University of Southern California

Viterbi School of Engineering

EE450

Computer Networks

Network Layer

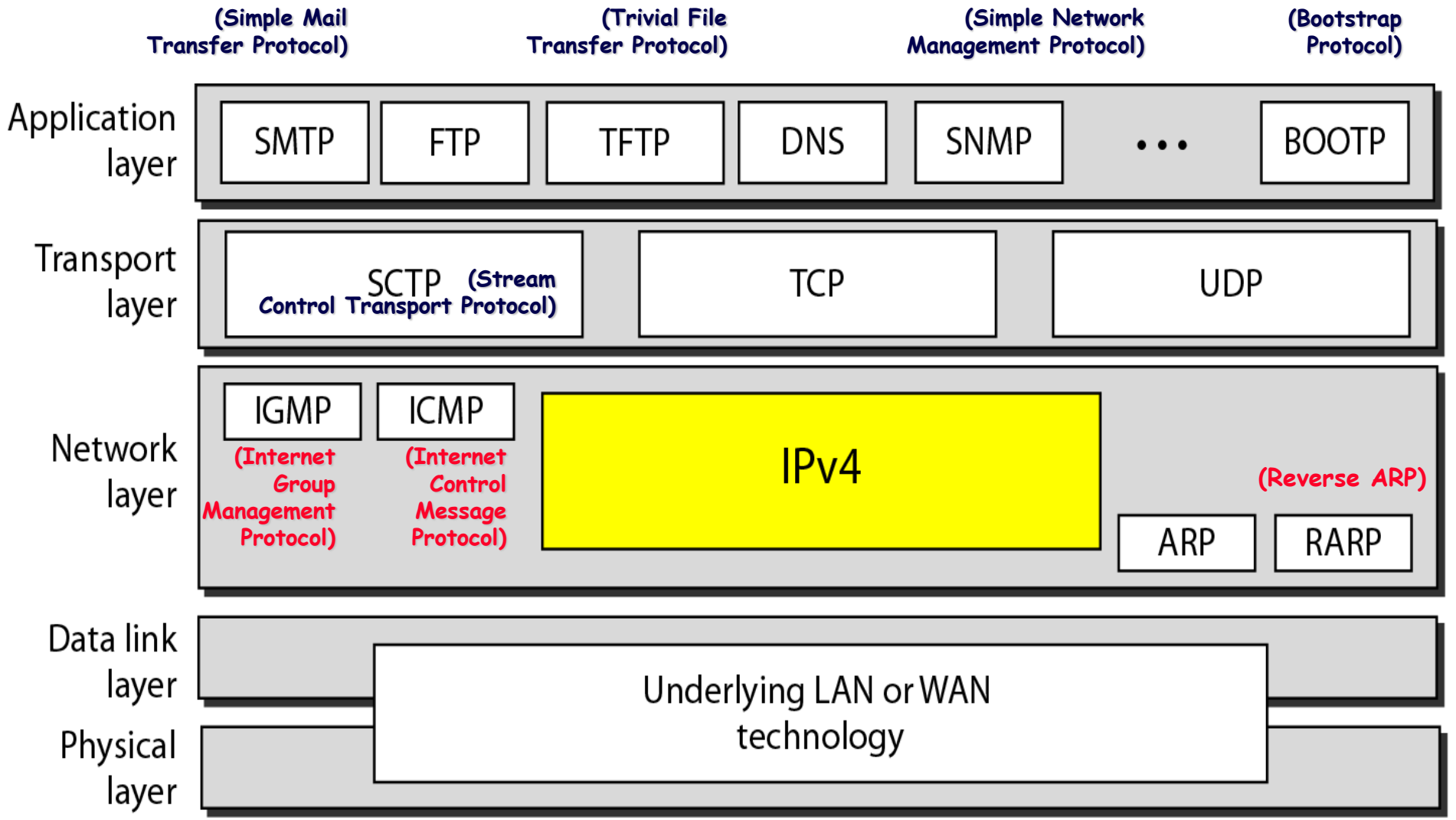
Shahin Nazarian

Spring 2013

Position of IPv4 in TCP/IP Protocol Suite

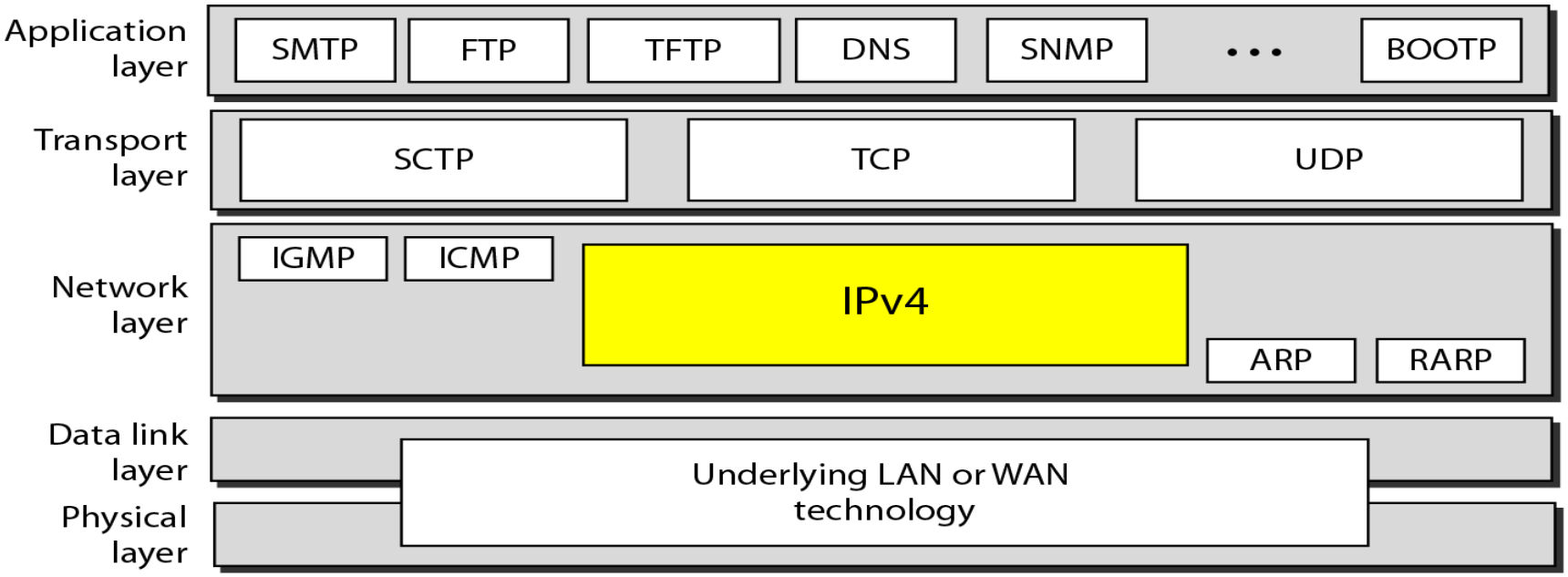
- The Internet Protocol version 4 (IPV4) is the delivery mechanism used by the TCP/IP protocols
- In TCP/IP, there are different MAC protocols and technologies, i.e., different Physical and Link layers
- One approach could have been to create a different IP for each of those technologies
- However in TCP/IP there is only one IP protocol; IP does not care about the bottom two layer technologies and protocols and can run over all of them, e.g., Ethernet, wireless, ATM, FR, T1, Cable, or any other MAC protocol or technology
- Layers 1 and 2 are hidden

TCP/IP Protocol Suite



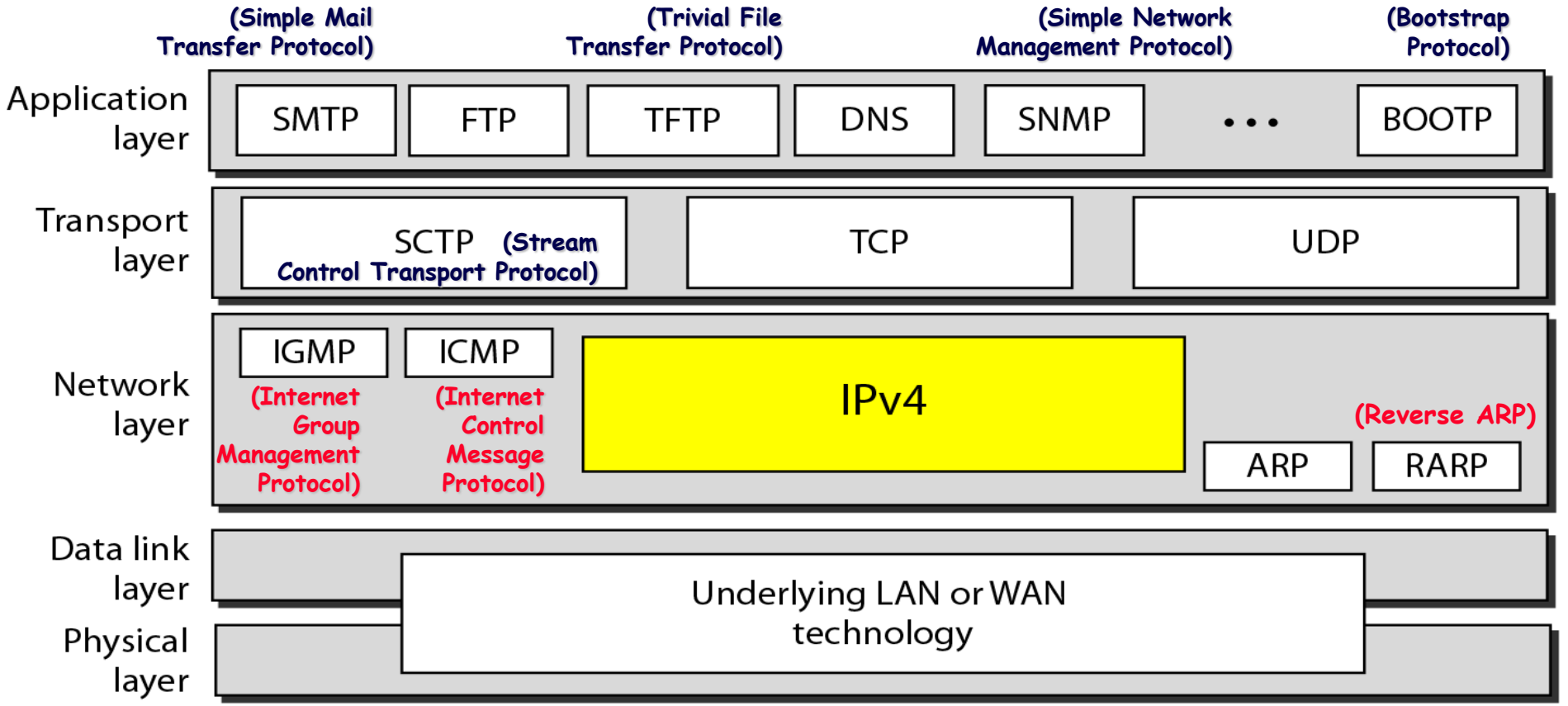
Position of IPv4 in TCP/IP Protocol Suite

- IP is a connectionless protocol, i.e., no connection is set up across the network before transmission
- IP is hence unreliable, in the sense that there is no guarantee for reliable delivery, error-free delivery, ordered delivery, or even delivery at all! It basically gives no guarantee (it's a best effort service)



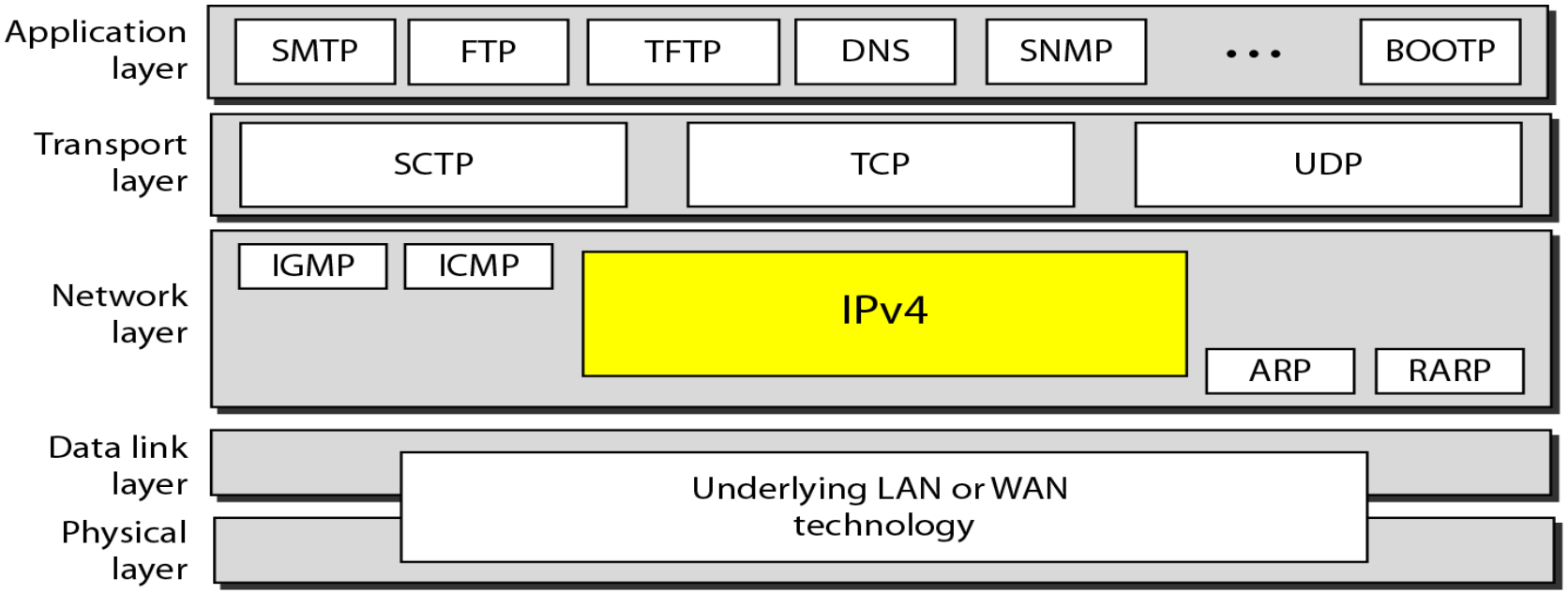
Position of IPv4 in TCP/IP Protocol Suite

- From the following diagram it can be seen that IP is the centerpiece of TCP/IP model



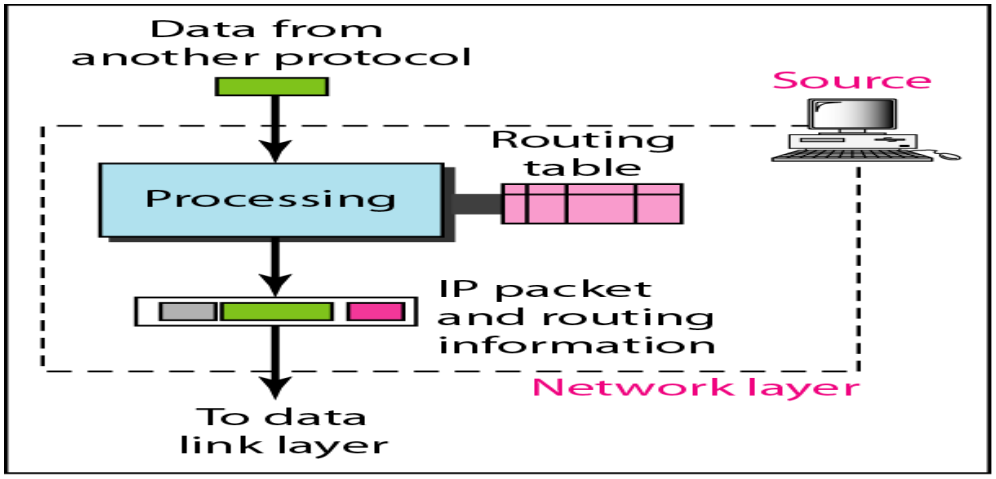
Position of IPv4 (Cont.)

- IP layer relies on the Transport layer (only TCP) to take care of these problems
- On top of the connectionless IP, we have connectionless or connection-oriented in Transport layer: UDP is connectionless, TCP is connection-oriented



Network Layer at Source

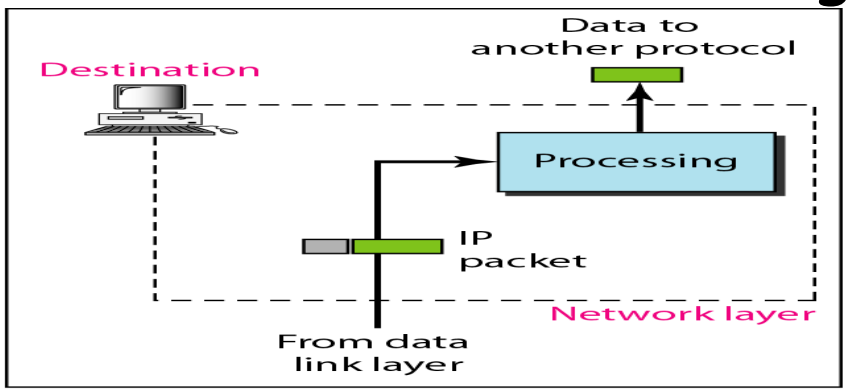
- Network layer at source is responsible for creating a packet from the data coming from another protocol, such as Transport layer protocol or a routing protocol
- It is also responsible for checking its routing table to find the routing information, such as the outgoing interface of the packet or the physical address of the next node
- Note that large packets are fragmented



a. Network layer at source

Network Layer at Destination

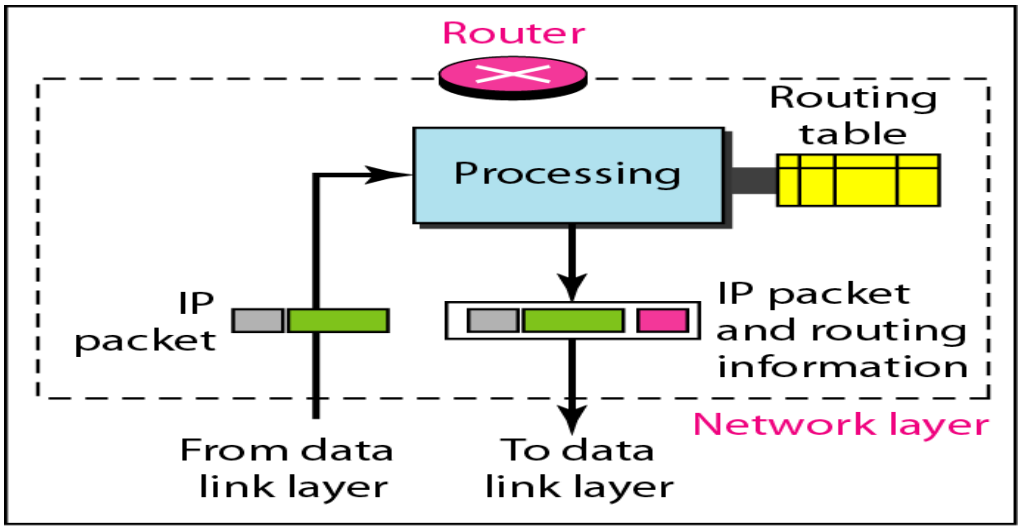
- Network layer at destination makes sure the destination address on the packet is the same as the host address. If the packet is a fragment, it waits until all fragments have arrived, and then reassembles them and delivers the reassembled packet to the Transport layer
- What is the main difference between source's and destination's Network layer? There is no routing table consultation at destination's. Both source and router consult with their routing table to find the next hop



b. Network layer at destination

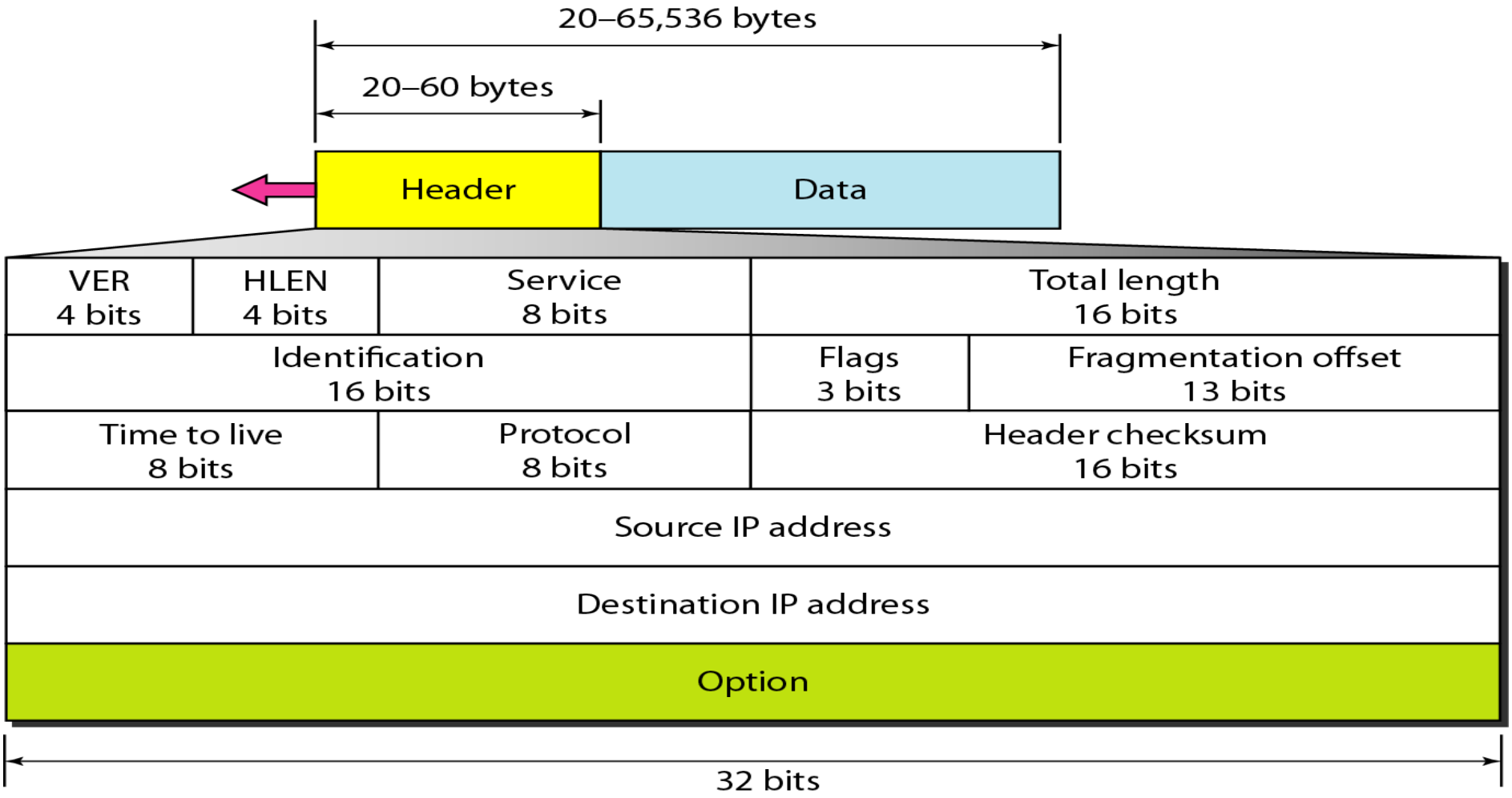
Network Layer at Routers

- The Network layer at a router or L3 switch is responsible for routing the packet; when the packet arrives at the router or L3 switch it consults its routing table and finds the interface from which the packet must be sent
- The packet after some changes in the header (routing information updated) is passed to the Data Link layer again



c. Network layer at a router

IPv4 – Packet format



IPv4 – Packet format

- Packet transmission is done serially, but we are sketching it vertically so that it fits the page!
- Packet is variable length.
- **VER**, represents the IP version: v4, v5, v6.
- **HLEN** represents the header length
- **Service** in best effort service is not used; some routers use Quality of Service, so network will treat packets differently, some get preferential treatment over others
- **Total length** is the total length of the packet including the header and the payload represented with 16bits
  - Maximum Total length is 65536

IPv4 – Service Type

- The "services" field, aka service type has 8 bits, 3 precedence bits and 4 TOS (type of services) bits and the last bit unused
- Precedence field defines the priority of the packet during network congestion, etc. For example if a router is congested and needs to drop a packet it starts with packets with lowest precedence
- TOS field has bit patterns with only one bit set as shown in the following table

TOS Bits	*Description*
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

IPv4 – Time to Live (TTL)

- **TTL** is to impose a limited lifetime in packet's travel through internet with the goal of preventing the packet to go into possible loops. Every time sender sends the packet it will set up TTL. Let's assume it is set to 8; when the packet reaches the first router, the 1st router will decrement it to 7, next to 6, next to 5, till it reaches 0 and if packet is not yet delivered, the router will drop the packet! This way, we are trying to prevent the packet going into possible loops
- If the TTL field reaches zero before the packet arrives at its destination, then the packet is discarded and an **ICMP** error packet (11 - Time Exceeded) is sent back to the sender
- Therefore a packet is dropped if its TTL is expired. Note: If sender sends a packet on its network, but does not want that packet to leave its network, it sets TTL to 1, because router will decrement it to 0 and then it gets dropped

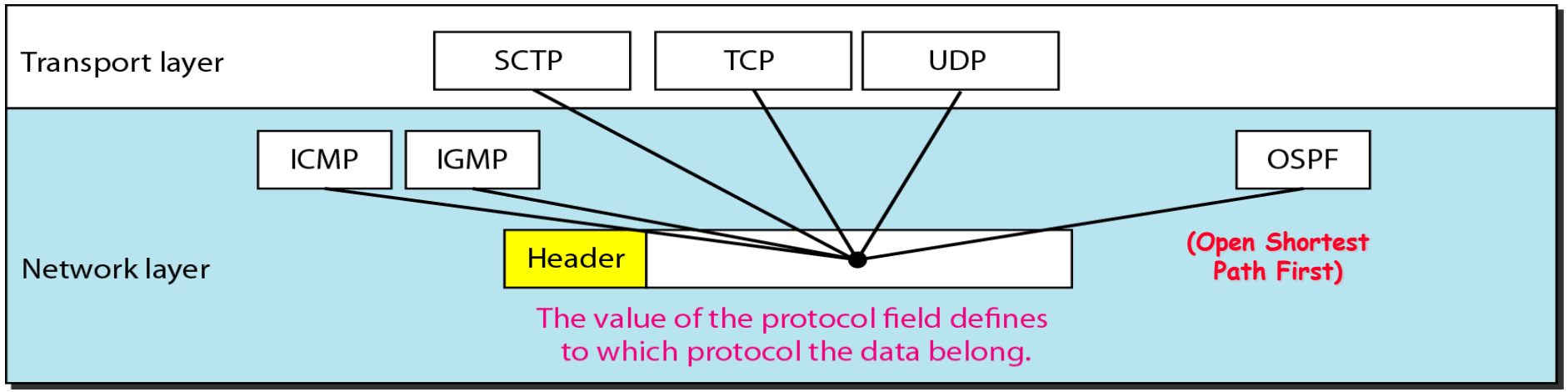
IPv4 – Protocol Bits

- Question: Eventually if the packet reaches destination, is the packet delivery a complete delivery?
- No, it is incomplete, because the ultimate destination is the application. Packet responsibility is delivery to the IP layer

Value	*Protocol*
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

IPv4 – Protocol Bits (Cont.)

- **Protocol bits** are to define the protocol that is using the service of IP. It defines the protocol that the packet should be delivered to at the final destination
- Routers inside the network do not care about the "Protocol bits" field. Router is not going to process what is inside the IP data, only destination does that



IPv4 – Checksum

- It is possible that header is error free and payload is erroneous. The packet will therefore be delivered to the right destination, but destination will drop it
- The checksum is used in the Internet by several protocols (but Data Link layer protocols are not among the ones using checksum)
- The main reasons why IPv4 checksum is not done on the whole packet: 1) data does not change from hop to hop, but header does, 2) all upper layer protocols (e.g., TCP and UDP) have checksum on the whole message, and 3) to reduce the processing delay of the routers

Checksum Basics

- Idea: Suppose our data is a list of five 4-bit numbers that we want to send to a destination. In addition to sending these numbers, we send the sum of the numbers
- E.g., if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers
- The receiver adds the five numbers and compares the result with the sum
- If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum
- Otherwise, there is an error somewhere and the data is not accepted

Checksum – Wrapping

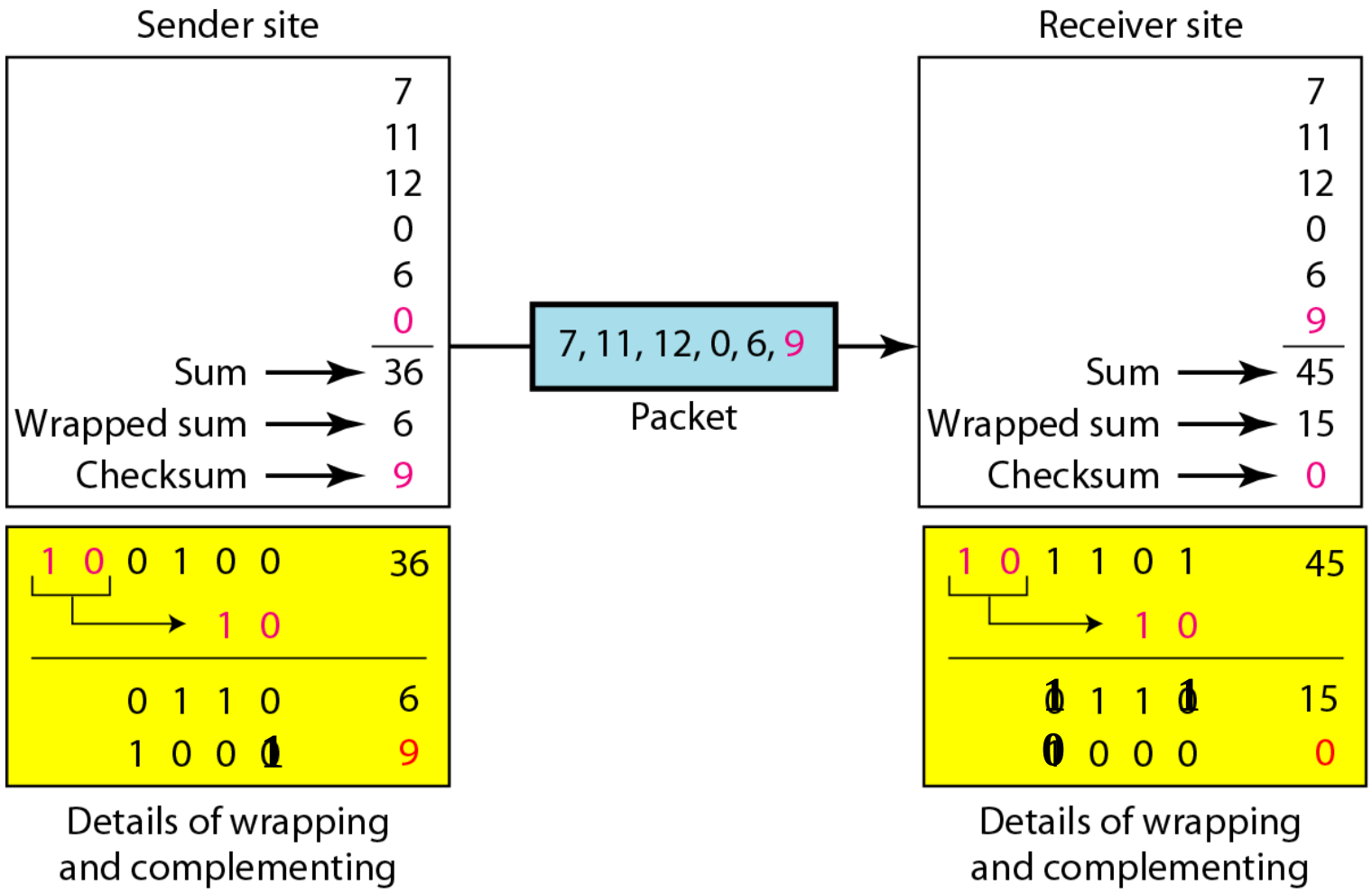
- A minor modification makes the job of the receiver easier: we will send the negative (complement) of the sum, also referred to as **Checksum**
- In previous example, we send (7, 11, 12, 0, 6, **-36**)
- The receiver can add all the numbers received (including the checksum). If the result is 0, it assumes no error; otherwise, there is an error
- One major drawback with this approach is that checksum needs more bits than the data
- Solutions: **Wrapping**: If the checksum has more than n bits, the extra leftmost bits need to be added to the n rightmost bits
  - Example: 21=10101 (needs five bits). Wrapping: (0101+ 1) = 0110 or **6** which is 4 bits

Checksum Using 1's Complement

- To represent 36=100100: Wrapping: 0110=6!
  - 6=0110
  - -6 (1's complement) = 1001
- 1001 in unsigned number system is 9, however in 1's complement system is -6

Checksum Using 1's Complement

- Now let's redo the example (7, 11, 12, 0, 6, -36)

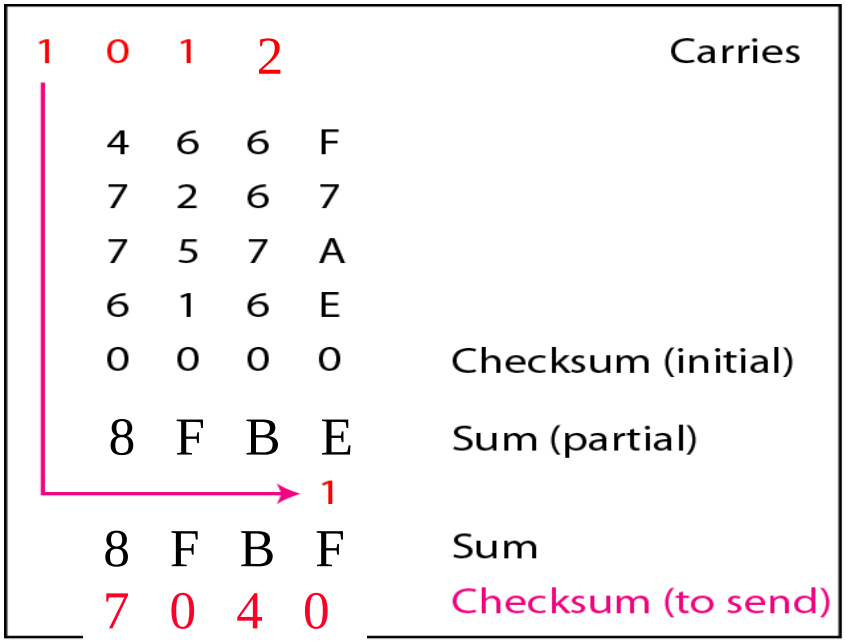


Internet Checksum

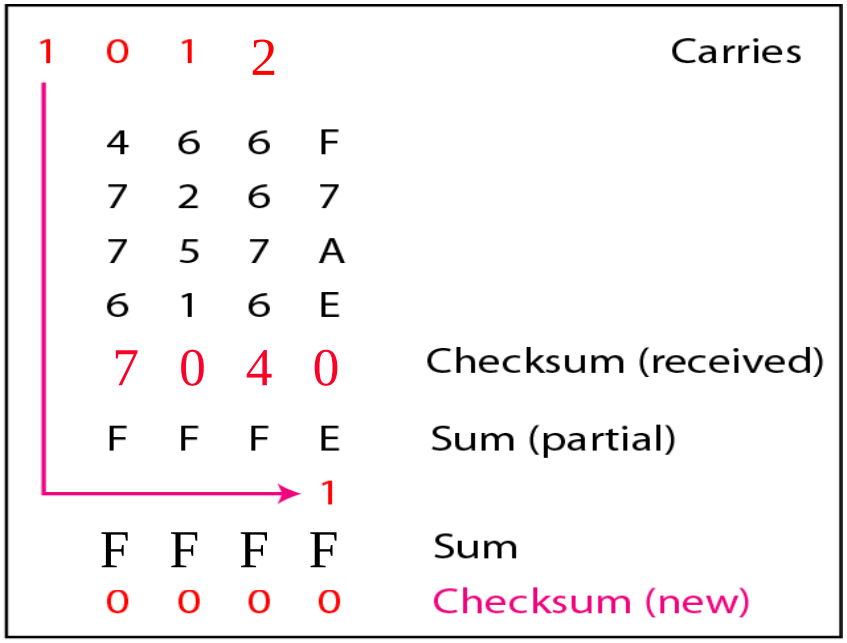
- **Sender site:**
  1. The message is divided into 16-bit words
  2. The value of the checksum word is set to 0
  3. All words including the checksum are added using 1's complement addition
  4. The sum is complemented and becomes the checksum
  5. The checksum is sent with the data

- **Receiver site:**
  1. The message (including checksum) is divided into 16-bit words
  2. All words are added using 1's complement addition
  3. The sum is complemented and becomes the new checksum
  4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected

Internet Checksum – Example



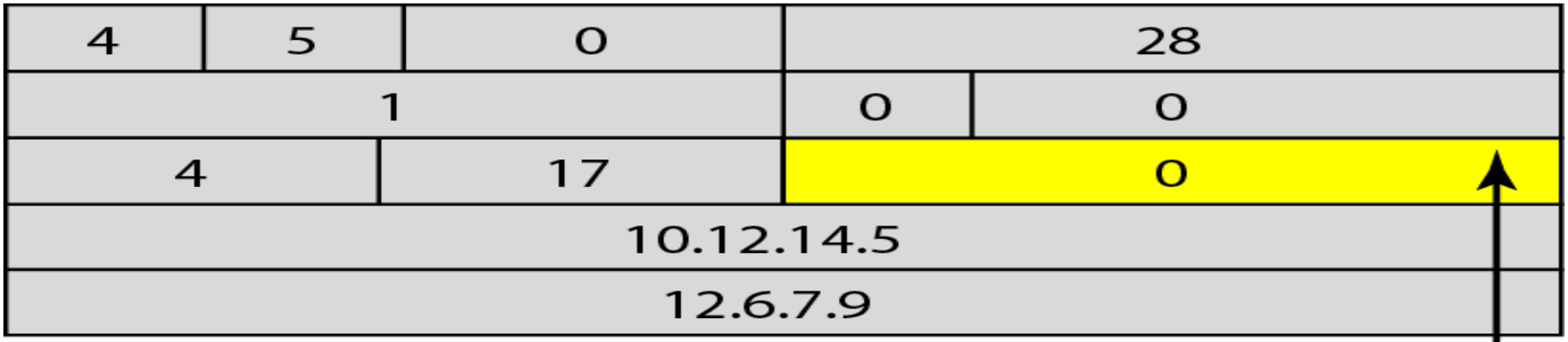
Checksum at the sender site



Checksum at the receiver site

Note: The header has the destination IP address, so it is important to check the header and make sure it is error free to avoid delivering it to wrong destination. Router will drop the packet if an error in the header is found. Check next slide for an IPv4 checksum example

Internet Checksum Example – IPv4 Header



4, 5, and 0	→	4	5	0	0
28	→	0	0	1	C
1	→	0	0	0	1
0 and 0	→	0	0	0	0
4 and 17	→	0	4	1	1
0	→	0	0	0	0
10.12	→	0	A	0	C
14.5	→	0	E	0	5
12.6	→	0	C	0	6
7.9	→	0	7	0	9
Sum	→	7	4	4	E
Checksum	→	8	B	B	1

IPv4 – Fragmentation

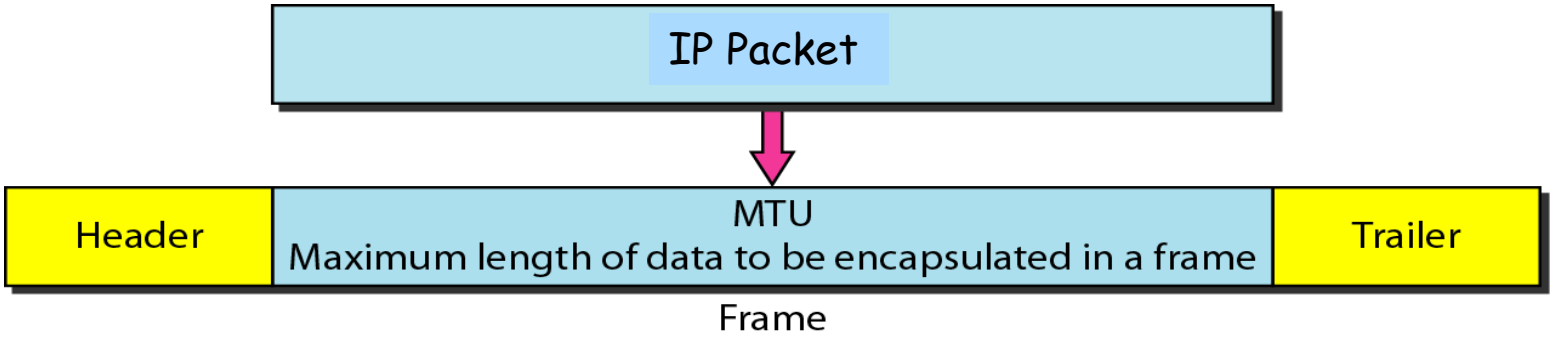
- IP packet needs to be fragmented into smaller pieces, e.g., the payload of the Ethernet frame can be 1500 bytes maximum
- IP packet goes through different networks and technologies, each time encapsulated into a frame. Larger packets need to be fragmented into shorter pieces to fit the frame size requirements
- The format and size of the received frame depend on the protocol used by the physical network (Data Link layer protocol) through which the frame has just travelled
- 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

IPv4 – Fragmentation (Cont.)

- If the router is connected to a network that needs smaller packets, then the router will fragment the packet
- A fragment may still get fragmented
- The fragments are reassembled by the destination, not the routers, first of all because there is no guarantee for all fragments to follow the same route, and no guarantee that they will be delivered in the same order. Secondly, even if a router had received all the fragments in order, still it would not be wise to dedicate router's time to reassemble fragments. It is better to let the end system (destination) do that
- Again note that frame size and format varies depending on the Data Link layer protocol used by the physical network through which the frame travels
- **MTU (Maximum Transmission Unit)** is the maximum size of **the frame payload**

Fragmentation – MTU

- If packet size > MTU, the packet should be fragmented
- The network is going to treat each fragment as an independent packet, therefore the header is not fragmented



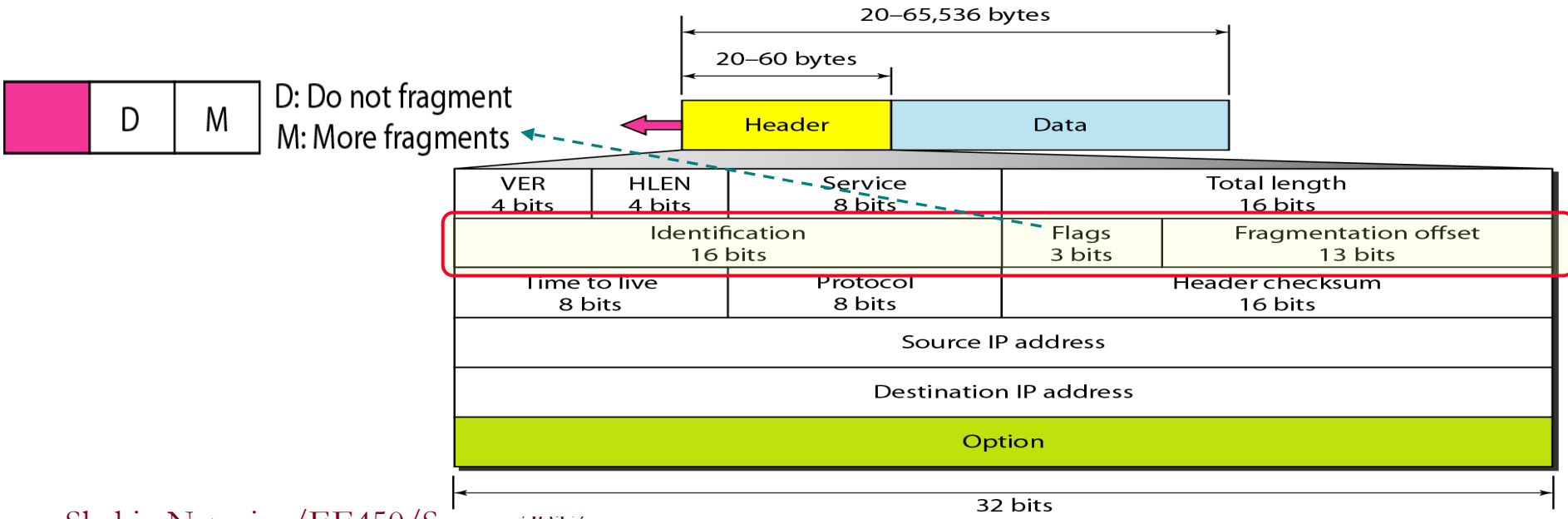
Protocol	*MTU*
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

IPv4 – Fragmentation (Cont.)

- However headers are not going to be identical: e.g., the "Total length" changes, and the "Header checksum" changes
- Besides those, when the packet goes from one router to another router, TTL changes; so even if the address parts do not change, some other parts of the header will change
- Fragments do not need to be the same length, like the packets that do not need to be the same length
- If the fragments are short, we need more headers, so throughput will go down, however the delay encountered in the routers is short. So short fragment is good for voice application
- However for data, longer is better; to have higher throughput

Fragmentation – Identification Bits & Flags

- Fragments belong to the same packet, but network treats them as if they were independent packets
- All fragments of a packet have the same **identification number** which is the same as the packet's identification number
- There are 3 flags, one of them is not used (is for future use)
- If the source does not want the router to fragment its packet, it sets **D**=1



IPv4 – Fragmentation Offset

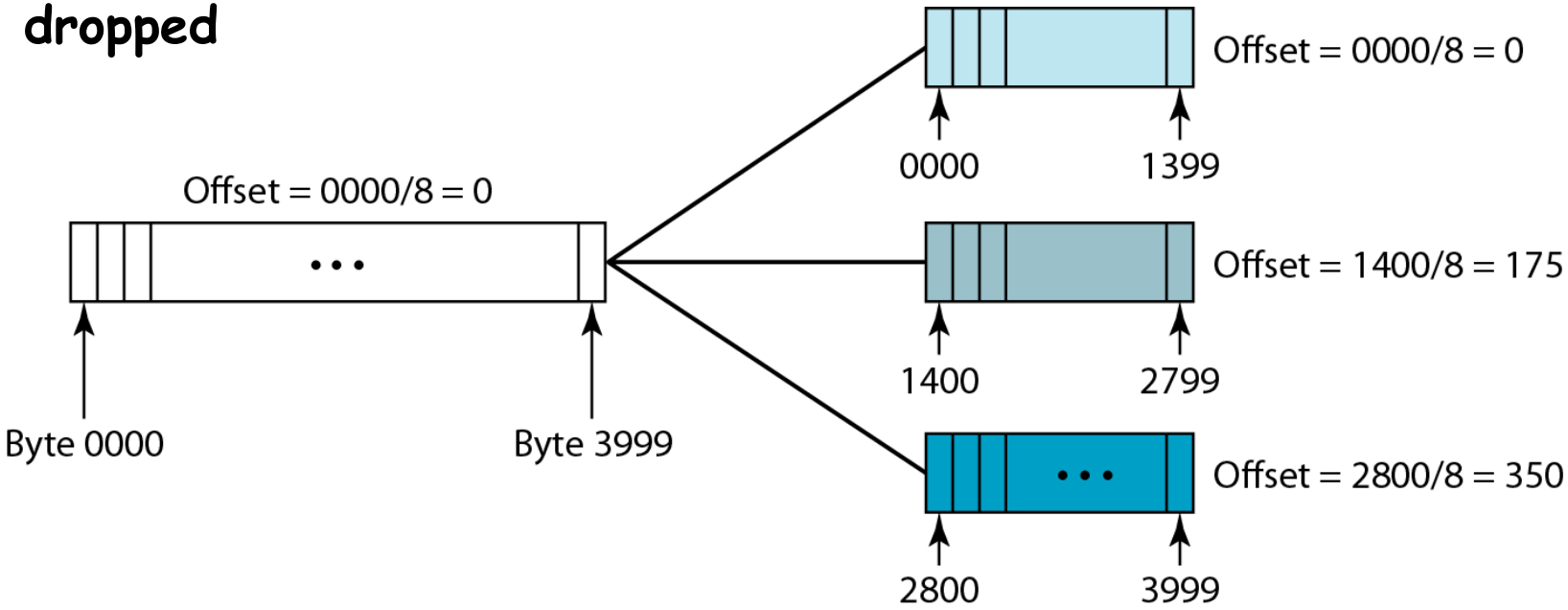
- Packet reaches a network that needs to fragment the packet, but it sees D as 1 so it drops the packet; however the router will send an ICMP message to the source, informing source that it could not support its packet without fragmentation
- Question: what are the reasons a packet is dropped? 1) Network congestion 2) TTL expiration 3) Error found in the header 4) D=1 and fragmentation is needed
- **M**=1 means the fragment is not the last fragment. M=0 means it is the last fragment of a packet
- **Fragmentation offset**: Relative position of the fragment to the whole packet measured in units of 8 octets (bytes)
  - Fragmentation offset value is the 1st byte number divided by 8

IPv4 – Fragmentation (Cont.)

- Question: Why did we not simply number the fragments from 0 to n and use a "fragmentation sequence" instead of fragmentation offset?
  - The issues is that a fragment can be further fragmented, therefore sequencing is not going to work, but offset does
- If a node receives a fragment with M=0 before another with M=1 the offset of the one with M=0 won't be zero, because it is not the first fragment
- Units of 8 bytes means that the fragment sizes have to be multiples of 8 bytes, **except the last one**
- Question: Let's assume it were guaranteed that only one level of fragmentation should exist, i.e., a fragment could not be further fragmented. Would sequencing fragments work?
  - Yes, in that case, numbering fragments based on a sequence would work

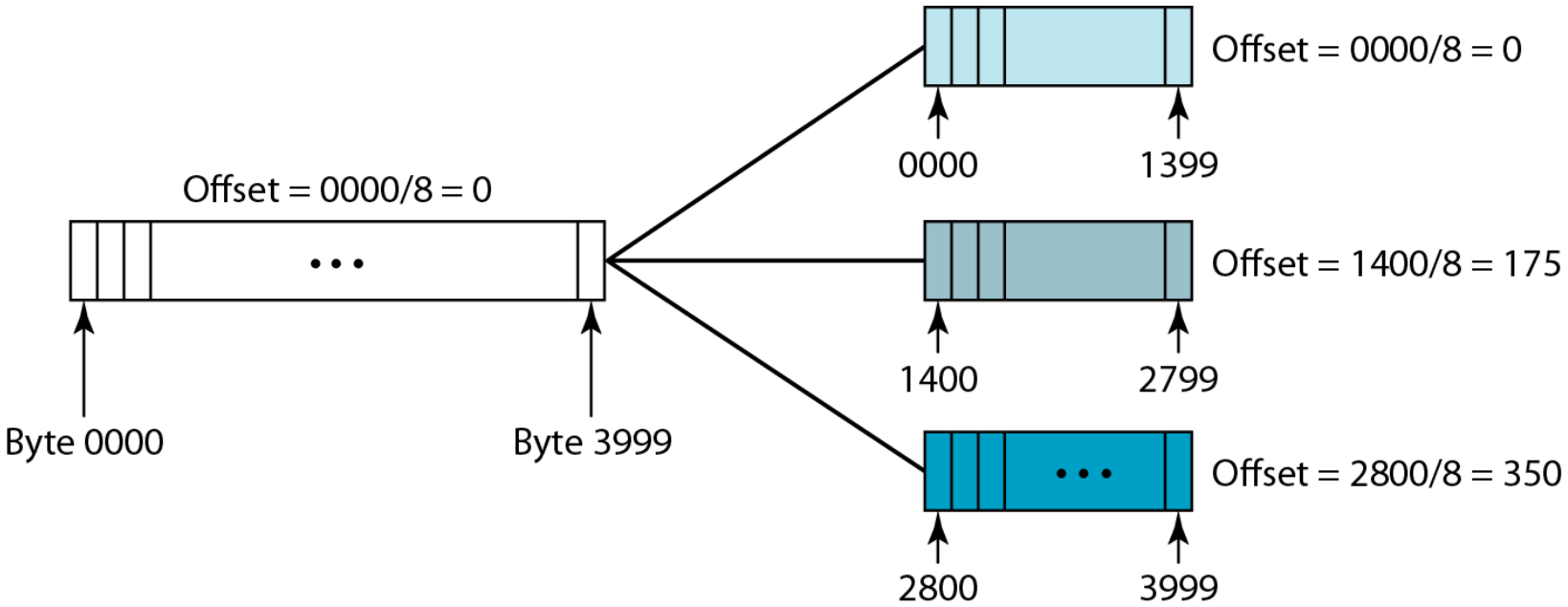
Fragmentation Example I

- Why offset is shown in multiples of 8? Because with 13 bits the max value is 8192 which is not enough to for large packets
- The receiver will buffer the last fragment if it reaches earlier than the first ones, but there is always a time limit for buffering, because it's possible that those fragments have gotten dropped on the way
- Note that if one fragment gets dropped, all other fragments (from the same packet) that got received by the destination will get dropped

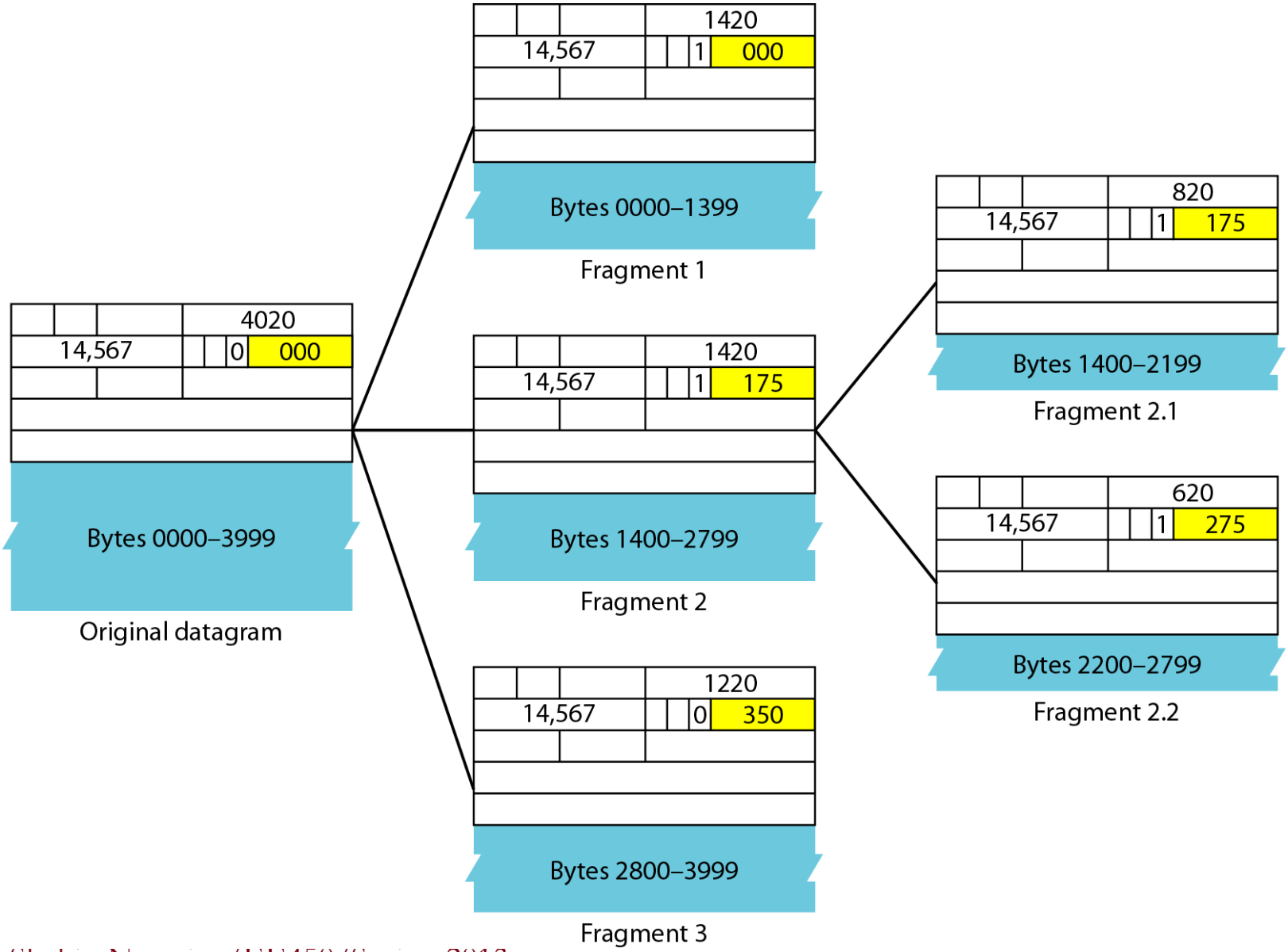


Fragmentation (Cont.)

- Lets assume the network is congested, and then assume routers in the network give preferential treatment. They will give preferential treatment to fragments more than non-fragmented packets because if the fragment gets delayed all previous fragments that were actually delivered will be dropped which means a waste the resources of the network for nothing, also because it's shorter than packet



Fragmentation Example II



Fragmentation – Choosing Size

- Fragmentation size can be chosen differently, so the answer is not unique
- However one good approach is to take the MTU of the frame and subtract from it, the header of the packet. The remaining is the maximum fragment size, then take the next lower one which is the multiple of 8. This will result in minimum number of fragments
- For example: Let's assume I gave you 1550 as the max size and the header is 20, subtract 20 from 1550, 1530, take the next smaller one which is a multiple of 8, i.e., 1528

- Question: Even with fragmentation offset, flag M is required. Why?

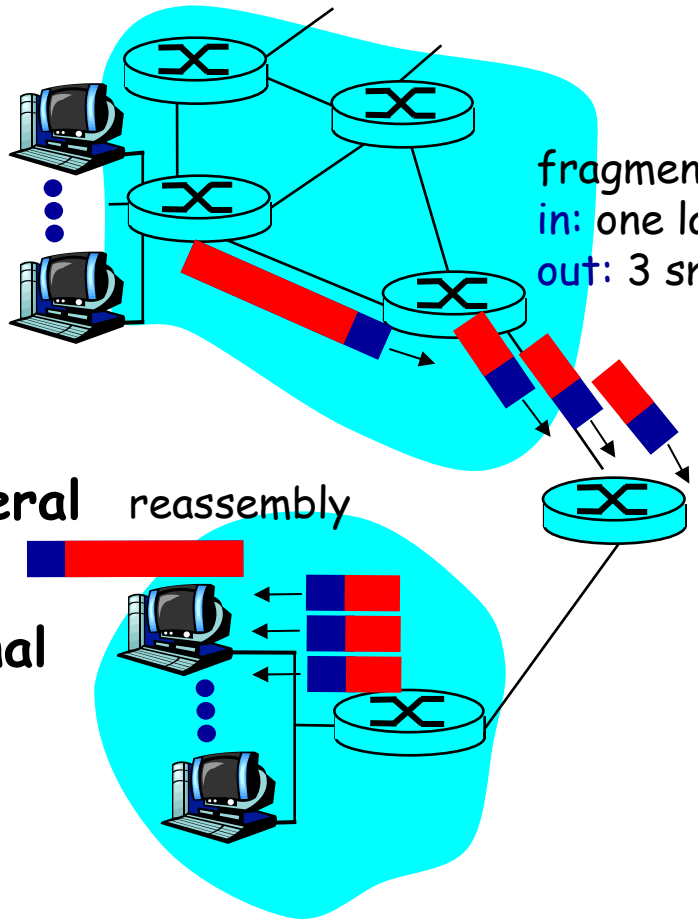
Reassembling the Fragments

- Destination host can use the following strategy to reassemble the fragments:

1. First fragment has an offset field value of 0
2. Divide the length of the first fragment by 8. The second fragment has an offset value equal to that result
3. Divide the total length of the 1st and 2nd fragments by 8. the 3rd fragment has an offset value equal to that result
4. Continue the process

IP Fragmentation & Reassembly Review

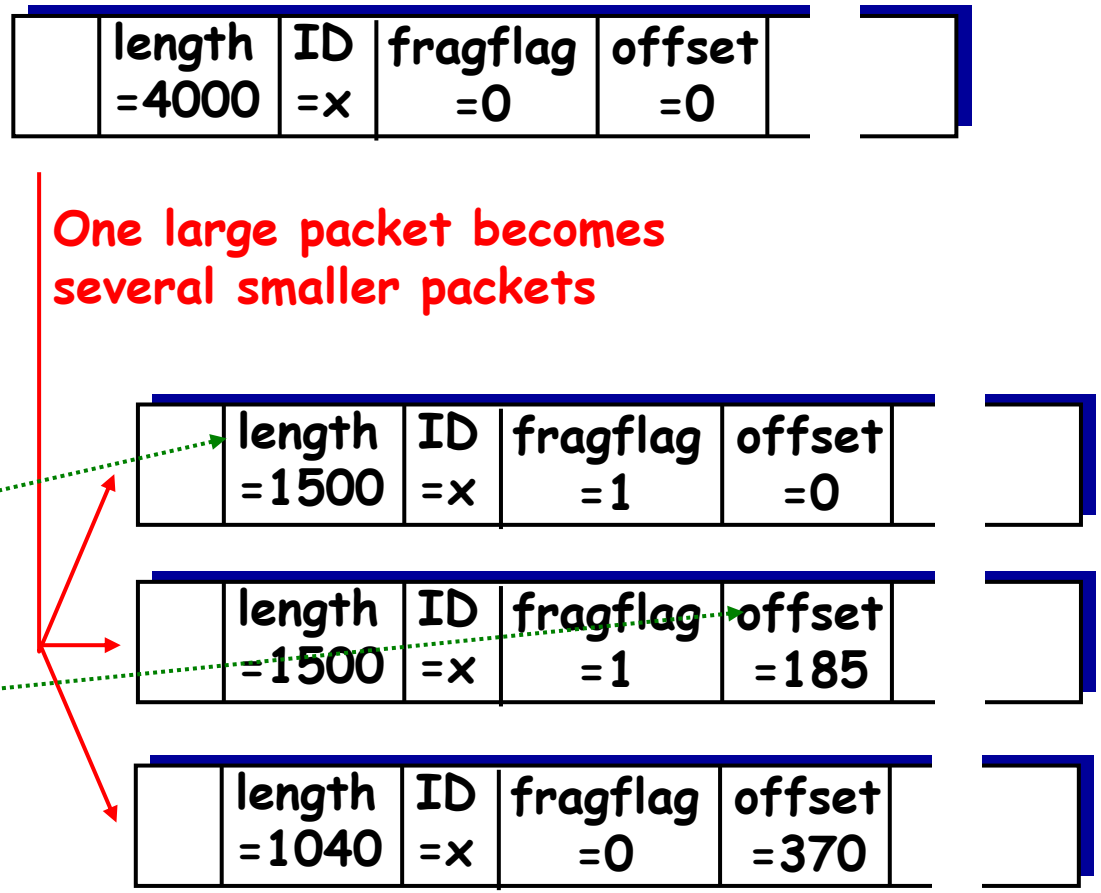
- Network links have MTU (max.transfer size) : largest possible link-level frame
  - different link types, different MTUs
- Large IP datagram* divided ("fragmented") within net
  - One packet becomes several packets
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



*"Datagram" and "packet" are both used in the literature and by people in industry to denote the whole header plus data in layer 3. We would like to use the term packet only in layer 3 and use datagram only in UDP layer 4

Fragmentation Example III

- 4000 byte packet
- MTU = 1500 bytes



IPv4 Addressing

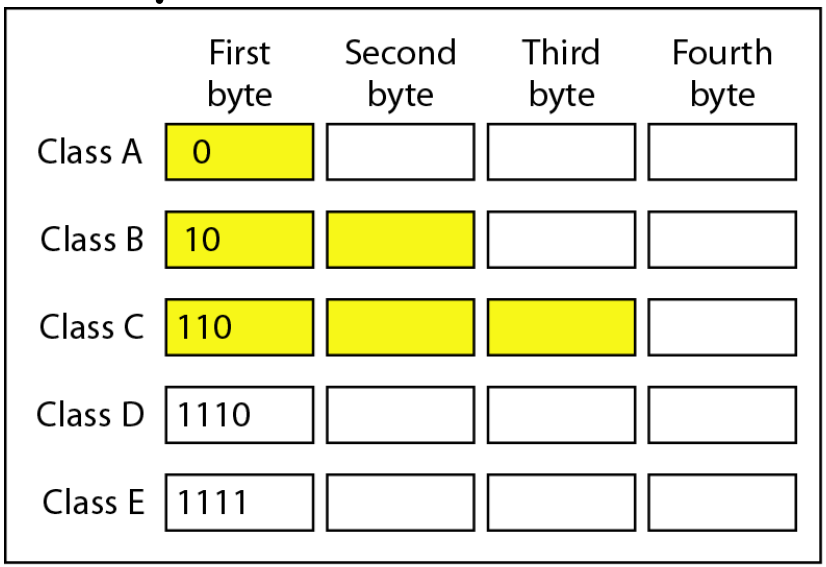
- Each address consists of network address (aka net id) and host address (host id) and is overall 32 bits (2^{32} or 4 billions was considered large in mid 70s)

- We saw that in PSTN after the connection is setup, it is going to be dedicated and no addressing is required, however in packet switching addressing is required

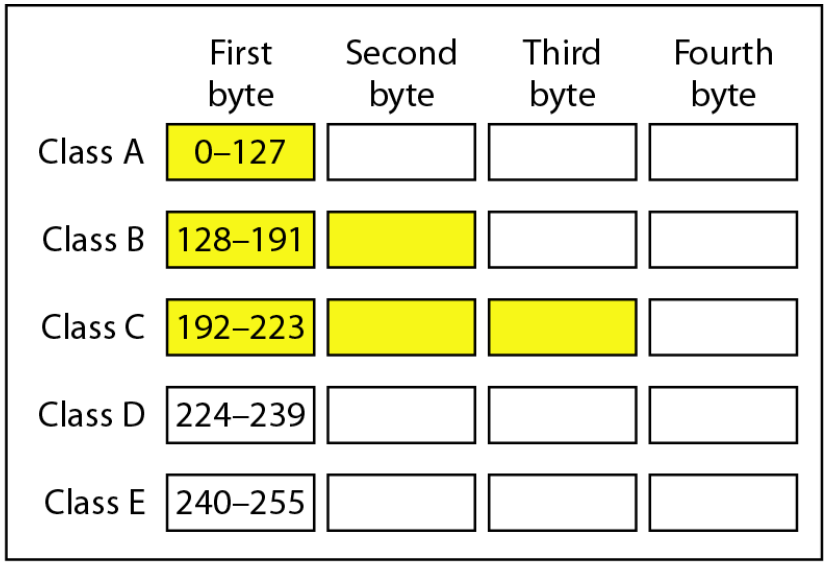
- Assigned by ICANN (Internet Corporation for Assigned Names and Numbers)

Classful IPv4 Addressing

- **Class A**: has 2^7 huge networks. Looking at the first byte, any network with 0 to 127 is class A (0 and 127 for special functions, e.g., 127 is reserved for loop back when transmitter is testing itself: before sending the packet it send it to itself to make sure, its IP stack is working in its machine
- **Class B**: 14 bits for the networks (one of them is USC) and 16 for the hosts
- 1st byte: 128: 10000000 to 191: 101111111



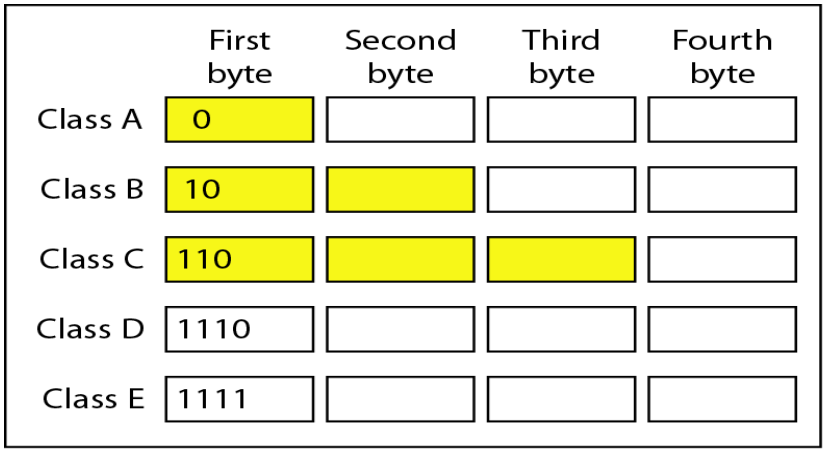
a. Binary notation



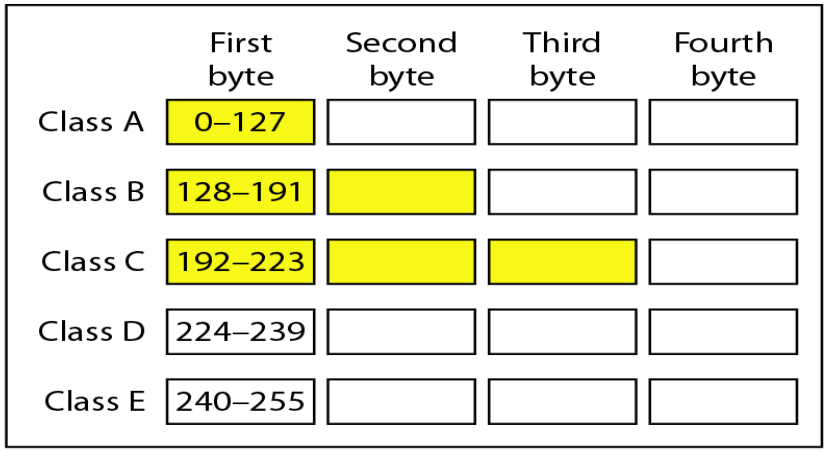
b. Dotted-decimal notation

Classful IPv4 Addressing

- **Class C**: each class has only 256 addresses, but there are 2^{21} networks
- 1st byte: 192: 11000000 to 223: 11011111
- **Class D**: reserved for multicasting. In class D the interest is in the network, it's the group and to see which group is interested in multicasting, because group members may be located in different networks. 1st byte: 224: 11100000 to 239: 11101111
- **Class E**: for experimental purposes (or for future use)
- **In classful addressing, a large part of the available addresses are wasted**



a. Binary notation



b. Dotted-decimal notation

Classful Addressing Example

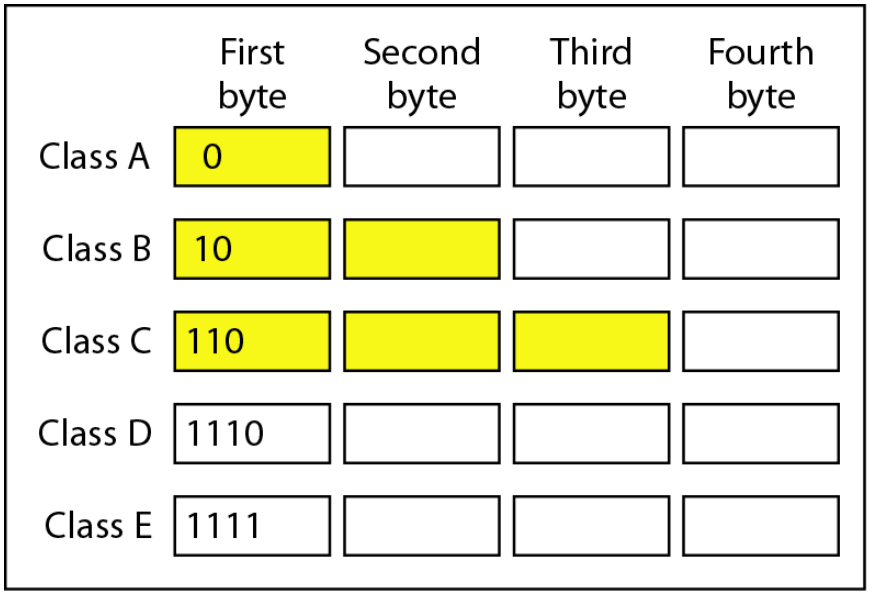
Find the class of each address.

a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 14.23.120.8
d. 252.5.15.111

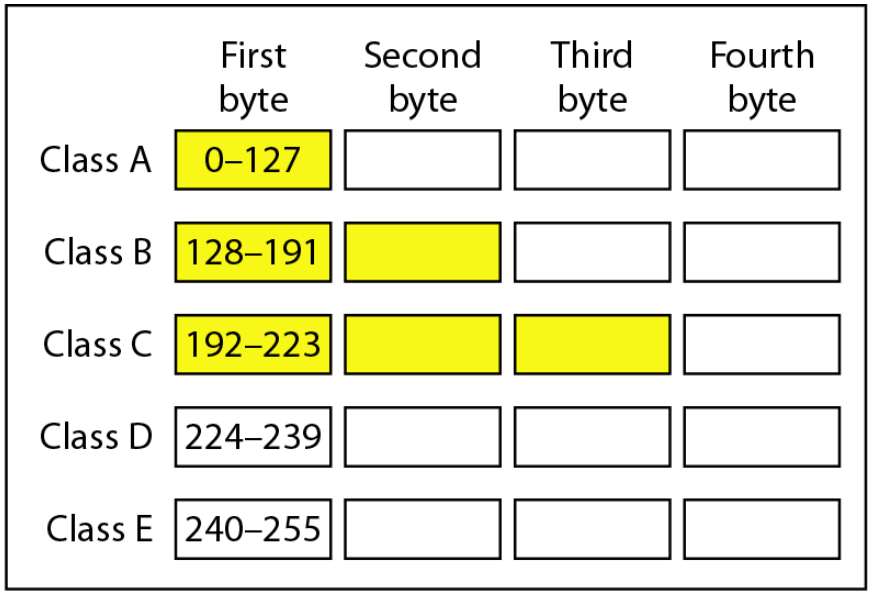
Solution

a. The first bit is 0. This is a class A address
b. The first 2 bits are 1; the third bit is 0. This is a class C address
c. The first byte is 14; the class is A
d. The first byte is 252; the class is E

Number of blocks and block size in classful IPv4 addressing



a. Binary notation



b. Dotted-decimal notation

Class	*Number of Blocks*	*Block Size*	*Application*
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

Special IP Addresses

- There are some IP addresses that are reserved for special purposes, meaning they cannot be used to identify the IP address of a machine (a node)

Special Address	Netid	Hostid	Source/Destination
Network address	Specific	All 0s	None
Direct broadcast address	Specific	All 1s	Destination
Limited broadcast address	All 1s	All 1s	Destination
This host on this network	All 0s	All 0s	Source
Specific host on this network	All 0s	Specific	Destination
Loopback address	127	Any	Destination

Special IP Addresses (Cont.)

- Question: What is a limited broadcast? It is the broadcast on our network, e.g., discovery phase in DHCP
- All 1s means an address on the same network (broadcast within the network) The router that is attached to the network when it sees 255.255.255.255 will drop the packet
- Question: What does a bridge do when it sees 255.255.255.255? The bridge does not recognize IP addresses, therefore it will flood the packet (the frame)
- All 0s: is a temporary address in this network: in DHCP in the beginning: to get an IP address, a hosts uses a temporary IP address of 0.0.0.0; so it's this host on this network, saying that it does not have an IP address yet and that's why it is using all 0s to get an address from DHCP

Special IP Addresses – Example

- Example 2: What address should be used if we want to broadcast on another network?
  - We should put the network id of that network and all 1s for the hosts
- Example 1: USC network address, its direct broadcast address, and number hosts
- Example 3: Maximum number of hosts in a class C network
- Example 4: What network is 0.0.32.41 in?
  - We do not know what network. It is used in any network, say you are at USC network, then it is the a specific host attached to the USC network

Special IP Addresses – Example

- Example 5: what happens if we send a packet to 127.254.254.254?
  - Any address starting with 127 (which is included in class A, but in reality is reserved) is a loopback address, therefore the packet will come back to sender, and won't leave sender's TCP/IP stack. 127 is to test your TCP/IP stack
- Question: What is the difference btn a limited broadcast and a direct broadcast? Limited broadcast (all 1s) is sent to all NICs on the same network segment as that of the source's NIC. This broadcast is not forwarded by router so will only appear on one network segment. However direct broadcast is sent to all hosts on a network. Routers may be configured to forward direct broadcasts on large networks

Default Masks for Classful Addressing

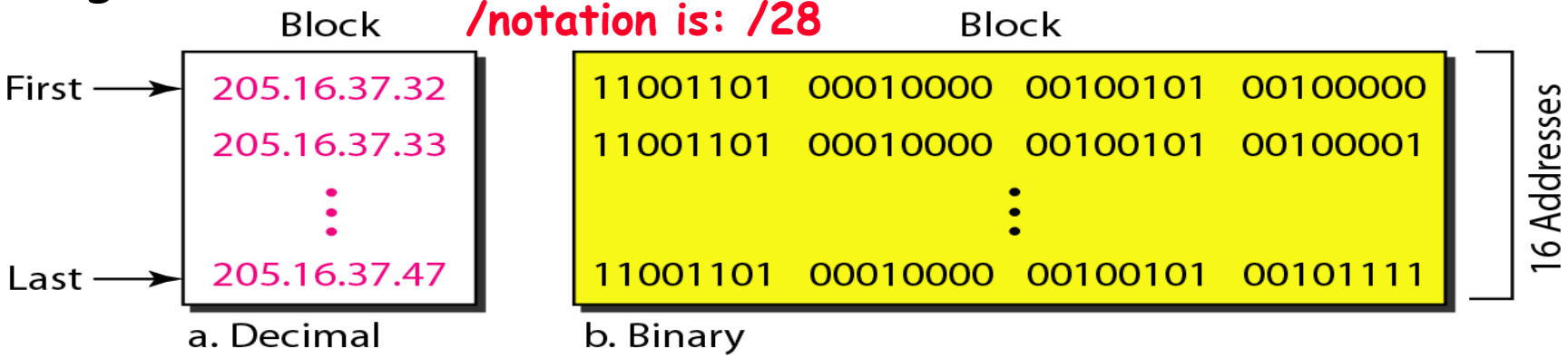
- Netid is chosen by ICANN, and hostid locally by the network administrator
- Mask can help us find the netid and hostid, e.g. the mask for class A has eight 1s, which means the 1st 8 bits in any class A address define netid and the next 24 define the hostid (mask, aka default mask is a 32 bit pattern where there is 1 in place of netid bits and 0 in place of hostid bits)

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	11111111 00000000 00000000 00000000	255.0.0.0	/8
B	11111111 11111111 00000000 00000000	255.255.0.0	/16
C	11111111 11111111 11111111 00000000	255.255.255.0	/24

- CIDR (Classless InterDomain Routing) is the notation that defines the mask and is mainly used for classless IP addressing, however classful IP addressing is a subset of classless, therefore CIDR notation can be used for classful addressing as well

Classless Addressing

- Classful addressing, which is almost obsolete, is replaced with classless addressing
- Example: A block of 16 addresses granted to a small organization



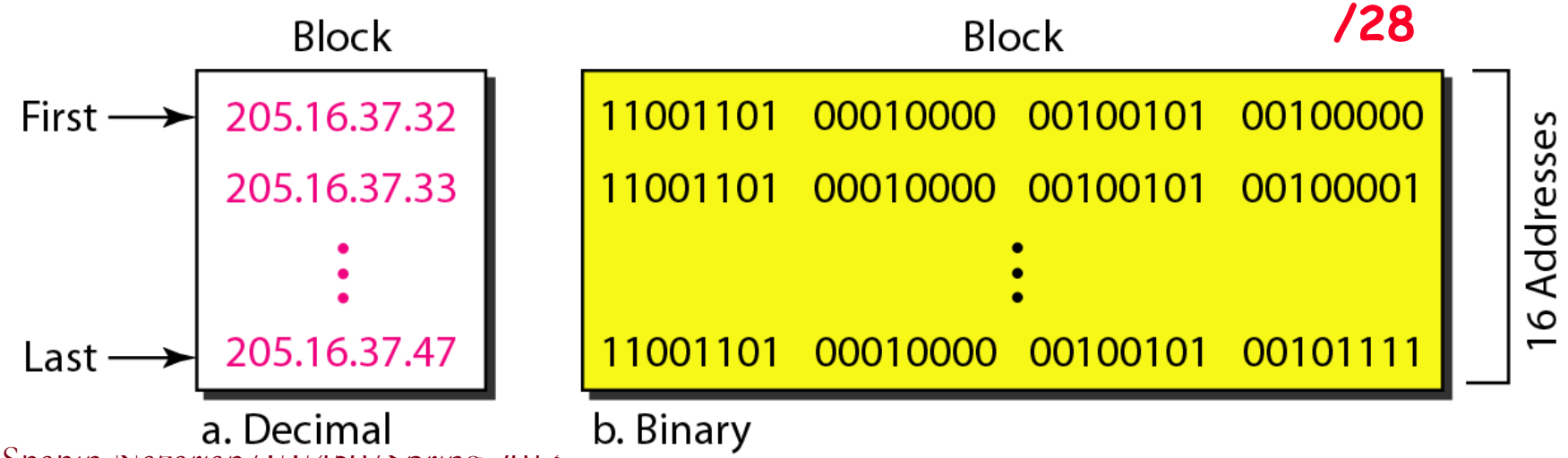
- In IPv4 addressing, a block of addresses can be defined as
- x.y.z.t /n, in which x.y.z.t defines one of the addresses and the /n defines the mask
- Claim: if you know one address from a block and the / notation, you can create, i.e., find from beginning address to the end address of the entire block [note: assumption is that IP addresses in the block are contiguous]

Classless Addressing – Restrictions

- To simplify classless address handling the following restrictions are applied by Internet authorities:
  - The addresses in a block must be contiguous
  - The number of addresses in a block must be 2^n where n ≥0
    - This means a block can have 1, or 2, or 4, … addresses

Classless Addressing – Building a Block

- In x.y.z.t /n: x.y.z.t defines one of the addresses and /n defines the mask
- The first address in the block can be found by setting the rightmost 32 – n bits to 0
- The last address in the block can be found by setting the rightmost 32 – n bits to 1
- The number of addresses in the block can be found by using the formula 2^{32-n}



Classless Addressing – Example

A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

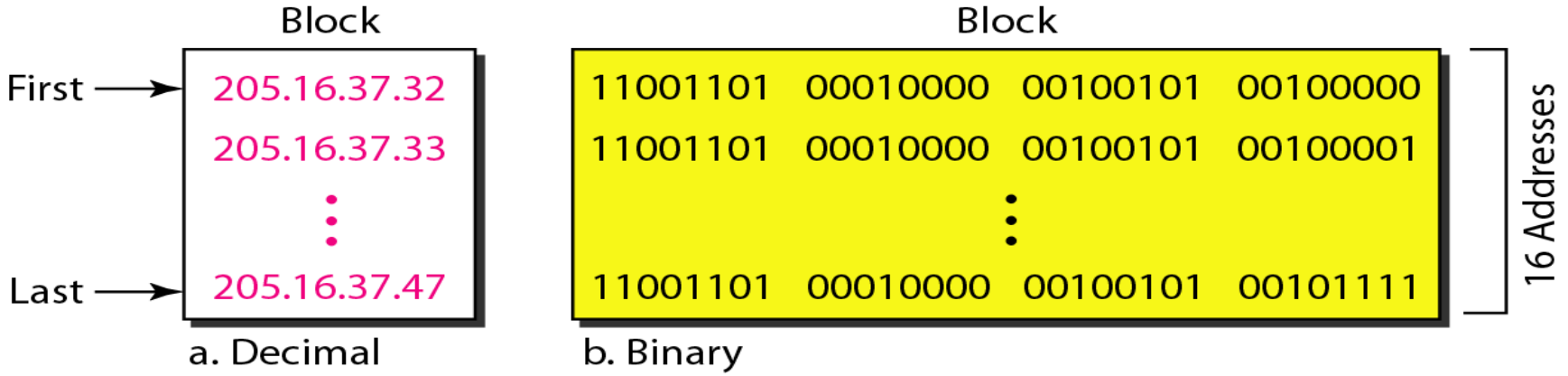
If we set 32–28 rightmost bits to 0, we get

11001101 00010000 00100101 00100000

or

205.16.37.32

This is actually the block:



Subnetting

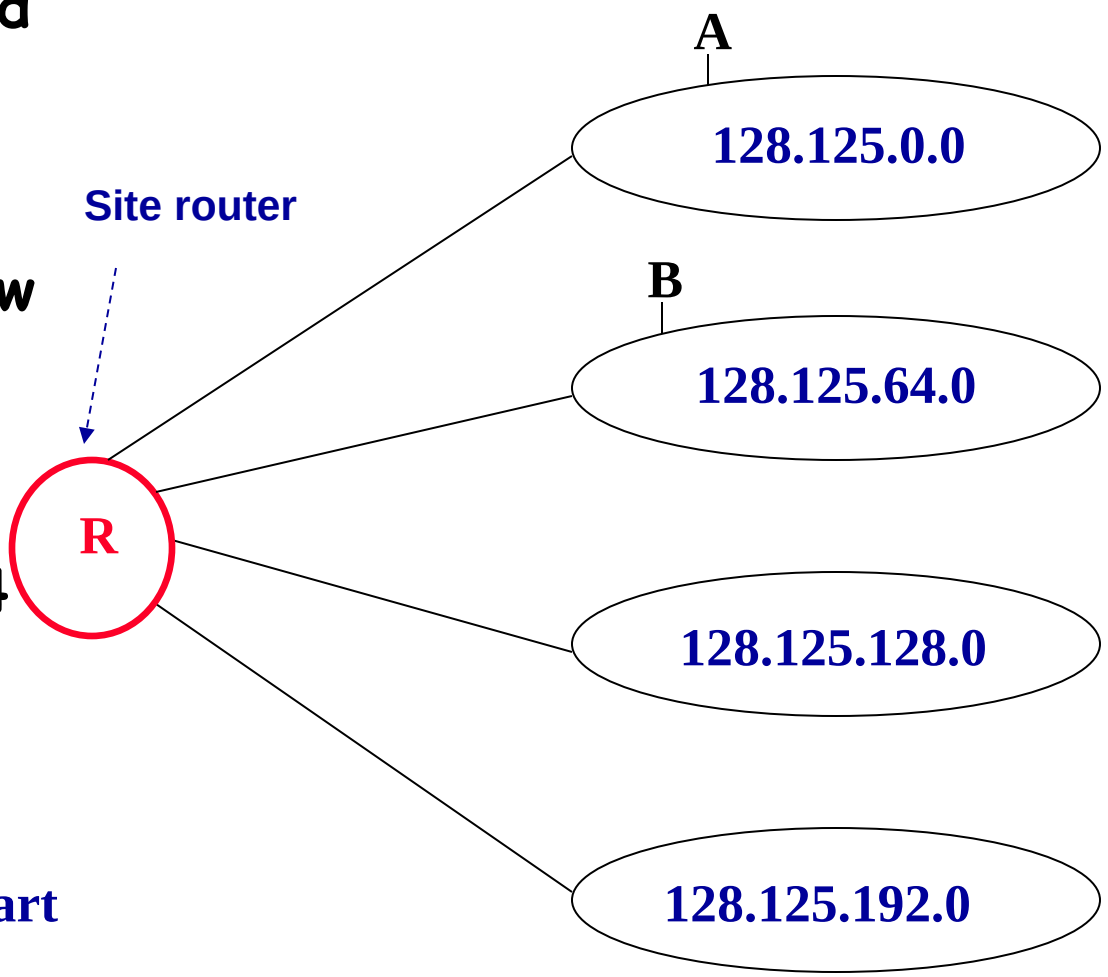
- **Subnetting** is the process of dividing a network into smaller size **clusters** (aka subnetworks or **subnets**)
- Both the network and its subnets are networks, so all we have talked about in networks applies to subnets as well
- Since subnets are different networks, they are connected thru routers; it means host A in subnet 1 needs the help of a router if it needs to message A in subnet 2
- Assume network N is divided into S subnets. Consider a node X outside N. Node X would look at all the nodes in network N as part of one network regardless of what subnets they belong to

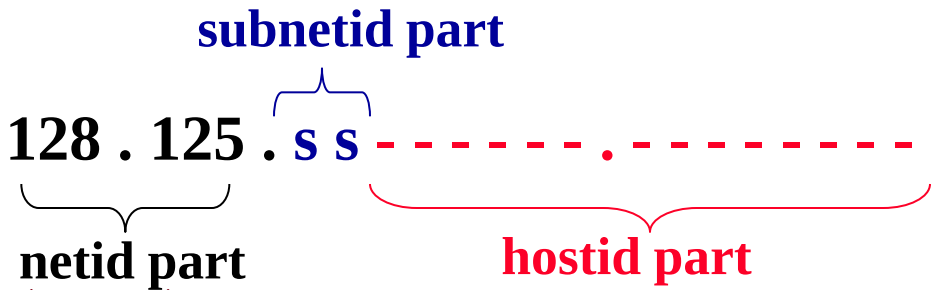
Subnetting Example – USC Network

- Is a Class B network
- USC network address is 128.125.0.0 (hostid should be considered 0 for the network address)
- The number of hosts are 2^{16}-2, ranging from 128.125.0.1 to 128.125.255.254
- It is very difficult for a network admin to manage all these hosts, so one option is to divide the USC network into subnets
- Subnet transparency
  - The hosts are then connected to different subnets (remember that subnets are different networks,) but from the point of the view of a host outside USC, all the hosts are connected to the same network

USC Subnets (Cont.)

- The netid cannot be used to specify subnets, however we know there are less hosts in each subnet, so we can borrow bits from hostid bits
- Here we have borrowed two bits from the 16 bits of hostid to make 4 subnets; so each subnet has 14 bits to identify its hosts





Subnetting – Hierarchy Levels

- A subnet is a network
- Node A wants (128.125.0.15) to send a packet to Node B. It gets the IP address of B from DNS as 128.125.64.17. Node A finds out that they are not on the same subnet, so it needs the help of a router
- Question: What is the common part in subnets? 128.125. This means from the rest of the world it is seen that all host addresses start with 128.125, and they think all the hosts are in the same network, but in reality they are not
- Note that Classes A, B, and C have two levels of hierarchy (netid and hostid) whereas subnetting adds one level of hierarchy (netid, subnetid, hostid)

Subnetting – Subnet Mask

- **Subnet mask** is a 32 bit pattern that is calculated based on the netid, subnetid and hostid bits
- In our USC example with 4 subnets the subnet mask is 255.255.192.0
- Example: Network 215.215.215.0 (class C) is divided into 32 subnets. What is the subnet mask?
  255.255.255.248
- In reality not all subnets are of the same size
- All hosts on the same subnet have the same subnet mask
- All subnets have the same subnet mask if the subnets are of the same size
- For subnets with different sizes, subnet masks are different

Subnetting – Masking

- Subnet mask needs to be configured manually or dynamically by DHCP in each node (host or router)
- Masking is the process of extracting the network address (if no subnetting was done) or subnet address (if subnetting was done) from an IP address
- Masking is the bit-by-bit logical ANDing of the default mask (w/o subnetting) or the subnet mask (w/ subnetting) with the IP address, meaning the result of that AND generates the network address

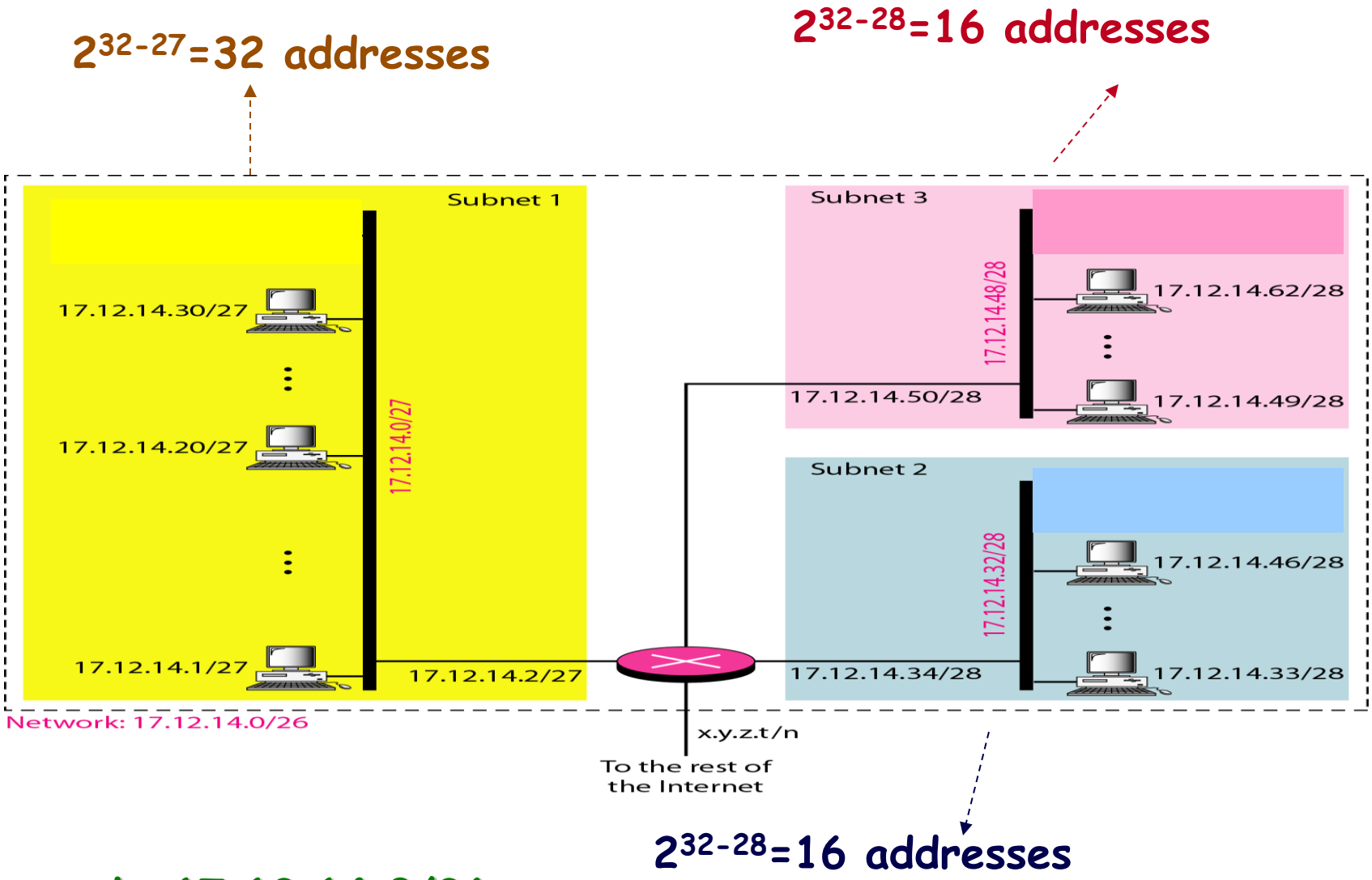
Subnetting Example II – Masking

- Going back to one of our subnetting examples, node A with IP address 128.125.0.15 wants to send a packet to B. It gets the IP address of B from DNS as 128.125.64.17

B's subnet address = 255.255.192.0 AND 128.125.64.17
= 128.125.64.0

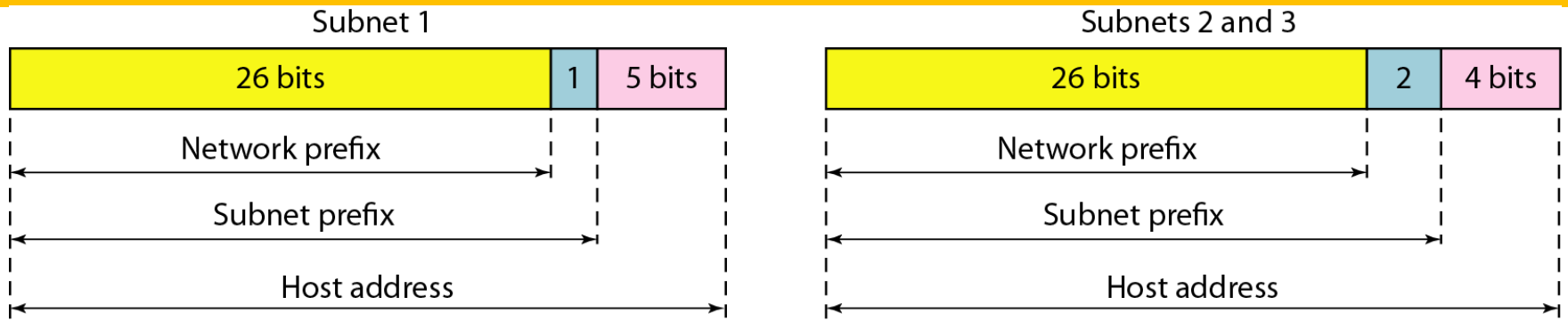
A's subnet address = 255.255.192.0 AND 128.125.0.15
= 128.125.0.0

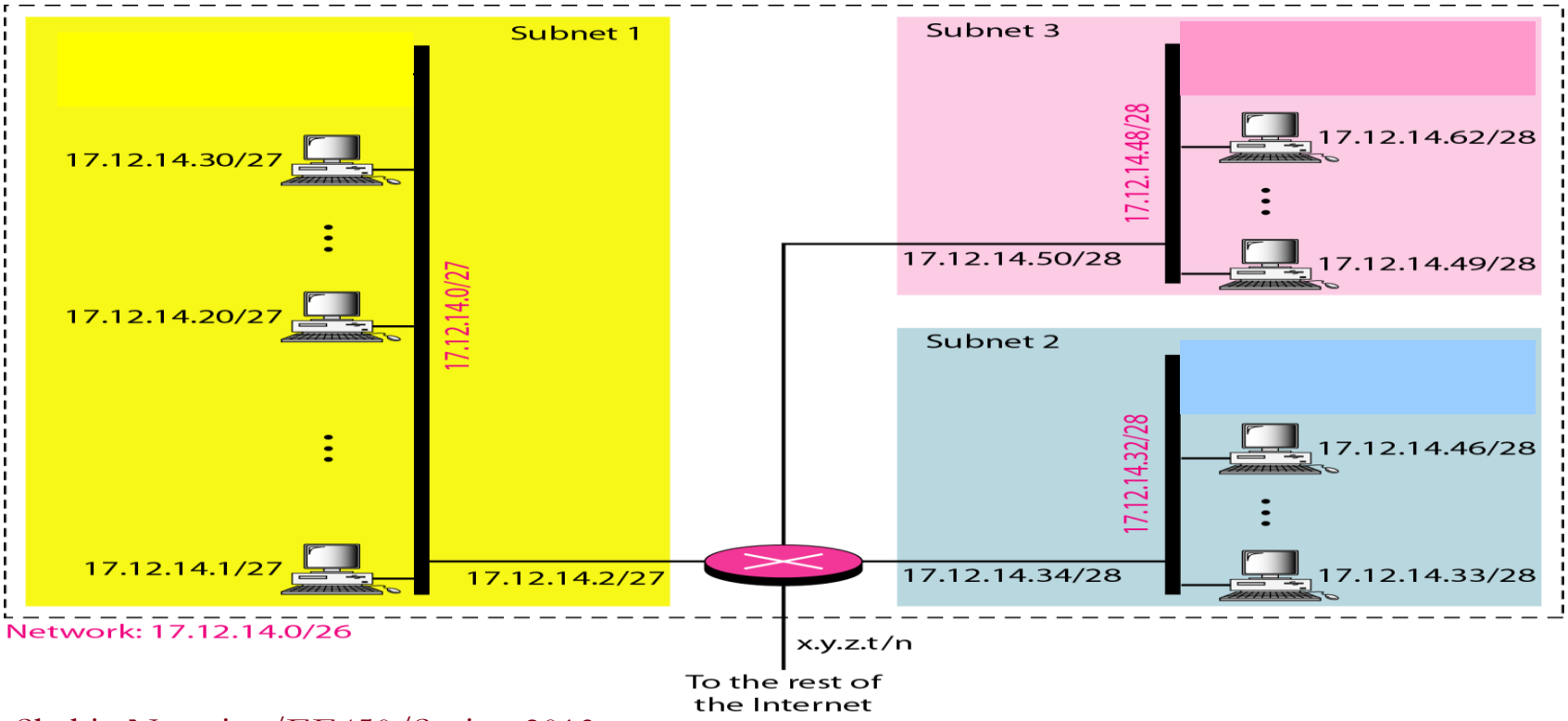
Subnetting Example III – Different Sizes



Network: 17.12.14.0/26

Subnetting Example III (Cont.)





Classless Addressing – Address allocation

- The ultimate responsibility of address allocation is on ICANN, however ICANN typically assigns a large block of addresses to an ISP

- The ISP, in turn, divides the assigned block into smaller subblocks and grants them to customers

- The process of receiving one large block and distributing it to Internet users is referred to as **address allocation** (many blocks of addresses are aggregated in one block and granted to one ISP)

Address Allocation – Example

- An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:

a. The first group has 64 customers; each needs 256 addresses

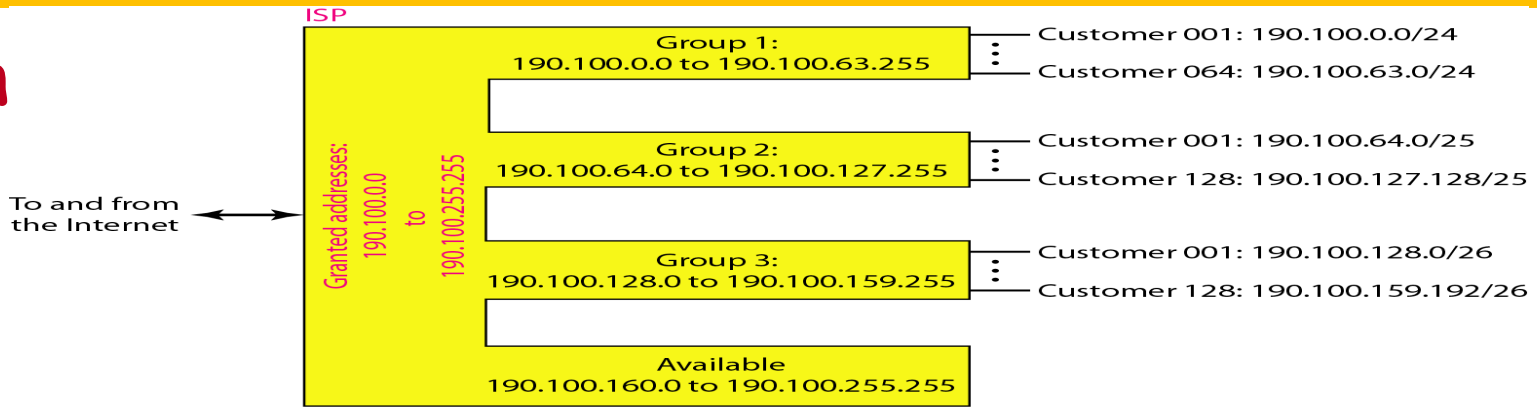
b. The second group has 128 customers; each needs 128 addresses

c. The third group has 128 customers; each needs 64 addresses

- Design the subblocks and find out how many addresses are still available after these allocations

Address Allocation – Example (Cont.)

Solution



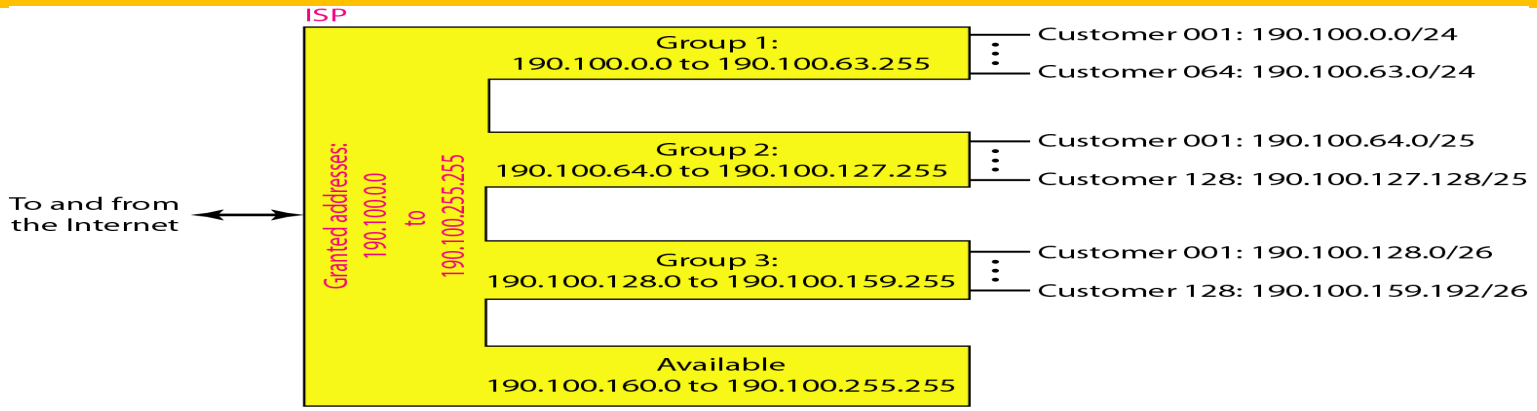
Group 1

For this group, each customer needs 256 addresses. This means that 8 (log2 256) bits are needed to define each host. The prefix length is then 32 – __ = __. The addresses are

1st Customer:	*190.100.0.0/24*	*190.100.0.255/24*
2nd Customer:	*190.100.1.0/24*	*190.100.1.255/24*
. . .		
64th Customer:	*190.100.63.0/24*	*190.100.63.255/24*

Total = 64 × 256 = 16,384

Address Allocation – Example (Cont.)



Group 2

For this group, each customer needs 128 addresses. This means that 7 (log2 128) bits are needed to define each host. The prefix length is then 32 – __ = __. The addresses are

1st Customer:	*190.100.64.0/25*	*190.100.64.127/25*
2nd Customer:	*190.100.64.128/25*	*190.100.64.255/25*
. . .		
128th Customer:	*190.100.127.128/25*	*190.100.127.255/25*

Total = 128 × 128 = 16,384

Address Allocation – Example (Cont.)

Group 3

For this group, each customer needs 64 addresses. This means that 6 ($\log_2 64$) bits are needed to each host. The prefix length is then 32 – 6 = 26. The addresses are

1st Customer:	*190.100.128.0/26*	*190.100.128.63/26*
2nd Customer:	*190.100.128.64/26*	*190.100.128.127/26*
. . .		
128th Customer:	*190.100.159.192/26*	*190.100.159.255/26*

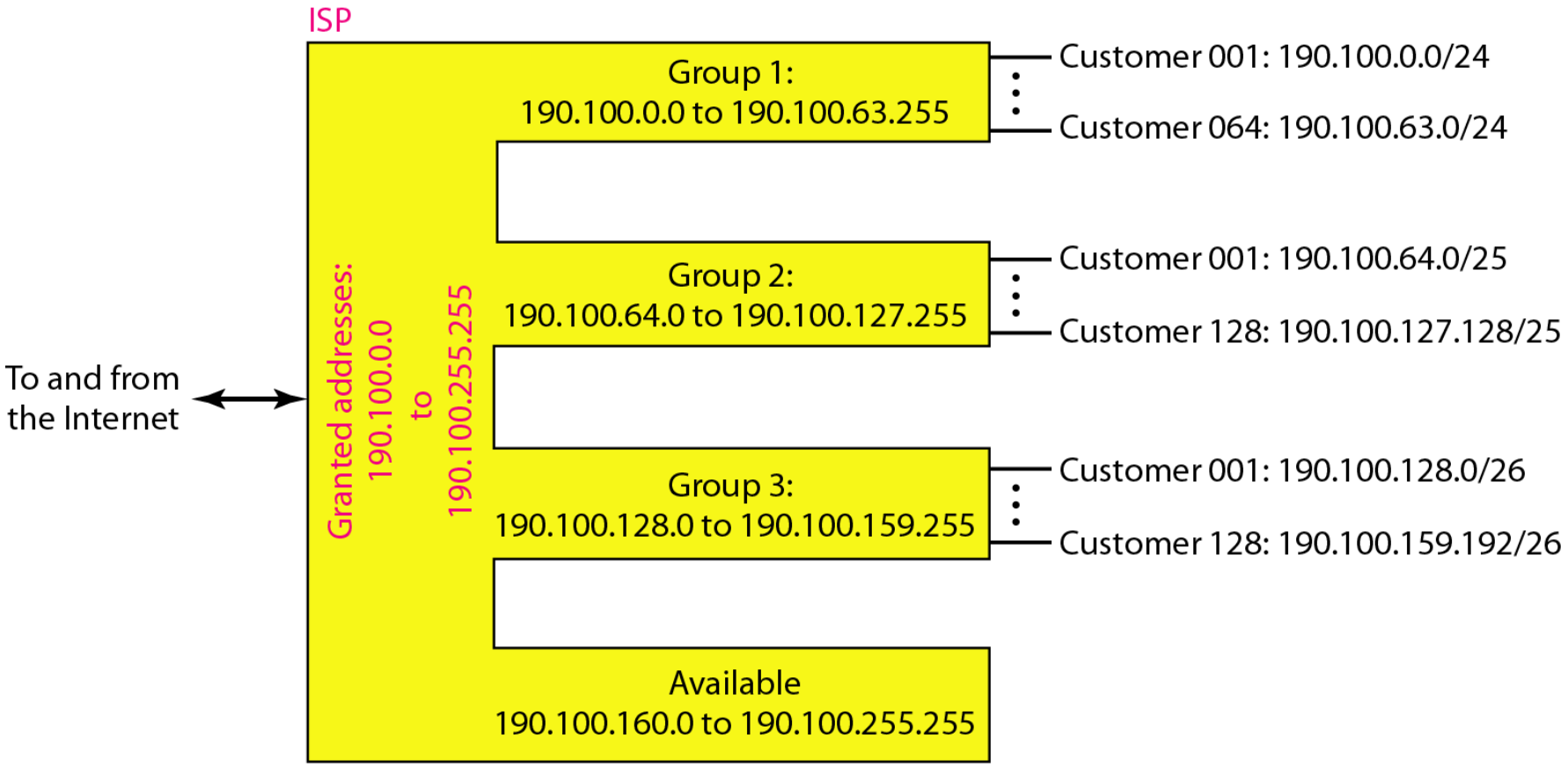
Total = 128 × 64 = 8192

Number of granted addresses to the ISP: 65,536

Number of allocated addresses by the ISP: 40,960

Number of available addresses: 24,576

Address Allocation – Example (Cont.)



Private IP Addressing

- A public IP address is required when we want to communicate across a public network, however if we are connected to a private network and our goal is to communicate internally, we do not need a public IP address
- Private networks are the ones that are limited to the hosts attached to them, and they are not going to communicate with the outside
- A host on a private network can hence have the same IP address as that in another private network

	Range			*Total*
Class A:	10.0.0.0	to	10.255.255.255	2^{24}
Class B:	172.16.0.0	to	172.31.255.255	2^{20}
Class C:	192.168.0.0	to	192.168.255.255	2^{16}

Private IP Addressing (Cont.)

- But of course, hosts on the same private network cannot have the same private address
- Some private addresses are allocated for use in private networks
- ICANN puts this specifications, to let the routers know the private addresses and drop the packets for private communication. Routers are configured with these addresses to know they are private
- Note that the router connected to the private network has a private IP address on its private network connection

	Range			*Total*
Class A:	10.0.0.0	to	10.255.255.255	2^{24}
Class B:	172.16.0.0	to	172.31.255.255	2^{20}
Class C:	192.168.0.0	to	192.168.255.255	2^{16}

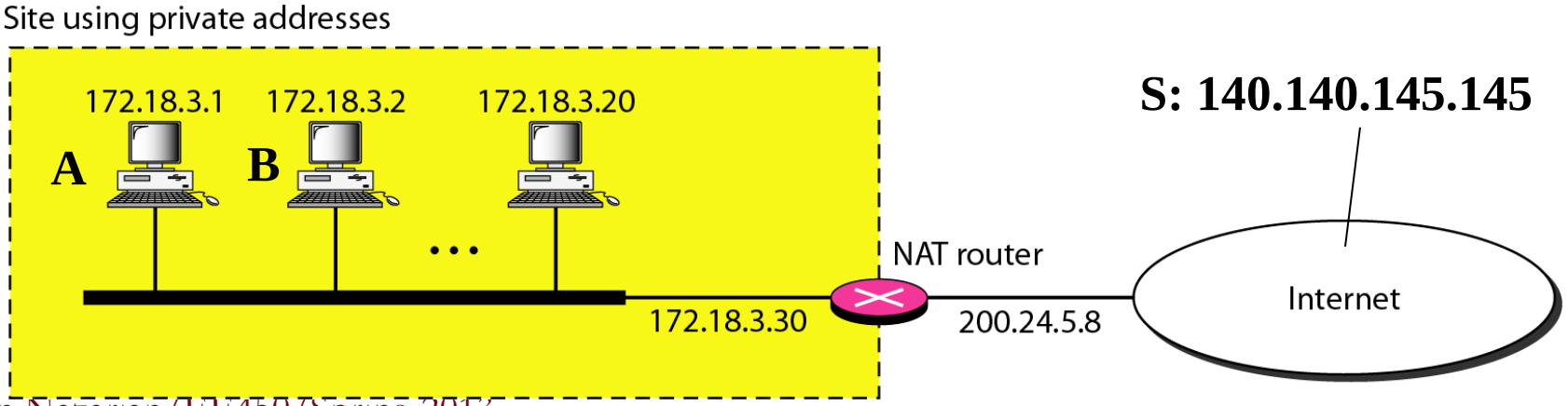
A Problem with Private IP Addressing

- Suppose the private network address 172.18.0.0
- Router has a private IP address on the private network side
- Assume two hosts Pr1 (172.18.0.1) and Pr2 (172.18.0.2) Communications between Pr1 and Pr2 use their private IP address
- Now if Pr1 wants to connect to a public server (e.g., a DNS server) outside this private network, it is not possible, because Pr1 does not have a public address and if Pr1 attempts to send a packet destined to the server, the router will drop it
- NAT (Network Address Translation) is used as a solution. NAT is either a standalone hardware, translating a private IP address to a public IP address or a piece of software that is installed into the border router that separates the private network from the public

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}

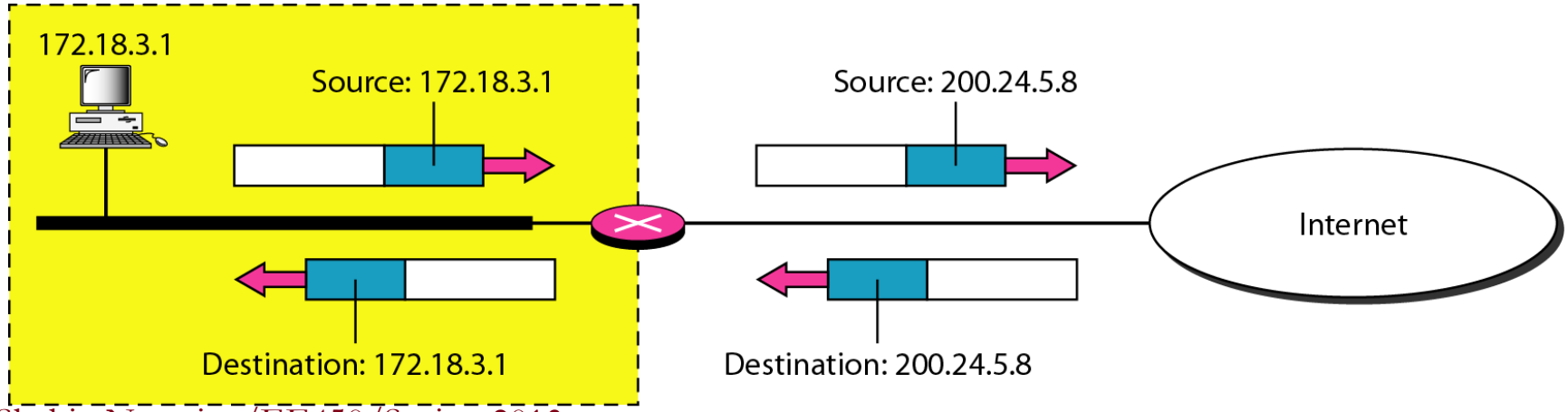
NAT

- A wants to send a packet to S: Packet of A: source IP address: 172.18.3.1, destination address: 140.140.145.145, with source's MAC and router's MAC addresses in the frame
- Frame gets to the router and router processes it, gets the packet, sees the private source IP address, it then maps it to a 200.24.5.8 and sends the packet. In the NAT table it will record this translation. The frame comes to the server
- S does not know the packet came from NAT; it thinks the packet came from a host with address: 200.24.5.8. When S replies, router will receive it and translates it back to 172.18.3.1 as the destination IP address

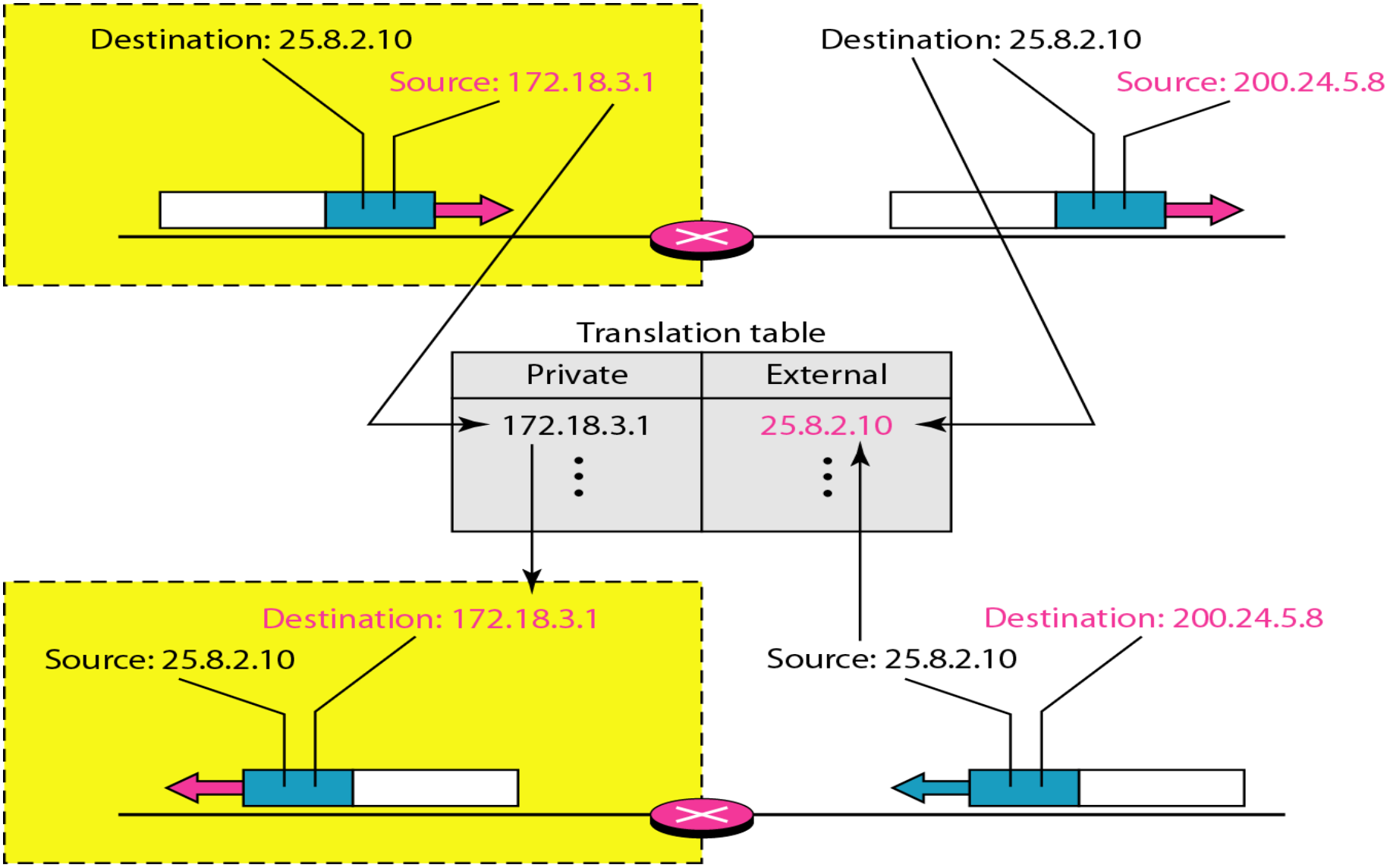


NAT (Cont.)

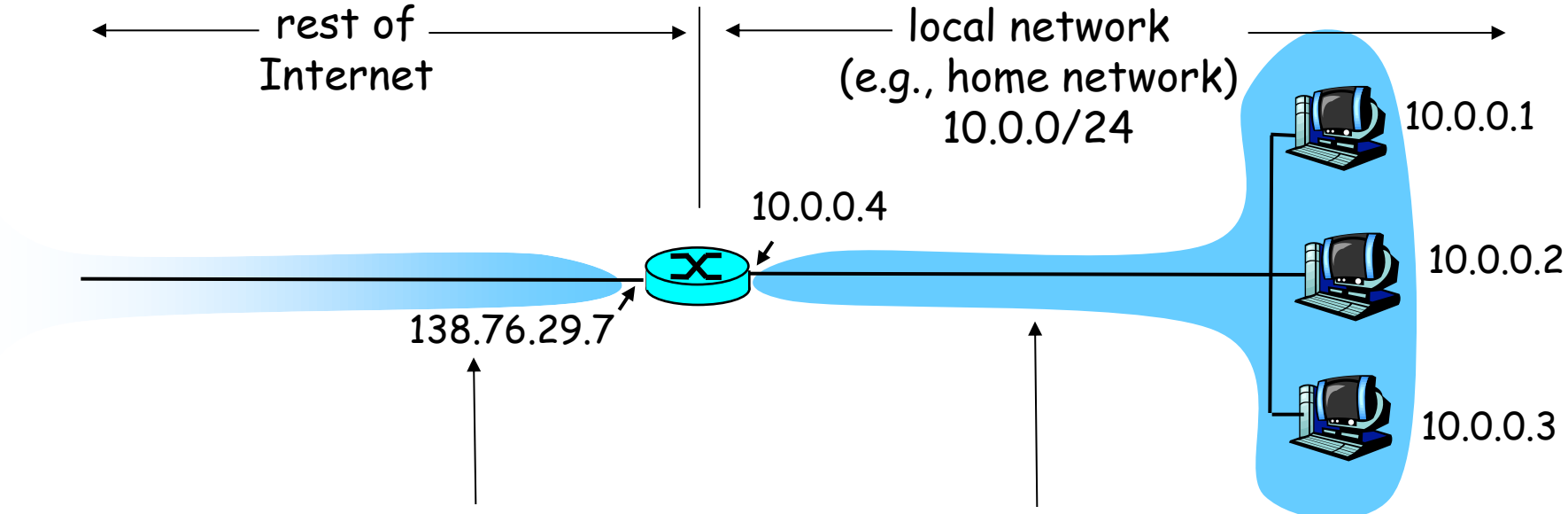
- NAT is a special type of a proxy server, remember that proxy indicates substitution (example of a NAT: the one in Windows 2000)
- NAT may be static (a private IP address to a public IP address, where the public IP address is always the IP address) or dynamic (a private IP address to a public IP address drawn from a pool of global IP addresses. 200.24.5.8 is just one of them and more can exist
- If we have multiple private hosts, we can have a pool of public IP addresses, but even one public address can support them all



NAT Example I



NAT – Example II



All packets *leaving* local network have same single source NAT IP address: 138.76.29.7, different source port numbers

Packets with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT and PAT

- NAT and PAT is used as a solution for the cases where multiple private hosts need to communicate with a public network simultaneously. Source port number (the application port number, which is chosen dynamically, randomly by the NOS) is used
- Here both 172.18.3.1 and 172.18.3.2 want to communicate with HTTP server (well-known port #80)
- Server thinks that the packets came from the same host with two applications within that host!

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
. . .	. . .	. . .	. . .	. . .

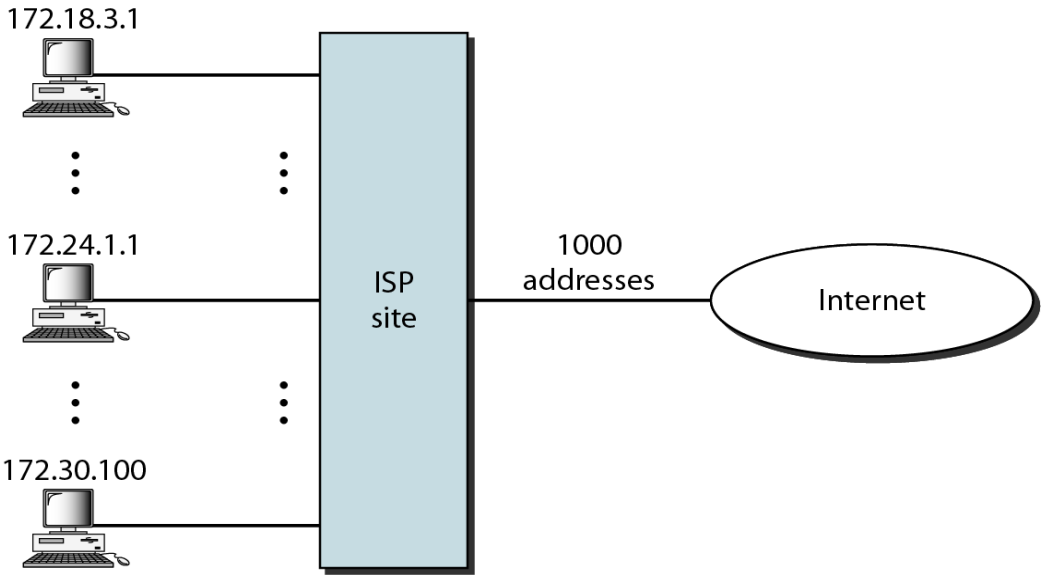
NAT and PAT (Cont.)

- When the source replies the source port # is 80 (and destination Port # is either 1400 or 1401,) source IP address is 140.140.145.145 and destination IP address is 25.8.3.2
- When the packets come to NAT and PAT, it can translate them properly to the port # and IP address in the private network
- You cannot see the port # in the packet. You will see that if you go to layer 4. NAT by itself cannot know what the port # is. In packet format, there is no port # seen, it's inside the Data. Layer 4 would see it
- Note that 1400 and 1401 could be chosen as the same by NOS. The NAT would reject those packets, however NAT and PAT can translate them to two different port numbers

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
. . .	. . .	. . .	. . .	. . .

NAT Application in Dialup Service

- An ISP that serves dialup customers can use NAT technology to conserve IP addresses
- Example: In the following figure an ISP is shown that is granted only 1000 addresses, however it has 100,000 customers
- Each of the customers is assigned a pool of network address. ISP translates each of the 100,000 source addresses in outgoing packets to one of the 1000 global addresses
- It also translates the global destination address in the incoming packets to the corresponding private address
-

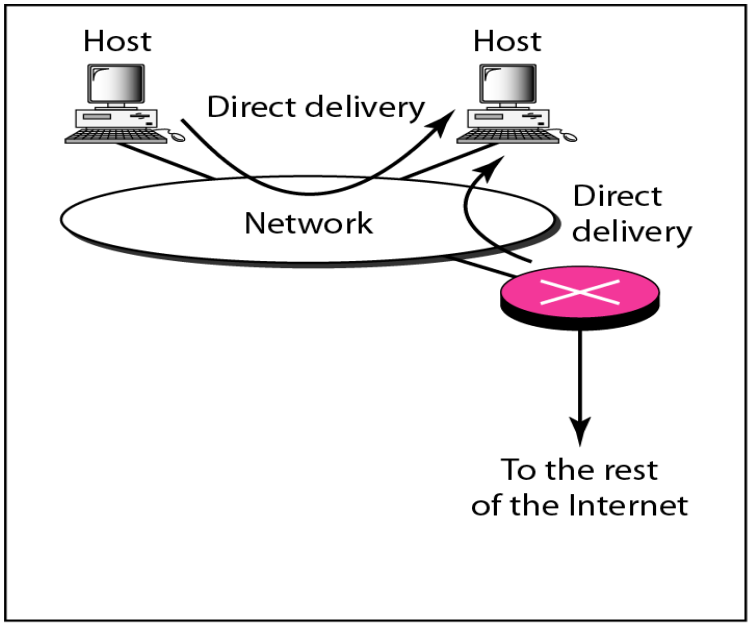


IP Packet – Delivery, Forwarding and Routing

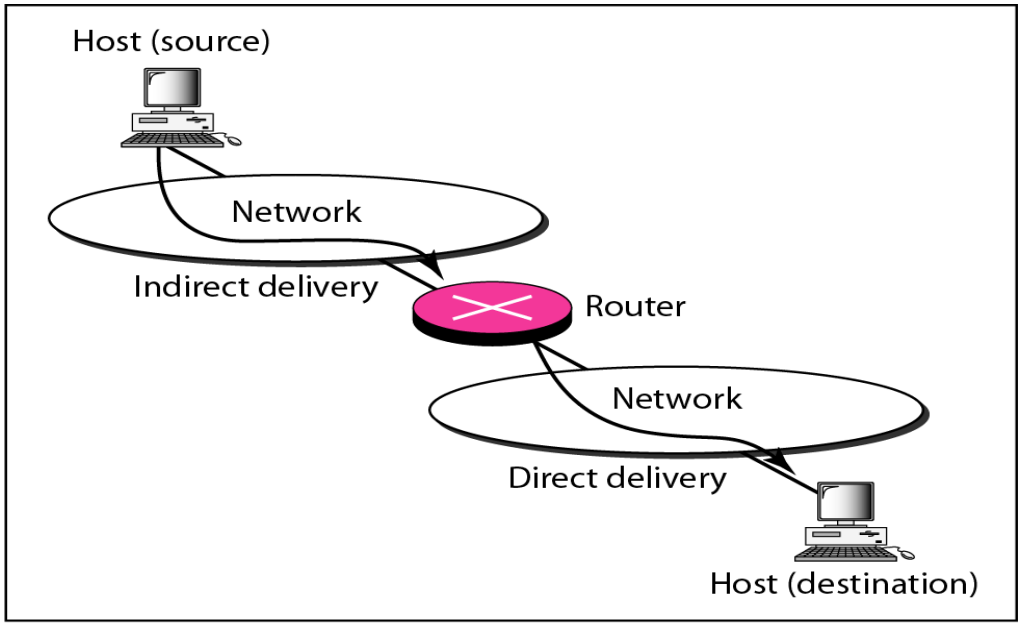
- **Delivery** refers to the way a packet is handled by the underlying networks **under the control of the network layer**
  - **Forwarding** refers to the way a packet is **delivered** to the next station
  - **Routing** refers to the way routing tables are created **to help in forwarding**

IP Packet – Delivery (Cont.)

- **Local delivery**: both source and destination hosts are on the same network. No router's help is required. IP packet is encapsulated into a MAC-frame and forwarded directly to the destination
- **Remote delivery**: source and destination hosts are on different networks. IP packet is encapsulated into a layer2 frame and then forwarded to the next hop router



a. Direct delivery



b. Indirect and direct delivery

Two Key Network Layer Functions

- IP packet delivery involves routing and forwarding
- Routing is the process of discovering and selecting the path to the destination based on some criteria

- Routing is a Network layer process, whereas forwarding is a Network layer/Data Link layer process

Routing Table

- Routing tables are generated by routing algorithms and consist of at least the following:
  - Network address (i.e., the network portion of the IP address of the network that destination is connected to
  - Mask
  - Next-hop address
  - Interface

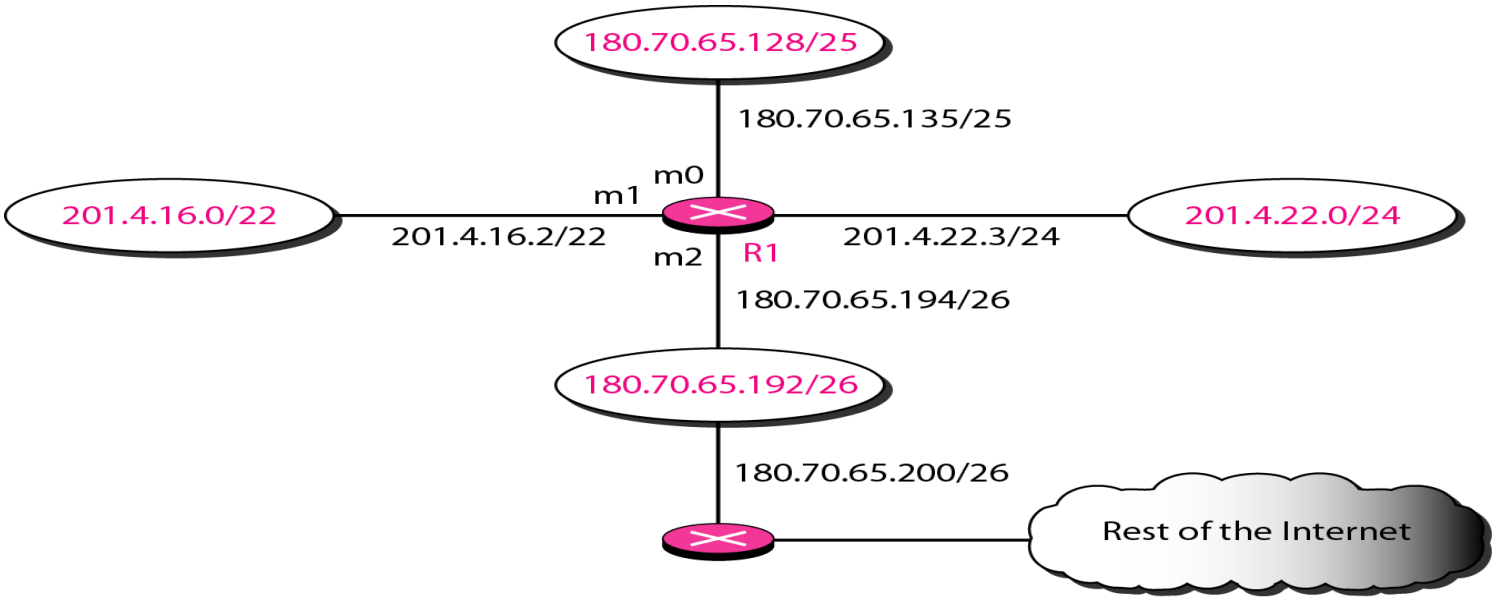
Mask (/n)	Network address	Next-hop address	Interface
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	.....
.....	.....	.....	.....

Forwarding Module and Routing Table

- When the router receives the packet it extracts (forms) the destination IP address and logically ANDs it bit by bit by every entry in the subnet mask column and matches the result with the network address. If there is a match, then router knows the next hop router and the interface through which the packet should be forwarded
- The reason Search table goes through ARP, is that when it finds the next hop IP address from the table, then it needs the MAC address of that next hop (whether the router or the destination)
- The ARP is a local protocol it is only used to get the MAC address of a node in the same network
- If the MAC address happens to be there in the ARP cache then it uses it, but if not, then the ARP module needs to execute the ARP protocol to get the MAC address

Routing Table Example

- **Make the routing table for R1 for the following configuration:**

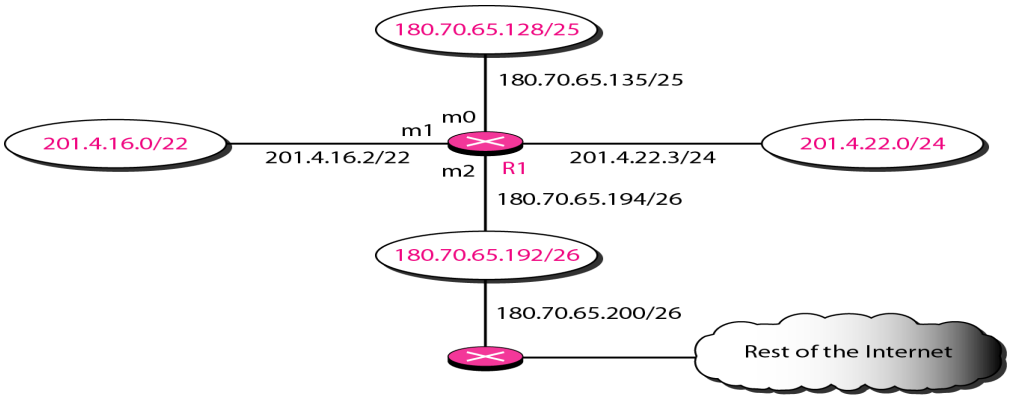


Answer

Mask	*Network Address*	*Next Hop*	*Interface*
/26	180.70.65.192	—	m2
/25	180.70.65.128	—	m0
/24	201.4.22.0	—	m3
/22	201.4.16.0	....	m1
Any	Any	180.70.65.200	m2

Routing Table Example (Cont.)

- Show the forwarding process if a packet arrives at R1 with the destination address 180.70.65.140
- The router performs the following steps:
  1. The first mask (/26) is applied to the destination address. Not a match
  2. The second mask (/25) is applied to the destination address. The next-hop address (the destination address in this case) and the interface number m0 are passed to ARP for further processing



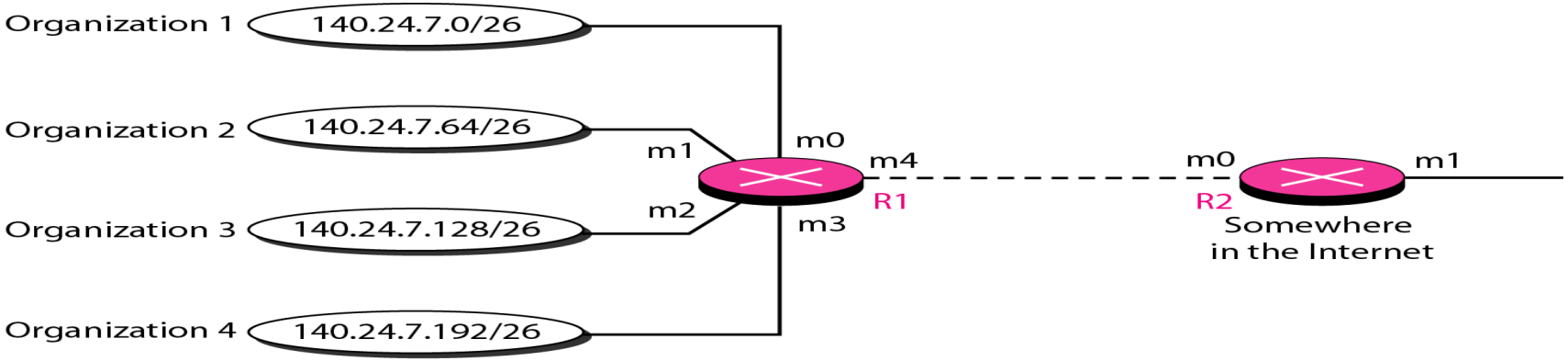
Mask	Network Address	Next Hop	Interface
/26	180.70.65.192	—	m2
/25	180.70.65.128	—	m0
/24	201.4.22.0	—	m3
/22	201.4.16.0	....	m1
Any	Any	180.70.65.200	m2

Routing Table Example II

- Repeat the previous example for the destination address 201.4.22.35
- The router performs the following steps:
  1. The first mask (/26) is applied to the destination address. Not a match
  2. The 2nd mask (/25) is applied. Not a match
  3. The result for the 3rd mask (/24) is 201.4.22.0, which matches the corresponding network address. The destination address of the packet and the interface number m3 are passed to ARP

Address Aggregation

- Classless addressing divides the network into manageable blocks and this would increase the size of the routing tables (and hence the time required to search the table.) To alleviate the problem the idea of **address aggregation** was developed
- R1 has a longer routing table because each packet must be correctly routed to the appropriate organization. However R2 can have a very small routing table



Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/26	140.24.7.192	----------	m3
/0	0.0.0.0	Default	m4

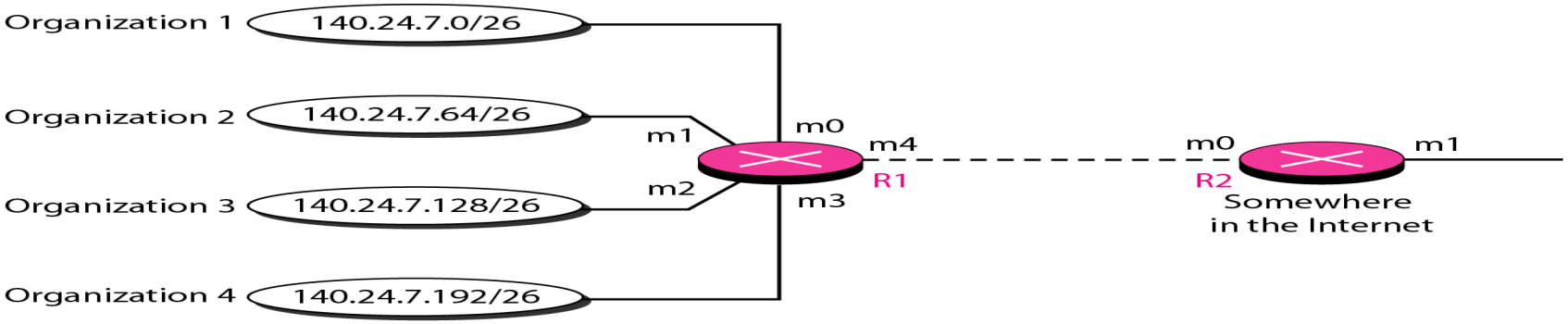
Routing table for R1

Mask	Network address	Next-hop address	Interface
/24	140.24.7.0	----------	m0
/0	0.0.0.0	Default	m1

Routing table for R2

Address Aggregation (Cont.)

- **Note:**
  - **The idea of address aggregation is similar to that of subnetting, however here the network for each organization can be independent**



Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/26	140.24.7.192	----------	m3
/0	0.0.0.0	Default	m4

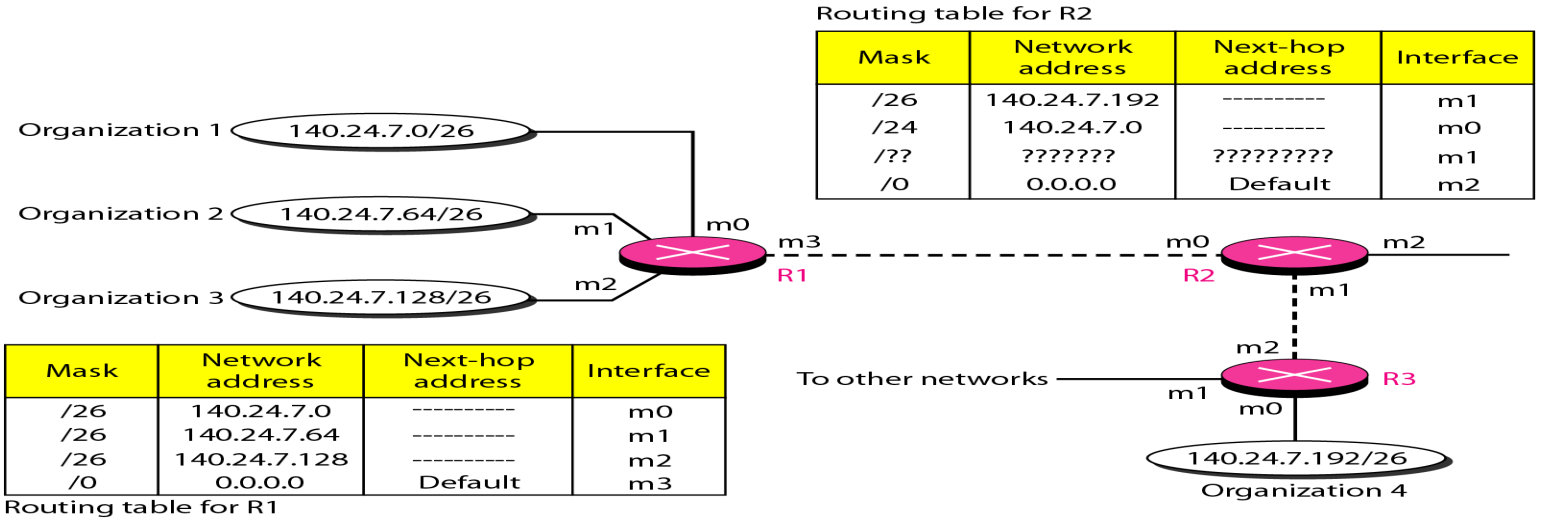
Routing table for R1

Mask	Network address	Next-hop address	Interface
/24	140.24.7.0	----------	m0
/0	0.0.0.0	Default	m1

Routing table for R2

Longest Mask Matching

- Can we still use the idea of address aggregation and still assign block 140.24.7.192/26 to the remote organization 4?
  - Yes, because routing in classless addressing uses another principle, called **longest mask matching**



Routing table for R1

Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/0	0.0.0.0	Default	m3

Routing table for R2

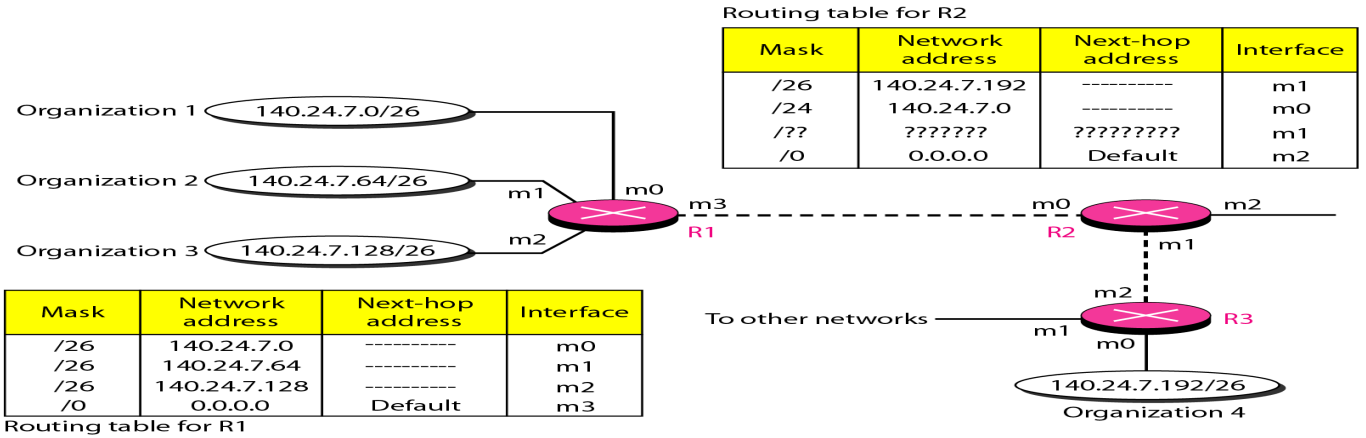
Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m1
/24	140.24.7.0	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2

Routing table for R3

Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2

Longest Mask Matching (Cont.)

- Therefore if a router gets several matches, it chooses the longest match, i.e., the one with more positions that are matched
- It is the one with the largest number of leading address bits matching those of the destination address
- Check whether 140.24.7.200 is routed properly by router R2, even if it sees several matches in its table



Routing table for R1

Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/0	0.0.0.0	Default	m3

Routing table for R2

Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m1
/24	140.24.7.0	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2

Routing table for R3

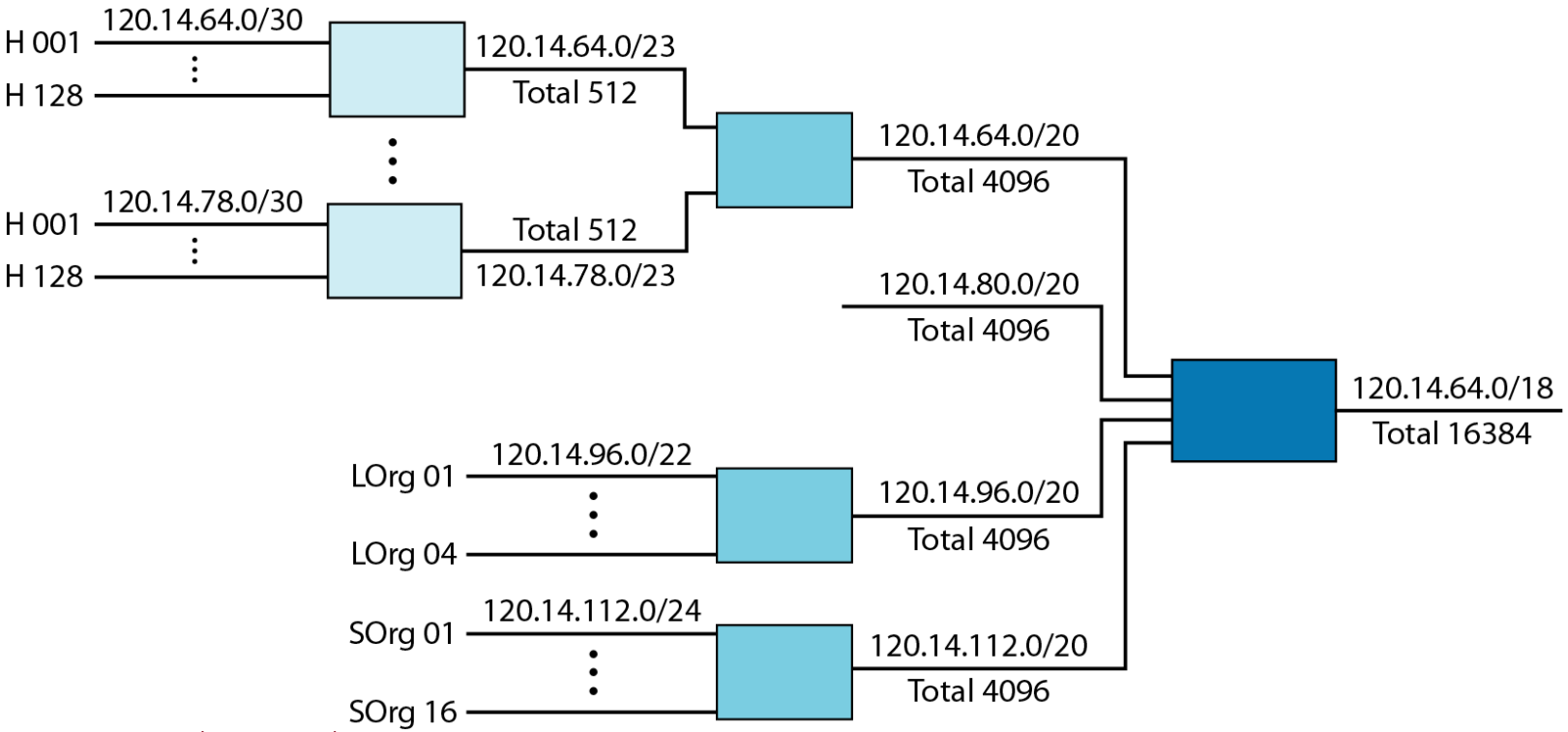
Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2

Hierarchical Routing

- Hierarchical routing can be used to reduce the size of the routing tables
- During introduction we saw the internet hierarchy and different levels if ISPs (international and national ISPs, regional ISPs and local ISPs.) Routing tables can also have a sense of hierarchy to help them have lower sizes
- In general if a ISP is assigned a block of a.b.c.d/n, it can create blocks of e.f.g.h/m where **m is greater than n** and can vary for each customer
- The reason this can reduce the size of routing tables is that the rest of the internet does not need to know about this division, therefore all customers of that certain ISP can be defined as a.b.c.d/n to the rest of the internet, which means there is only one entry in every router in the whole world for all those customers
- Routers inside the certain ISP are the ones that need to recognize the subblocks and route the packet to the destined customer
- In classless routing, levels of hierarchy are limited or unlimited ?

Hierarchical Routing – Example

- A regional ISP is granted 16,384 addresses starting from 120.14.64.0 and has decided to divide this block into four subblocks, each with 4096 addresses. Three of these subblocks are assigned to three local ISPs; the second subblock is reserved for future use. Note that the mask for each block is /20 because the original block with mask /18 is divided into 4 blocks

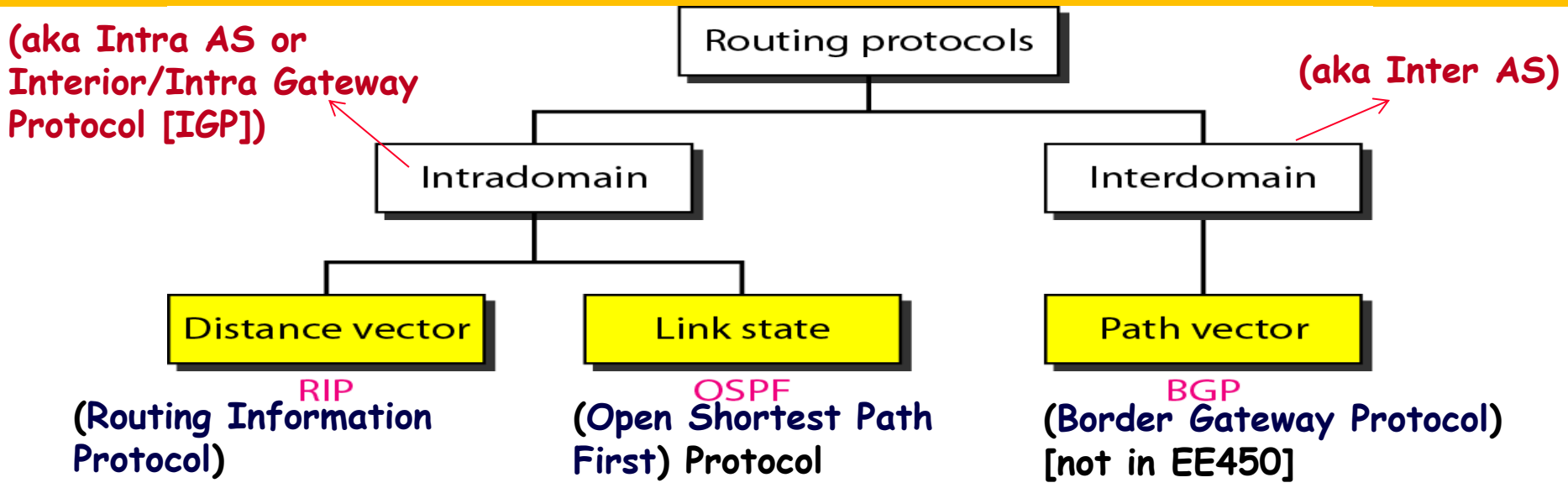


Hierarchical Routing – Example (Cont.)

- The first local ISP has divided its assigned subblock into 8 smaller blocks (that's the reason for mask /23) and assigned each to a small ISP. Each small ISP provides services to 128 households (H001-H0128), each using 4 addresses (that's why we have /30)
- The second local ISP has divided its block into 4 blocks and has assigned the addresses to four large organizations (LOrg01-LOrg04 each with mask /22)
- The third local ISP has divided its block into 16 blocks and assigned each block to a small organization (SOrg1-SOrg16). Each small organization has 256 addresses, and the mask is /24

- There is a sense of hierarchy in this configuration. All routers in the Internet send a packet with destination address 120.14.64.0 to 120.14.127.255 to the regional ISP

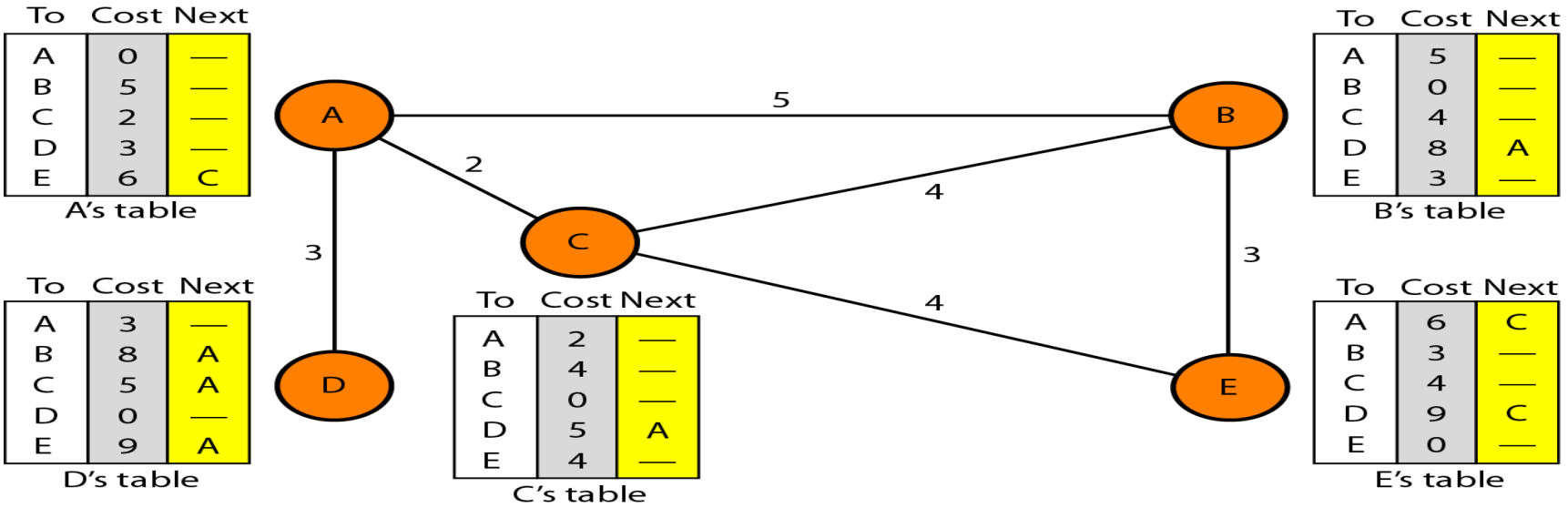
Popular Routing Algorithms



- Intra means within domain (or the autonomous system,) inter means between domains (or autonomous systems)
  - An Autonomous System (AS) is a group of networks and routers under the authority of a single admin, management or organization
  - Internet consists of 1000s of autonomous systems
- We are interested in intradomain routing algorithms

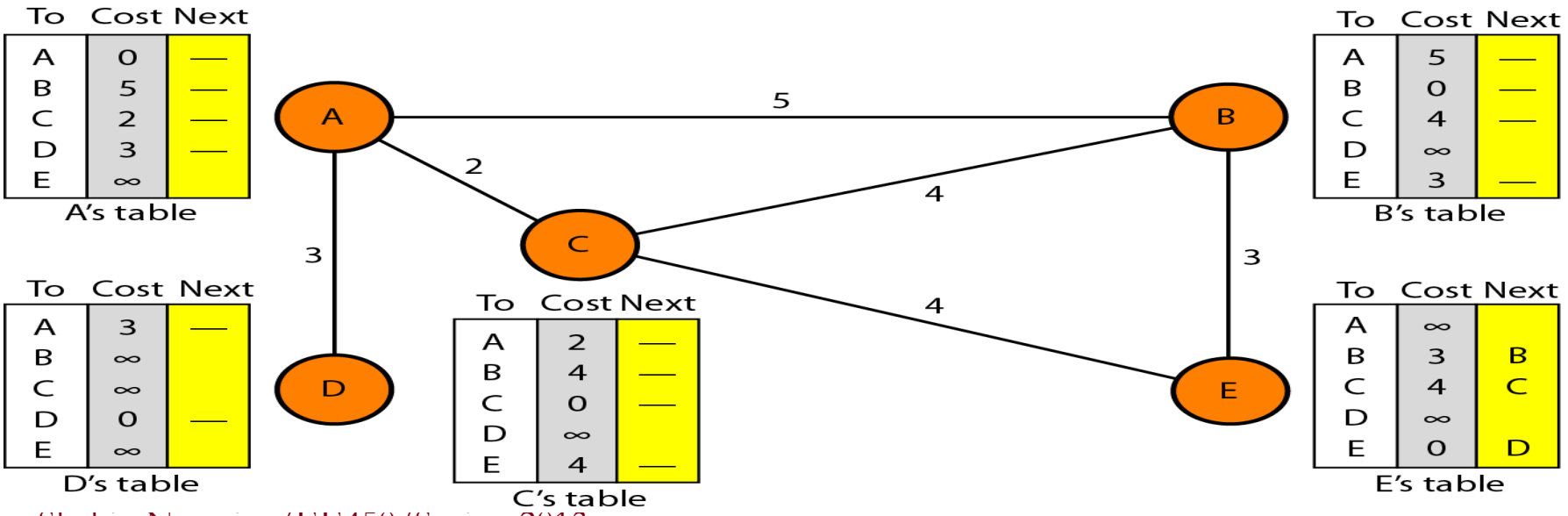
Distance Vector Routing Tables

- A vector is a sequence of entries. A distance vector [used in RIP algorithm] is a vector that shows the distance between that router and every other router in the network that the router knows of
- Router A will send to its neighbors (the routers that A is connected to directly e.g., B, C, and D) its distance vector that has the distance between A and all other routers in the network that A is aware of, not only the neighbors (B, C, and D)
- Every router exchanges its distance vector



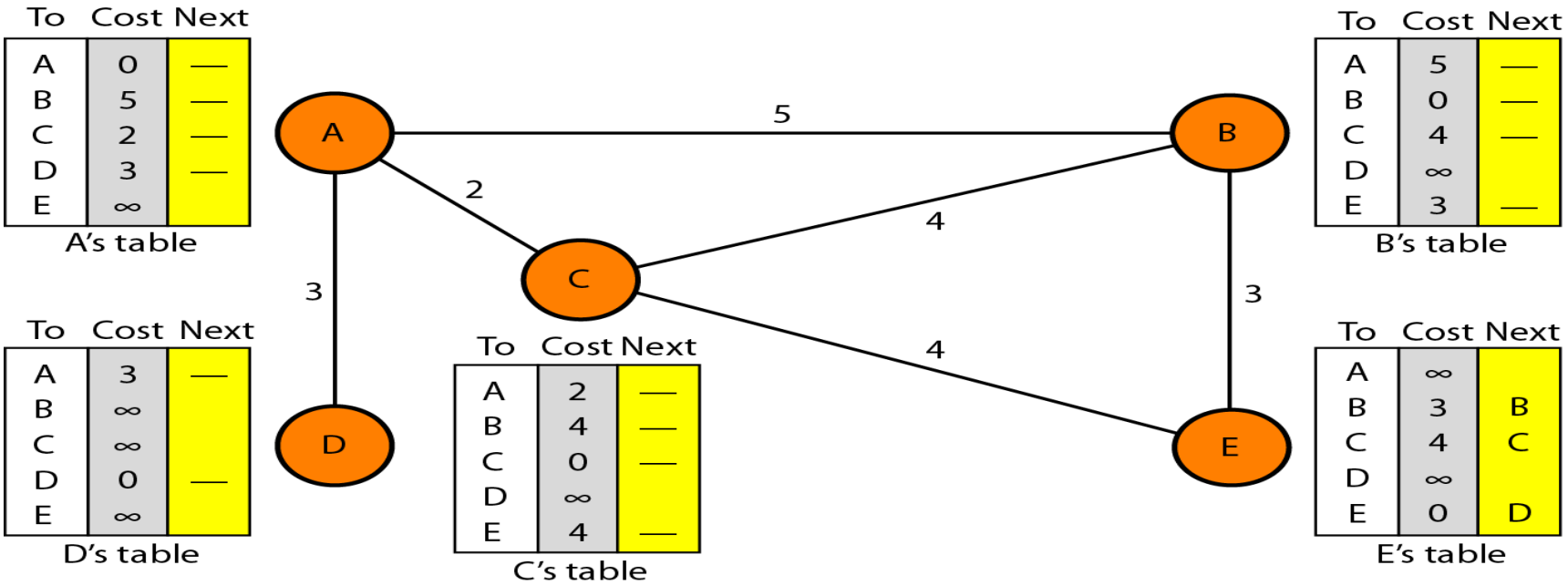
Distance Vector Routing Tables – Initialization

- Initially, each node can know only the distance btn itself and its immediate neighbors, i.e., the ones directly connected to it
- The main idea of distance vector routing is sharing information btn neighbors. Although node A does not know about node E, node C does, so if node C shares its table with A, on the other hand C does not know about D, but A does, therefore nodes A and C can improve their routing tables if they help each other



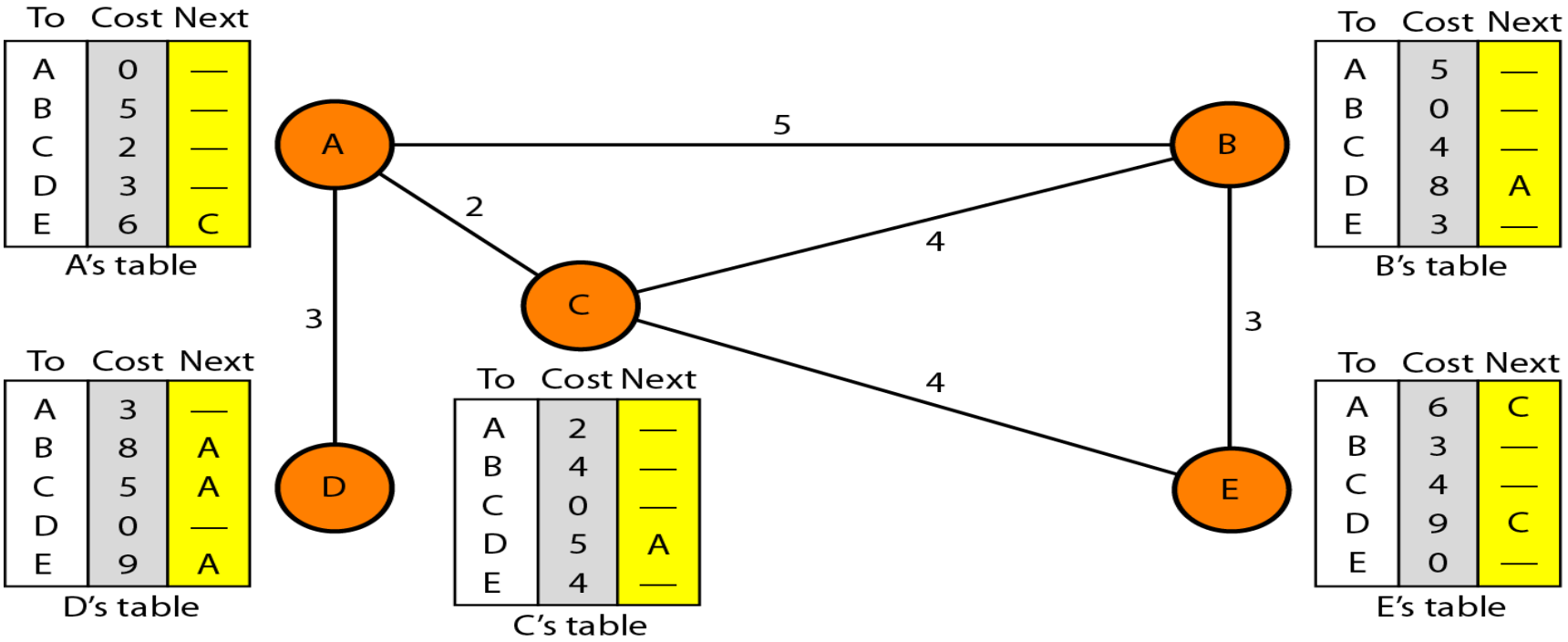
Distance Vector Routing Table – Example

- Router B gets the distance vector from router A. It then compares its own distance vector with that of A
- Router B will find it shorter to reach D through A, so it will update its distance vector
- Router B then exchanges its distance vector with its neighbors including A



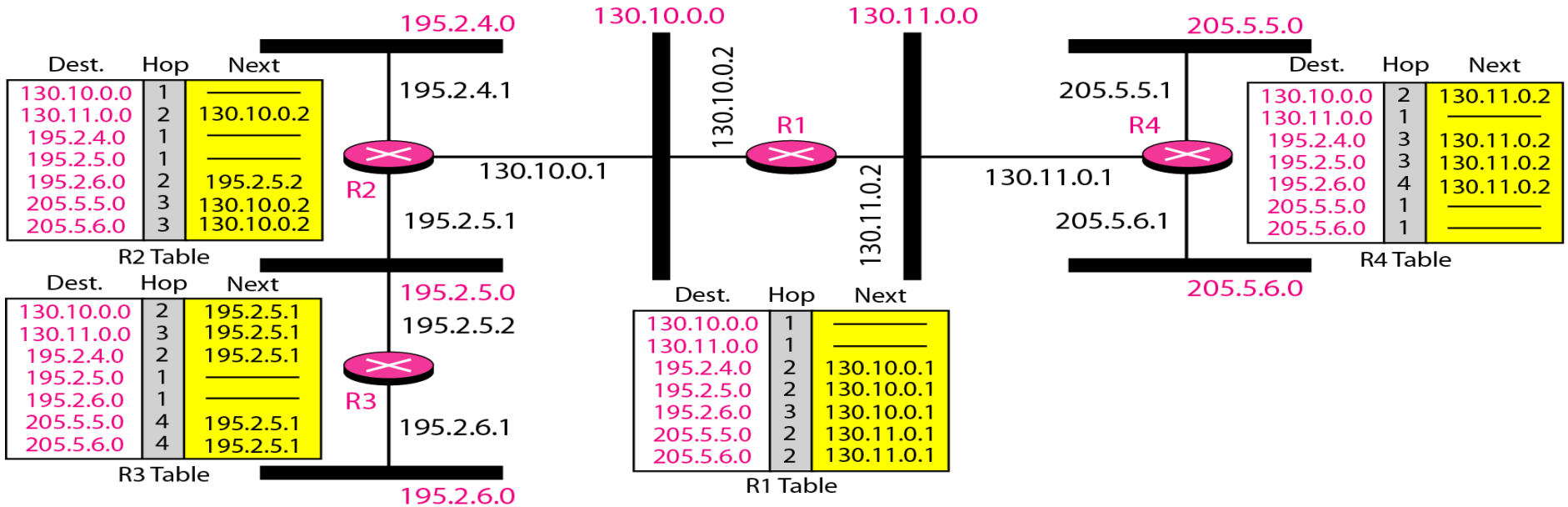
Distance Vector Routing Tables

- Every neighbor when it gets a distance vector from its neighbor, it will update its distance vector
- In distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change



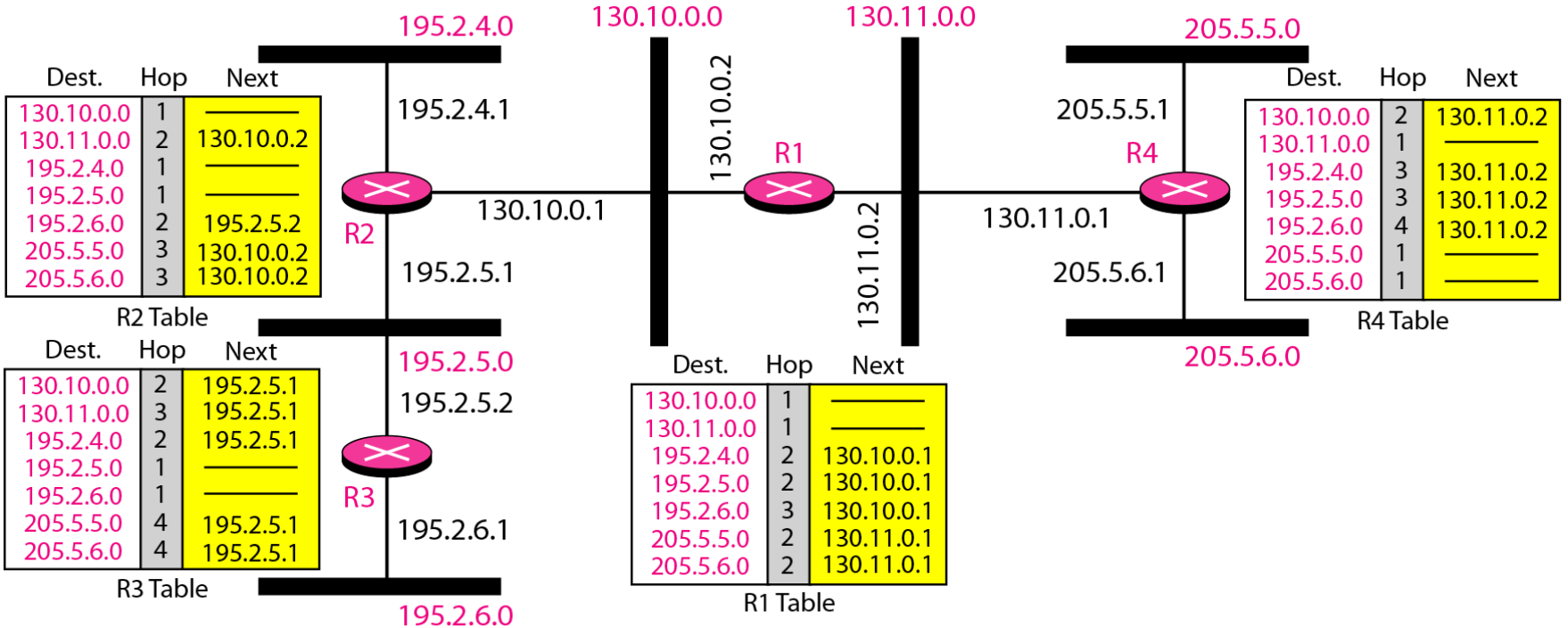
RIP (Routing Information Protocol)

- RIP is a very simple protocol based on distance vector routing
- In an autonomous system, we are dealing with routers and networks (links). Routers have routing tables, networks do not
- The destination in a routing table is a network
- Distance is based on hop count
- Infinity is defined as 16, i.e., any route in an AS using RIP cannot have more than 15 hops



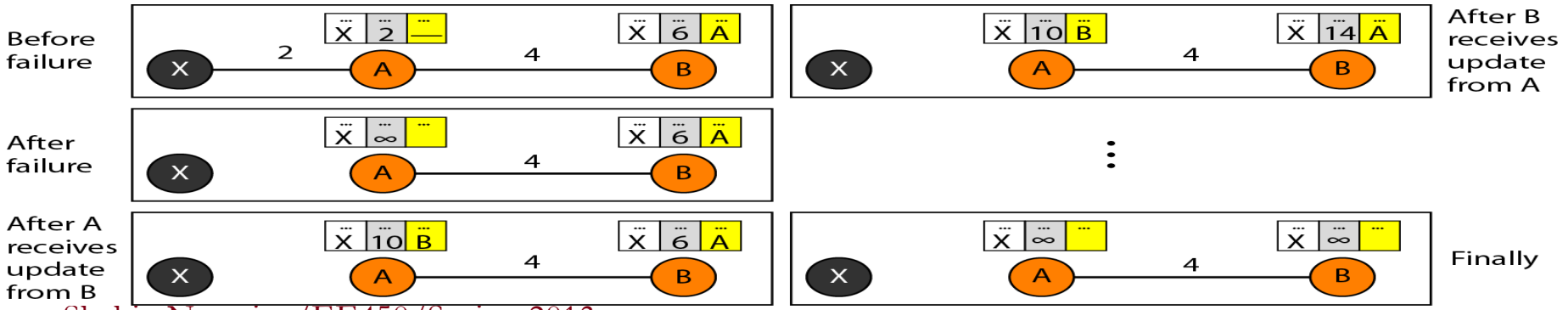
RIP Example

- RIP is not used for big networks. It is because RIP is not scalable
- This example is an AS with 7 networks and 4 routers. Let's focus on R1's table. It has 7 entries, one for each network. R1 is directly connected to networks 130.10.0.0 and 130.11.0.0, so there is no next-hop entries for them



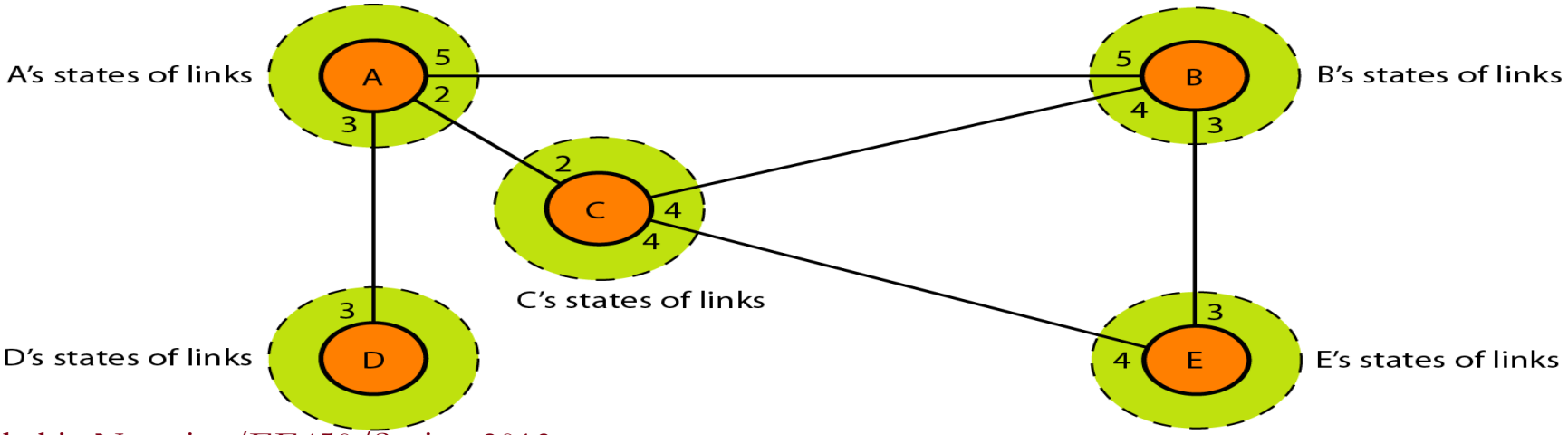
Instability Problem in RIP

- If A sends the packet, the packet goes back and forth, i.e., bounces and never reaches X
- One solution to instability problem is that B mentions that it can reach X through A itself (instead of sending one column, it sends another column, next hop as well) but that means more routing information over the link and more waste of capacity
- Another solution: redefine infinity, e.g., to 16, but this means, RIP cannot be used for large networks



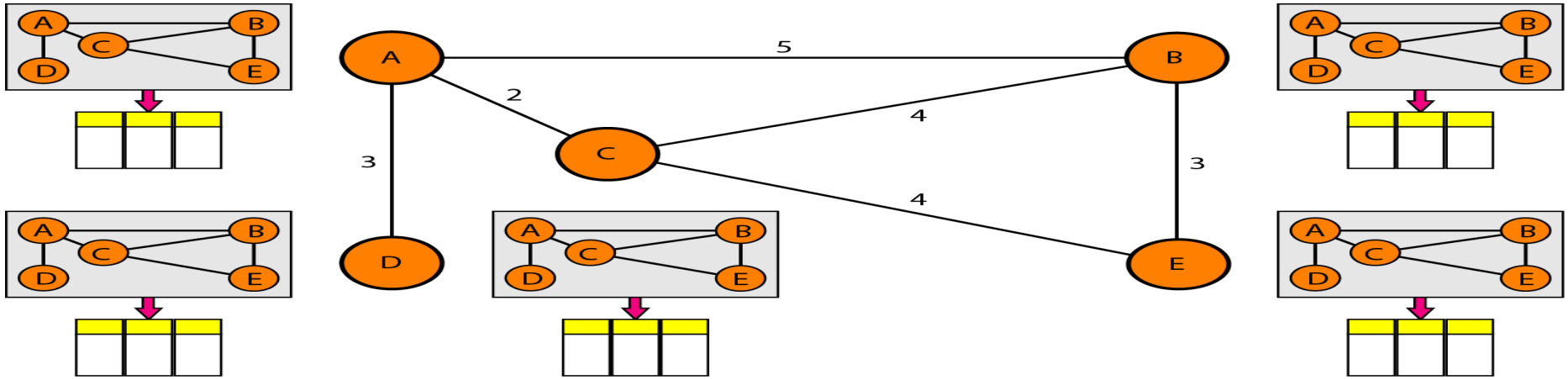
Link State for OSPF Protocol

- This example is the same as the original example we used to present distance vector table for the RIP algorithm
- The information is just about the states of its links, not the distance to other routers
- For example, A will broadcast to all routers a **Link State Packet** (**LSP**) saying it has 3 links: 5 to B, 2 to C, and 3 to D
- All routers will receive this information, some (such as B) directly and some (such as E) through other routers



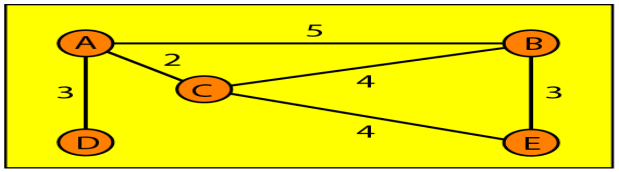
Link State (Cont.)

- All routers will eventually know that A has 3 links and their status is fine (not failed,) they also know the distance of each link (5 to B, 2 to C and 3 to D)
- This means all routers can sketch the diagram (map) of the whole network. Each router has all the information to create a map for itself and does not share the map with others, because every router knows about it; therefore every router implements the routing algorithm and creates the map for itself
- OSPF is more scalable than RIP; in terms of network growth



Link State Routing – Dijkstra's Algorithm

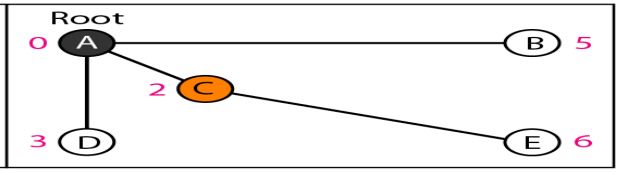
- The amount of information exchange is less than that in RIP algorithm. LS is scalable
- If there is a change LSPs are generated; also periodically, but with longer periods than those in RIP
- Usually there is a timer that expires and then exchange of information should be done, even if the there is no change of state, because there may be new routers that were added to the network and they don't know about the state of your links



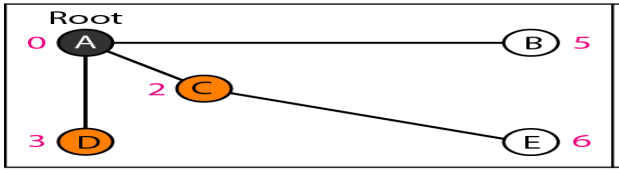
Topology

1. Set root to A and move A to tentative list.

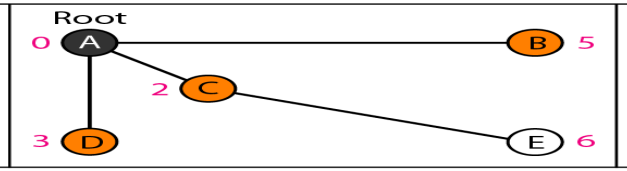
2. Move A to permanent list and add B, C, and D to tentative list.



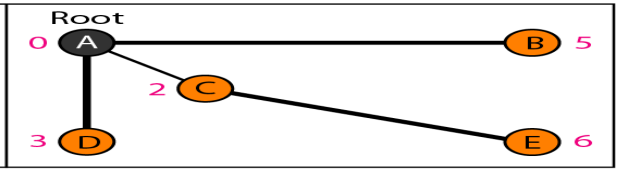
3. Move C to permanent and add E to tentative list.



4. Move D to permanent list.



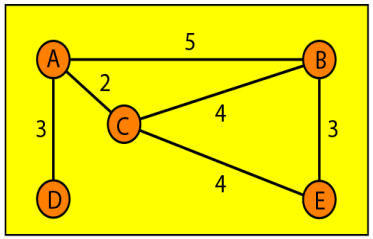
5. Move B to permanent list.



6. Move E to permanent list (tentative list is empty).

Link State Routing – Dijkstra's (Cont.)

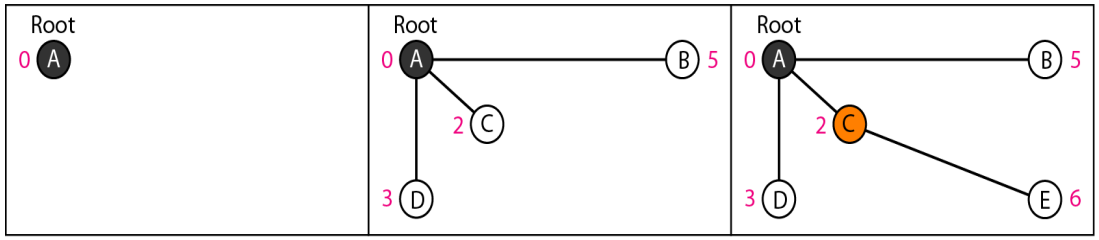
- The map is identical for all routers, however the trees are not. Each router will have its own tree
- **Dijkstra's algorithm** is a single-source shortest path [spanning tree, i.e., no loop] algorithm, which is preformed by each router after it has received all LSPs and created the map



Topology

Routing table in A

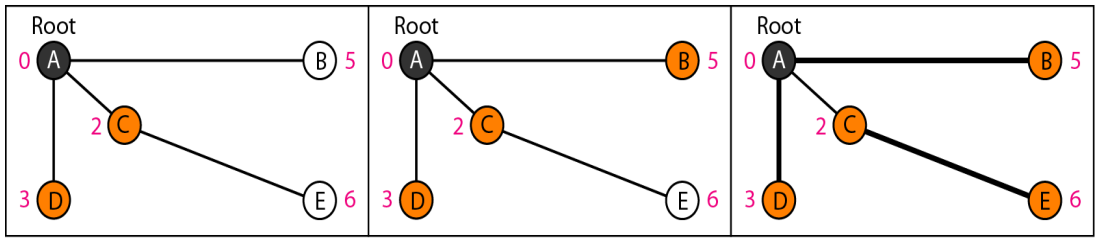
Node	*Cost*	*Next Router*
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C



1. Set root to A and move A to tentative list.

2. Move A to permanent list and add B, C, and D to tentative list.

3. Move C to permanent and add E to tentative list.



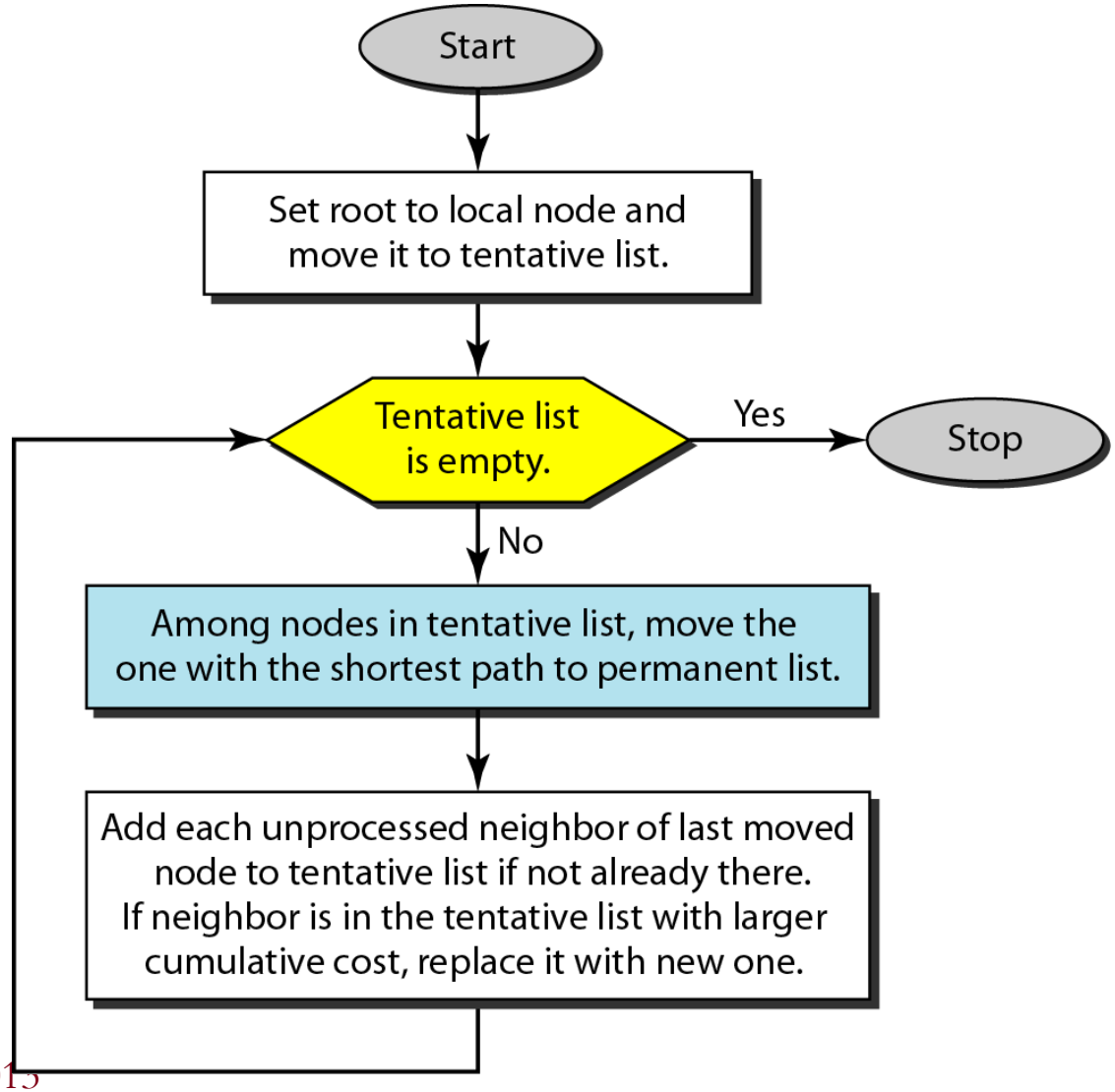
4. Move D to permanent list.

5. Move B to permanent list.

6. Move E to permanent list (tentative list is empty).

Dijkstra's Algorithm – Flowchart

- **The flowchart is short and simple which means Dijkstra's algorithm is simple**



Dijkstra's Algorithm – Notation

Dijkstra's algorithm

- Net topology, link costs known to all nodes
  - Accomplished via "link state broadcast"
  - All nodes have same info
- Computes least cost paths from one node ('source") to all other nodes
  - Gives forwarding table for that node
- Iterative: after k iterations, know least cost path to k dest.'s

Notation:

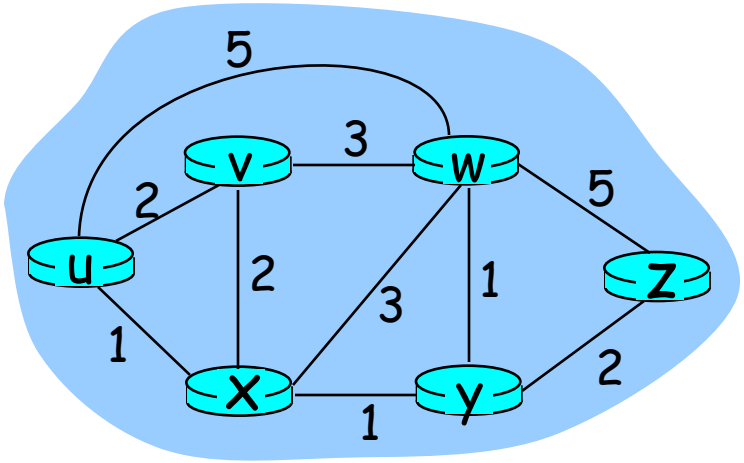
- **c(x,y):** link cost from node x to y; = ∞ if not direct neighbors
- **D(v):** current value of cost of path from source to dest. v
- **p(v):** predecessor node along path from source to v
- **N':** set of nodes whose least cost path definitively known

Dijkstra's Algorithm – Pseudocode

```
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'
```

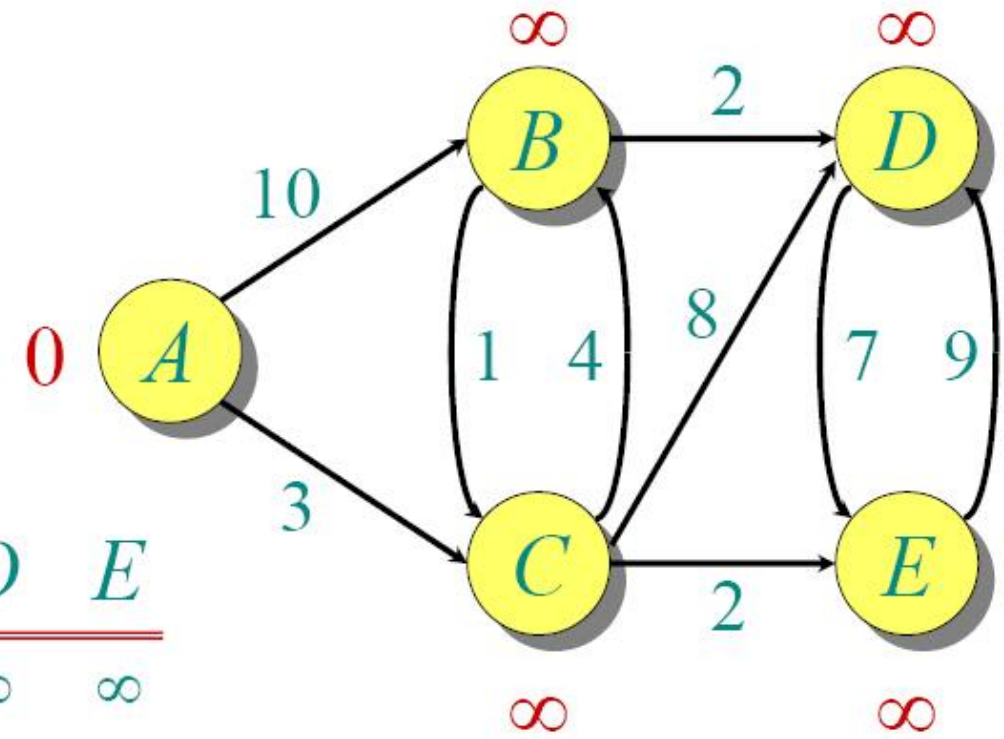
Dijkstra's Algorithm: Example I

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					



Dijkstra's Algorithm: Example II

Initialize:

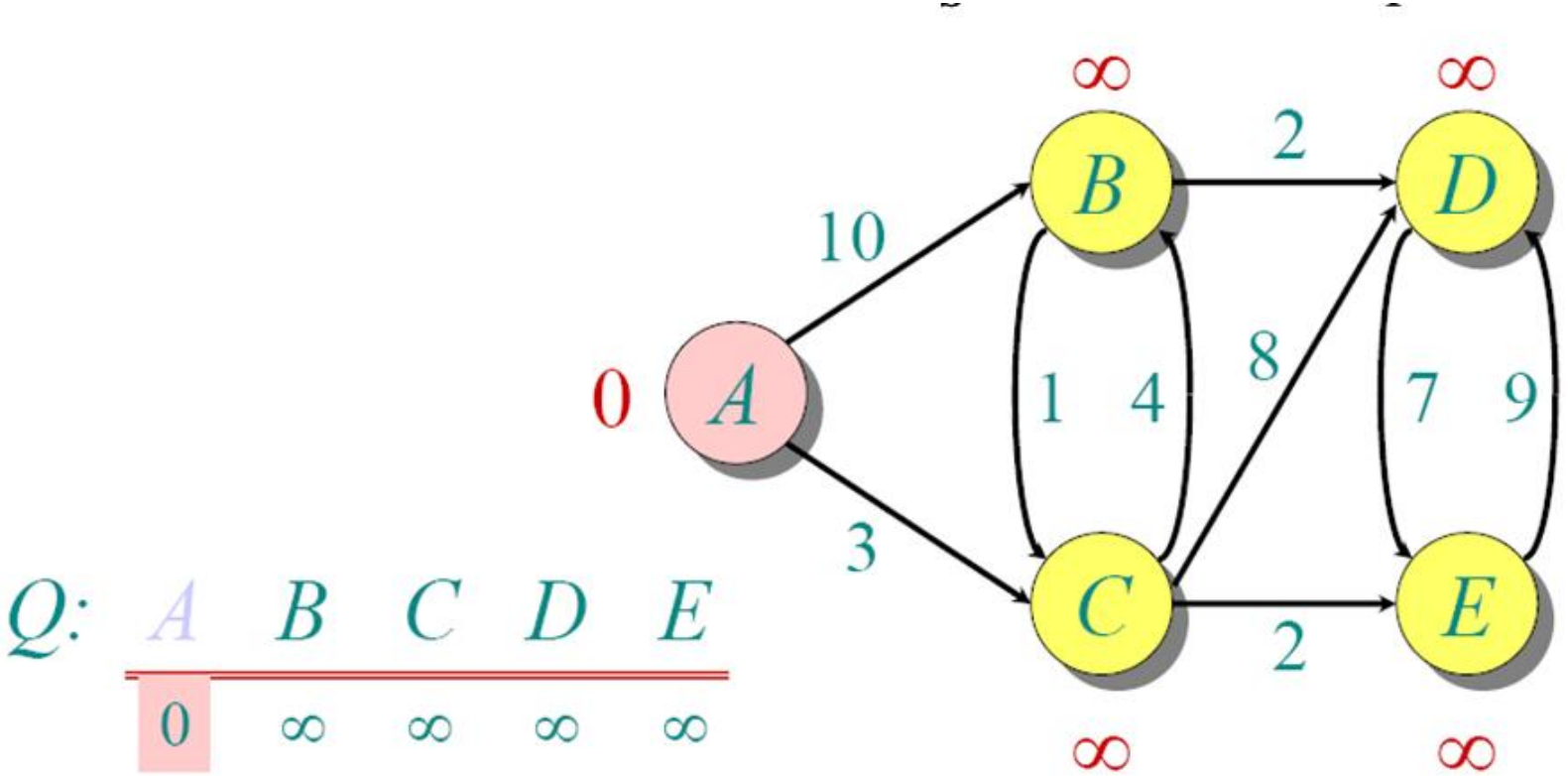


Q:	A	B	C	D	E
	0	∞	∞	∞	∞

S: {}

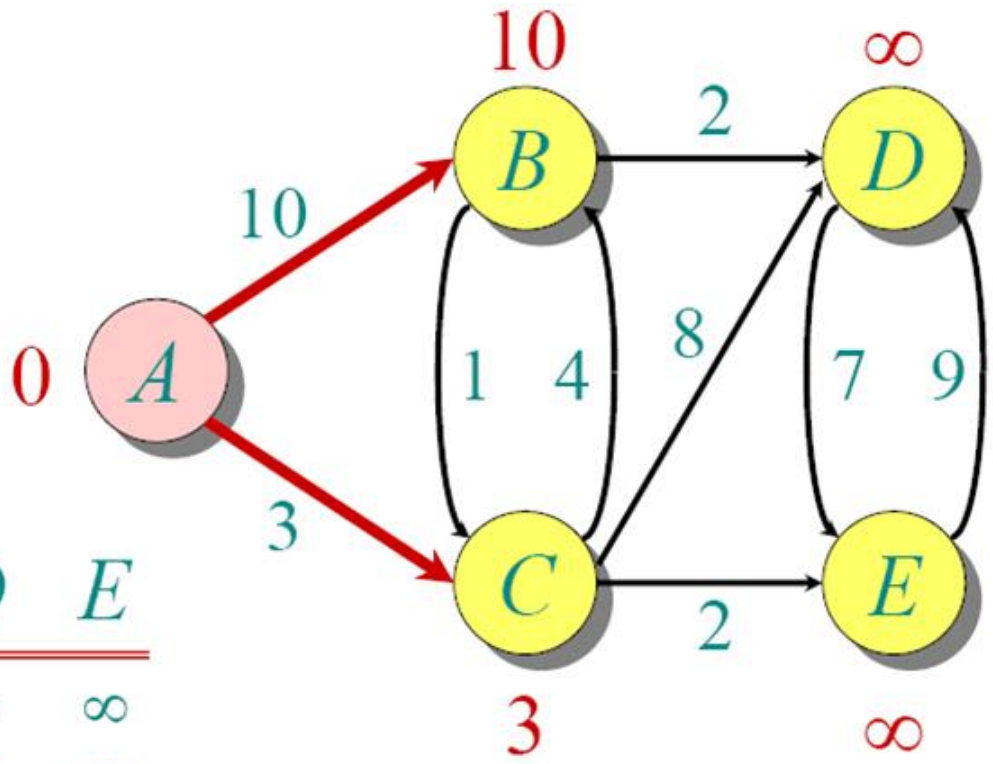
By Laksman Veeravagu and Luis Barrera, UT Austin

Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞

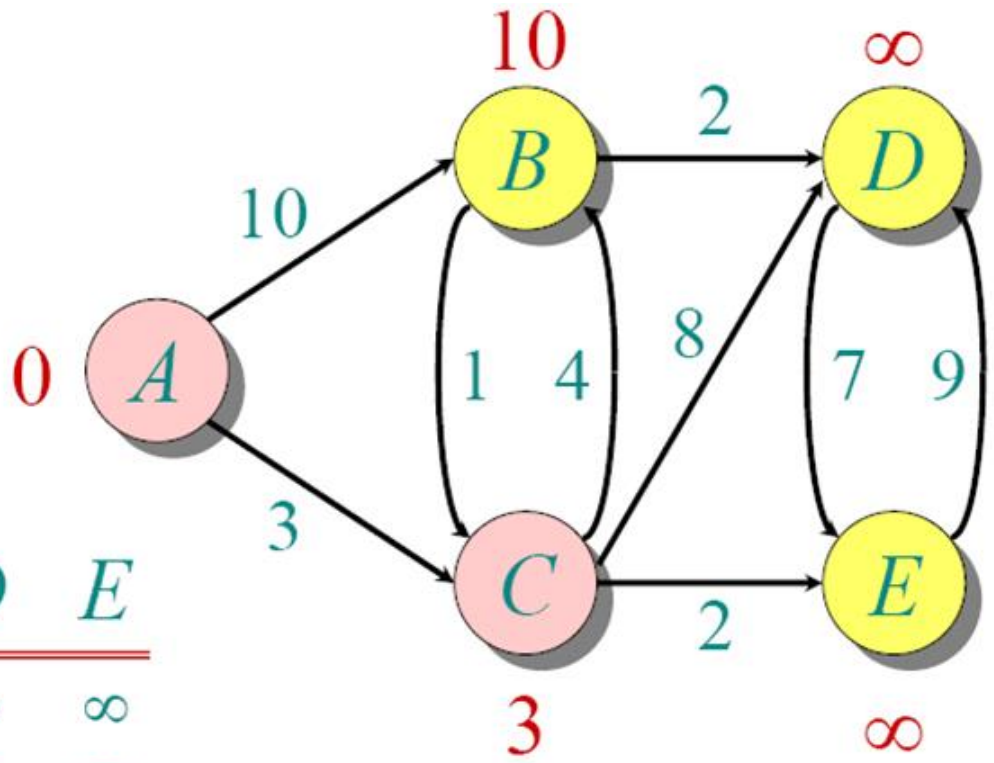
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞

$S: \{ A \}$

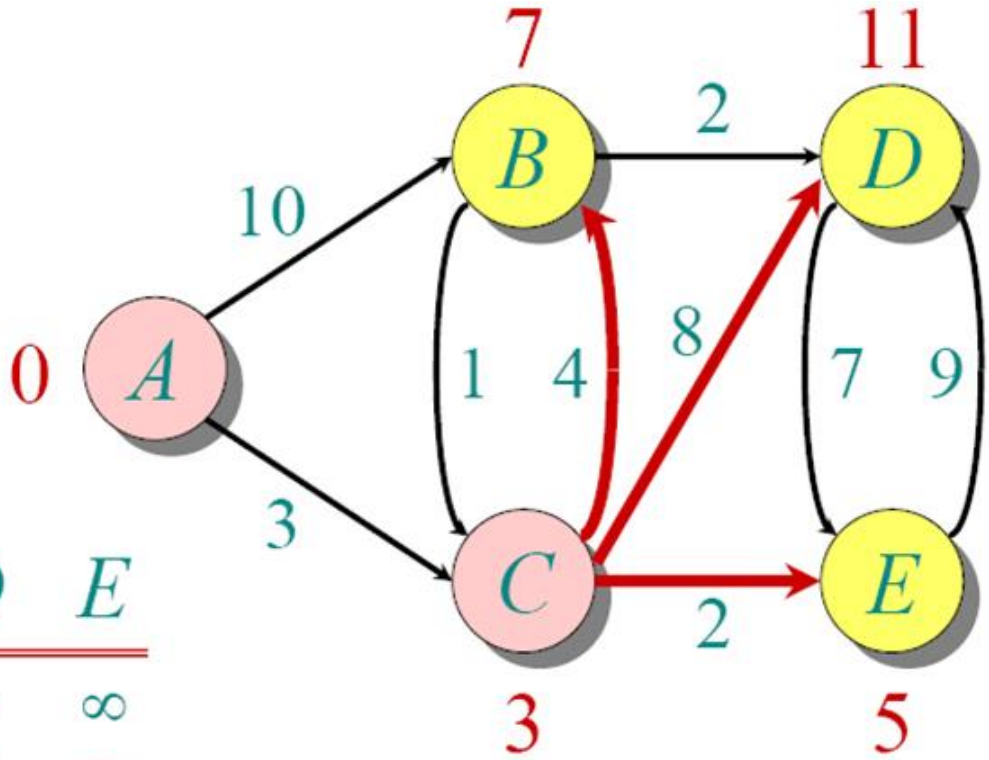
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞

$S: \{ A, C \}$

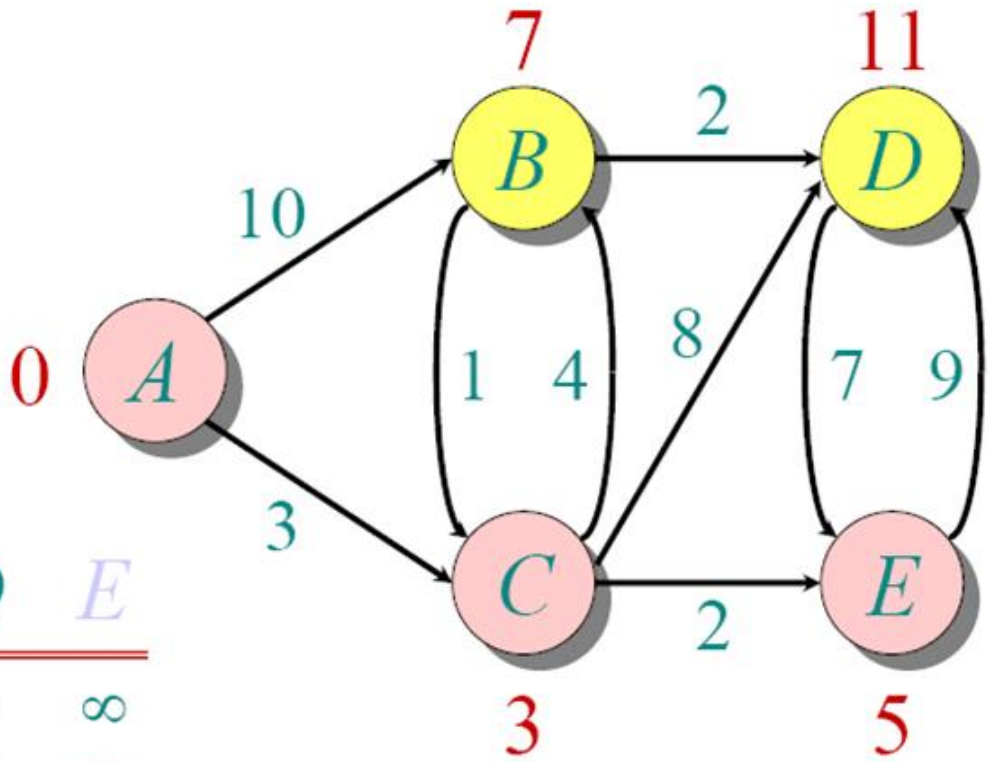
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5

$S: \{ A, C \}$

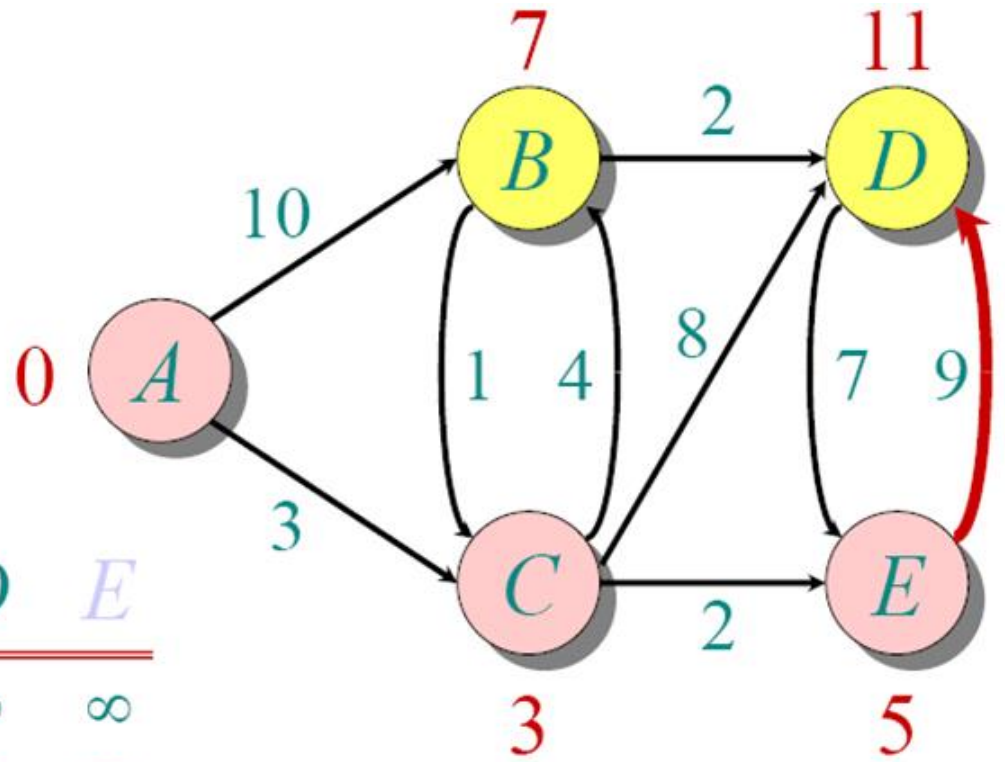
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5

$S: \{ A, C, E \}$

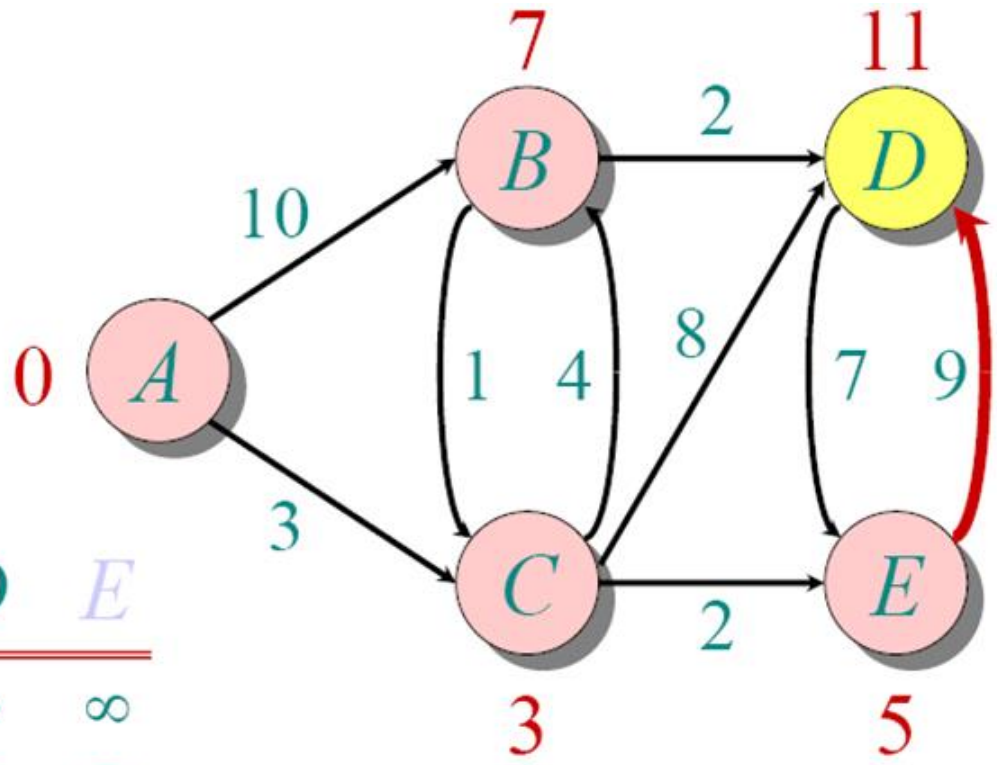
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	

$S: \{ A, C, E \}$

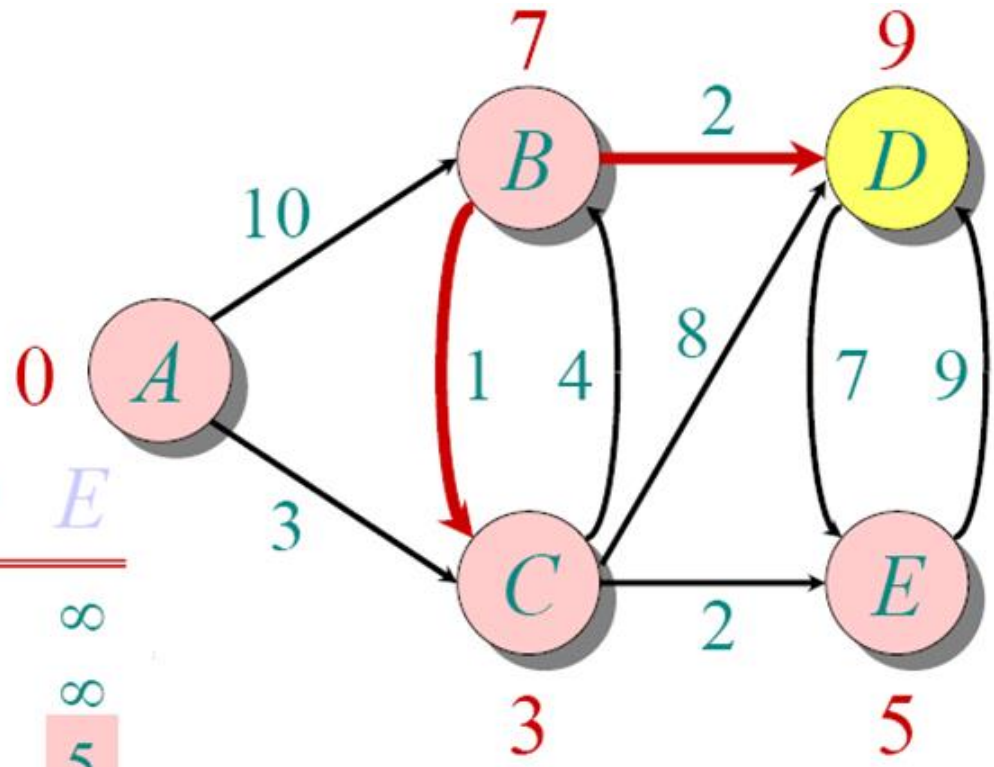
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	

$S: \{ A, C, E, B \}$

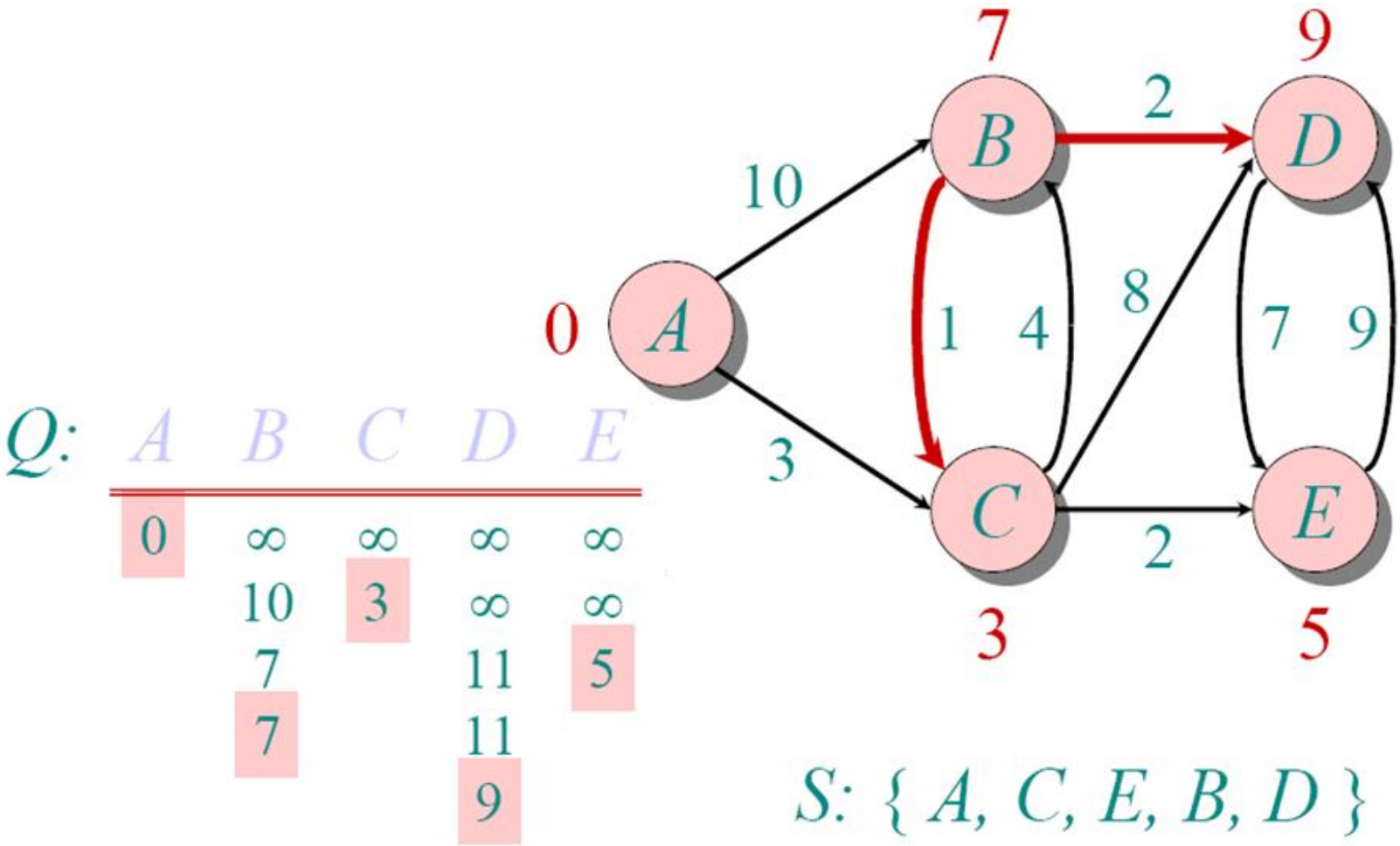
Dijkstra's Algorithm: Example II (Cont.)



Q:	A	B	C	D	E
	0	∞	∞	∞	∞
		10	3	∞	∞
		7		11	5
		7		11	
				9	

$S: \{ A, C, E, B \}$

Dijkstra's Algorithm: Example II (Cont.)

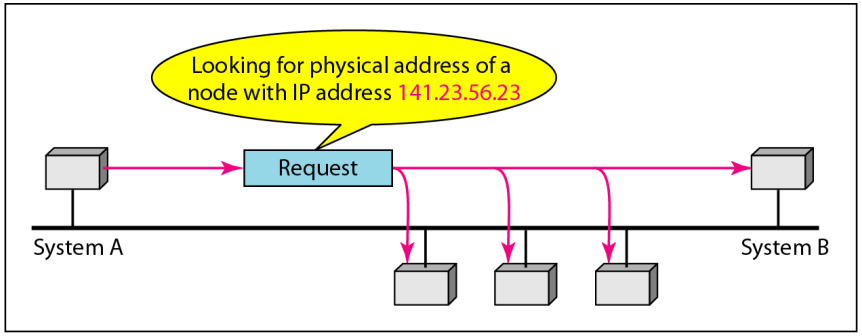


Dijkstra's Algorithm: Example III

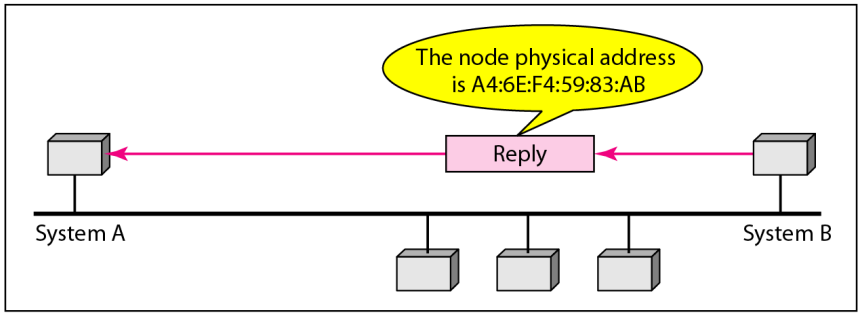
- http://optlab-server.sce.carleton.ca/POAnimations2007/DijkstrasAlgo.html

Address Mapping – Logical to Physical Address: ARP

- Any time a host or router has a packet to send to another host or router it needs the IP address of the receiver that it gets from DNS. In addition the packet needs to be encapsulated into a frame to be able to pass through physical network



a. ARP request is broadcast



b. ARP reply is unicast

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

- There are occasions in which a host knows its physical address but needs to know its logical address. This may happen in two cases:
- A diskless station is just booted. The station can find its physical address by checking its interface but it does not know its IP address
- An organization does not have enough IP addresses to assign to each station; it needs to assign IP address on demand. The station can send its physical address and ask for a short time lease
- **RARP** (**Reverse Address Resolution Protocol**) finds the logical address for a machine that knows only its physical address. A diskless machine is usually booted from ROM, which has minimum booting information. The ROM is installed by the manufacturer and cannot include the IP address. The machine can read its physical address, e.g. by reading its NIC. It can then use the physical address to get the logical address by using RARP

RARP (Cont.)

- An RARP request is created and broadcasted on the local network. Another machine on the local network that knows all the IP addresses will respond with a RARP reply. The requesting machine must be running a RARP client program; the responding machine must be running a RARP sever program
- The main issue with RARP is that broadcasting is done at the data link layer and the physical broadcast address of all 1s in case of Ethernet does not pass the boundaries of a network. This means an admin of several networks or subnets, needs to assign a RARP server in each network or subnet. This is the reason RARP is almost obsolete. Two protocols BOOTP and DHCP are replacing RARP

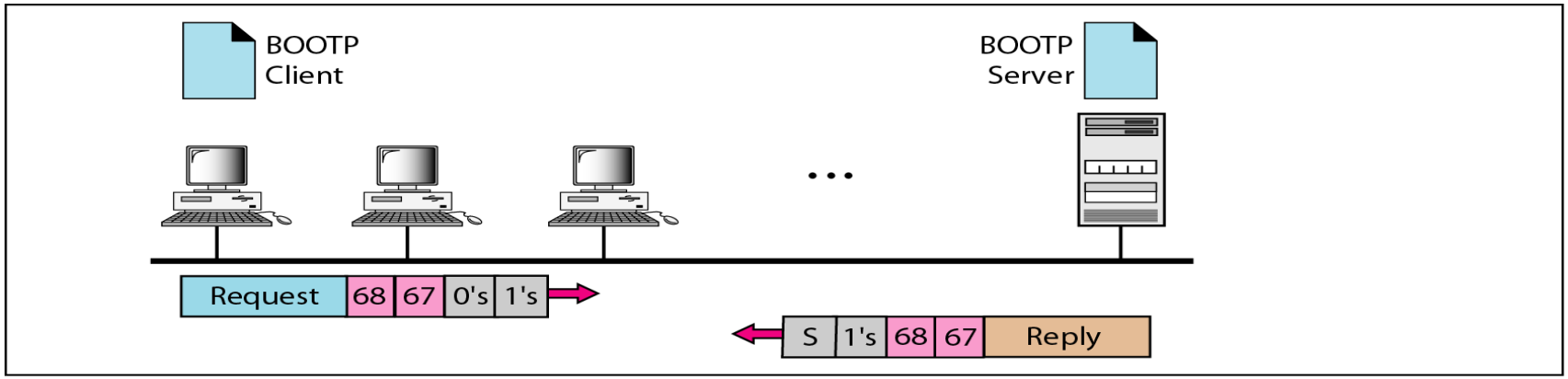
BOOTP (Bootstrap Protocol)

- BOOTP is an application layer client-server protocol designed to provide a physical to logical address mapping. The network admin may put the client and server on the same network or on different networks
- BOOTP messages are encapsulated in a UDP datagram and UDP datagram itself is encapsulated in an IP packet
- The client uses all 0s as the source address and all 1s as the destination address in the IP packet (as it knows neither its IP address nor server's IP address)
- One advantage of BOOTP over RARP is that client and server are application layer processes, a client can be in one network and the server in another, separated by several other networks

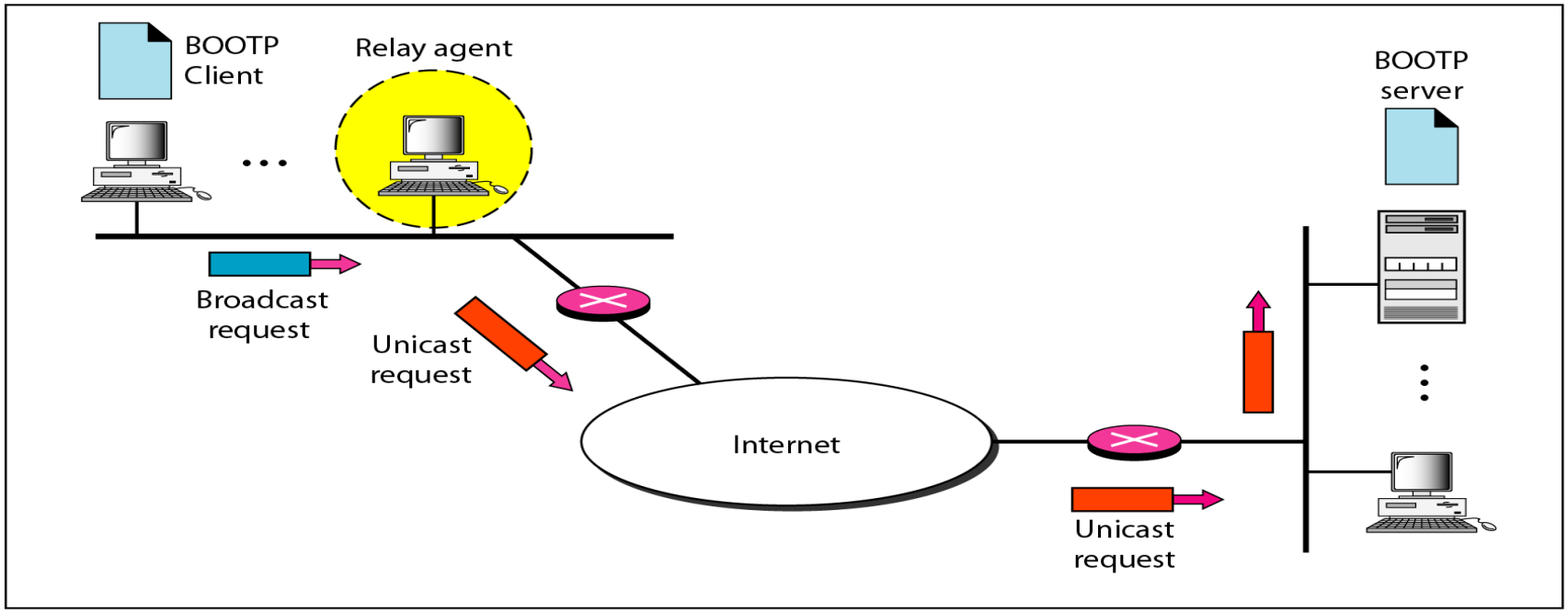
BOOTP (Cont.)

- One issue is that BOOTP broadcast IP packet cannot pass through any router. To resolve this issue, one of the hosts or a router aka gateway that can be configured to operate at the application layer, can be used as a relay. That host or router is then called a relay agent. The relay agent knows the unicast address of a BOOTP server. When it receives this type of packet, it encapsulates the message in a unicast packet and sends the request to the BOOTP server
- The packet carrying a unicast destination address is routed by any router and reaches the BOOTP server. The BOOTP server knows the message comes from a relay agent because one of the fields in the request message defines the IP address of the relay agent. The relay agent after receiving the reply sends it to the BOOTP client

BOOTP – Client/Server Networks



a. Client and server on the same network

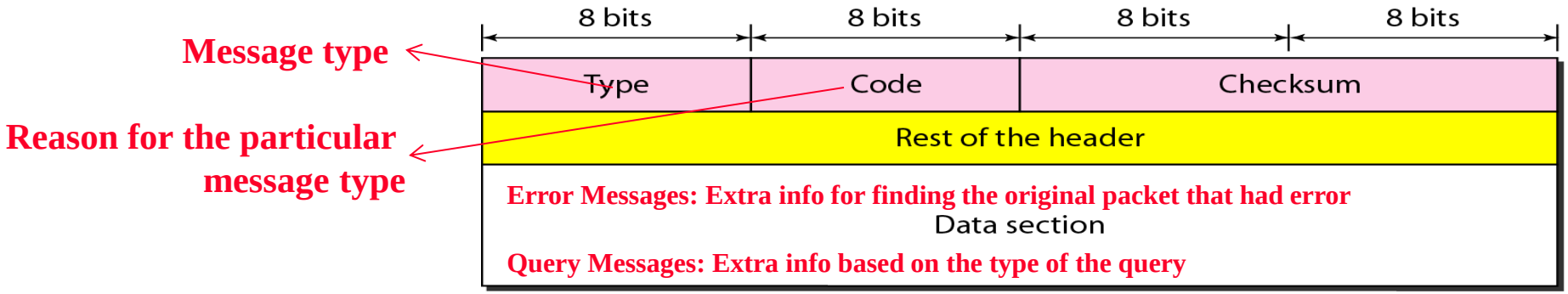


b. Client and server on different networks

ICMP (Internet Control Message Protocol)

- What happens if a router must discard a packet because it cannot find a router to the final destination or because TTL has a zero value? Or if the final destination host must discard all fragments of a packet because it has not received all fragments within a predetermined time limit? These are some examples where an error has occurred and IP has no built-in mechanism to notify the original host
- IP also lacks a mechanism for host and management queries, examples: a host may need to determine if a router or another host is alive, sometimes a network admin needs info form another host or router
- ICMP has been designed to compensate for above two deficiencies. It is a companion to IP protocol

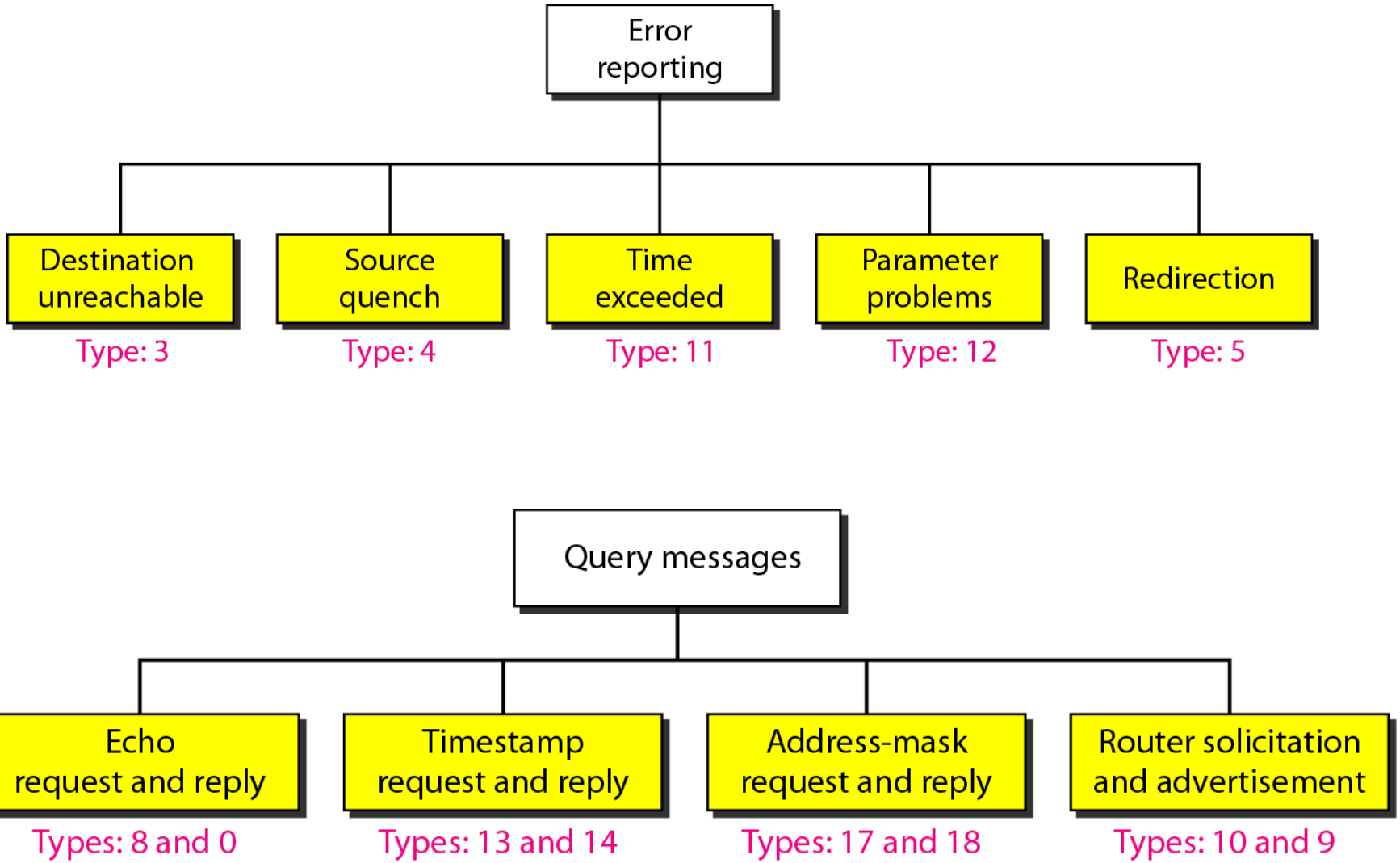
ICMP – Format and Messages



General format of ICMP messages

- ICMP messages are either **error-reporting (always to the original source)** or **query** messages
- Note: No ICMP error message will be generated:
  - In response to a datagram carrying an ICMP error message
  - For a fragmented datagram that is not the first fragment
  - For a datagram having a multicast address
  - For a datagram having a special address such as 127.0.0.0 or 0.0.0.0

ICMP – Format and Messages



ICMP – Some of Descriptions

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

IGMP (Internet Group Management Protocol)

- Both unicasting (one-to-one) and **multicasting** (one-to-many) can be applied in IP

- Multicasting has many applications, Examples: multiple stockbrokers can simultaneously be informed of stock price change or travel agents be informed of a plane cancellation, also distance education, and video on demand

- IGMP is one of the necessary but not sufficient protocols involved in multicasting. It is a companion to IP

Features of IPv6

- Larger Address Space
- Aggregation-based address hierarchy
  - Efficient backbone routing
- Efficient and Extensible IP datagram
- Stateless Address Autoconfiguration
- Security (IPsec mandatory)
- Mobility

Why IP6?

- Ipv4 address space is becoming depleted
- Routing tables in the Internet core routers have become very large
- Explosion in the number of addressable devices (think cell phones)
- India and China are increasing pressure
- Reliance on private addresses and NAT causes a continuous level of pain for network administrators and application developers

128-bit IPv6 Address

3FFE:085B:1F1F:0000:0000:0000:00A9:1234

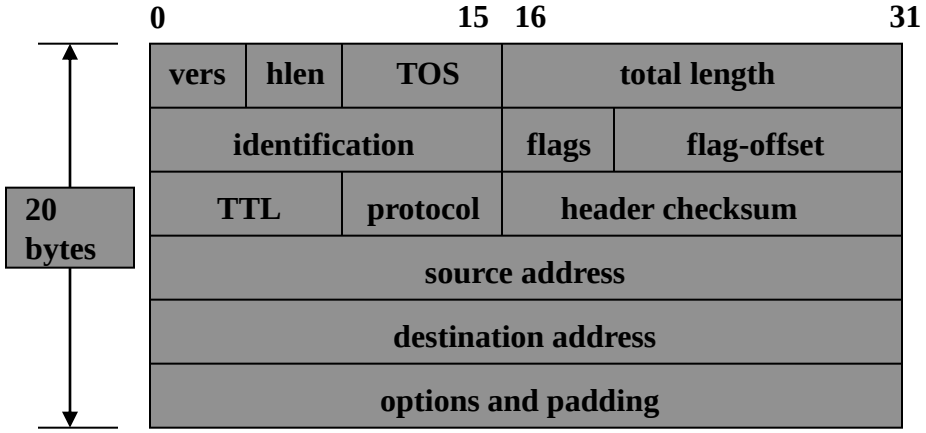
8 groups of 16-bit hexadecimal numbers separated by "**:**"

Leading zeros can be removed

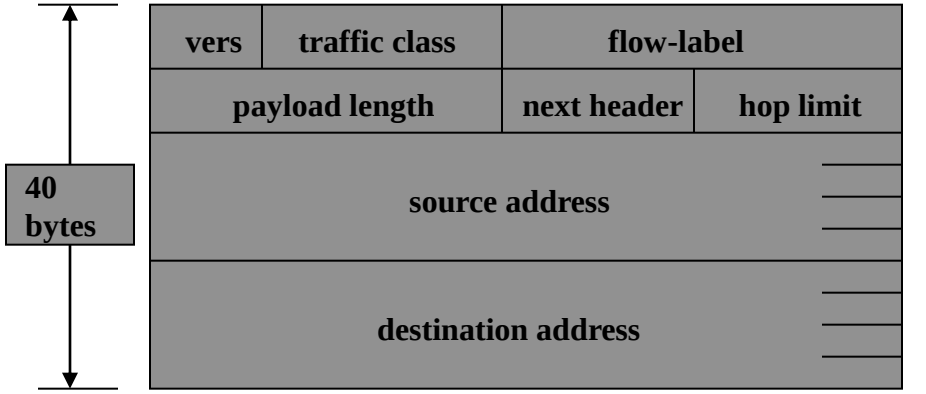
3FFE:85B:1F1F::A9:1234

:: = all zeros in one or more group of 16-bit hexadecimal numbers

Header Comparison



IPv4



IPv6

Removed (6)

- ID, flags, flag offset
- TOS, hlen
- header checksum

Changed (3)

- total length => payload
- protocol => next header
- TTL => hop limit

Added (2)

- traffic class
- flow label

Expanded

- address 32 to 128 bits

Major Improvements of IPv6 Header

- **No option field**: Replaced by extension header.
  - Results in a fixed length, 40-byte IP header.
- **No header checksum**: Result in fast processing.
- **No fragmentation at intermediate nodes**: Result in fast IP forwarding

Extension Headers

- Routing – Extended routing, like IPv4 loose list of routers to visit
- Fragmentation – Fragmentation and reassembly
- Authentication – Integrity and authentication, security
- Encapsulation – Confidentiality
- Hop-by-Hop Option – Special options that require hop-by-hop processing
- Destination Options – Optional information to be examined by the destination node

Stateless Address Autoconfiguration

- 3 ways to configure network interfaces: Manually, Stateful, Stateless
- IPSAA→ IPv6 addr. Separated into 2 parts: network and interface id
- Link- local addresses: prefix FE80::0 + interface identifier (EUI-64 format)
- Obtain network id through Router solicitation (RS)

Stateless Address Autoconfiguration (Cont.)

- **Phase 1:** An identifier is generated, supposedly on the link
  - by using the EUI-48 MAC address
- **Phase 2:** A tentative address is built: by following mechanism
  - Prefix advertisement
  - Router advertisement and Router solicitation
- **Phase 3**: The uniqueness of this address on the link is verified
  - By Duplicate Address Detection (DAD) mechanism
- **Phase 4:** If unique, the address from phase 2 is assigned to the interface

Stateless Address Autoconfiguration (Cont.)

- Prefix Advertisement
  - Advertises prefixes & parameters on a local link. This information is used by IPv6 nodes to configure their IPv6 addresses
- Duplicate Address Detection (DAD)
  - Ensures that each IPv6 address configured on an interface using stateless auto configuration is unique
- Prefix renumbering
  - Advertises modified prefixes or new prefixes and parameters the local link to renumber a prefix already advertised.

Conclusion

- IPv6 is NEW …
  - built on the experiences learned from IPv4
  - new features
  - large address space
  - new efficient header
  - autoconfiguration
- … and OLD
  - still IP
  - build on a solid base
  - started in 1995, a lot of implementations and tests done