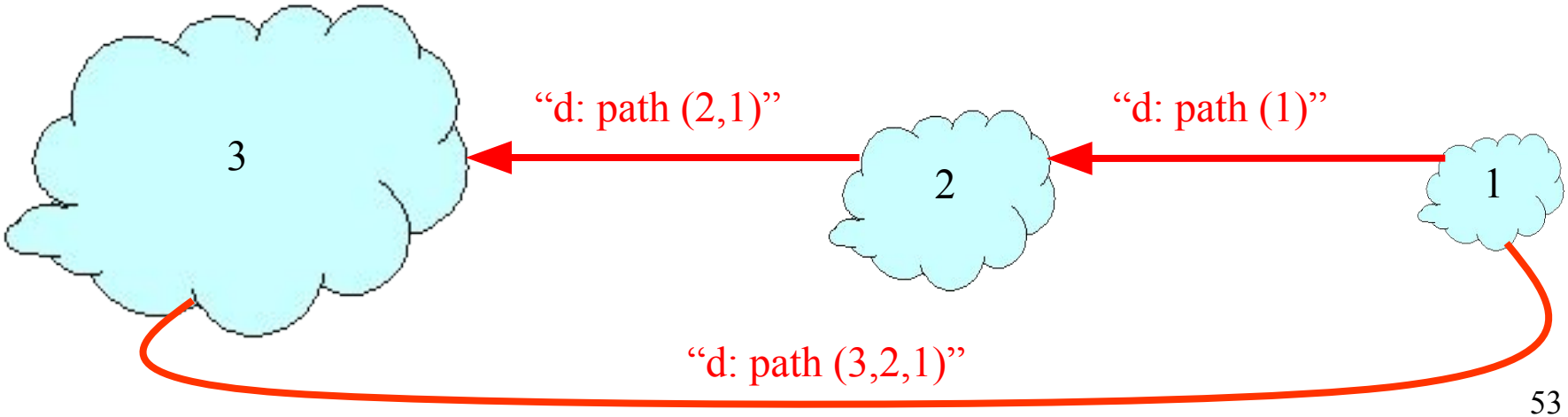


Destination	Path
A1	AS1
A2	AS1

Destination	Path
A1 . .	AS1
A3	AS1
B1	AS1-AS2
B2	AS1-AS2
C1	AS1-AS3
C2	AS1-AS3
C3	AS1-AS3

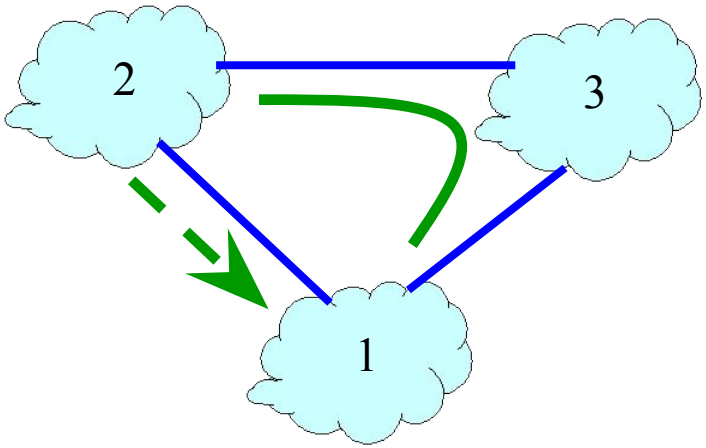
Faster Loop Detection

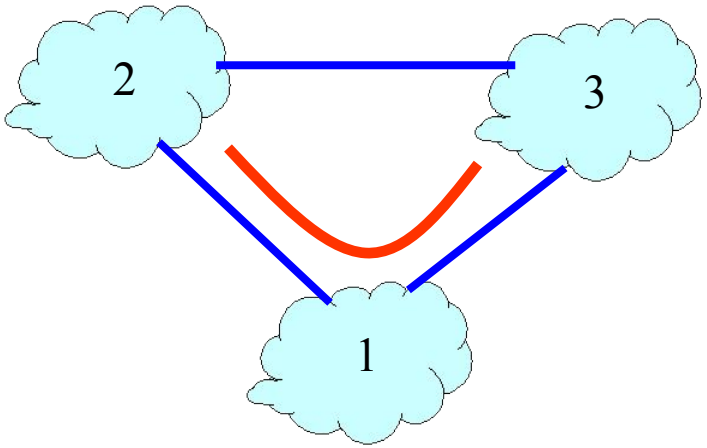
- Node can easily detect a loop
  - ✔ Look for its own node identifier in the path
  - ✔ E.g., node 1 sees itself in the path “3, 2, 1”
- Node can simply discard paths with loops
  - ✔ E.g., node 1 simply discards the advertisement



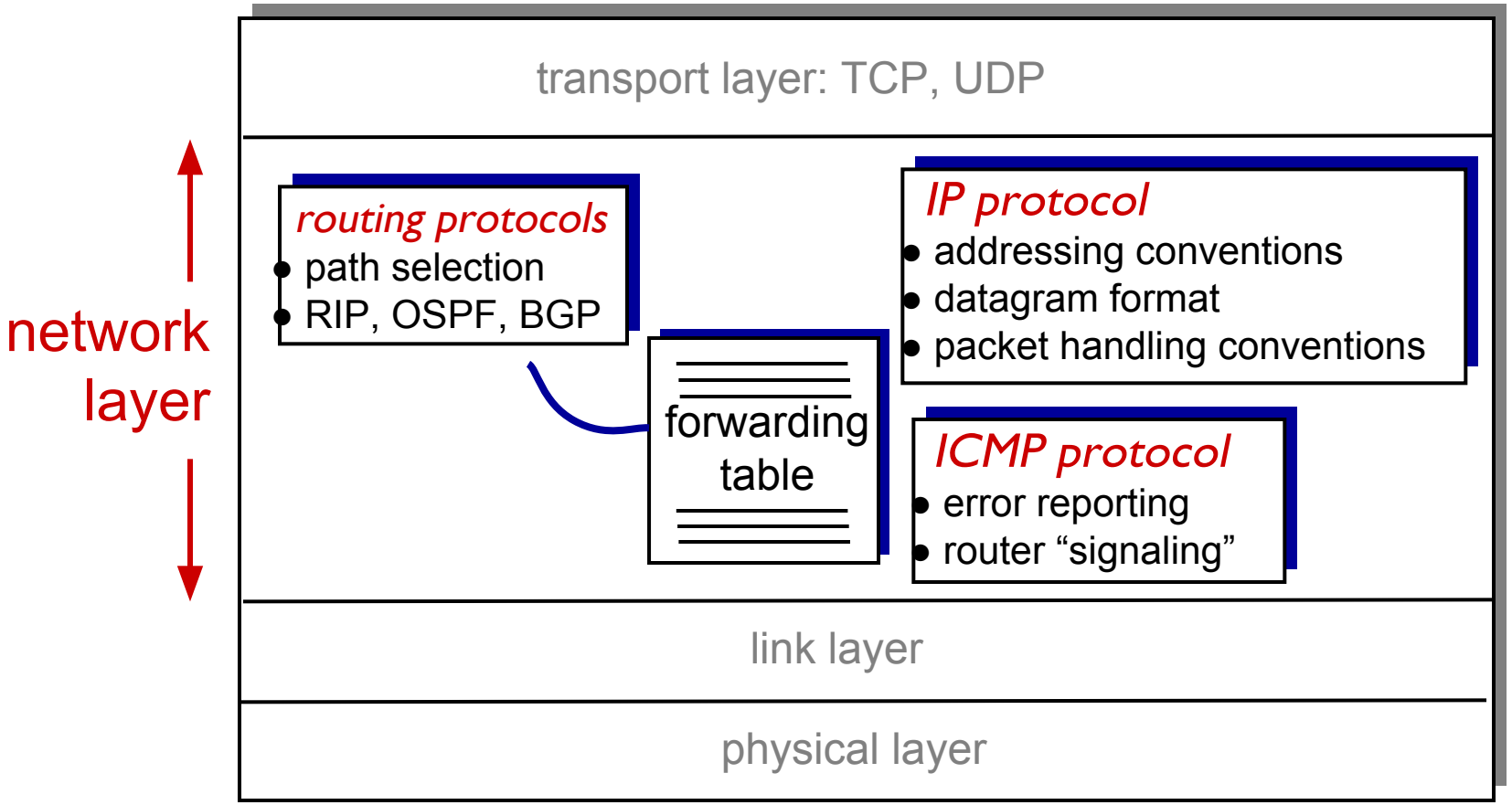
Flexible Policies

- Each node can apply local policies
  - ✔ Path selection: Which path to use?
  - ✔ Path export: Which paths to advertise?
- Examples
  - ✔ Node 2 may prefer the path “2, 3, 1” over “2, 1”
  - ✔ Node 1 may not let node 3 hear the path “1, 2”

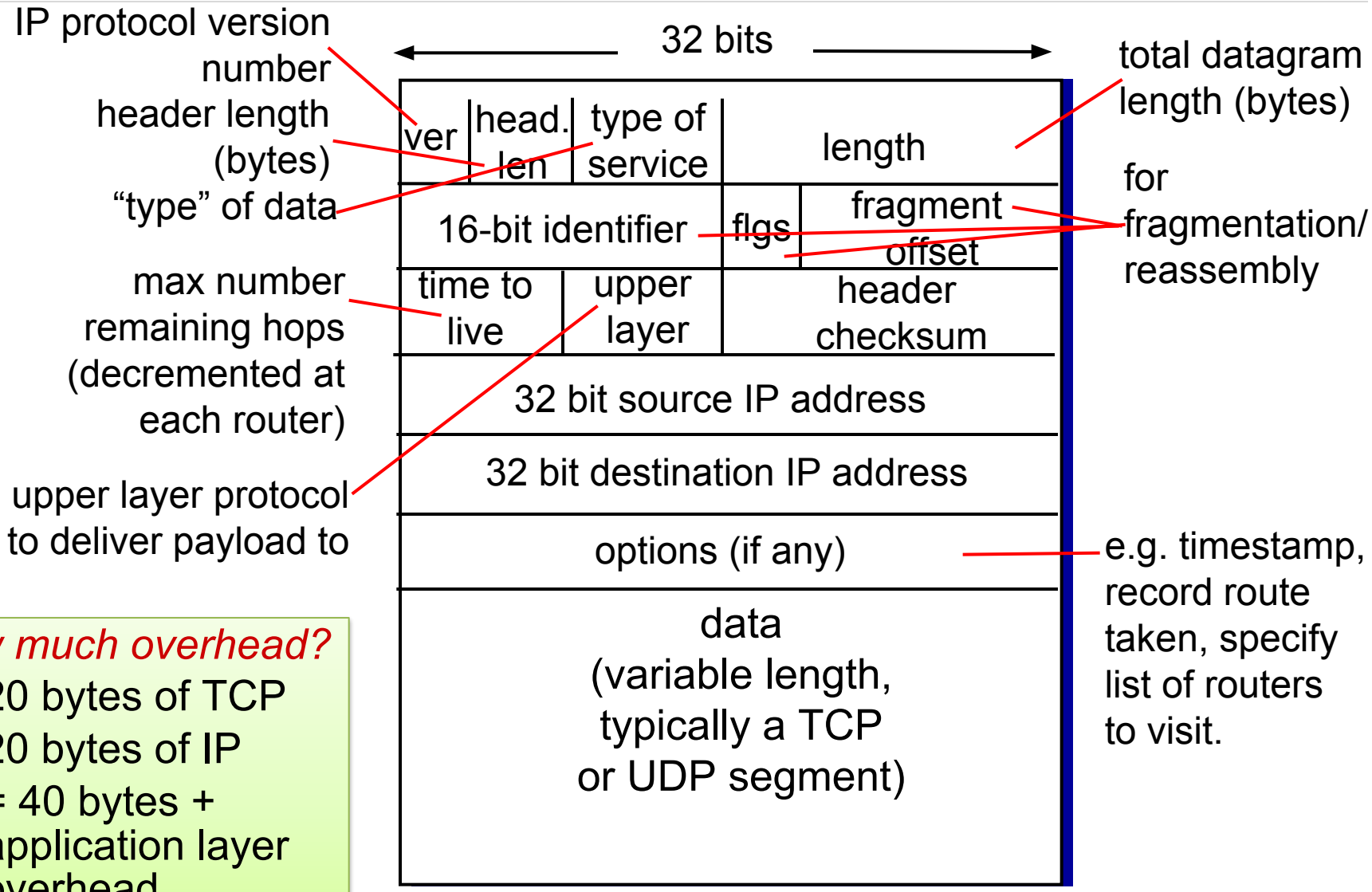




Internet Network Layer: Internet Protocol



IPv4 Datagram format

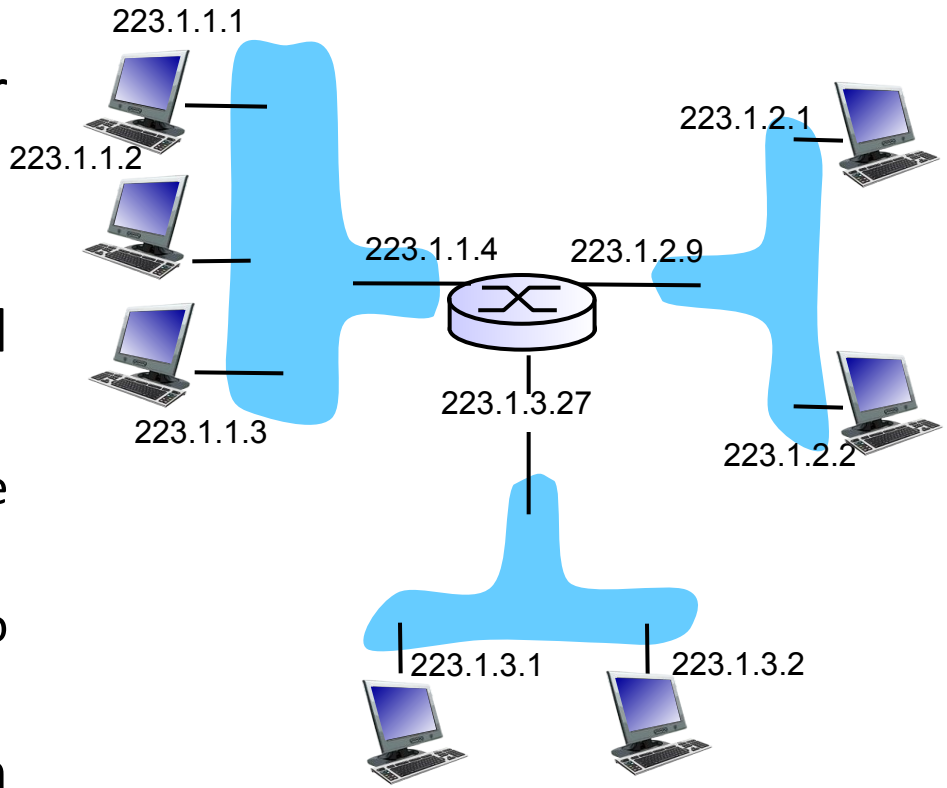


how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + application layer overhead

IP Addressing - Example

- **IP address:** It is 32-bit identifier for host, router interface
- **Interface:** It is a connection between host/router and physical link.
  - ✔ A router's typically have multiple interfaces
  - ✔ A host typically has one or two interfaces
- IP addresses associated with each interface.



223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

IP Address

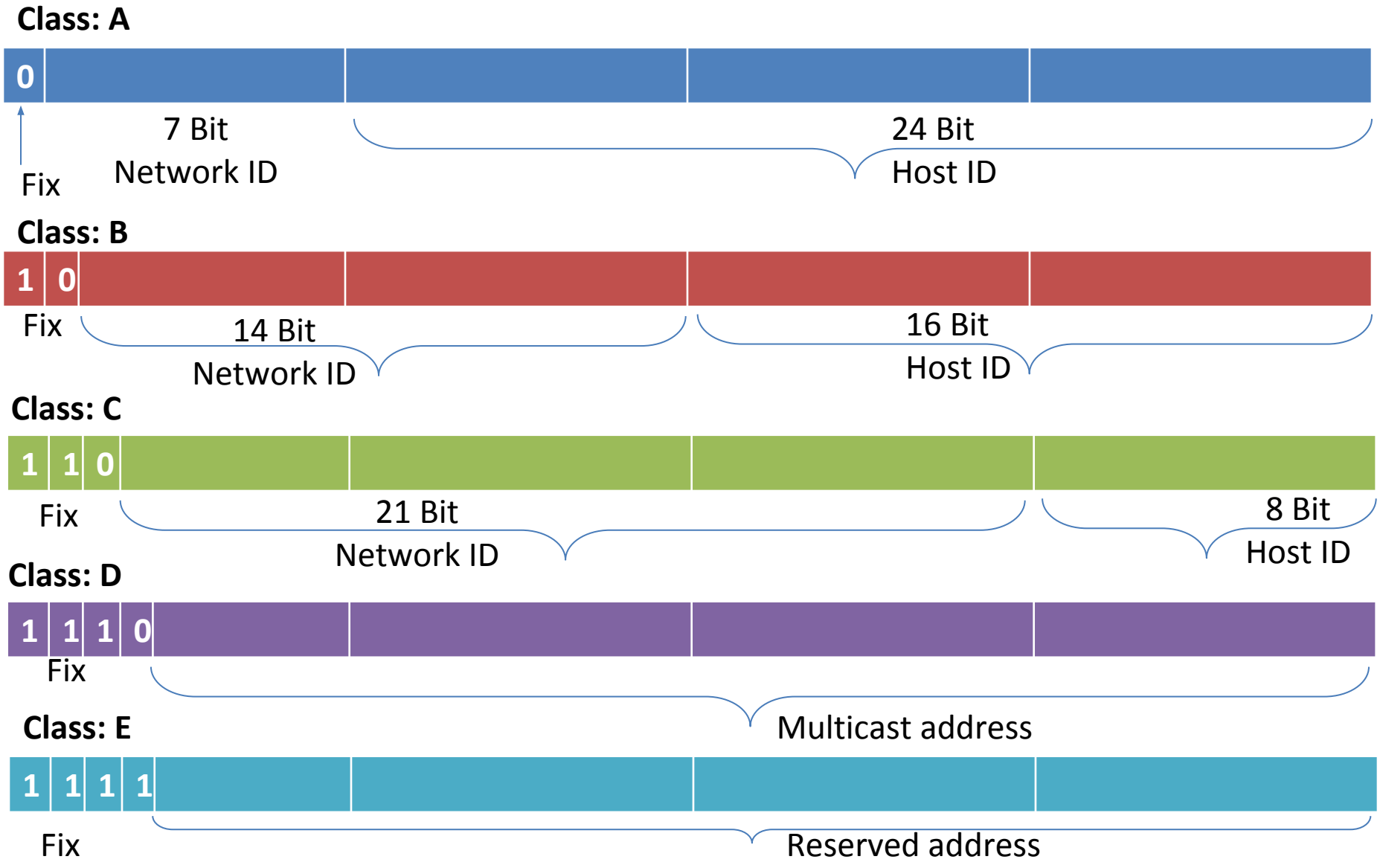
- IP addresses are useful in identifying a specific host in a network.
- IP addresses are 32 bit numbers which are divided into 4 octets. Each octet represents 8 bit binary number.
- Below is an example of an IP address:

10101100	00010000	11111110	00000001
172	**16**	**254**	**1**

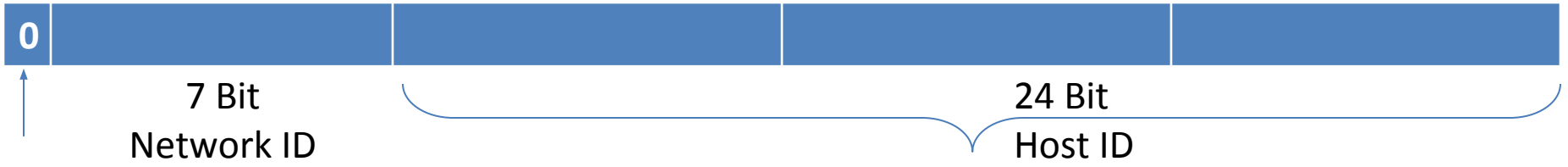
IP addresses are divided into 2 parts:
Network ID & Host ID

<NID> <HID> = IP Address

Classification of IP Addresses (Classful Addressing)



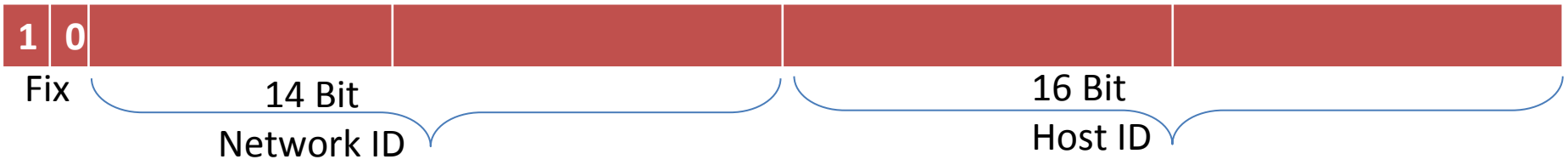
Class A: Range(0.0.0.0 to 127.255.255.255) Dotted decimal notation



- Only 126 addresses are used for network address.
- All 0's and 1's in Network-ID are dedicated for special IP address.
  So, total number of IP address in class A can be represented:

0.0.0.0	Special IP Address
00000001.0.0.1 1.0.0.2 1.0.0.3 . . . 126.255.255.254	$2^{24} - 2$ are Host IP
127.255.255.255	Special IP Address – Loopback

Class B: Range (128.0.0.0 to 191.255.255.255)



128.0.0.0	Special IP Address
10000001.0.0.1 130.0.0.2 130.0.0.3 . . . 190.255.255.254	2^{16} – 2 are Host IP
10111111.255.255.255	Special IP Address – Loopback

Class C: Range(192.0.0.0 to 223.255.255.255)

192.0.0.0	Special IP Address
11000001.0.0.1 194.0.0.2 194.0.0.3 . . . 222.255.255.254	2^8 – 2 are Host IP
11011111.255.255.255	Special IP Address – Loopback

Class D: Range (224.0.0.0 to 239.255.255.255)

- Very first four bits of the first octet in Class D IP addresses are set to 1110, giving a range of:

11100000 – **1110**1111
224 – 239

- Class D has IP address rage from 224.0.0.0 to 239.255.255.255.
- Class D is reserved for Multicasting.
- In multicasting data is not destined for a particular host, that is why there is no need to extract host address from the IP address, and Class D does not have any subnet mask.

Class E: (240.0.0.0 to 255.255.255.255)

- This IP Class is reserved for experimental purposes only for R&D or Study.
- IP addresses in this class ranges from 240.0.0.0 to 255.255.255.254.
- Like Class D, this class too is not equipped with any subnet mask.

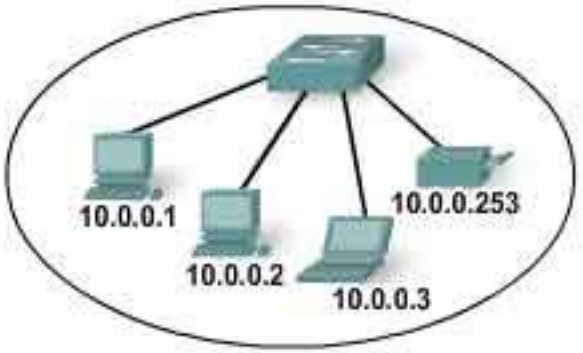
Type of addresses in IPv4 Network

- **Network address** - The address by which we refer to the network.
  - ✔ E.g.: 10.0.0.0
- **Broadcast address** - A special address used to send data to all hosts in the network.
  - ✔ The broadcast address uses the highest address in the network range.
  - ✔ E.g.: 10.0.0.255
- **Host addresses** - The addresses assigned to the end devices in the network.
  - ✔ E.g.: 10.0.0.1

Type of addresses – Cont…

Address Types

	Network		Host
Network Address	10	0	0	0
	00001010	00000000	00000000	00000000
Broadcast Address	10	0	0	255
	00001010	00000000	00000000	11111111
Host Address	10	0	0	1
	00001010	00000000	00000000	00000001



Assignment

1. Change the following IP addresses from dotted-decimal notation to binary notation.

a. 114.34.2.8

b. 129.14.6.8

2. Change the following IP addresses from binary notation to dotted-decimal notation.

a. 01111111 11110000 01100111 01111101

b. 10101111 11000000 11111000 00011101

3. Find the class of the following IP addresses.

a. 11110111 11110011 10000111 11011101

b. 10101111 11000000 11110000 00011101

4. Find the netid and the hostid of the following IP addresses.

a. 114.34.2.8

b. 132.56.8.6

Masking Example To extract network and host network

- Extract network:

```
IP Address:   10110101 11011101 01010100 01110010
      Mask: & 11111111 11111111 00000000 00000000
-------------------------------------------------
    Result:   10110101 11011101 00000000 00000000
```

- Extract host:

```
IP Address:   10110101 11011101 01010100 01110010
Subnet Mask: & ~(11111111 11111111 00000000 00000000)
-------------------------------------------------
    Result:   00000000 00000000 01010100 01110010
```

Subnets

- Problem: need to break up large A and B classes
- Solution: add another layer to the hierarchy
  - ✔ From the outside, appears to be a single network
    - Only 1 entry in routing tables
  - ✔ Internally, manage multiple subnetworks
    - Split the address range using a subnet mask

Subnet Mask: 11111111 11111111 11000000 00000000

Subnet Example

- Extract network:

IP Address: 10110101 11011101 01010100 01110010

Subnet Mask: & 11111111 11111111 11000000 00000000

Result: 10110101 11011101 01000000 00000000

- Extract host:

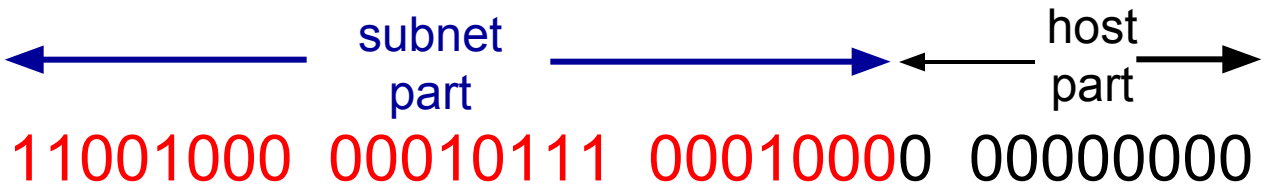
IP Address: 10110101 11011101 01010100 01110010

Subnet Mask: & ~(11111111 11111111 11000000 00000000)

Result: 00000000 00000000 00010100 01110010

Classless Inter-Domain Routing(CIDR)

- CIDR is a slash notation of subnet mask. CIDR tells us number of on bits in a network address.



200.23.16.0/23

- A single IP address can be used to designate many unique IP addresses with CIDR.
- A CIDR IP address looks like a normal IP address except that it ends with a slash followed by a number, called the IP network prefix.
- CIDR addresses reduce the size of routing tables and make more IP addresses available within organizations.

Subnetting

- Subnetting take places when we extend the default subnet mask.
- We cannot perform subnetting with default subnet mask and every classes have default subnet mask.
- Now find the host bits borrowed to create subnets and convert them in decimal.
- For example find the subnet mask of address 188.25.45.48/20 ?
  1. Class B, Default Subnet mask: 255.255.0.0
  2. Borrowed 4 bit from host part so mask is now:

     11111111 11111111 11110000 00000000

     255 255 240 0

How many subnets from given subnet mask?

- To calculate the number of subnets provided by given subnet mask we use 2^N , where N = number of bits borrowed from host bits to create subnets.
- For example in 192.168.1.0/27, N is 3.
- By looking at address we can determined that this address is belong to class C and default subnet mask 255.255.255.0 [/24 in CIDR].
- In given address we borrowed 27 - 24 = 3 host bits to create subnets.
- Now $2^3 = 8$, so our answer is 8 (number of subnets).

What are the valid subnets?

- Calculating valid subnet is two steps process.
- First calculate total subnet by using formula 2^N.
- In second step find the block size and count from zero in block until subnet mask value.
- For example calculate the valid subnets for 192.168.1.0/26
  1. Borrowed host bits are 2 [26-24]
  2. Total subnets are $2^2 = 4$
  3. Subnet mask would be 255.255.255.192
  4. Block size would be 256-192 = 64
  5. Start counting from zero at blocks of 64, so our valid subnets would be 0,64,128,192

What are the total hosts?

- Total hosts are the hosts available per subnet
- To calculate total hosts use formula 2^H = Total hosts
- H is the number of host bits
- For example in address 192.168.1.0/26
- We have 32 - 26
  1. [Total bits in IP address - Bits consumed by network address] = 6
  2. Total hosts per subnet would be $2^6 = 64$

IP Addressing Summary

Class	Leading bits	Size of *network number* bit field	Size of *rest* bit field	Number of networks	Addresses per network	Total addresses in class	Start address	End address	Default subnet mask in dot-decimal notation	CIDR notation
Class A	0	8	24	128 (2^7)	16,777,216 (2^{24})	2,147,483,648 (2^{31})	0.0.0.0	127.255.255.255	255.0.0.0	/8
Class B	10	16	16	16,384 (2^{14})	65,536 (2^{16})	1,073,741,824 (2^{30})	128.0.0.0	191.255.255.255	255.255.0.0	/16
Class C	110	24	8	2,097,152 (2^{21})	256 (2^8)	536,870,912 (2^{29})	192.0.0.0	223.255.255.255	255.255.255.0	/24
Class D (multicast)	1110	not defined	not defined	not defined	not defined	268,435,456 (2^{28})	224.0.0.0	239.255.255.255	not defined	not defined
Class E (reserved)	1111	not defined	not defined	not defined	not defined	268,435,456 (2^{28})	240.0.0.0	255.255.255.255	not defined	not defined

Assignment

1. In a block of addresses, we know the IP address of one host is 25.34.12.56/16. What are the first address (network address) and the last address (limited broadcast address) in this block?

2. Find the range of addresses in the following blocks (For both network and host).

a. 123.56.77.32/29

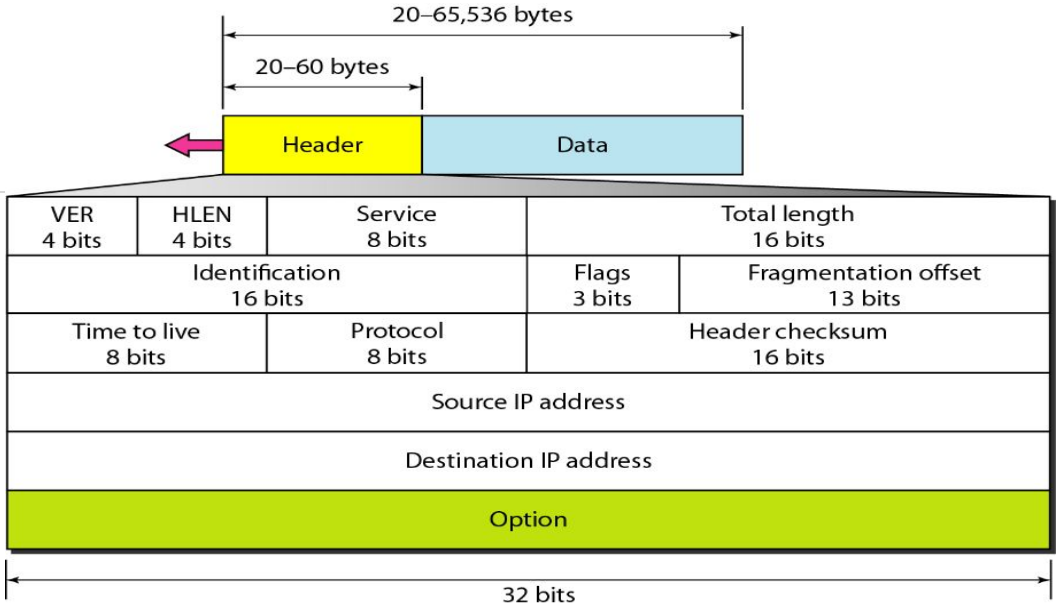
b. 200.17.21.128/27

3. Write the following masks in slash notation *(In).*

a. 255.255.255.0

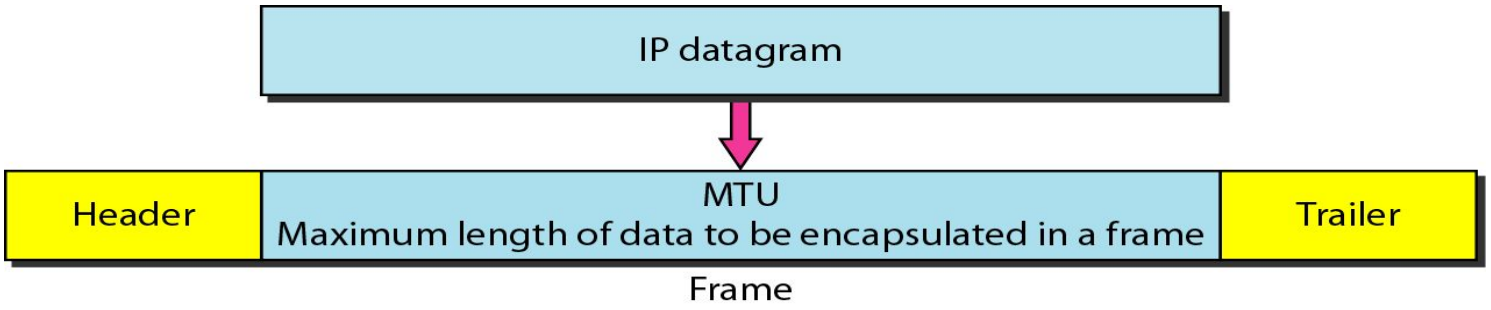
b. 255.0.0.0

Fragmentation



- A datagram can travel through different networks. Each router decapsulates the IPv4 datagram from the frame it receives, processes it, and then encapsulates it in another frame.
- The format and size of the received frame depend on the protocol used by the physical network through which the frame has just traveled. The format and size of the sent frame depend on the protocol used by the physical network through which the frame is going to travel.
- For example, if a router connects a LAN to a WAN, it receives a frame in the LAN format and sends a frame in the WAN format.

- To make the IPv4 protocol independent of the physical network, the designers decided to make the maximum length of the IPv4 datagram equal to 65,535 bytes.
- This makes transmission more efficient if we use a protocol with an MTU of this size. However, for other physical networks, we must divide the datagram to make it possible to pass through these networks. This is called **fragmentation.**
- The source usually does not fragment the IPv4 packet. The transport layer will instead segment the data into a size that can be accommodated by IPv4 and the data link layer in use.
- When a datagram is fragmented, each fragment has its own header with most of the fields repeated, but with some changed.



Fields Related to Fragmentation

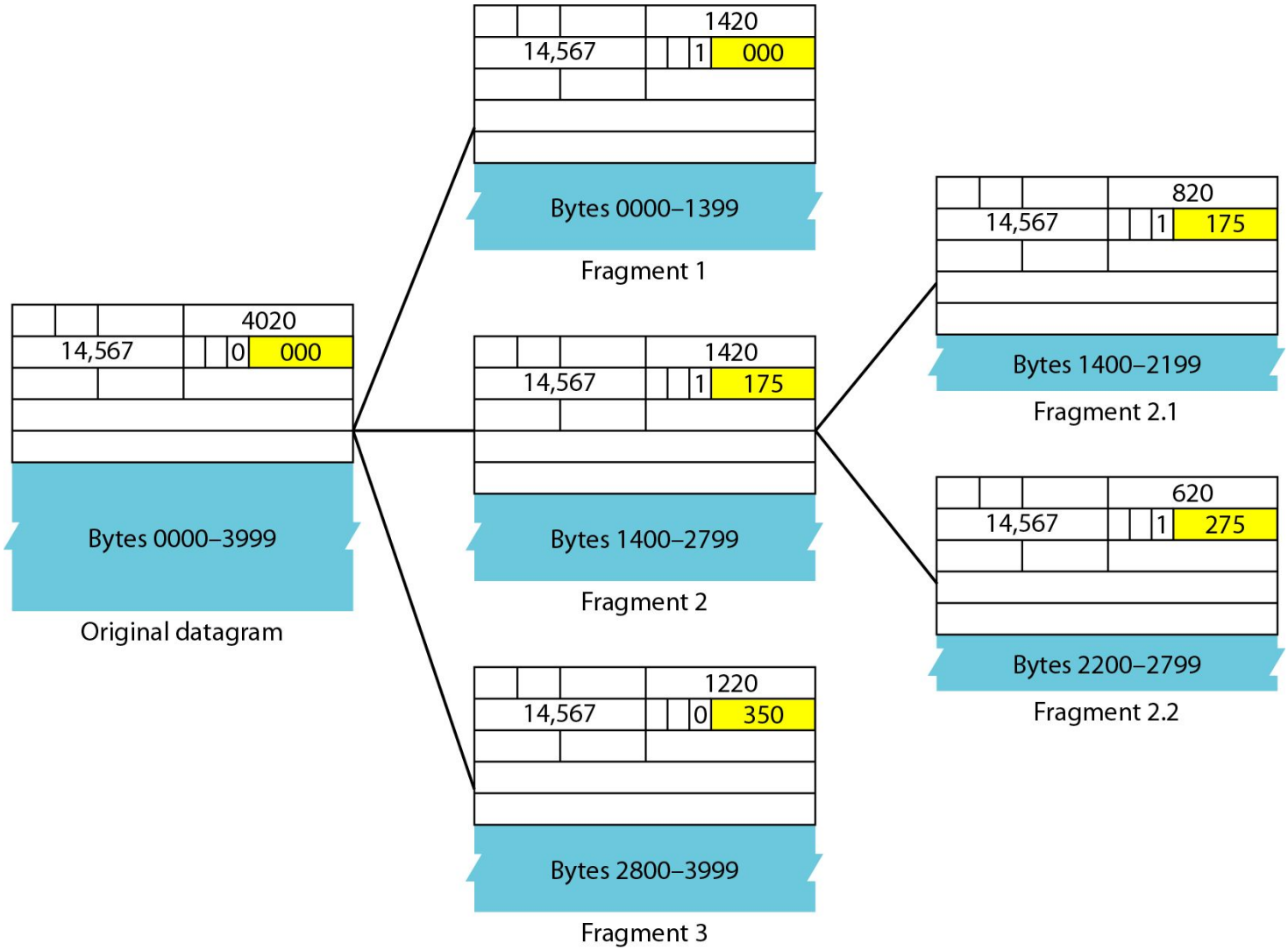
- **Identification:** identifies a datagram originating from the source host. A combination of the identification and source address must uniquely define a datagram as it leaves the source node.
- **Flags:** see next slide.
- **Fragmentation offset:** is the offset of the data in the original datagram measured in units of 8 bytes.

Figure 20.10 *Flags (3 bits) used in fragmentation*

- first bit: reserved (not used)
- second bit: = 1 requires the packet not to be fragmented
  drops the packet if it is > MTU

- third bit: =1 more fragmented packets later
  =0 the last fragmented packet

Figure 20.12 *Detailed fragmentation example*



20-3 IPv6

The network layer protocol in the TCP/IP protocol suite is currently IPv4. Although IPv4 is well designed, data communication has evolved since the inception of IPv4 in the 1970s. IPv4 has some deficiencies that make it unsuitable for the fast-growing Internet.

Topics discussed in this section:

Advantages
Packet Format
Extension Headers

IPv6: Advantages

- Larger address space.
- Better header format.
- New options.
- Allowance for extensions.
- Support for resource allocation.
- Support for more security.

Figure 20.15 *IPv6 datagram header and payload*

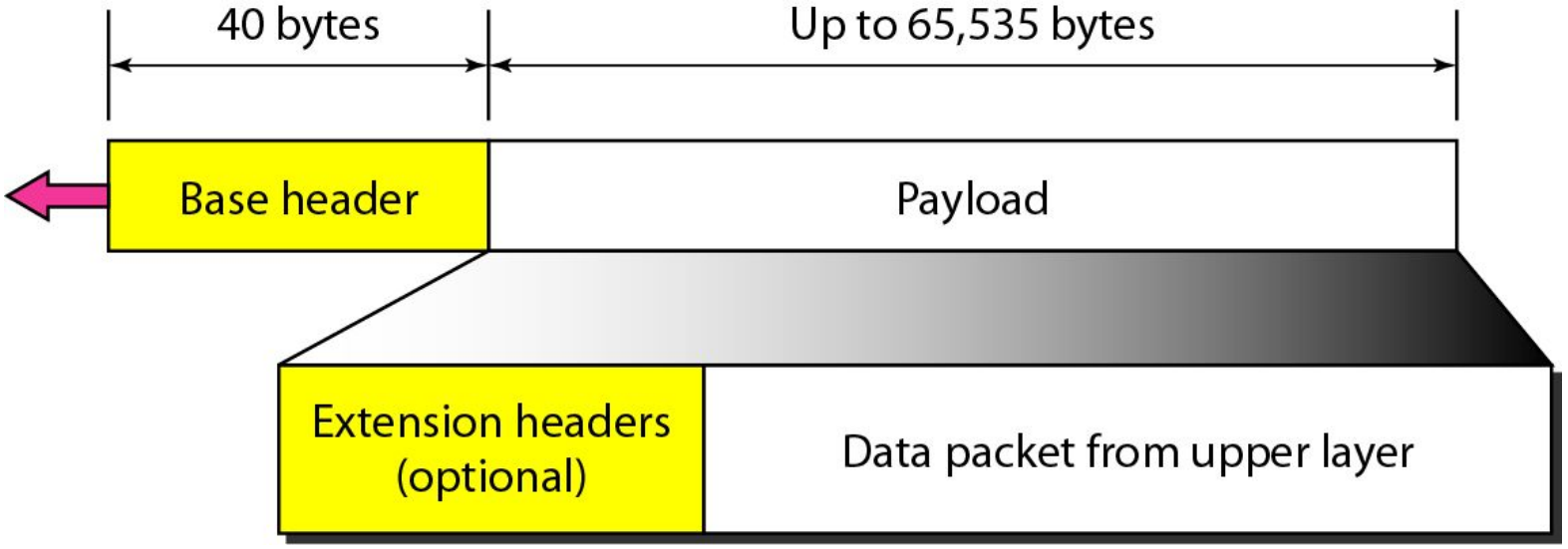


Figure 20.16 *Format of an IPv6 datagram*

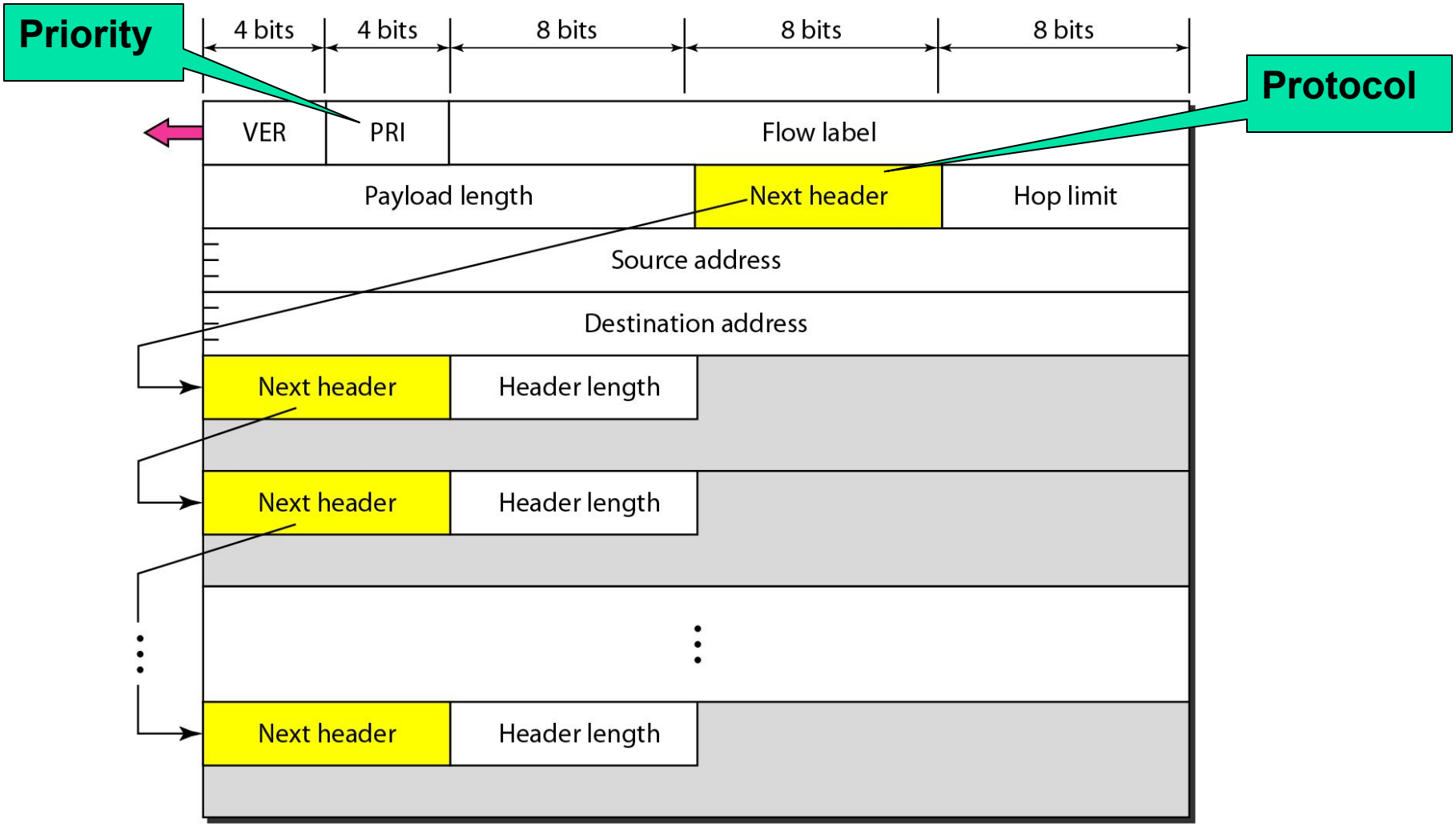


Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

Comparison
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

Difference between IPv4 & IPv6

IPv4	IPv6
✔ 32 bit length	✔ 128 bit length
✔ Fragmentation is done by sender and forwarding routers	✔ Fragmentation is done only by sender
✔ No packet flow identification	✔ Packet flow identification is available within the IPv6 header using the Flow Label field
✔ Checksum field in header	✔ No checksum field in header
✔ Options fields are available in header	✔ No option fields, but Extension headers are available
✔ Address Resolution Protocol (ARP)is available to map IPv4 addresses to MAC addresses	✔ Address Resolution Protocol (ARP) is replaced with Neighbor Discovery Protocol
✔ Broadcast messages are available	✔ Broadcast messages are not available
✔ Static IP addresses or DHCP is required to configure IP addresses	✔ Auto-configuration of addresses is available

21-1 ADDRESS MAPPING

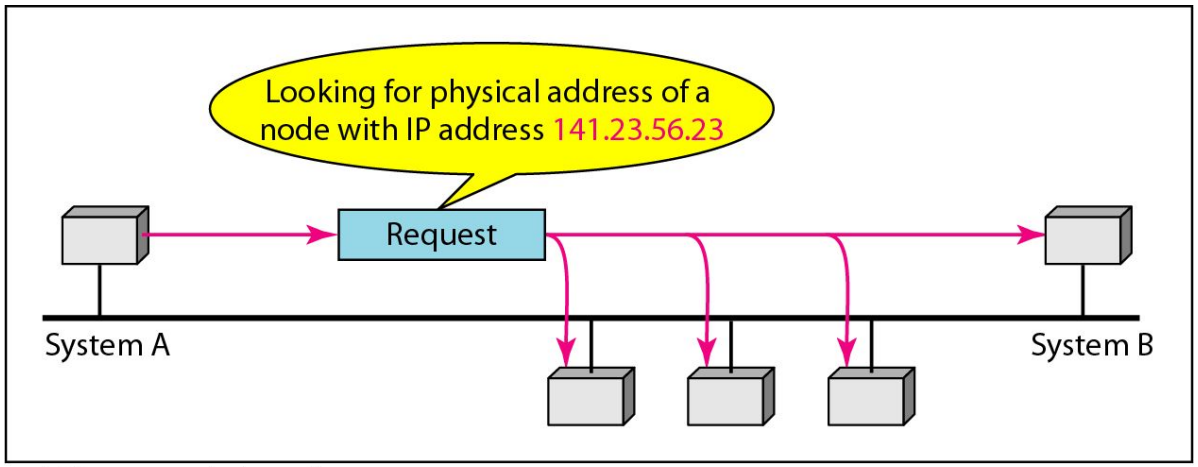
The delivery of a packet to a host or a router requires two levels of addressing: logical and physical. We need to be able to map a logical address to its corresponding physical address and vice versa. This can be done by using either static or dynamic mapping.

Topics discussed in this section:

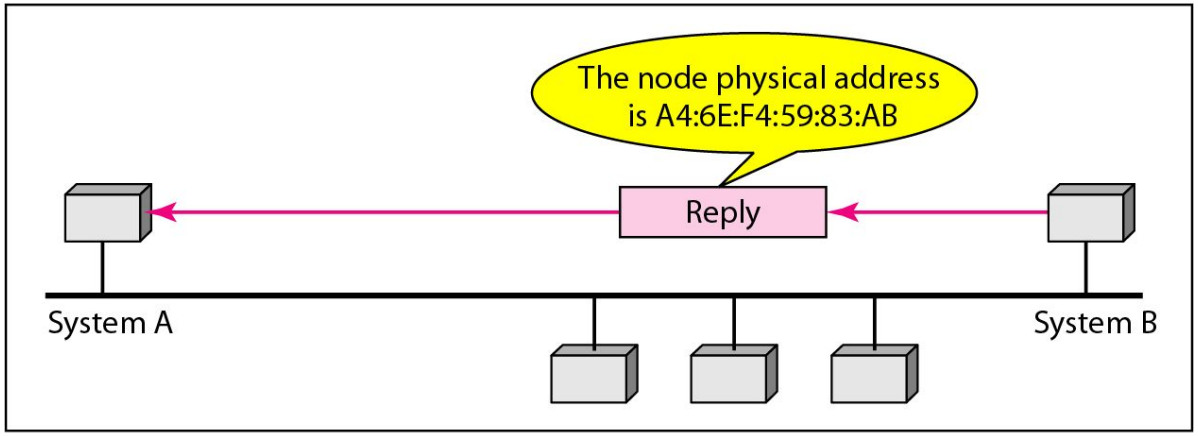
Mapping Logical to Physical Address
Mapping Physical to Logical Address

Figure 21.1 Mapping Logical to Physical Address

ARP (address resolution protocol)



a. ARP request is broadcast

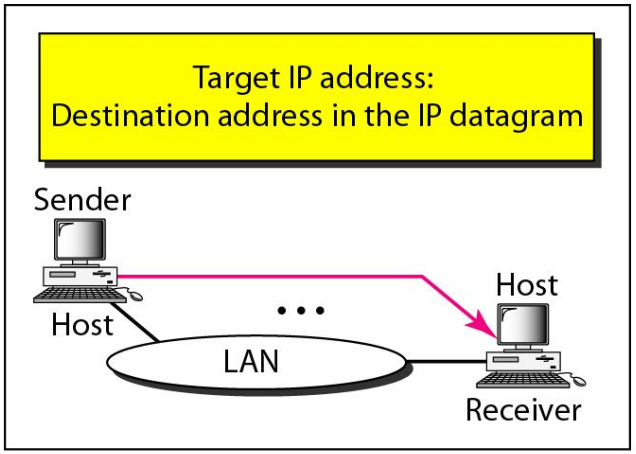


b. ARP reply is unicast

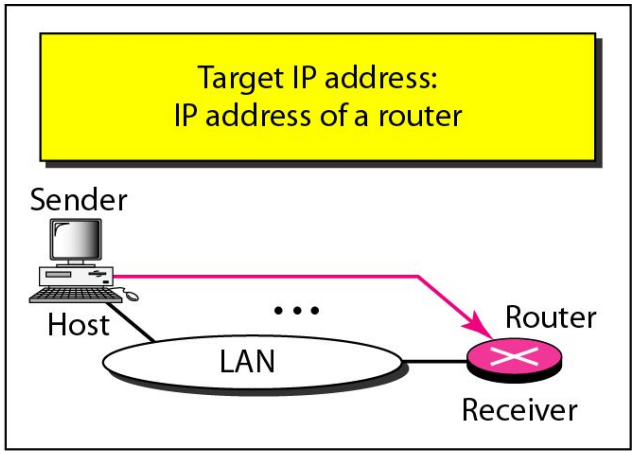
Note

ARP can be useful if the ARP reply is cached (kept in cache memory for a while).

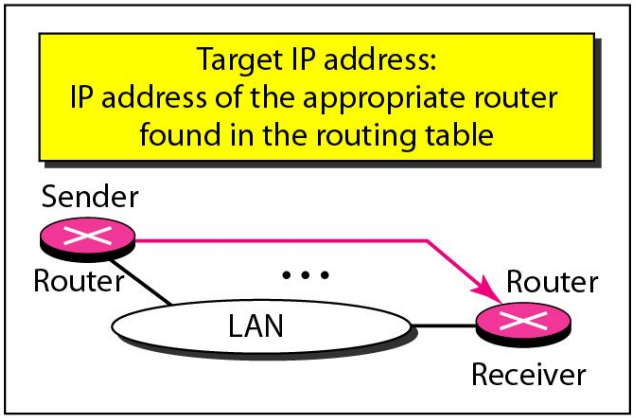
Figure 21.4 *Four cases using ARP*



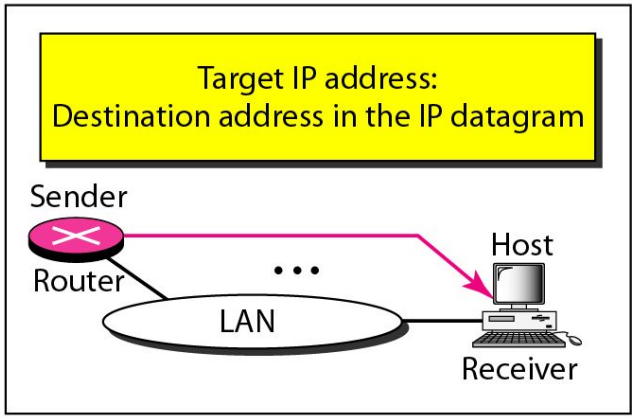
Case 1. A host has a packet to send to another host on the same network.



Case 2. A host wants to send a packet to another host on another network. It must first be delivered to a router.



Case 3. A router receives a packet to be sent to a host on another network. It must first be delivered to the appropriate router.



Case 4. A router receives a packet to be sent to a host on the same network.

Mapping Physical to Logical Address: RARP, BOOTP, and DHCP

- A diskless station just booted.
- An organization does not have enough IP addresses to assign to each station.

Reverse Address Resolution Protocol (ARP)

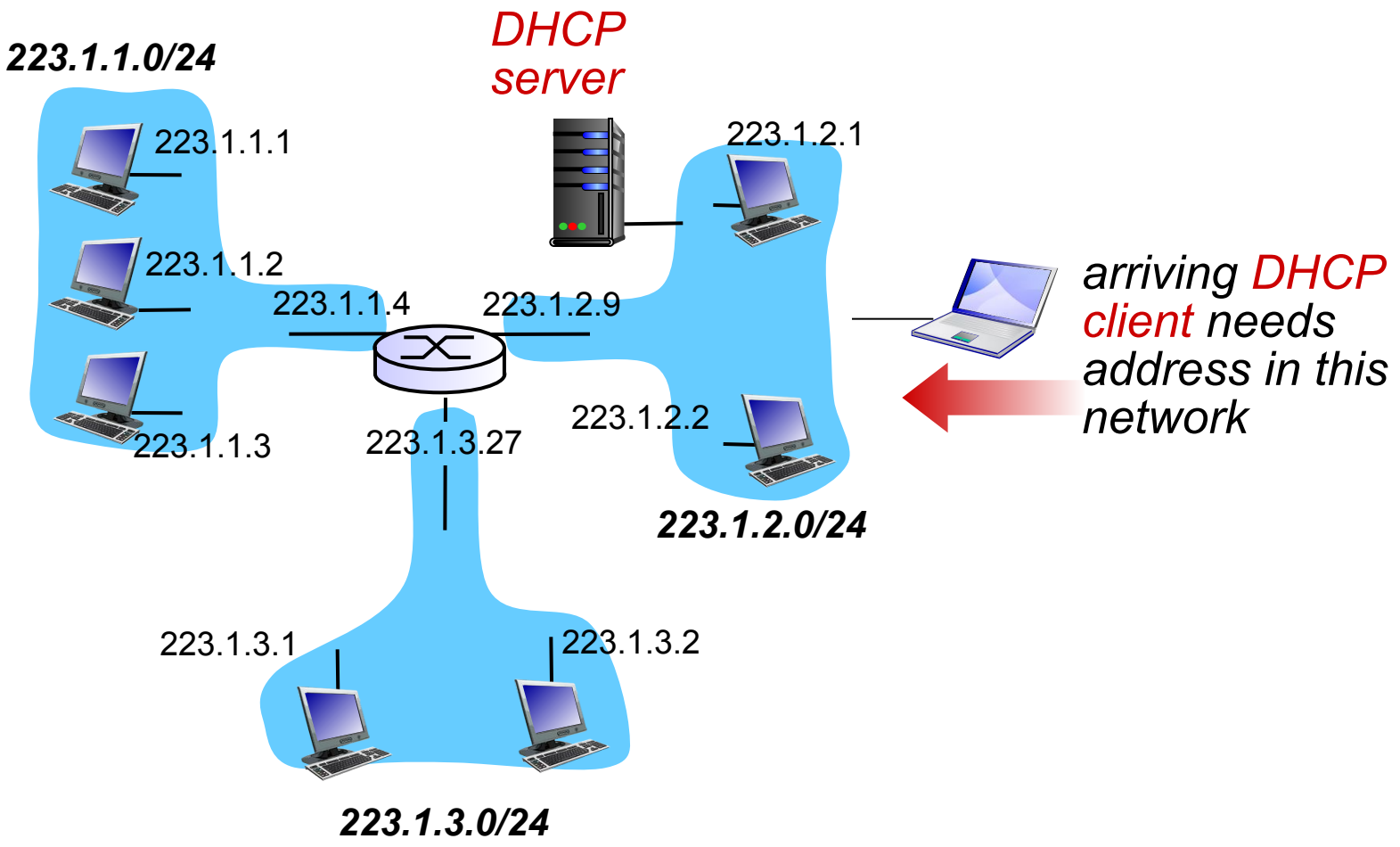
- A machine can use the physical address to get the logical address using RARP.
- A RARP messages is created and brodcast on the local network.
- The machine on the local network that knows the logical address will respond with a RARP reply.
- Broadcasting is done at data link layer.
- Broadcast requests does not pass the boundaries of a network.

Note

DHCP provides static and dynamic address allocation that can be manual or automatic.

Dynamic Host Configuration Protocol - DHCP

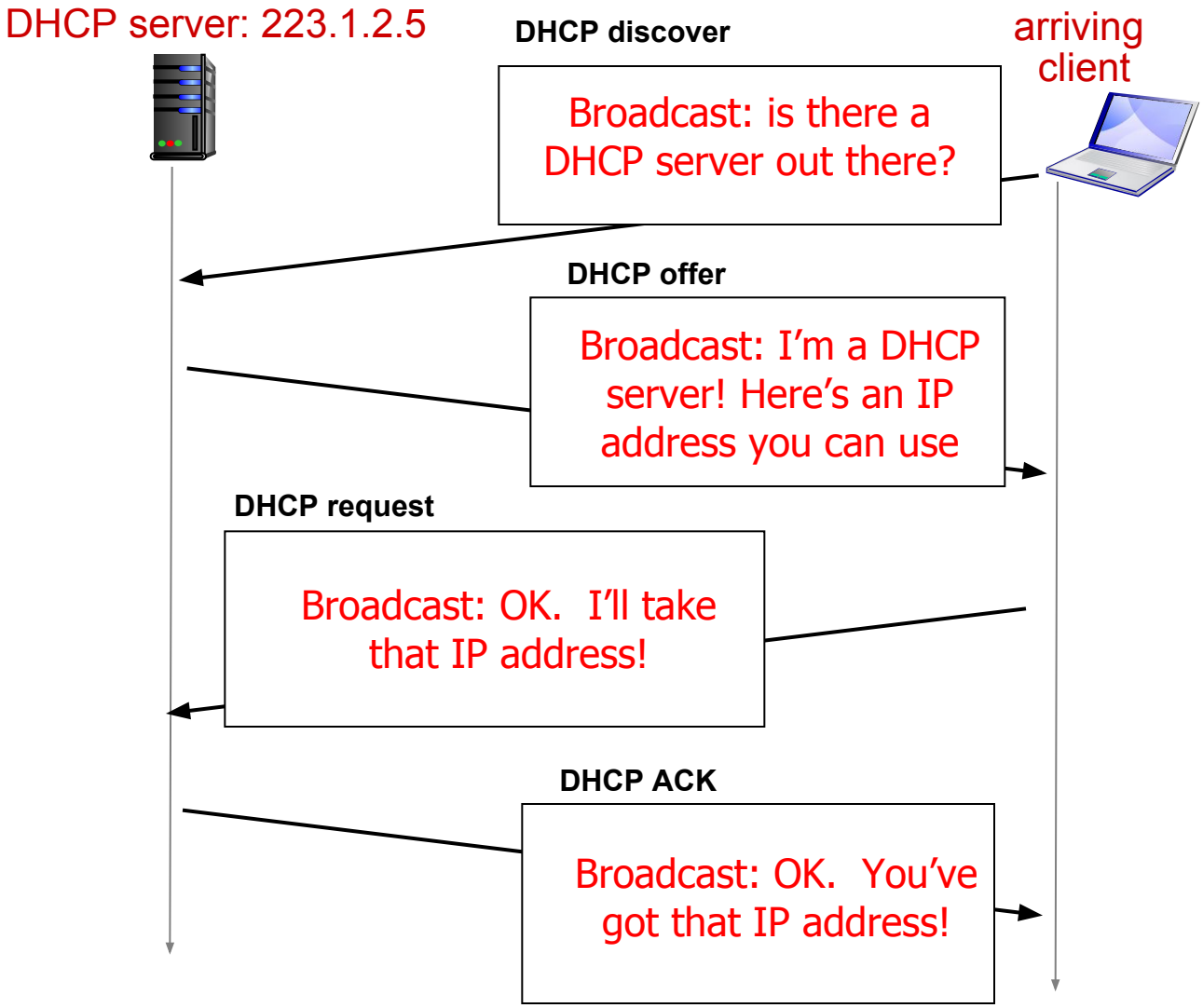
- Dynamic Host Configuration Protocol is a protocol for assigning dynamic IP addresses to devices on a network.



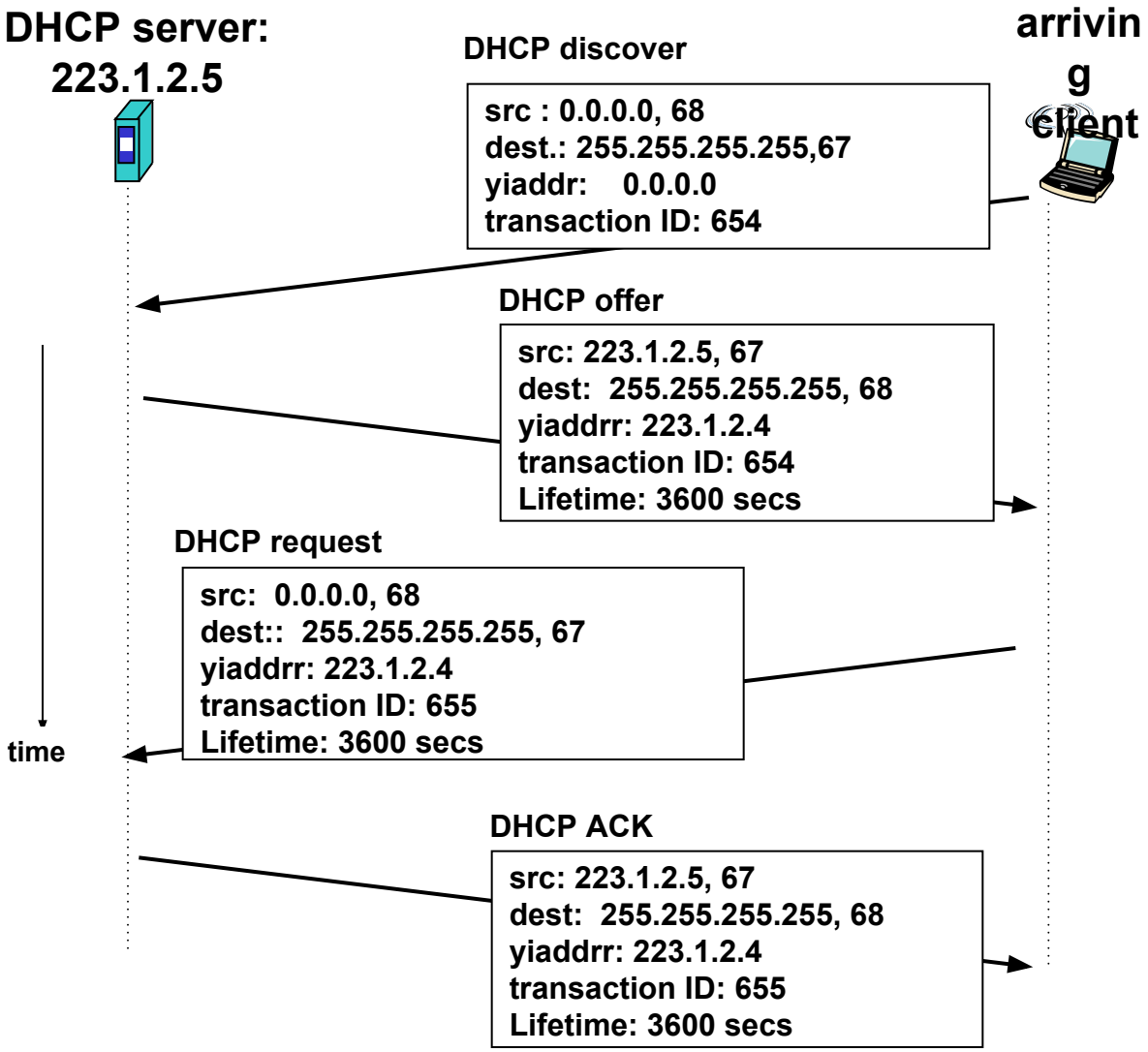
DHCP – Cont…

- With dynamic addressing, a device can have a different IP address every time it connects to the network.
- In some systems, the device's IP address can even change while it is still connected.
- It allows reuse of addresses (only hold address while connected “on”).
- It also support mobile users who want to join network.

DHCP Client Server Interaction

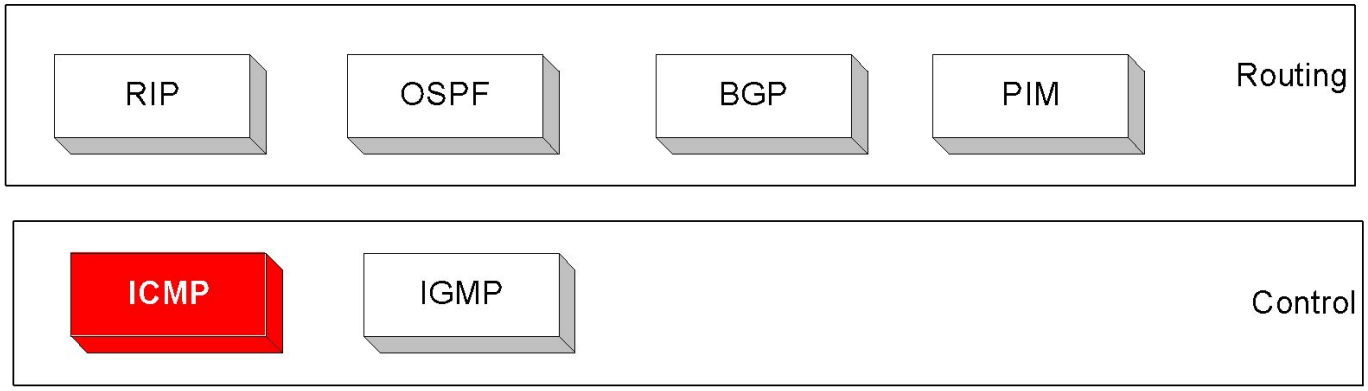


DHCP client-server scenario



Internet Control Message Protocol (ICMP)

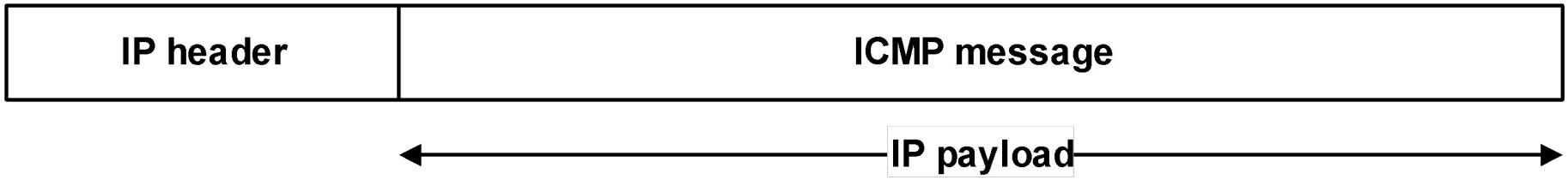
- The IP (Internet Protocol) relies on several other protocols to perform necessary control and routing functions:
  - Control functions (ICMP)
  - Multicast signaling (IGMP)
  - Setting up routing tables (RIP, OSPF, BGP, PIM, …)



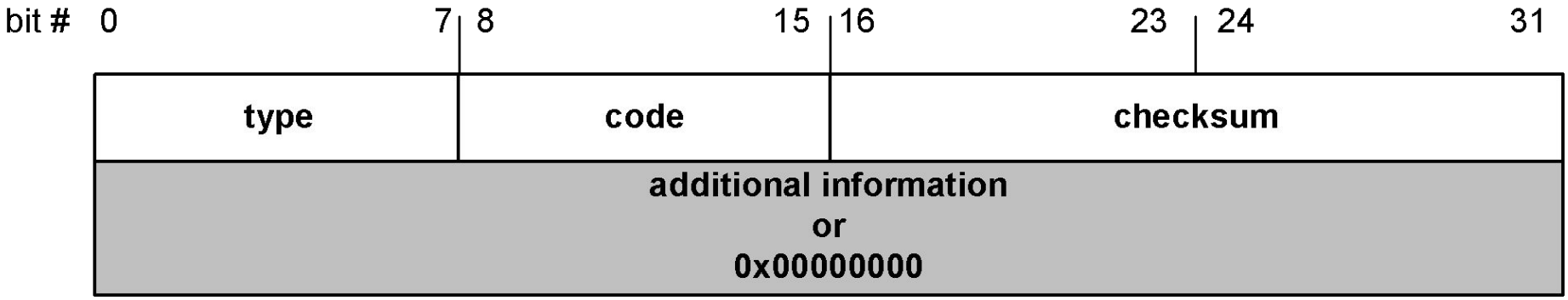
Overview

- The **Internet Control Message Protocol (ICMP)** is a helper protocol that supports IP with facility for
  - Error reporting
  - Simple queries

- ICMP messages are encapsulated as IP datagrams:



ICMP message format



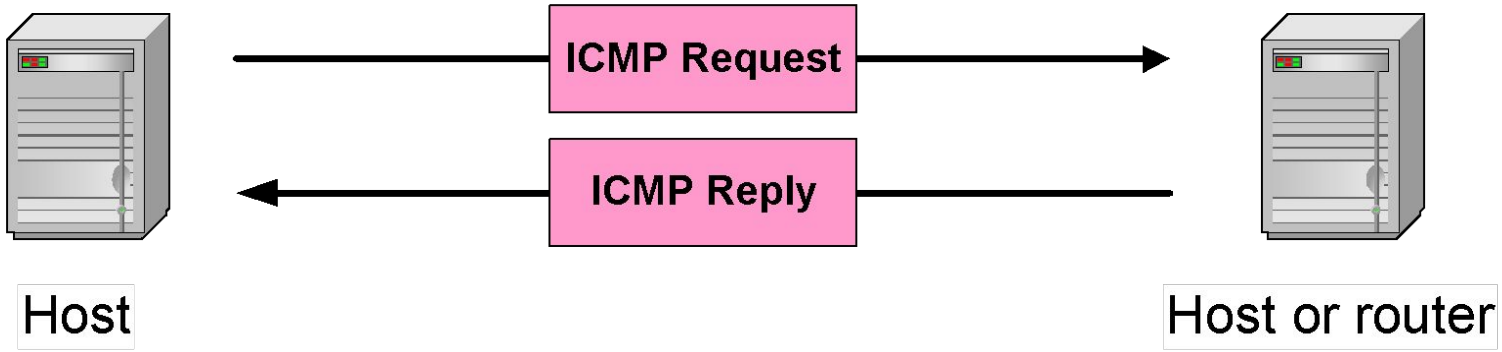
4 byte header:

- Type (1 byte): type of ICMP message (Error reporting or Query)
- Code (1 byte): subtype of ICMP message
- Checksum (2 bytes): similar to IP header checksum. Checksum is calculated over entire ICMP message

If there is no additional data, there are 4 bytes set to zero.

 each ICMP messages is at least 8 bytes long

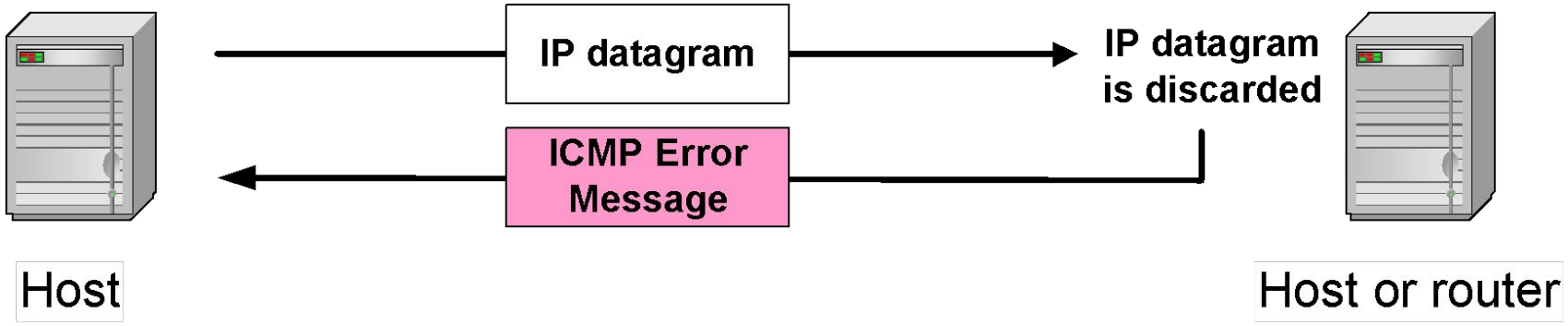
ICMP Query message



ICMP query:

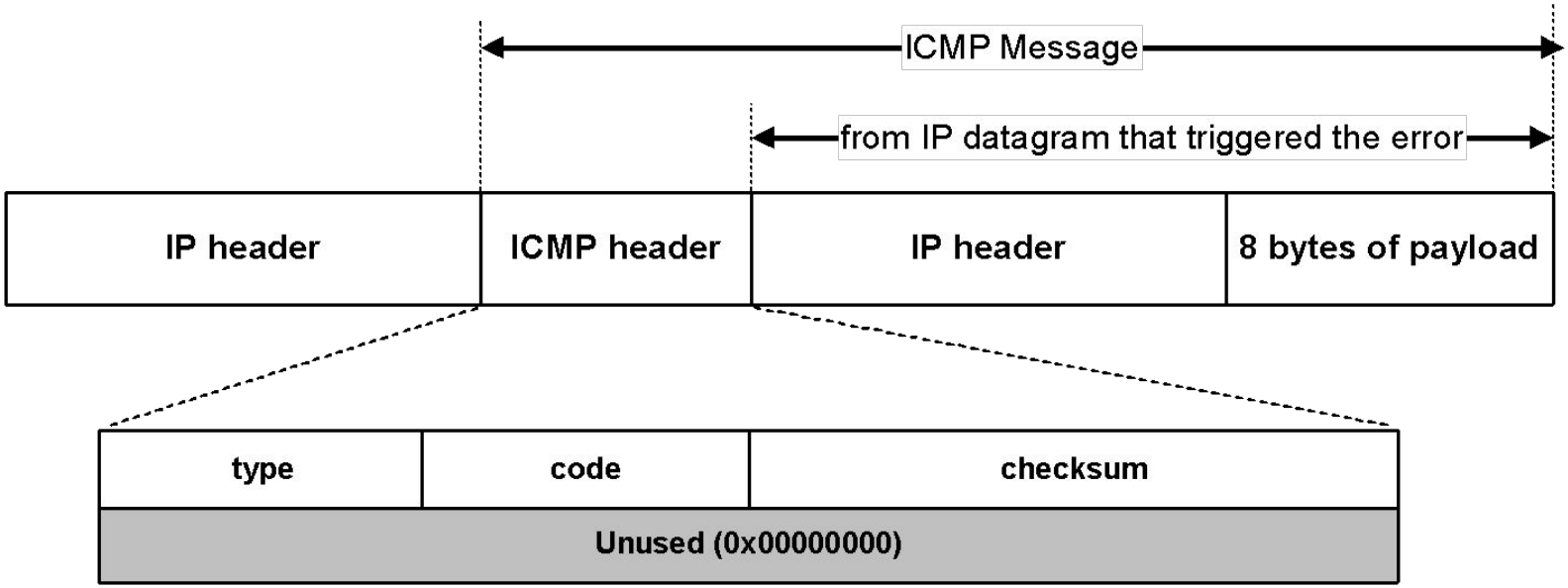
- Request sent by host to a router or host
- Reply sent back to querying host

ICMP Error message



- ICMP error messages report error conditions
- Typically sent when a datagram is discarded
- Error message is often passed from ICMP to the application program

ICMP Error message



- ICMP error messages include the complete IP header and the first 8 bytes of the payload (typically: UDP, TCP)

Frequent ICMP Error message

Type	Code	Description	
3	0–15	Destination unreachable	Notification that an IP datagram could not be forwarded and was dropped. The code field contains an explanation.
5	0–3	Redirect	Informs about an alternative route for the datagram and should result in a routing table update. The code field explains the reason for the route change.
11	0, 1	Time exceeded	Sent when the TTL field has reached zero (Code 0) or when there is a timeout for the reassembly of segments (Code 1)
12	0, 1	Parameter problem	Sent when the IP header is invalid (Code 0) or when an IP header option is missing (Code 1)

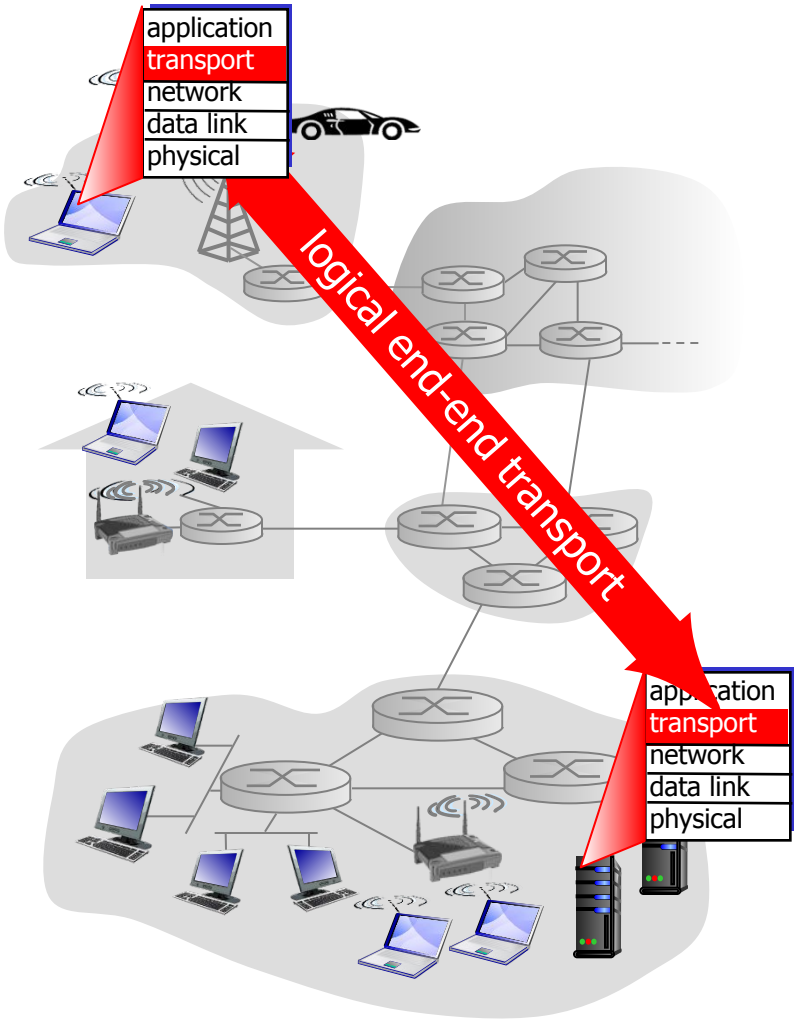
Some subtypes of the “Destination Unreachable”

Code	Description	Reason for Sending
0	Network Unreachable	No routing table entry is available for the destination network.
1	Host Unreachable	Destination host should be directly reachable, but does not respond to ARP Requests.
2	Protocol Unreachable	The protocol in the protocol field of the IP header is not supported at the destination.
3	Port Unreachable	The transport protocol at the destination host cannot pass the datagram to an application.
4	Fragmentation Needed and DF Bit Set	IP datagram must be fragmented, but the DF bit in the IP header is set.

Transport Layer

Transport Layer Services and Protocols

- It provides logical communication between application processes running on different hosts.
- A transport protocols run in end systems.
- Sender side: It breaks application messages into segments, then passes to network layer.
- Receiver side: It reassembles segments into messages, then passes to application layer.
- E.g. TCP and UDP



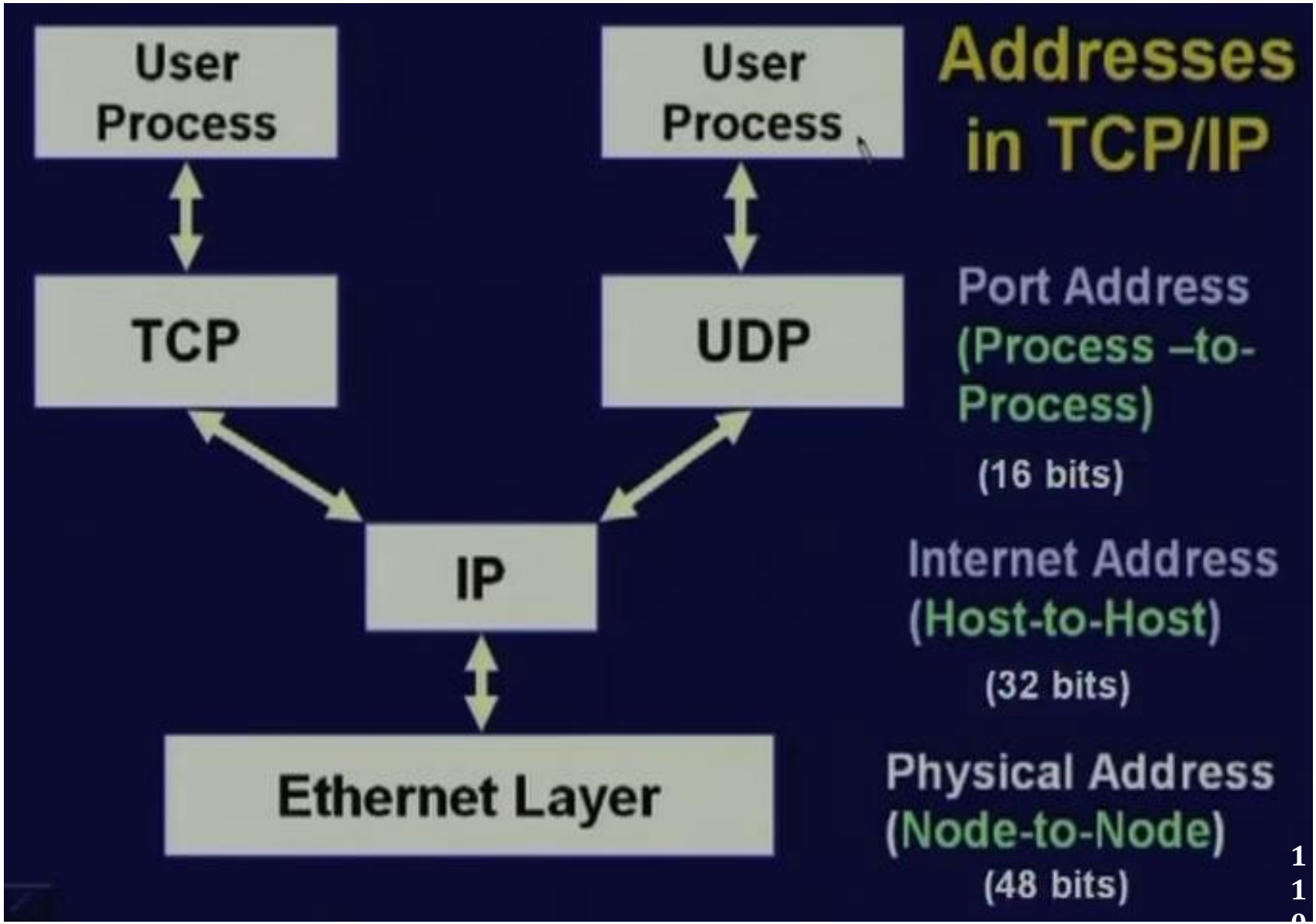

User Process
User Process
Addresses in TCP/IP
TCP
UDP
Port Address (Process –to-Process)
(16 bits)
IP
Internet Address (Host-to-Host)
(32 bits)
Ethernet Layer
Physical Address (Node-to-Node)
(48 bits)

Figure 23.8 ***Position of UDP, TCP, and SCTP in TCP/IP suite***

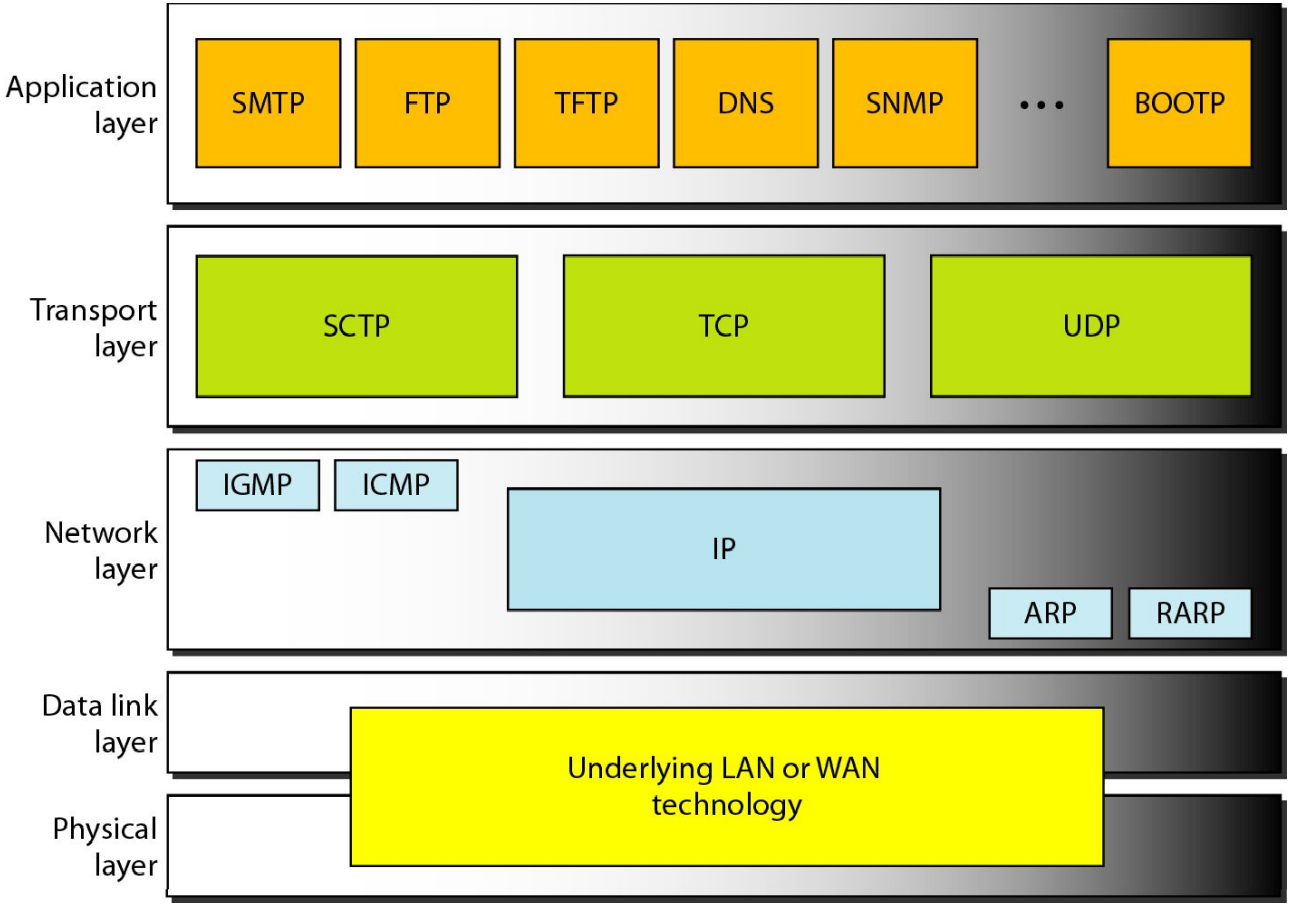
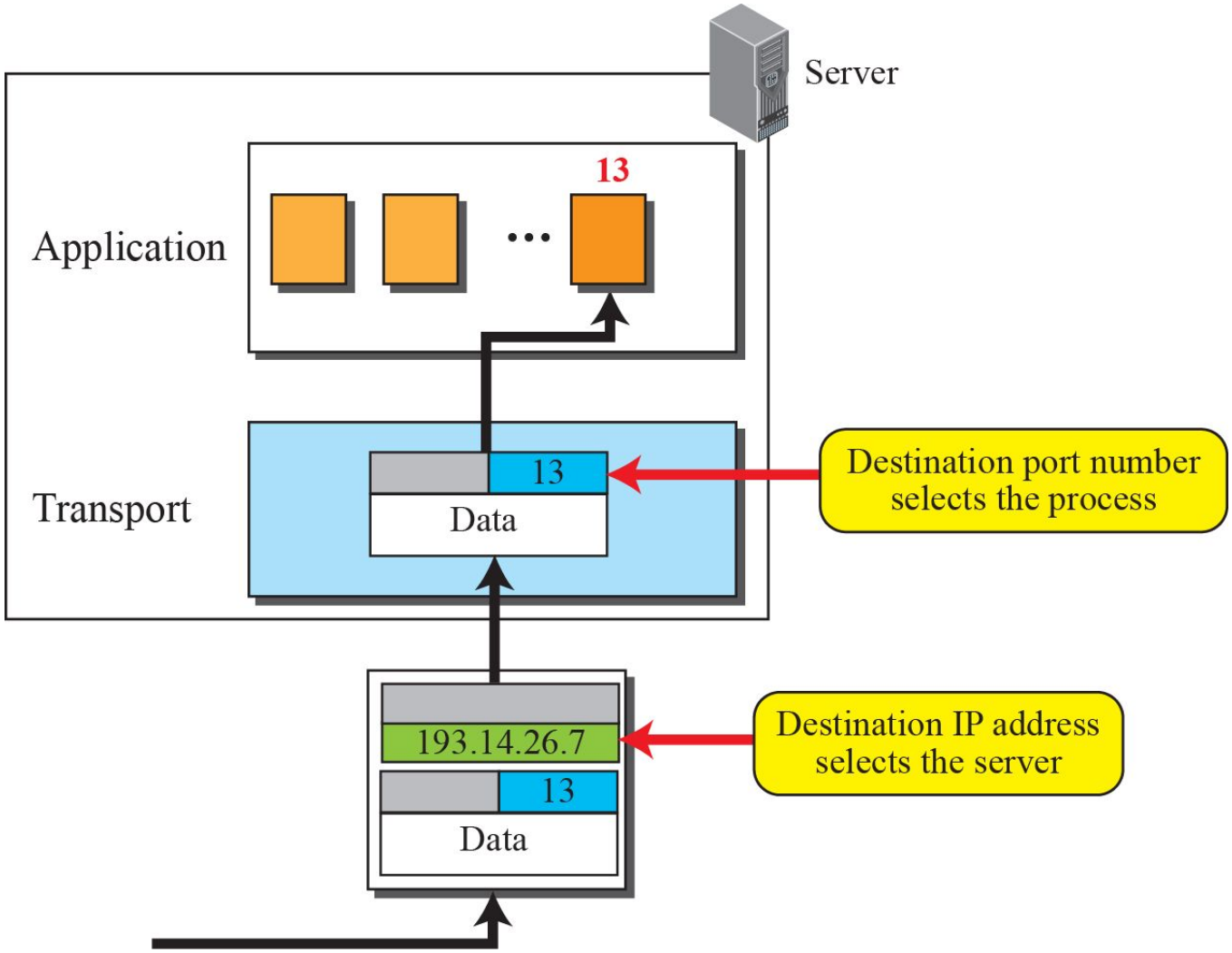


Figure 23.4: IP addresses versus port numbers



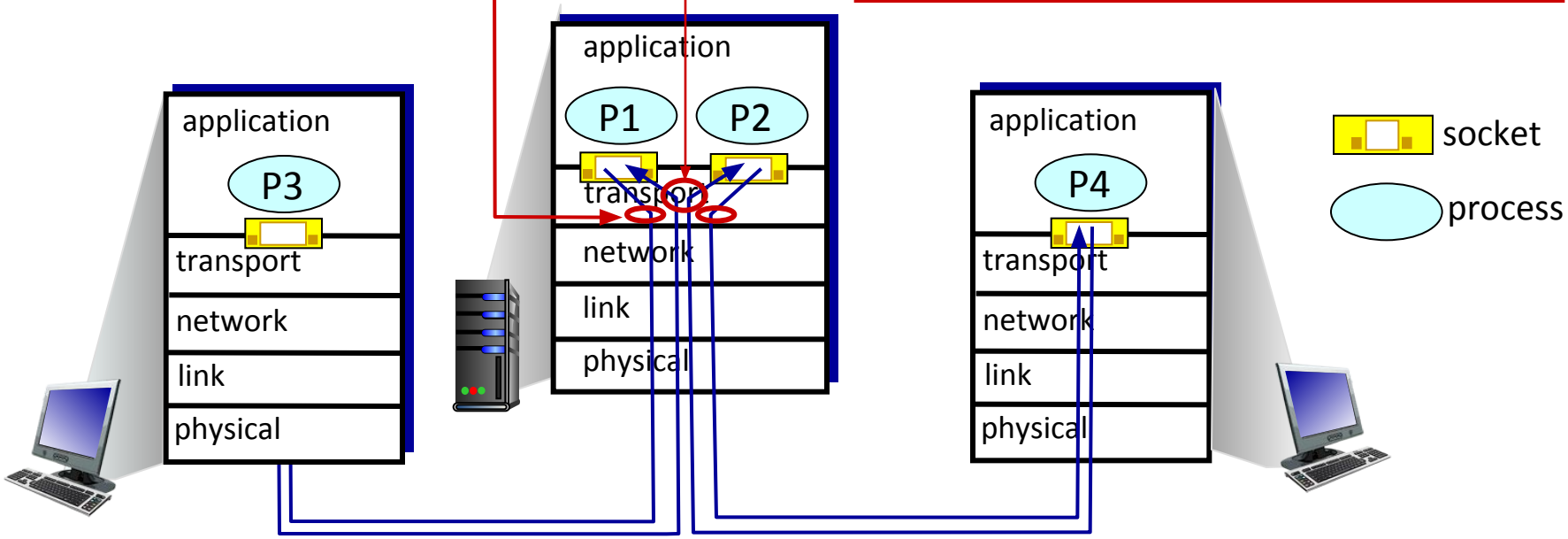
Multiplexing / Demultiplexing

Multiplexing at sender:

To handle data from multiple sockets, add transport header (later used for demultiplexing)

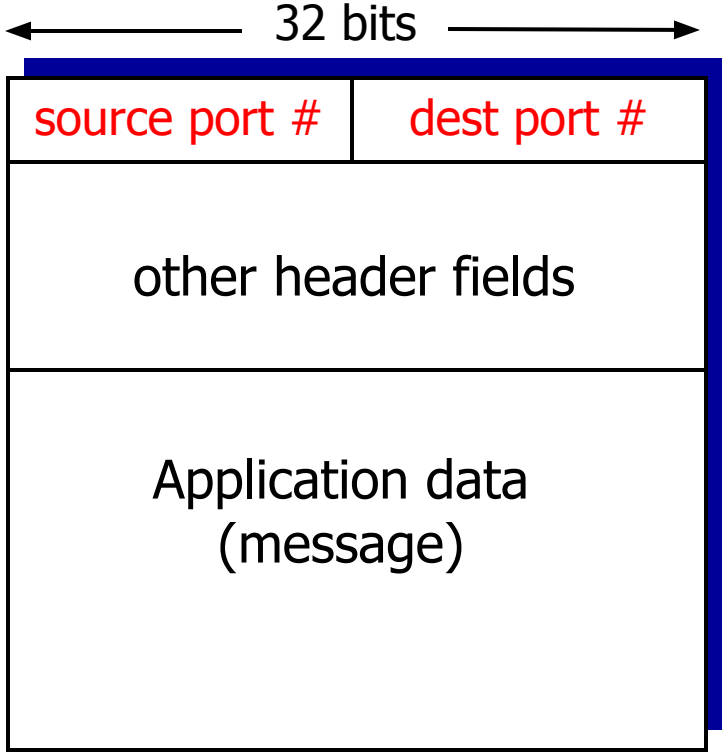
Demultiplexing at receiver:

Use header information to deliver received segments to correct socket



How demulptiplexing works?

- A host PC receives IP datagrams.
- Each datagram has source IP address and destination IP address.
- Each datagram carries one transport-layer segment.
- Each segment has source and destination port number.
- A host PC uses IP addresses & port numbers to direct segment to appropriate socket.



TCP/UDP segment format

Figure 23.8: Multiplexing and demultiplexing

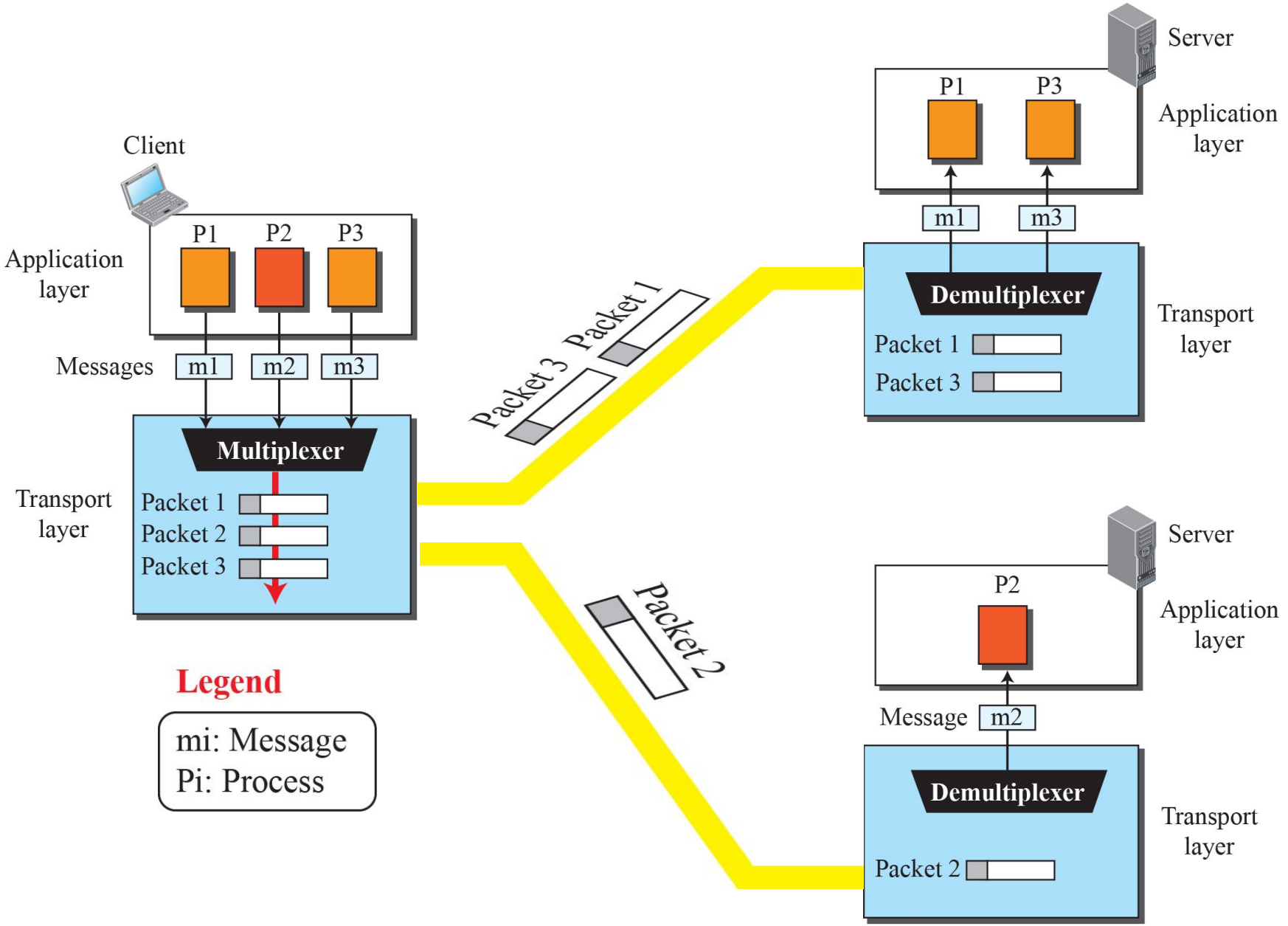


Figure 3.14: ***Connectionless service***

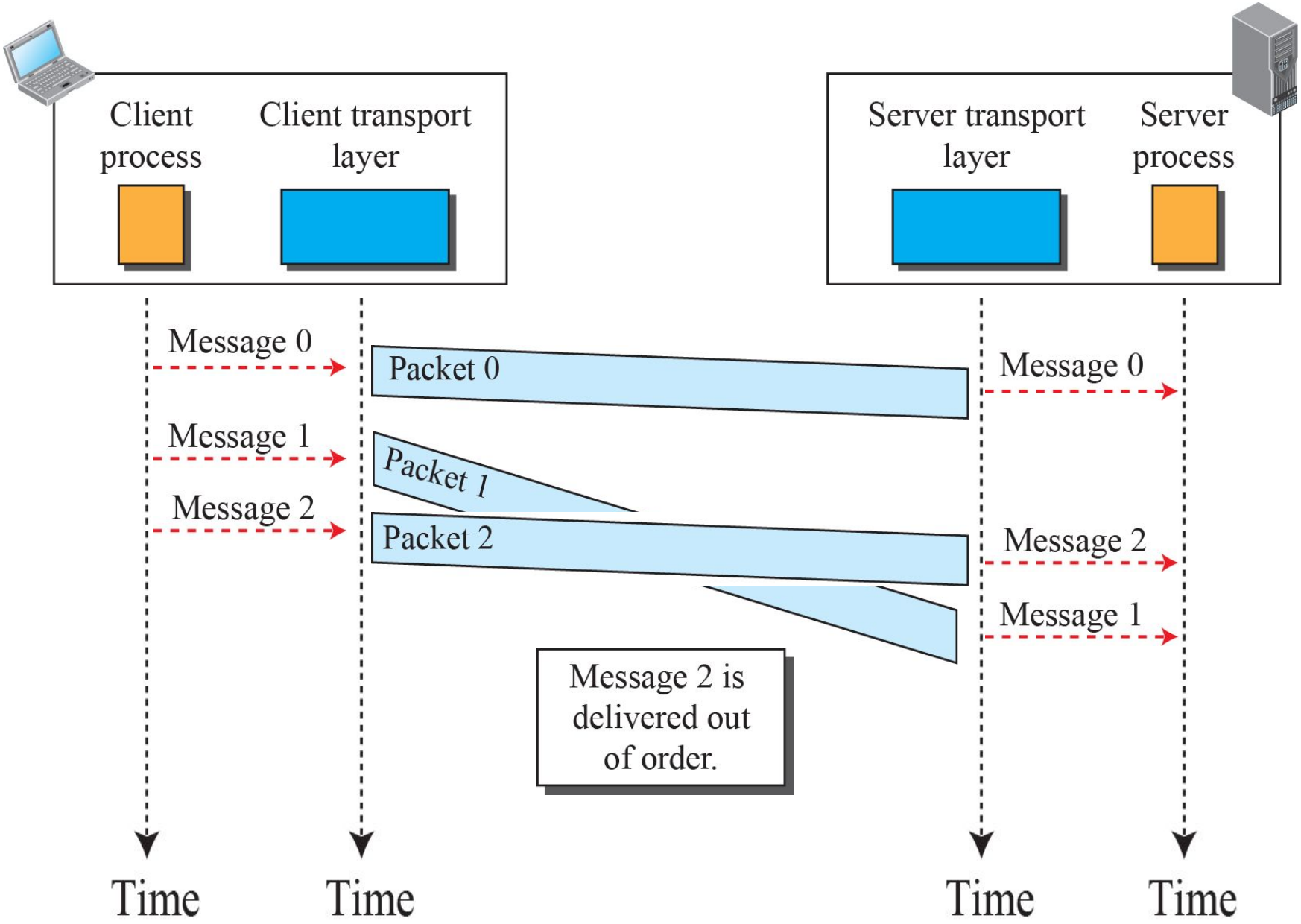
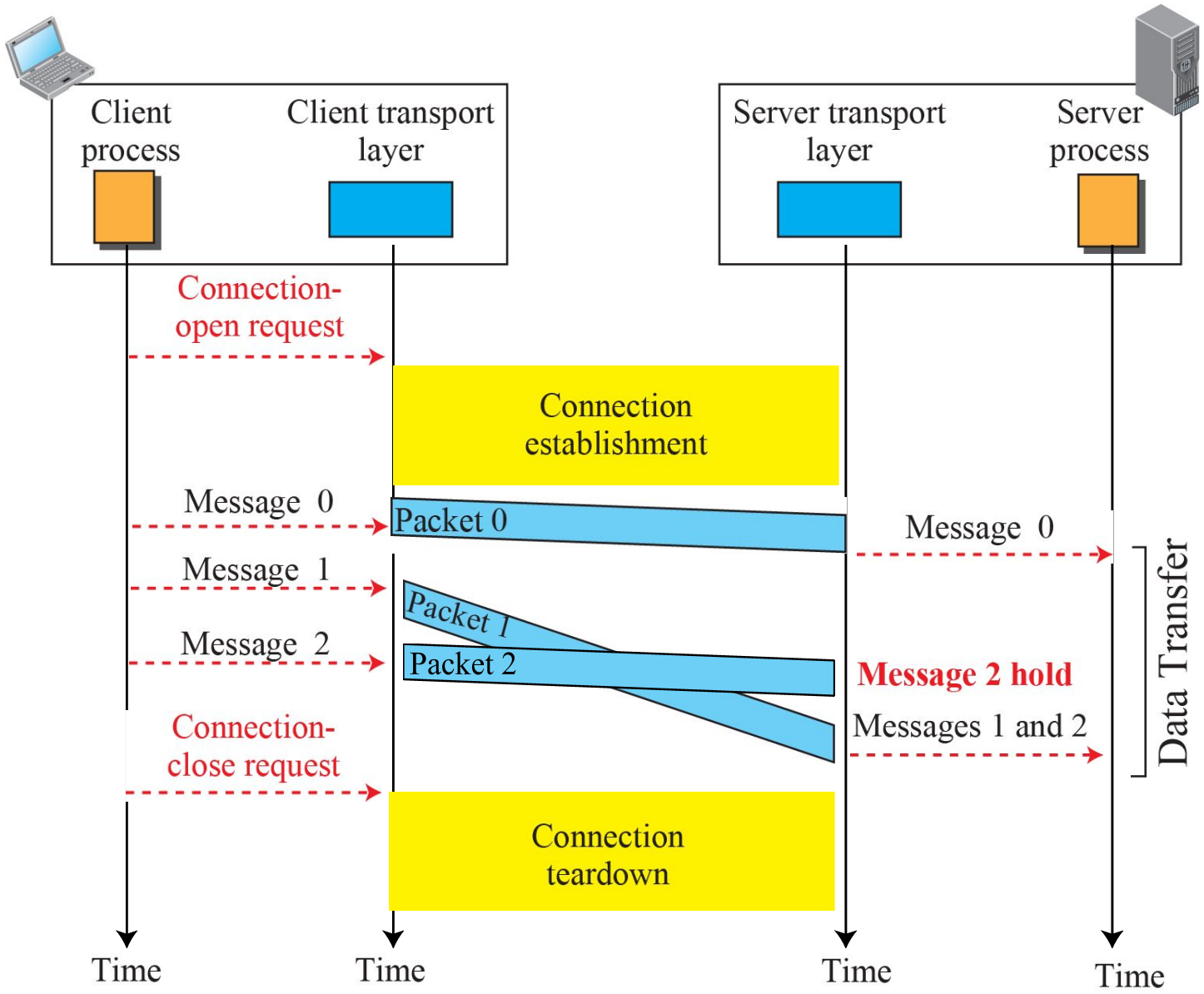


Figure 23.15: Connection-oriented service



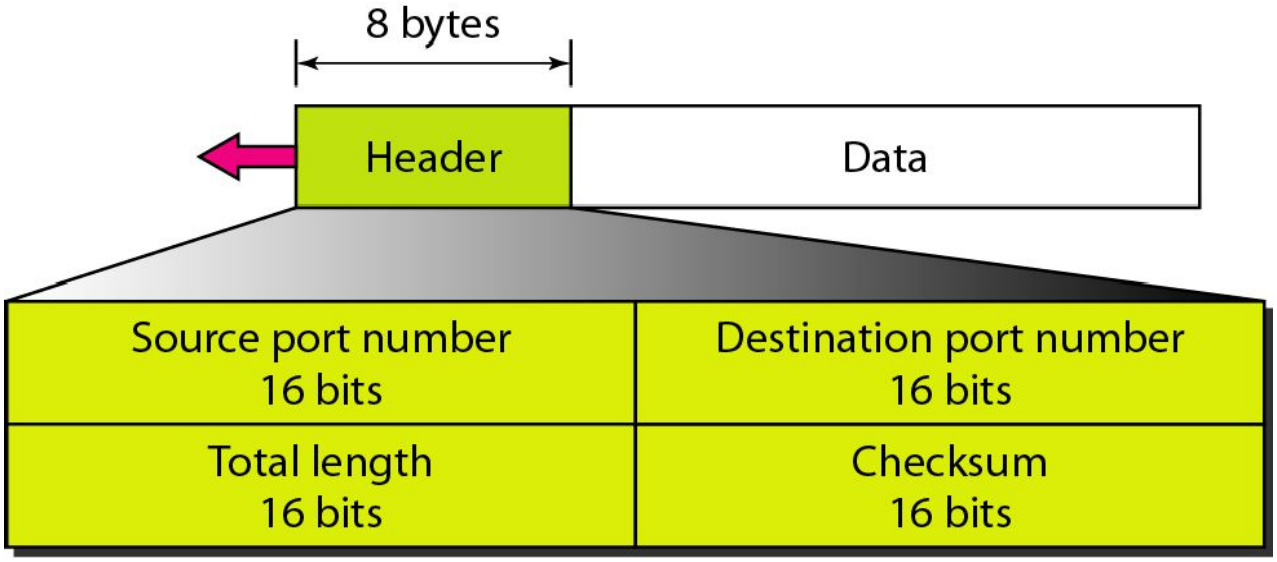
23-2USER DATAGRAM PROTOCOL

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.

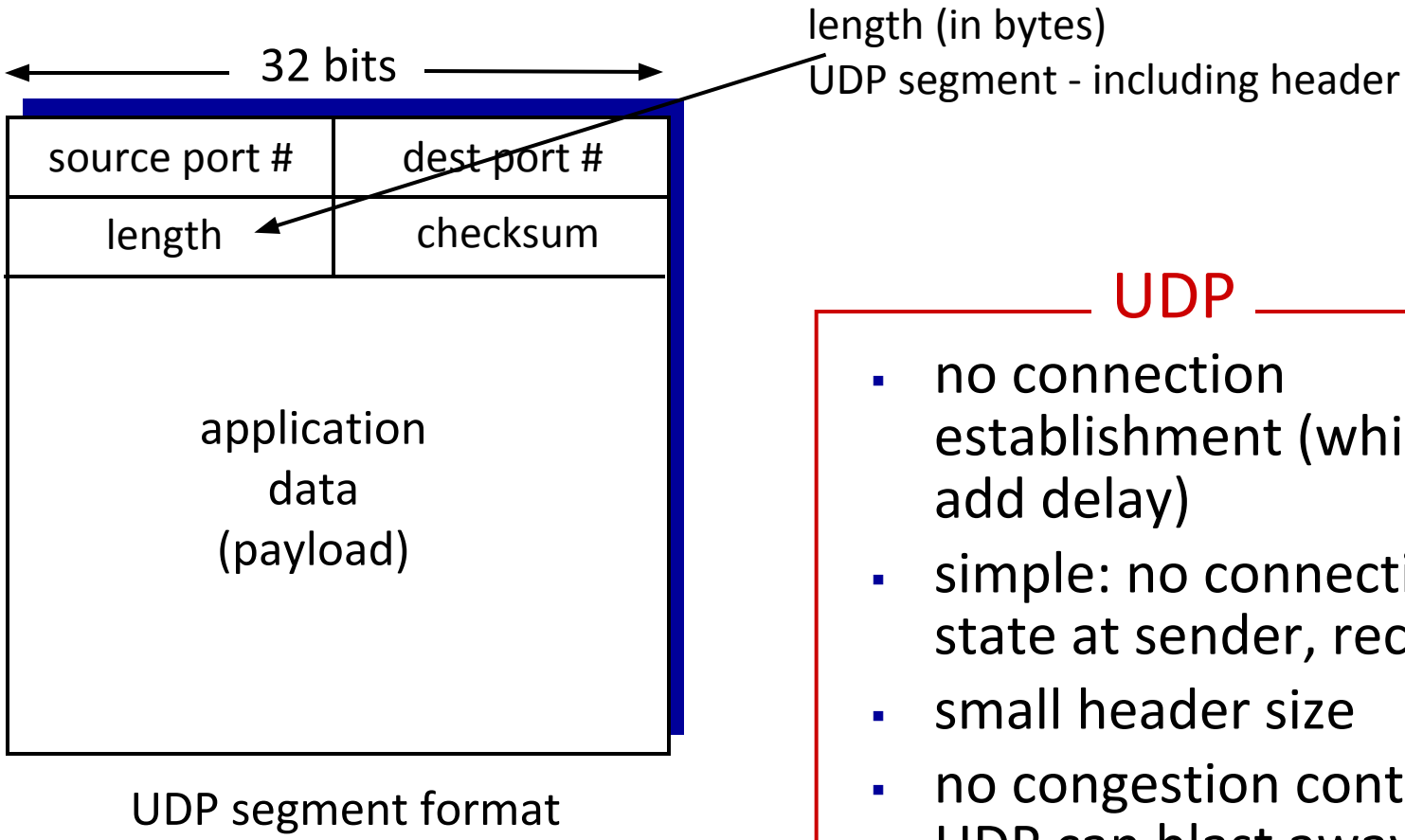
Table 23.1 *Well-known ports used with*

Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

Figure 23.9 *User datagram format*



UDP Segment - Header



UDP segment format

UDP

- no connection establishment (which can add delay)
- simple: no connection state at sender, receiver
- small header size
- no congestion control: UDP can blast away as fast as desired

Characteristics of UDP

- UDP provides an unreliable connectionless delivery service using IP to transport messages between two processes
- UDP messages can be lost, duplicated, delayed and can be delivered out of order
- UDP is a thin protocol, which does not add significantly to the functionality of IP
- It cannot provide reliable stream transport service

TCP Transmission Control Protocol

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPs to send data. In addition, TCP uses flow and error control mechanisms at the transport level.

- TCP transmits data in full-duplex mode.
- When two TCPs in two machines are connected, they are able to send segments to each other simultaneously.
- This implies that each party must initialize communication and get approval from the other party before any data are transferred.

Table 23.2 *Well-known ports used by TCP*

Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
20	FTP, Data	File Transfer Protocol (data connection)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	Bootstrap Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

Figure 23.13 *Stream delivery*

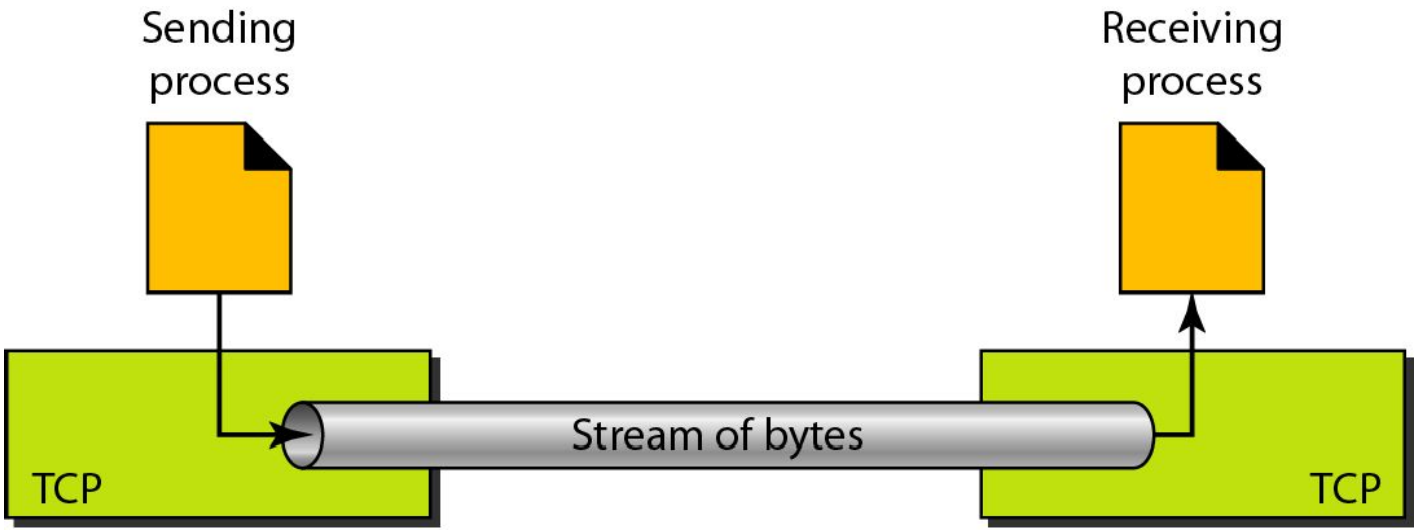


Figure 23.16 *TCP segment format*

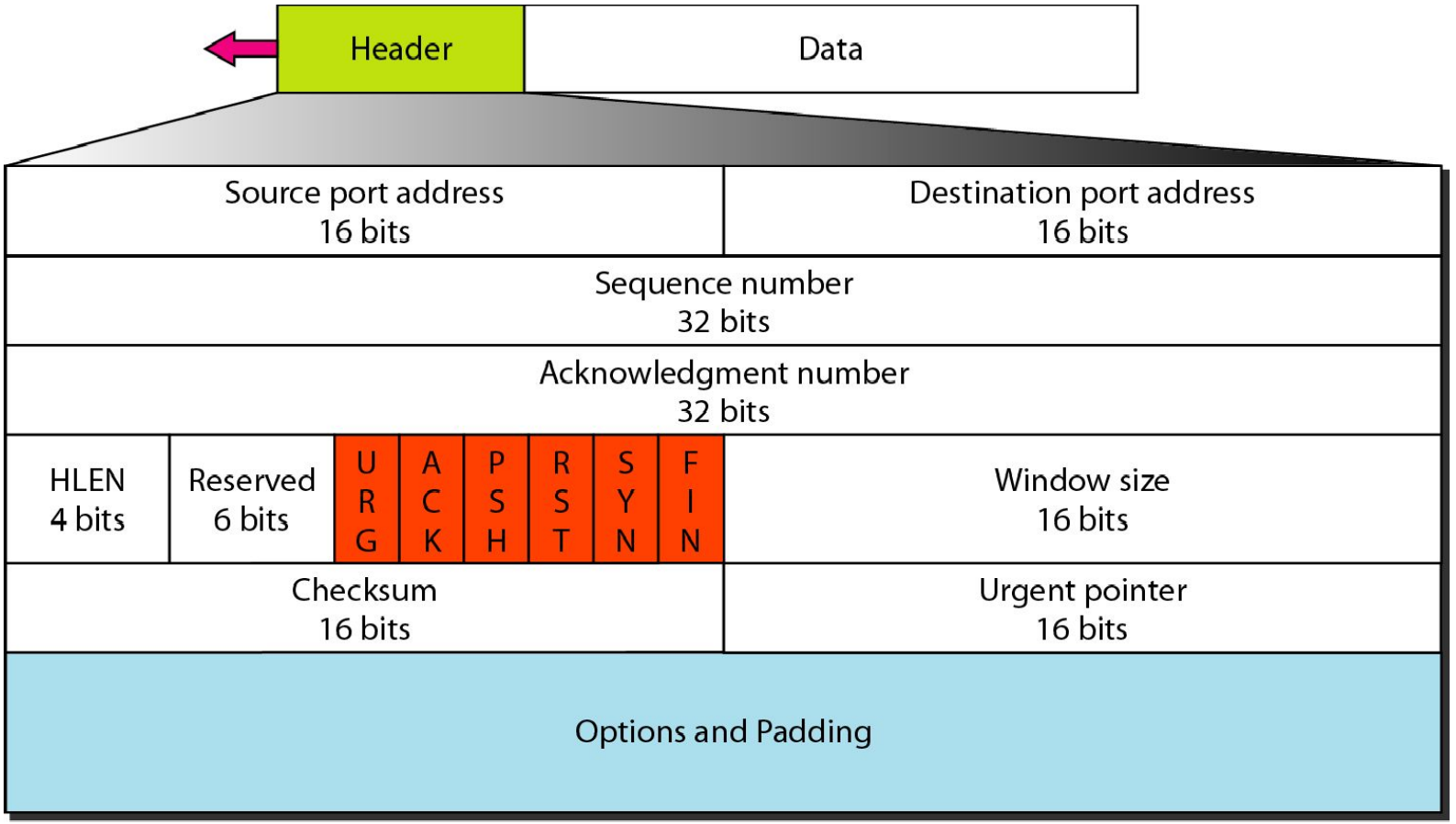


Figure 23.17 *Control field*

URG: Urgent pointer is valid
ACK: Acknowledgment is valid
PSH: Request for push
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: Terminate the connection

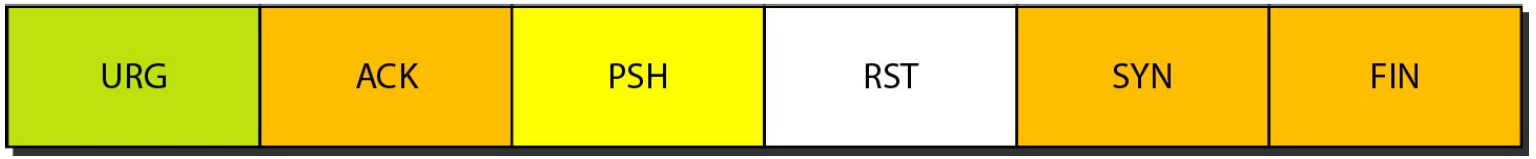
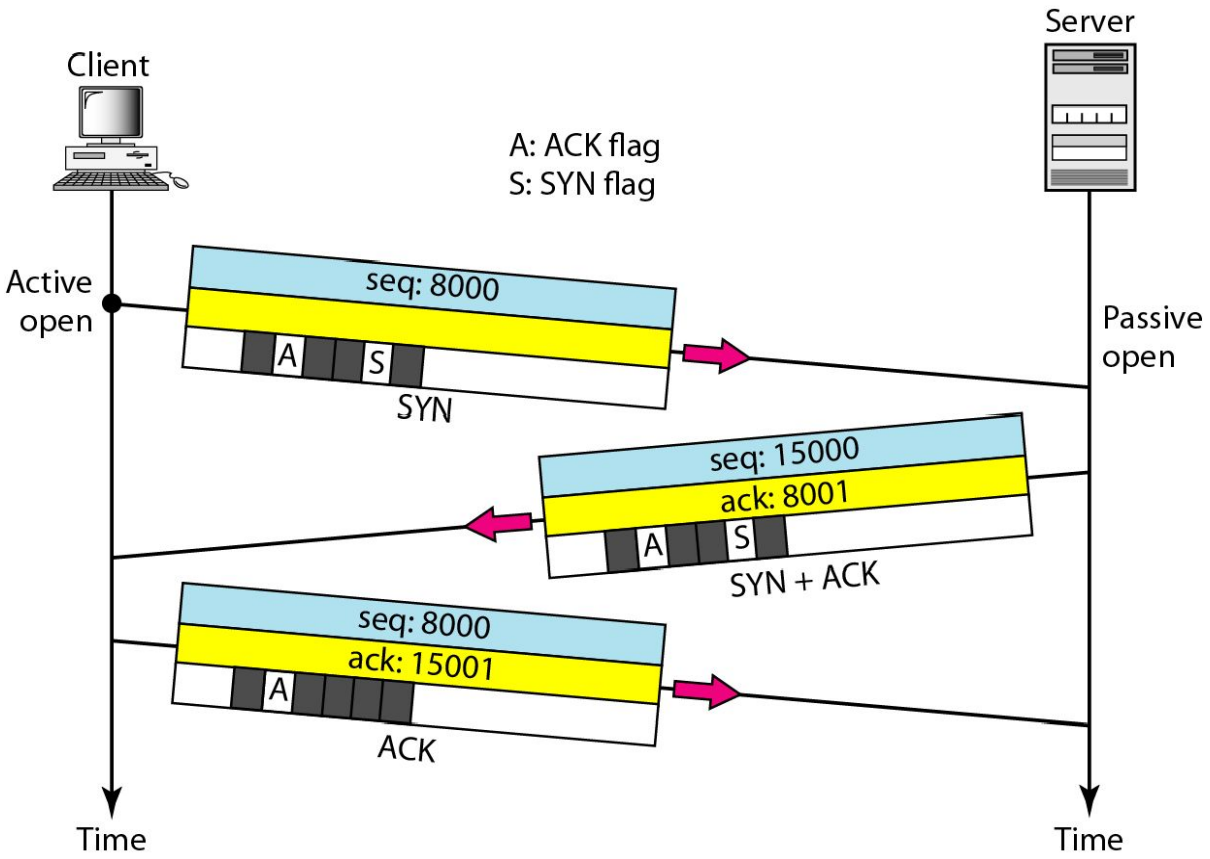


Table 23.3 ***Description of flags in the control***

Flag	*Description*
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledgment field is valid.
PSH	Push the data.
RST	Reset the connection.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

Figure 23.18 *Connection establishment using three-way handshaking*



A SYN segment cannot carry data, but it consumes one sequence number.

A SYN + ACK segment cannot carry data, but does consume one sequence number.

An ACK segment, if carrying no data, consumes no sequence number.

Figure 23.19 *Data transfer*

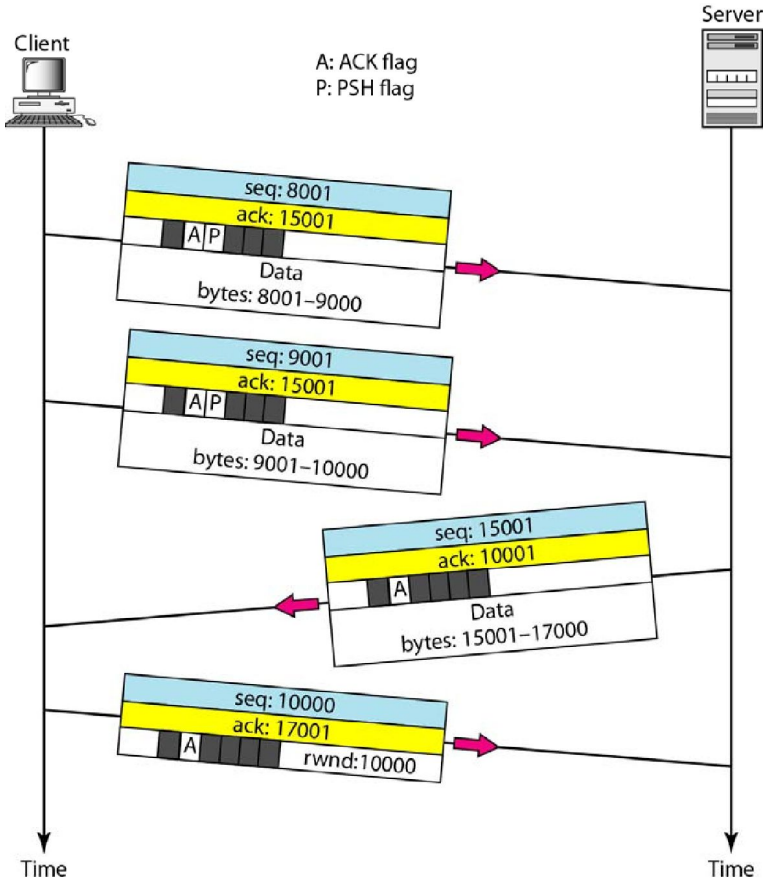
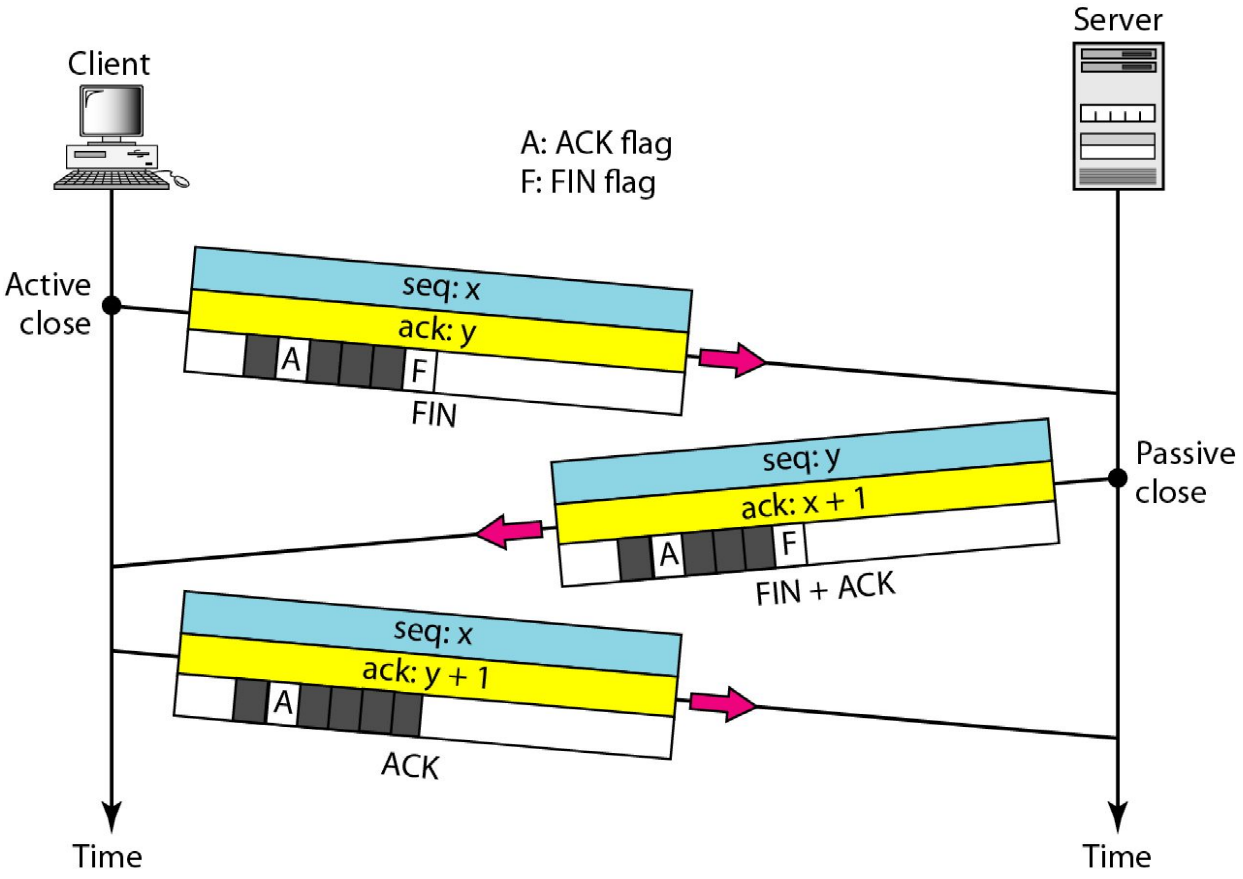
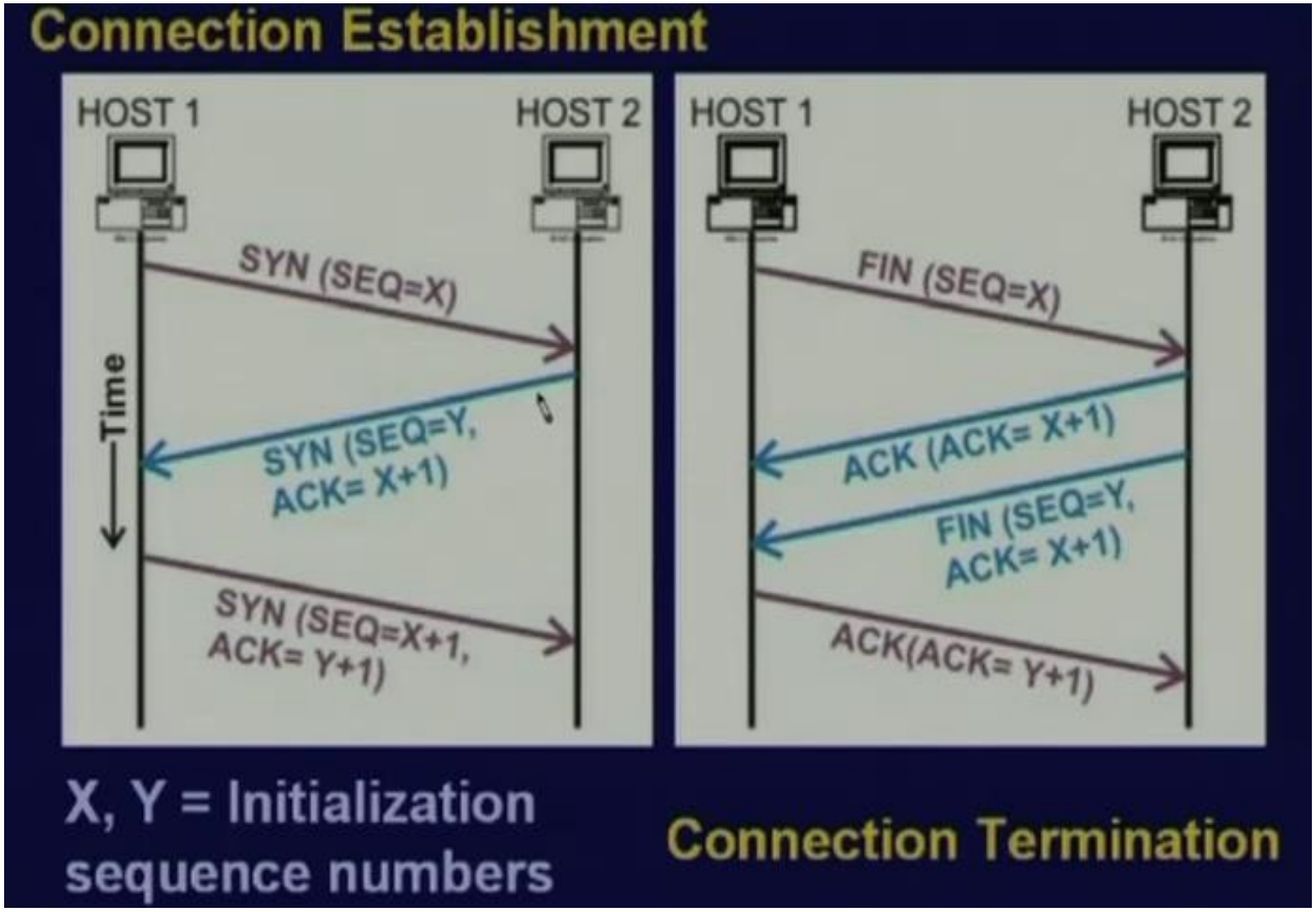


Figure 23.20 *Connection termination using three-way handshaking*



Connection establishment using three-way handshaking



TCP	UDP
1. TCP = Transmission Control Protocol	- UDP = User Datagram Protocol
2. Connection oriented protocol	- Connectionless protocol
3. TCP operation consists of three phases (a) connection establishment (b) data transmission (c) connection release	- UDP operation consists of single phase
4. Uses 3-way hand shake for connection establishment	- No connection establishment
5. Uses sequence numbers for orderly transmission	- No use of sequence numbers
6. Uses ACK packets	- No use of ACK packets
7. Uses flow control	- No use of flow control
8. Also called "Elastic protocol" i.e. data rates are adjustable	- "Non-elastic protocol"
9. Provides loss recovery i.e. uses error control	- No provision of loss recovery
10. Provides high QoS	- No or less QoS
11. Mostly used for Real Time applications	- Used for Non RT application
12. Data rates are low	- High data rates
13. Uses Circuit switched n/w	- Uses Packet switched n/w
14. TCP is a reliable protocol	- Non-reliable

Thank You