

Introduction to Computer Networks

Slide Source: Cisco Networking

Networking Today

- Network has no boundary and supports the way we:
 - Learn
 - Communicate
 - Work
 - Play



LANs and WANs

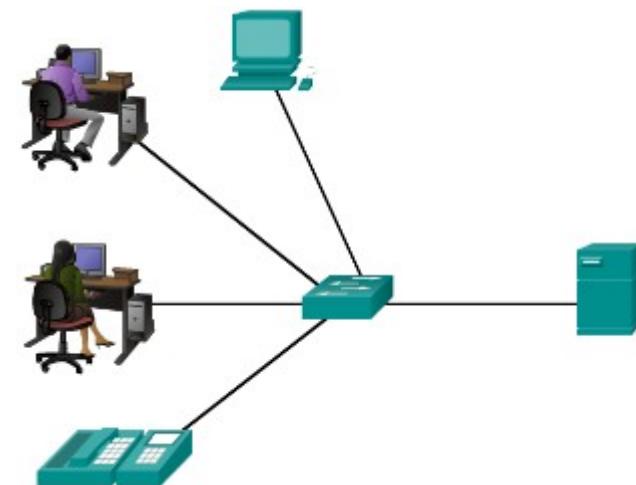
■ Local Area Networks

- Spans across small geographical area
- Interconnects end devices
- Administrated by a single organization
- Provide high speed bandwidth to internal devices

■ WAN Area Networks

- Interconnects LAN
- Administrated by multiple service providers
- Provide slower speed links between LANS

■ Can you name more network types?

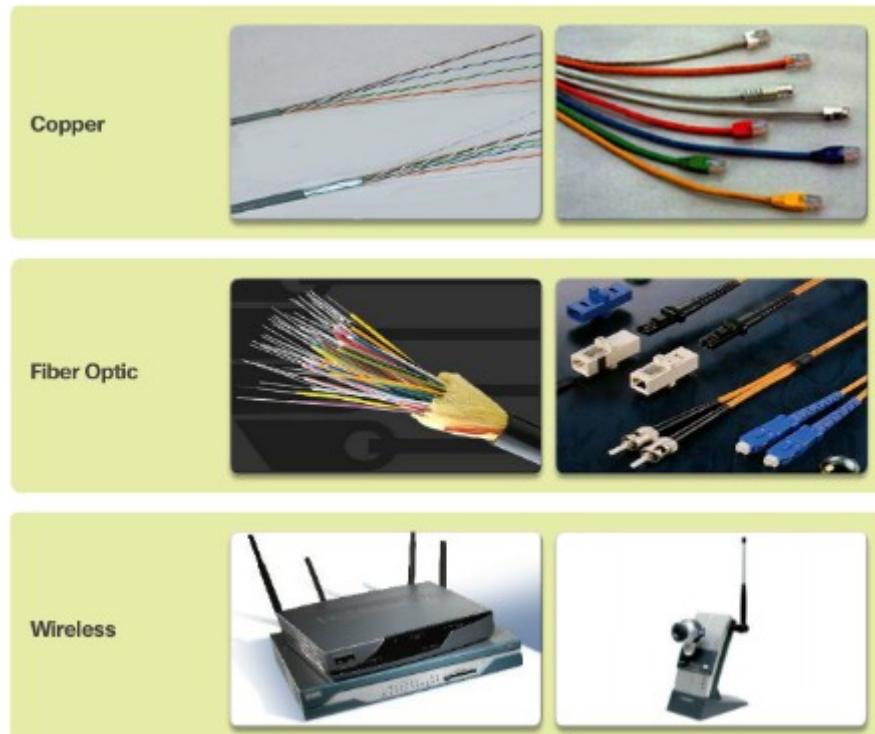


Network Components

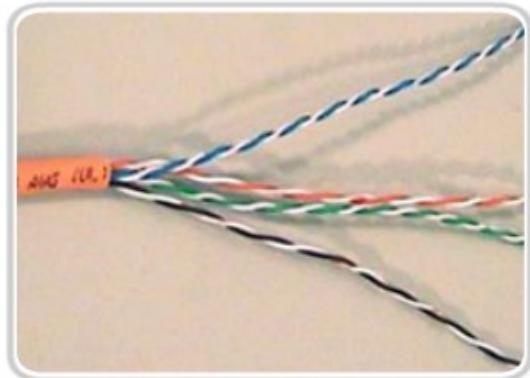


Network Components

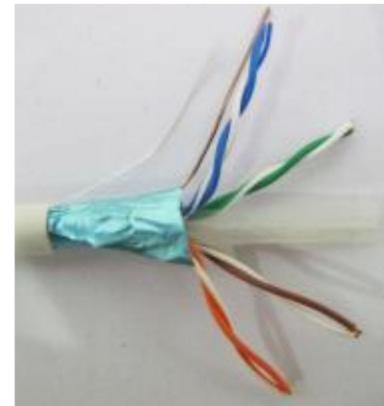
- End Devices
 - Either the source or destination of a message
- Intermediary Network Devices
 - Connect multiple individual networks to form an internetwork
 - Connect the individual end devices to the network
- Network Media
 - Provide the pathway for data transmission
 - Interconnect devices



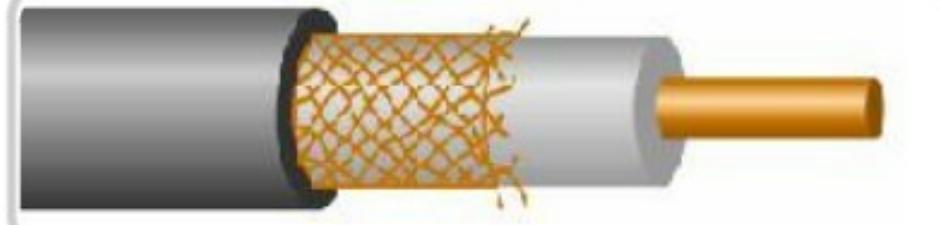
Copper Media



Unshielded Twisted
Pair (UTP) Cable

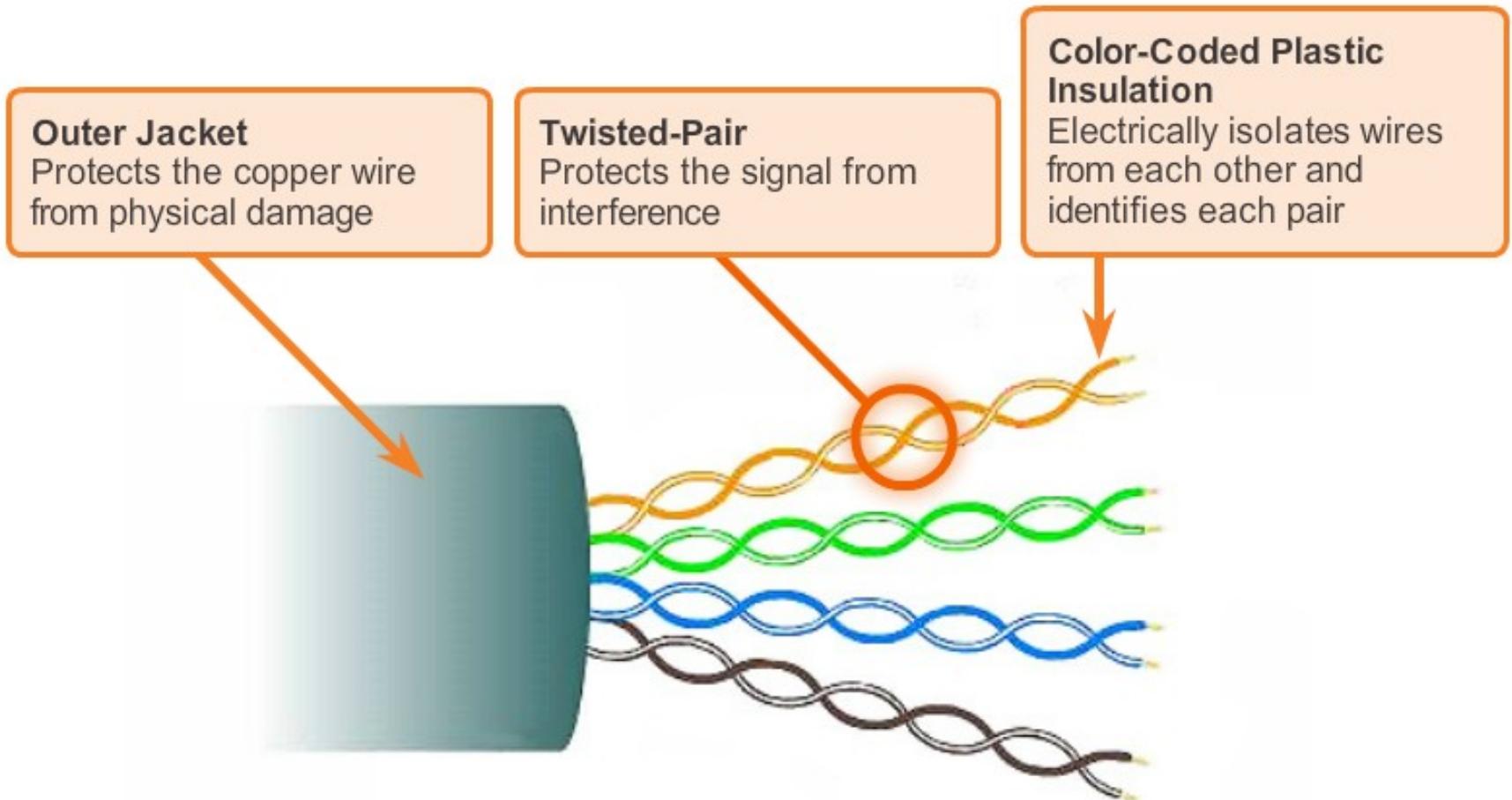


Shielded Twisted
Pair (STP) Cable



Coaxial Cable

UTP Cable

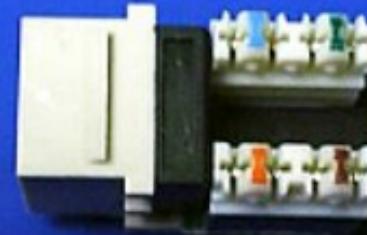


UTP Connectors

RJ-45 UTP Plugs



RJ-45 UTP Socket



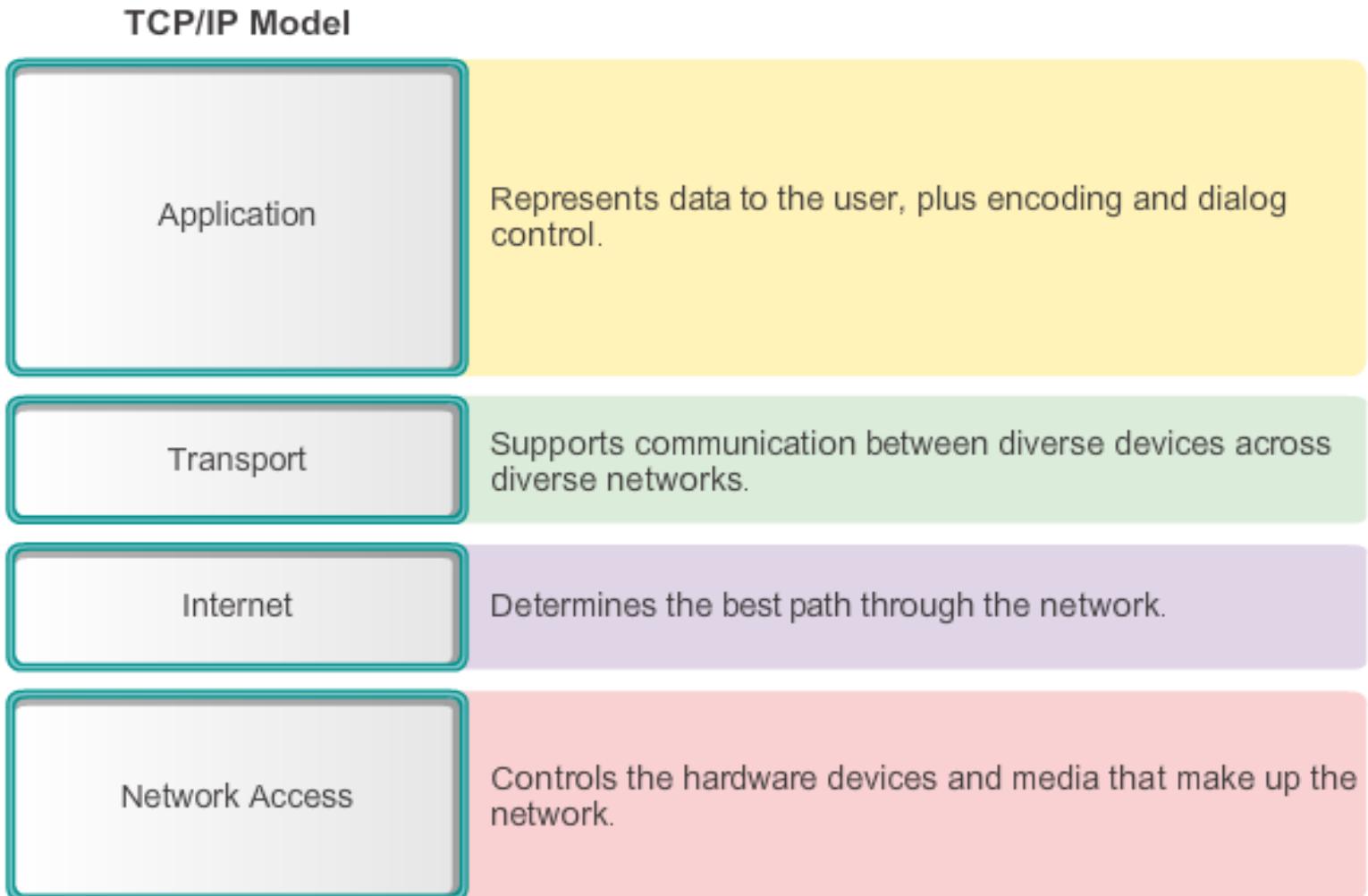
Creation of Internet, Development of TCP/IP

- The first packet switching network and predecessor to today's Internet was the Advanced Research Projects Agency Network (**ARPANET**), which came to life in **1969** by connecting **mainframe computers at four locations**.
- ARPANET was funded by the **U.S. Department of Defense** for use by universities and research laboratories. Bolt, Beranek and Newman (BBN) was the contractor that did much of the initial development of the ARPANET, including creating the first router known as an Interface Message Processor (IMP).
- In 1973, Robert Kahn and Vinton Cerf began work on **TCP** to develop the next generation of the ARPANET. TCP was designed to replace ARPANET's current Network Control Program (NCP).
- In 1978, **TCP was divided into two protocols**: TCP and IP. Later, other protocols were added to the TCP/IP suite of protocols including Telnet, FTP, DNS, and many others.

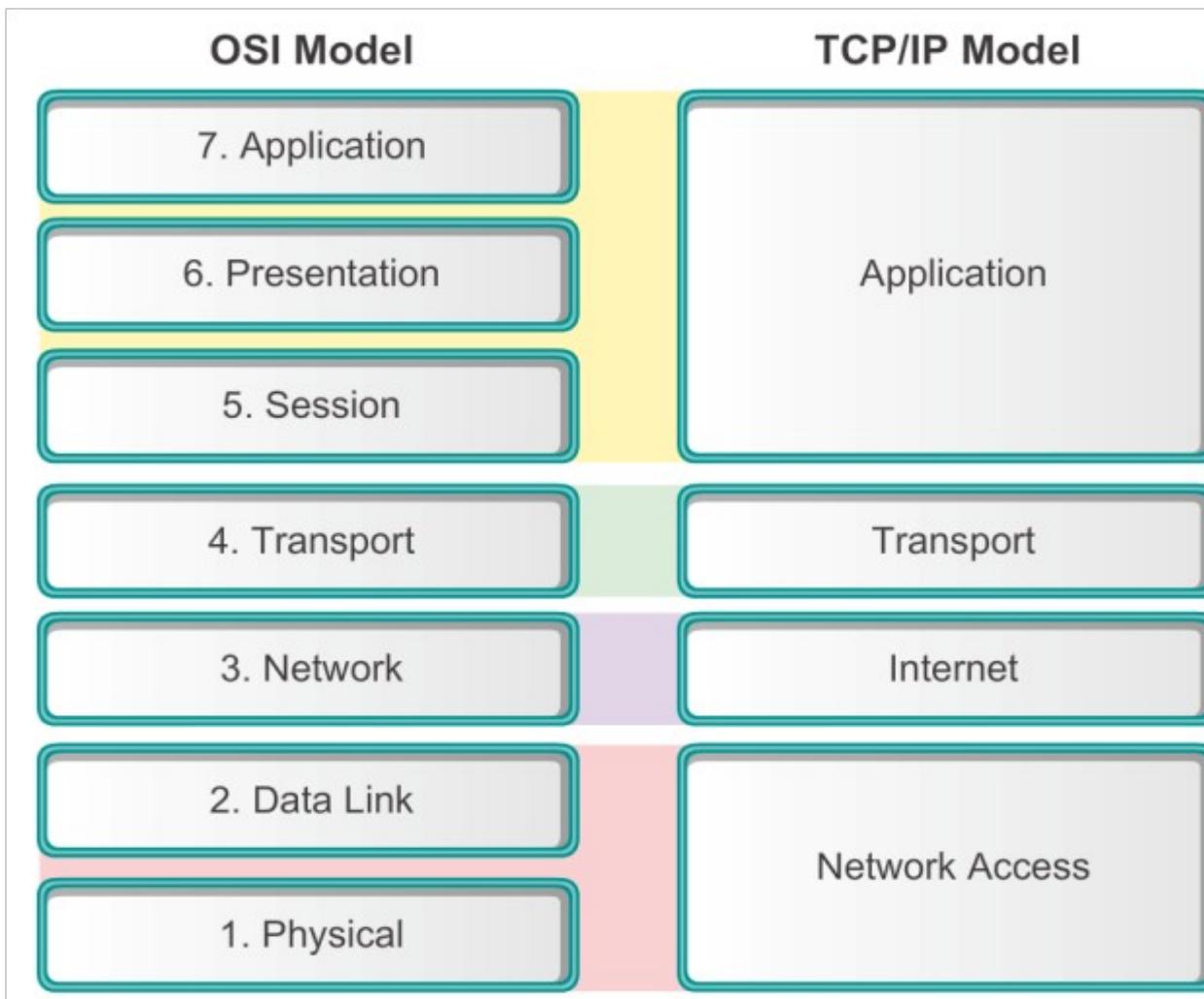
The OSI Reference Model



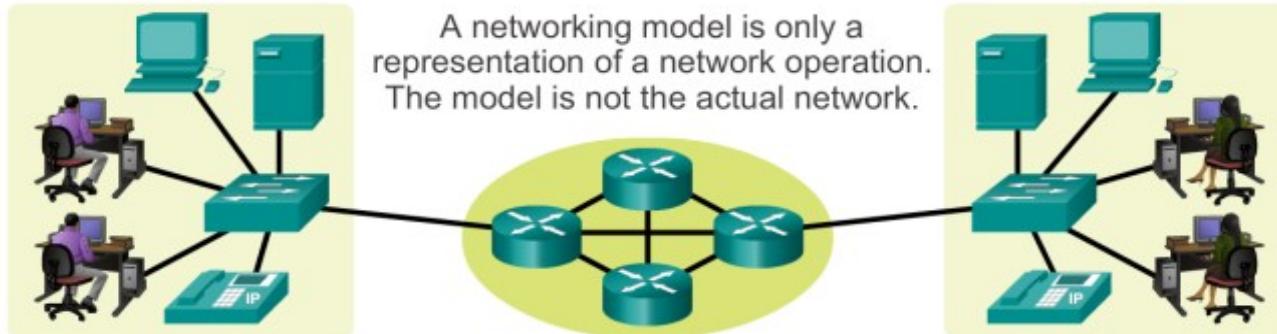
The TCP/IP Reference Model



Comparing the OSI and TCP/IP Models



Benefits of Using a Layered Model



OSI Model	TCP/IP Protocol Suite	TCP/IP Model
Application		Application
Presentation	HTTP, DNS, DHCP, FTP	
Session		
Transport	TCP, UDP	Transport
Network	IPv4, IPv6, ICMPv4, ICMPv6	Internet
Data Link	PPP, Frame Relay, Ethernet	Network Access
Physical		

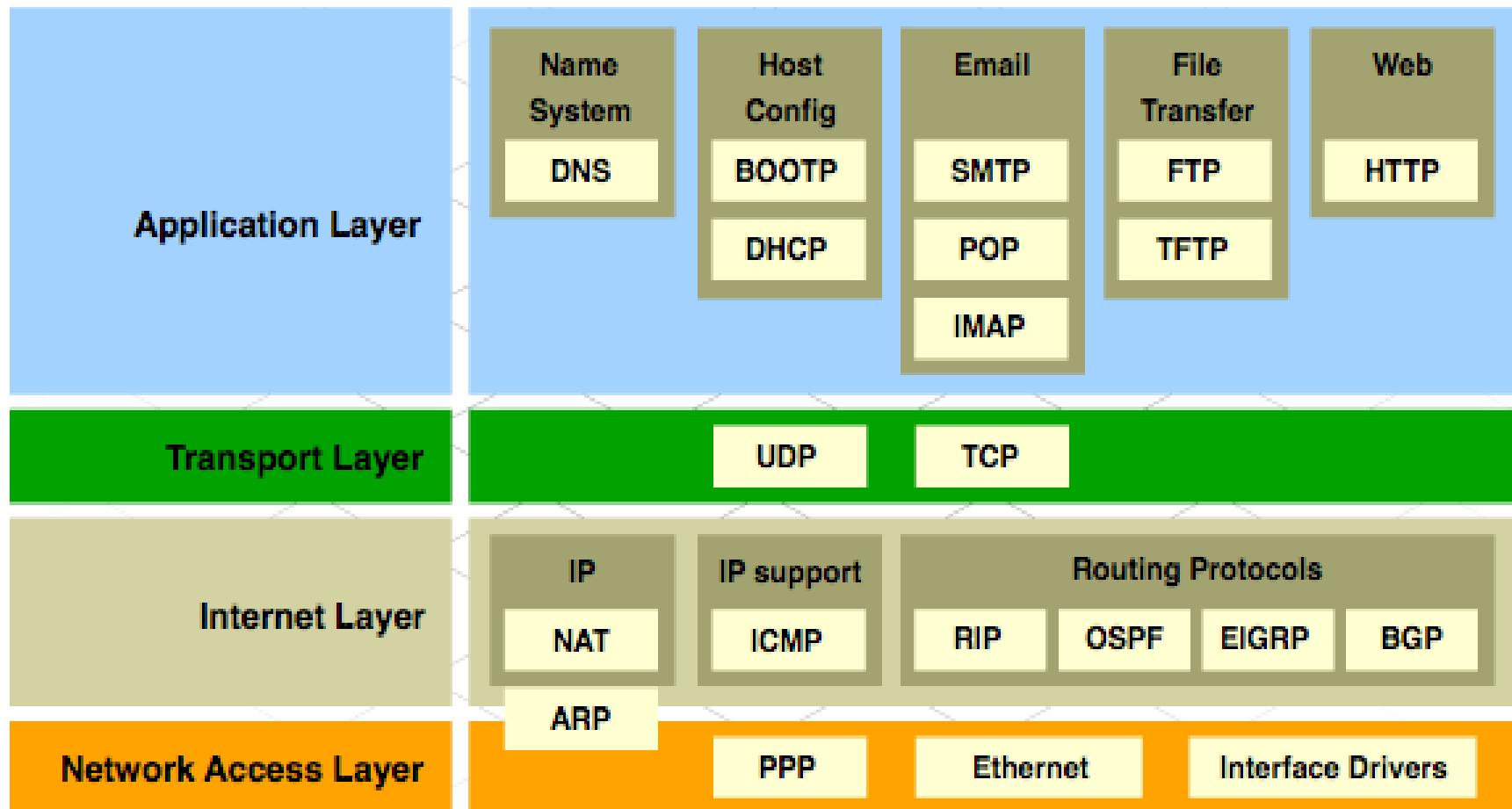
Establishing the Rules

- An identified sender and receiver
- Common language and grammar
- Speed and timing of delivery
- Confirmation or acknowledgment requirements

Network Protocols

- How the message is formatted or structured
- The process by which networking devices share information about pathways with other networks
- How and when error and system messages are passed between devices
- The setup and termination of data transfer sessions

TCP/IP Protocol Suite and Communication



Message Formatting and Encapsulation

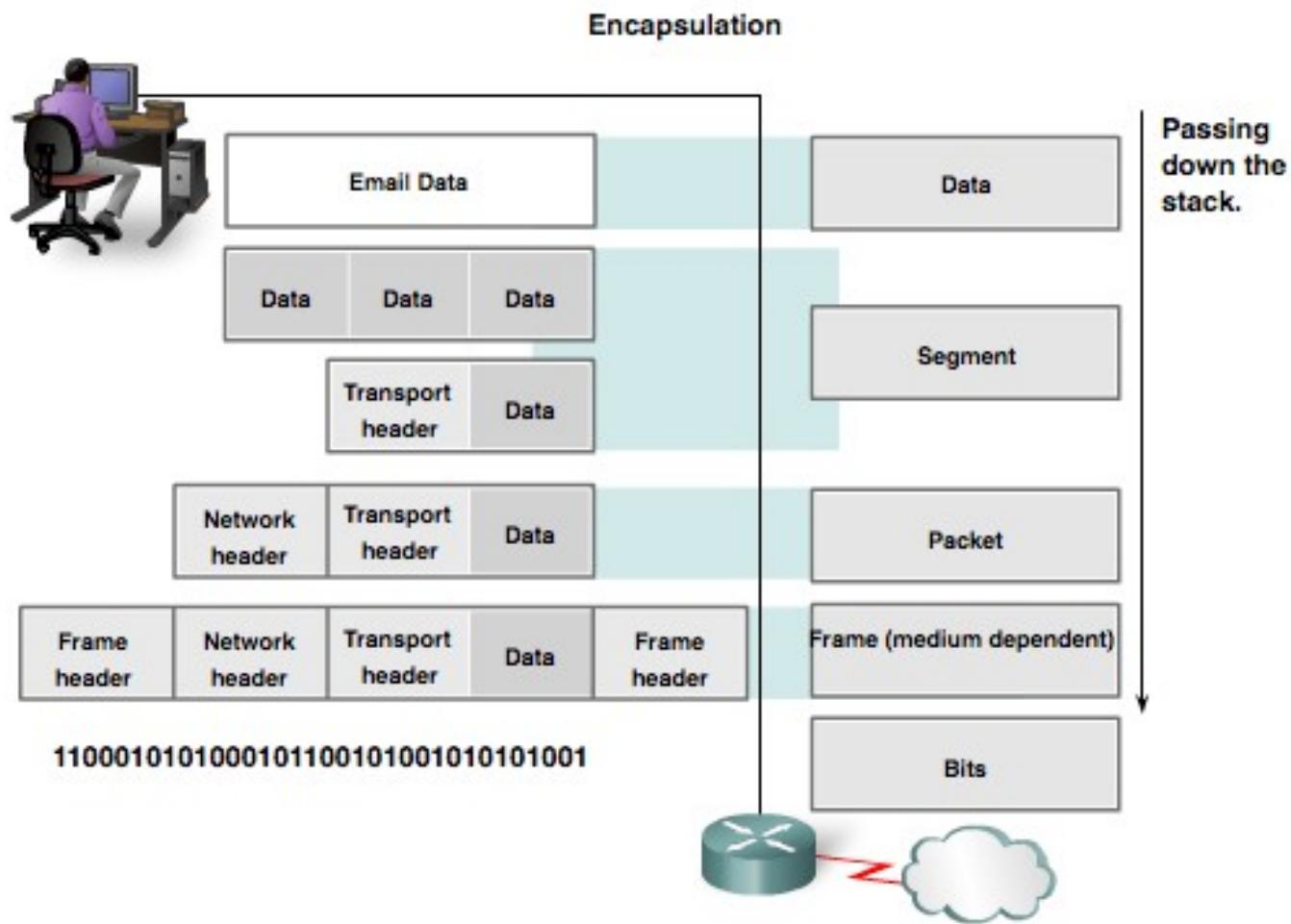
Example: Personal letter contains the following elements:

- Identifier of the recipient's location
- Identifier of the sender's location
- Salutation or greeting
- Recipient identifier
- The message content
- Source identifier
- End of message indicator



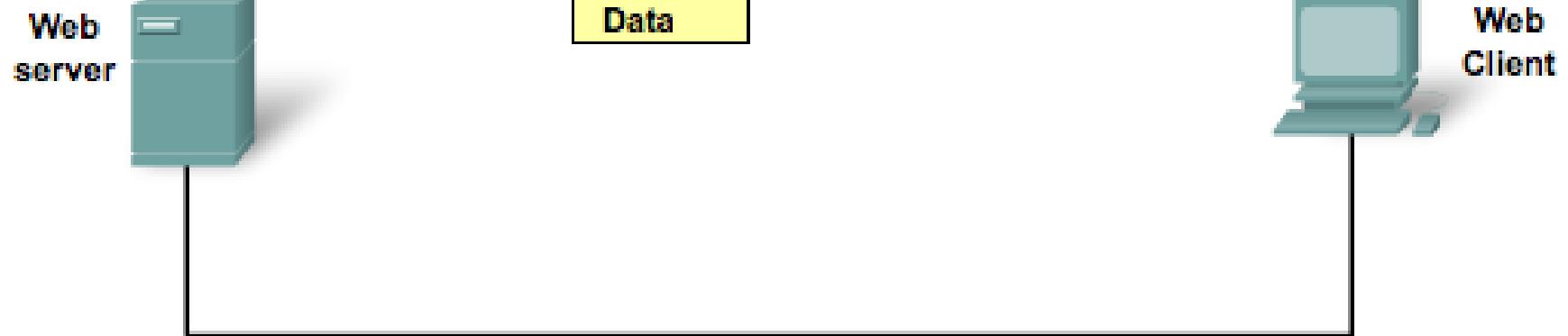
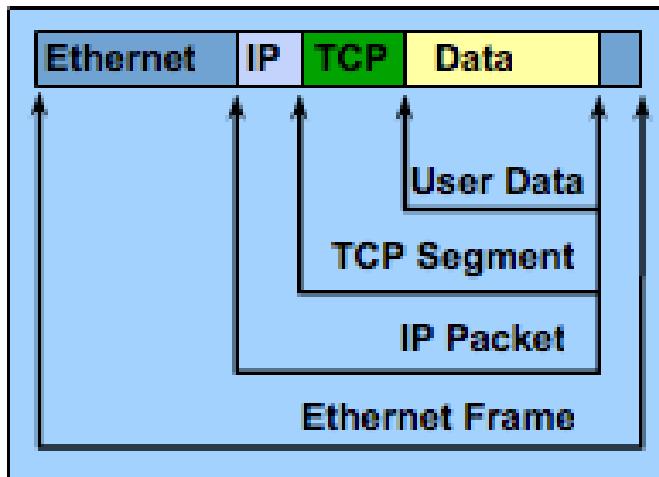
Protocol Data Units (PDUs)

- Data
 - Segment
 - Packet
 - Frame
 - Bits



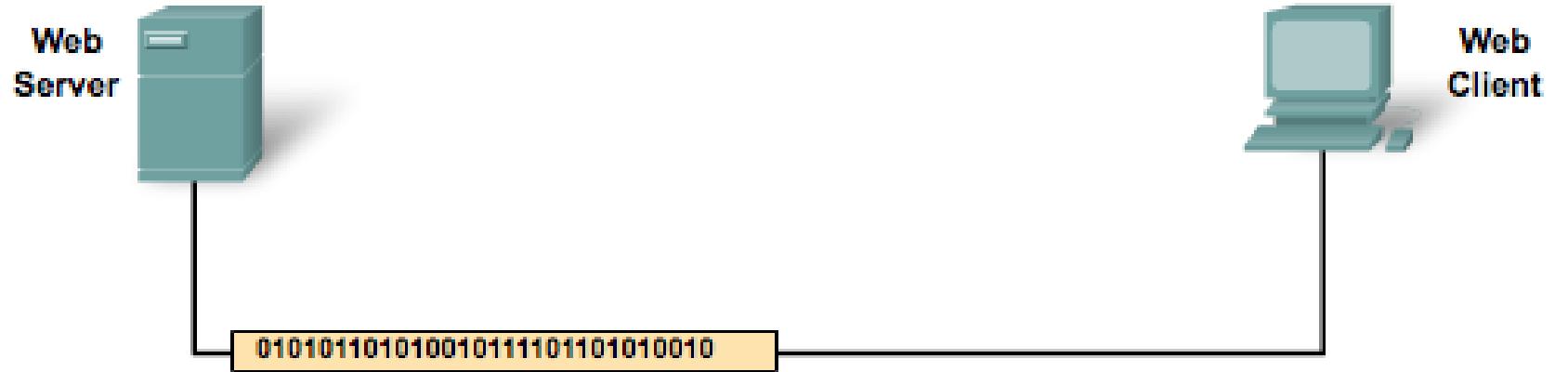
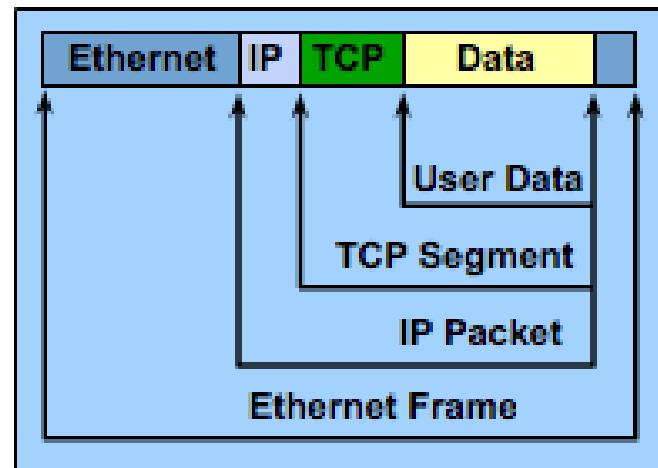
Protocol Encapsulation

Protocol Encapsulation Terms

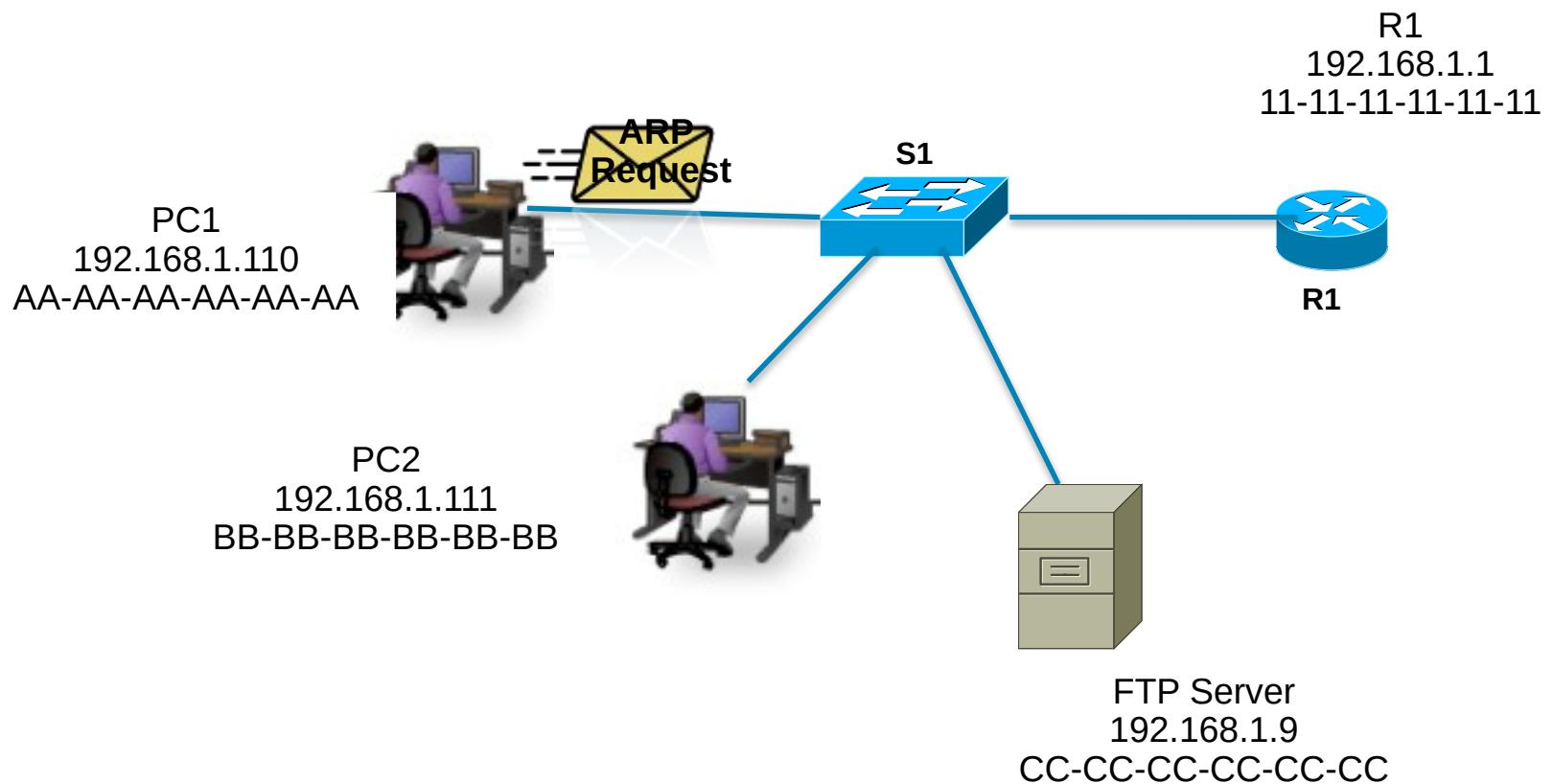


Protocol De-encapsulation

Protocol Encapsulation Terms



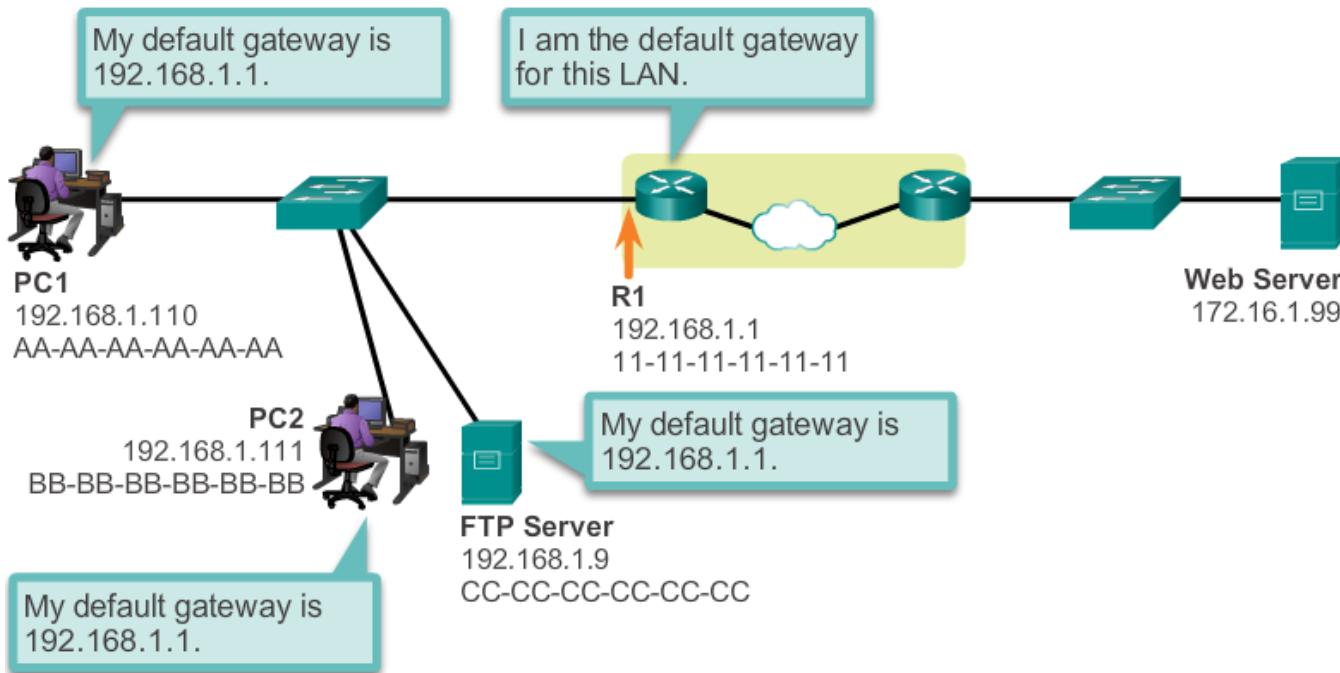
MAC and IP Addresses



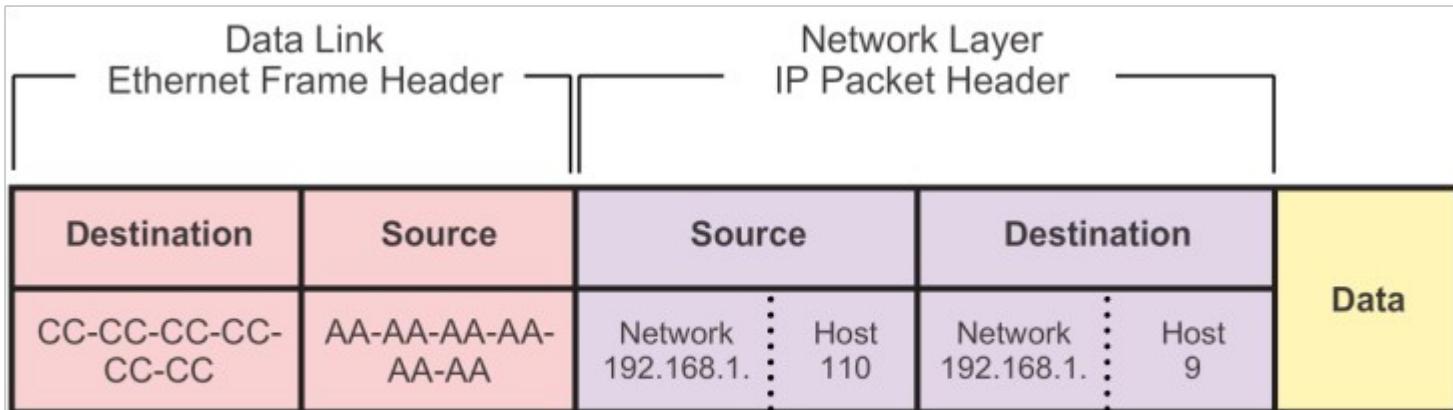
Default Gateway

Getting the Pieces to the Correct Network

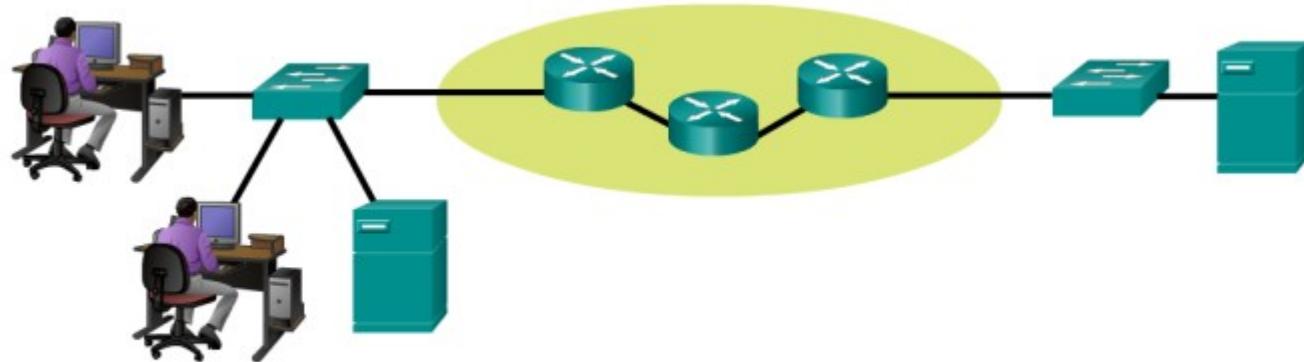
Protocol Data Unit (PDU)				Data	
Source		Destination			
Network	Device	Network	Device		
192.168.1	110	172.16.1	99		



Communicating with Device / Same Network

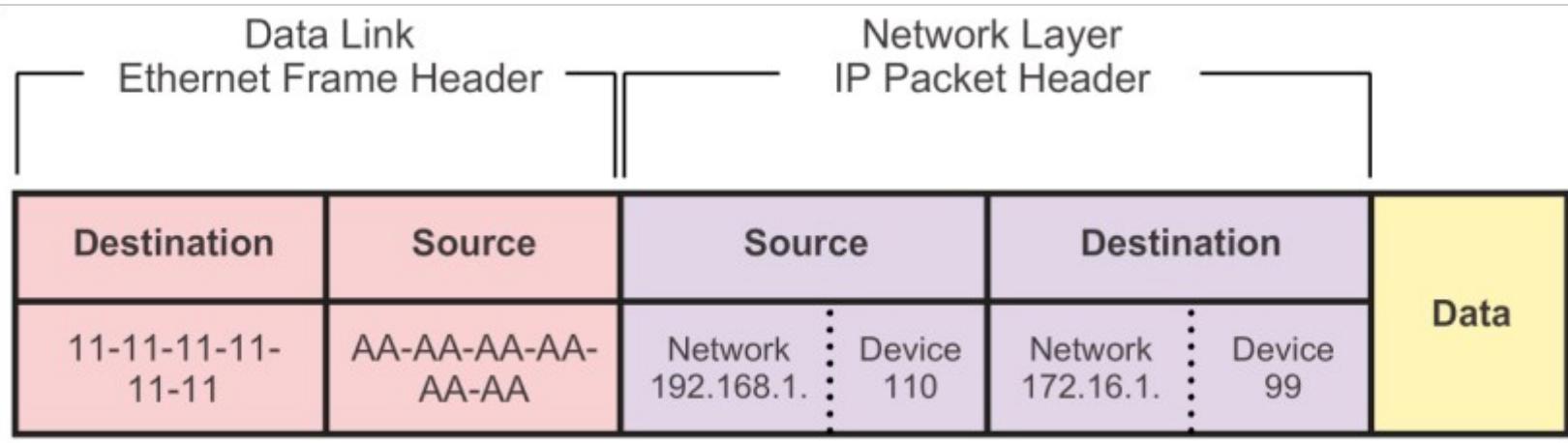


PC1
192.168.1.110
AA-AA-AA-AA-AA-AA



FTP Server
192.168.1.9
CC-CC-CC-CC-CC-CC

Communicating Device / Remote Network

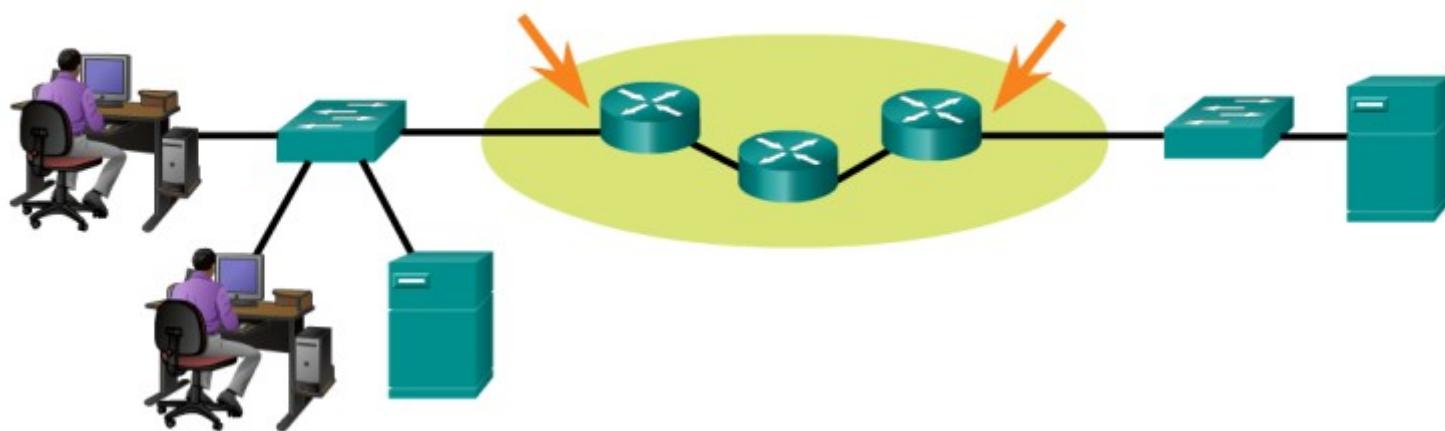


PC1
192.168.1.110
AA-AA-AA-AA-AA-AA

R1
192.168.1.1
11-11-11-11-11-11

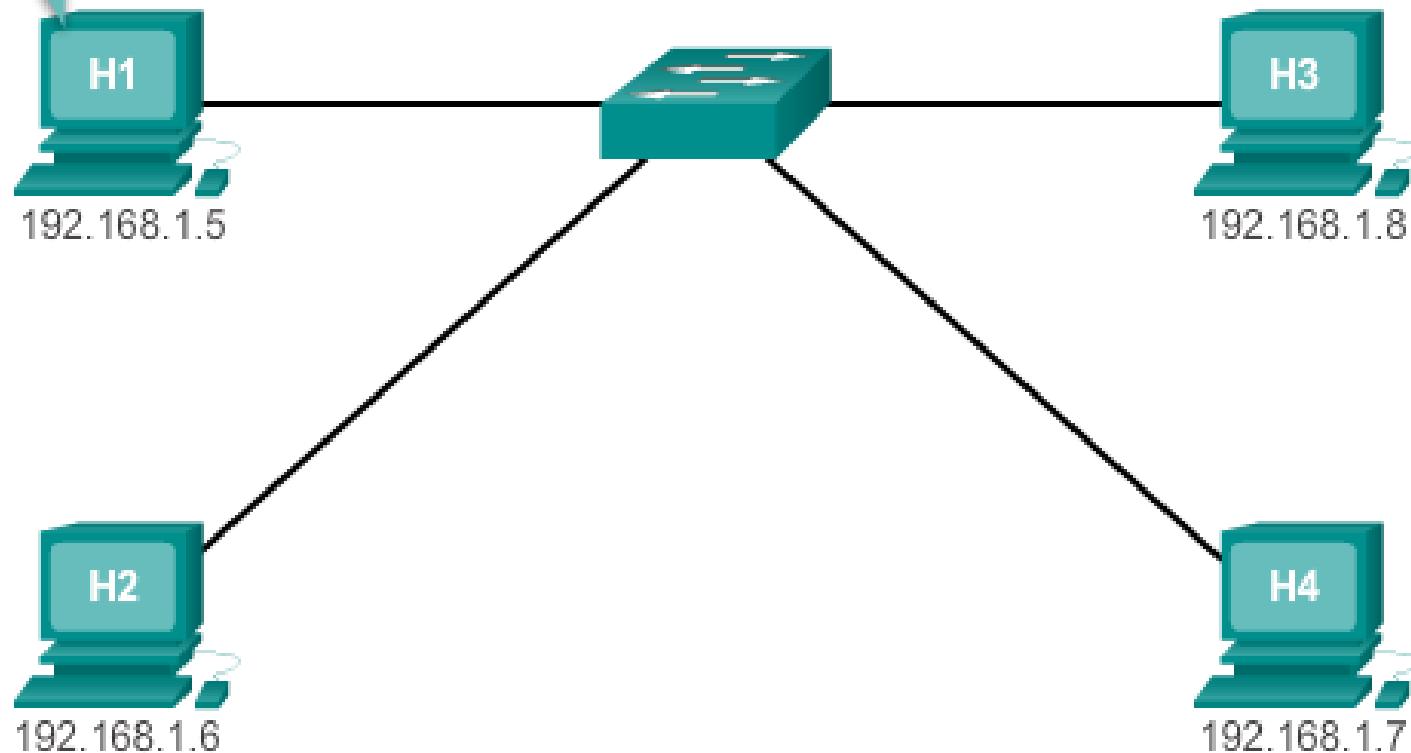
R2
172.16.1.99
22-22-22-22-22-22

Web Server
172.16.1.99
AB-CD-EF-12-34-56



Address Resolution Protocol (ARP)

I need to send information to 192.168.1.7, but I only have the IP address. I don't know the MAC address of the device that has that IP.



ARP Operation

ARP Table

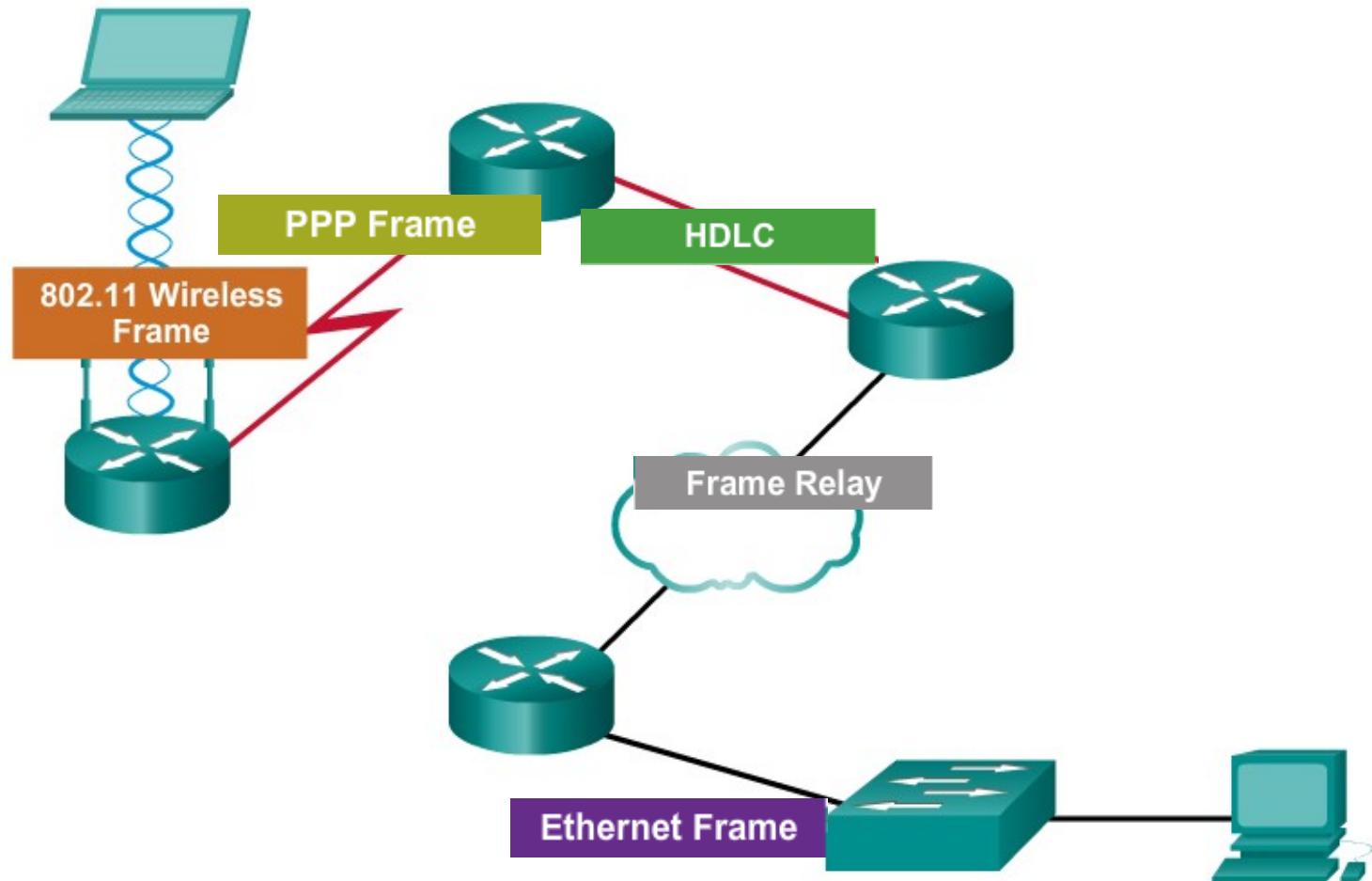
- Maps data link layer address to destination IPv4 address.
- As a node receives frames from the media, it records the source IP and MAC address as a mapping in the ARP table.

ARP Request

- Layer 2 broadcast to all devices on the Ethernet LAN.
- The node that matches the IP address in the broadcast will reply.
- If no device responds to the ARP request, the packet is dropped because a frame cannot be created.

LAN and WAN Frames

Examples of Layer 2 Protocols



Ethernet Frame format

IEEE 802.3

7 Preamble	1 Start of Frame Delimiter	6 Destinatio n Address	6 Source Address	2 Length	46 to 1500 802.2 Header and Data	4 Frame Check Sequence
---------------	-------------------------------------	------------------------------	------------------------	-------------	---	------------------------------

Preamble and Start Frame Delimiter Fields –
Used for synchronization between the sending and receiving devices.

Length/Type Field –
Defines the exact length of the frame's data field; describes which protocol is implemented.

Data and Pad Fields –
Contains the encapsulated data from a higher layer, an IPv4 packet.

The Network Layer

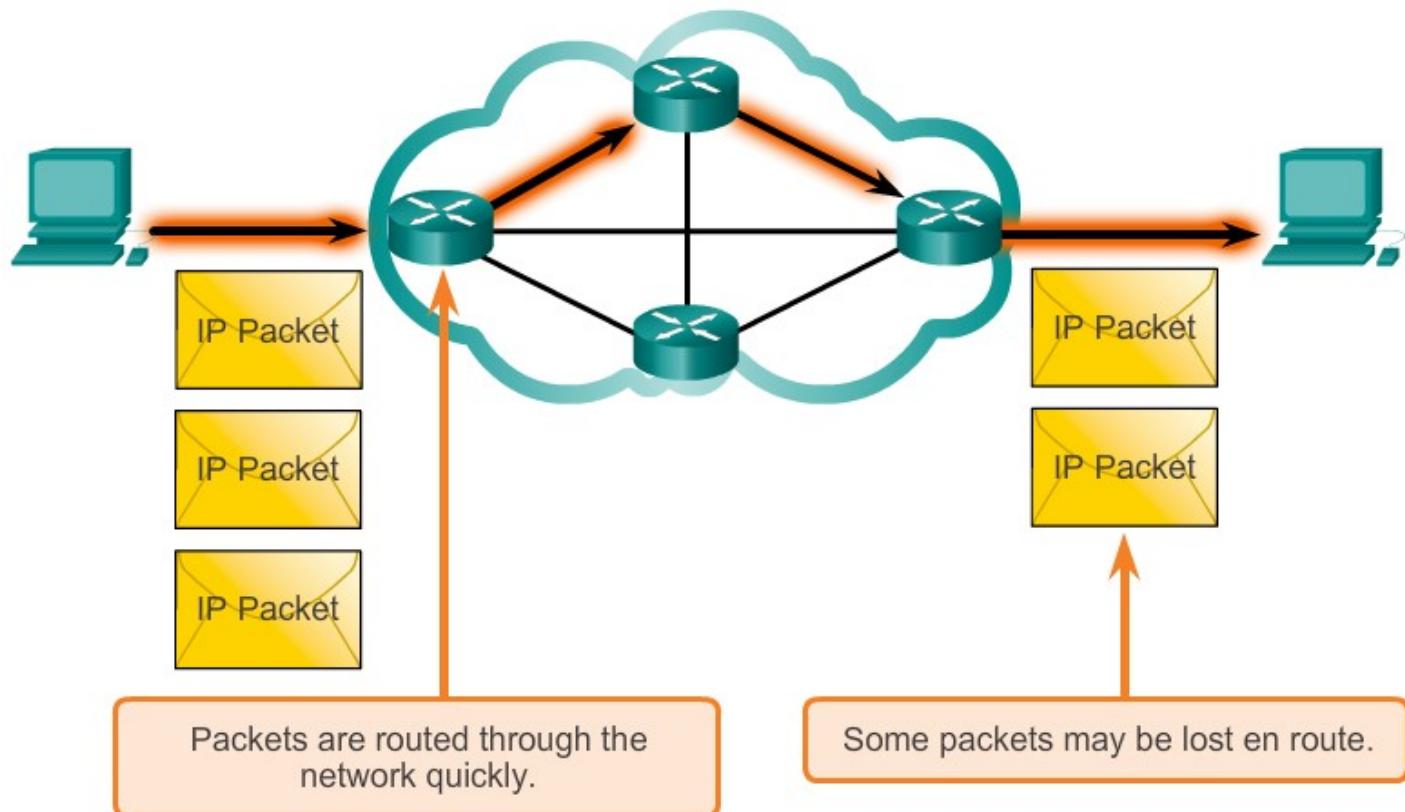
The network layer, or OSI Layer 3, provides services to allow end devices to exchange data across the network. To accomplish this end-to-end transport, the network layer uses four basic processes:

- Addressing end devices
- Encapsulation
- Routing
- De-encapsulating

Network Layer Protocols

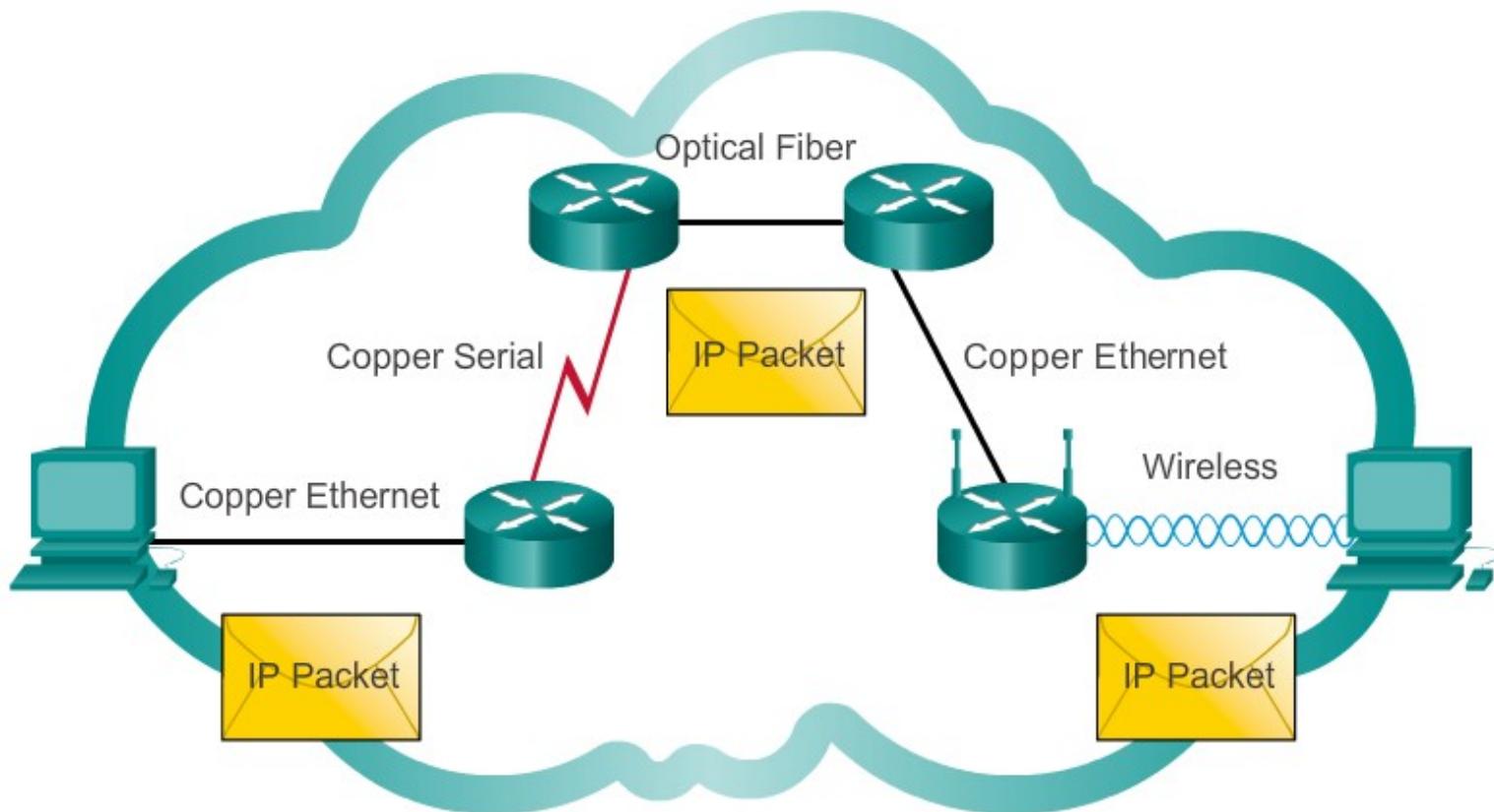
- IP version 4 (IPv4)
- IP version 6 (IPv6)

Best Effort Delivery



As an unreliable network layer protocol, IP does not guarantee that all sent packets will be received. Other protocols manage the process of tracking packets and ensuring their delivery.

IP – Media Independent



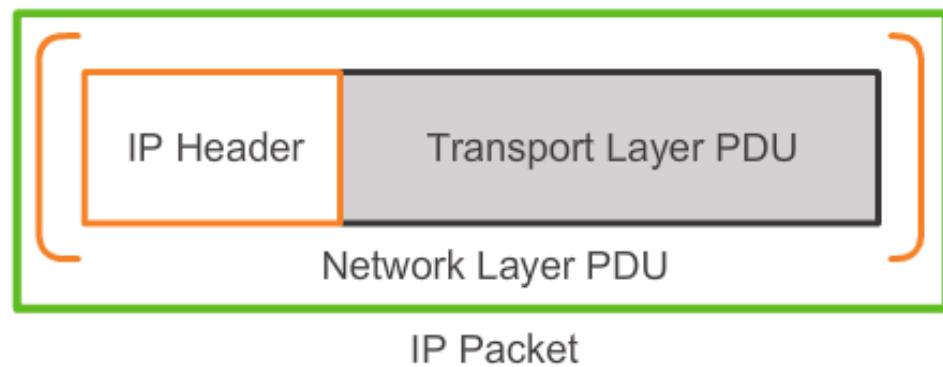
IP packets can travel over different media.

Encapsulating IP

Transport Layer
Encapsulation

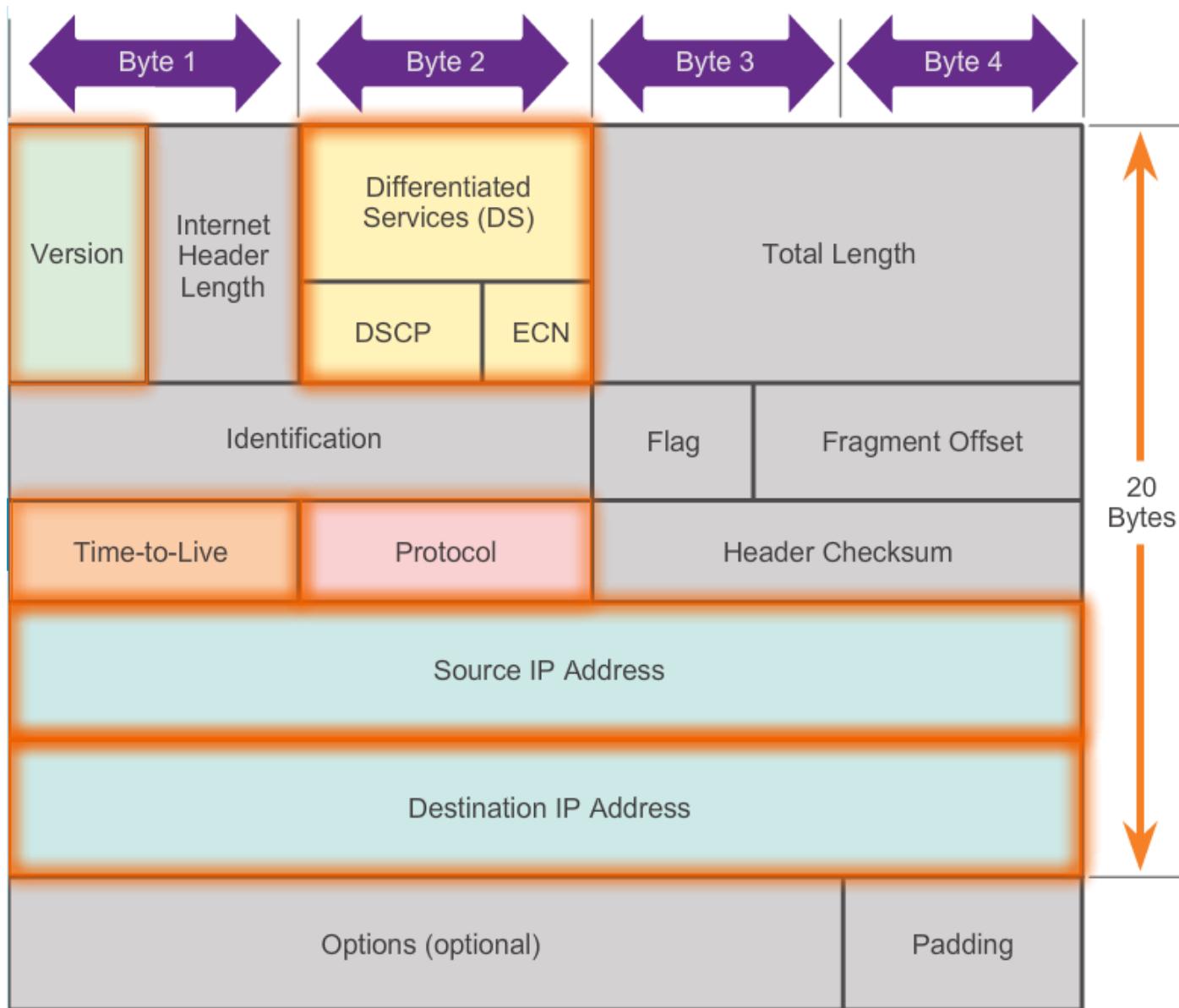


Network Layer Encapsulation

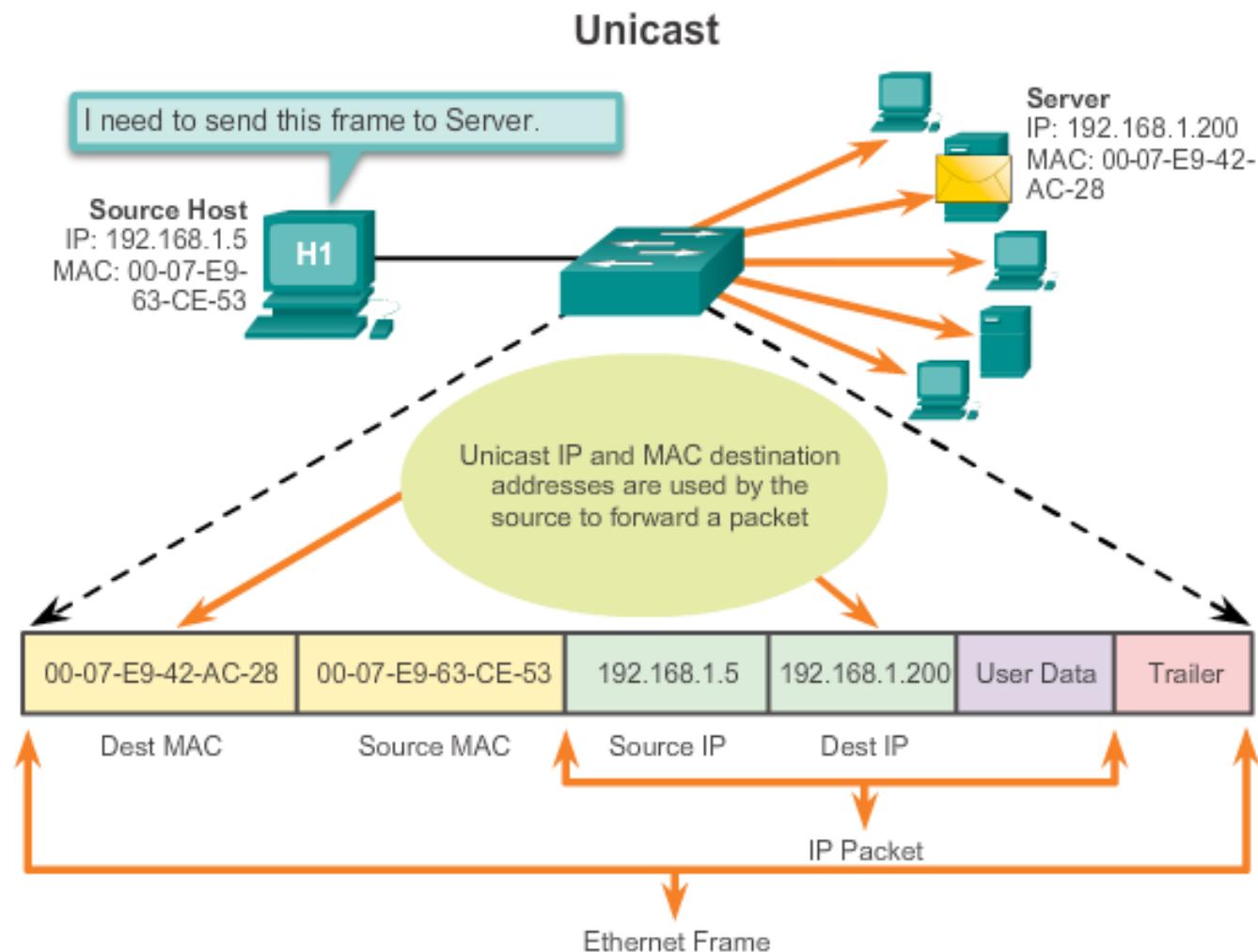


The network layer adds a header so packets can be routed through complex networks and reach their destination. In TCP/IP based networks, the network layer PDU is the IP packet.

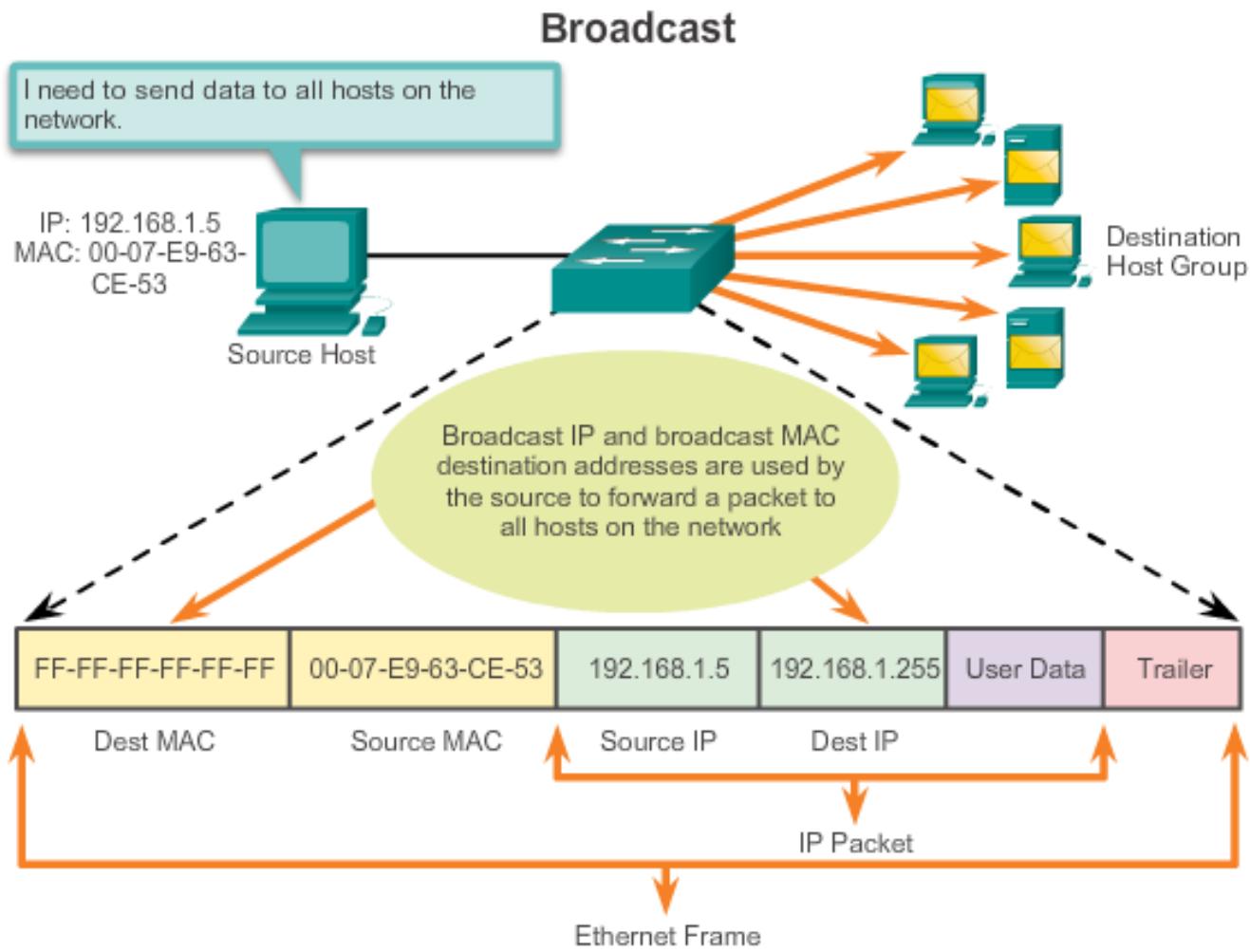
IPv4 Packet Header



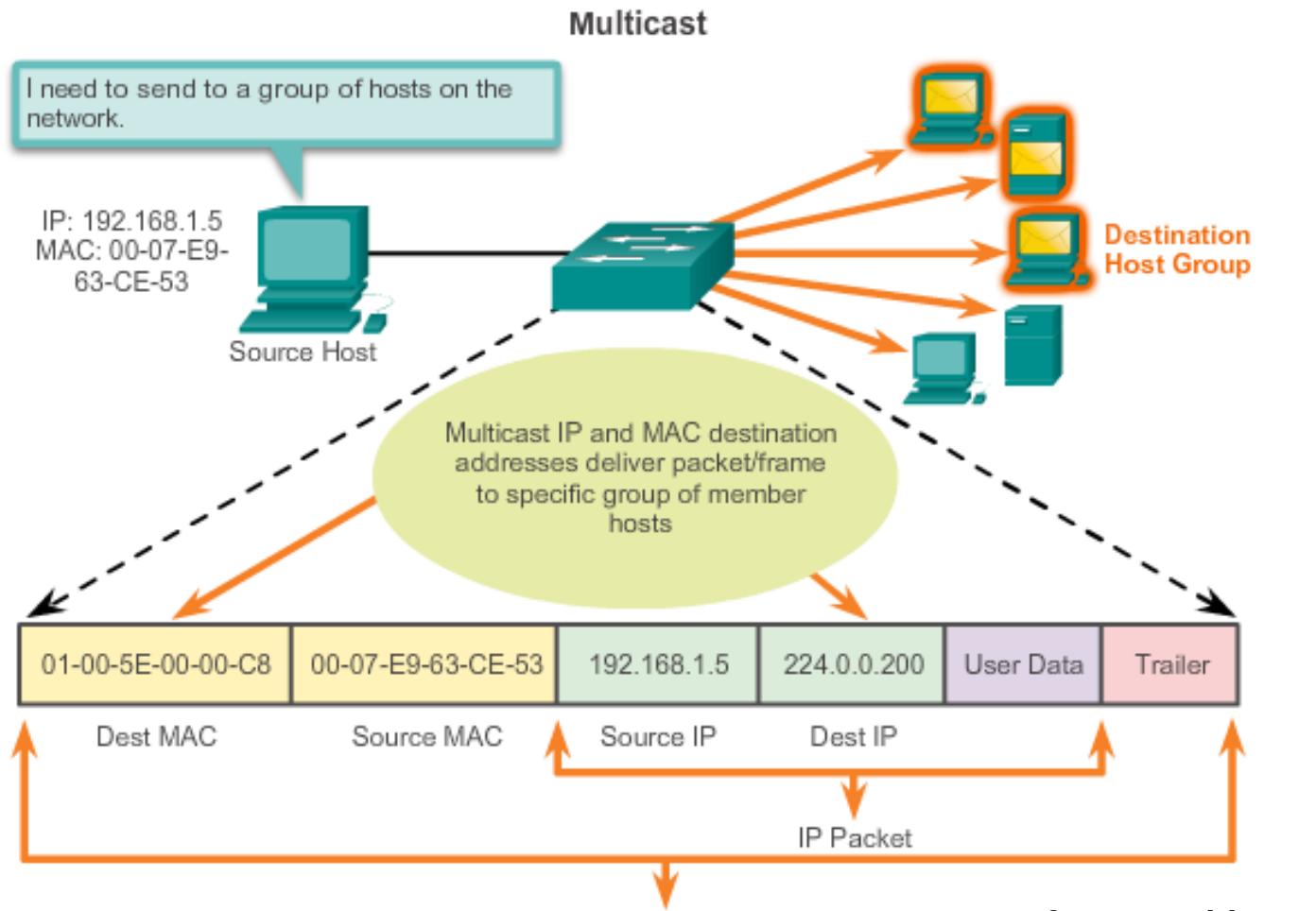
Unicast Address



Broadcast Address



Multicast Address



Multicast MAC address is a special value that begins with 01-00-5E in hexadecimal

Range of IPV4 multicast addresses is 224.0.0.0 to 239.255.255.255

Sample IPv4 Headers in Wireshark

The screenshot shows a Wireshark capture window titled "Microsoft: \Device\NPF_{7BB3C130-30C5-4419-B79E-C0868085ABED} [Wireshark 1.8.2 (SVN Rev 44520 from /trunk-1.8)]". The main pane displays a list of network packets, with packet 16 selected. The details pane shows the structure of the selected IPv4 header:

- Frame 16: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
- Ethernet II, Src: IntelCor_45:5d:c4 (24:77:03:45:5d:c4), Dst: Cisco-Li_a0:d1:be (00:18:39:a0:d1:be)
- Internet Protocol Version 4, Src: 192.168.1.109 (192.168.1.109), Dst: 192.168.1.1 (192.168.1.1)
 - Version: 4
 - Header length: 20 bytes
 - Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
 - Total Length: 60
 - Identification: 0x3704 (14084)
 - Flags: 0x00
 - Fragment offset: 0
 - Time to live: 128
 - Protocol: ICMP (1)
 - Header checksum: 0x7ffe [correct]
 - Source: 192.168.1.109 (192.168.1.109)
 - Destination: 192.168.1.1 (192.168.1.1)
 - [Source GeoIP: Unknown]
 - [Destination GeoIP: Unknown]
- Internet Control Message Protocol

The bytes pane shows the raw hex and ASCII data for the selected IPv4 header:

Hex	Dec	ASCII
0000 00 18 39 a0 d1 be 24 77	00 18 39 a0 d1 be 24 77 \$w . E] . . E.
0010 00 3c 37 04 00 00 80 01	00 3c 37 04 00 00 80 01	. <7 m . .
0020 7f fe c0 a8 01 6d c0 a8	7f fe c0 a8 01 6d c0 a8 M . . . abcdef
0030 01 01 08 00 4d 56 00 01	01 01 08 00 4d 56 00 01	00 05 61 62 63 64 65 66
0040 6f 70 71 72 73 74 75 76	6f 70 71 72 73 74 75 76	ghijklmn opqrstuv
0040 77 61 62 63 64 65 66 67	77 61 62 63 64 65 66 67	wabcefg hi

At the bottom, status bars show "Internet Protocol Version 4 (ip), 20 bytes", "Packets: 35 Displayed: 35 Marked: 0 Dropped: 0", and "Profile: Default".

Limitations of IPv4

- IP Address depletion
- Internet routing table expansion
- Lack of end-to-end connectivity

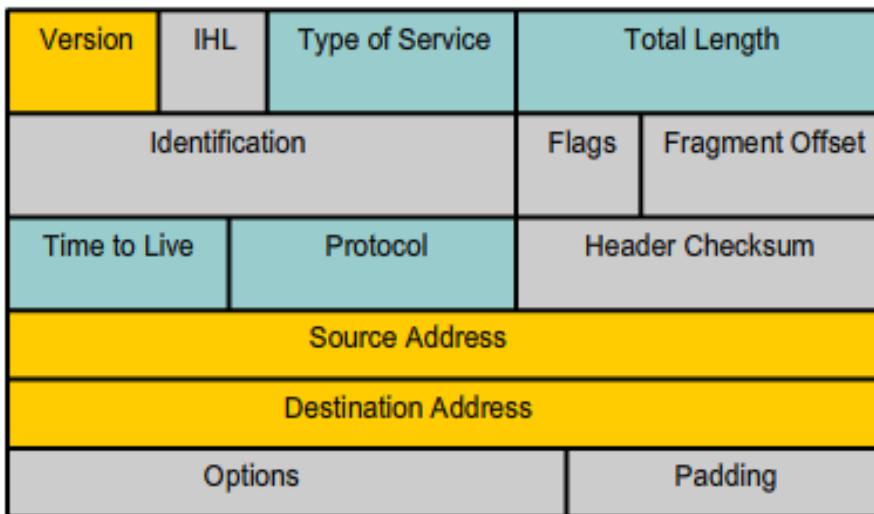


IPv6

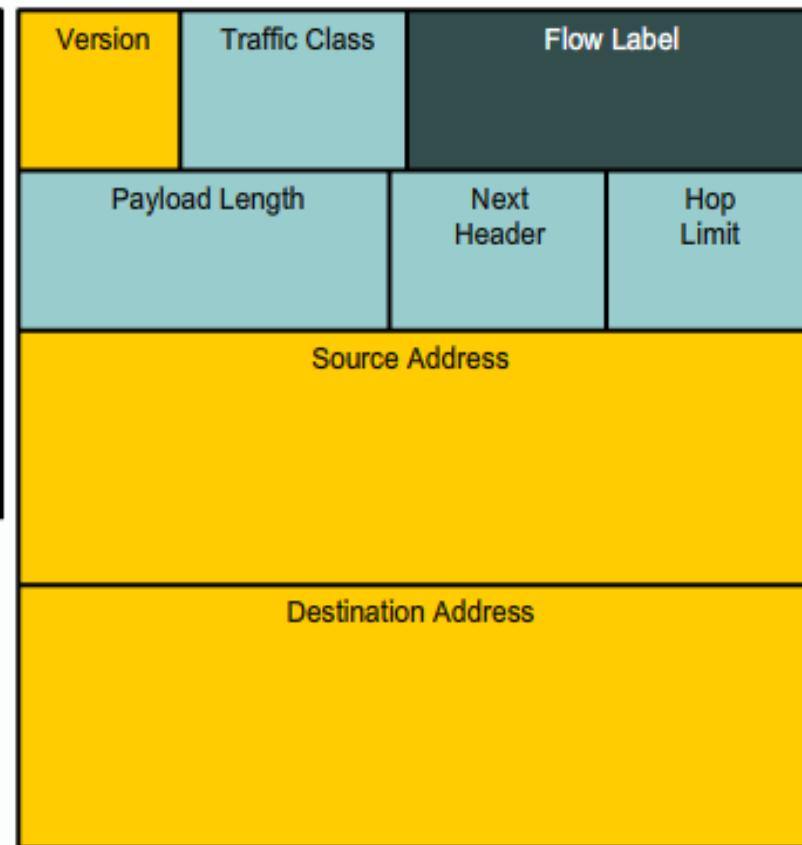
IPv4 and IPv6 Headers

IPv4 and IPv6 Headers

IPv4 Header



IPv6 Header



Legend

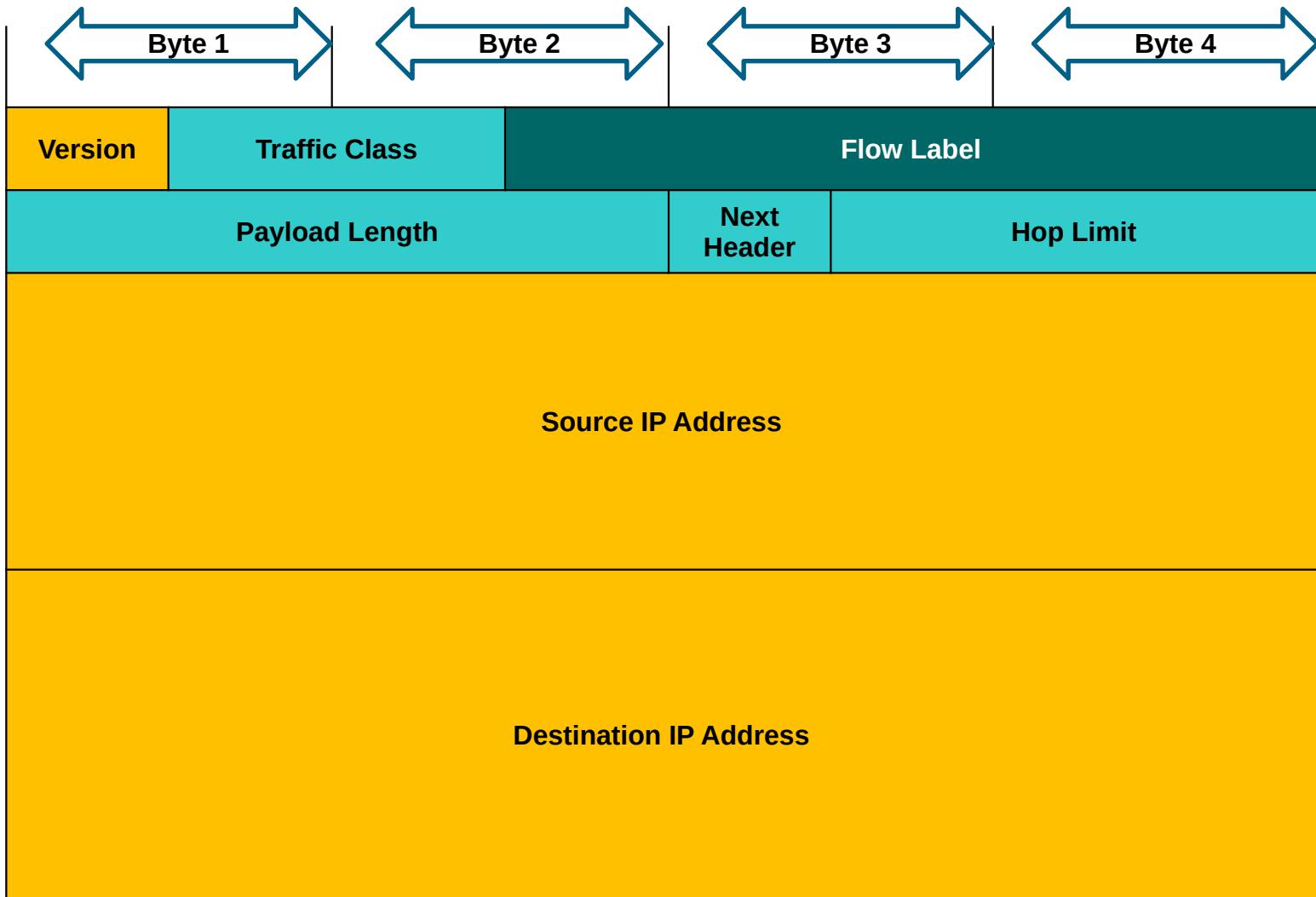
- Field names kept from IPv4 to IPv6

- Fields not kept in IPv6

- Name & position changed in IPv6

- New field in IPv6

IPv6 Packet Header



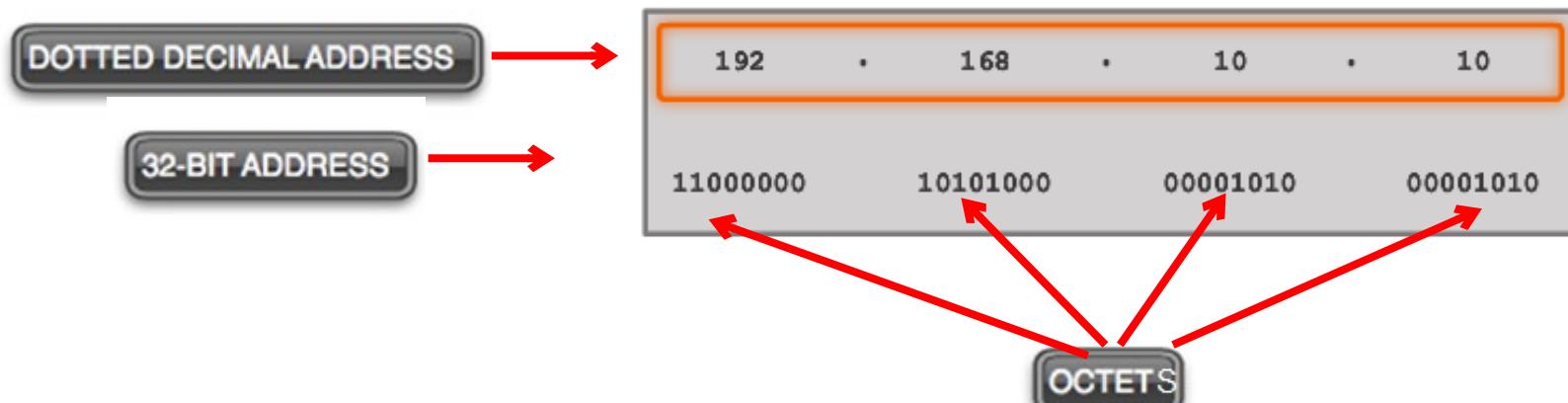
IP addressing & Subnet

**Dr. Md. Shohrab Hossain
Professor, CSE, BUET**

Slide Source: Cisco Networking

- IP addressing
- Public / Private IP address
- Network Address Translation
- Subnetting

Binary Number System

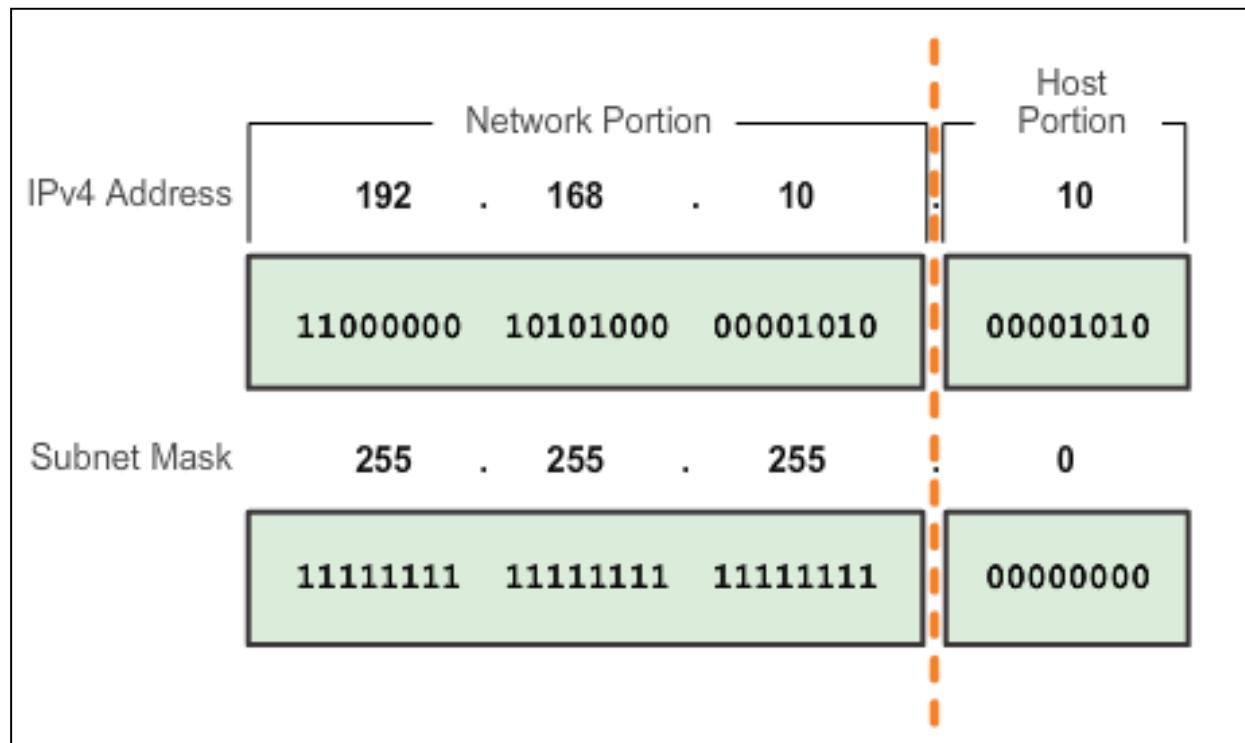


Radix	2	2	2	2	2	2	2	2	2
Exponent	7	6	5	4	3	2	1	0	
Octet Bit Values	128	64	32	16	8	4	2	1	
Binary Address	1	1	0	0	0	0	0	0	
Binary Bit Values	128	64	0	0	0	0	0	0	

Add the binary bit values.
128 + 64 = 192

Network Portion and Host Portion of an IPv4 Address

- The subnet mask just says where **to look for the network portion** in a given IPv4 address



Bitwise AND Operation

IPv4 Address	192	.	168	.	10	.	10
	11000000		10101000		00001010		00001010
Subnet Mask	255	.	255	.	255	.	0
	11111111		11111111		11111111		00000000
Network Address	192	.	168	.	10	.	0
	11000000		10101000		00001010		00000000

IPv4 Subnet masks

Valid Subnet Masks

Subnet Value
255
254
252
248
240
224
192
128
0

Bit Value

128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Network Address and Broadcast address

- For a network, 192.168.10.0/24 network

There are 8-bit allocated for host part.

- Network Address:
 - All 0's in the host part: $0000\ 0000 = 0$
 - So, network address = 192.168.10.0
- Broadcast address:
 - All 1's in the host part: $1111\ 1111 = 255$
 - So, broadcast address = 192.168.10.255
- IP range: 1- 254

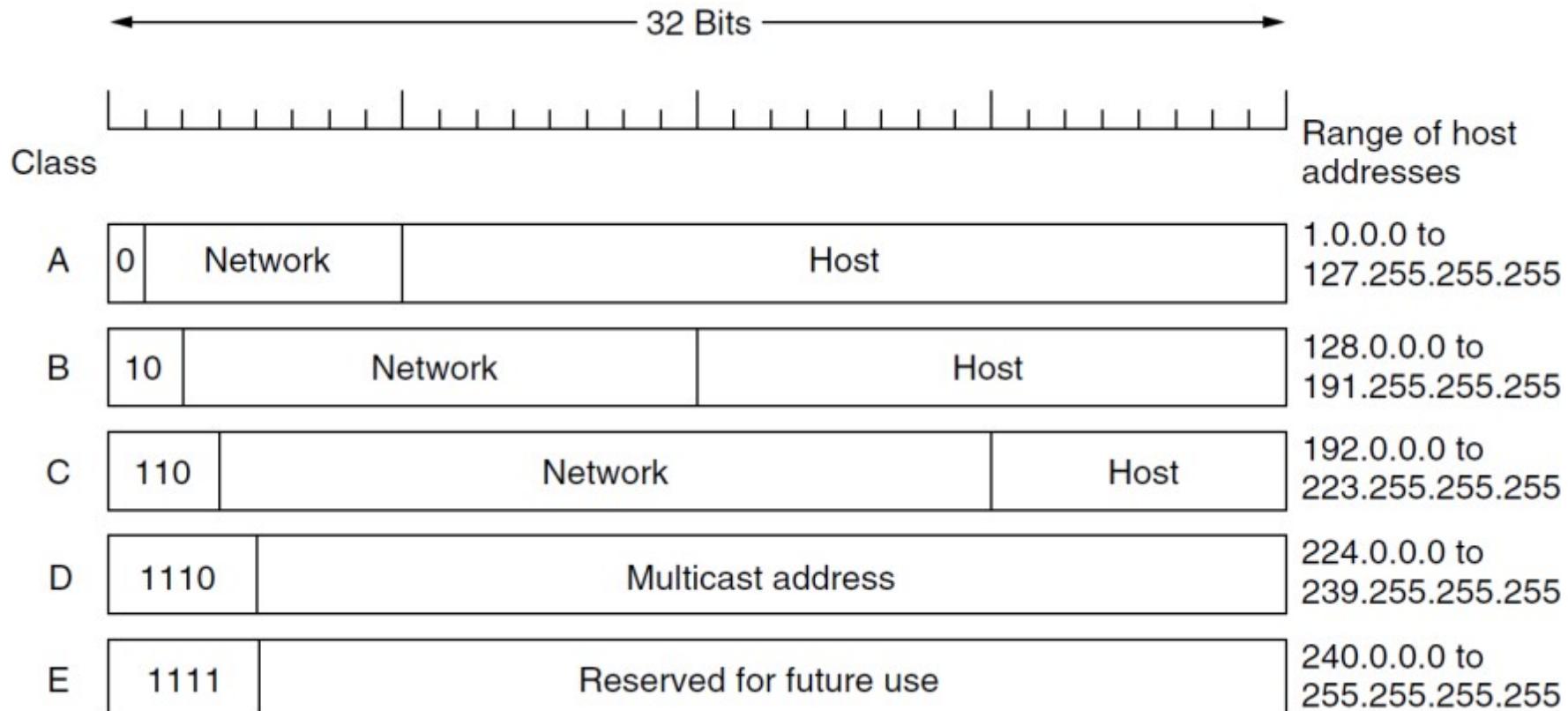
Types of IPv4 Address

Legacy Classful Addressing

IP Address Classes

Address Class	1st octet range (decimal)	1st octet bits (green bits do not change)	Network(N) and Host(H) parts of address	Default subnet mask (decimal and binary)	Number of possible networks and hosts per network
A	1-127**	00000000-01111111	N.H.H.H	255.0.0.0	128 nets (2^7) 16,777,214 hosts per net (2^{24-2})
B	128-191	10000000-10111111	N.N.H.H	255.255.0.0	16,384 nets (2^{14}) 65,534 hosts per net (2^{16-2})
C	192-223	11000000-11011111	N.N.N.H	255.255.255.0	2,097,150 nets (2^{21}) 254 hosts per net (2^{8-2})
D	224-239	11100000-11101111	NA (multicast)		
E	240-255	11110000-11111111	NA (experimental)		

Classful Addressing



Classless Addressing

- Formal name is **Classless Inter-Domain Routing** (CIDR)
- Created a new set of standards that allowed service providers to allocate IPv4 addresses on any address bit boundary (prefix length) instead of only by a class **A, B, or C** address
- Example:
192.168.10.0/23

Examining the Prefix Length

	Dotted Decimal	Significant bits shown in binary
Network Address	10.1.1.0/24	10.1.1.00000000
First Host Address	10.1.1.1	10.1.1.00000001
Last Host Address	10.1.1.254	10.1.1.11111110
Broadcast Address	10.1.1.255	10.1.1.11111111
Number of hosts:	$2^8 - 2 = 254$ hosts	

	Dotted Decimal	Significant bits shown in binary
Network Address	10.1.1.0/25	10.1.1.00000000
First Host Address	10.1.1.1	10.1.1.00000001
Last Host Address	10.1.1.126	10.1.1.01111110
Broadcast Address	10.1.1.127	10.1.1.01111111
Number of hosts:	$2^7 - 2 = 126$ hosts	

	Dotted Decimal	Significant bits shown in binary
Network Address	10.1.1.0/26	10.1.1.00000000
First Host Address	10.1.1.1	10.1.1.00000001
Last Host Address	10.1.1.62	10.1.1.00111110
Broadcast Address	10.1.1.63	10.1.1.00111111
Number of hosts:	$2^6 - 2 = 62$ hosts	

Examining the Prefix Length (cont.)

		Dotted Decimal	Significant bits shown in binary
Network Address	10.1.1.0/27	10.1.1.00000000	
First Host Address	10.1.1.1	10.1.1.00000001	
Last Host Address	10.1.1.30	10.1.1.00011110	
Broadcast Address	10.1.1.31	10.1.1.00011111	
Number of hosts: $2^5 - 2 = 30$ hosts			

Network Address	10.1.1.0/28	10.1.1.00000000
First Host Address	10.1.1.1	10.1.1.00000001
Last Host Address	10.1.1.14	10.1.1.00001110
Broadcast Address	10.1.1.15	10.1.1.00001111
Number of hosts: $2^4 - 2 = 14$ hosts		

Private IP address

Public and Private IPv4 Addresses

Private address blocks are:

- Hosts that do not require access to the Internet can use private addresses
 - 10.0.0.0 to 10.255.255.255 (10.0.0.0/8)
 - 172.16.0.0 to 172.31.255.255 (172.16.0.0/12)
 - 192.168.0.0 to 192.168.255.255 (192.168.0.0/16)
- Not globally routable
- Address translation is done at Router to convert private IP into Public IP address and vice versa

What is NAT?

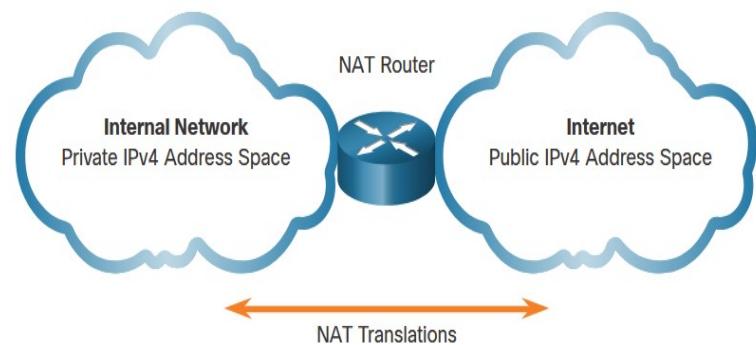
- NAT is a process used to translate network addresses.
- NAT's primary use is to conserve public IPv4 addresses.
- NAT is usually **implemented at border network devices**, such as firewalls or routers.
- NAT allows the networks to use private addresses internally, only translating to public addresses when needed.
- **Devices within the organization** can be assigned private addresses and operate with locally unique addresses.
- When traffic must be sent or received to or from other organizations or the Internet, the **border router translates** the addresses to a public and globally unique address.

NAT Characteristics

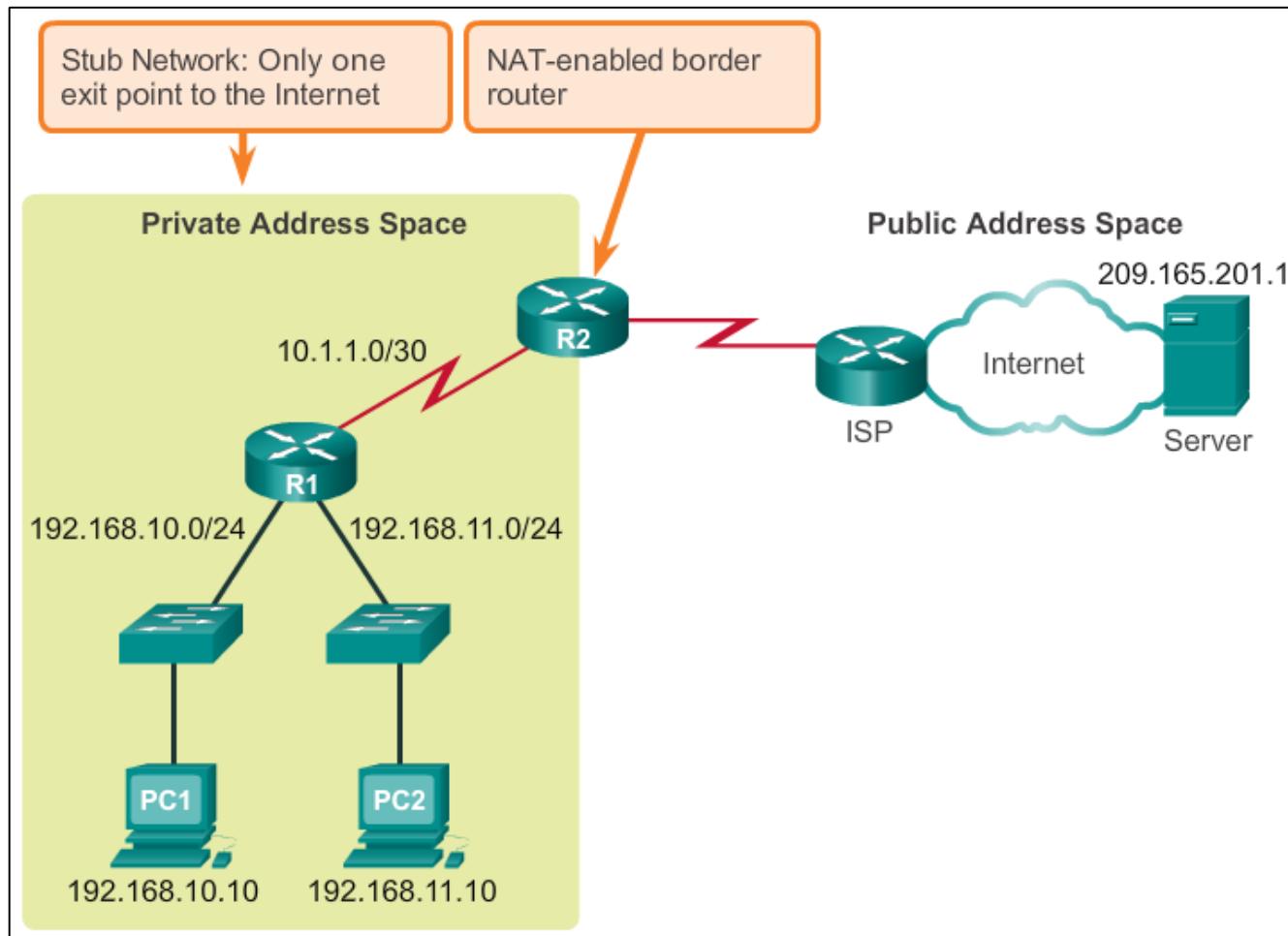
IPv4 Address Space

- Networks are commonly implemented using private IPv4 addresses, as defined in RFC 1918.
- Private IPv4 addresses cannot be routed over the internet and are used within an organization or site to allow devices to communicate locally.
- To allow a device with a private IPv4 address to access devices and resources outside of the local network, the private address must first be translated to a public address.
- NAT provides the translation of private addresses to public addresses.

Class	Activity Type	Activity Name
A	10.0.0.0 – 10.255.255.255	10.0.0.0/8
B	172.16.0.0 – 172.31.255.255	172.16.0.0/12
C	192.168.0.0 – 192.168.255.255	192.168.0.0/16



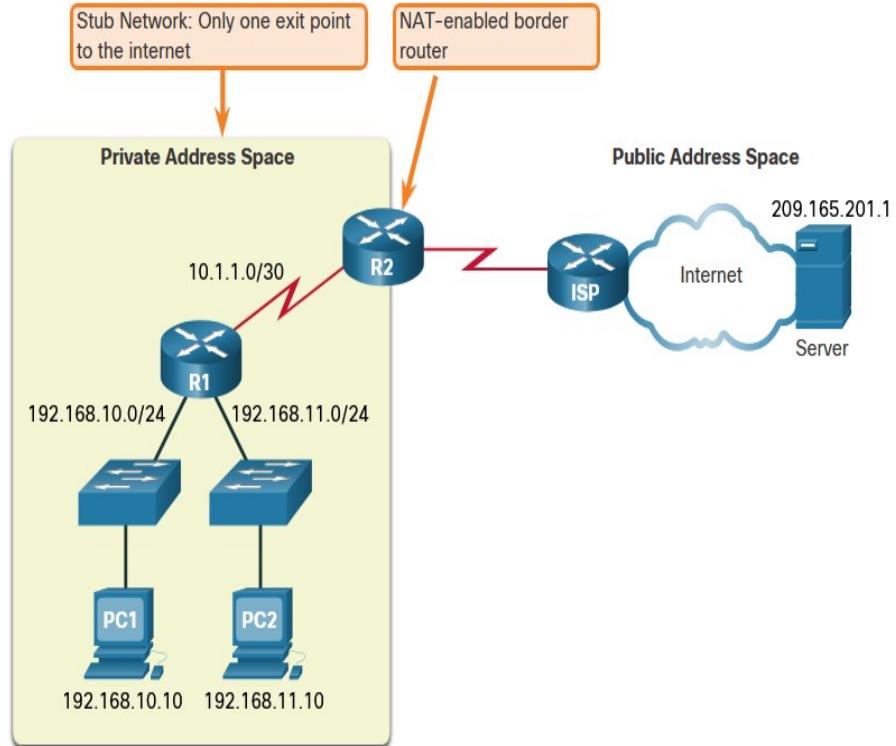
NAT



NAT Characteristics

What is NAT

- The primary use of NAT is to conserve public IPv4 addresses.
- NAT allows networks to use private IPv4 addresses internally and translates them to a public address when needed.
- A NAT router typically operates at the border of a stub network.
- When a device inside the stub network wants to communicate with a device outside of its network, the packet is forwarded to the border router which performs the NAT process, translating the internal private address of the device to a public, outside, routable address.

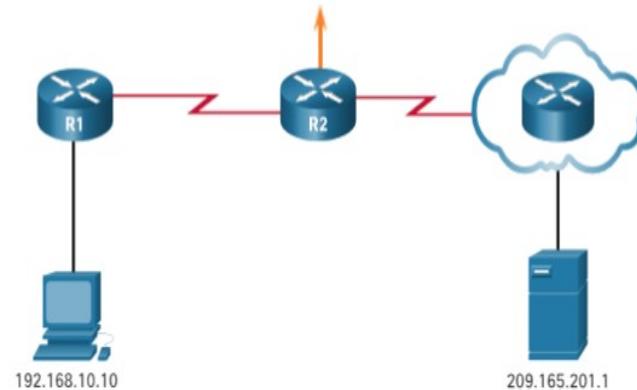


How NAT Works

PC1 wants to communicate with an outside web server with public address 209.165.201.1.

1. PC1 sends a packet addressed to the web server.
2. R2 receives the packet and reads the source IPv4 address to determine if it needs translation.
3. R2 adds mapping of the local to global address to the NAT table.
4. R2 sends the packet with the translated source address toward the destination.
5. The web server responds with a packet addressed to the inside global address of PC1 (209.165.200.226).
6. R2 receives the packet with destination address 209.165.200.226. R2 checks the NAT table and finds an entry for this mapping. R2 uses this information and translates the inside global address (209.165.200.226) to the inside local address (192.168.10.10), and the packet is forwarded toward PC1.

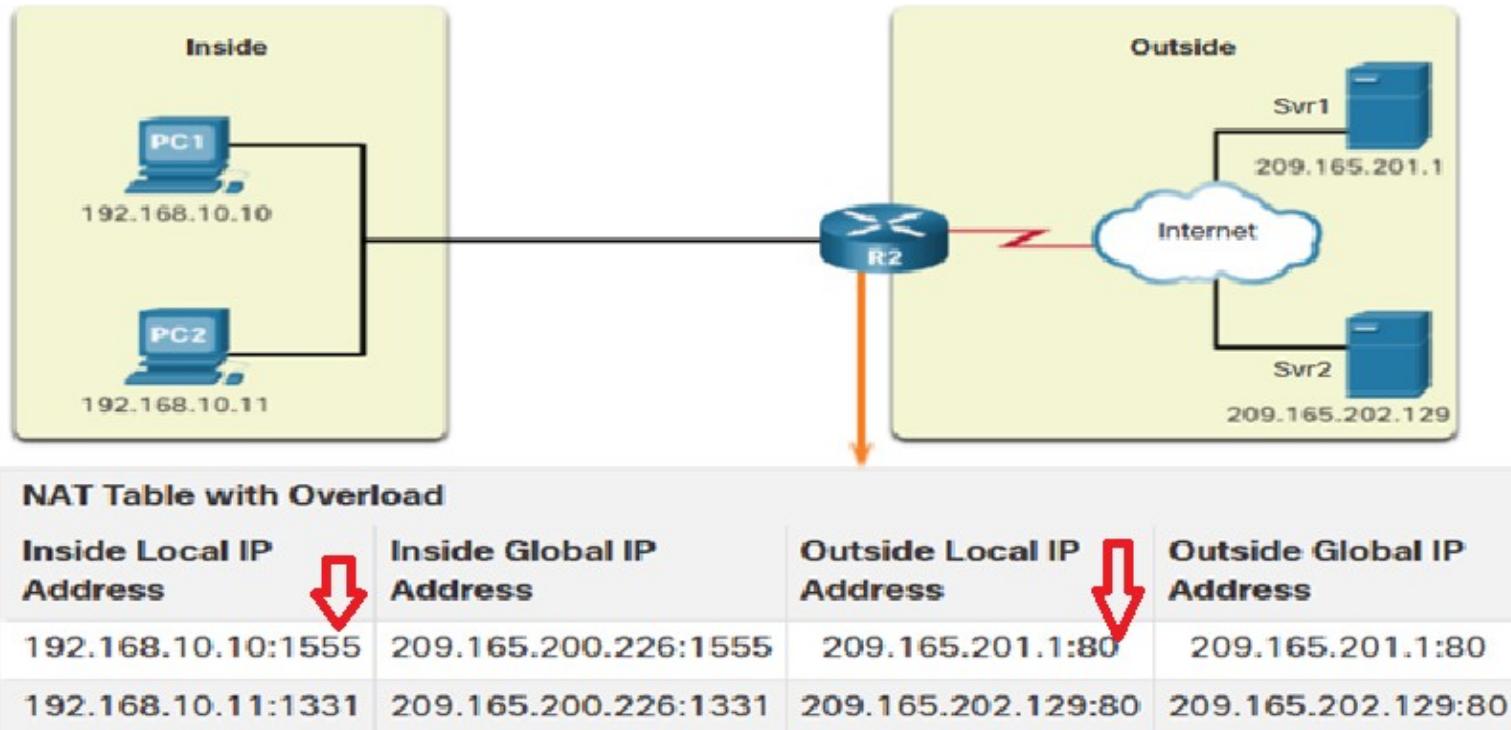
NAT Table			
Inside Local	Inside Global	Outside Local	Outside Global
192.168.10.10	209.165.200.226	209.165.201.1	209.165.201.1



Port Address Translation (PAT)

Port Address Translation (PAT), also known as NAT overload, maps multiple private IPv4 addresses to a single public IPv4 address or a few addresses.

- With PAT, when the NAT router receives a packet from the client, it uses the source port number to uniquely identify the specific NAT translation.
- PAT ensures that devices use a different TCP port number for each session with a server on the internet.



Subnetting

Network Segmentation

Reasons for Subnetting

Subnetting is the process of segmenting a network into multiple smaller network spaces called subnetworks or subnets.

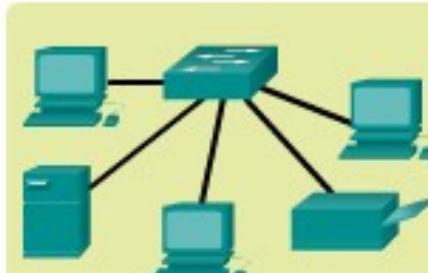
- Large networks must be segmented into smaller subnetworks, creating smaller groups of devices and services to:
 - Control traffic by containing broadcast traffic within each subnetwork.
 - **Reduce overall network traffic** and improve network performance.

Communication Between Subnets

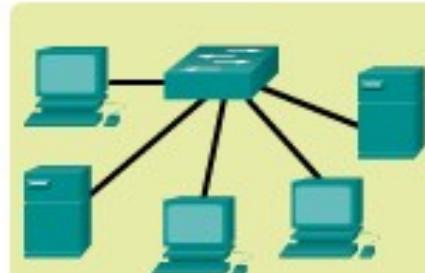
- **A router is necessary for devices** on different networks and subnets to communicate.
- Devices on a network and subnet use the router interface attached to their LAN as their default gateway.

Different networks for different groups

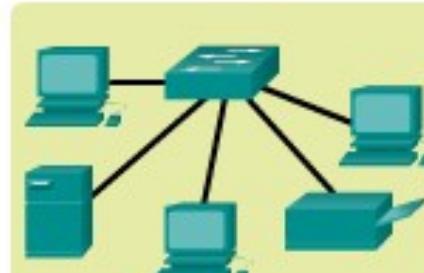
Planning the Network



Student LAN



Faculty LAN

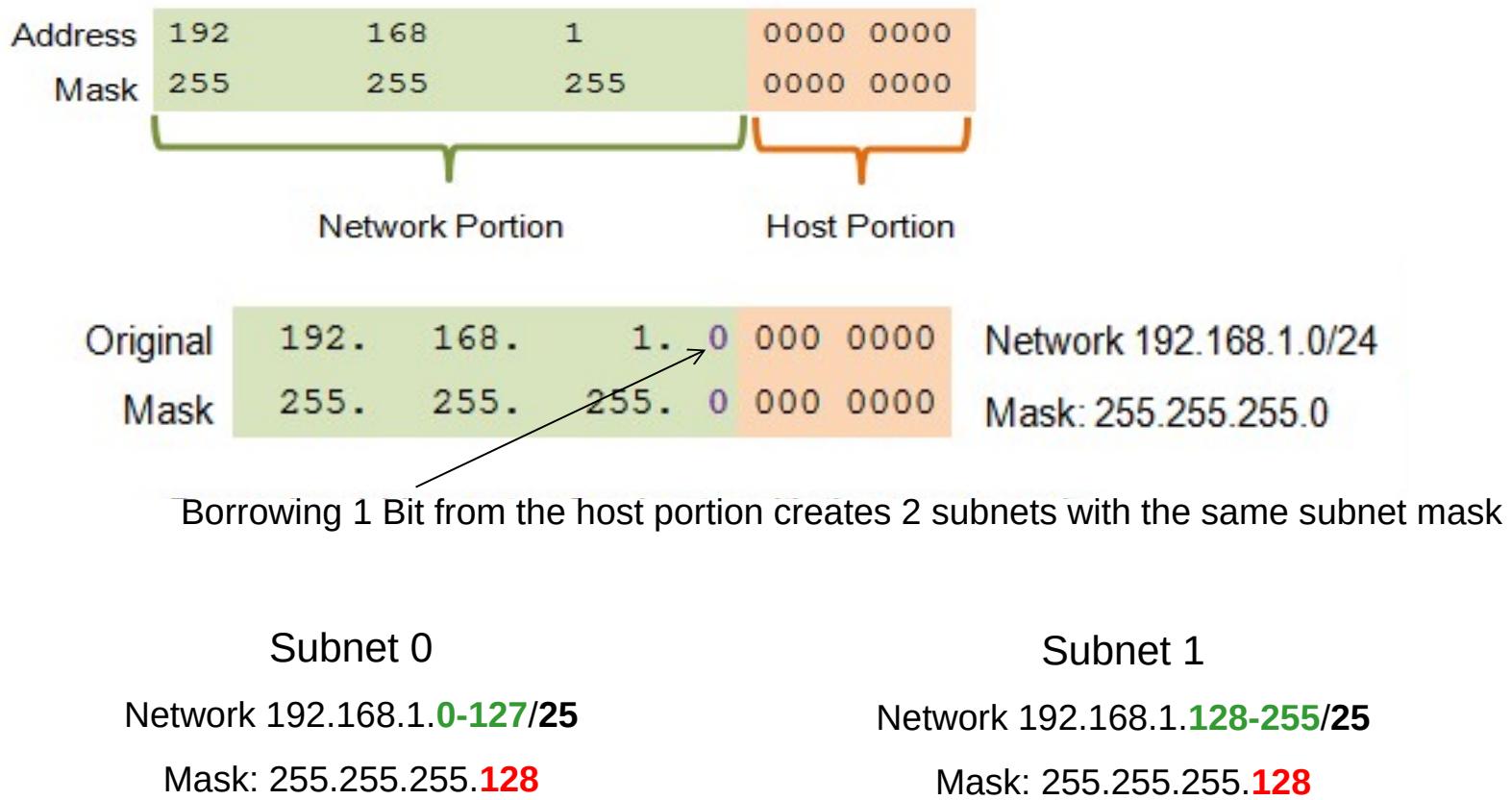


Admin LAN

Planning requires decisions on each subnet in terms of size, the number of hosts per subnet, and how host addresses will be assigned.

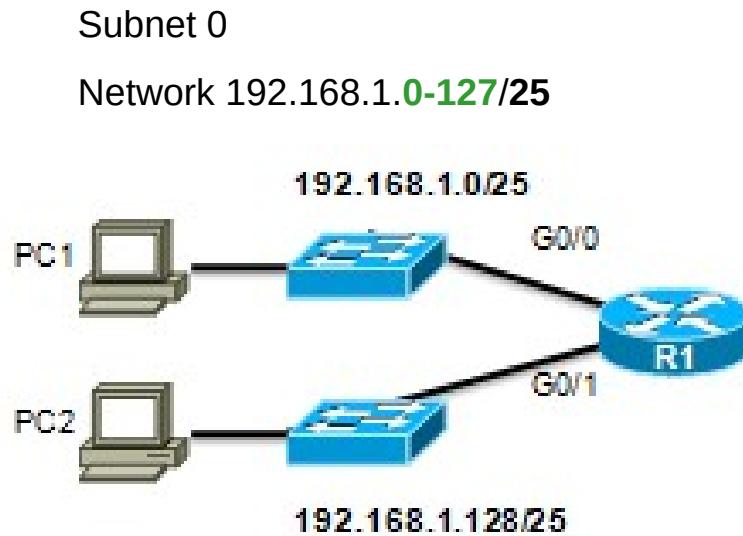
Basic Subnetting

- Borrowing Bits to Create Subnets
- Borrowing 1 bit $2^1 = 2$ subnets



Subnets in Use

Subnets in Use



Subnet 1

Network 192.168.1.128-255/25

Address Range for 192.168.1.0/25 Subnet

Network Address

192. 168. 1. 0 000 0000 = 192.168.1.0

First Host Address

192. 168. 1. 0 000 0001 = 192.168.1.1

Last Host Address

192. 168. 1. 0 111 1110 = 192.168.1.126

Broadcast Address

192. 168. 1. 0 111 1111 = 192.168.1.127

Address Range for 192.168.1.128/25 Subnet

Network Address

192. 168. 1. 1 000 0000 = 192.168.1.128

First Host Address

192. 168. 1. 1 000 0001 = 192.168.1.129

Last Host Address

192. 168. 1. 1 111 1110 = 192.168.1.254

Broadcast Address

192. 168. 1. 1 111 1111 = 192.168.1.255

Subnetting Formulas

Subnets = 2^n
(where n = bits borrowed)

Calculate number of subnets

192. 168. 1. 0 000 0000



1 bit was borrowed

$2^1 = 2$ subnets

Hosts = 2^n
(where n = host bits remaining)

Calculate number of hosts

192. 168. 1. 0 000 0000



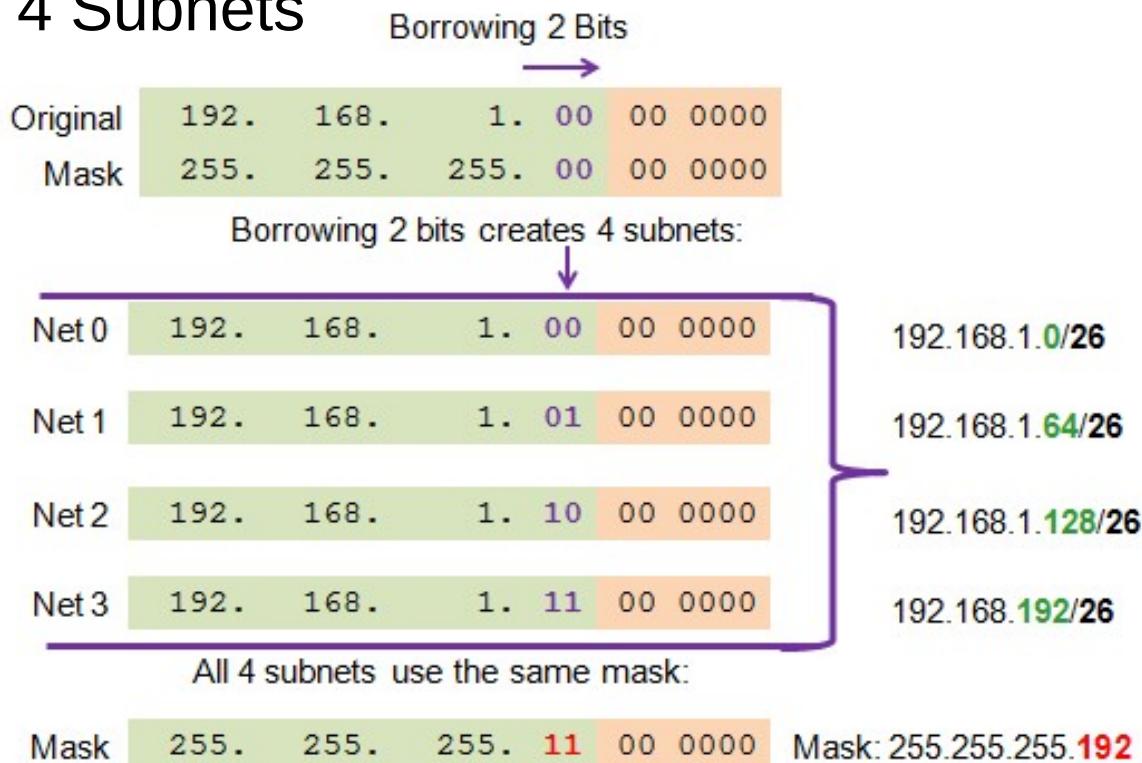
7 bits remain in host field

$2^7 = 128$ addresses per subnet
 $2^7 - 2 = 126$ valid hosts per subnet

Creating 4 Subnets

Borrowing 2 bits to create 4 subnets. $2^2 = 4$ subnets

Creating 4 Subnets



Creating Eight Subnets

Borrowing 3 bits to **Create 8 Subnets**. $2^3 = 8$ subnets

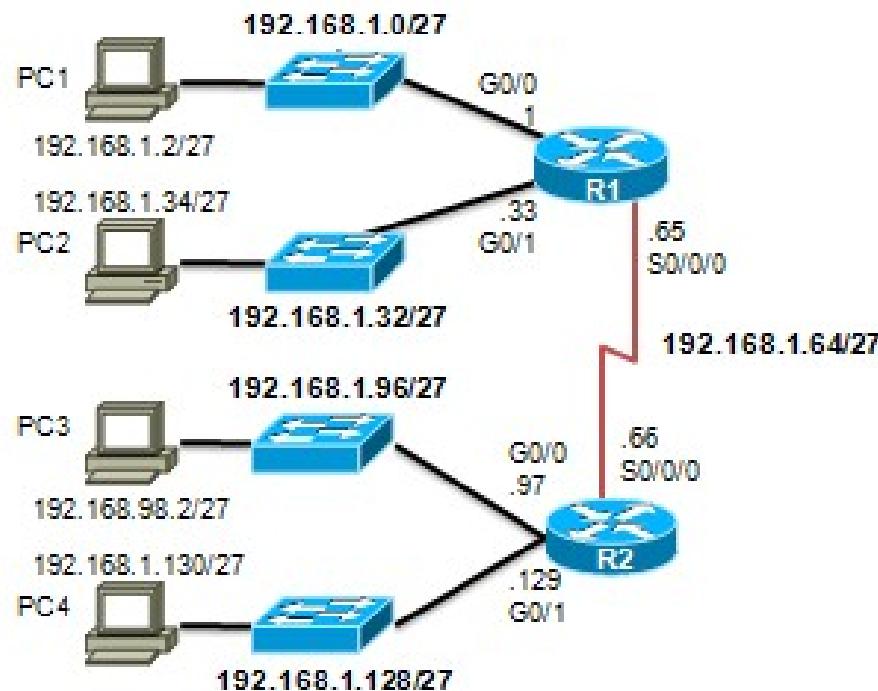
	Network	192.	168.	1.	000	0 0000	192.168.1.0
Net 0	First	192.	168.	1.	000	0 0001	192.168.1.1
	Last	192.	168.	1.	000	1 1110	192.168.1.30
	Broadcast	192.	168.	1.	000	1 1111	192.168.1.31
	Network	192.	168.	1.	001	0 0000	192.168.1.32
Net 1	First	192.	168.	1.	001	0 0001	192.168.1.33
	Last	192.	168.	1.	001	1 1110	192.168.1.62
	Broadcast	192.	168.	1.	001	1 1111	192.168.1.63
	Network	192.	168.	1.	010	0 0000	192.168.1.64
Net 2	First	192.	168.	1.	010	0 0001	192.168.1.65
	Last	192.	168.	1.	010	1 1110	192.168.1.94
	Broadcast	192.	168.	1.	010	1 1111	192.168.1.95
	Network	192.	168.	1.	011	0 0000	192.168.1.96
Net 3	First	192.	168.	1.	011	0 0001	192.168.1.97
	Last	192.	168.	1.	011	1 1110	192.168.1.126
	Broadcast	192.	168.	1.	011	1 1111	192.168.1.127

Creating Eight Subnets (Cont.)

Net 4	Network	192.	168.	1.	100	0 0000	192.168.1.128
	Fist	192.	168.	1.	100	0 0001	192.168.1.129
	Last	192.	168.	1.	100	1 1110	192.168.1.158
	Broadcast	192.	168.	1.	100	1 1111	192.168.1.159
Net 5	Network	192.	168.	1.	101	0 0000	192.168.1.160
	Fist	192.	168.	1.	101	0 0001	192.168.1.161
	Last	192.	168.	1.	101	1 1110	192.168.1.190
	Broadcast	192.	168.	1.	101	1 1111	192.168.1.191
Net 6	Network	192.	168.	1.	110	0 0000	192.168.1.192
	Fist	192.	168.	1.	110	0 0001	192.168.1.193
	Last	192.	168.	1.	110	1 1110	192.168.1.222
	Broadcast	192.	168.	1.	110	1 1111	192.168.1.223
Net 7	Network	192.	168.	1.	111	0 0000	192.168.1.224
	Fist	192.	168.	1.	111	0 0001	192.168.1.225
	Last	192.	168.	1.	111	1 1110	192.168.1.254
	Broadcast	192.	168.	1.	111	1 1111	192.168.1.255

Creating Eight Subnets (Cont.)

Subnet Allocation



Subnetting Based on Host Requirements

Two considerations when planning subnets:

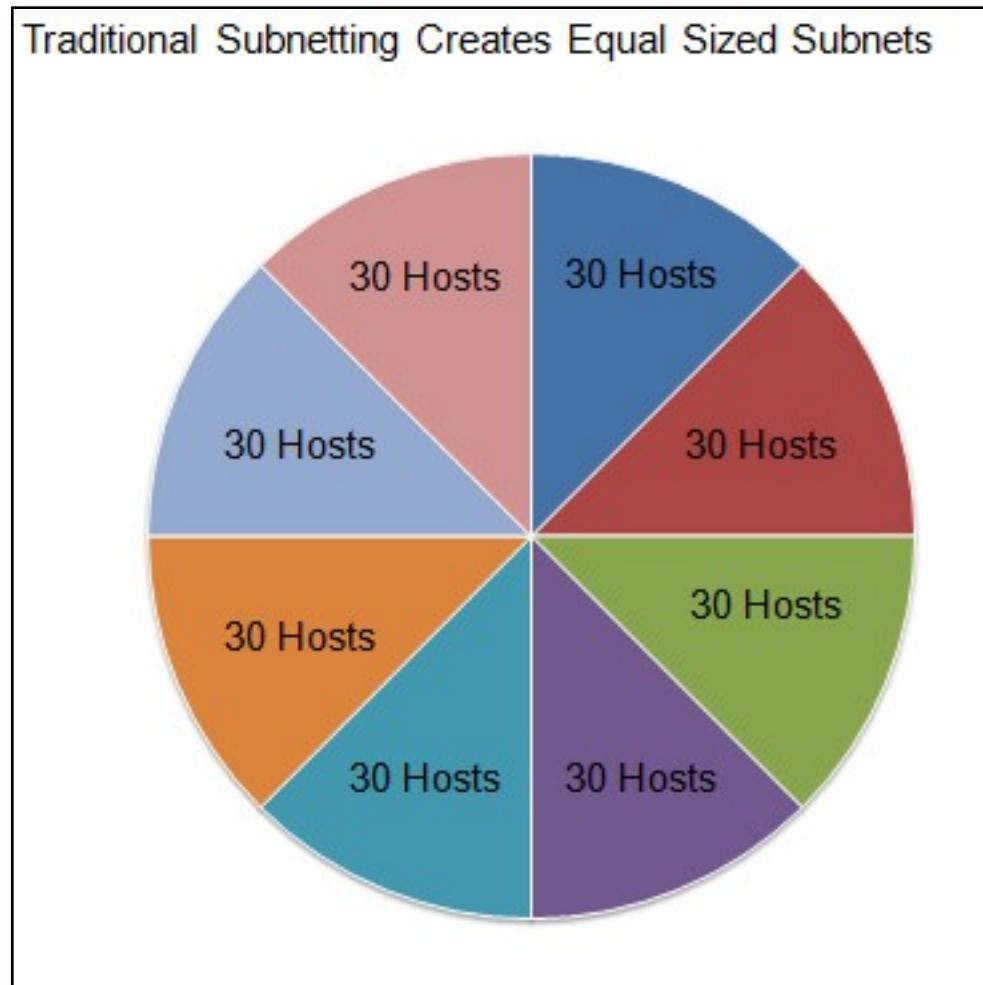
- Number of subnets required
- Number of host addresses required

Formula to determine number of usable hosts: $2^n - 2$

- 2^n (where n is the number of remaining host bits) is used to calculate the number of hosts.
- -2 (The subnetwork ID and broadcast address cannot be used on each subnet.)

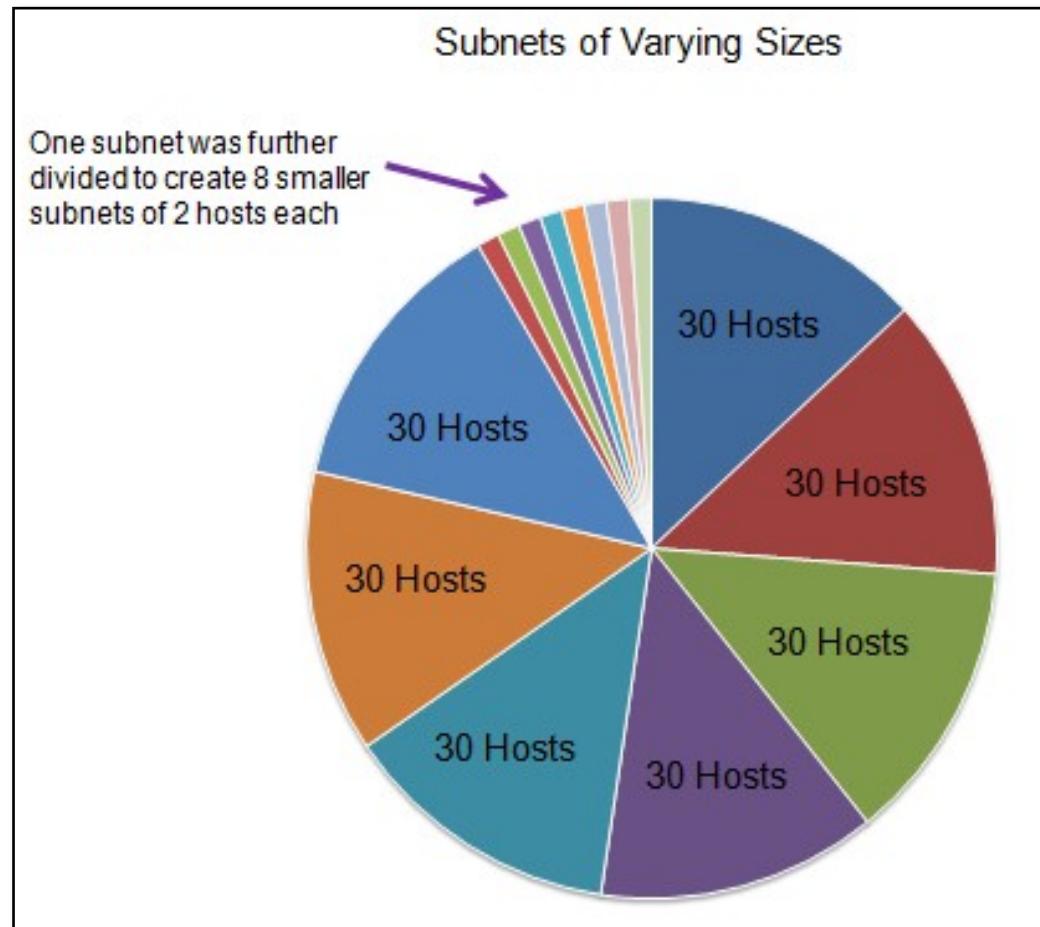
Traditional Subnetting: Equal Addresses

- Subnets that require fewer addresses have **unused (wasted)** addresses
- For example, WAN links only need two addresses.



Variable Length Subnet Masks (VLSM)

- The **variable-length subnet mask (VLSM)** or subnetting a subnet provides more efficient use of addresses.
- VLSM allows a network space to be divided in **unequal parts**.
- Subnet mask varies, depending on how many bits have been borrowed for a particular subnet.
- Network is first subnetted, and then the **subnets are resubnetted**.



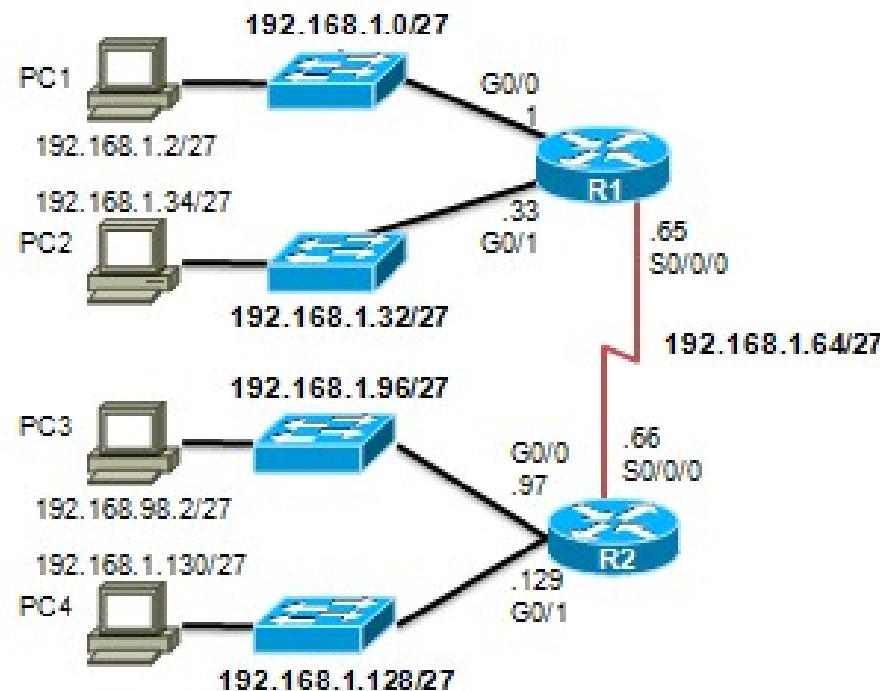
Benefits of Variable Length Subnet Masking

Basic VLSM

VLSM Subnetting Scheme							
11000000.10101000.00010100 .000 00000 192.168.20.0/24							
0	11000000.10101000.00010100	.000	00000	192.168.20.0/27			
1	11000000.10101000.00010100	.001	00000	192.168.20.32/27			
2	11000000.10101000.00010100	.010	00000	192.168.20.64/27			
3	11000000.10101000.00010100	.011	00000	192.168.20.96/27			
4	11000000.10101000.00010100	.100	00000	192.168.20.128/27			
5	11000000.10101000.00010100	.101	00000	192.168.20.160/27			
6	11000000.10101000.00010100	.110	00000	192.168.20.192/27			
7	11000000.10101000.00010100	.111	00000	192.168.20.224/27			
3 more bits borrowed from subnet 7:							
7:0	11000000.10101000.00010100	.111000	00	192.168.20.224/30			
7:1	11000000.10101000.00010100	.111001	00	192.168.20.228/30			
7:2	11000000.10101000.00010100	.111010	00	192.168.20.232/30			
7:3	11000000.10101000.00010100	.111011	00	192.168.20.236/30			
7:4	11000000.10101000.00010100	.111100	00	192.168.20.240/30			
7:5	11000000.10101000.00010100	.111101	00	192.168.20.244/30			
7:6	11000000.10101000.00010100	.111110	00	192.168.20.248/30			
7:7	11000000.10101000.00010100	.111111	00	192.168.20.252/30			

Creating Eight Subnets (Cont.)

Subnet Allocation



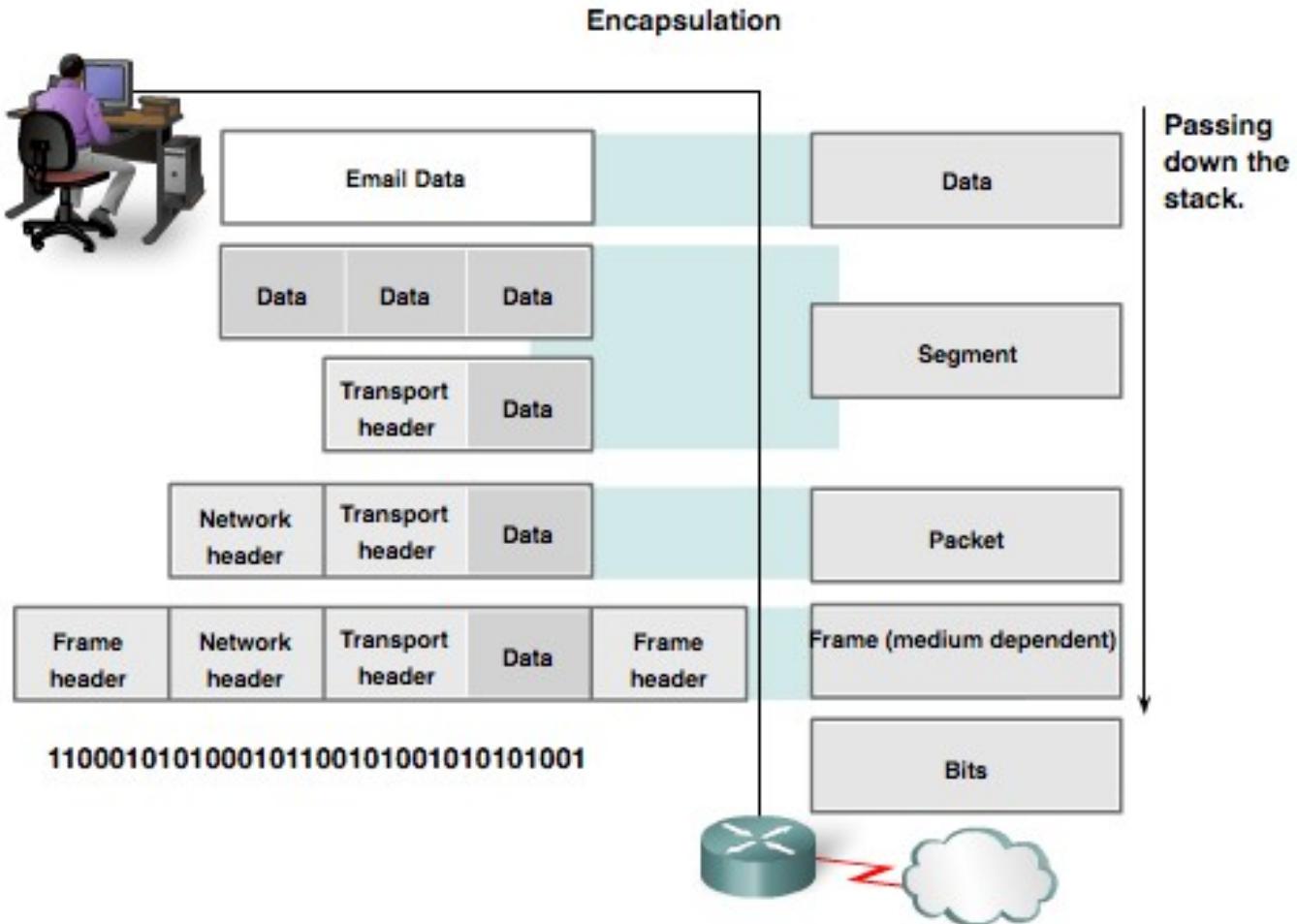
Let us do another example ..

Chapter 5

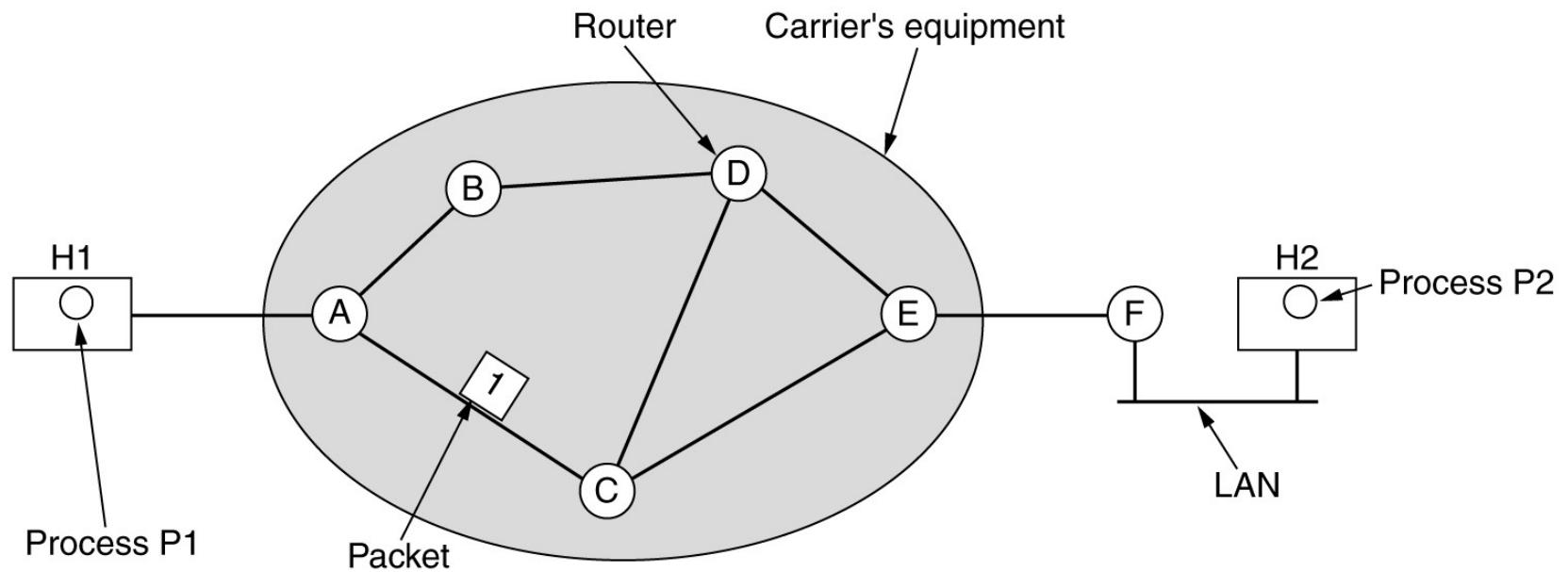
The Network Layer

Protocol Data Units (PDUs)

- a) Data
- b) Segment
- c) Packet
- d) Frame
- e) Bits



Store-and-Forward Packet Switching



The environment of the network layer protocols.

Circuit Switching

- Dedicated communication path between two stations

- Three phases

- Establish

- Transfer

- Disconnect

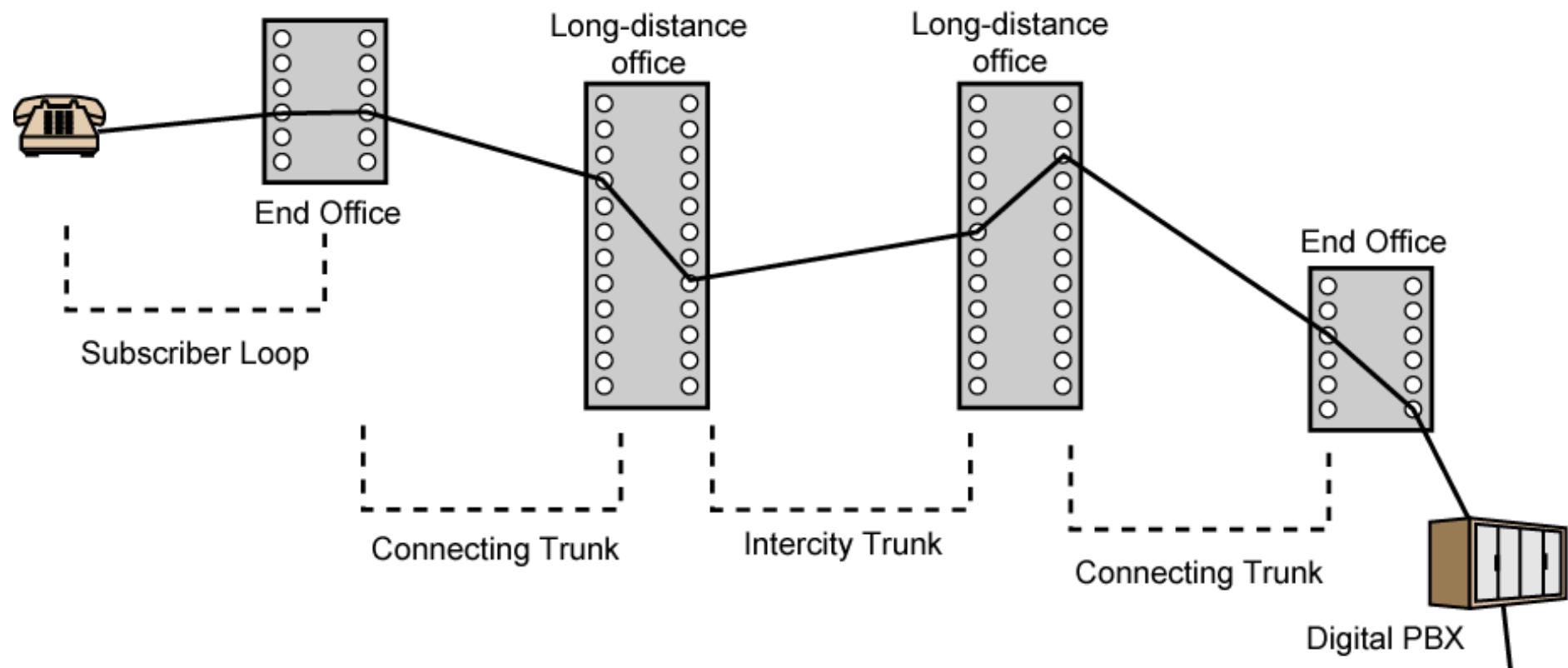
- Must have switching capacity and channel capacity to establish connection

- Must have intelligence to work out routing

Circuit Switching - Applications

- Inefficient
 - Channel capacity dedicated for duration of connection
 - If no data, capacity wasted
 - Set up (connection) takes time
 - Once connected, transfer is transparent
 - Developed for **voice traffic (phone)**

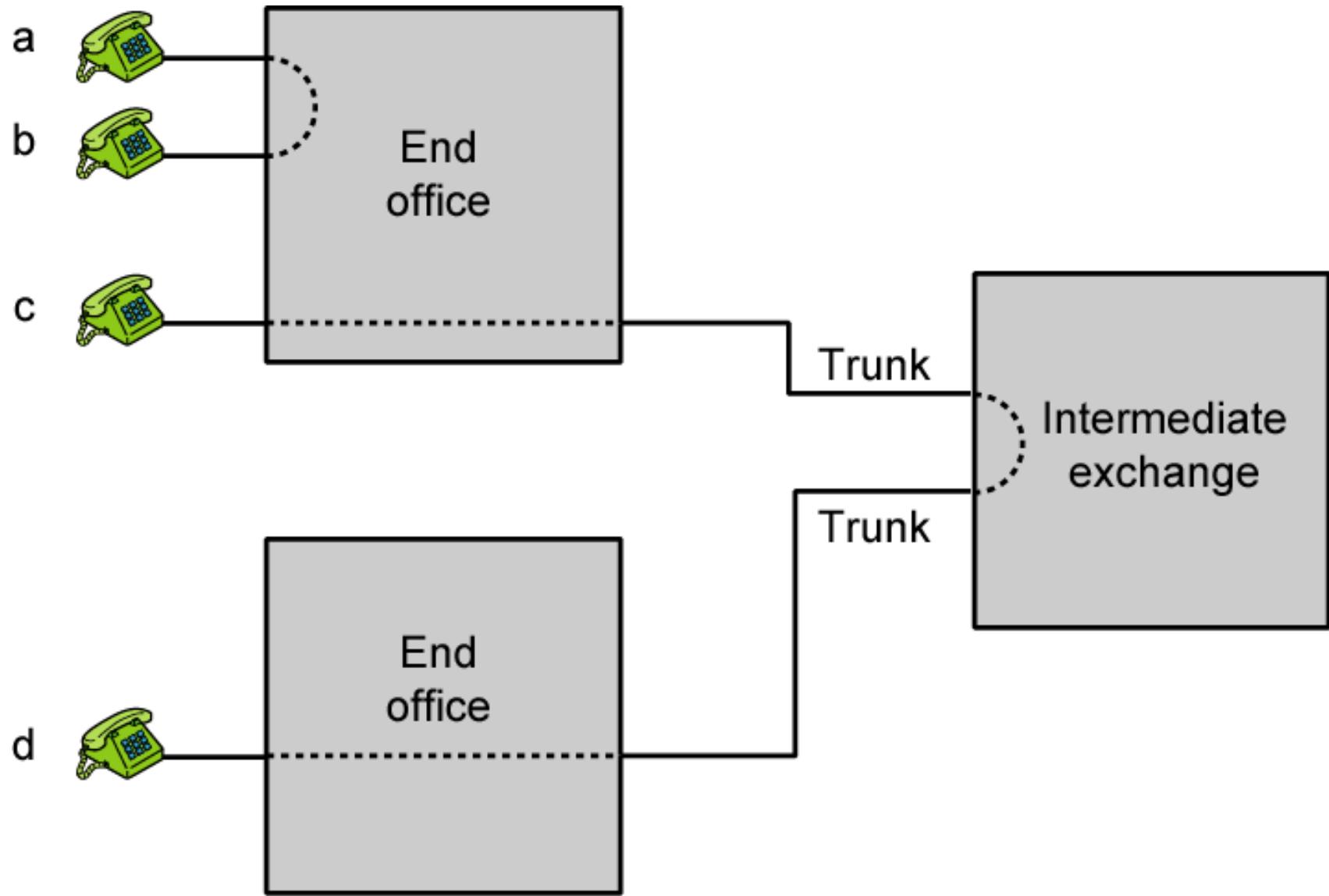
Public Circuit Switched Network



Telecomms Components

- Subscriber
 - Devices attached to network
- Subscriber line
 - Local Loop
 - Subscriber loop
 - Connection to network
 - Few km up to few tens of km
- Exchange
 - Switching centers
 - End office - supports subscribers
- Trunks
 - Branches between exchanges
 - Multiplexed

Circuit Establishment



Packet Switching: Basic Operation

- Data transmitted in small packets

- Typically 1000 octets

- Longer messages **split** into series of **packets**

- Each packet contains a portion of user data plus some control info

- Control info

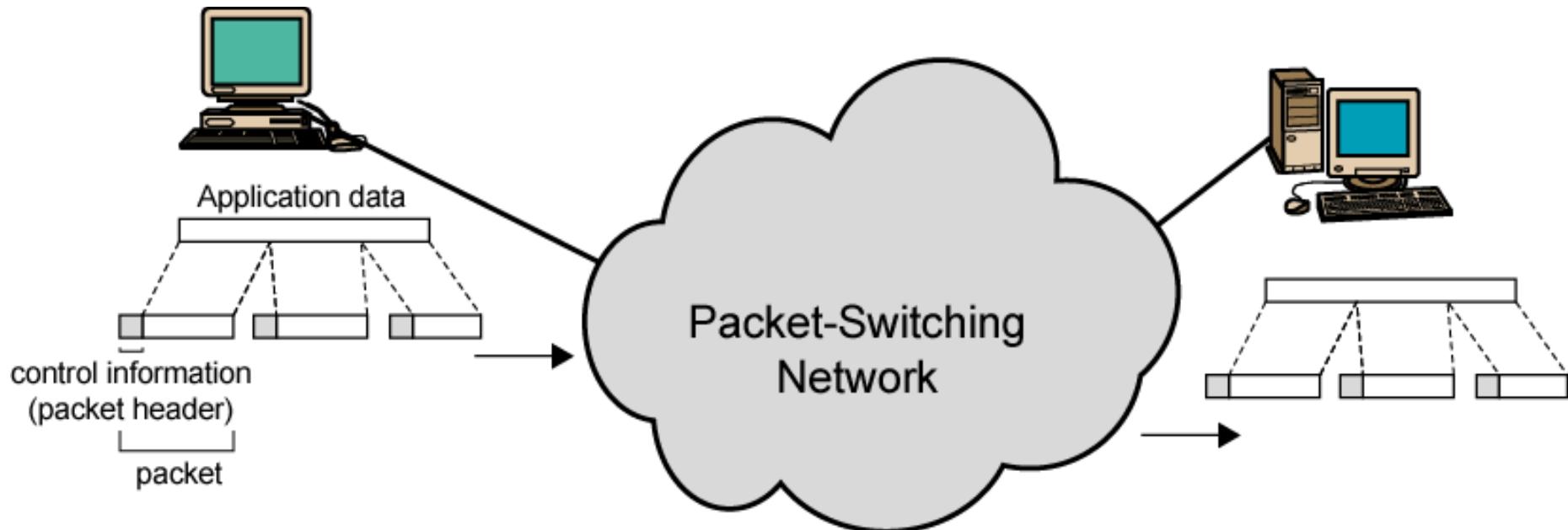
- Routing (addressing) info

- Packets are received, stored briefly (buffered) and passed on to the next node

- Store and forward

-

Use of Packets



Advantages

- Line efficiency

- Single node to node link can be shared by many packets over time

- Packets queued and transmitted as fast as possible

- Data rate conversion

- Each station connects to the local node at its own speed

- Nodes buffer data if required to equalize rates

- Packets are accepted even when network is busy

- Delivery may slow down

- Priorities can be used

Switching Technique

- a) Station breaks long message into packets
- b) Packets sent one at a time to the network
- c) Packets handled in two ways

Datagram

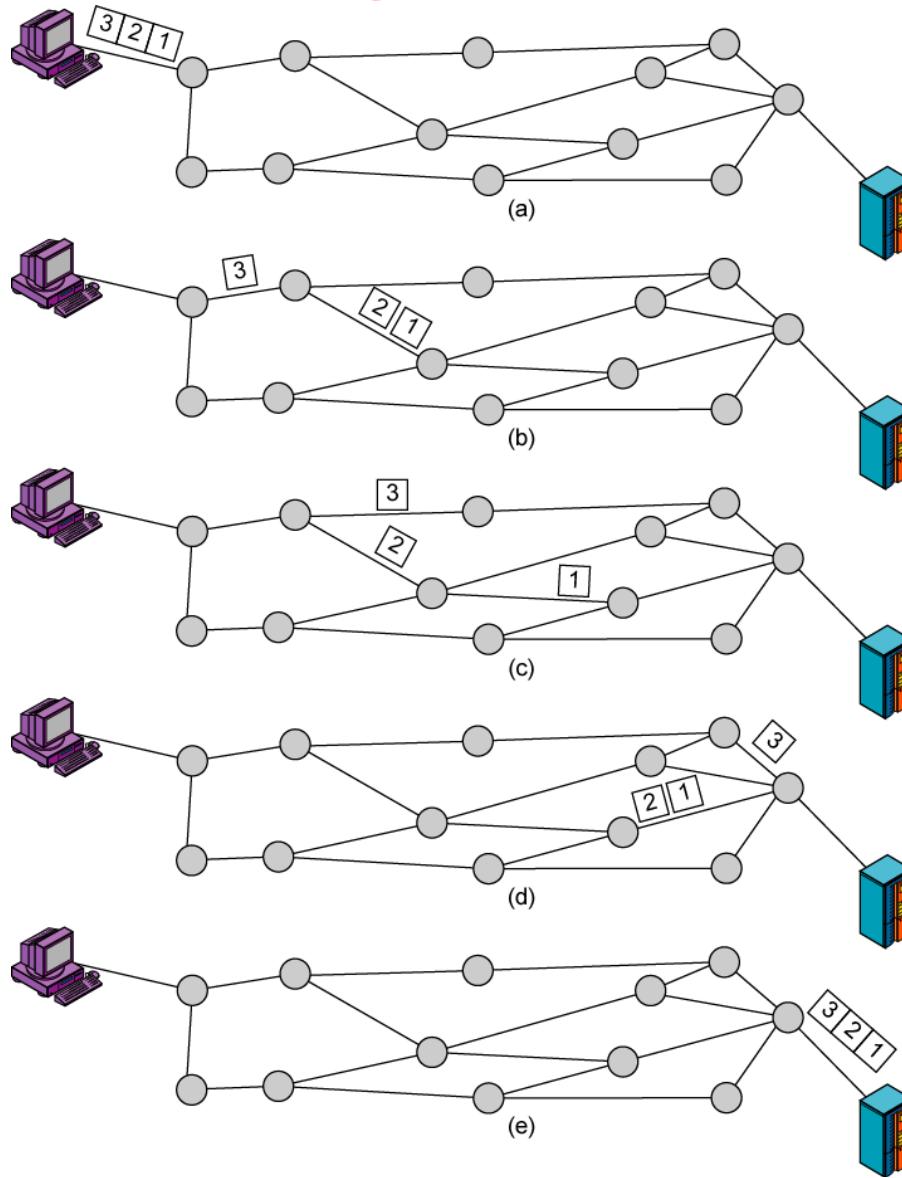
— **Virtual circuit**

—

Datagram

- Each packet treated independently
- Packets can take [any](#) practical route
- Packets may arrive **out of order**
- Packets may go **missing**
- Up to receiver to [re-order packets](#) and recover from missing packets
-

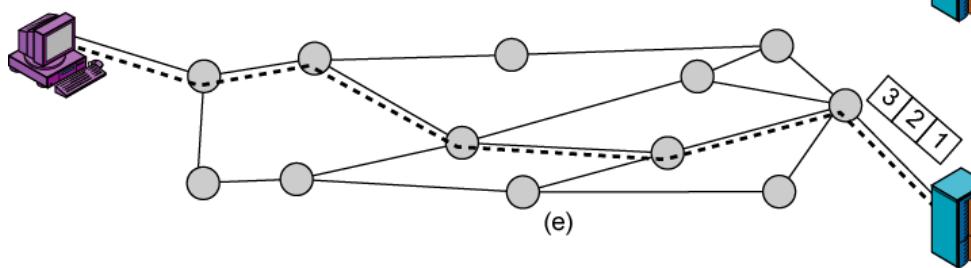
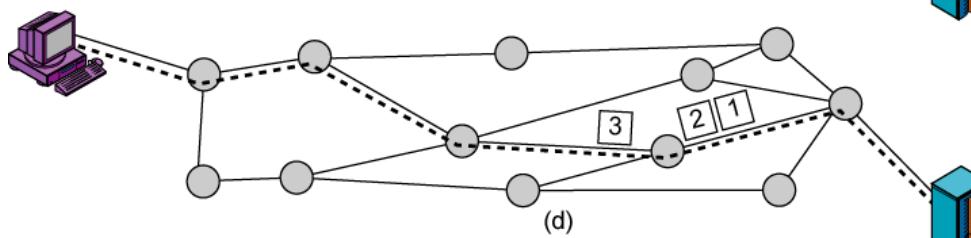
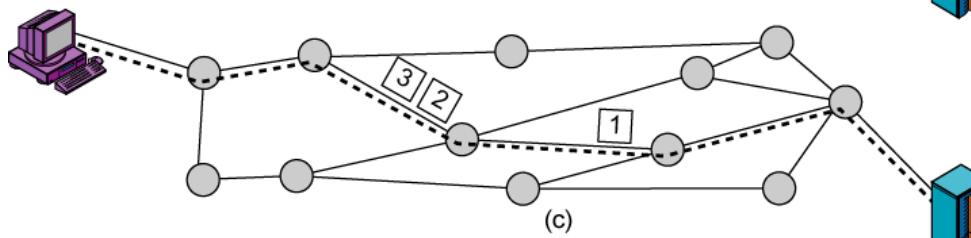
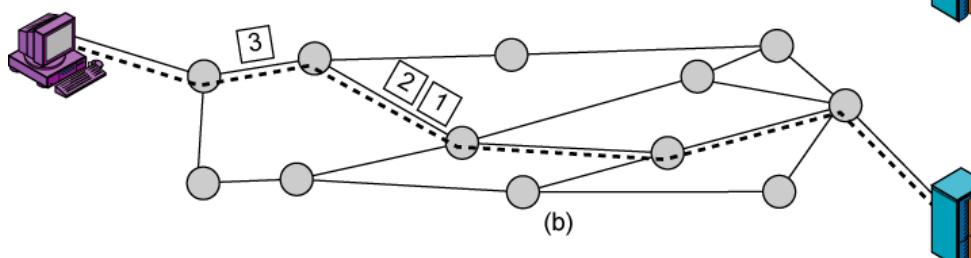
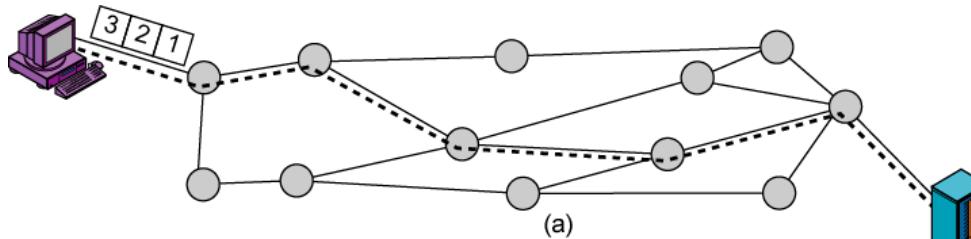
Datagram Example



Virtual Circuit

- Preplanned route established before any packets sent
- Call request and call accept packets establish connection (handshake)
- Each packet contains a virtual circuit identifier instead of destination address
- No routing decisions required for each packet
- Clear request to drop circuit
- Not a dedicated path: **sharing possible**

Virtual Circuit



Virtual Circuits vs. Datagram

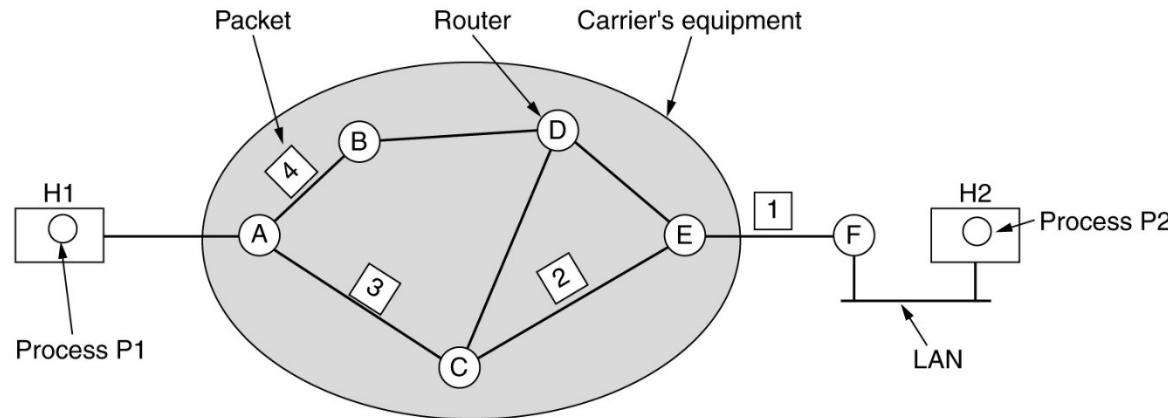
Virtual circuits

- Network can provide sequencing and error control
- Packets are forwarded more quickly
 - No routing decisions to make
- Less reliable
 - Loss of a node loses all circuits through that node

Datagram

- No call setup phase
- Better if few packets
- More flexible
 - Routing can be used to avoid congested parts of the network

Routing in Datagram Subnet

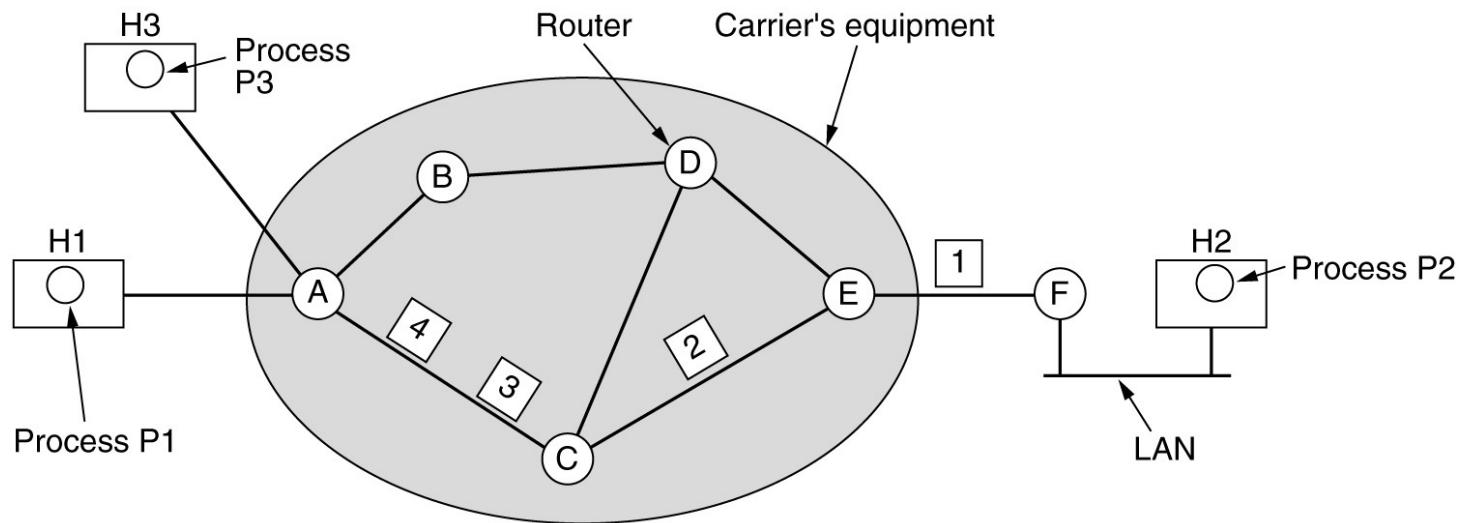


A's table

initially	later	C's table	E's table
A -	A -	A A	A C
B B	B B	B A	B D
C C	C C	C -	C C
D B	D B	D D	D D
E C	E B	E E	E -
F C	F B	F E	F F

Dest. Line

Routing in Virtual Circuit



A's table		C's table		E's table	
H1	1	C	1	A	1
H3	1	C	2	E	1
In		Out		In	

Comparison of Virtual-Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Main Tasks of Network Layer

The network layer, or OSI Layer 3, provides services to allow end devices to exchange data across the network. To accomplish this end-to-end transport, the network layer uses four basic processes:

- Addressing end devices --- Already covered
- Encapsulation
- Routing => Will be covered next
- De-encapsulating

Routing Algorithms

- The Optimality Principle
- Shortest Path Routing
- Flooding
- Distance Vector Routing
- Link State Routing
- Hierarchical Routing
- Broadcast Routing
- Multicast Routing
- Routing for Mobile Hosts
- Routing in Ad Hoc Networks

Static and Dynamic Routing

Reach Remote Networks

A router can learn about remote networks in one of two ways:

- **Manually** - Remote networks are manually entered into the route table using static routes.
- **Dynamically** - Remote routes are automatically learned using a dynamic routing protocol.

Static Routing

Advantages	Disadvantages
Easy to implement in a small network.	Suitable only for simple topologies or for special purposes such as a default static route. Configuration complexity increases dramatically as network grows.
Very secure. No advertisements are sent as compared to dynamic routing protocols.	
Route to destination is always the same.	Manual intervention required to re-route traffic.
No routing algorithm or update mechanism required; therefore, extra resources (CPU or RAM) are not required.	

Use of Static Routes

- Providing ease of routing table maintenance in smaller networks that are not expected to grow significantly.
- Routing to and from **stub networks**. A stub network is a network accessed by a single route, and the router has no other neighbors.
- Using **a single default route** : Default routes are used to send traffic to any destination beyond the next upstream router.

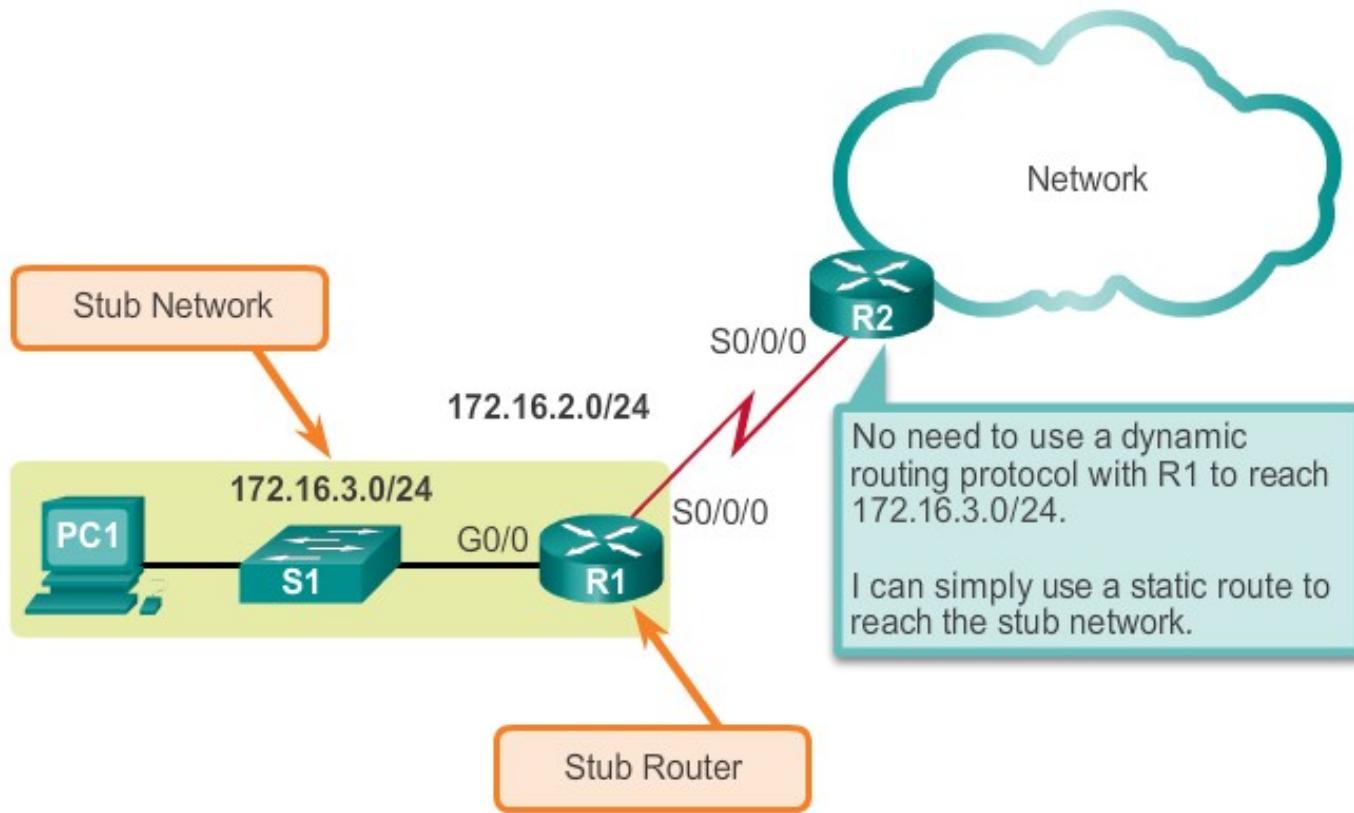
Static Route Applications

Static Routes are often used to:

- Connect to a specific network.
- Provide a Gateway of Last Resort for a stub network.
- Reduce the number of routes advertised by summarizing several contiguous networks as one static route.
- Create a backup route in case a primary route link fails.

Standard Static Route

Connecting to a Stub Network

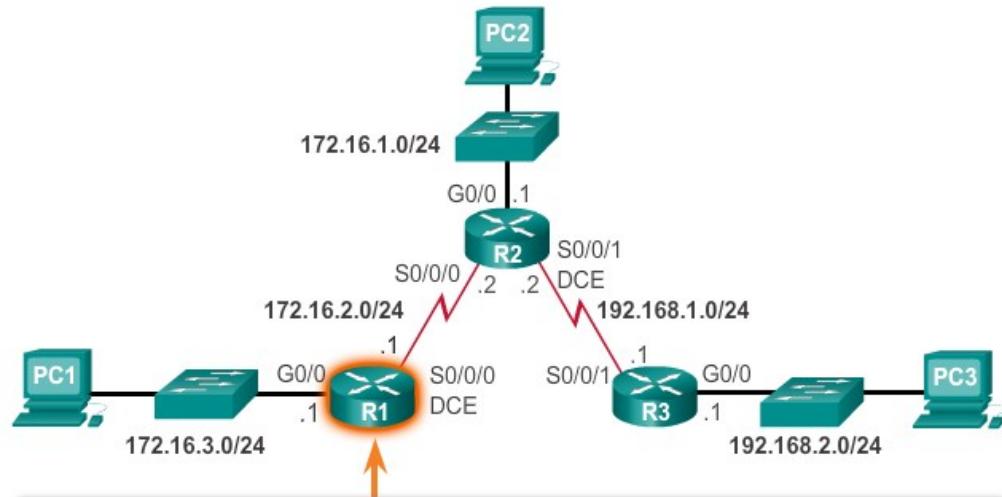


Default Static Route

- A default static route is a route that matches all packets.
- A default route identifies the gateway IP address to which the router sends all IP packets that it does not have a learned or static route.
- A default static route is simply a static route with **0.0.0.0/0** as the destination IPv4 address.

Configure Static Route

Configure Directly Attached Static Routes on R1



```
R1(config) #ip route 172.16.1.0 255.255.255.0 s0/0/0
R1(config) #ip route 192.168.1.0 255.255.255.0 s0/0/0
R1(config) #ip route 192.168.2.0 255.255.255.0 s0/0/0
R1(config) #
```

```
S      172.16.1.0/24 is directly connected, Serial0/0/0
C      172.16.2.0/24 is directly connected, Serial0/0/0
L      172.16.2.1/32 is directly connected, Serial0/0/0
C      172.16.3.0/24 is directly connected, GigabitEthernet0/0
L      172.16.3.1/32 is directly connected, GigabitEthernet0/0
S      192.168.1.0/24 is directly connected, Serial0/0/0
S      192.168.2.0/24 is directly connected, Serial0/0/0
R1#
```

ip route Command to configure

ip route Command Syntax

```
Router(config)#ip route network-address subnet-mask  
{ip-address | exit-intf}
```

Parameter	Description
network-address	Destination network address of the remote network to be added to the routing table.
subnet-mask	<ul style="list-style-type: none">Subnet mask of the remote network to be added to the routing table.The subnet mask can be modified to summarize a group of networks.
ip-address	<ul style="list-style-type: none">Commonly referred to as the next-hop router's IP address.Typically used when connecting to a broadcast media (i.e., Ethernet).Commonly creates a recursive lookup.
exit-intf	<ul style="list-style-type: none">Use the outgoing interface to forward packets to the destination network.Also referred to as a directly attached static route.Typically used when connecting in a point-to-point configuration.

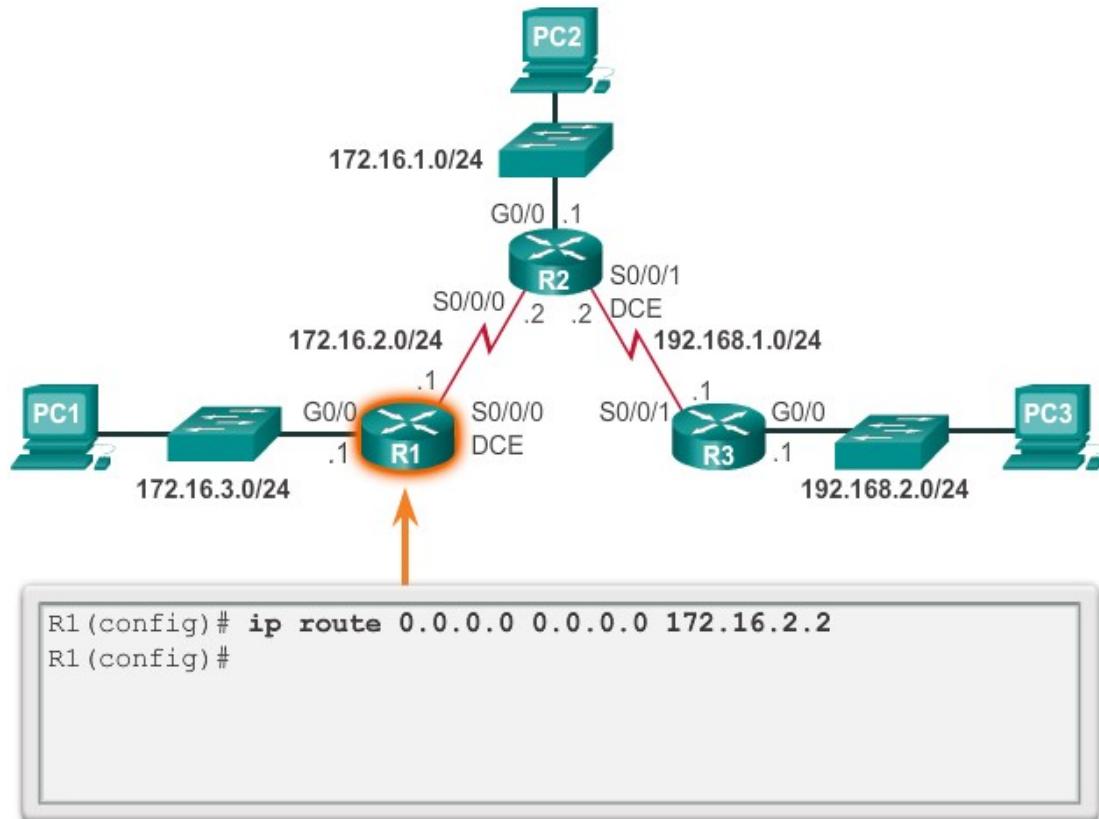
Verify a Static Route

Along with **ping** and **traceroute**, useful commands to verify static routes include:

- **show ip route**
- **show ip route static**
- **show ip route** network

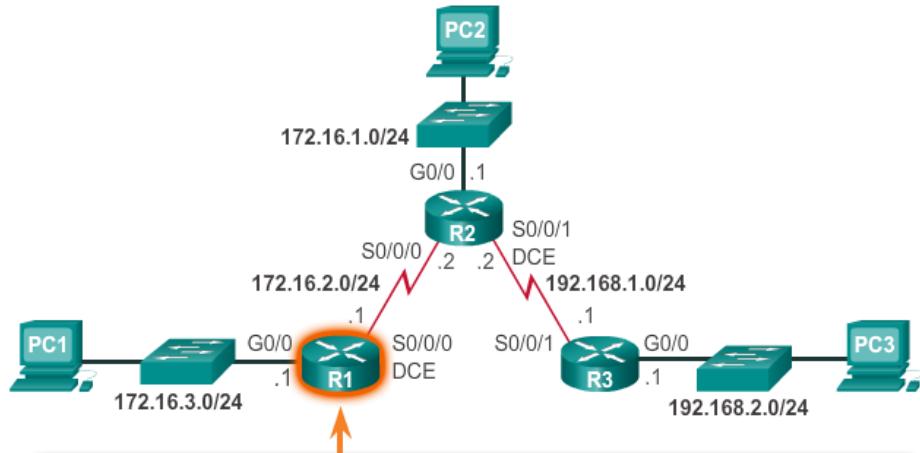
Configure a Default Static Route

Configuring a Default Static Route



Verify a Default Static Route

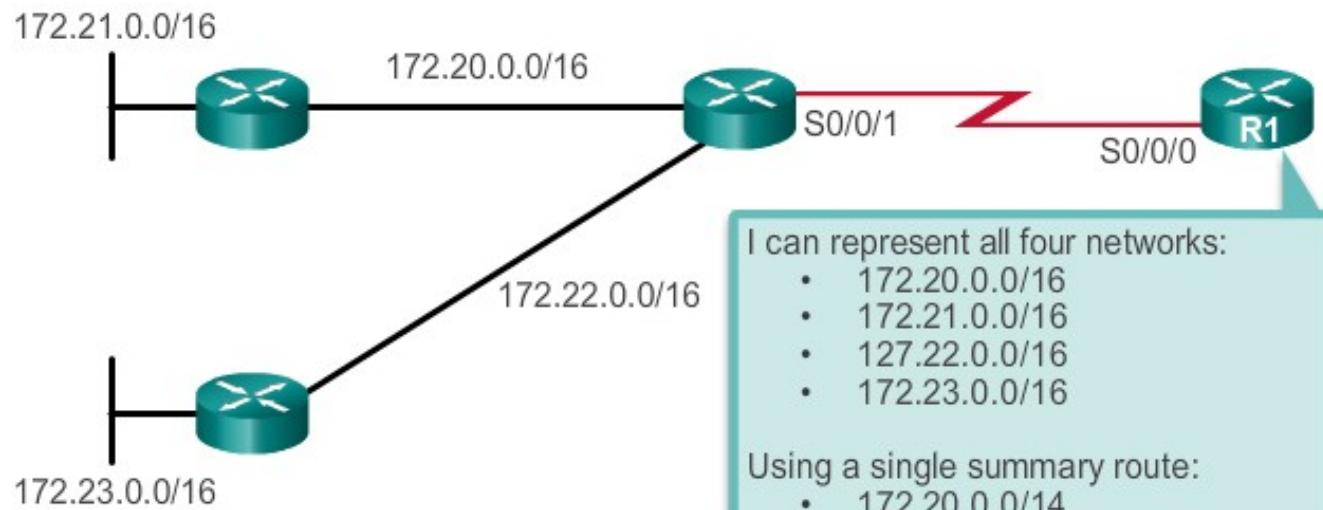
Verifying the Routing Table of R1



```
R1#show ip route static
Codes: L - local, C - connected, S - static, R - RIP,
      M - mobile, B - BGP, D - EIGRP,
      EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA extern
      N2 - OSPF NSSA extern
      E1 - OSPF external typ
      E2 - OSPF external typ
      su - IS-IS summarization, L
      * - candidate default, U - per-user static route
      o - ODR, P - periodic downloaded static route,
      H - NHRP, l - LISP, + - replicated route,
      % - next hop override
      2   Gateway of last resort is 172.16.2.2 to network 0.0.0.0
      1   S*   0.0.0.0/0 [1/0] via 172.16.2.2
R1#
```

Summary Static Route

Using One Summary Static Route

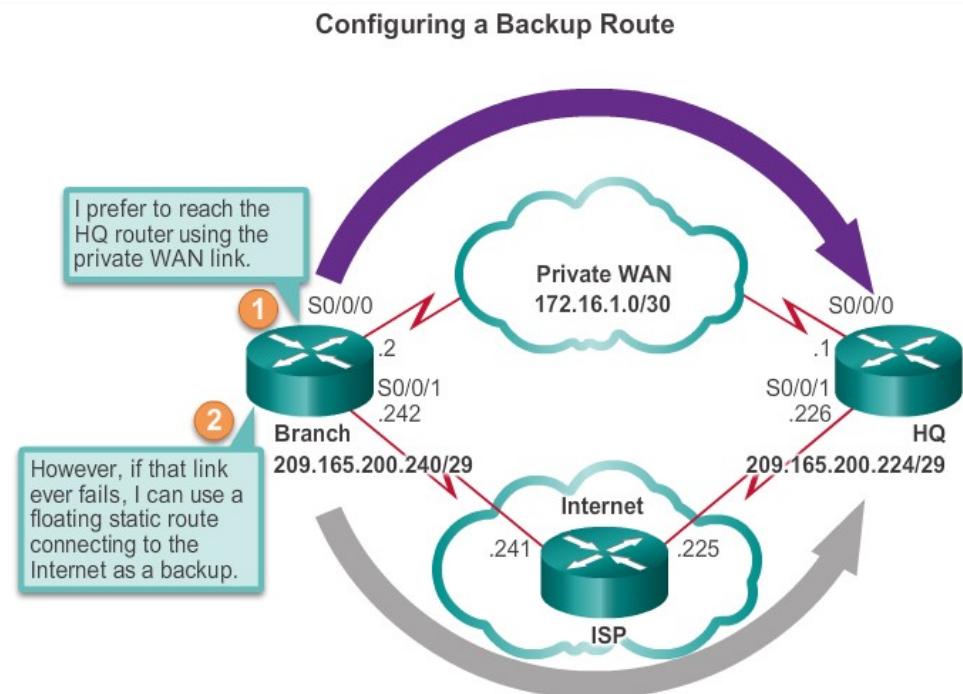


Floating Static Routes

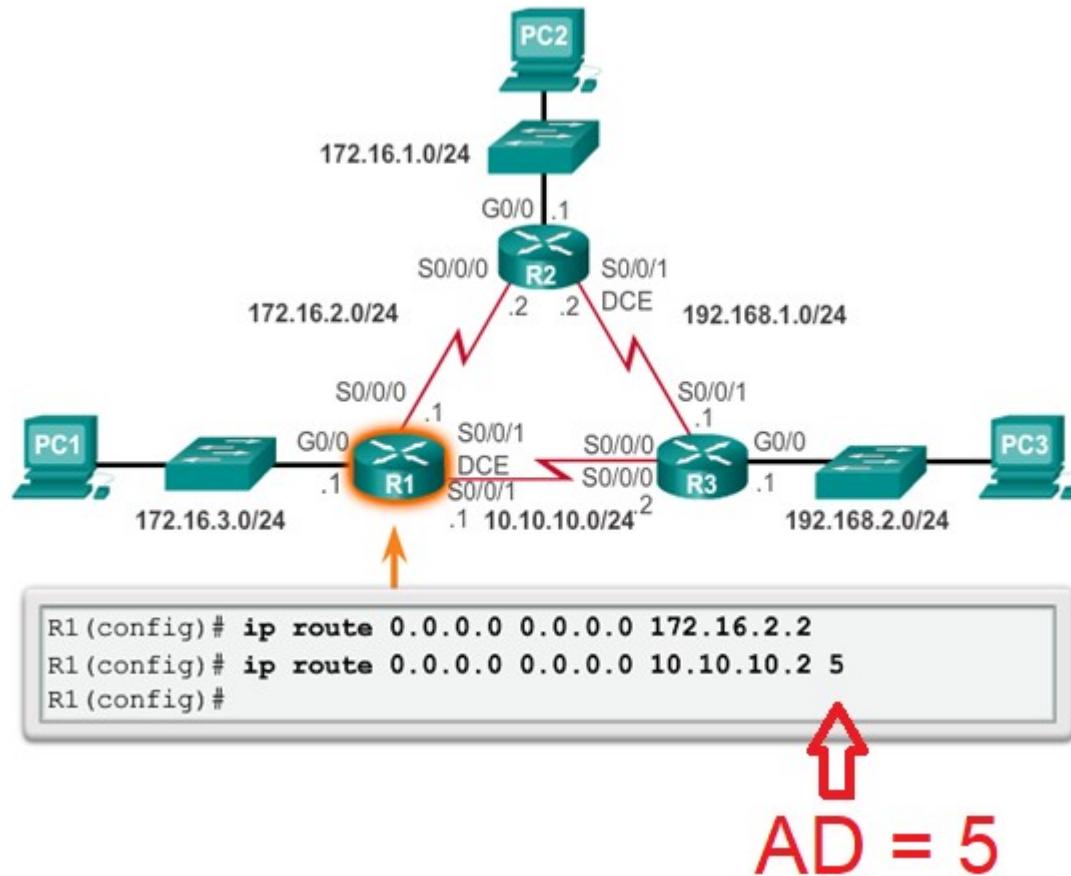
- Floating static routes are static routes that have an administrative distance **greater than** the administrative distance of another static route or dynamic routes:
- The **administrative distance** of a static route can be increased to make the route less desirable than that of another static route or a route learned through a dynamic routing protocol.
- In this way, the static route “floats” and is not used when the route with the better administrative distance is active.
- However, **if the preferred route is lost**, the floating static route can take over, and traffic can be sent through this alternate route.

Floating Static Route

- Floating static routes are static routes that are used to provide a backup path to a primary static or dynamic route, in the event of a link failure.
- The floating static route is only used when the primary route is not available.
- To accomplish this, the floating static route is configured with a higher administrative distance than the primary route.



Configure a Floating Static Route



Dynamic Routing Protocols

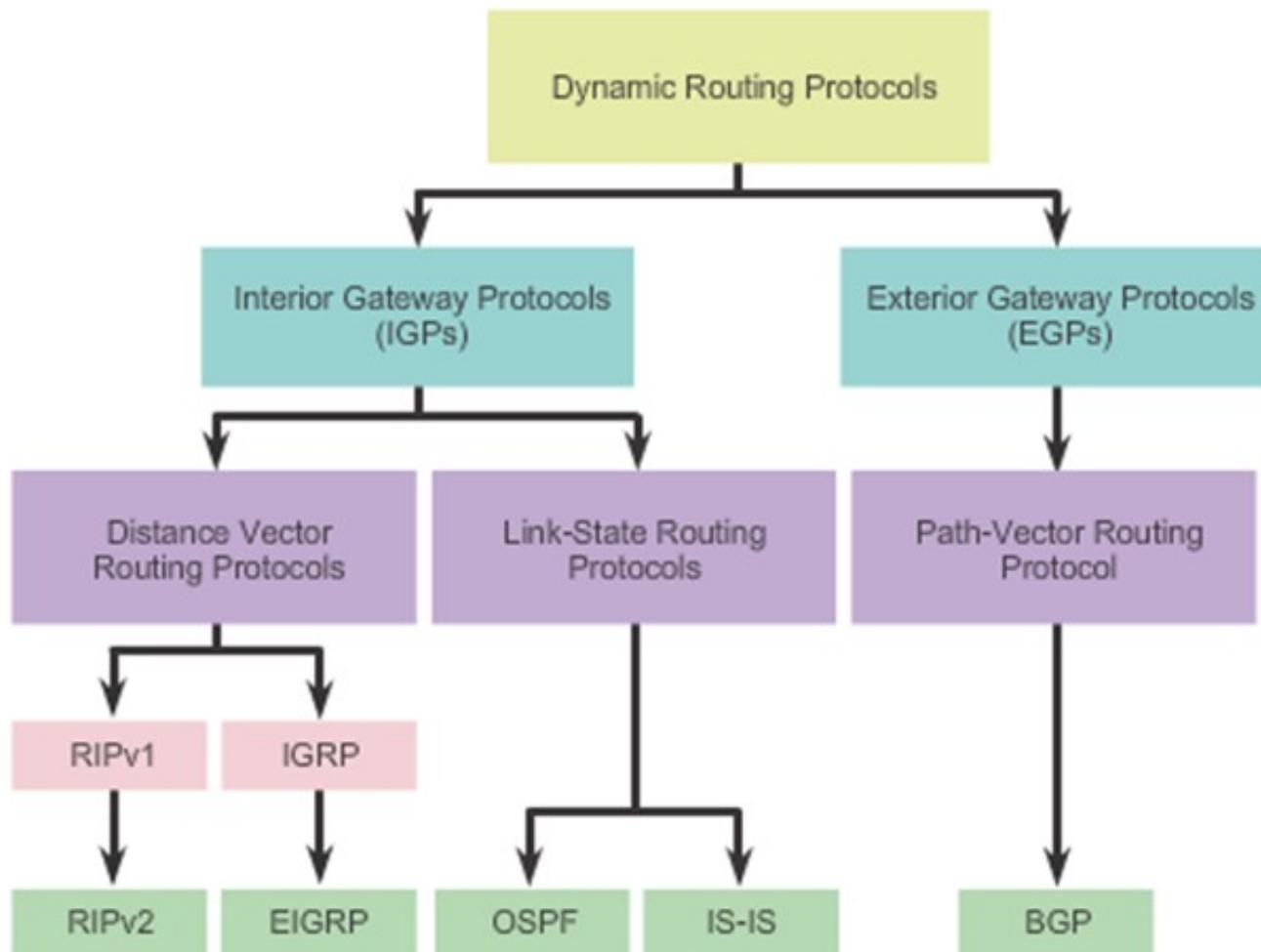
Purpose of Dynamic Routing Protocols

Routing Protocols are used to facilitate the **exchange of routing information** between routers.

The purpose of dynamic routing protocols includes:

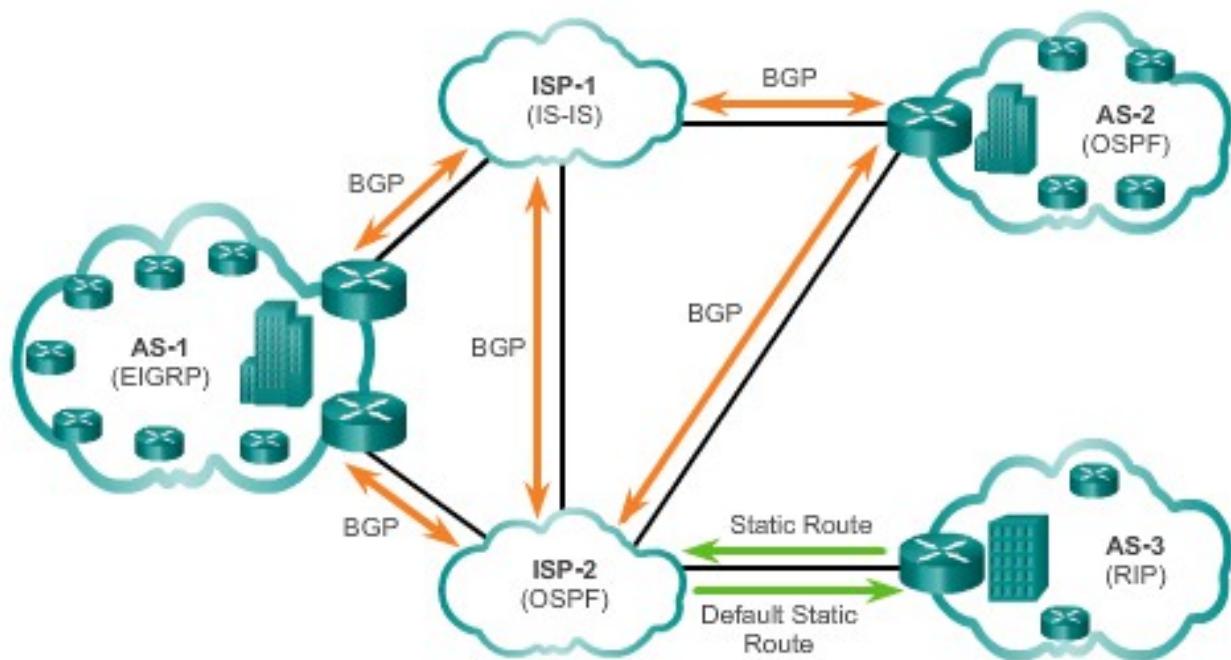
- Discovery of remote networks
- Maintaining **up-to-date** routing information
- Choosing the **best path** to destination networks
- Ability to find a new best path if the current path is no longer available

Classifying Routing Protocols



IGP and EGP Routing Protocols

IGP versus EGP Routing Protocols



Interior Gateway Protocols (IGP) -

- Used for routing within an AS
- Include RIP, EIGRP, OSPF, and IS-IS

Exterior Gateway Protocols (EGP) -

- Used for routing between AS
- Official routing protocol used by the Internet

Dynamic Routing Protocols

- Dynamic routing protocols used in networks since the late 1980s
- Newer versions support the communication based on IPv6

	Interior Gateway Protocols				Exterior Gateway Protocols
	Distance Vector		Link-State		Path Vector
IPv4	RIPv2	EIGRP	OSPFv2	IS-IS	BGP-4
IPv6	RIPng	EIGRP for IPv6	OSPFv3	IS-IS for IPv6	BGP-MP

Components of Dynamic Routing Protocols

- **Data structures** - Routing protocols typically use tables or databases for its operations. This information is kept in RAM.
- **Routing protocol messages** - Routing protocols use various types of messages to discover neighboring routers, exchange routing information, and other tasks to learn and maintain accurate information about the network.
- **Algorithm** - Routing protocols use algorithms for facilitating routing information for best path determination.

The Role of Dynamic Routing Protocols

Advantages

- Automatically share information about remote networks
- Determine the best path to each network and add this information to their routing tables
- Compared to static routing, dynamic routing protocols require less administrative overhead
- Help the network administrator manage the time-consuming process of configuring and maintaining static routes

Disadvantages :

- Part of a router's resources are dedicated for protocol operation, including CPU time and network link bandwidth
- Times when static routing is more appropriate

Dynamic Routing Protocol Operation

1. The router sends and receives routing messages on its interfaces.
2. The router shares routing messages and routing information with other routers that are using the same routing protocol.
3. Routers exchange routing information to learn about remote networks.
4. When a router detects a topology change the routing protocol can advertise this change to other routers.

Routing Protocol Metrics

A metric is a measurable value that is assigned by the routing protocol to different routes based on the usefulness of that route:

- Used to determine the overall “cost” of a path from source to destination.
- Routing protocols determine the best path based on the route with the lowest cost.
- RIP: Hop count is the metric
- OSPF: Bandwidth

Routing Protocol Operating Fundamentals

Cold Start

Directly Connected Networks Detected



Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0

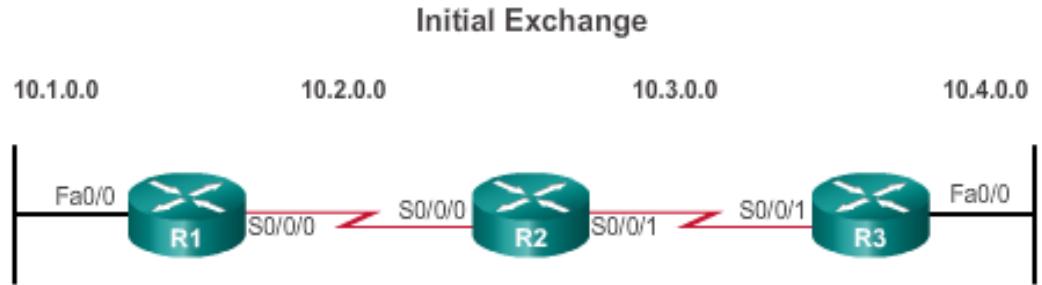
Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0

Routers running RIPv2

- All routers have the information of their own interfaces which are directly connected

Network Discovery



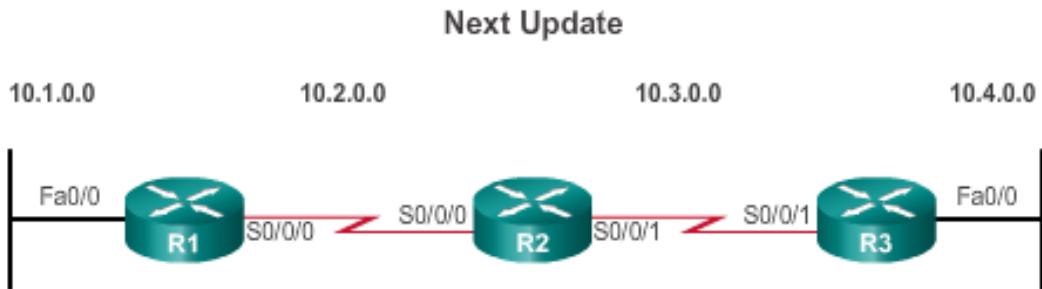
Network	Interface	Hop	Network	Interface	Hop	Network	Interface	Hop
10.1.0.0	Fa0/0	0	10.2.0.0	S0/0/0	0	10.3.0.0	S0/0/0	0
10.2.0.0	S0/0/0	0	10.3.0.0	S0/0/1	0	10.4.0.0	Fa0/0	0
10.3.0.0	S0/0/0	1	10.1.0.0	S0/0/0	1	10.2.0.0	S0/0/1	1
			10.4.0.0	S0/0/1	1			

Routers running RIPv2

Router R1:

- Sends an update about network 10.1.0.0 out the Serial0/0/0 interface
- Sends an update about network 10.2.0.0 out the FastEthernet0/0 interface
- Receives update from R2 about network 10.3.0.0 with a metric of 1
- Stores network 10.3.0.0 in the routing table with a metric of 1

Exchanging the Routing Information



Network	Interface	Hop	Network	Interface	Hop	Network	Interface	Hop
10.1.0.0	Fa0/0	0	10.2.0.0	S0/0/0	0	10.3.0.0	S0/0/1	0
10.2.0.0	S0/0/0	0	10.3.0.0	S0/0/1	0	10.4.0.0	Fa0/0	0
10.3.0.0	S0/0/0	1	10.1.0.0	S0/0/0	1	10.2.0.0	S0/0/1	1
10.4.0.0	S0/0/0	2	10.4.0.0	S0/0/1	1	10.1.0.0	S0/0/1	2

Routers running RIPv2

Router R1:

- Sends an update about network 10.1.0.0 out the Serial 0/0/0 interface
- Sends an update about networks 10.2.0.0 and 10.3.0.0 out the FastEthernet0/0 interface
- Receives** an update from R2 about network **10.4.0.0** with a metric of 2
- Stores network 10.4.0.0 in the routing table with a metric of 2
- Same update from R2 contains information about network 10.3.0.0 with a metric of 1. There is no change; therefore, the routing information is not updated.

Achieving Convergence

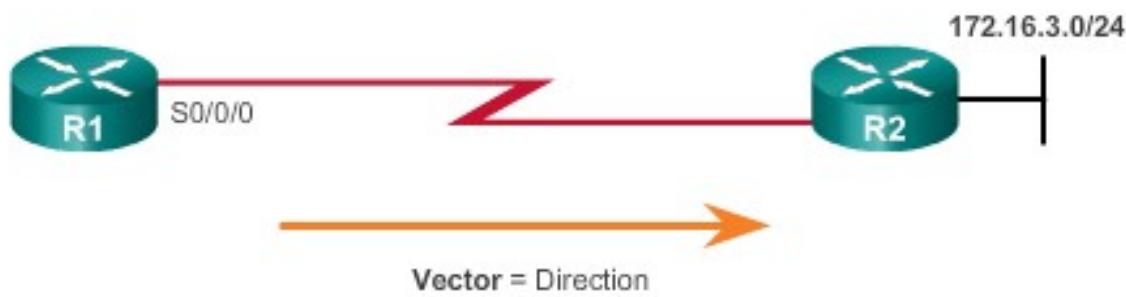
The network is converged when all routers have complete and accurate information about the entire network:

- Convergence time is the time it takes routers to share information, calculate best paths, and update their routing tables.
- A network is not completely operable until the network has converged.
- Convergence properties include the speed of propagation of routing information and the calculation of optimal paths. The speed of propagation refers to the amount of time it takes for routers within the network to forward routing information.
- Generally, older protocols, such as RIP, are slow to converge, whereas modern protocols, such as EIGRP and OSPF, converge more quickly.

Distance Vector Routing Protocols: Examples

The Meaning of Distance Vector

Distance = How Far



For R1, 172.16.3.0/24 is one hop away (distance). It can be reached through R2 (vector).

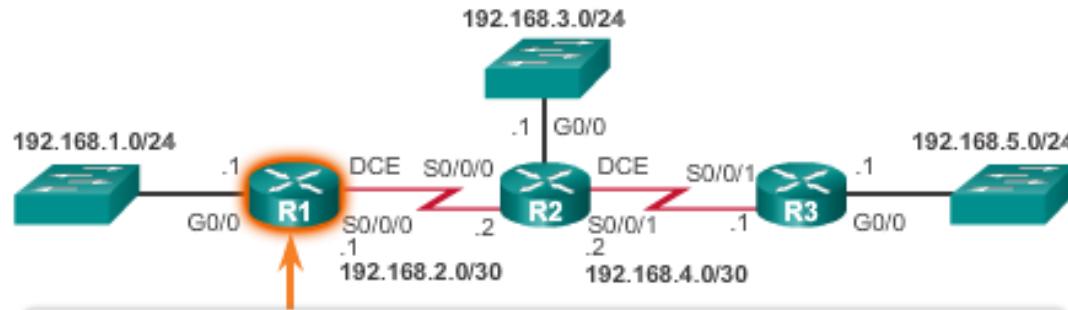
Distance vector IPv4 IGPs:

- **RIPv1** - First generation legacy protocol
- **RIPv2** - Simple distance vector routing protocol
- **IGRP** - First generation Cisco proprietary protocol (obsolete)
- **EIGRP** - Advanced version of distance vector routing

Router RIP Configuration

```
R1# conf t  
Enter configuration commands, one per line. End with CNTL/Z.  
R1(config)# router rip  
R1(config-router)#{
```

Advertising the R1 Networks

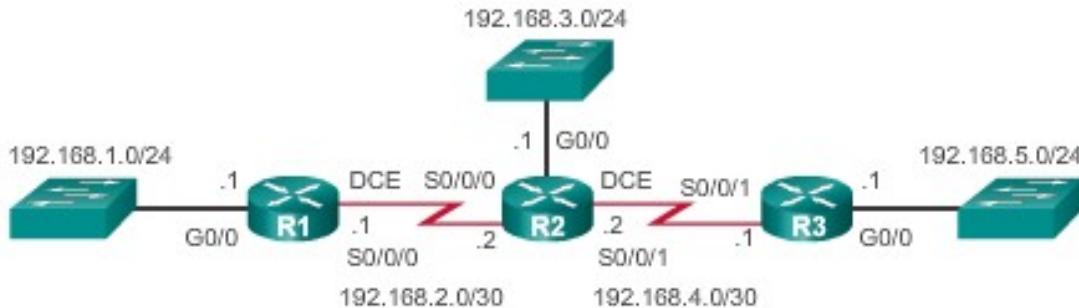


```
R1(config)#router rip  
R1(config-router)#network 192.168.1.0  
R1(config-router)#network 192.168.2.0  
R1(config-router)#{
```

Configuring the RIP Protocol

Configuring Passive Interfaces

Configuring Passive Interfaces on R1



Sending out unneeded updates on a LAN impacts the network in three ways:

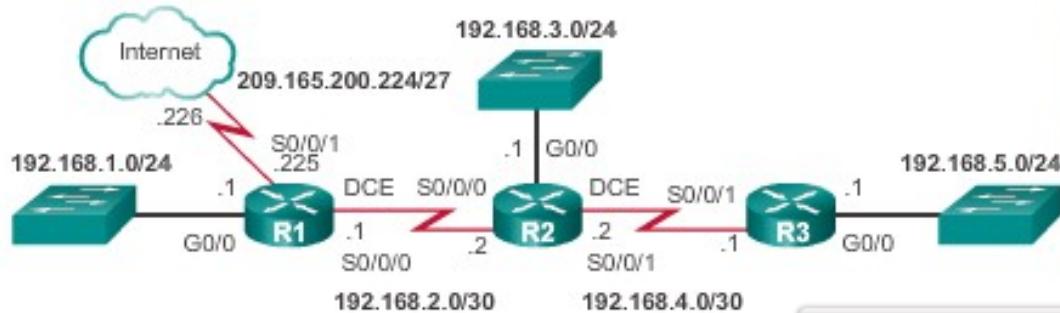
- Wasted Bandwidth
- Wasted Resources
- Security Risk

```
R1(config)# router rip
R1(config-router)# passive-interface g0/0
R1(config-router)# end
R1#
R1# show ip protocols | begin Default
Default version control: send version 2, receive version 2
  Interface          Send   Recv  Triggered RIP  Key-chain
  Serial0/0/0           2      2
Automatic network summarization is not in effect
Maximum path: 4
Routing for Networks:
  192.168.1.0
  192.168.2.0
Passive Interface(s):
  GigabitEthernet0/0
Routing Information Sources:
  Gateway          Distance      Last Update
  192.168.2.2            120        00:00:06
Distance: (default is 120)

R1#
```

Configuring the RIP Protocol Propagating a Default Route

Propagating a Default Route on R1

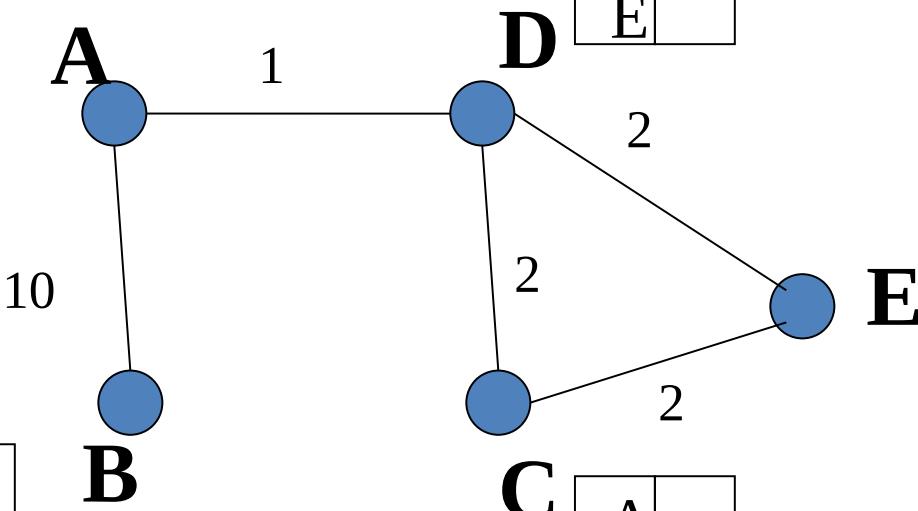


```
R1(config)# ip route 0.0.0.0 0.0.0.0 S0/0/1 209.165.200.226
R1(config)# router rip
R1(config-router)# default-information originate
R1(config-router)# ^Z
R1#
*Mar 10 23:33:51.801: %SYS-5-CONFIG_I: Configured from
console by console
R1# show ip route | begin Gateway
Gateway of last resort is 209.165.200.226 to network
0.0.0.0

S*    0.0.0.0/0 [1/0] via 209.165.200.226, Serial0/0/1
      192.168.1.0/24 is variably subnetted, 2 subnets, 2
masks
C          192.168.1.0/24 is directly connected,
GigabitEthernet0/0
L          192.168.1.1/32 is directly connected,
GigabitEthernet0/0
      192.168.2.0/24 is variably subnetted, 2 subnets, 2
masks
C          192.168.2.0/24 is directly connected, Serial0/0/0
L          192.168.2.1/32 is directly connected, Serial0/0/0
R      192.168.3.0/24 [120/1] via 192.168.2.2, 00:00:08,
```

Example: DV routing

A	0
B	
C	
D	
E	



A	
B	0
C	
D	
E	

A	
B	
C	
D	0
E	

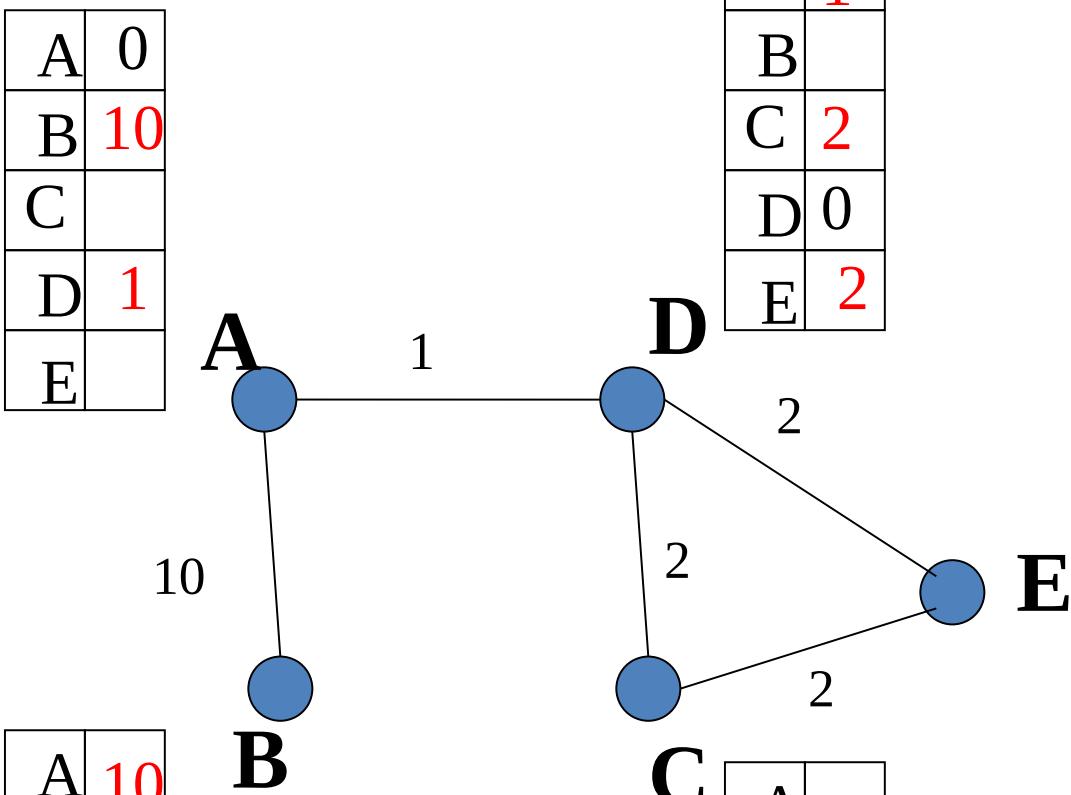
A	
B	
C	
D	
E	0

A	
B	
C	0
D	
E	

Initialization

A	0
B	10
C	
D	1
E	

A	10
B	0
C	
D	
E	



A	1
B	
C	2
D	0
E	2

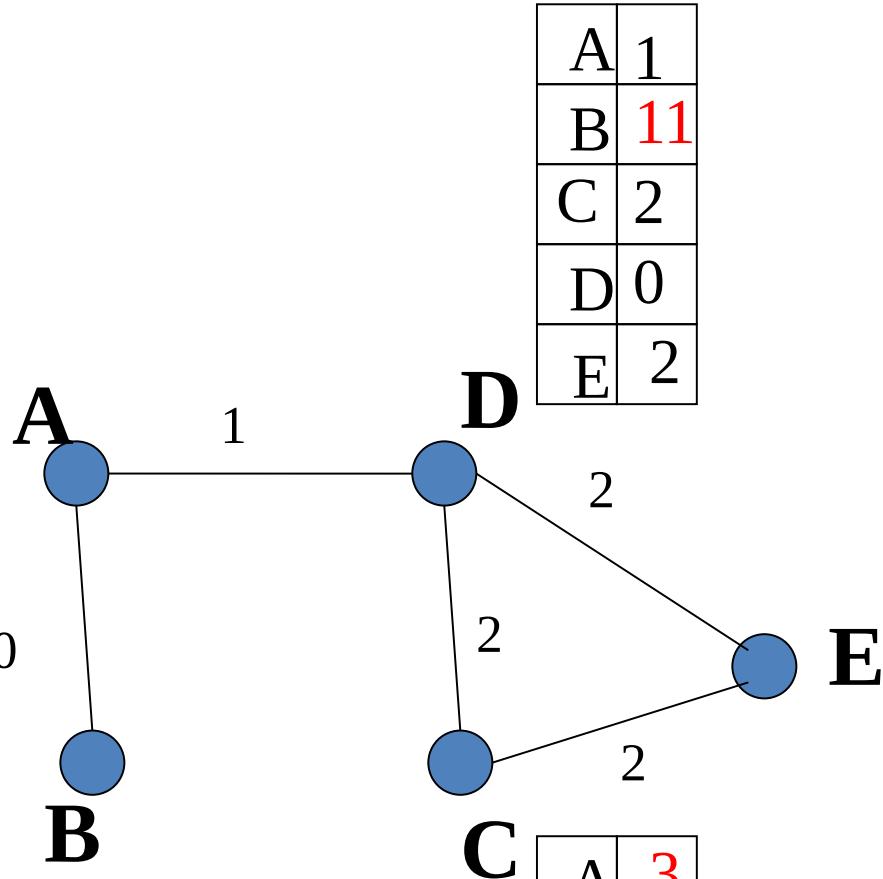
A	
B	
C	0
D	2
E	2

A	
B	
C	2
D	2
E	0

Direct
Neighbours

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	
D	11
E	



A	1
B	11
C	2
D	0
E	2

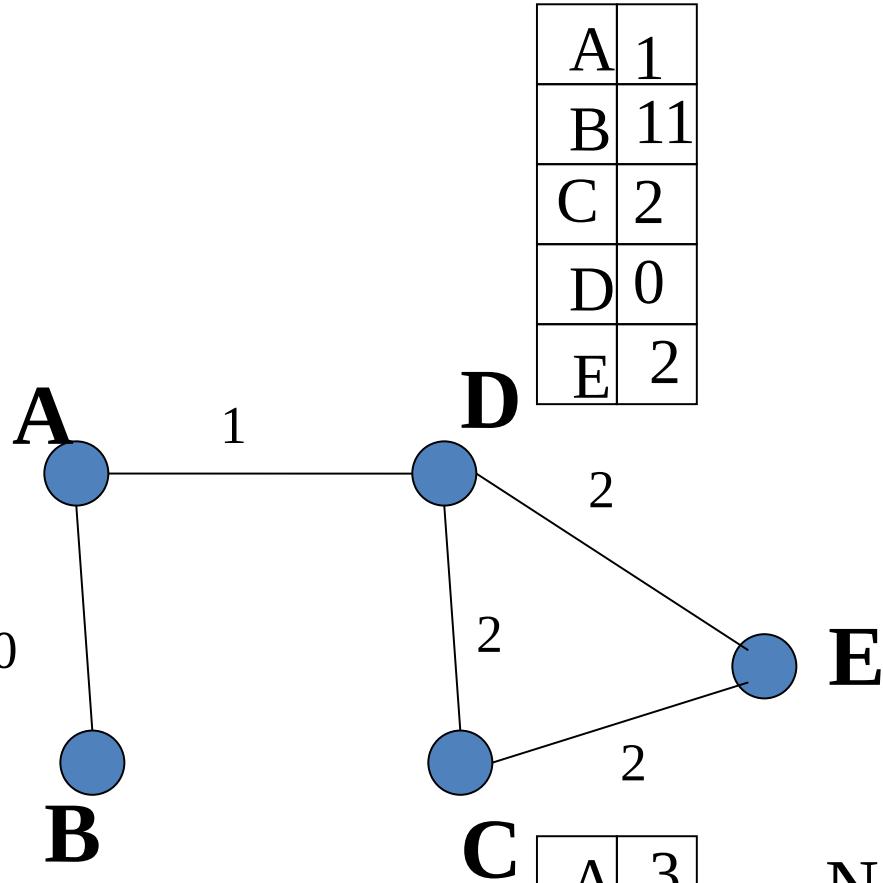
A	3
B	
C	0
D	2
E	2

A	3
B	
C	2
D	2
E	0

Neighbours
of neighbours

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	11
C	2
D	0
E	2

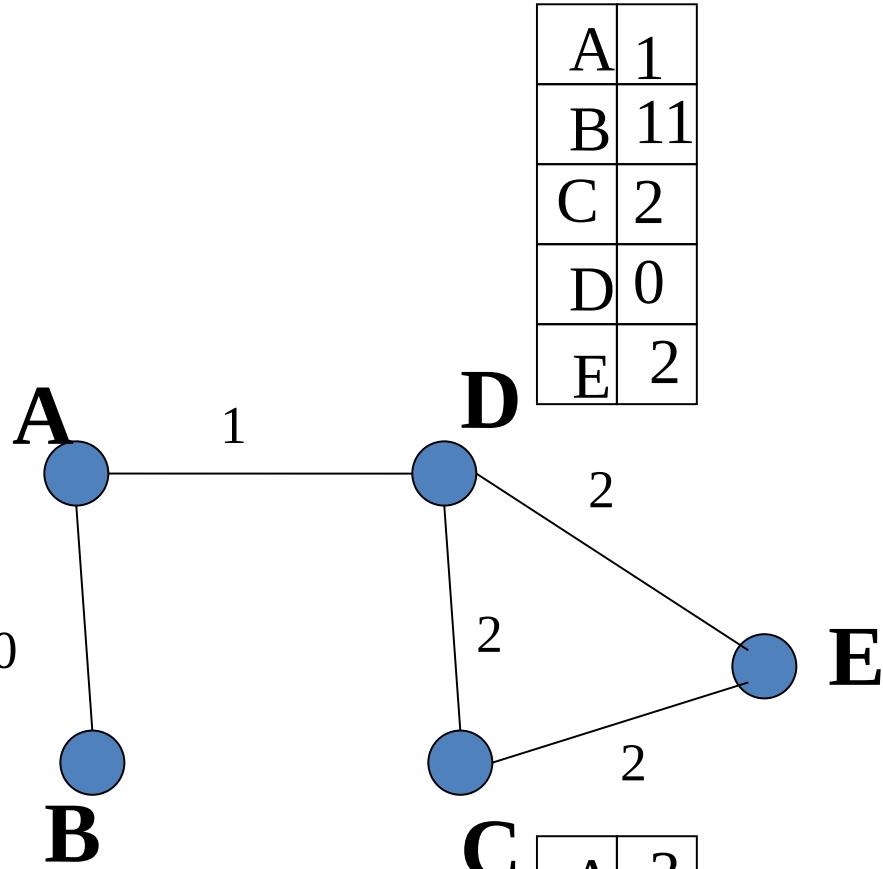
A	3
B	13
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

Neighbours
of neighbours
of neighbours

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	11
C	2
D	0
E	2

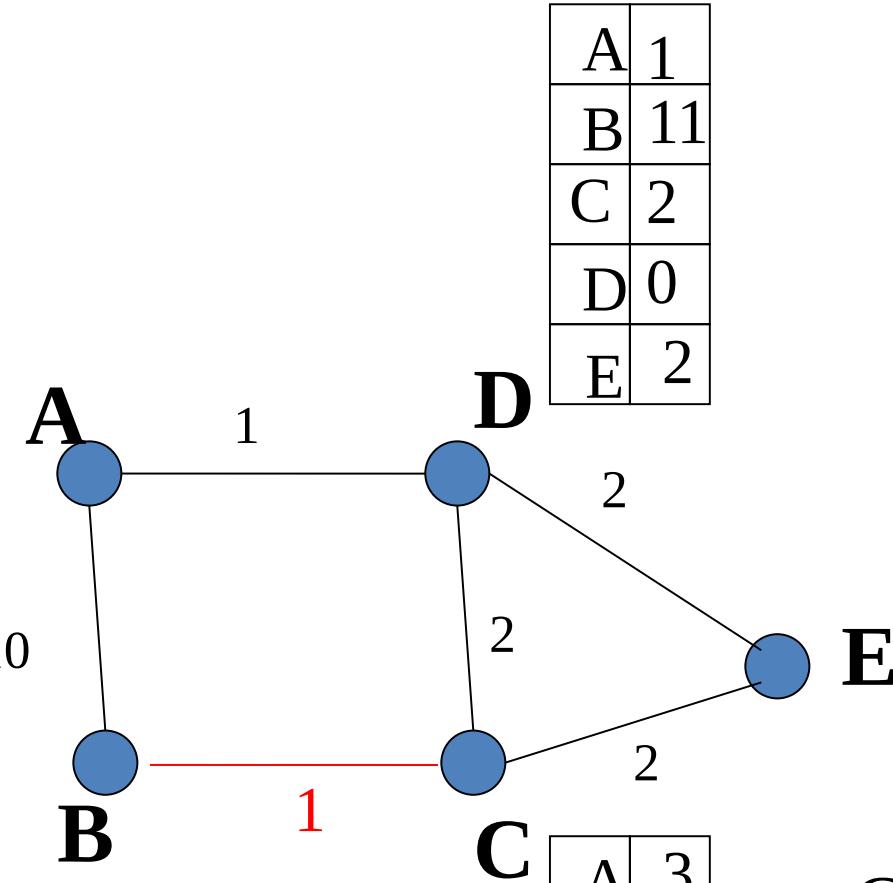
A	3
B	13
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

Stable
convergence

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	11
C	2
D	0
E	2

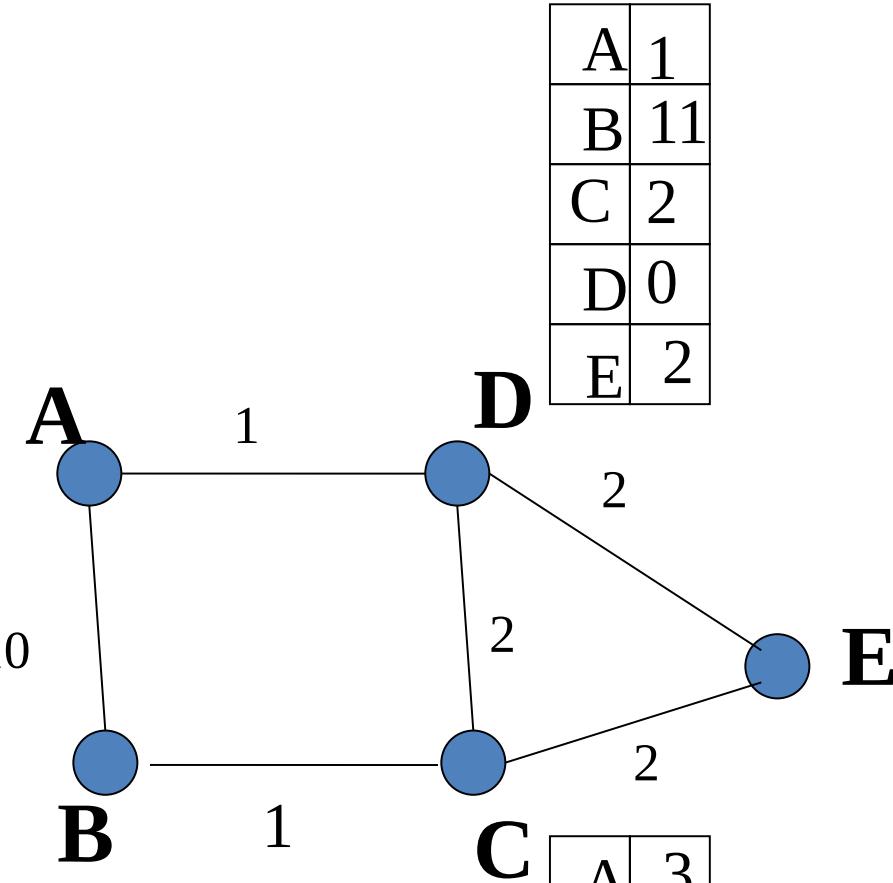
A	3
B	13
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

Good news:
A new link!

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	1
D	11
E	13



A	1
B	11
C	2
D	0
E	2

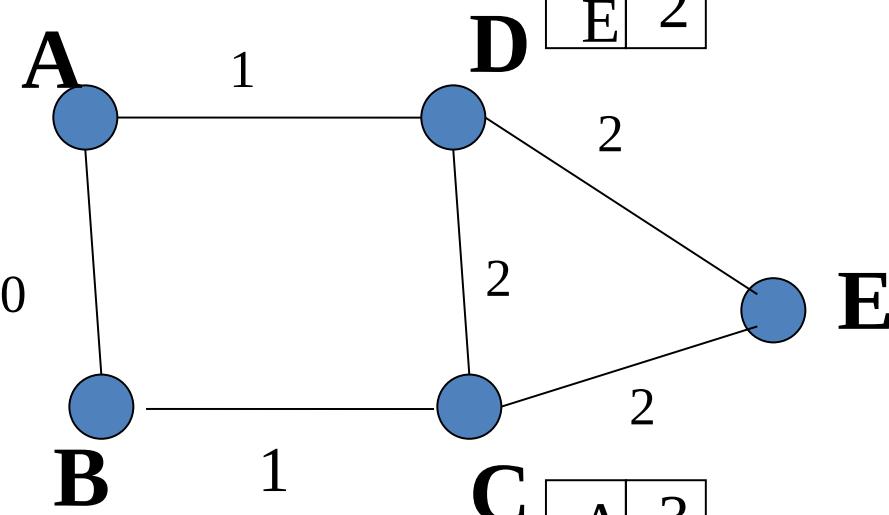
A	3
B	1
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

Direct
endpoints
know

A	0
B	10
C	3
D	1
E	3

A	4
B	0
C	1
D	3
E	3



A	1
B	3
C	2
D	0
E	2

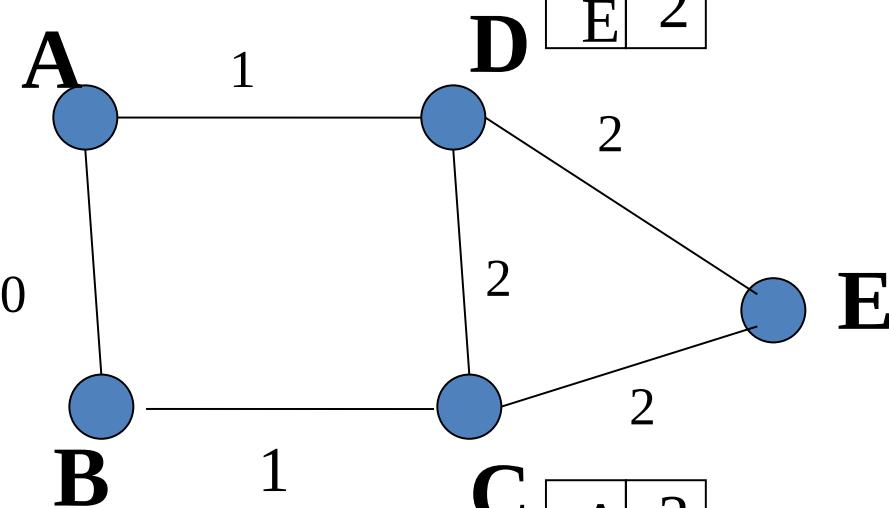
A	3
B	1
C	0
D	2
E	2

A	3
B	3
C	2
D	2
E	0

Neighbours
know

A	0
B	4
C	3
D	1
E	3

A	4
B	0
C	1
D	3
E	3



A	1
B	3
C	2
D	0
E	2

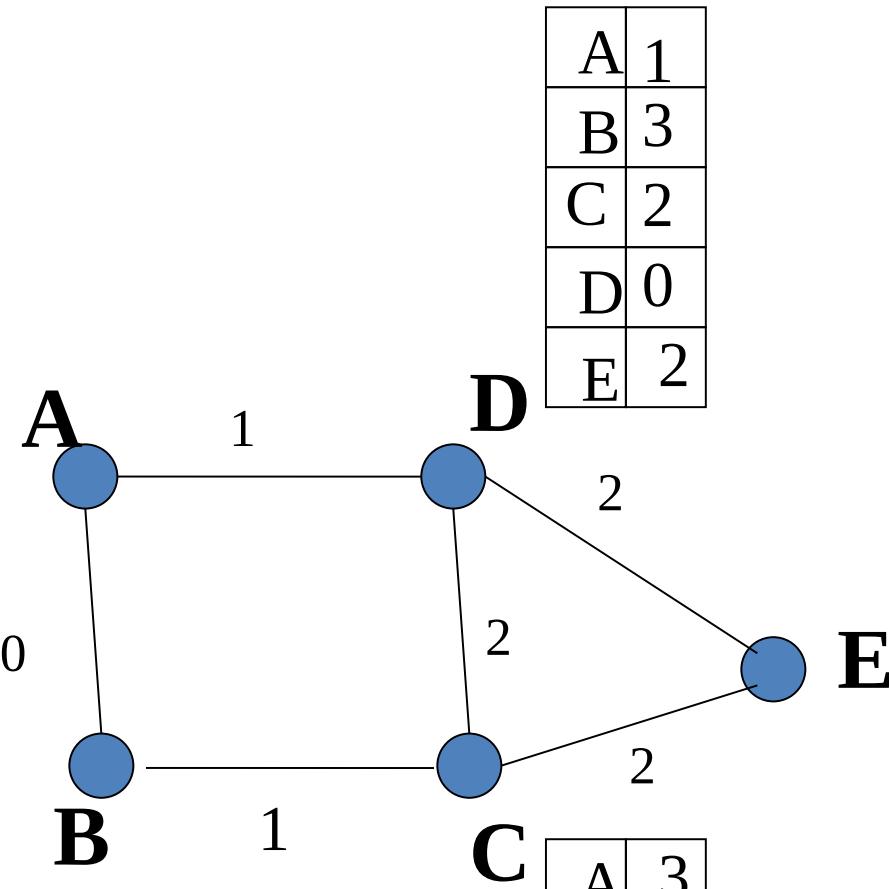
A	3
B	1
C	0
D	2
E	2

A	3
B	3
C	2
D	2
E	0

Neighbours
of neighbours
know

A	4
B	0
C	1
D	3
E	3

A	0
B	4
C	3
D	1
E	3



A	1
B	3
C	2
D	0
E	2

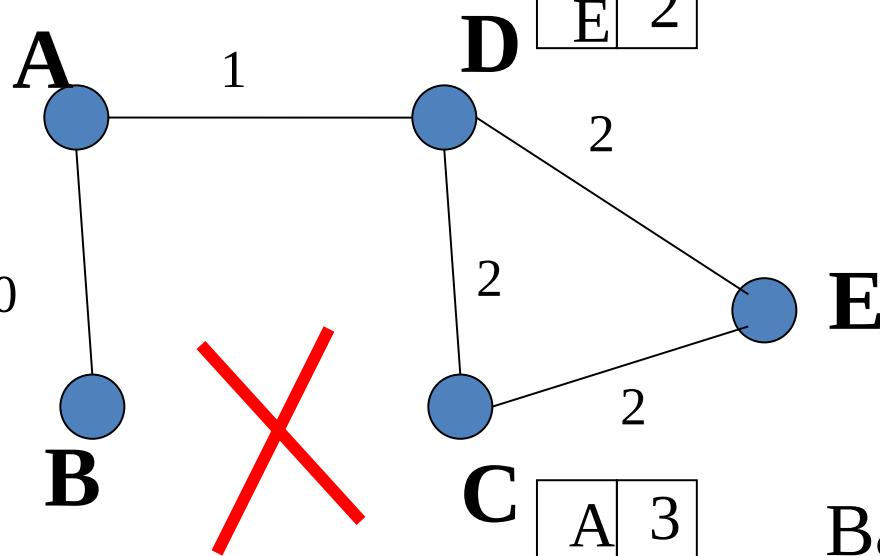
A	3
B	3
C	2
D	2
E	0

A	3
B	1
C	0
D	2
E	2

A happy and stable network

A	0
B	4
C	3
D	1
E	3

A	4
B	0
C	1
D	3
E	3



A	1
B	3
C	2
D	0
E	2

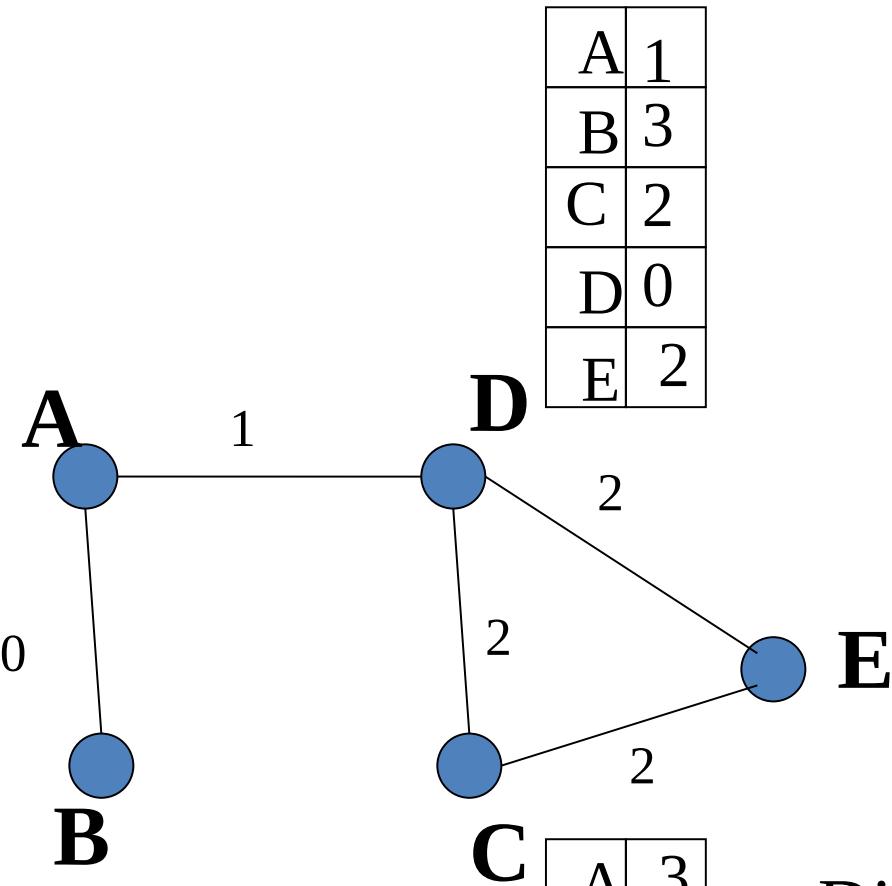
A	3
B	1
C	0
D	2
E	2

A	3
B	3
C	2
D	2
E	0

Bad news:
Link crash!!

A	
B	0
C	
D	
E	

A	0
B	4
C	3
D	1
E	3



A	1
B	3
C	2
D	0
E	2

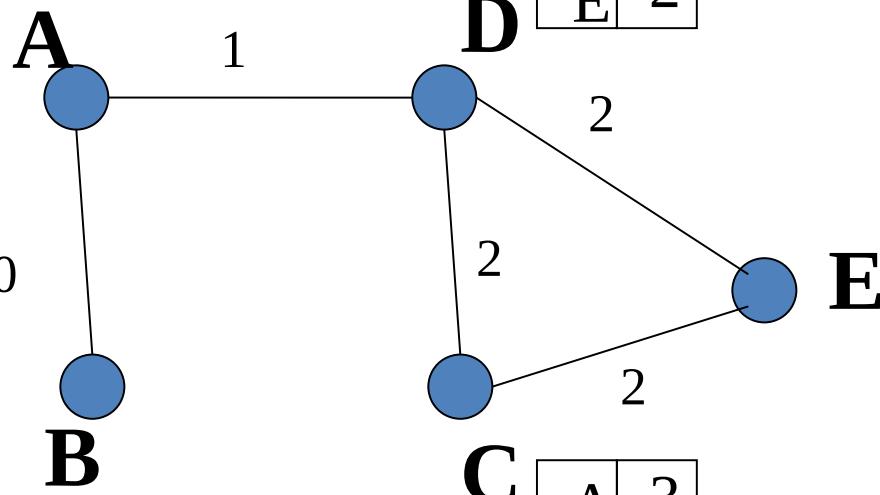
A	3
B	3
C	2
D	2
E	0

A	3
B	
C	0
D	2
E	2

Direct
endpoints
know

A	0
B	4
C	3
D	1
E	3

A	10
B	0
C	
D	
E	



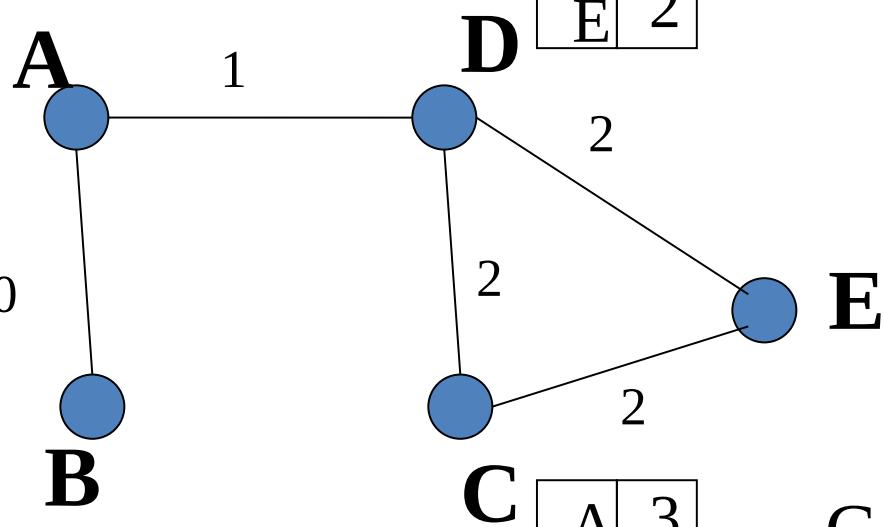
A	1
B	3
C	2
D	0
E	2

A	3
B	
C	0
D	2
E	2

A	3
B	3
C	2
D	2
E	0

A	0
B	4
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	3
C	2
D	0
E	2

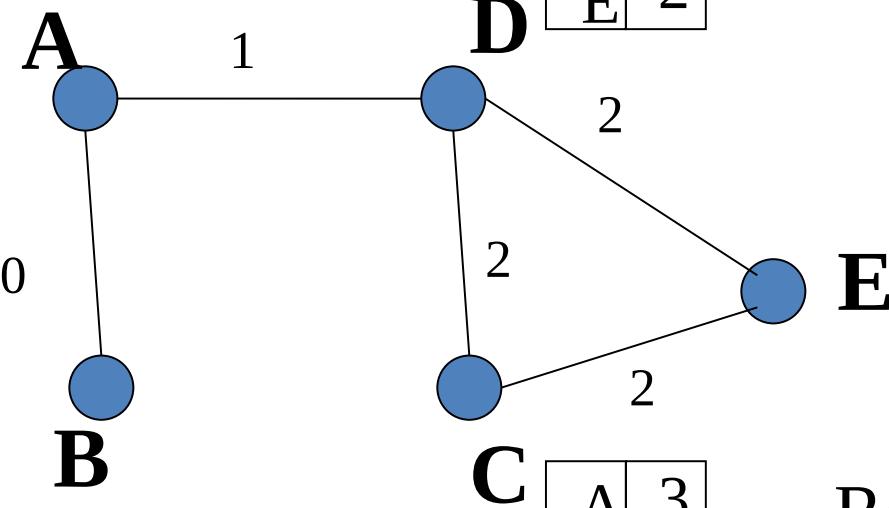
A	3
B	3
C	2
D	2
E	0

A	3
B	5
C	0
D	2
E	2

Get help
from
neighbours

A	10
B	0
C	13
D	11
E	13

A	0
B	4
C	3
D	1
E	3



A	1
B	7
C	2
D	0
E	2

A	3
B	7
C	2
D	2
E	0

A	3
B	5
C	0
D	2
E	2

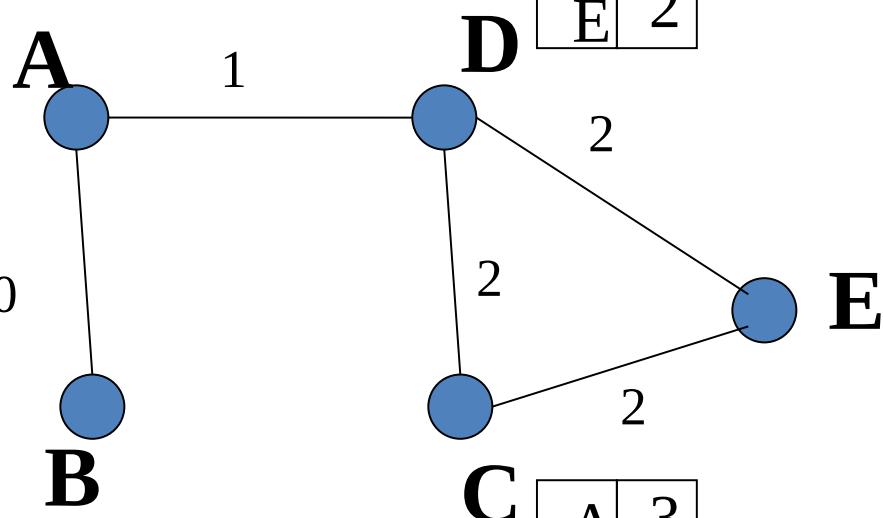
Routing loop
(due to
inconsistent
state info)

Problem of DV Routing: Count-to-Infinity

- Count-to-Infinity problem
- **Counting to infinity** is just another name for a **routing loop**.
- In distance vector routing, routing loops usually occur when an **interface goes down**.
- It can also occur when two routers send updates to each other at the same time.

A	0
B	8
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



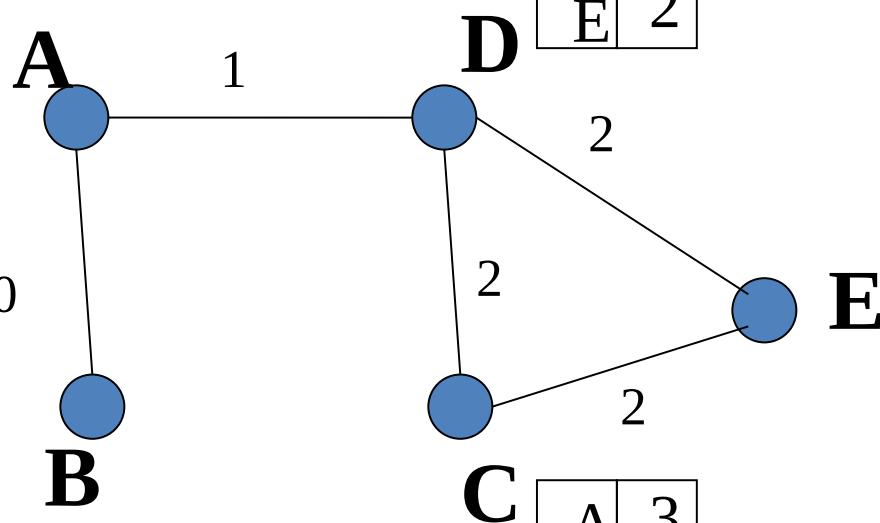
A	1
B	7
C	2
D	0
E	2

A	3
B	9
C	0
D	2
E	2

A	3
B	7
C	2
D	2
E	0

A	0
B	8
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	11
C	2
D	0
E	2

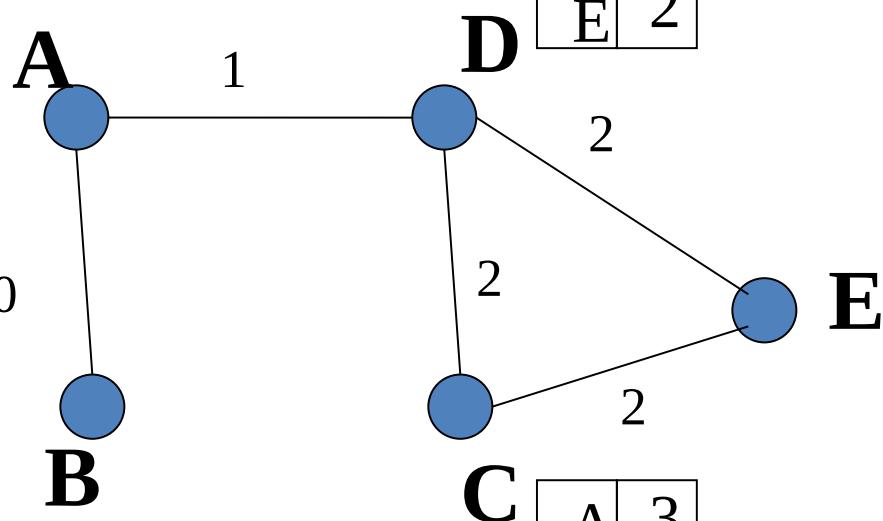
A	3
B	9
C	0
D	2
E	2

A	3
B	11
C	2
D	2
E	0

Counting
to infinity...

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13

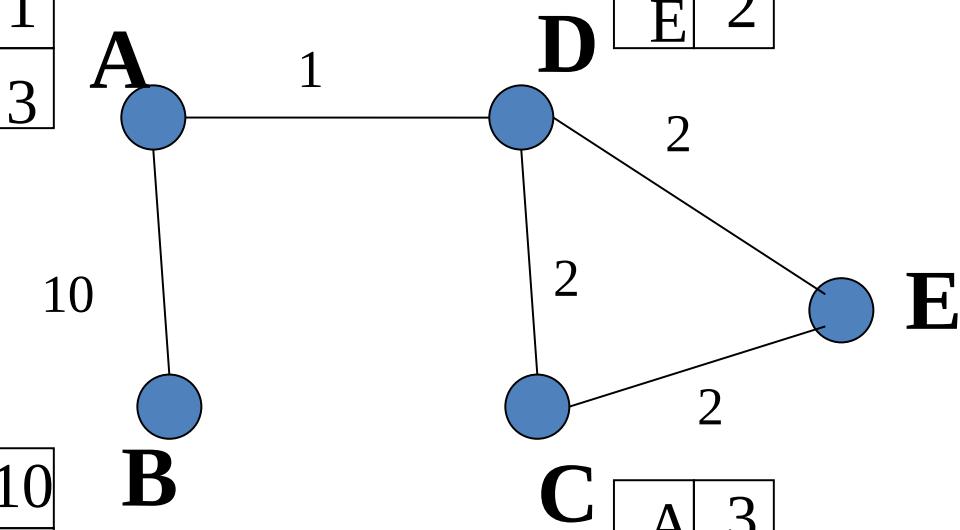
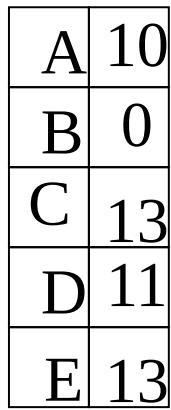


A	1
B	11
C	2
D	0
E	2

A	3
B	13
C	0
D	2
E	2

A	3
B	11
C	2
D	2
E	0

A	0
B	10
C	3
D	1
E	3



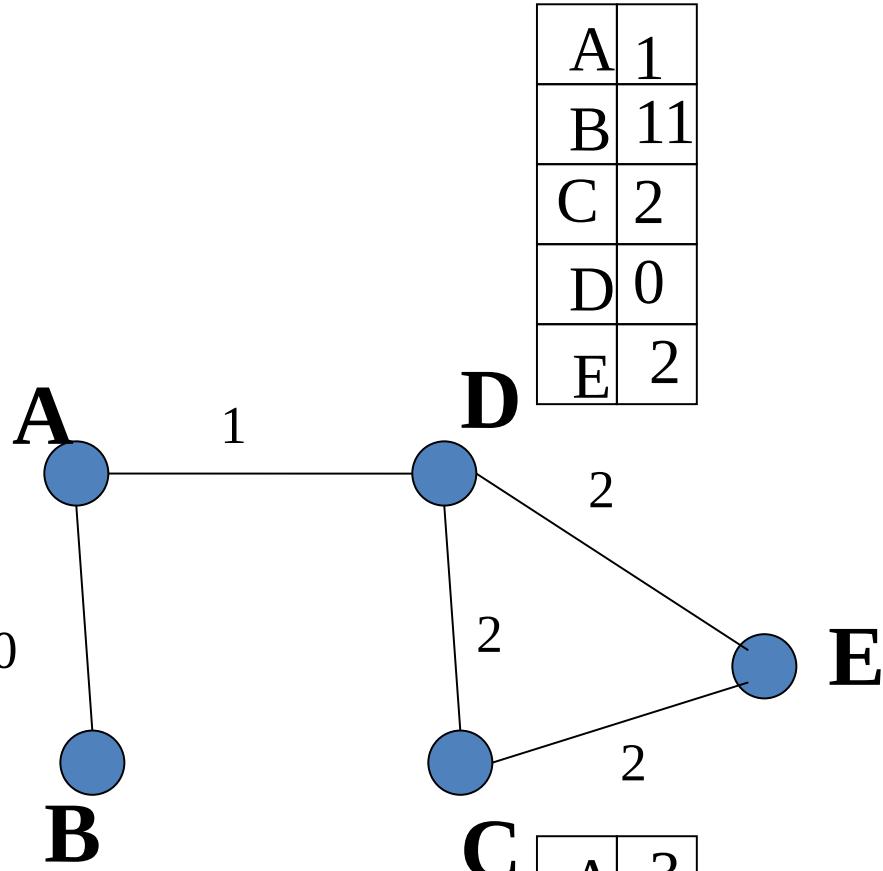
A	1
B	11
C	2
D	0
E	2

A	3
B	13
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

A	0
B	10
C	3
D	1
E	3

A	10
B	0
C	13
D	11
E	13



A	1
B	11
C	2
D	0
E	2

A	3
B	13
C	0
D	2
E	2

A	3
B	13
C	2
D	2
E	0

Stability
again

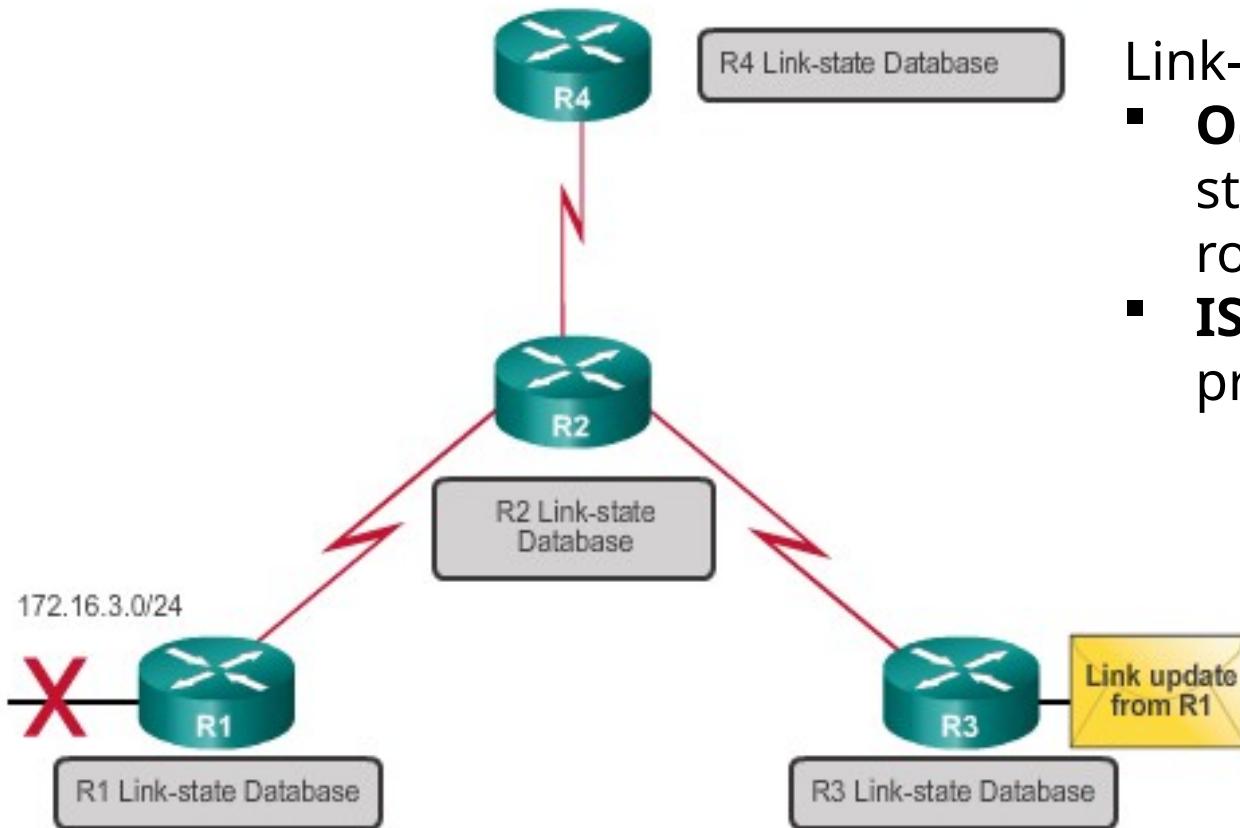
Loop Breaking Heuristics

- Set infinity to a limited number,
e.g. 16.
- Split horizon
- Split horizon with poison reverse

Link State Routing Protocols

Link-State Routing Protocols

Link-State Protocol Operation



- Link-state IPv4 IGPs:
- **OSPF** - Popular standards based routing protocol
 - **IS-IS** - Popular in provider networks.

Link-state protocols forward updates when the state of a link changes.

Routing Protocol Characteristics

	Distance Vector				Link State	
	RIPv1	RIPv2	IGRP	EIGRP	OSPF	IS-IS
Speed Convergence	Slow	Slow	Slow	Fast	Fast	Fast
Scalability - Size of Network	Small	Small	Small	Large	Large	Large
Use of VLSM	No	Yes	No	Yes	Yes	Yes
Resource Usage	Low	Low	Low	Medium	High	High
Implementation and Maintenance	Simple	Simple	Simple	Complex	Complex	Complex

Open Shortest Path First

Features of OSPF



Link-State Routing Process

Link-State Routing Process

- Each router learns about each of its own directly connected networks.
- Each router is responsible for "saying hello" to its neighbors on directly connected networks.
- Each router builds a Link State Packet (LSP) containing the state of each directly connected link.
- Each router floods the LSP to all neighbors who then store all LSP's received in a database.
- Each router uses the database to construct a complete map of the topology and computes the best path to each destination networks.

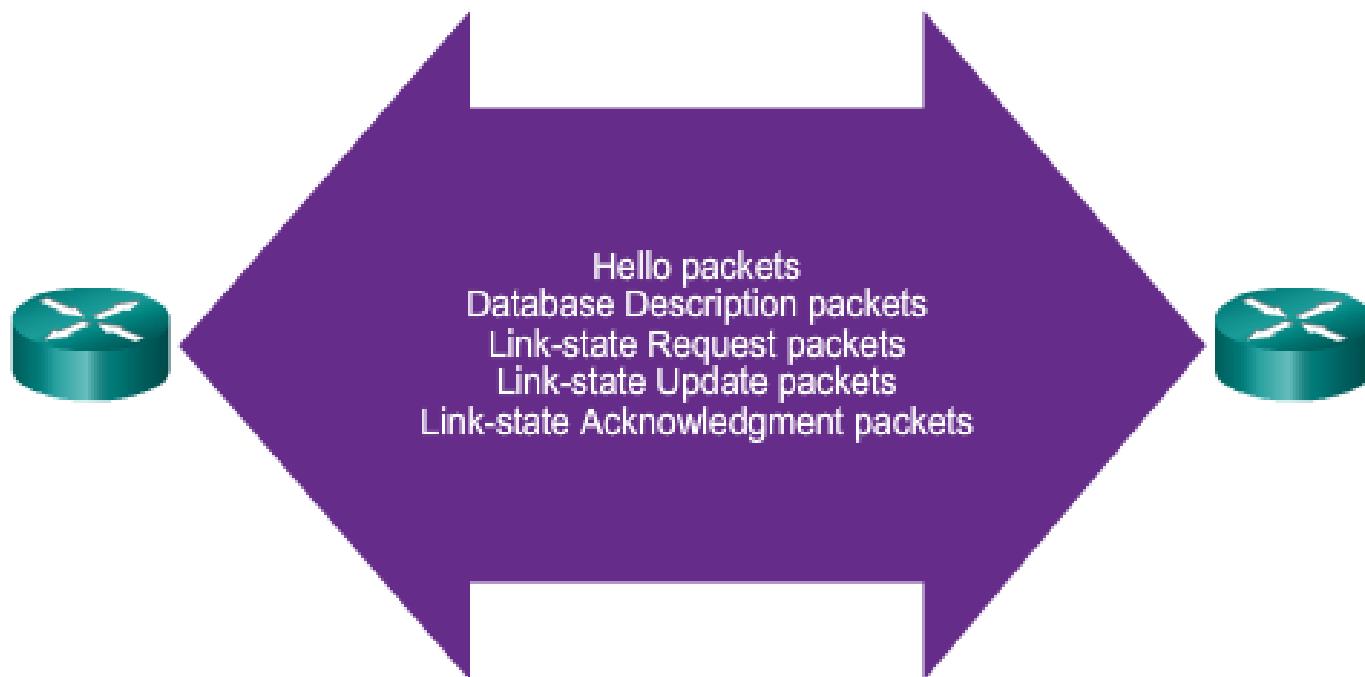
Components of OSPF

OSPF Data Structures

Database	Table	Description
Adjacency Database	Neighbor Table	<ul style="list-style-type: none">• List of all neighbor routers to which a router has established bidirectional communication.• This table is unique for each router.• Can be viewed using the show ip ospf neighbor command.
Link-state Database (LSDB)	Topology Table	<ul style="list-style-type: none">• Lists information about all other routers in the network.• The database shows the network topology.• All routers within an area have identical LSDB.• Can be viewed using the show ip ospf database command.
Forwarding Database	Routing Table	<ul style="list-style-type: none">• List of routes generated when an algorithm is run on the link-state database.• Each router's routing table is unique and contains information on how and where to send packets to other routers.• Can be viewed using the show ip route command.

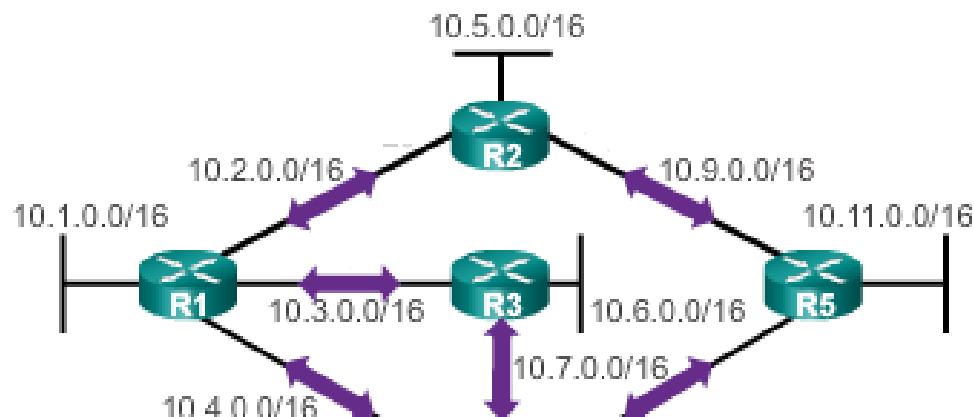
Components of OSPF

OSPF Routers Exchange Packets - These packets are used to discover neighboring routers and also to exchange routing information to maintain accurate information about the network.



Hello Exchange

Routers Exchange Hello Packets



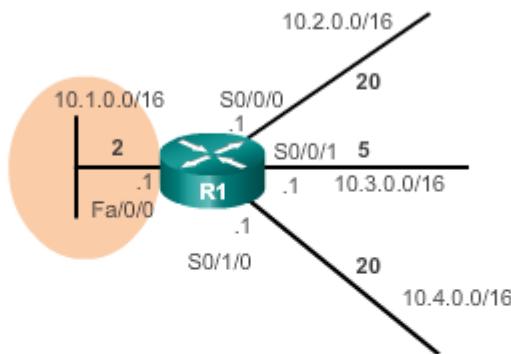
If a neighbor is present, the OSPF-enabled router attempts to establish a neighbor adjacency with that neighbor



Link and Link-State

The first step in the link-state routing process is that each router learns about its own links and its own directly connected networks.

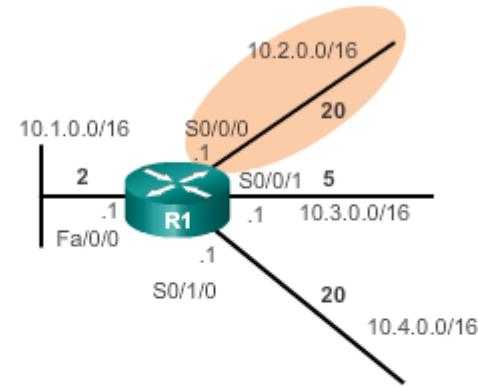
Link-State of Interface Fa0/0



Link 1

- Network: **10.1.0.0/16**
- IP address: **10.1.0.1**
- Type of network: **Ethernet**
- Cost of that link: **2**
- Neighbors: **None**

Link-State of Interface S0/0/0

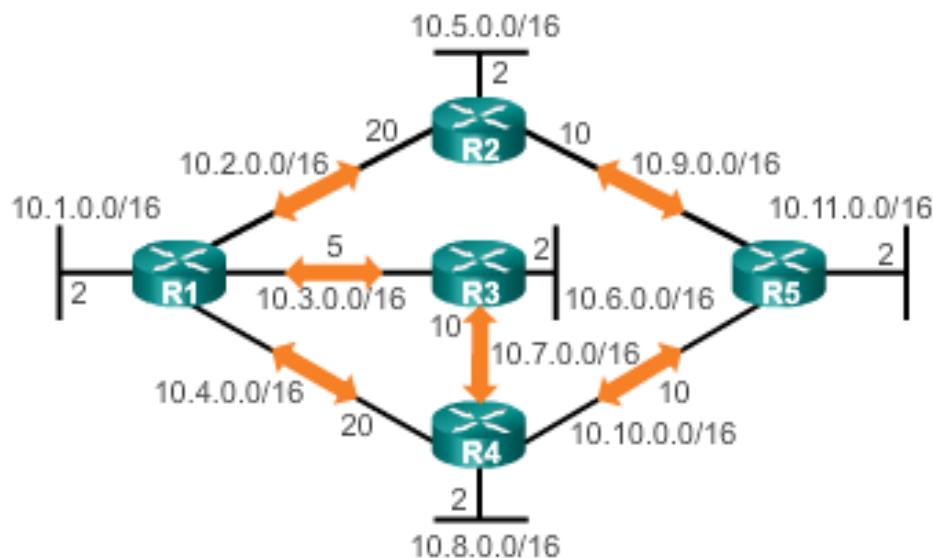


Link 2

- Network: **10.2.0.0/16**
- IP address: **10.2.0.1**
- Type of network: **Serial**
- Cost of that link: **20**
- Neighbors: **R2**

Exchange of LSA

Routers Exchange LSAs



- LSAs contain the state and cost of each directly connected link.
- Routers flood their LSAs to adjacent neighbors.
- Adjacent neighbors receiving the LSA immediately flood the LSA to other directly connected neighbors, until all routers in the area have all LSAs.

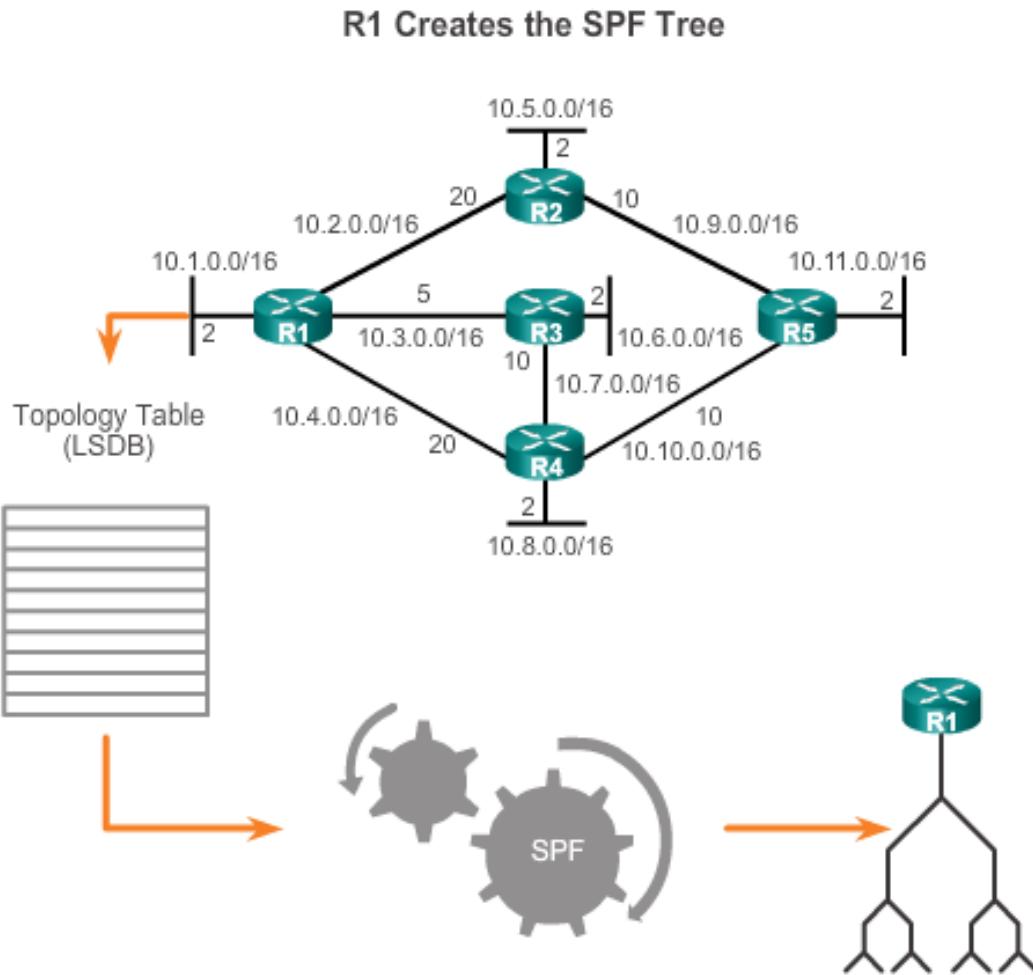
Building the Link-State Database

The final step in the link-state routing process is that each router uses the database to construct **a complete map of the topology** and computes the best path to each destination network.

Contents of the Link-State Database

R1 Link-State Database
R1 Link-states: <ul style="list-style-type: none">Connected to network 10.1.0.0/16, cost = 2Connected to R2 on network 10.2.0.0/16, cost = 20Connected to R3 on network 10.3.0.0/16, cost = 5Connected to R4 on network 10.4.0.0/16, cost = 20
R2 Link-states: <ul style="list-style-type: none">Connected to network 10.5.0.0/16, cost = 2Connected to R1 on network 10.2.0.0/16, cost = 20Connected to R5 on network 10.9.0.0/16, cost = 10
R3 Link-states: <ul style="list-style-type: none">Connected to network 10.6.0.0/16, cost = 2Connected to R1 on network 10.3.0.0/16, cost = 5Connected to R4 on network 10.7.0.0/16, cost = 10
R4 Link-states: <ul style="list-style-type: none">Connected to network 10.8.0.0/16, cost = 2Connected to R1 on network 10.4.0.0/16, cost = 20Connected to R3 on network 10.7.0.0/16, cost = 10Connected to R5 on network 10.10.0.0/16, cost = 10
R5 Link-states: <ul style="list-style-type: none">Connected to network 10.11.0.0/16, cost = 2Connected to R2 on network 10.9.0.0/16, cost = 10Connected to R4 on network 10.10.0.0/16, cost = 10

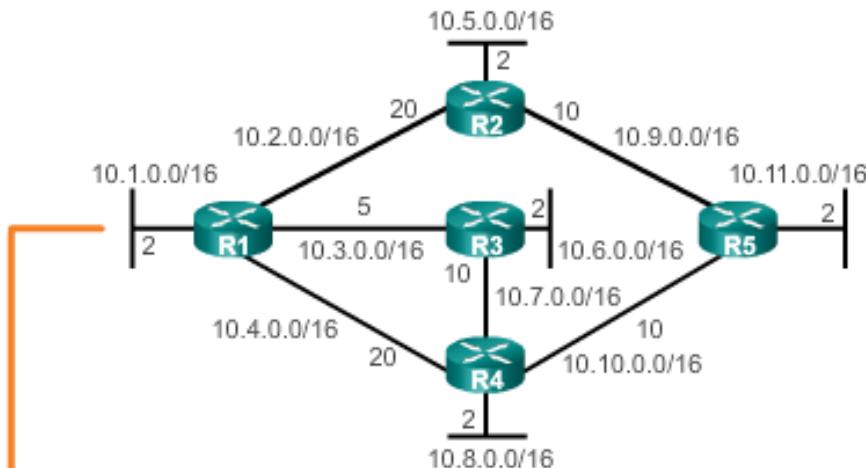
Creating SPF Tree



- Build the topology table based on the received LSAs.
- This database eventually holds all the information about the topology of the network.
- Execute the SPF Algorithm.

SPF Tree to best path Calculation

Content of the R1 SPF Tree



Destination	Shortest Path	Cost
10.5.0.0/16	R1 → R2	22
10.6.0.0/16	R1 → R3	7
10.7.0.0/16	R1 → R3	15
10.8.0.0/16	R1 → R3 → R4	17
10.9.0.0/16	R1 → R2	30
10.10.0.0/16	R1 → R3 → R4	25
10.11.0.0/16	R1 → R3 → R4 → R5	27

From the SPF tree, the best paths are inserted into the routing table.

Adding OSPF Routes to the Routing Table

Populate the Routing Table

Destination	Shortest Path	Cost
10.5.0.0/16	R1 → R2	22
10.6.0.0/16	R1 → R3	7
10.7.0.0/16	R1 → R3	15
10.8.0.0/16	R1 → R3 → R4	17
10.9.0.0/16	R1 → R2	30
10.10.0.0/16	R1 → R3 → R4	25
10.11.0.0/16	R1 → R3 → R4 → R5	27

R1 Routing Table

Directly Connected Networks

- 10.1.0.0/16 Directly Connected Network
- 10.2.0.0/16 Directly Connected Network
- 10.3.0.0/16 Directly Connected Network
- 10.4.0.0/16 Directly Connected Network

Remote Networks

- 10.5.0.0/16 via R2 serial 0/0/0,cost=22
- 10.6.0.0/16 via R3 serial 0/0/1,cost=7
- 10.7.0.0/16 via R3 serial 0/0/1,cost=15
- 10.8.0.0/16 via R3 serial 0/0/1,cost=17
- 10.9.0.0/16 via R2 serial 0/0/0,cost=30
- 10.10.0.0/16 via R3 serial 0/0/1,cost=25
- 10.11.0.0/16 via R3 serial 0/0/1,cost=27

OSPF Messages

Types of OSPF Packets

OSPF Packet Descriptions

Type	Packet Name	Description
1	Hello	Discovers neighbors and builds adjacencies between them
2	Database Description (DBD)	Checks for database synchronization between routers
3	Link-State Request (LSR)	Requests specific link-state records from router to router
4	Link-State Update (LSU)	Sends specifically requested link-state records
5	Link-State Acknowledgment (LSAck)	Acknowledges the other packet types

Hello Packet

OSPF Type 1 packet = Hello packet:

- Discover OSPF neighbors and establish neighbor adjacencies.
- Advertise parameters on which two routers must agree to become neighbors.
- Elect the Designated Router (DR) and Backup Designated Router (BDR) on multiaccess networks like Ethernet and Frame Relay.

Hello Packet Intervals

OSPF Hello packets are transmitted:

- To 224.0.0.5 in IPv4 and FF02::5 in IPv6 (all OSPF routers)
- Every 10 seconds (default on multiaccess and point-to-point networks)
- **Every 30 seconds**
- Dead interval is the period that the router waits to receive a Hello packet before declaring the neighbor down
- Router floods the LSDB with information about down neighbors out all OSPF enabled interfaces

Link-State Updates

LSUs Contain LSAs

Type	Packet Name	Description
1	Hello	Discovers neighbors and builds adjacencies between them
2	DBD	Checks for database synchronization between router
3	LSR	Requests specific link-state records from router to router
4	LSU	Sends specifically requested link-state records
5	LSAck	Acknowledges the other packet types



- An LSU contains one or more LSAs.
- LSAs contain route information for destination networks.

LSA Type	Description
1	Router LSAs
2	Network LSAs
3 or 4	Summary LSAs
5	Autonomous System External LSAs
6	Multicast OSPF LSAs
7	Defined for Not-So-Stubby Areas
8	External Attributes LSA for Border Gateway Protocol (BGP)
9,10,11	Opaque LSAs

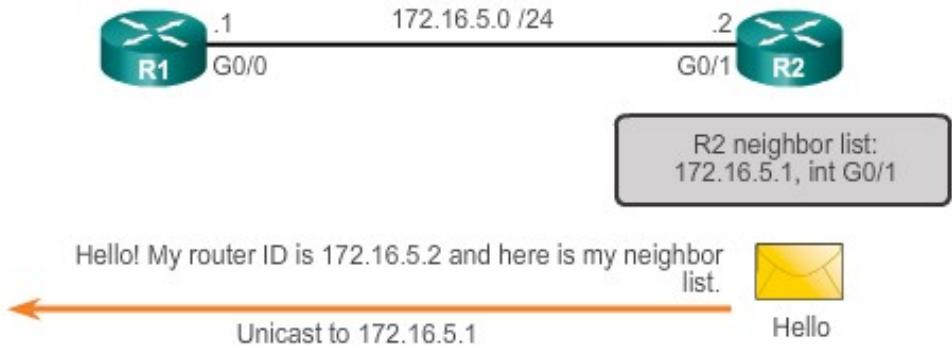
Establish Neighbor Adjacencies

Down State to Init State



The Init State

Hello! My router ID is 172.16.5.1. Is there anyone else on this link?
Hello



OSPF Operation

Establish Neighbor Adjacencies (cont.)

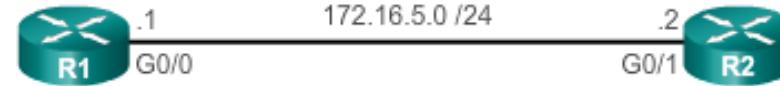
Two-Way State



R1 neighbor list:
172.16.5.2, int Fa0/0

Two-Way State

Elect the DR and BDR



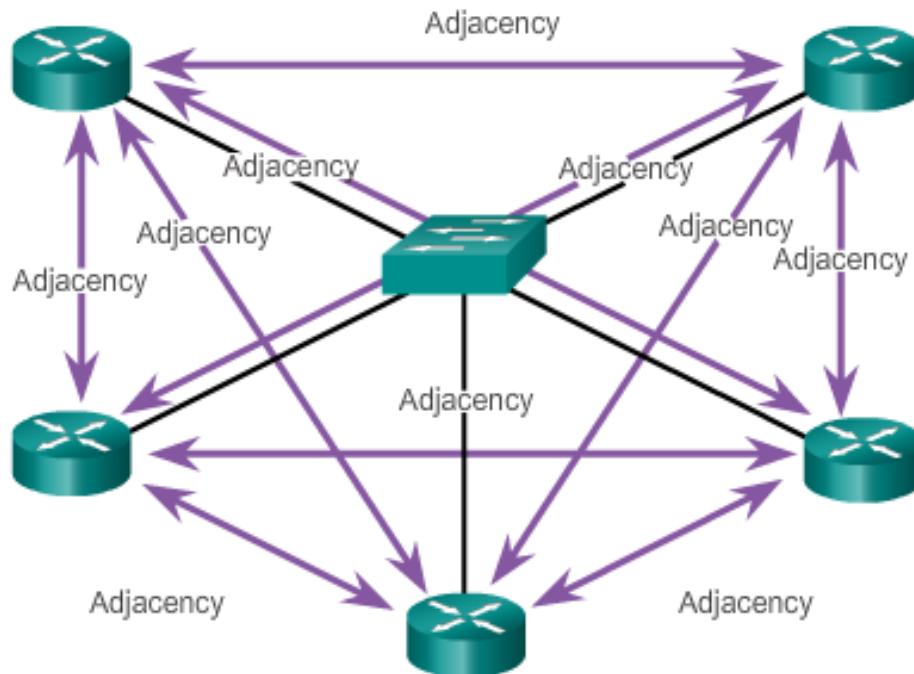
R1 has a default priority of 1 and the second highest router ID. It will be the BDR on this link.

R2 has a default priority of 1 and the highest router ID. It will be the DR on this link.

DR and BDR election only occurs on multi-access networks such as Ethernet LANs.

OSPF DR and BDR

Creating Adjacencies With Every Neighbor



Number of Adjacencies= $n(n-1)/2$
n=number of routers
Example: 5 routers $(5-1)/2=10$ adjacencies

Configure Single-area OSPFv2

Assigning Interfaces to an OSPF Area

```
R1(config)# router ospf 10
R1(config-router)# network 172.16.1.0 0.0.0.255 area 0
R1(config-router)# network 172.16.3.0 0.0.0.3 area 0
R1(config-router)# network 192.168.10.4 0.0.0.3 area 0
R1(config-router)#
R1#
```

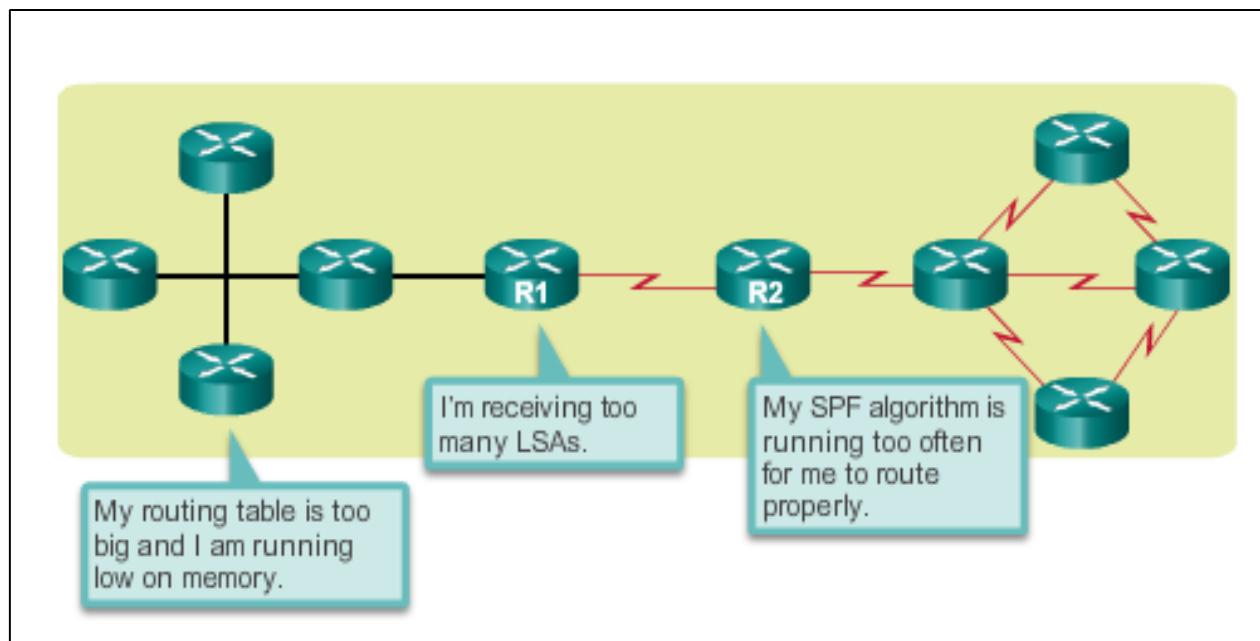
Single vs. Multi-Area OSPF

Why Multiarea OSPF?

Single-Area OSPF

Single-area OSPF is useful in smaller networks. If an area becomes too big, the following issues must be addressed:

- Large routing table (no summarization by default)
- Large link-state database (LSDB)
- Frequent SPF algorithm calculations

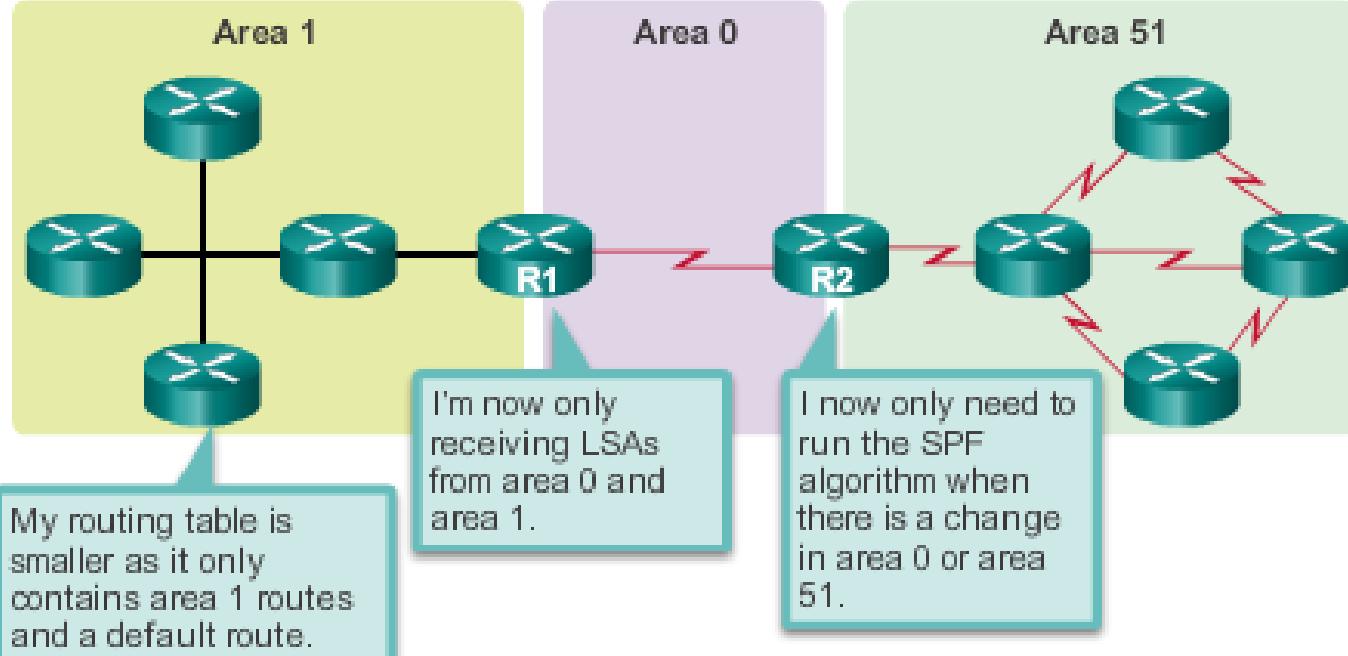


Why Multiarea OSPF?

Multiarea OSPF

Multiarea OSPF requires a hierarchical network design and the main area is called the backbone area, or area 0, and all other areas must connect to the backbone area.

Multi-Area OSPF Advantages

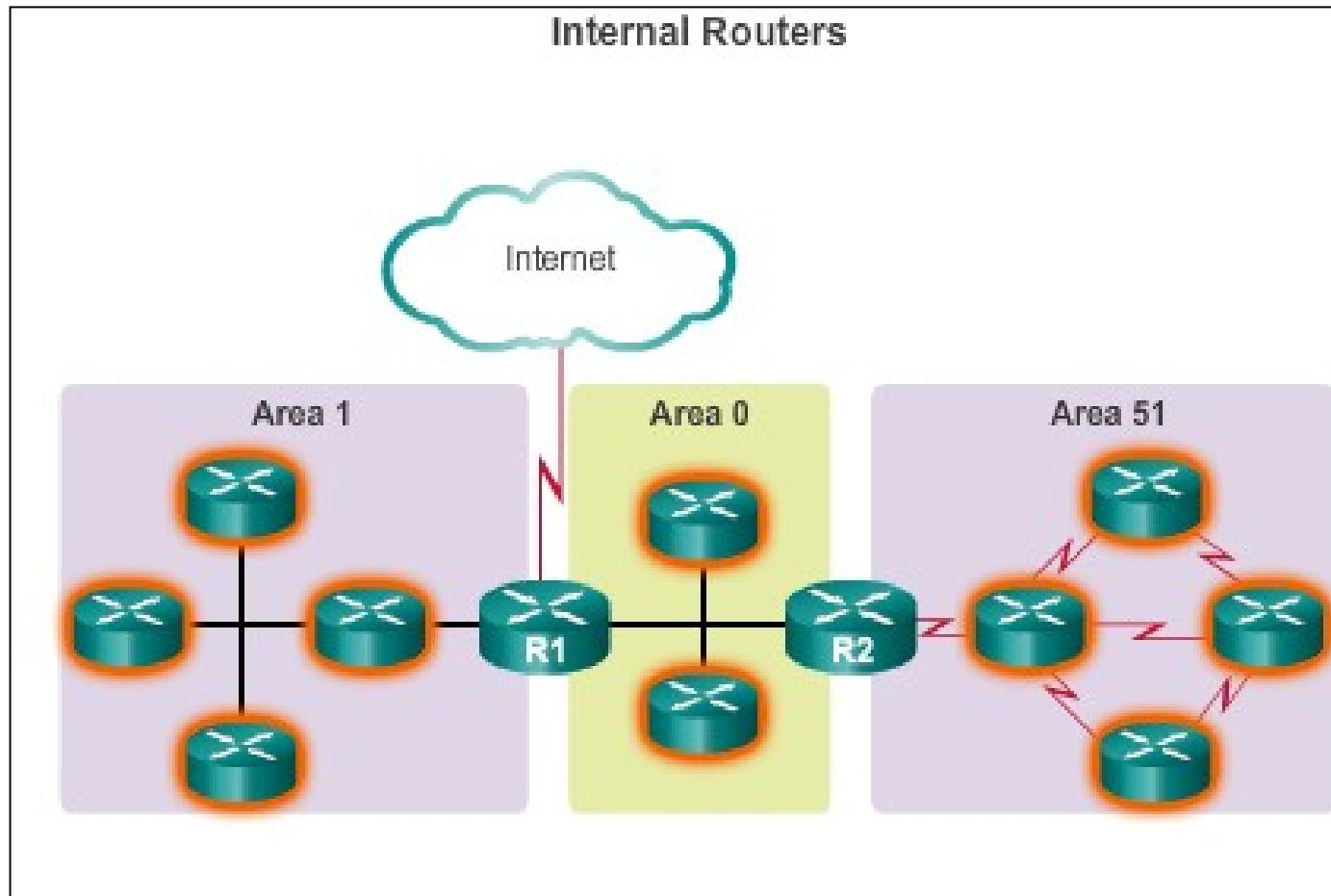


OSPF Two-Layer Area Hierarchy

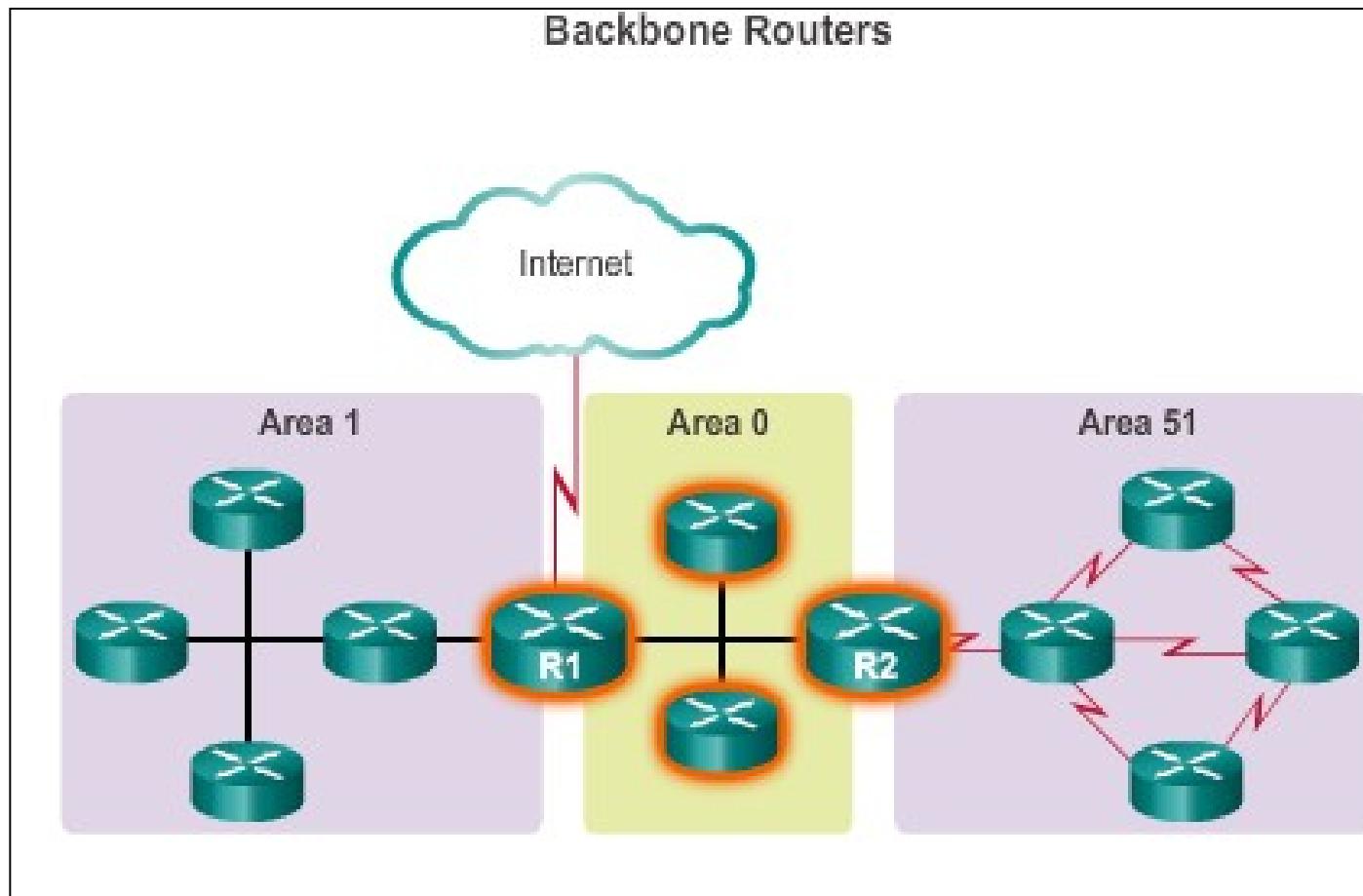
Multiarea OSPF is implemented in a two-layer area hierarchy:

- **Backbone (transit) area**
 - Area whose primary function is the fast and efficient movement of IP packets.
 - Interconnects with other OSPF area types.
 - Called OSPF area 0, to which all other areas directly connect.
- **Regular (nonbackbone) area**
 - Connects users and resources.
 - A regular area does not allow traffic from another area to use its links to reach other areas.

Types of OSPF Routers

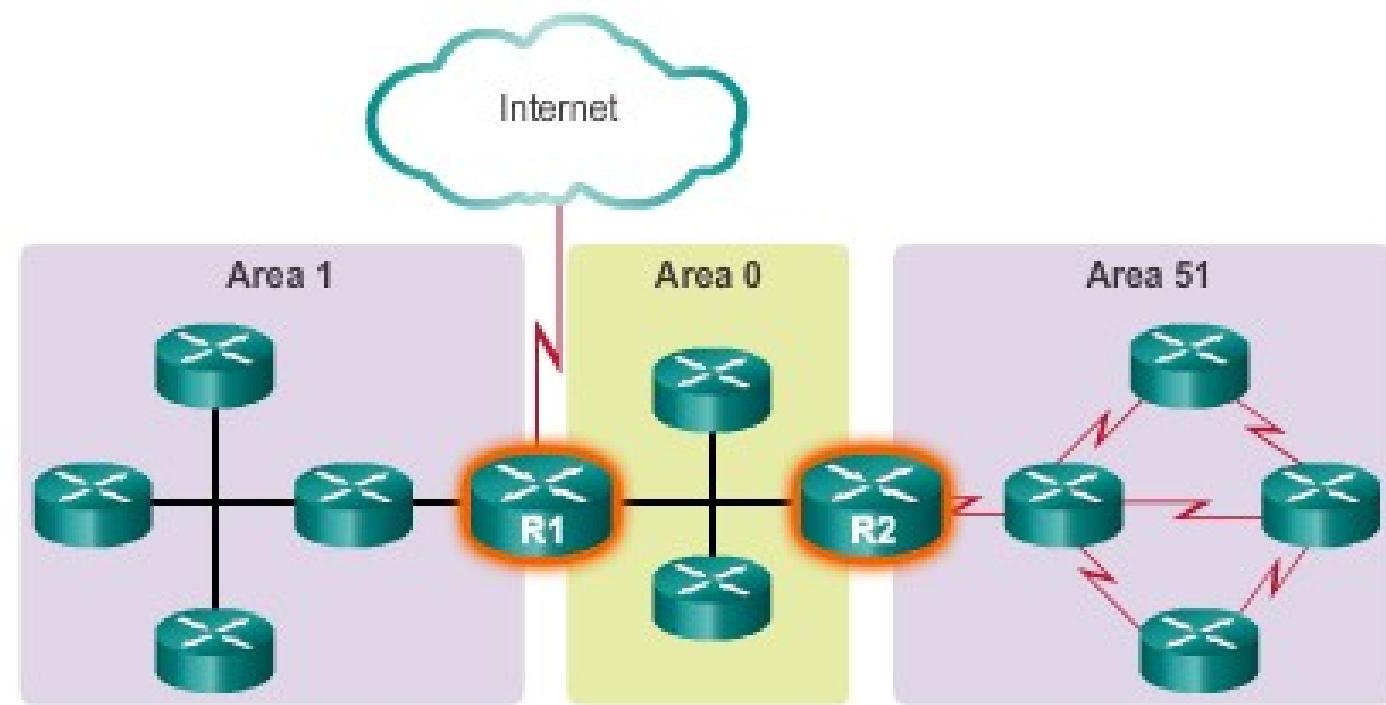


Types of OSPF Routers (cont.)



Types of OSPF Routers (cont.)

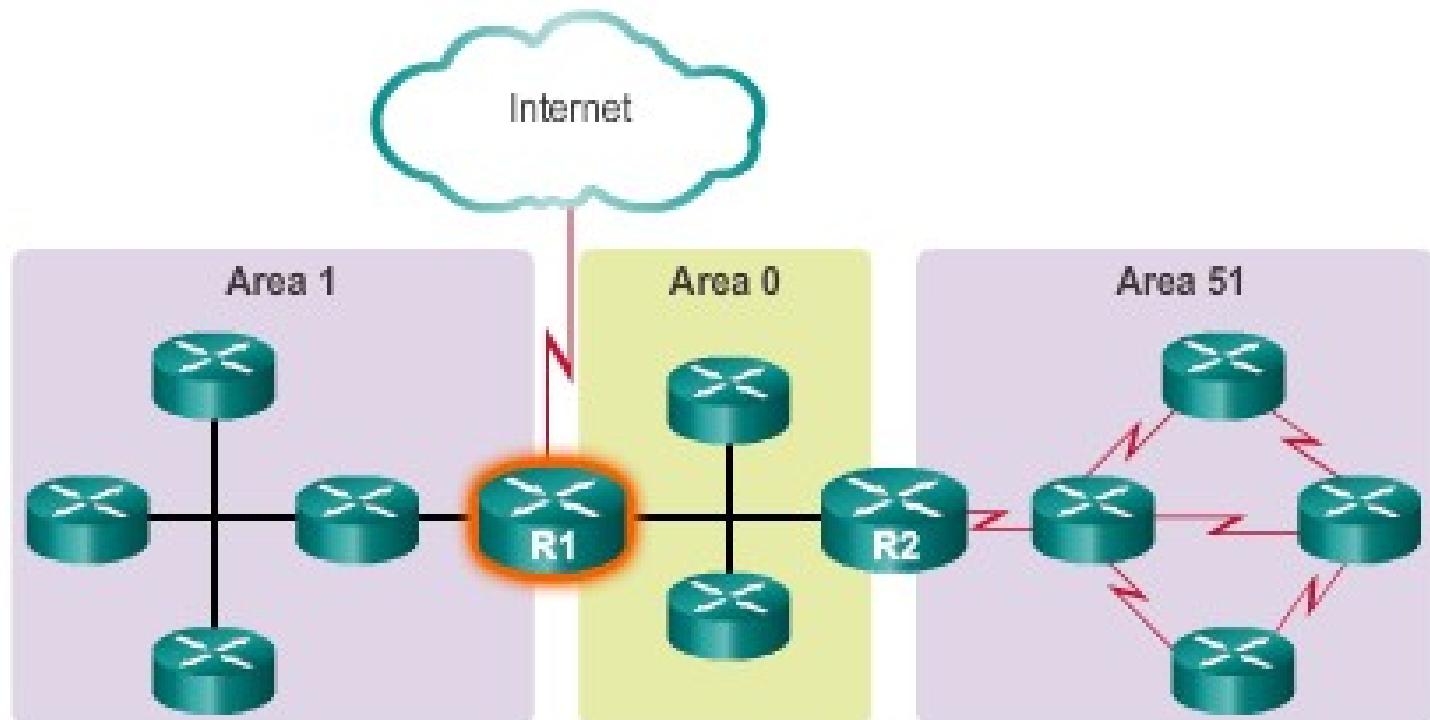
Area Border Routers (ABRs)



Why Multiarea OSPF?

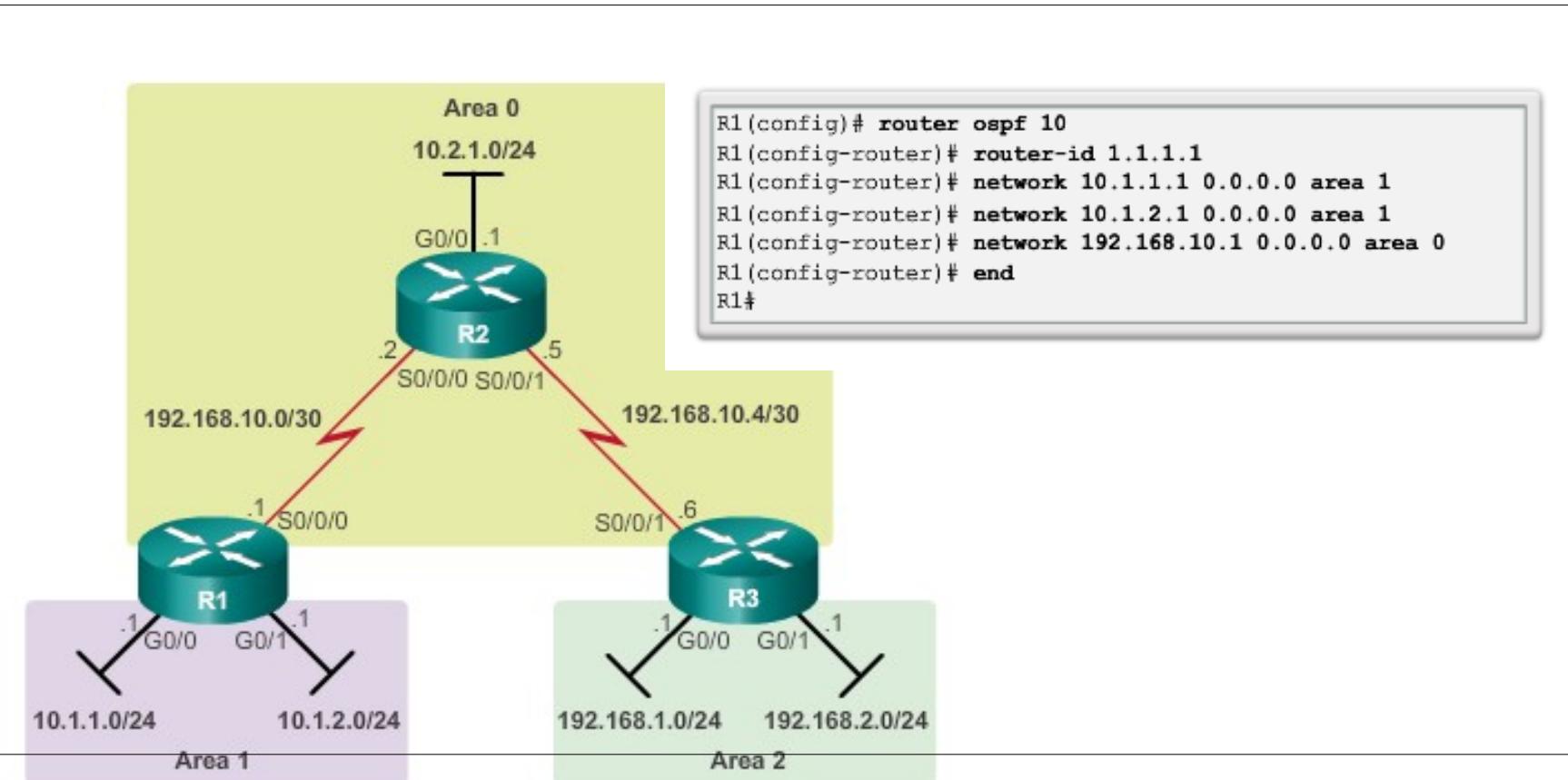
Types of OSPF Routers (cont.)

Autonomous System Boundary Router (ASBR)



Configuring Multi Area OSPF

Configuring Multiarea OSPF



Verifying Multiarea OSPF

The same verification commands are used to verify single-area OSPF and can be used to verify multiarea OSPF:

- **show ip ospf neighbor**
- **show ip ospf**
- **show ip ospf interface**

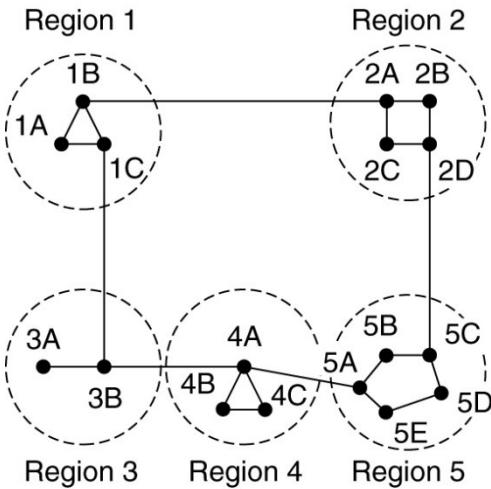
Commands specific to multiarea information include:

- **show ip protocols**
- **show ip ospf interface brief**
- **show ip route ospf**
- **show ip ospf database**

Chapter 5

The Network Layer Part 3

Hierarchical Routing



Full table for 1A

Dest.	Line	Hops
1A	-	-
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

Hierarchical table for 1A

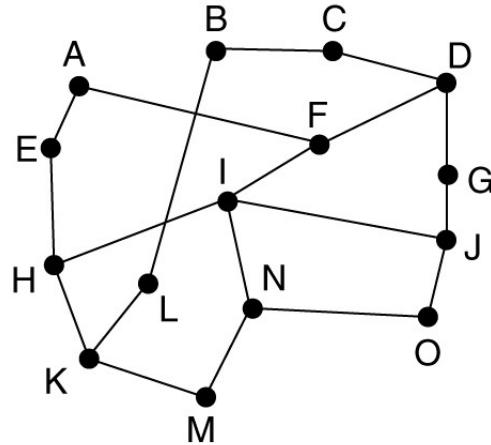
Dest.	Line	Hops
1A	-	-
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

Hierarchical routing.

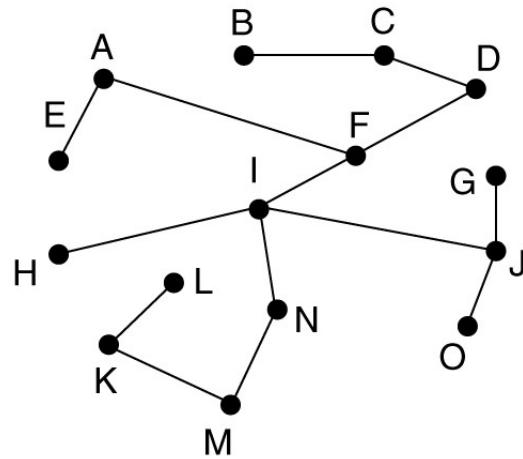
Broadcast Routing

- a) Applications: Weather reports to be sent to all hosts
 - To all
 - Flooding
 - Multi-destination routing
 - Using Sink tree (Spanning tree)
 - Reverse Path forwarding

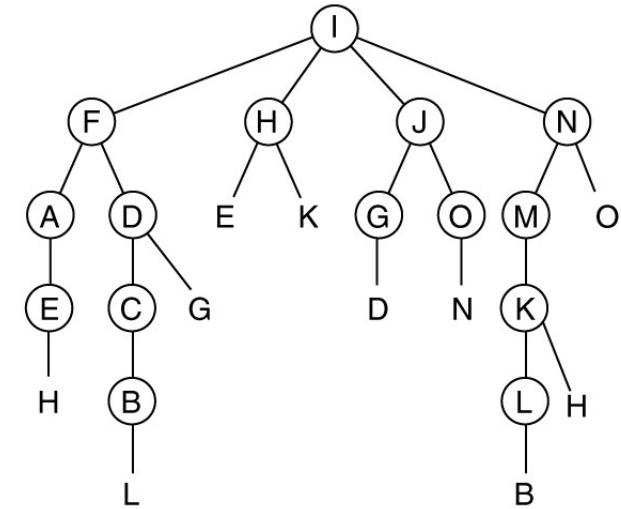
Broadcast Routing



(a)



(b)



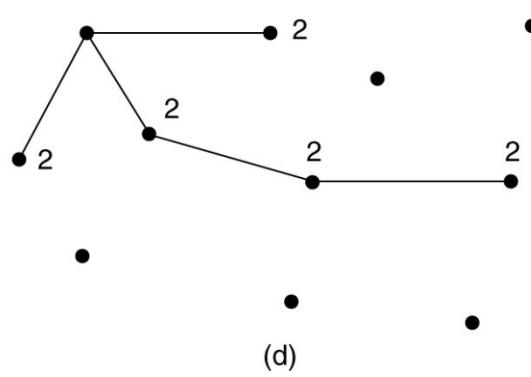
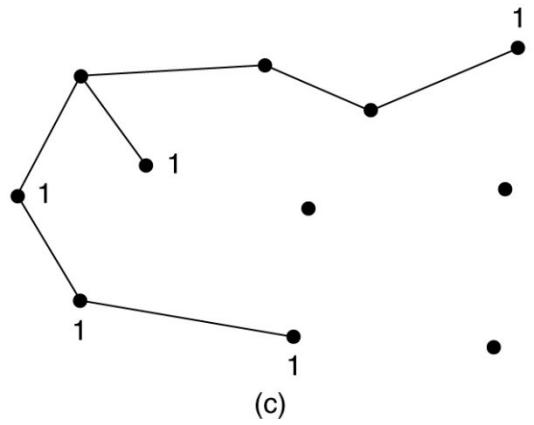
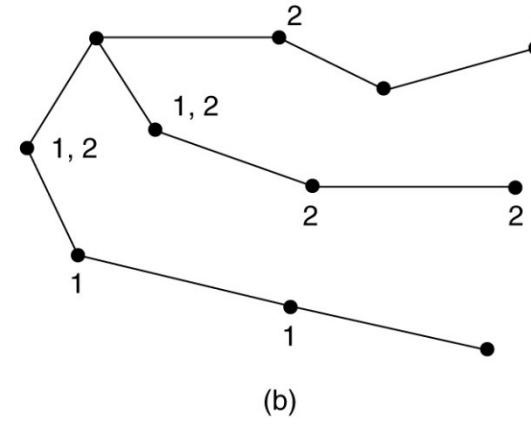
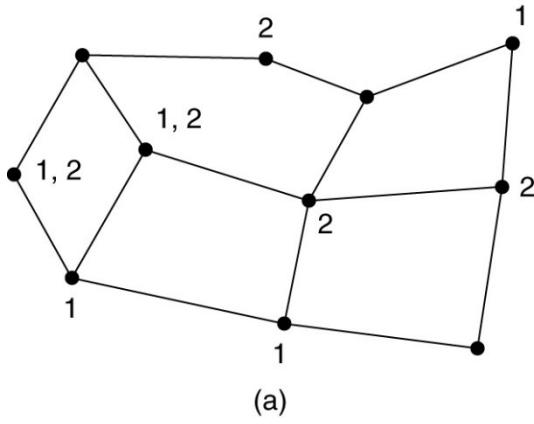
(c)

Reverse path forwarding. (a) A subnet. (b) a Sink tree. (c) The tree built by reverse path forwarding.

Reverse Path Forwarding

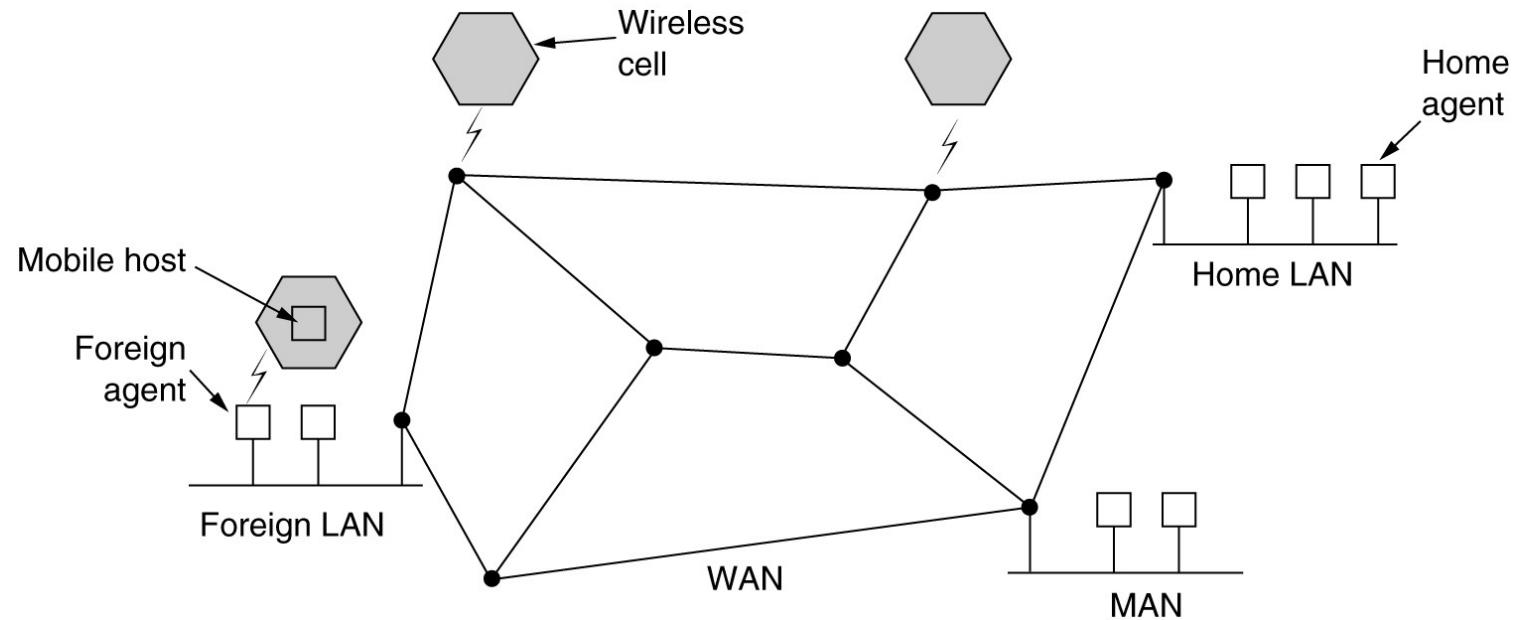
- Ensures loop-free forwarding of multicast packets
- Decision to forward traffic is based upon **source address** and not on destination address as in unicast routing
- When a multicast packet enters a router's interface, the router looks up the **list of networks that are reachable via that interface**
- If the router finds **a matching routing entry for the source IP address** of the multicast packet, the RPF check passes and the packet is forwarded to all other interfaces that are participating in that multicast group
- If the RPF check fails, the packet is dropped.
- By only forwarding packets that come into the interface that also holds the routing entry for the source of the packet, loops are prevented

Multicast Routing



- (a) A network. (b) A spanning tree for the leftmost router.
(c) A multicast tree for group 1. (d) A multicast tree for group 2.

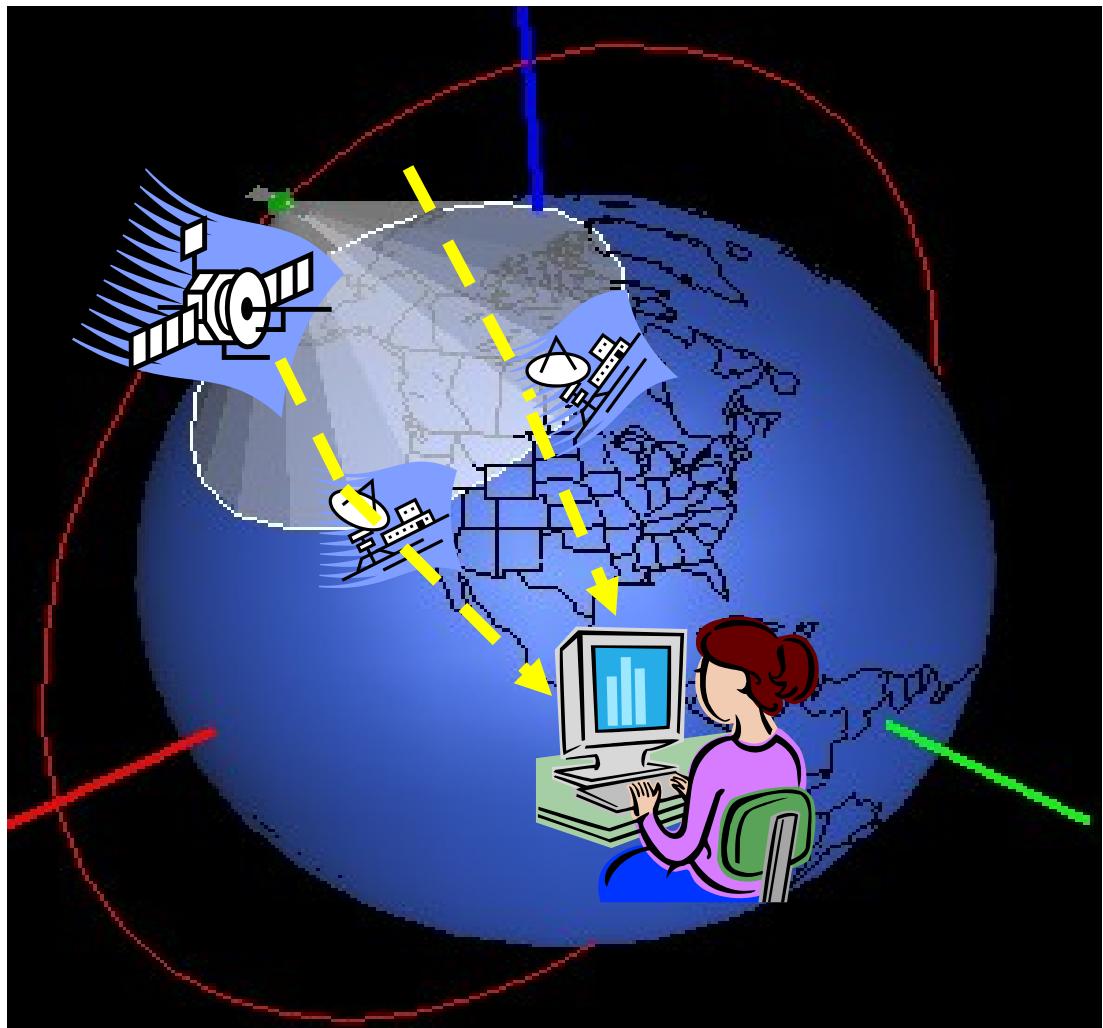
Routing for Mobile Hosts



A WAN to which LANs, MANs, and wireless cells are attached.

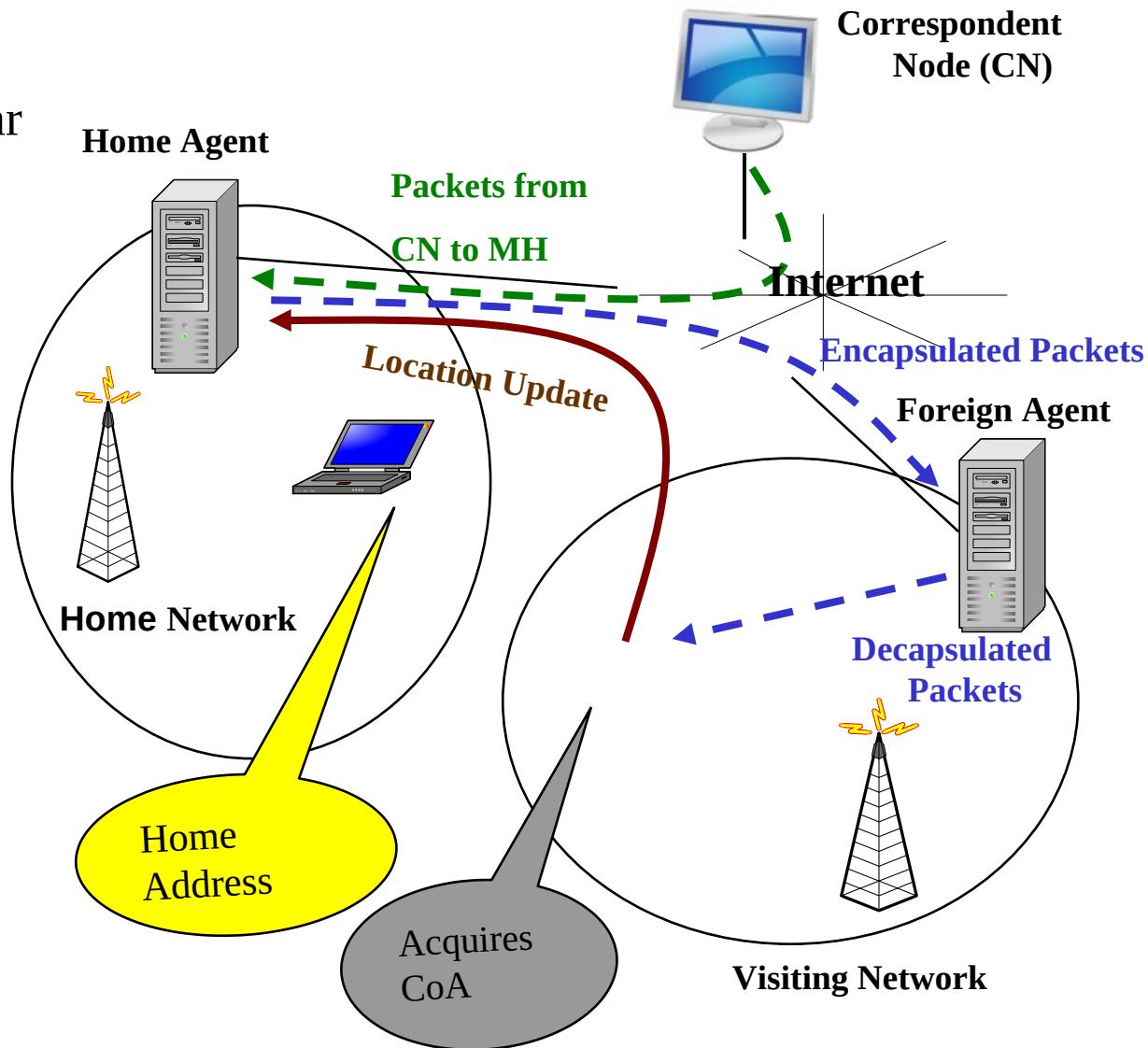
Why Mobility Protocols

- a) Satellites with IP-enabled devices capture videos, images and send them to control centers on earth
- b) Need to maintain continuous connectivity with remote computer
- c) Mobility protocols are required to ensure session continuity



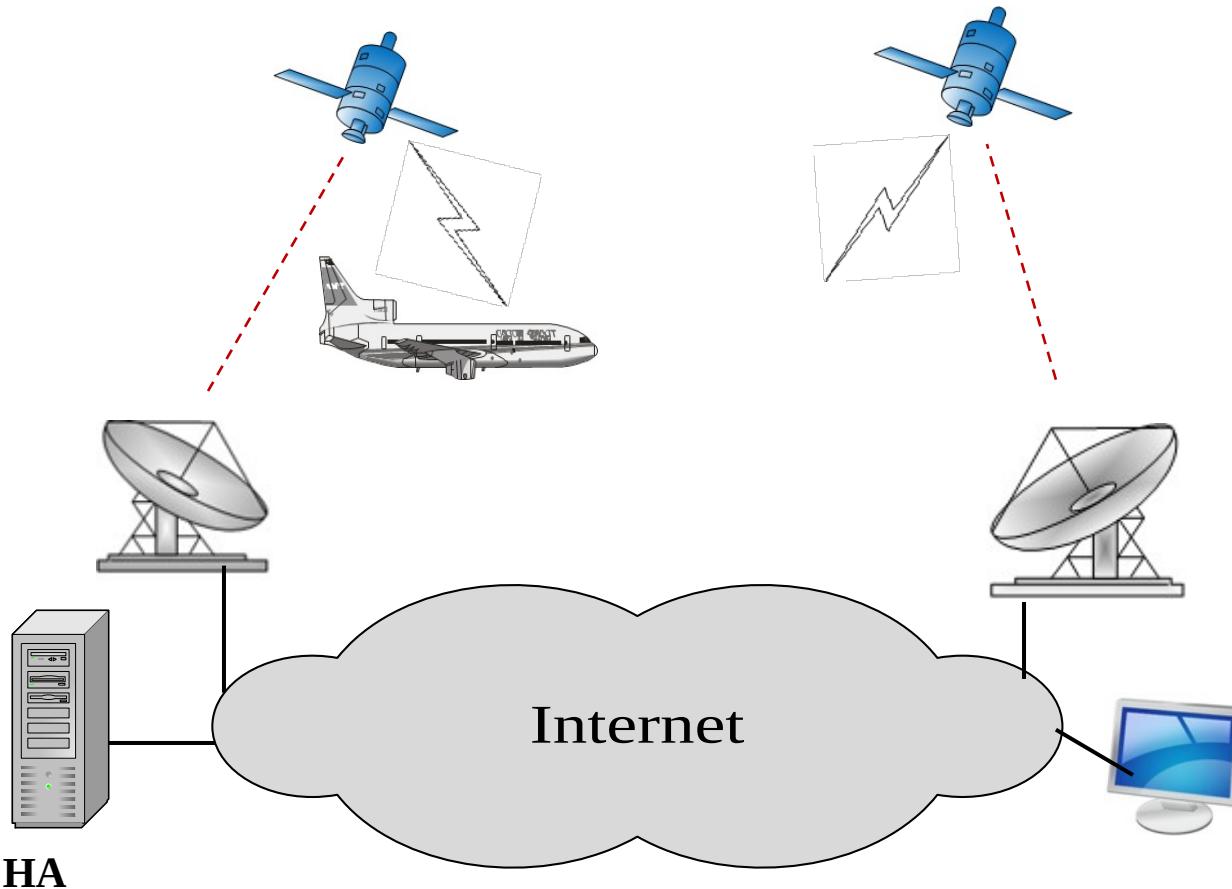
IETF Solution to IP Mobility: Mobile IP

- Employs mechanism similar to postal service mail forwarding
- Problems:
 - Inefficient routing
 - High handover latency
 - Packet loss



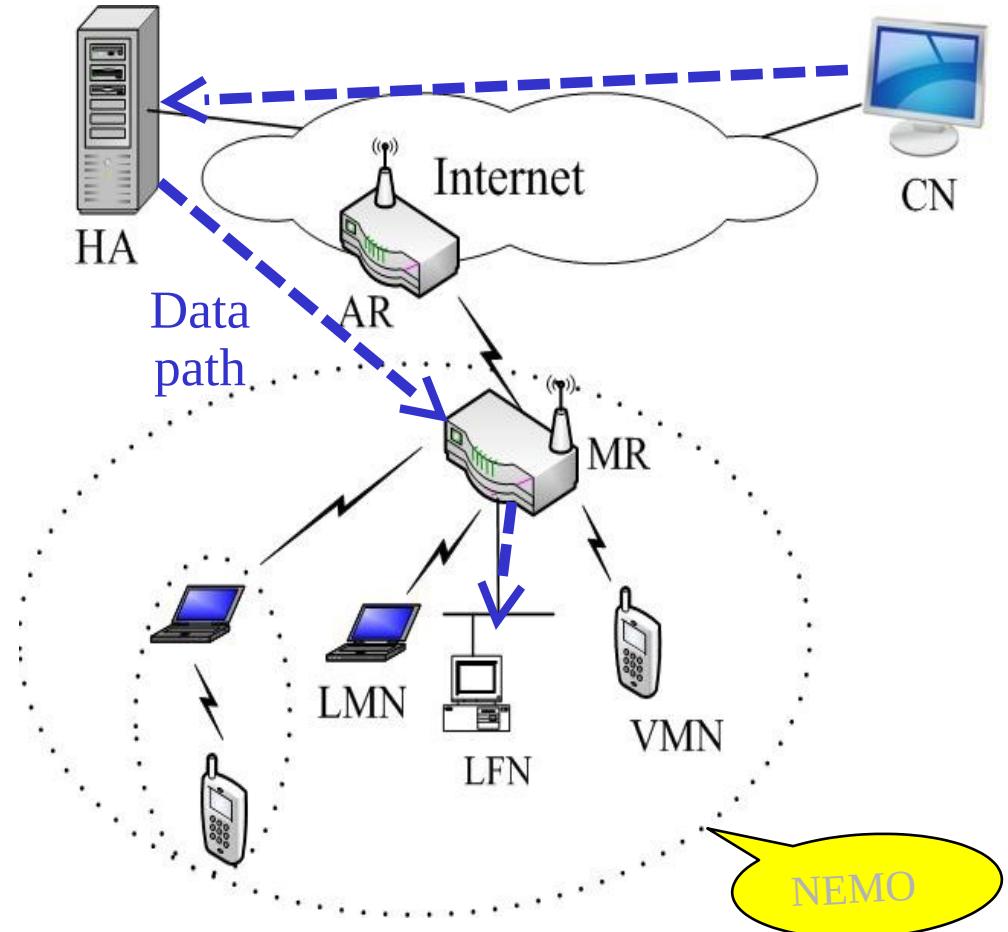
Network Mobility (NEMO)

- a) A collection of nodes moving as a unit (Example: airplanes, trains, ships)
- b) Mobility can be managed in an aggregated way in NEMO
- c) Mobile Router acts as default gateway and manages mobility on behalf of mobile network nodes

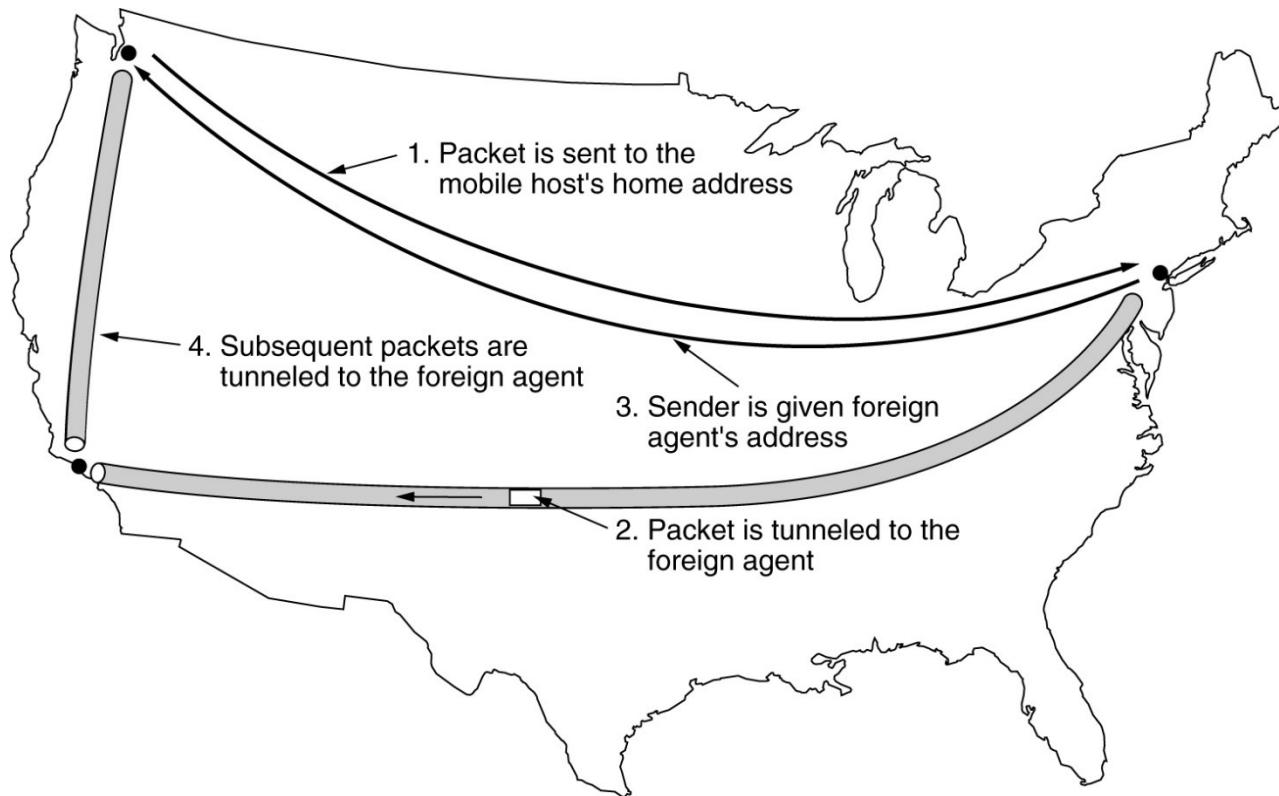


NEMO Architecture

- Inside NEMO
 - MR: Mobile Router
 - LFN: Local Fixed Node
 - LMN: Local Mobile node
 - VMN: Visiting Mobile Node
- Problems:
 - Routing through HA
 - Heavy load on HA
 - Drop in throughput during handover



Routing for Mobile Hosts (2)



Packet routing for mobile users.

Routing in Ad Hoc Networks

Possibilities when the routers are mobile:

1. Military vehicles on battlefield.
 - No infrastructure.
2. A fleet of ships at sea.
 - All moving all the time
3. Emergency works at earthquake .
 - The infrastructure destroyed.
4. A gathering of people with notebook computers.
 - In an area lacking 802.11.

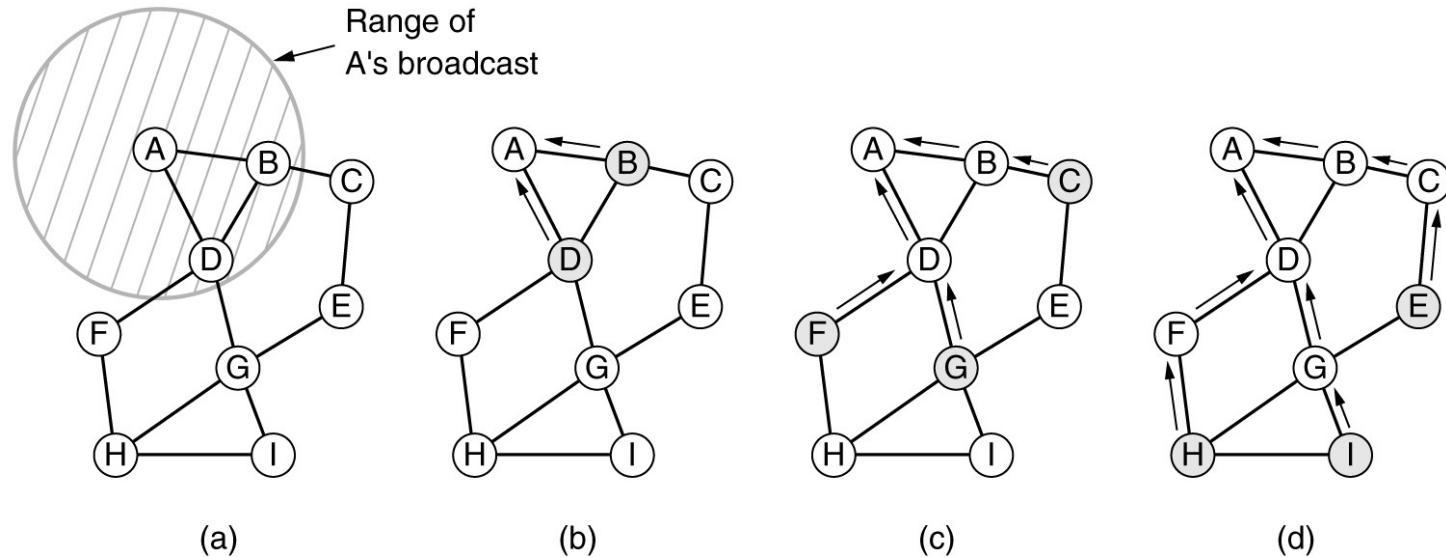
Ad hoc networks

- The topology may be changing all the time
- Hence, routing will also change frequently
- MANET
- VANET
- FANET

AODV

- AODV (Ad hoc On-demand Distance Vector) routing protocol is a classic Ad hoc routing protocol.
 - A route is determined when somebody wants to send packet to that destination
- There are other protocols not discussed here

Route Discovery



- a) Range of A's broadcast.
- b) After B and D have received A's broadcast.
- c) After C, F, and G have received A's broadcast.
- d) After E, H, and I have received A's broadcast.

Shaded nodes are new recipients. Arrows show possible reverse routes.

Route Discovery (2)

Source address	Request ID	Destination address	Source sequence #	Dest. sequence #	Hop count
----------------	------------	---------------------	-------------------	------------------	-----------

Format of a ROUTE REQUEST packet.

Route Discovery (3)

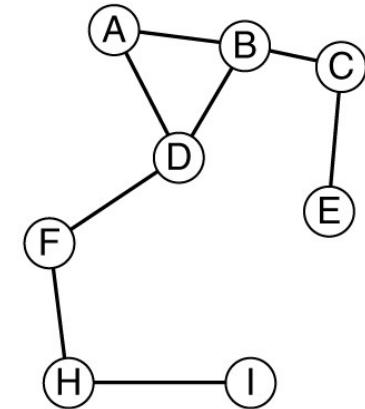
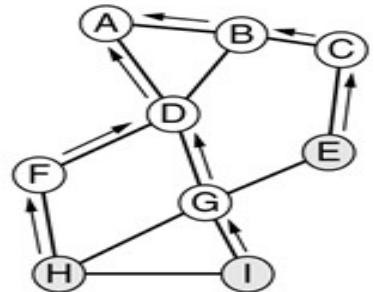
Source address	Destination address	Destination sequence #	Hop count	Lifetime
----------------	---------------------	------------------------	-----------	----------

Format of a ROUTE REPLY packet.

Route Maintenance

Dest.	Next hop	Distance	Active neighbors	Other fields
A	A	1	F, G	
B	B	1	F, G	
C	B	2	F	
E	G	2		
F	F	1	A, B	
G	G	1	A, B	
H	F	2	A, B	
I	G	2	A, B	

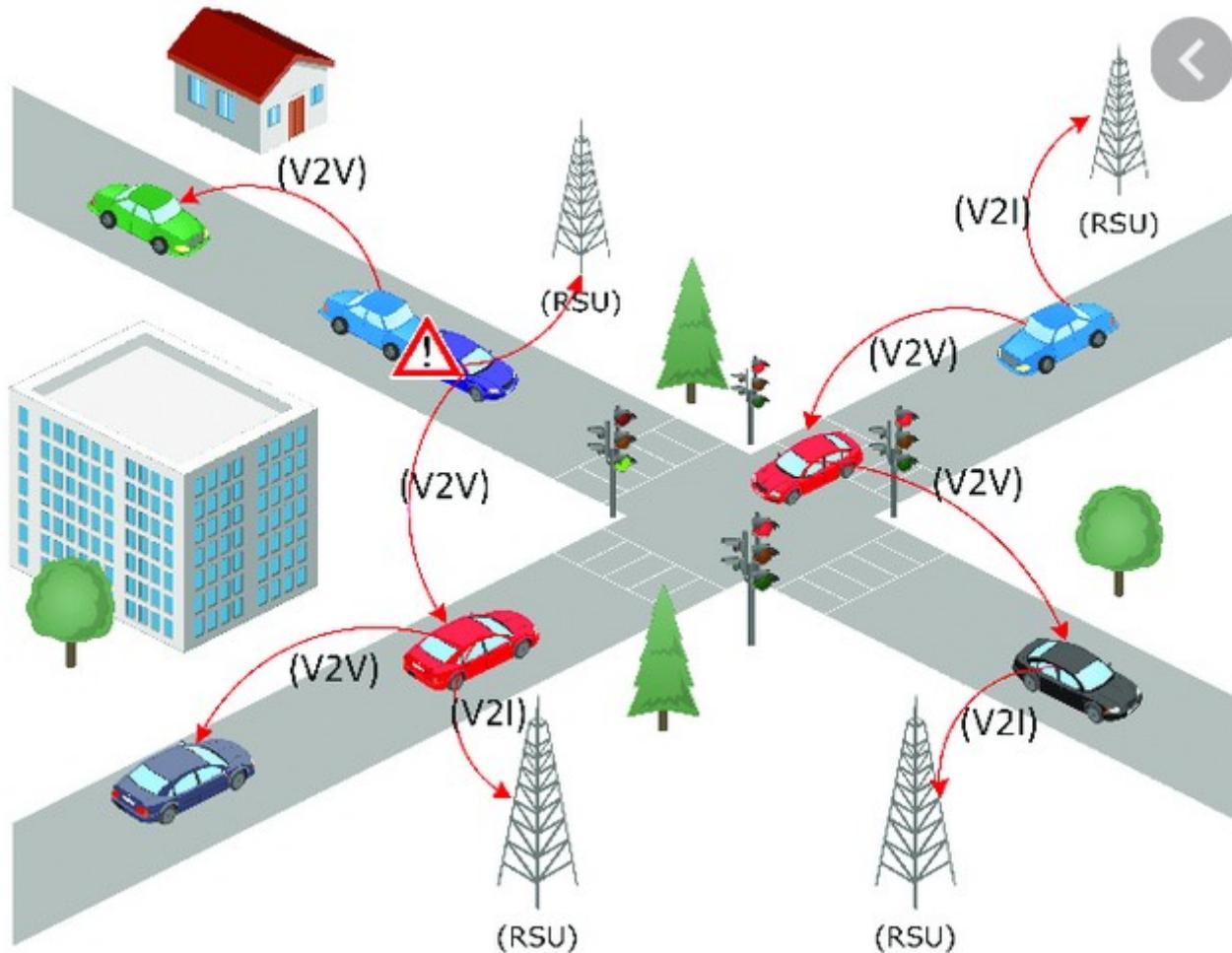
(a)



(b)

- (a) D's routing table before G goes down.
- (b) The graph after G has gone down.

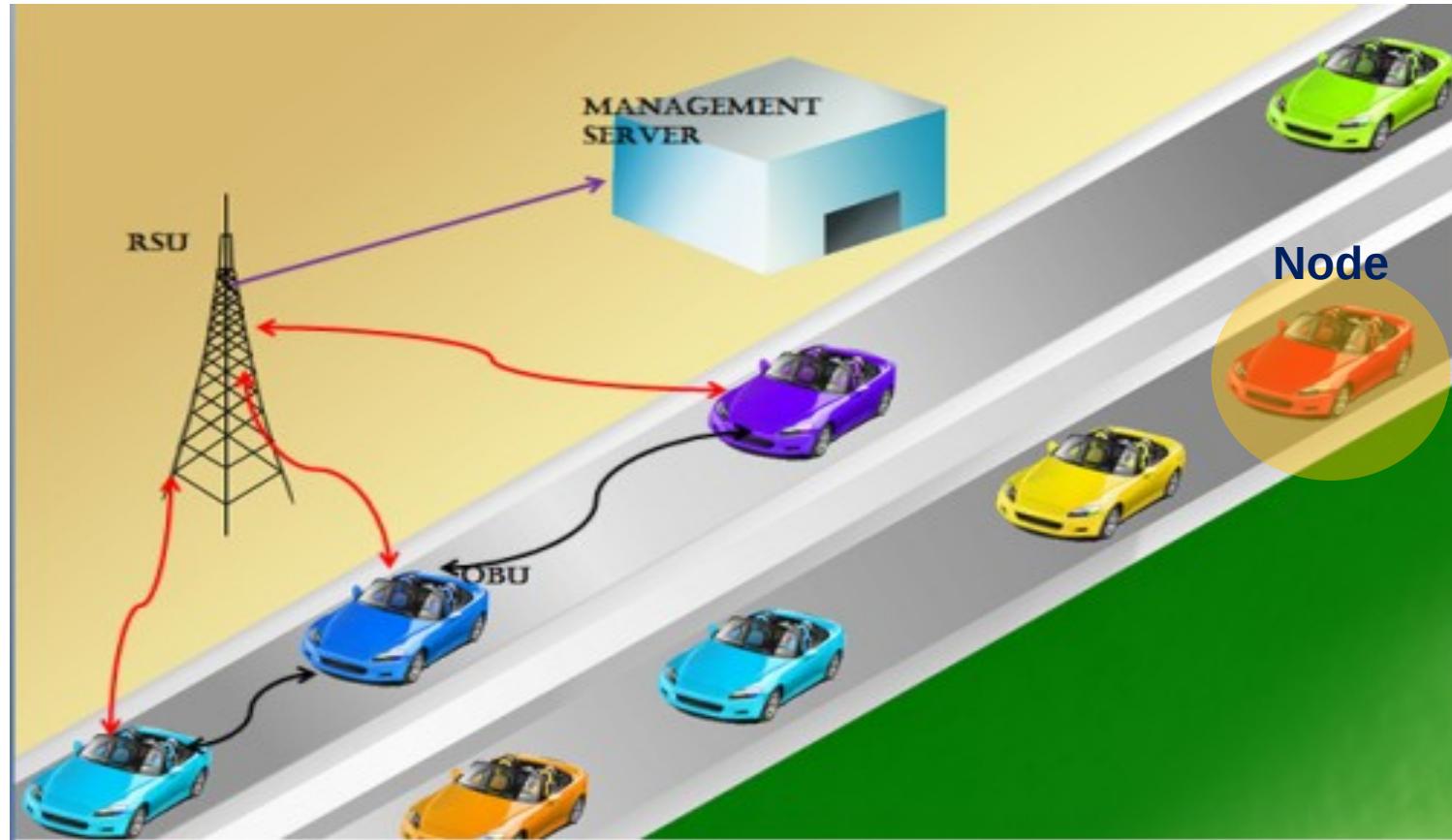
VANET



What is VANET ?

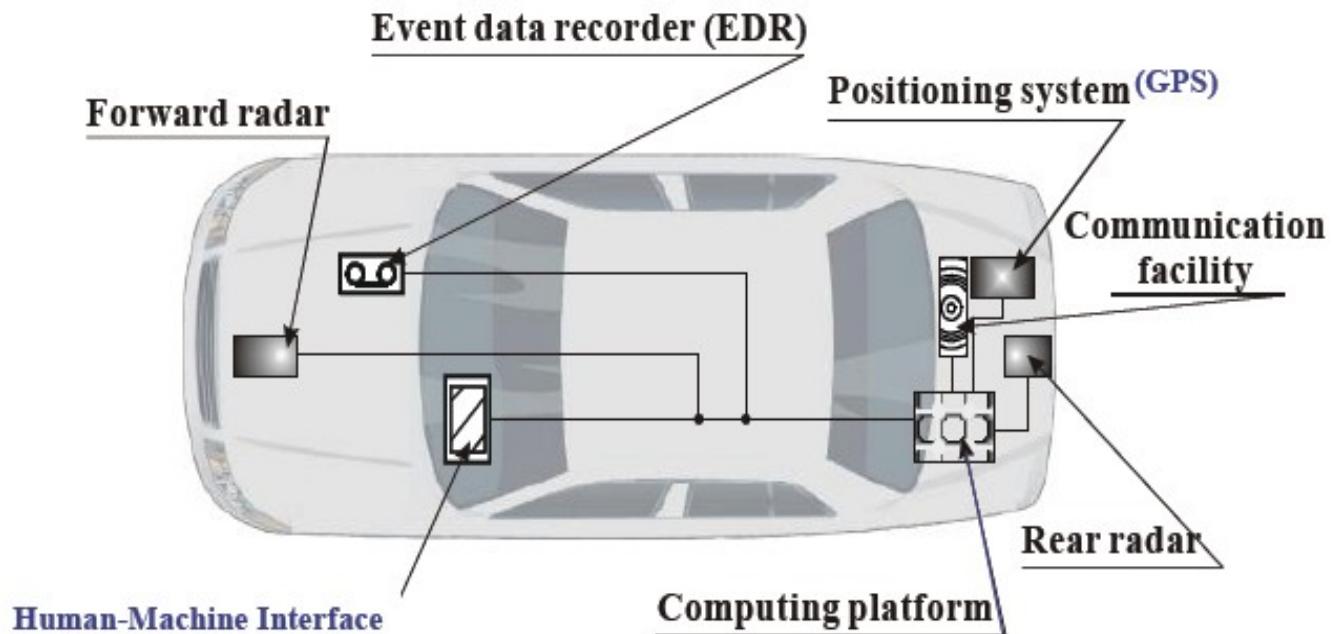
Next generation intelligent vehicular networking technologies

Uses moving car as nodes to create mobile network



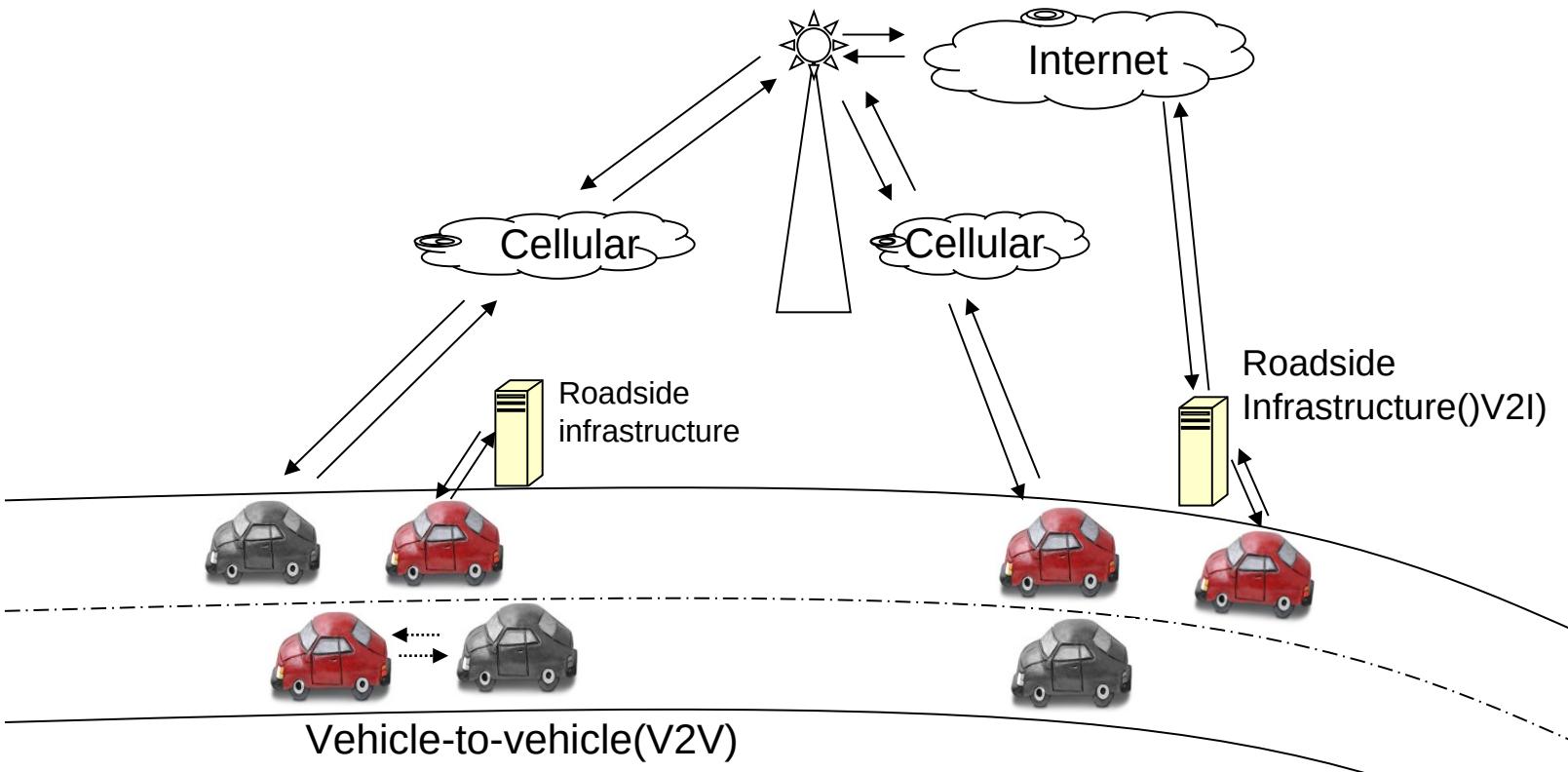
Sensors in Modern car

Equipped with GPS, sensors, computing and comm devices



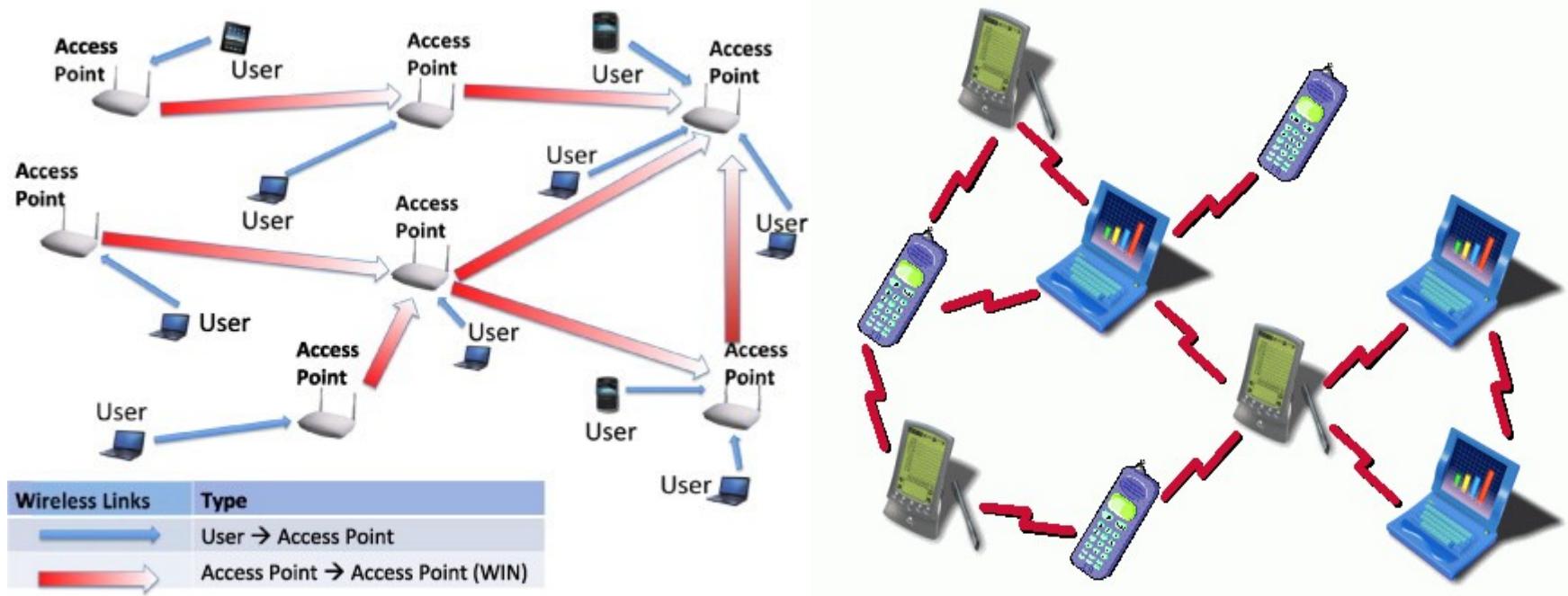
V2V and V2I Communication

Vehicles talk to other vehicles (V2V) and road-side infrastructure(V2I)

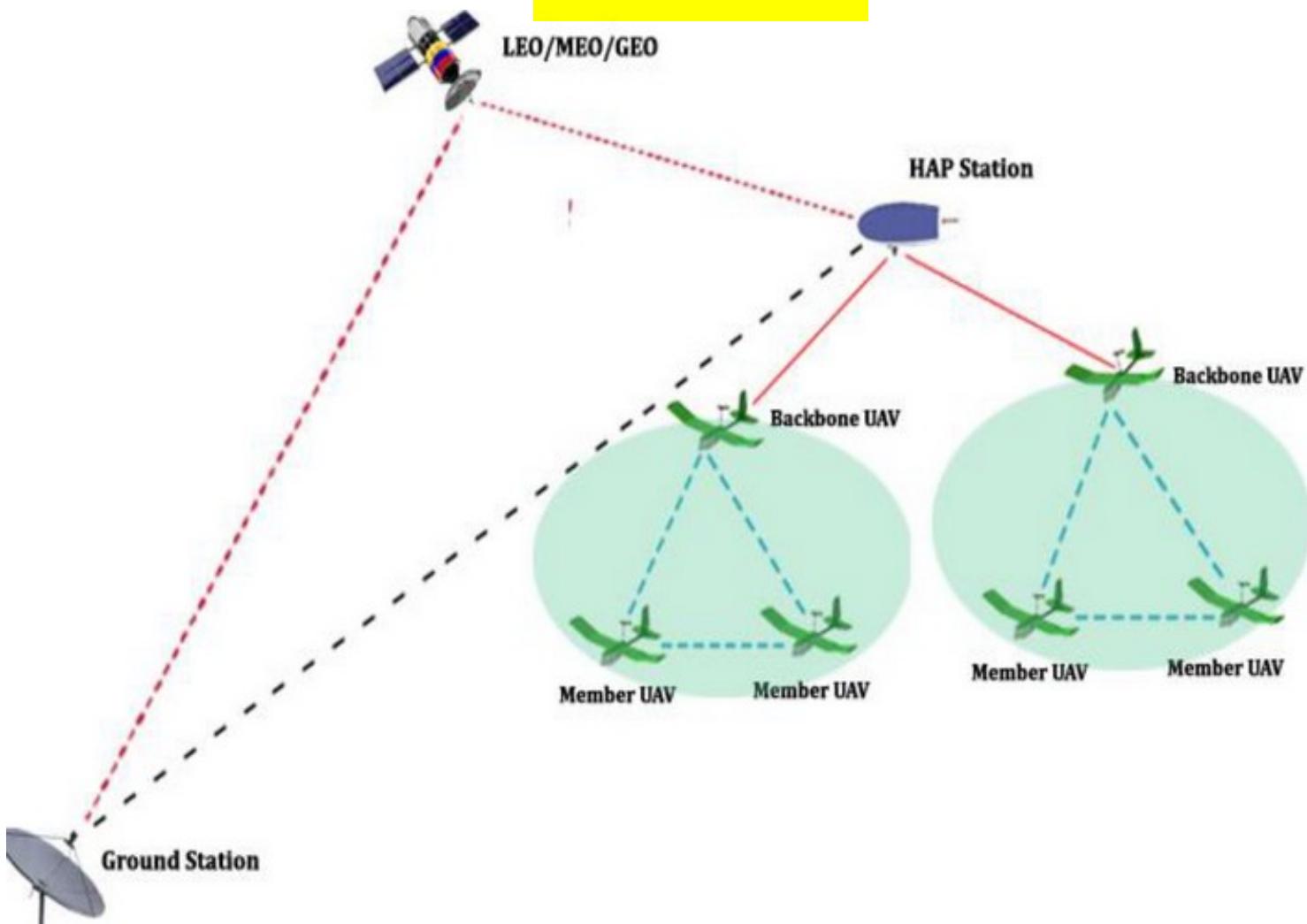


Infrastructure-less

- A wireless ad hoc network is a decentralized type of wireless network
 - Not rely on a preexisting infrastructure: routers in wired networks or access points



FANET



Unmanned Aerial Vehicle (UAV)

- Unmanned aerial vehicle (UAV), normally known as **drone**, is an aircraft without a human pilot involved.
- UAV is either controlled autonomously by on-board computers or by the remote control of a pilot on the ground
- Previously, used for remotely piloted aircraft.
- Nowadays, UAV's are used for growing number of civil applications.

Fanet vs. Vanet vs. Manet

- a) FANET can be viewed as a special form of MANET and
- b) VANET. But there are some difference
- ✓ Mobility degree

FANET nodes is much higher than the mobility degree of MANET or VANET nodes. MANET and VANET nodes are walking men and cars respectively, FANET nodes fly in the sky.

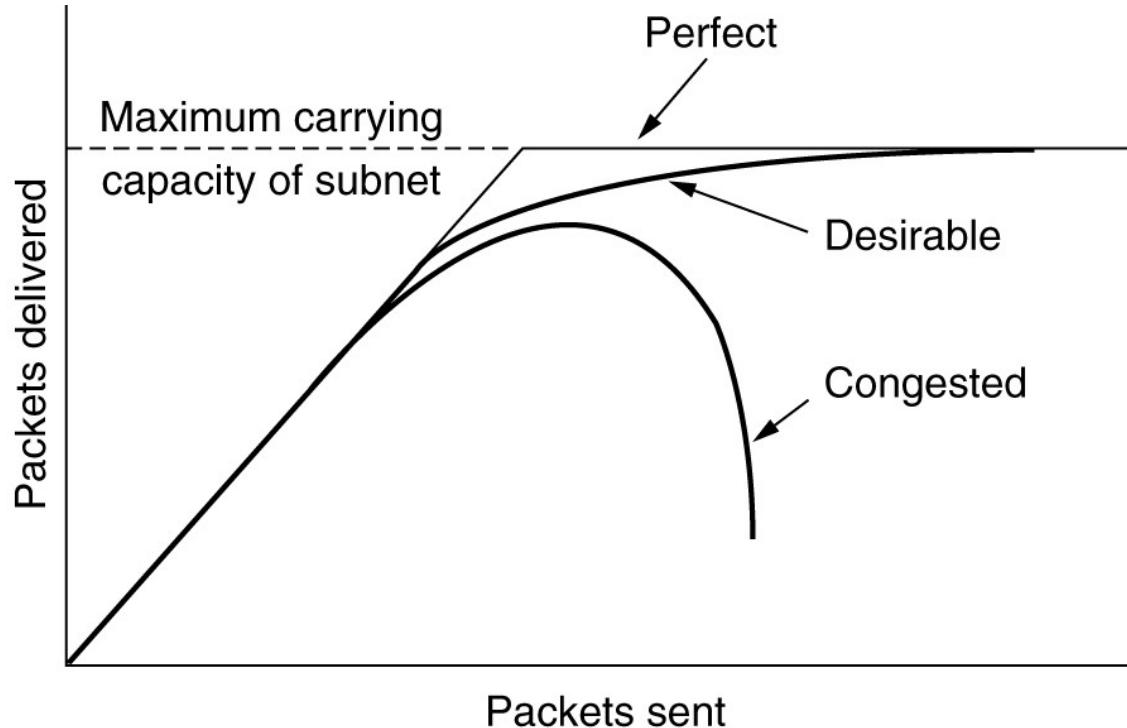
- ✓ Changing topology
- ✓ For FANET nodes, the topology changes more frequently than the network topology of a typical MANET or even VANET.

	MANET	VANET	FANET
Node mobility	Low	High	Very high
Mobility model	Random	Regular	Regular for predetermined paths, but special mobility models for autonomous multi-UAV systems
Node density	Low	High	Very low
Topology change	Slow	Fast	Fast
Radio propagation model	Close to ground, LoS is not available for all cases	Close to ground, LoS is not available for all cases	High above the ground, LoS is available for most of the cases
Power consumption and network lifetime	Energy efficient protocols	Not needed	Energy efficiency for mini UAVs, but not needed for small UAVs
Computational power	Limited	High	High
Localization	GPS	GPS, AGPS, DGPS	GPS, AGPS, DGPS, IMU

Congestion Control Algorithms

- General Principles of Congestion Control
- Congestion Prevention Policies
- Congestion Control in Virtual-Circuit Subnets
- Congestion Control in Datagram Subnets
- Load Shedding
- Jitter Control

Congestion



When too much traffic is offered, congestion sets in and performance degrades sharply.

General Principles of Congestion Control

1. Monitor the system .
 - detect when and where congestion occurs.
2. Pass information to where action can be taken.
3. Adjust system operation to correct the problem.

Congestion Prevention Policies

Layer	Policies
Transport	<ul style="list-style-type: none">• Retransmission policy• Out-of-order caching policy• Acknowledgement policy• Flow control policy• Timeout determination
Network	<ul style="list-style-type: none">• Virtual circuits versus datagram inside the subnet• Packet queueing and service policy• Packet discard policy• Routing algorithm• Packet lifetime management
Data link	<ul style="list-style-type: none">• Retransmission policy• Out-of-order caching policy• Acknowledgement policy• Flow control policy

Policies that affect congestion.

Data Link layer policies

- GoBack N (heavy load) vs. Selective repeat (less load)
- Out of order caching policy is closely related
- Ack Policy
 - Ack immediately will generate extra traffic
 - Piggybacked Ack
- Flow control policy
 - Tight window bound reduces data rate and can fight congestion

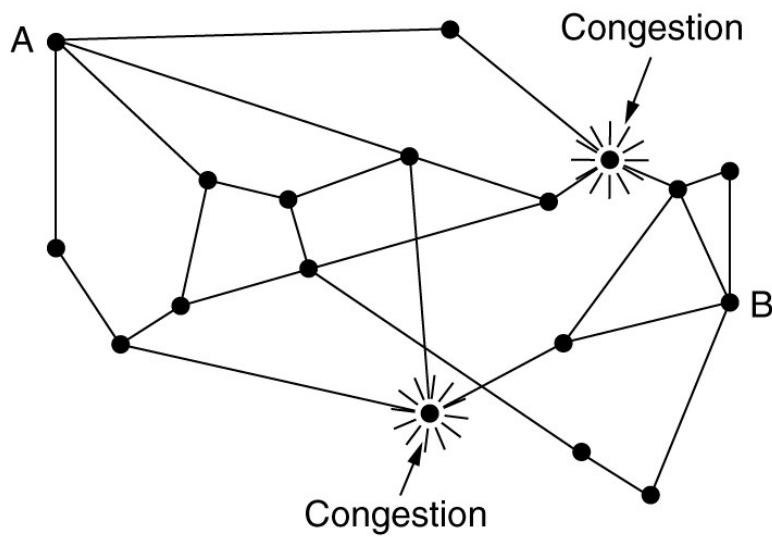
Network layer policies

- VC vs Datagram
- Routers to have one queue per line or shared queue
- Discard policy: Good policy vs bad policy (New / old packet):
 - Milk policy /Wine Policy
- Good routing alg: can spread the traffic

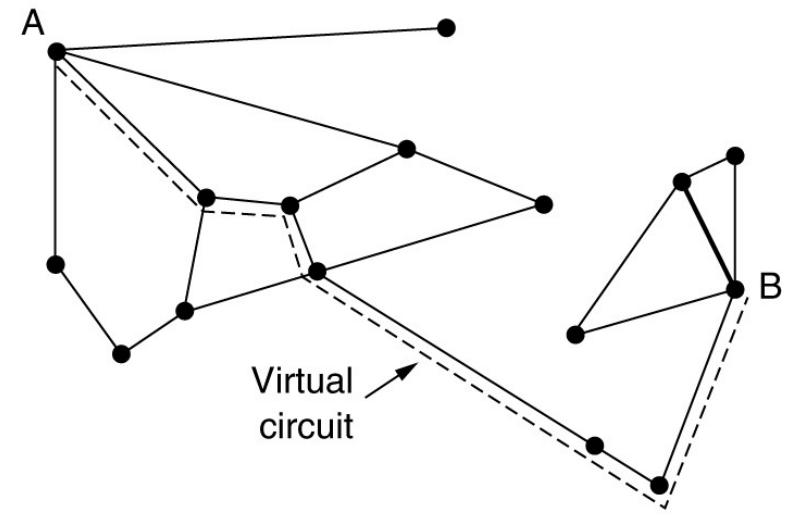
Transport layer policies

- Similar to DL layer policies => one single link
- However, timeout determination is more complex in Transport layer than DL layer (multi-hop)
- Timeout too short: extra packets are sent
- Timeout too long: slow response in case of lost segment

Congestion Control in Virtual-Circuit Subnets



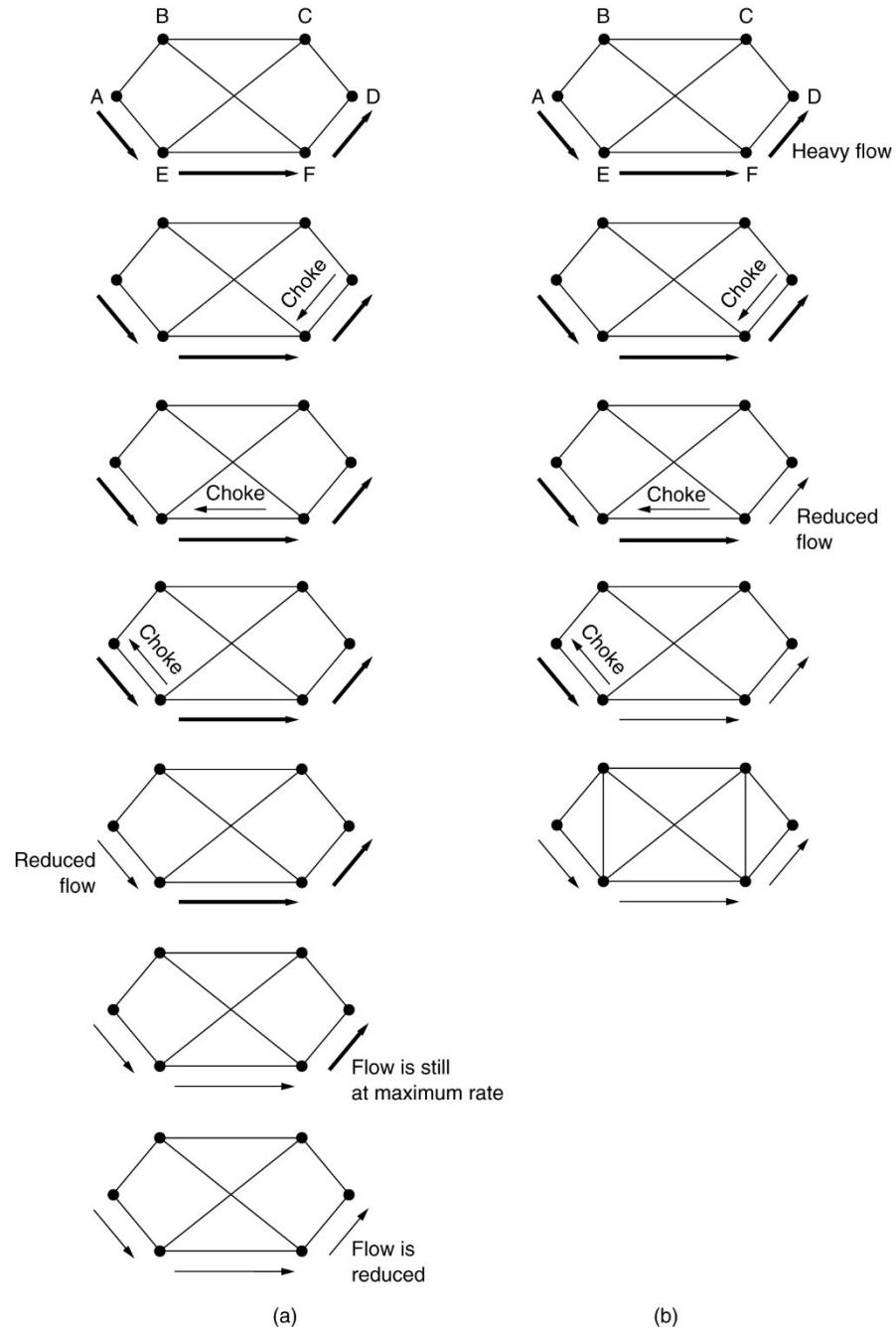
(a)



(b)

(a) A congested subnet. (b) A redrawn subnet, eliminates congestion and a virtual circuit from A to B.

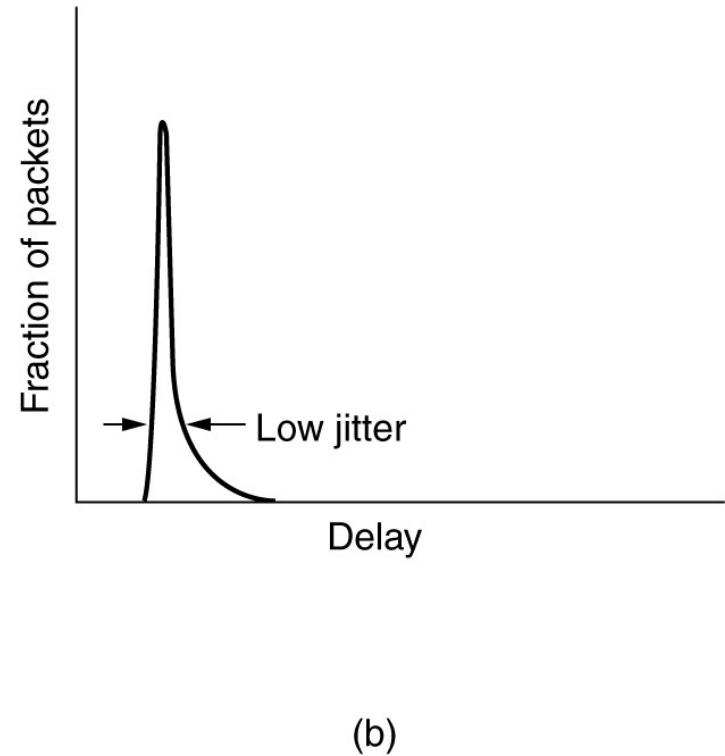
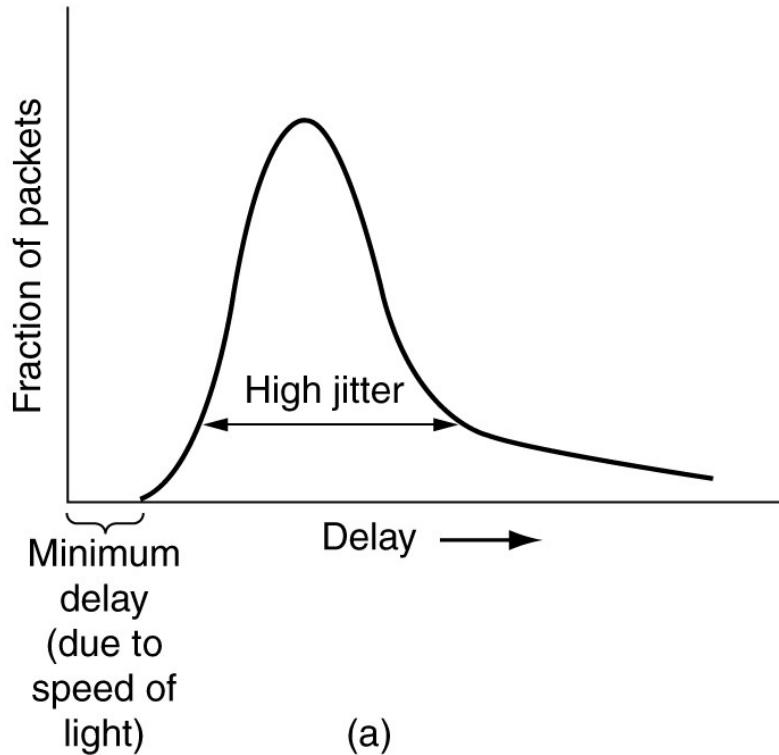
Hop-by-Hop Choke Packets



Random Early Detection

- Routers maintain a running average of their queue length
- If the average queue length exceeds some threshold, the line is said to be congested.
- Packets are dropped
- Inform other routers: Choke packets (adds traffic)
- Alternative approach, implicit (not to notify, just drop)

Jitter Control



(a) High jitter. (b) Low jitter.

Quality of Service

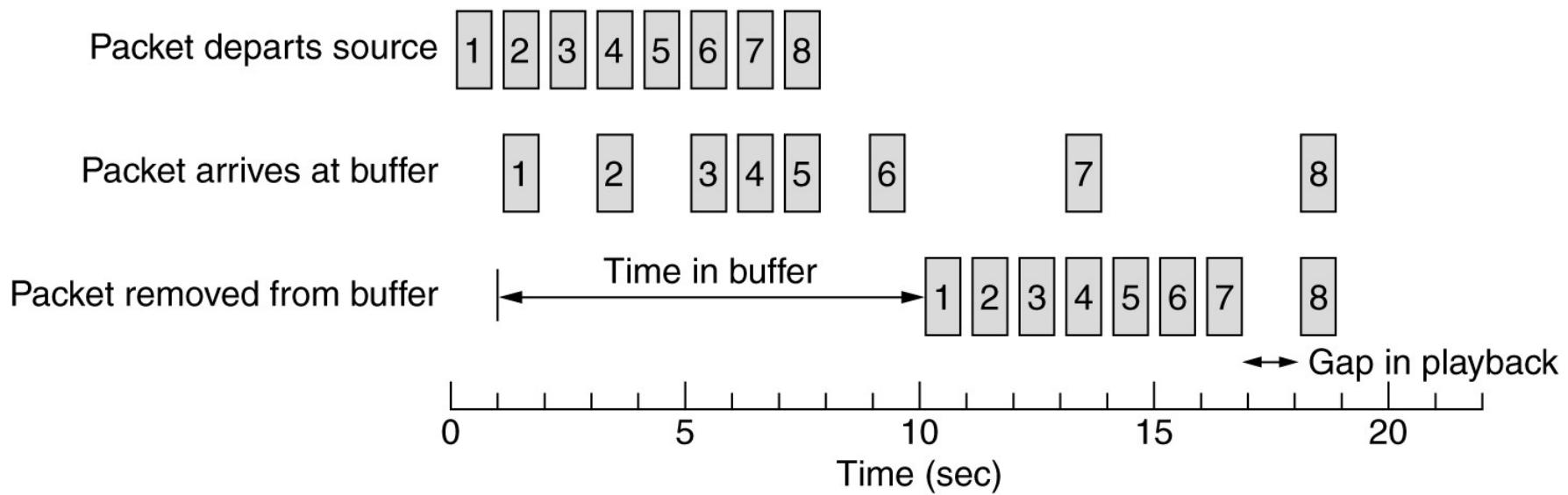
- Requirements
- Techniques for Achieving Good Quality of Service
- Integrated Services
- Differentiated Services
- Label Switching and MPLS

Requirements

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	High	Low	Low	Low
File transfer	High	Low	Low	Medium
Web access	High	Medium	Low	Medium
Remote login	High	Medium	Medium	Low
Audio on demand	Low	Low	High	Medium
Video on demand	Low	Low	High	High
Telephony	Low	High	High	Low
Videoconferencing	Low	High	High	High

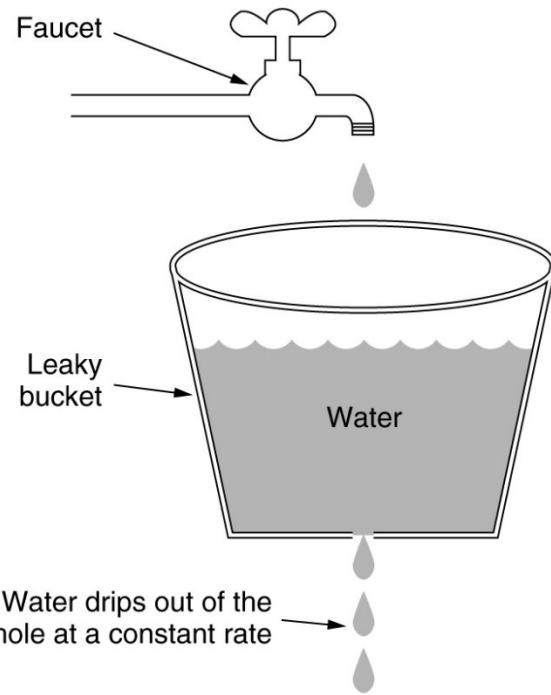
How stringent the quality-of-service requirements are.

Buffering

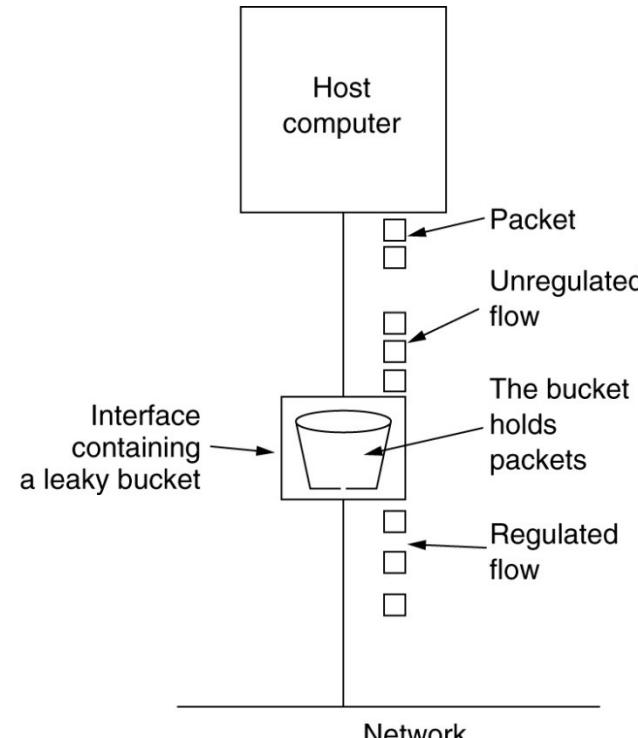


Smoothing the output stream by buffering packets.

The Leaky Bucket Algorithm



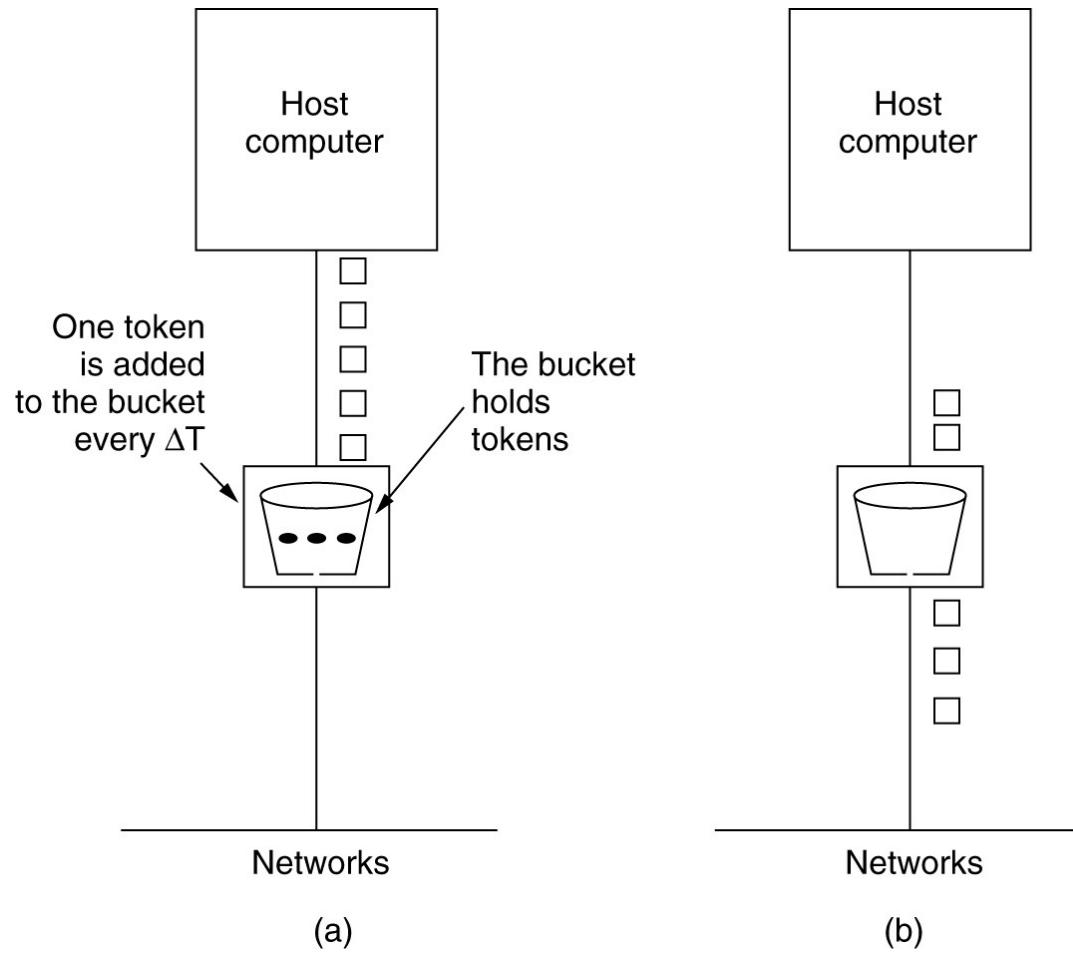
(a)



(b)

(a) A leaky bucket with water. (b) a leaky bucket with packets.

The Token Bucket Algorithm



(a) Before. (b) After.

Burst size calculation in token bucket

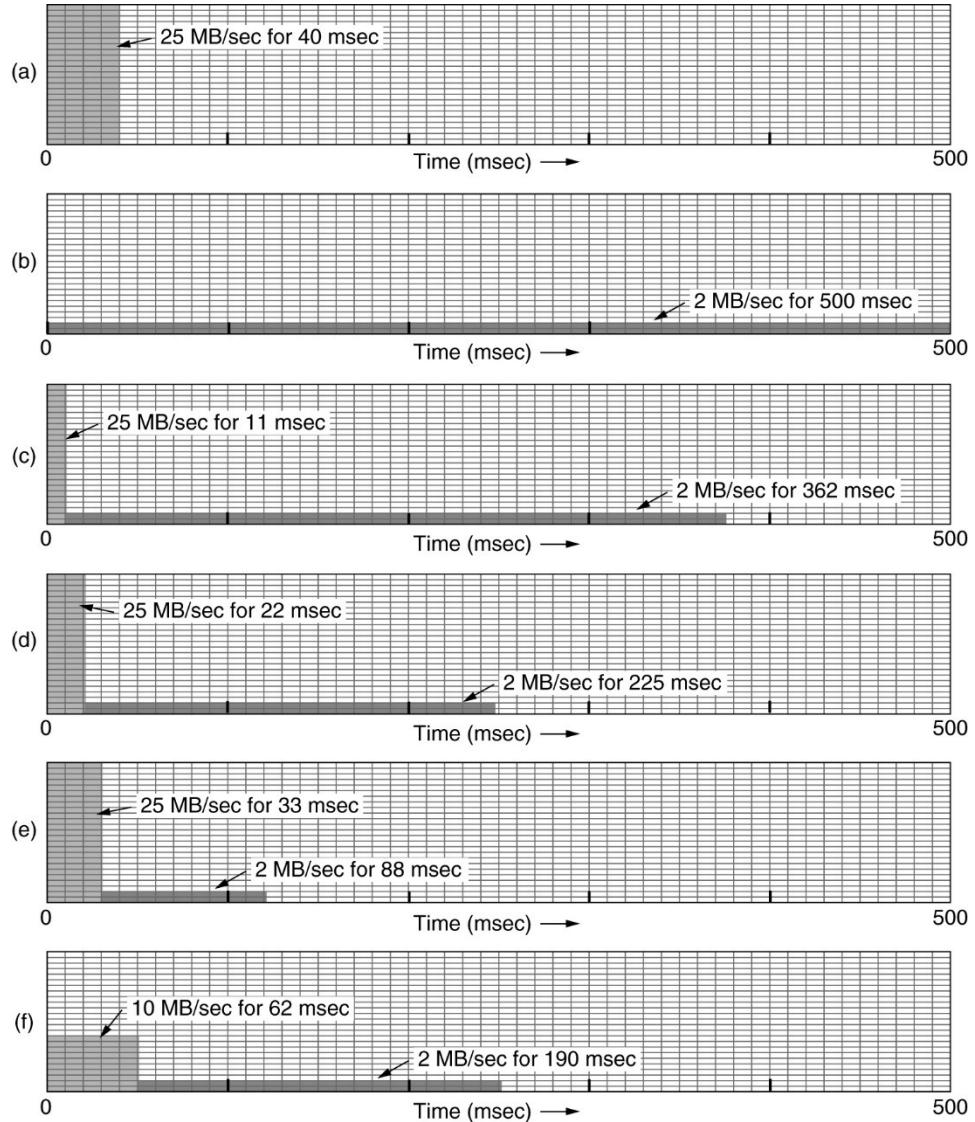
Calculating the length of the maximum burst (until the bucket empties) is slightly tricky. It is longer than just 9600 KB divided by 125 MB/sec because while the burst is being output, more tokens arrive. If we call the burst length S sec., the maximum output rate M bytes/sec, the token bucket capacity B bytes, and the token arrival rate R bytes/sec, we can see that an output burst contains a maximum of $B + RS$ bytes. We also know that the number of bytes in a maximum-speed burst of length S seconds is MS . Hence, we have

$$B + RS = MS$$

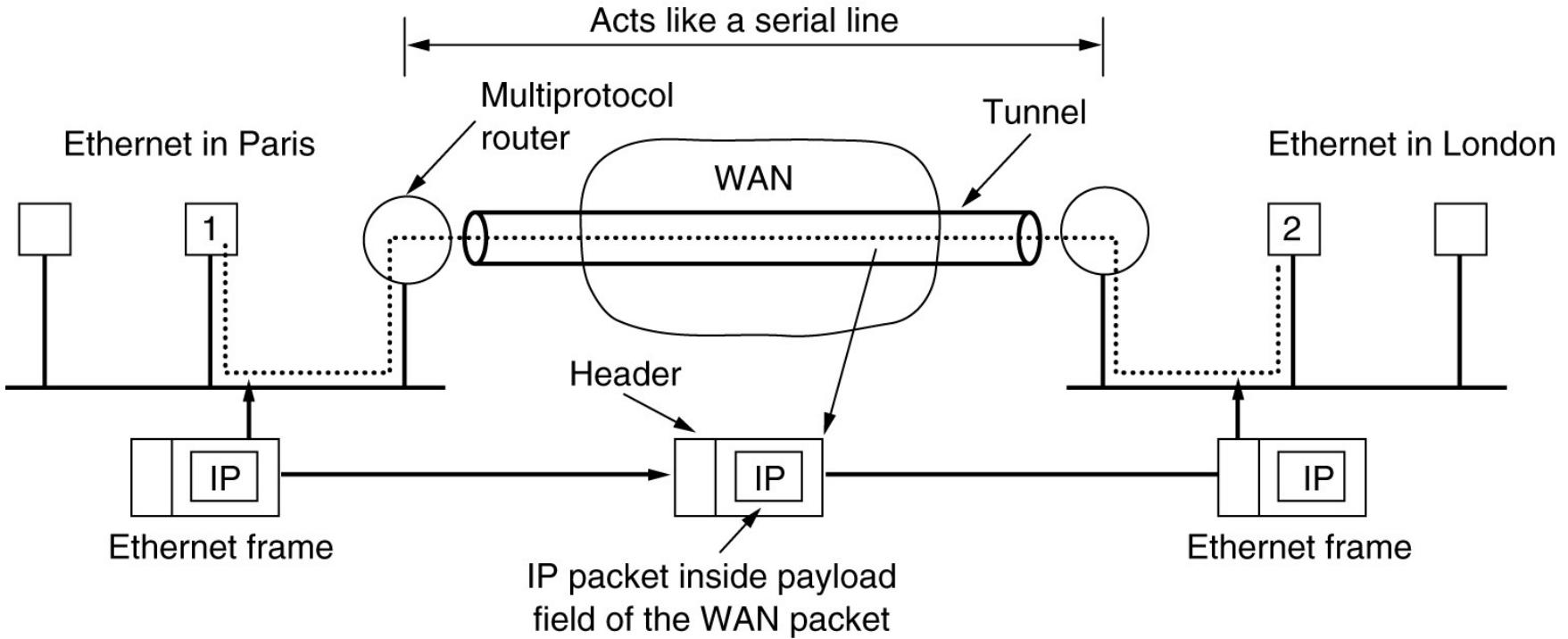
We can solve this equation to get $S = B/(M - R)$. For our parameters of $B = 9600$ KB, $M = 125$ MB/sec, and $R = 25$ MB/sec, we get a burst time of about 94 msec.

The Leaky Bucket Algorithm

- (a) Input to a leaky bucket.
- (b) Output from a leaky bucket. Output from a token bucket with capacities of (c) 250 KB, (d) 500 KB, (e) 750 KB, (f) Output from a 500KB token bucket feeding a 10-MB/sec leaky bucket.

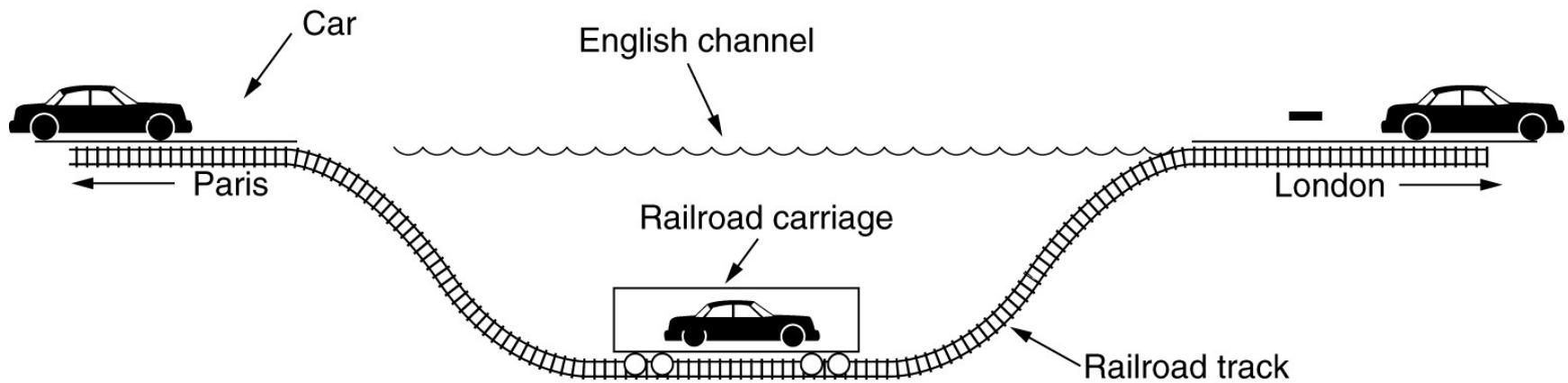


Tunneling



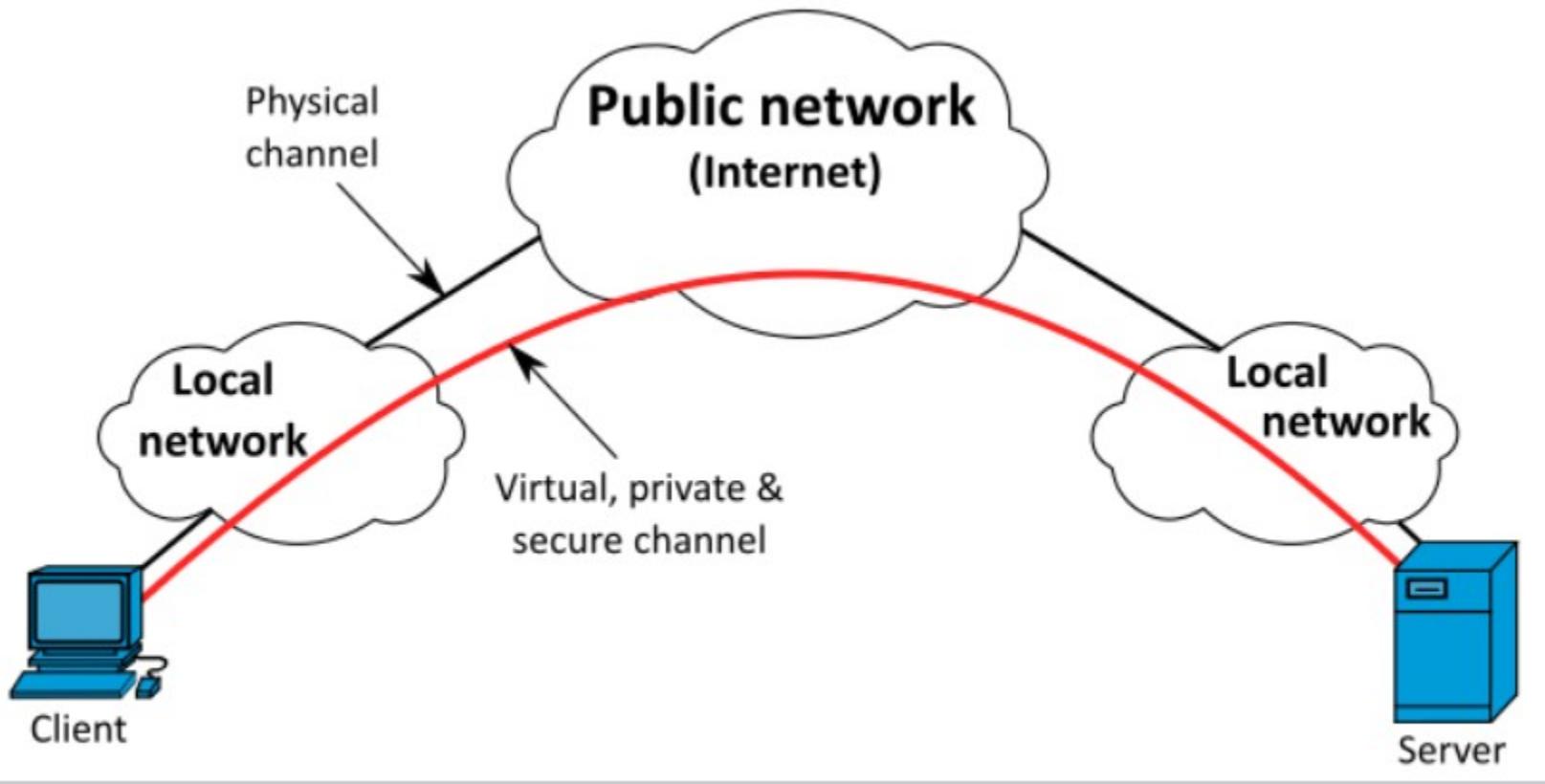
Tunneling a packet from Paris to London.

Tunneling (2)



Tunneling a car from France to England.

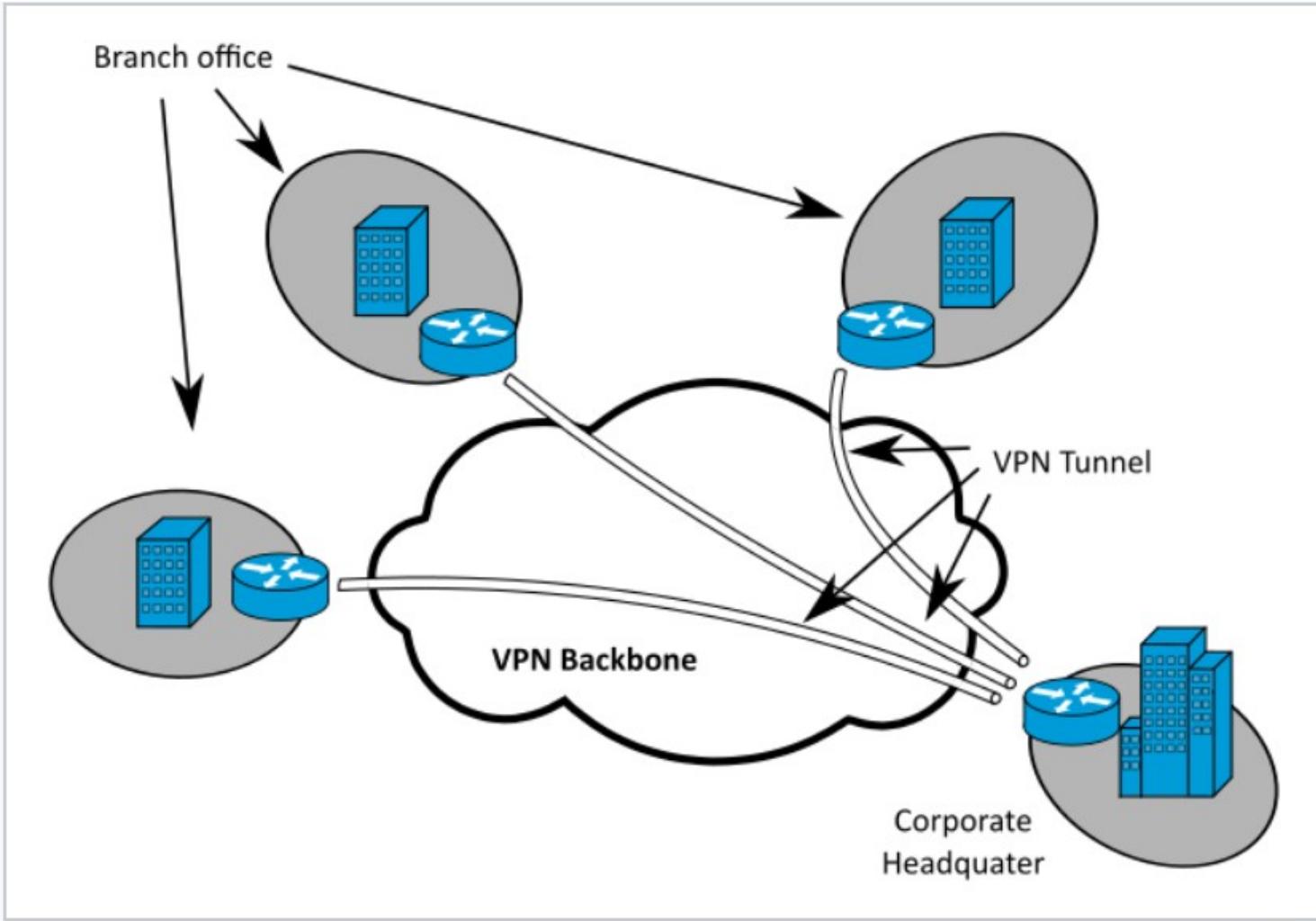
VPN connectivity



VPN connectivity overview



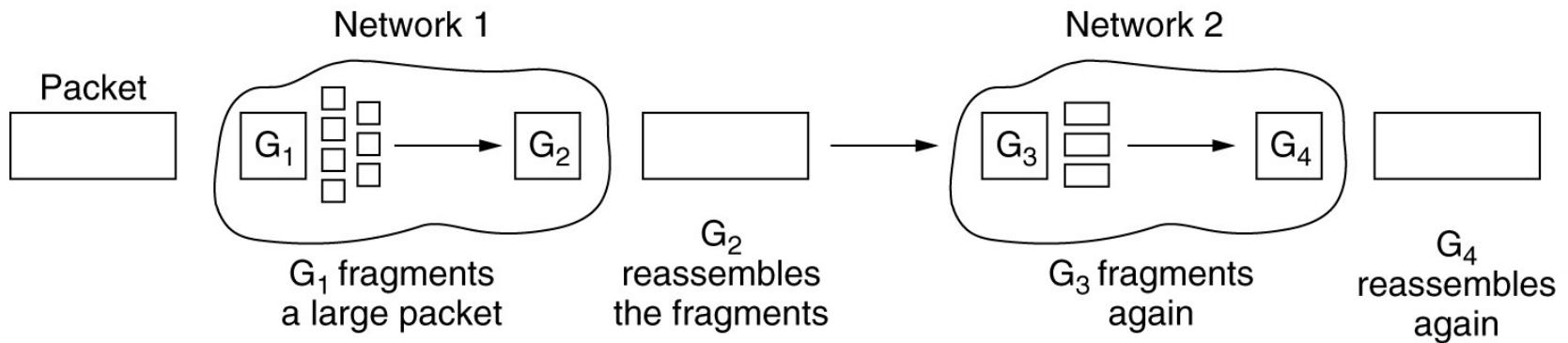
Site-to-Site VPN



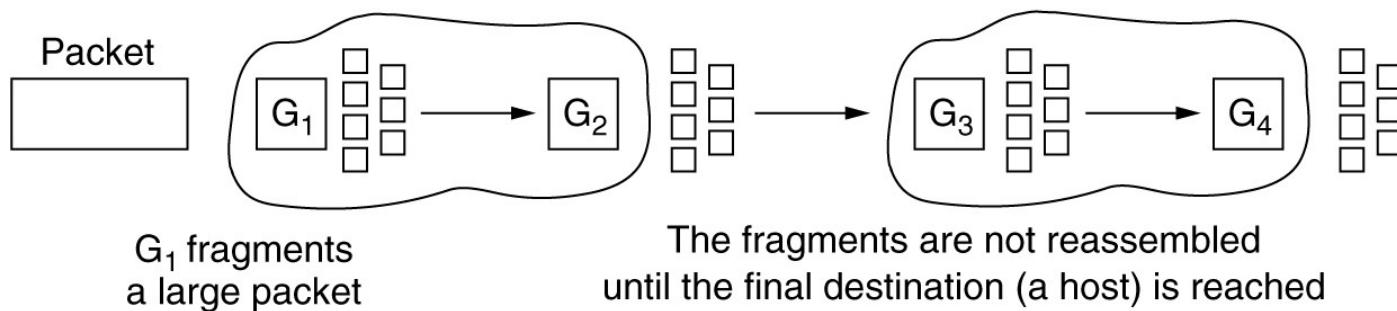
A typical site-to-site VPN.



Fragmentation



(a)

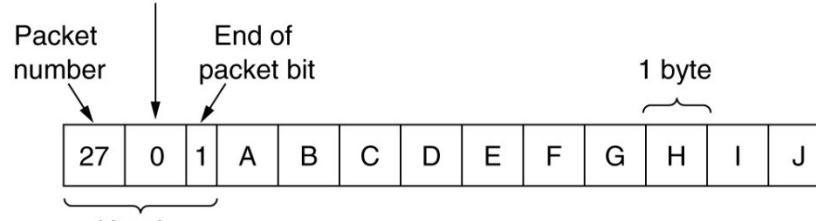


(b)

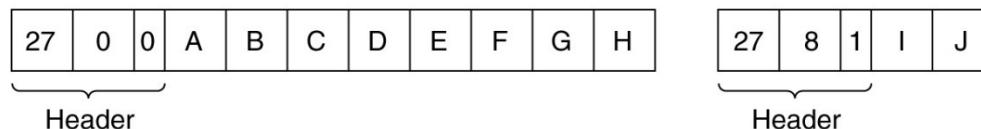
(a) Transparent fragmentation. (b) Nontransparent fragmentation.

Fragmentation (2)

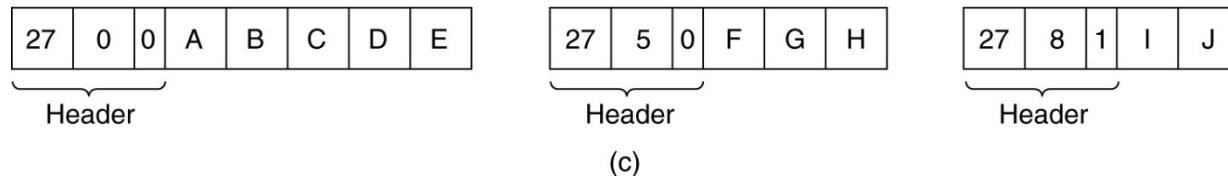
Number of the first elementary fragment in this packet



(a)



(b)



(c)

Fragmentation when the elementary data size is 1 byte.

- (a) Original packet, containing 10 data bytes.
- (b) Fragments after passing through a network with maximum packet size of 8 payload bytes plus header.
- (c) Fragments after passing through a size 5 gateway.

Internet Control Message Protocol

- Internet Control Message Protocol (ICMP) provides feedback about issues related to the processing of IP packets under certain conditions.
- ICMPv4 is the messaging protocol for IPv4 (supporting it).
- The ICMP messages common to both ICMPv4 and ICMPv6 include:
 - Host reachability
 - Destination or Service Unreachable
 - Time exceeded

ICMP message types

Ping 

Message type	Description
Destination unreachable	Packet could not be delivered
Time exceeded	Time to live field hit 0
Parameter problem	Invalid header field
Source quench	Choke packet
Redirect	Teach a router about geography
Echo request	Ask a machine if it is alive
Echo reply	Yes, I am alive
Timestamp request	Same as Echo request, but with timestamp
Timestamp reply	Same as Echo reply, but with timestamp

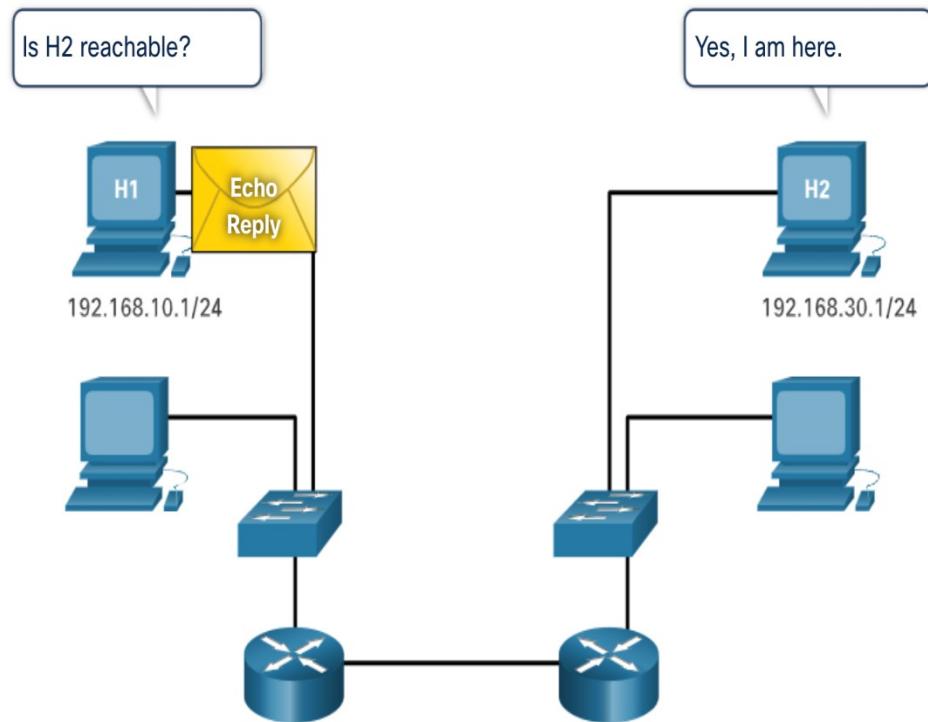
The principal ICMP message types.

Host Reachability

ICMP Echo Message can be used to test the reachability of a host on an IP network.

In the example:

- The local host sends an ICMP Echo Request to a host.
- If the host is available, the destination host responds with an Echo Reply.



Destination or Service Unreachable

- An ICMP Destination Unreachable message can be used to notify the source that a destination or service is unreachable.
- The ICMP message will include a code indicating why the packet could not be delivered.

A few Destination Unreachable codes for ICMPv4 are as follows:

- 0 - Net unreachable
- 1 - Host unreachable
- 2 - Protocol unreachable
- 3 - Port unreachable

Time Exceeded

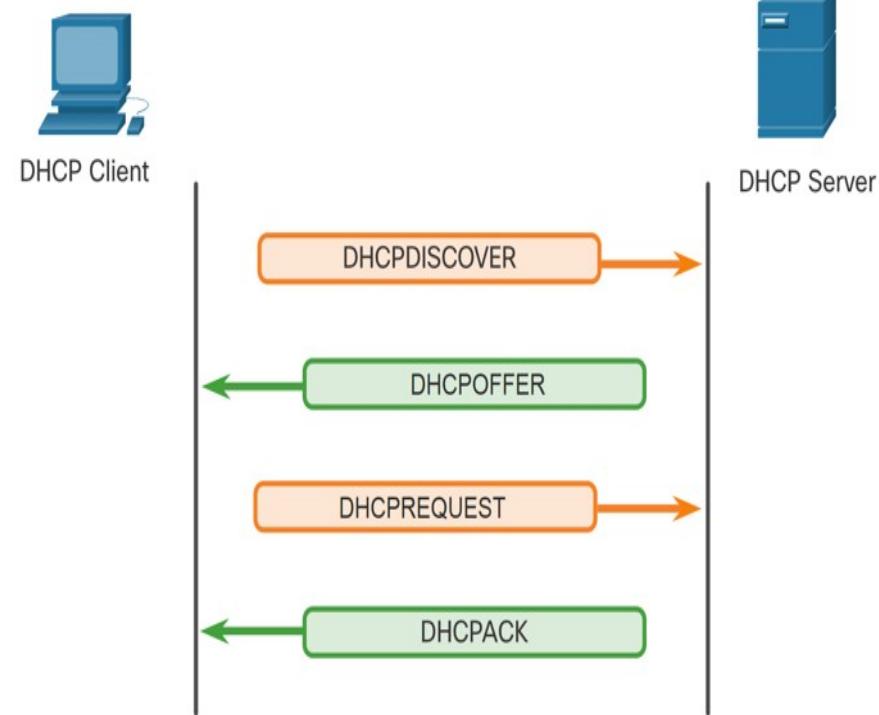
- When the Time to Live (TTL) field in a packet is decremented to 0, an ICMPv4 Time Exceeded message will be sent to the source host.

```
Pinging 8.8.8.8 with 32 bytes of data:  
Reply from 192.168.1.1: TTL expired in transit.  
  
Ping statistics for 8.8.8.8:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

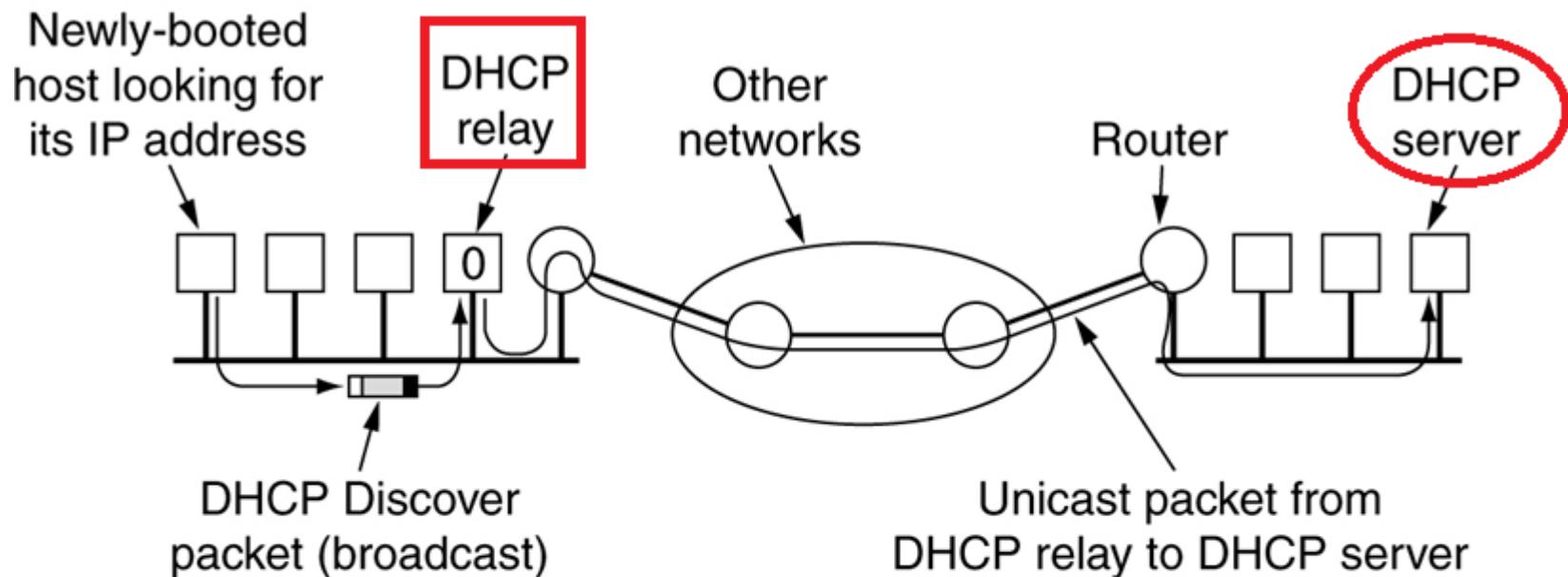
Note: Time Exceeded messages are used by the **traceroute** tool.

DHCP Operation

- When DHCP-configured client boots up, it broadcasts a DHCP discover (DHCPDISCOVER) message to identify any available DHCP servers on the network.
- A DHCP server replies with a DHCP offer (DHCPOFFER) message, which offers a lease to the client. (If a client receives more than one offer due to **multiple DHCP servers** on the network, it must choose one.)
- The client sends a DHCP request (DHCPREQUEST) message that identifies the explicit server and lease offer that the client is accepting.
- The server then returns a DHCP acknowledgment (DHCPACK) message that acknowledges to the client that the lease has been finalized



DHCP Relay agent



DHCP Relay agent forwards DHCP broadcast msg to DHCP server which is located in another network

DHCP Server responds with the OFFER msg to the DHCP Relay agent which then forwards it to the DHCP client

Chapter 6: Transport Layer

Summary

Part A: Introduction

Part B: Socket

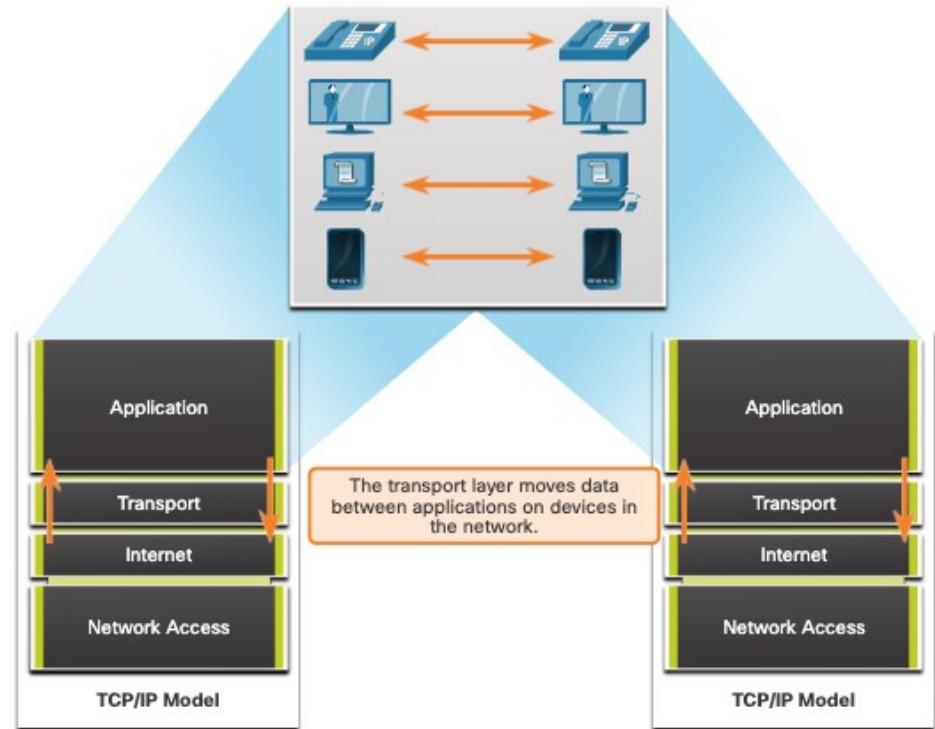
Part C: TCP

Part D: UDP

Introduction: Role of the Transport Layer

The transport layer is:

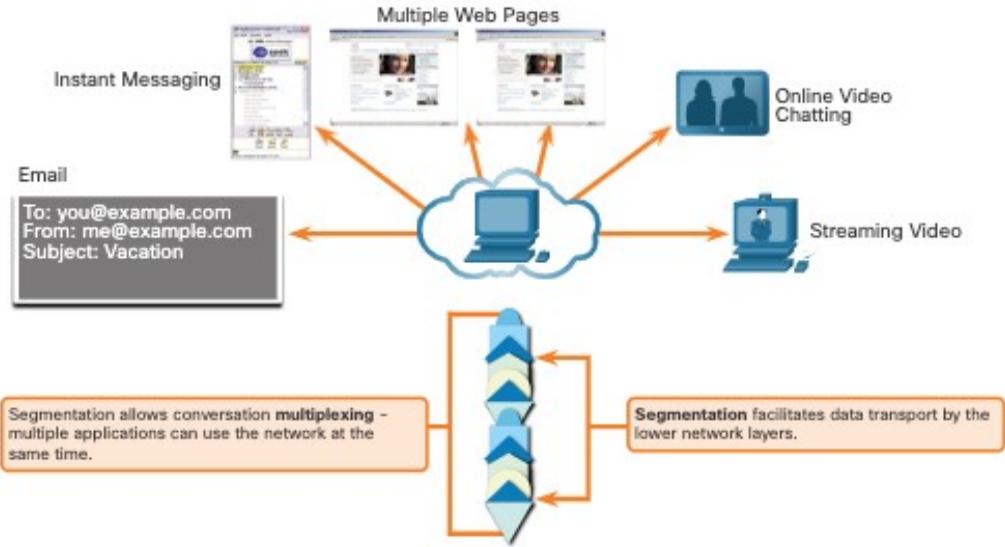
- responsible for logical communications between applications running on different hosts.
- The link between the application layer and the lower layers that are responsible for network transmission.



Transport Layer Responsibilities

The transport layer has the following responsibilities:

- Tracking individual conversations
- Segmenting data and reassembling segments
- Adds header information
- Identify, separate, and manage multiple conversations
- Uses segmentation and multiplexing to enable different communication conversations to be interleaved on the same network

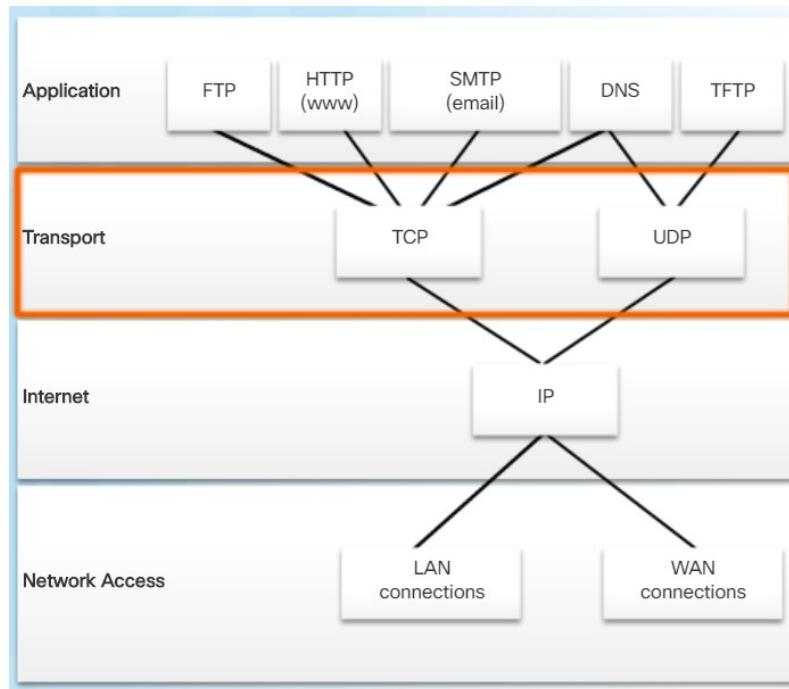


Transport Layer Session Identifier

- Source IP address
- Destination IP address
- Sort Port
- Destination Port

Transport Layer Protocols

- IP does not specify how the delivery or transportation of the packets takes place.
- Transport layer protocols specify how to transfer messages between hosts, and are responsible for managing reliability requirements of a conversation.
- The transport layer includes the **TCP and UDP** protocols.

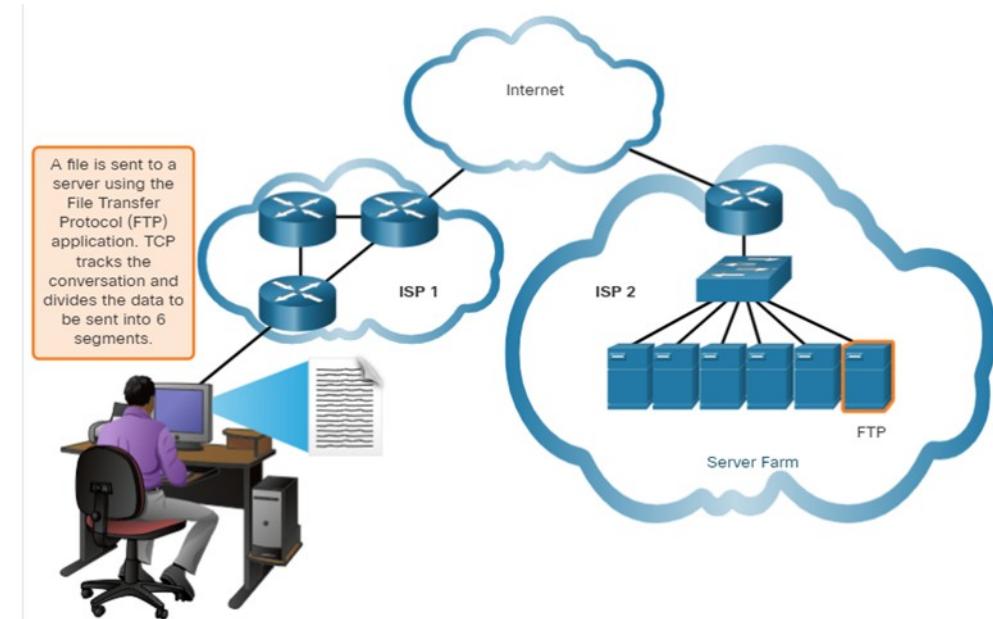


Transmission Control Protocol

TCP provides **reliability and flow control**.

TCP basic operations:

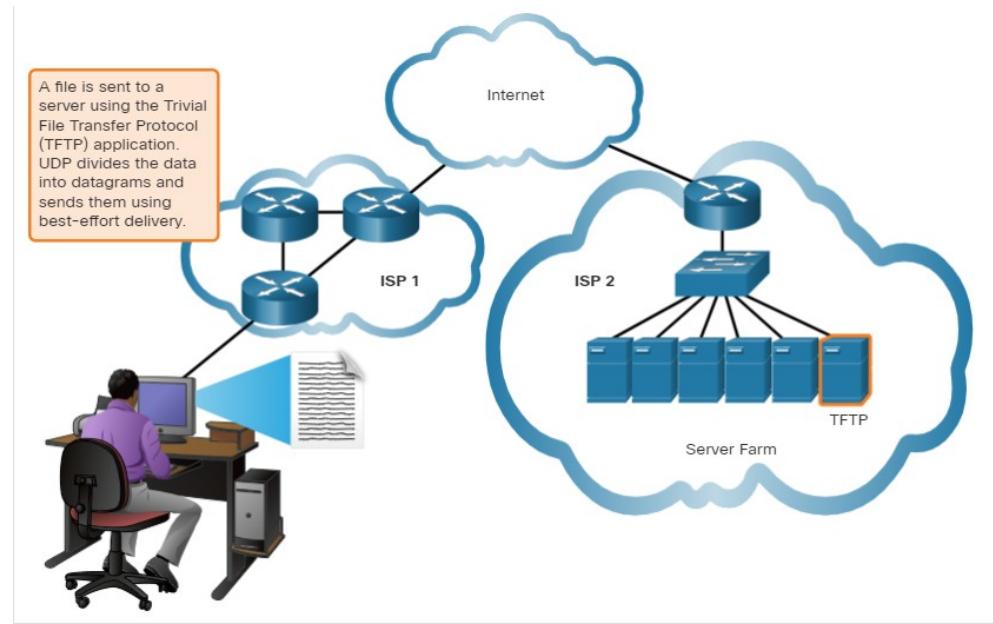
- Number and track data segments transmitted to a specific host from a specific application
- Acknowledge received data
- Retransmit any unacknowledged data after a certain amount of time
- Sequence data that might arrive in wrong order
- Send data at an efficient rate that is acceptable by the receiver



User Datagram Protocol (UDP)

UDP provides the basic functions for delivering datagrams between the appropriate applications, with very little overhead and data checking.

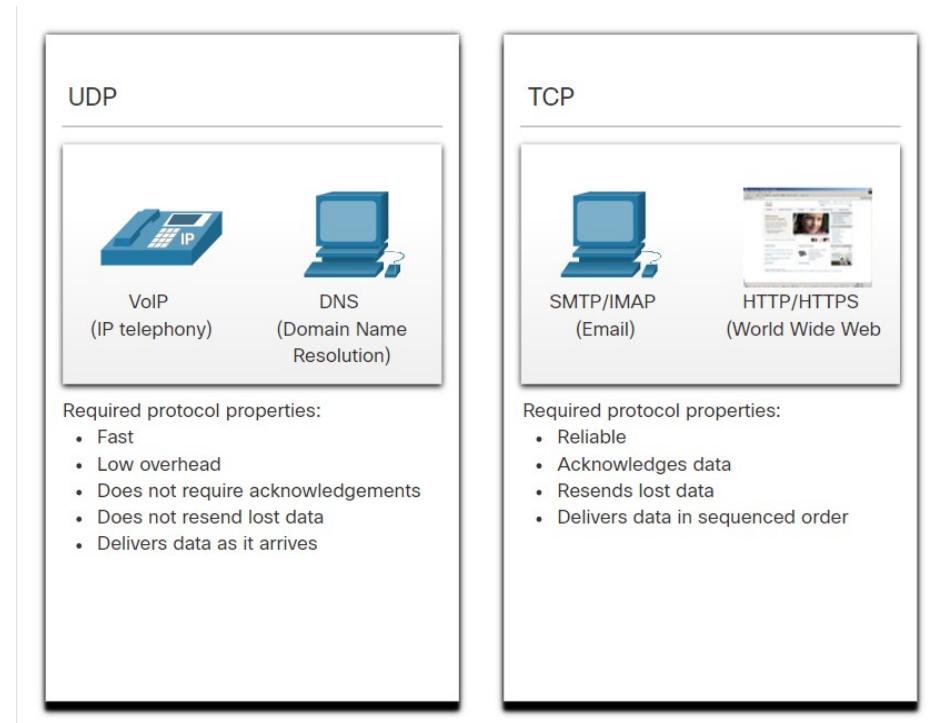
- UDP is a connectionless protocol.
- UDP is known as a best-effort delivery protocol because there is no acknowledgment that the data is received at the destination.



Right Transport Layer Protocol for the Right Application

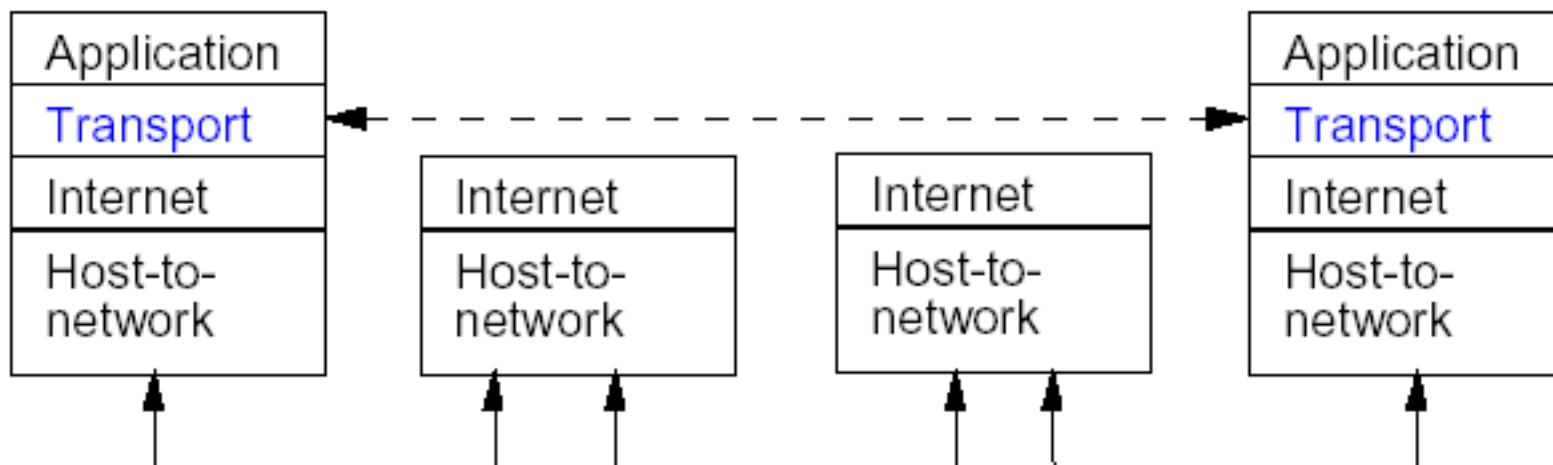
UDP is also used by request-and-reply applications where the data is minimal, and retransmission can be done quickly.

If it is important that all the data arrives and that it can be processed in its proper sequence, TCP is used as the transport protocol.



Part A: Introduction

- The transport layer is an end-to-end layer – this means that nodes within the subnet do not participate in transport layer protocols – only the end hosts.
- As with other layers, transport layer protocols send data as a sequence of packets (**segments**).



Multiplexing (1)

- The **network layer** provides communication between two hosts.
- The **transport layer** provides communication between two **processes** running on different hosts.
- A process is an instance of a program that is running on a host.
- There may be multiple processes communicating between two hosts – for example, there could be a FTP session and a Telnet session between the same two hosts.

Multiplexing (2)

- The transport layer provides a way to multiplex / demultiplex communication between various processes.
- To provide multiplexing, the transport layer adds an address to each segment indicating the source and destination processes.
- Note these addresses need only be unique locally on a given host.
- In TCP/IP these transport layer addresses are called ***port-numbers***.

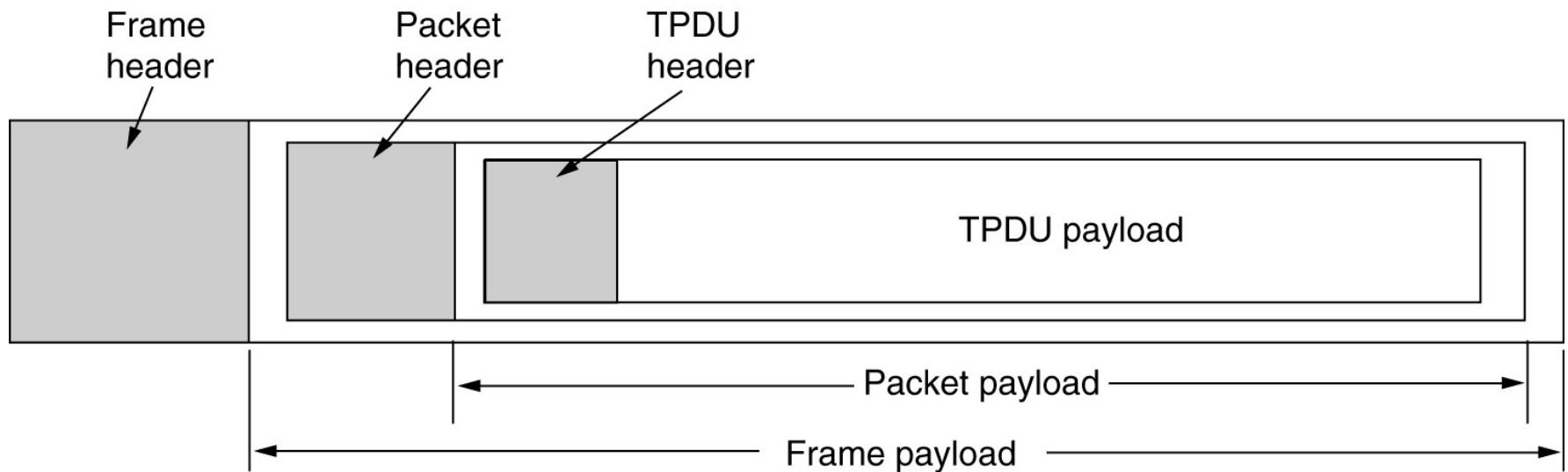
Multiplexing (3)

Session Identifier (5-tuple): uniquely identifies a process-to-process connection in the Internet.

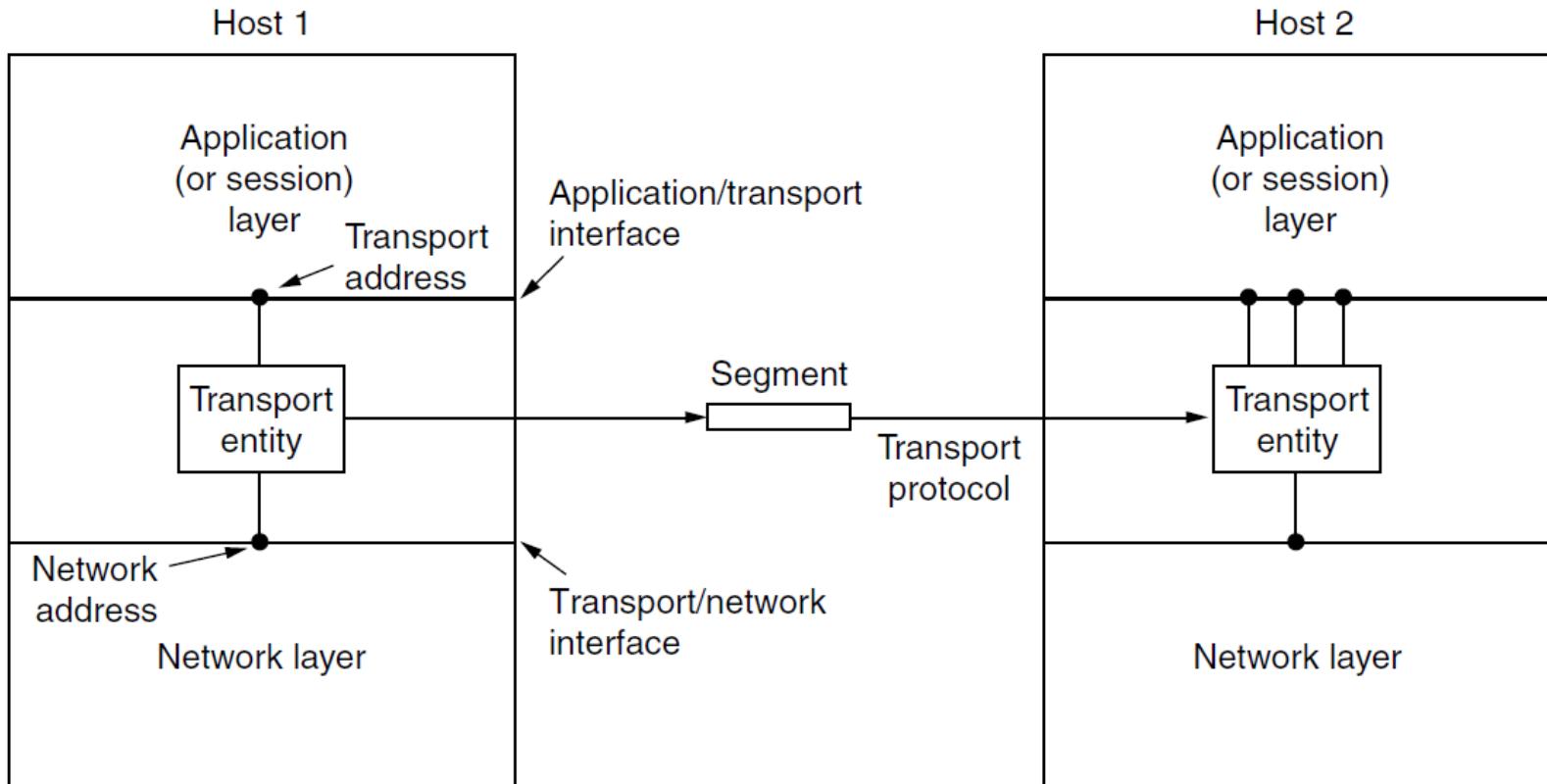
- Sender IP address
- Sender port
- Destination IP address
- Destination port
- Transport layer protocol

TPDU

Nesting of Transport Protocol Data Unit (TPDU),
packets, and frames.



Network, Transport and Application



Transport Service Primitives

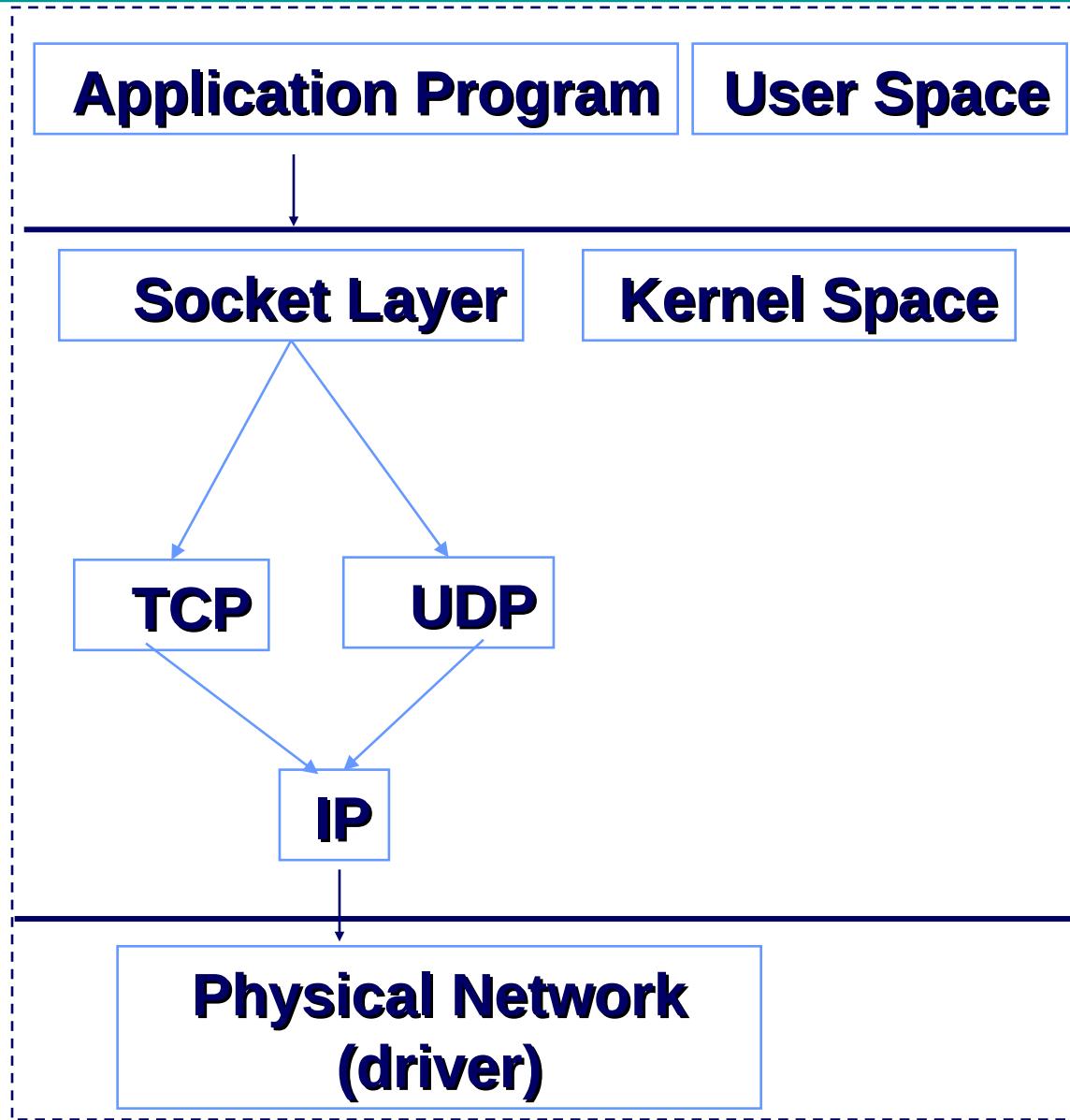
Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

The primitives for a simple transport service.

PART B: SOCKET

- Software interface designed to communicate between the user program and TCP/IP protocol stack
- Implemented by a set of library function calls
- Socket is a data structure inside the program
- Both client and server programs communicate via a pair of sockets

Socket Framework



Socket Families

There are several significant socket domain families:

- ⇒ Internet Domain Sockets (AF_INET)
 - implemented via IP addresses and port numbers
- ⇒ Unix Domain Sockets (AF_UNIX)
 - implemented via filenames (think “named pipe”)
- ⇒ Novell IPX (AF_IPX)
- ⇒ AppleTalk DDS (AF_APPLETALK)

Type of Socket

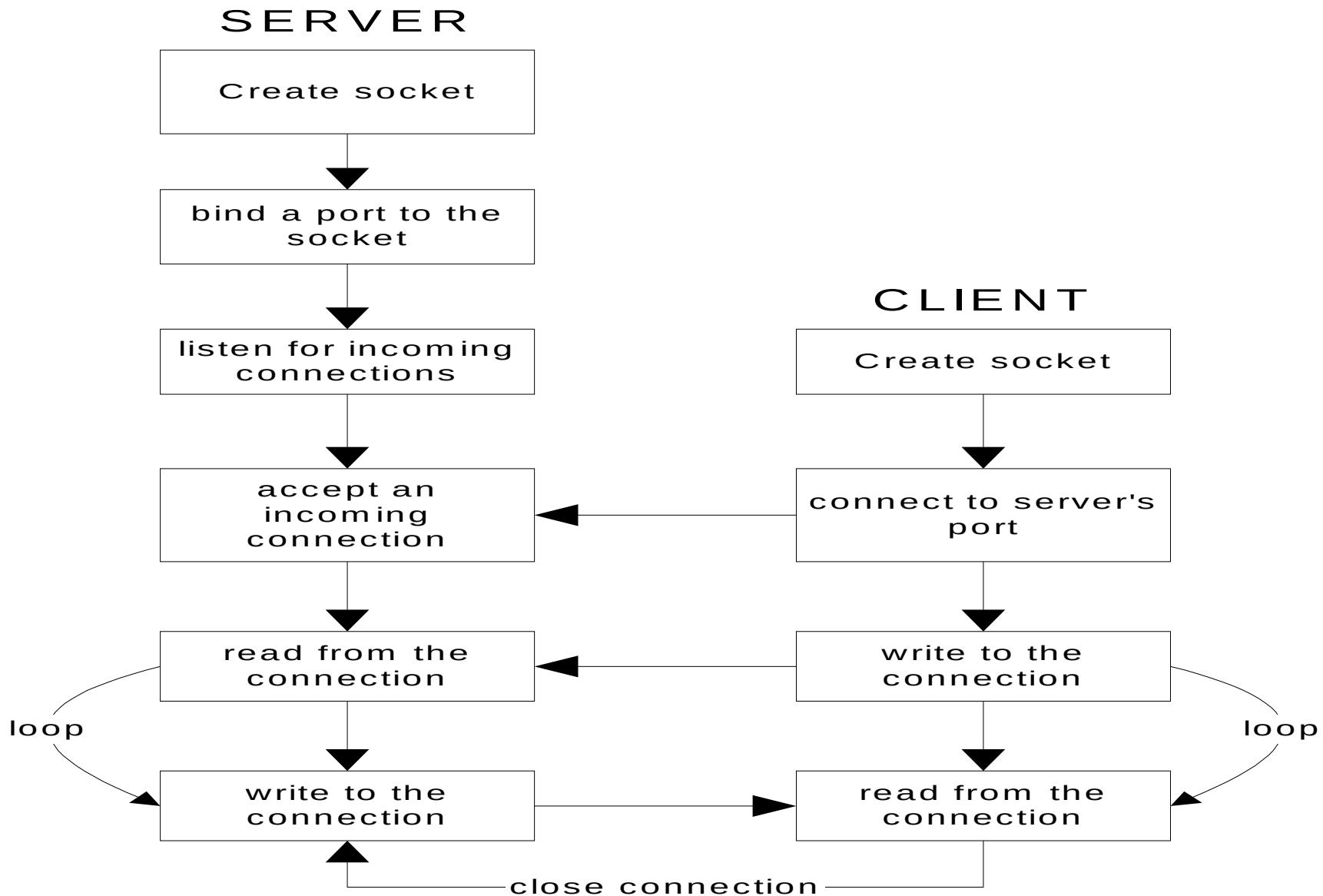
- Stream (SOCK_STREAM) Uses TCP protocol. Connection-oriented service
- Datagram (SOCK_DGRAM) Uses UDP protocol. Connectionless service
- Raw (SOCK_RAW) Used for testing

Creating a Socket

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int domain, int type, int protocol);
```

- domain is one of the *Protocol Families* (PF_INET, PF_UNIX, etc.)
- type defines the communication protocol semantics, usually defines either:
 - SOCK_STREAM: connection-oriented stream (TCP)
 - SOCK_DGRAM: connectionless, unreliable (UDP)
- protocol specifies a particular protocol, just set this to 0 to accept the default

TCP Client/Server



Socket Primitives for TCP

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

TCP Server

Sequence of calls

sock_init() Create the socket

bind() Register port with the system

listen() Establish client connection

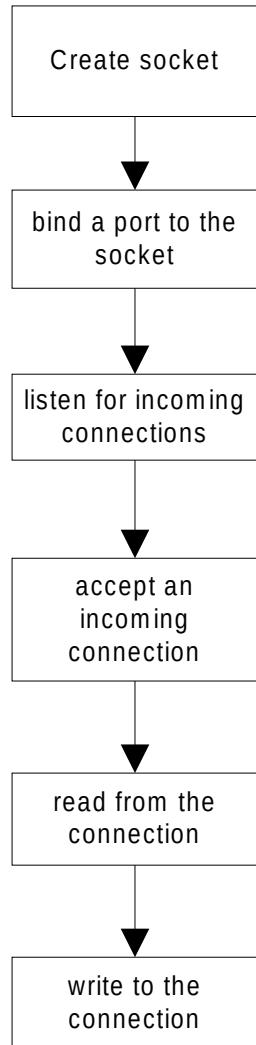
accept() Accept client connection

read/write read/write data

close() shutdown

Server Side Socket Details (Functions)

SERVER



```
int socket(int domain, int type, int protocol)  
sockfd = socket(PF_INET, SOCK_STREAM, 0);
```

```
int bind(int sockfd, struct sockaddr *server_addr, socklen_t length)  
bind(sockfd, &server, sizeof(server));
```

```
int listen( int sockfd, int num_queued_requests)  
listen( sockfd, 5);
```

```
int accept(int sockfd, struct sockaddr *incoming_address, socklen_t length)  
newfd = accept(sockfd, &client, sizeof(client)); /* BLOCKS */
```

```
int read(int sockfd, void * buffer, size_t buffer_size)  
read(newfd, buffer, sizeof(buffer));
```

```
int write(int sockfd, void * buffer, size_t buffer_size)  
write(newfd, buffer, sizeof(buffer));
```

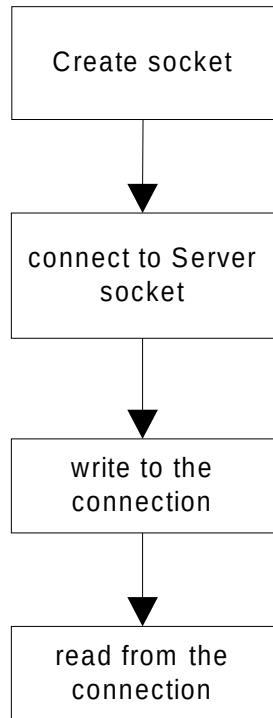
TCP Client

Sequence of calls

sock_init ()	Create socket
connect()	Set up connection
write/read	write/read data
close()	Shutdown

Client Side Socket Details

CLIENT



```
int socket(int domain, int type, int protocol)  
sockfd = socket(PF_INET, SOCK_STREAM, 0);
```

```
int connect(int sockfd, struct sockaddr *server_address, socklen_t length)  
connect(sockfd, &server, sizeof(server));
```

```
int write(int sockfd, void * buffer, size_t buffer_size)  
write(sockfd, buffer, sizeof(buffer));
```

```
int read(int sockfd, void * buffer, size_t buffer_size)  
read(sockfd, buffer, sizeof(buffer));
```

UDP Clients and Servers

Connectionless clients and servers create a socket using **SOCK_DGRAM** instead of **SOCK_STREAM**

Connectionless servers do not call `listen()` or `accept()`, and *usually* do not call `connect()`

Since connectionless communications lack a **sustained connection**, several methods are available that allow you to *specify a destination address with every call*:

- `sendto(sock, buffer, buflen, flags, to_addr, tolen);`
- `recvfrom(sock, buffer, buflen, flags, from_addr, fromlen);`

UDP Server

Sequence of calls

sock_init()	Create socket
bind()	Register port with the system
recfrom/sendto	Receive/send data
close()	Shutdown

UDP Client

Sequence of calls

socket_init()	Create socket
sendto/recfrom	send/receive data
close()	Shutdown

Socket Programming Example: Internet File Server

```
/* This page contains a client program that can request a file from the server program
 * on the next page. The server responds by sending the whole file.
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define SERVER_PORT 12345          /* arbitrary, but client & server must agree */
#define BUF_SIZE 4096             /* block transfer size */

int main(int argc, char **argv)
{
    int c, s, bytes;
    char buf[BUF_SIZE];           /* buffer for incoming file */
    struct hostent *h;            /* info about server */
    struct sockaddr_in channel;   /* holds IP address */

    if (argc != 3) fatal("Usage: client server-name file-name");
    h = gethostbyname(argv[1]);    /* look up host's IP address */
    if (!h) fatal("gethostbyname failed");

    s = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (s < 0) fatal("socket");
    memset(&channel, 0, sizeof(channel));
    channel.sin_family= AF_INET;
    memcpy(&channel.sin_addr.s_addr, h->h_addr, h->h_length);
    channel.sin_port= htons(SERVER_PORT);

    c = connect(s, (struct sockaddr *) &channel, sizeof(channel));
    if (c < 0) fatal("connect failed");

    /* Connection is now established. Send file name including 0 byte at end. */
    write(s, argv[2], strlen(argv[2])+1);

    /* Go get the file and write it to standard output. */
    while (1) {
        bytes = read(s, buf, BUF_SIZE);      /* read from socket */
        if (bytes <= 0) exit(0);            /* check for end of file */
        write(1, buf, bytes);              /* write to standard output */
    }
}

fatal(char *string)
{
    printf("%s\n", string);
    exit(1);
}
```

Client code using
sockets.

Socket Programming Example: Internet File Server (2)

Server code using sockets.

```
#include <sys/types.h>           /* This is the server code */
#include <sys/fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#define SERVER_PORT 12345          /* arbitrary, but client & server must agree */
#define BUF_SIZE 4096              /* block transfer size */
#define QUEUE_SIZE 10
int main(int argc, char *argv[])
{
    int s, b, l, fd, sa, bytes, on = 1;
    char buf[BUF_SIZE];           /* buffer for outgoing file */
    struct sockaddr_in channel;   /* hold's IP address */
    /* Build address structure to bind to socket. */
    memset(&channel, 0, sizeof(channel)); /* zero channel */
    channel.sin_family = AF_INET;
    channel.sin_addr.s_addr = htonl(INADDR_ANY);
    channel.sin_port = htons(SERVER_PORT);

    /* Passive open. Wait for connection. */
    s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP); /* create socket */
    if (s < 0) fatal("socket failed");
    setsockopt(s, SOL_SOCKET, SO_REUSEADDR, (char *) &on, sizeof(on));

    b = bind(s, (struct sockaddr *) &channel, sizeof(channel));
    if (b < 0) fatal("bind failed");

    l = listen(s, QUEUE_SIZE);        /* specify queue size */
    if (l < 0) fatal("listen failed");

    /* Socket is now set up and bound. Wait for connection and process it. */
    while (1) {
        sa = accept(s, 0, 0);         /* block for connection request */
        if (sa < 0) fatal("accept failed");
        read(sa, buf, BUF_SIZE);      /* read file name from socket */
        /* Get and return the file. */
        fd = open(buf, O_RDONLY);     /* open the file to be sent back */
        if (fd < 0) fatal("open failed");
        while (1) {
            bytes = read(fd, buf, BUF_SIZE); /* read from file */
            if (bytes <= 0) break;          /* check for end of file */
            write(sa, buf, bytes);        /* write bytes to socket */
        }
        close(fd);                   /* close file */
        close(sa);                   /* close connection */
    }
}
```

PART C: TCP

Introduction

- **connection oriented.**
- **full duplex.**

TCP Services:

- **Reliable transport**
- **Flow control**
- **Congestion control**

UDP does not provide any of these services

TCP Features

Establishes a Session - TCP is a connection-oriented protocol that negotiates and establishes a permanent connection (or session) between source and destination devices prior to forwarding any traffic.

Ensures Reliable Delivery - For many reasons, it is possible for a segment to become corrupted or lost completely, as it is transmitted over the network. TCP ensures that each segment that is sent by the source arrives at the destination.

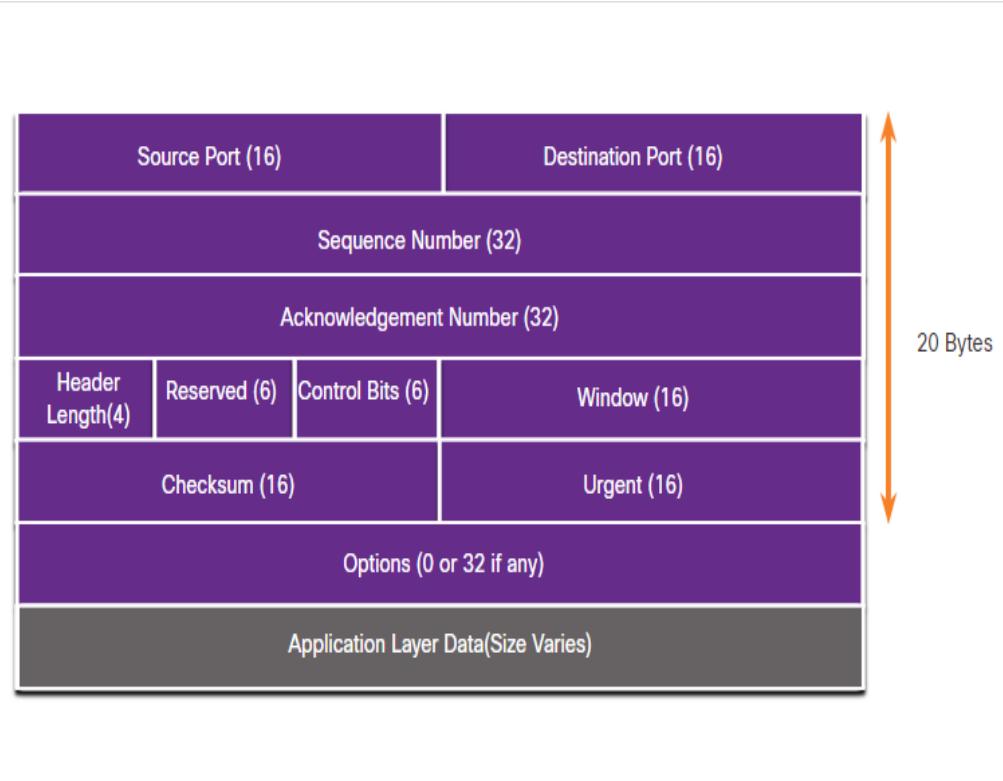
Provides Same-Order Delivery - Because networks may provide multiple routes that can have different transmission rates, data can arrive in the wrong order.

Supports Flow Control - Network hosts have limited resources (i.e., memory and processing power). When TCP is aware that these resources are overtaxed, it can request that the sending application reduce the rate of data flow.

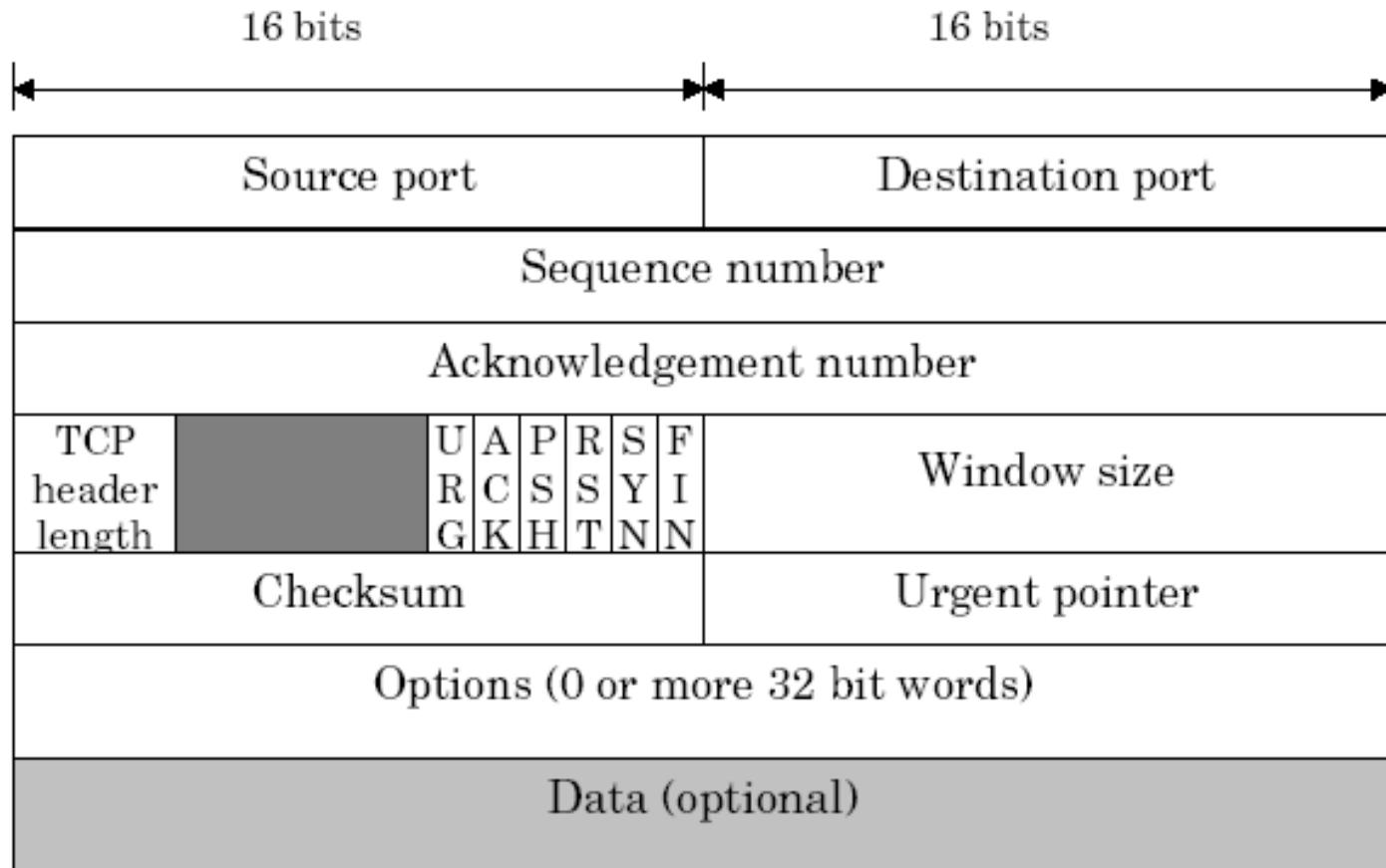
TCP Header

TCP is a stateful protocol which means it keeps track of the state of the communication session.

TCP records which information it has sent, and which information has been acknowledged.



TCP Header



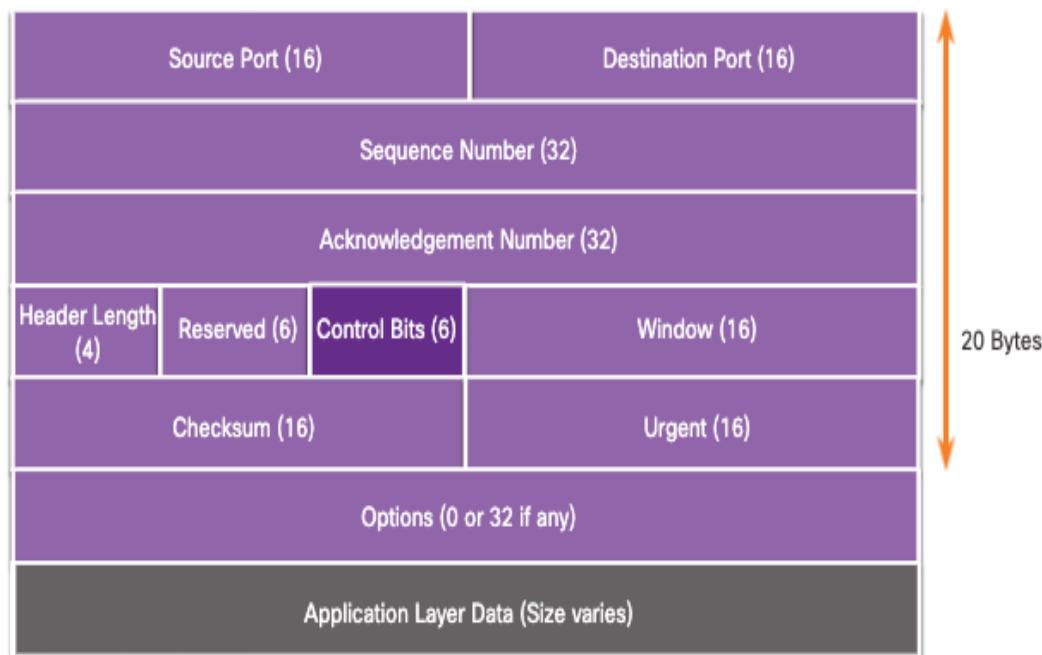
TCP Header Fields

TCP Header Field	Description
Source Port	A 16-bit field used to identify the source application by port number.
Destination Port	A 16-bit field used to identify the destination application by port number.
Sequence Number	A 32-bit field used for data reassembly purposes.
Acknowledgment Number	A 32-bit field used to indicate that data has been received and the next byte expected from the source.
Header Length	A 4-bit field known as "data offset" that indicates the length of the TCP segment header.
Reserved	A 6-bit field that is reserved for future use.
Control bits	A 6-bit field used that includes bit codes, or flags, which indicate the purpose and function of the TCP segment.
Window size	A 16-bit field used to indicate the number of bytes that can be accepted at one time.
Checksum	A 16-bit field used for error checking of the segment header and data.
Urgent	A 16-bit field used to indicate if the contained data is urgent.

TCP Flags

The six control bit flags are as follows:

- **URG** - Urgent pointer field significant
- **ACK** - Acknowledgment flag used in connection establishment and session termination
- **PSH** - Push function
- **RST** - Reset the connection when an error or timeout occurs
- **SYN** - Synchronize sequence numbers used in connection establishment
- **FIN** - No more data from sender and used in session termination



Sample TCP Packet

Sample TCP Packet

5416	25
4162801	
268124	
6	0
8192	0
0x8C4F	0
timestamp: 0xFF736681	
stand on guard for thee	

Port Numbers

Multiple Separate Communications

TCP and UDP transport layer protocols use port numbers to manage multiple, simultaneous conversations.

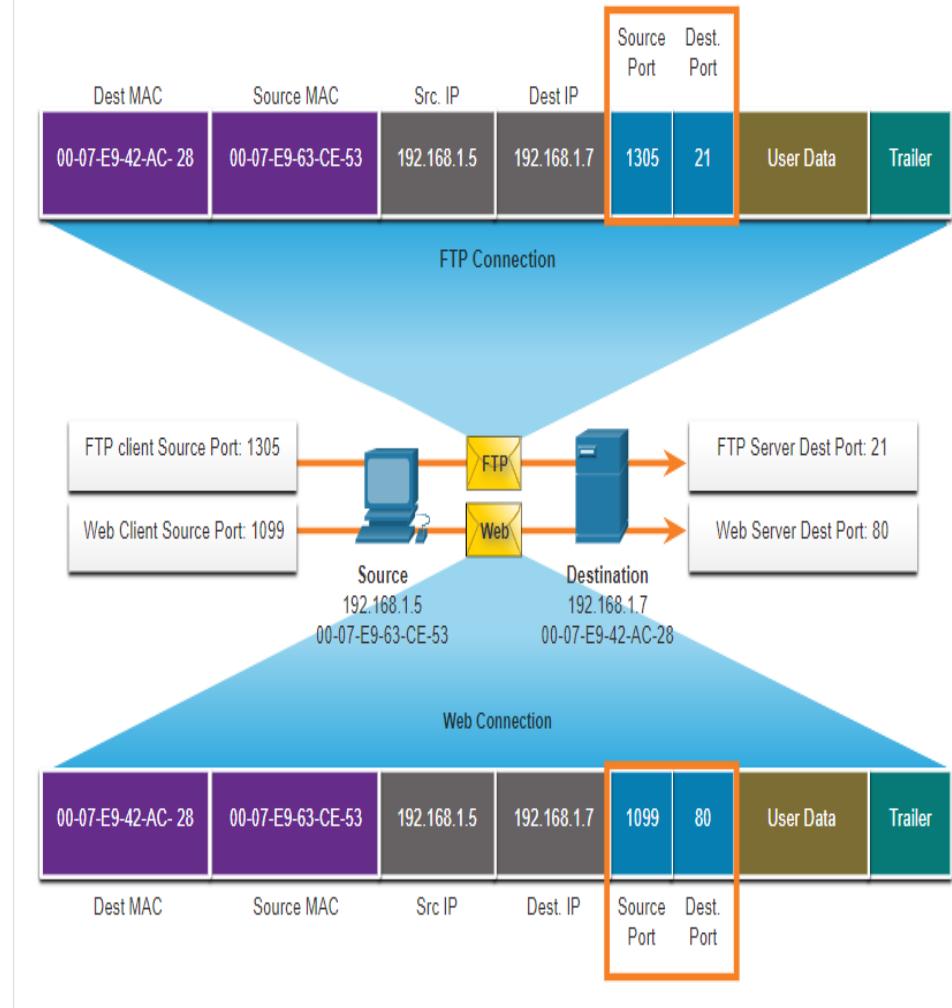
The source port number is associated with the originating application on the local host whereas the destination port number is associated with the destination application on the remote host.

Source Port (16)

Destination Port (16)

Socket Pairs

- The source and destination ports are placed within the segment.
- The segments are then encapsulated within an IP packet.
- The combination of the source IP address and source port number, or the destination IP address and destination port number is known as a socket.
- Sockets enable multiple processes, running on a client, to distinguish themselves from each other, and multiple connections to a server process to be distinguished from each other.



Port Number Groups

Port Group	Number Range	Description
Well-known Ports	0 to 1,023	<ul style="list-style-type: none">These port numbers are reserved for common or popular services and applications such as web browsers, email clients, and remote access clients.Defined well-known ports for common server applications enables clients to easily identify the associated service required.
Registered Ports	1,024 to 49,151	<ul style="list-style-type: none">These port numbers are assigned by IANA to a requesting entity to use with specific processes or applications.These processes are primarily individual applications that a user has chosen to install, rather than common applications that would receive a well-known port number.For example, Cisco has registered port 1812 for its RADIUS server authentication process.
Private and / or Dynamic Ports	49,152 to 65,535	<ul style="list-style-type: none">These ports are also known as <i>ephemeral ports</i>.The client's OS usually assign port numbers dynamically when a connection to a service is initiated.The dynamic port is then used to identify the client application during communication.

Well-Known Port Numbers

Port #	Protocol	Application
20	TCP	File Transfer Protocol (FTP) - Data
21	TCP	File Transfer Protocol (FTP) - Control
22	TCP	Secure Shell (SSH)
23	TCP	Telnet
25	TCP	Simple Mail Transfer Protocol (SMTP)
53	UDP, TCP	Domain Name Service (DNS)
67	UDP	Dynamic Host Configuration Protocol (DHCP) - Server
68	UDP	Dynamic Host Configuration Protocol - Client
69	UDP	Trivial File Transfer Protocol (TFTP)
80	TCP	Hypertext Transfer Protocol (HTTP)
110	TCP	Post Office Protocol version 3 (POP3)
143	TCP	Internet Message Access Protocol (IMAP)
161	UDP	Simple Network Management Protocol (SNMP)
443	TCP	Hypertext Transfer Protocol Secure (HTTPS)

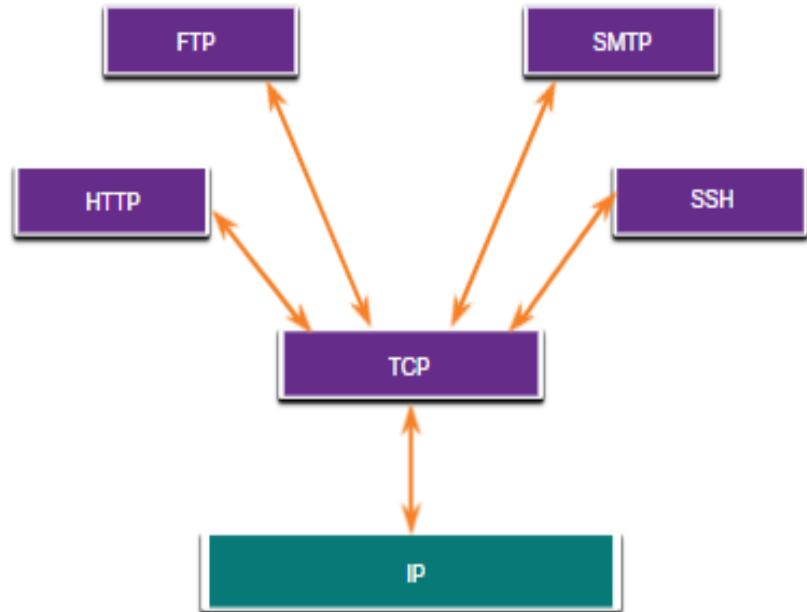
The netstat Command

Unexplained TCP connections can pose a major security threat. Netstat is an important tool to verify connections.

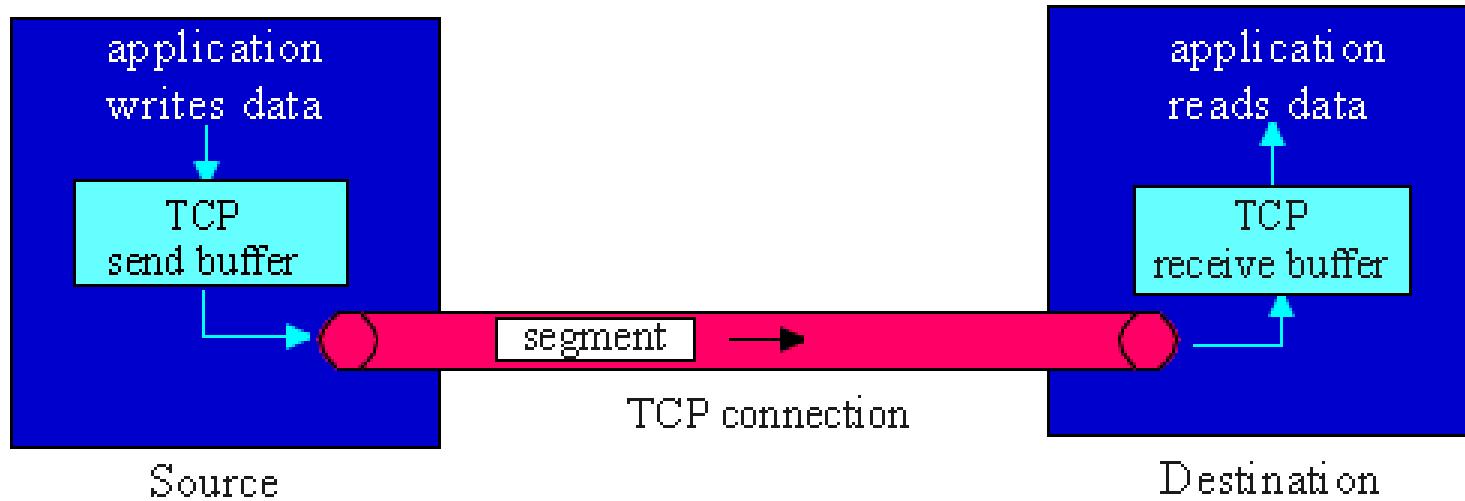
```
C:\> netstat
Active Connections
Proto Local Address          Foreign Address
State
TCP   192.168.1.124:3126    192.168.0.2:netbios-ssn
ESTABLISHED
TCP   192.168.1.124:3158    207.138.126.152:http
ESTABLISHED
TCP   192.168.1.124:3159    207.138.126.169:http
ESTABLISHED
TCP   192.168.1.124:3160    207.138.126.169:http
ESTABLISHED
TCP   192.168.1.124:3161    sc.msn.com:http
ESTABLISHED
TCP   192.168.1.124:3166    www.cisco.com:http
ESTABLISHED
```

Applications that use TCP

TCP handles all tasks associated with dividing the data stream into segments, providing reliability, controlling data flow, and reordering segments.



Flow of TCP Segments



TCP Services (1)

TCP converts the **unreliable, best effort service of IP** into a **reliable service**, i.e., it ensures that each segment is delivered correctly, only once, and in order.

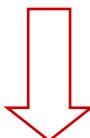
Converting an unreliable connection into a reliable connection is basically the same problem we have considered at the data link layer, and essentially the same solution is used:

- TCP numbers each segment and uses an ARQ protocol to recover lost segments.
- Some versions of TCP implement **Go Back N** and other versions implement **Selective Repeat**.

TCP Services (2)

However, there are a few important differences between the transport layer and the data link layer.

- At the data link layer, we viewed an ARQ protocol as being operated between two nodes connected by a point-to-point link.
- At the transport layer, this protocol is implemented between two hosts connected over network.

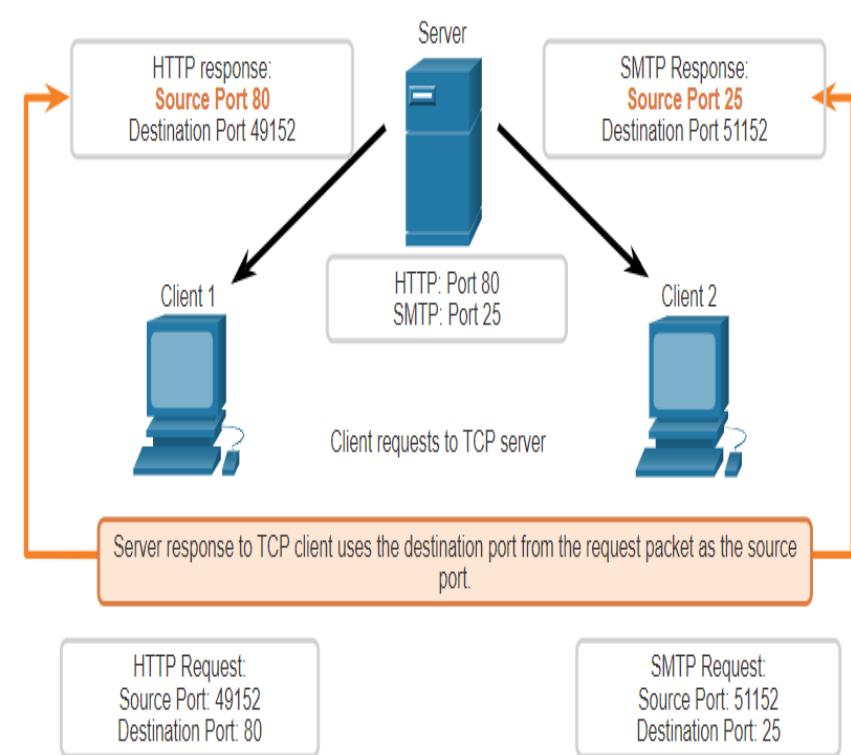


- Packets can arrive out-of-order, and packets may also be stored in buffers within the network and then arrive at much later times.
- The round-trip time will change with different connections and connection establishment is more complicated.

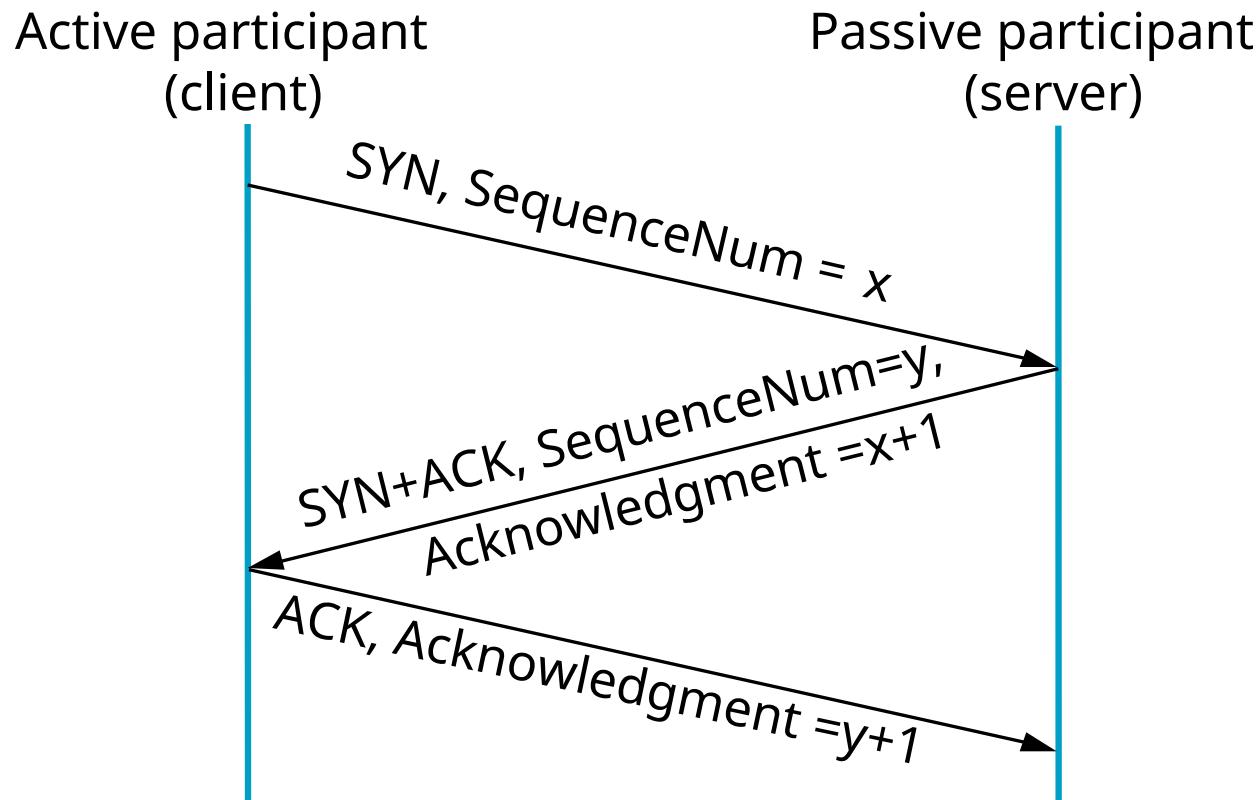
TCP Server Processes

Each application process running on a server is configured to use a port number.

- An individual server cannot have two services assigned to the same port number within the same transport layer services.
- An active server application assigned to a specific port is considered open, which means that the transport layer accepts, and processes segments addressed to that port.
- Any incoming client request addressed to the correct socket is accepted, and the data is passed to the server application.



TCP Connection Establishment



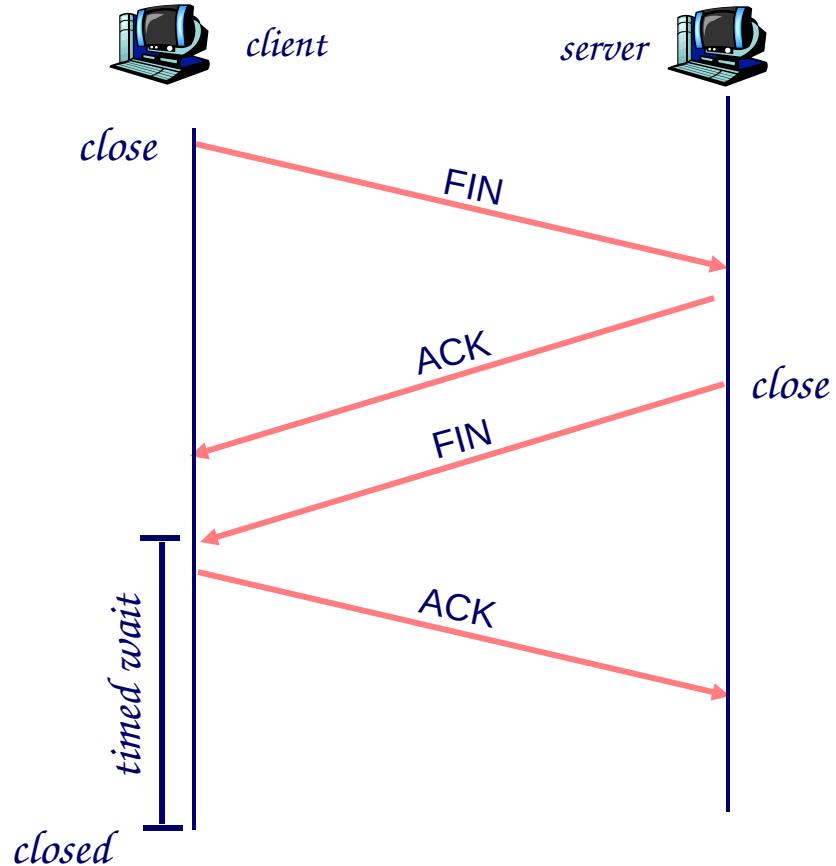
Closing a TCP Connection (1)

client closes socket:

```
clientSocket.close();
```

Step 1: client end system sends TCP FIN control segment to server

Step 2: server receives FIN, replies with ACK. Closes connection, sends FIN.



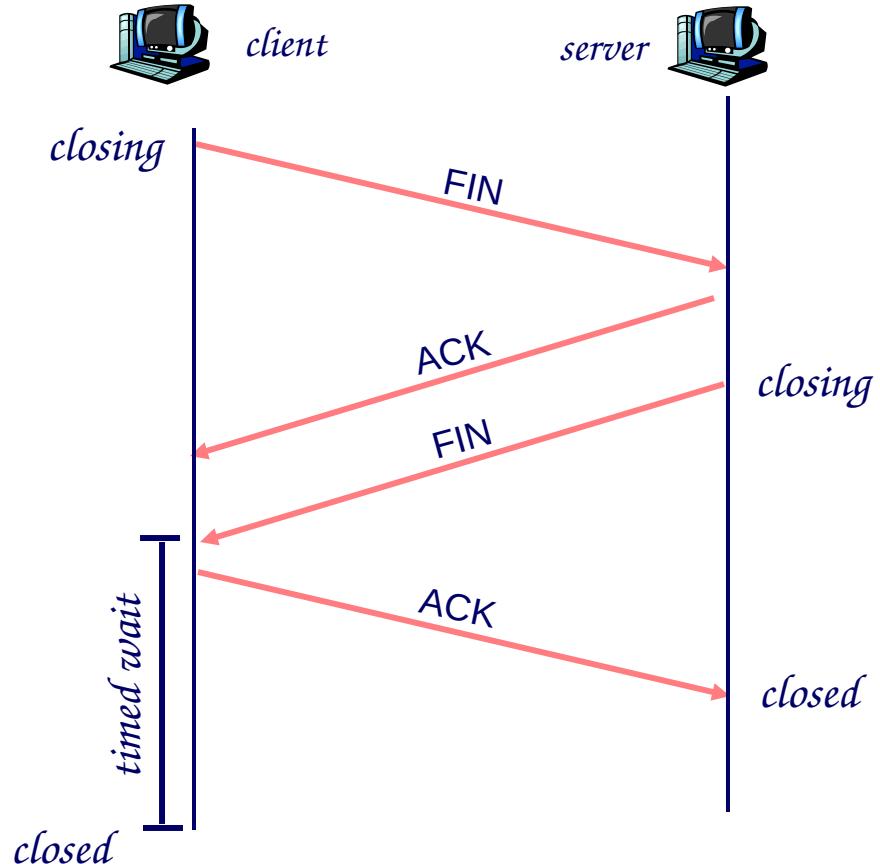
Closing a TCP Connection (2)

Step 3: client receives FIN, replies with ACK.

- Enters “timed wait” - will respond with ACK to received FINs

Step 4: server, receives ACK. Connection closed.

Note: with small modification, can handle simultaneous FINs.



Reliability and Flow Control in TCP

Flow Control and Congestion Control

- In a network, it is often desirable to **limit the rate** at which a source can send traffic into the subnet.
- If this is not done and sources send at too high of a rate, then **buffers** within the network will **fill-up** resulting in long delays and eventually **packets being dropped**.
- Moreover as packets gets dropped, **retransmission** may occur, leading to even more traffic.
- When **sources are not regulated** this can lead to **congestion collapse** of the network, where very little traffic is delivered to the destination.

Flow Control and Congestion Control

Two different factors can limit the rate at which a source sends data.

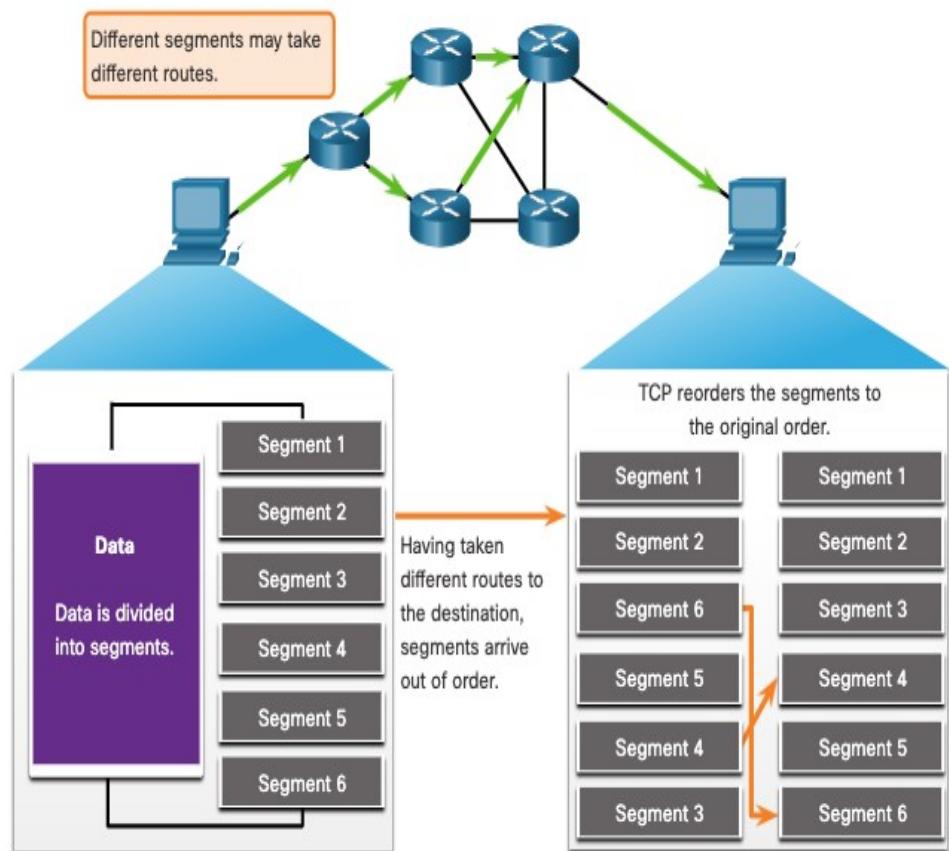
- inability of the destination to accept new data. Techniques that address this: ***flow control***.
- number of packets within the subnet. Techniques that address this: ***congestion control***.

Flow Control and Congestion Control

- Flow control and congestion control can be addressed at the **transport layer**, but may also be addressed at other layers.
- For example, some **DLL protocols** perform **flow control** on each link. And some congestion control approaches are done at the network layer.
- Both flow control and congestion control are **part of TCP**.

TCP Reliability- Guaranteed and Ordered Delivery

- TCP can also help maintain the flow of packets so that devices do not become overloaded.
- There may be times when TCP segments do not arrive at their destination or arrive out of order.
- All the data must be received and the data in these segments must be reassembled into the original order.
- Sequence numbers are assigned in the header of each packet to achieve this goal.

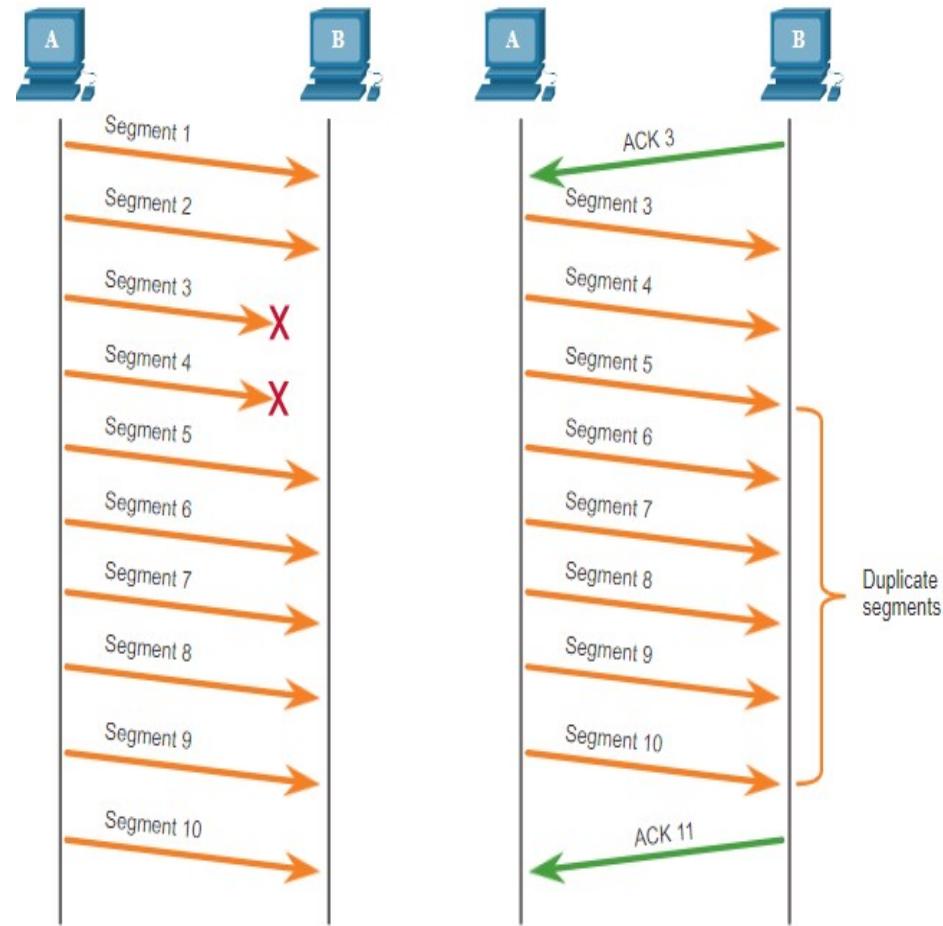


Data Loss and Retransmission

No matter how well designed a network is, data loss occasionally occurs.

TCP provides methods of managing these segment losses. Among these is a mechanism to retransmit segments for unacknowledged data.

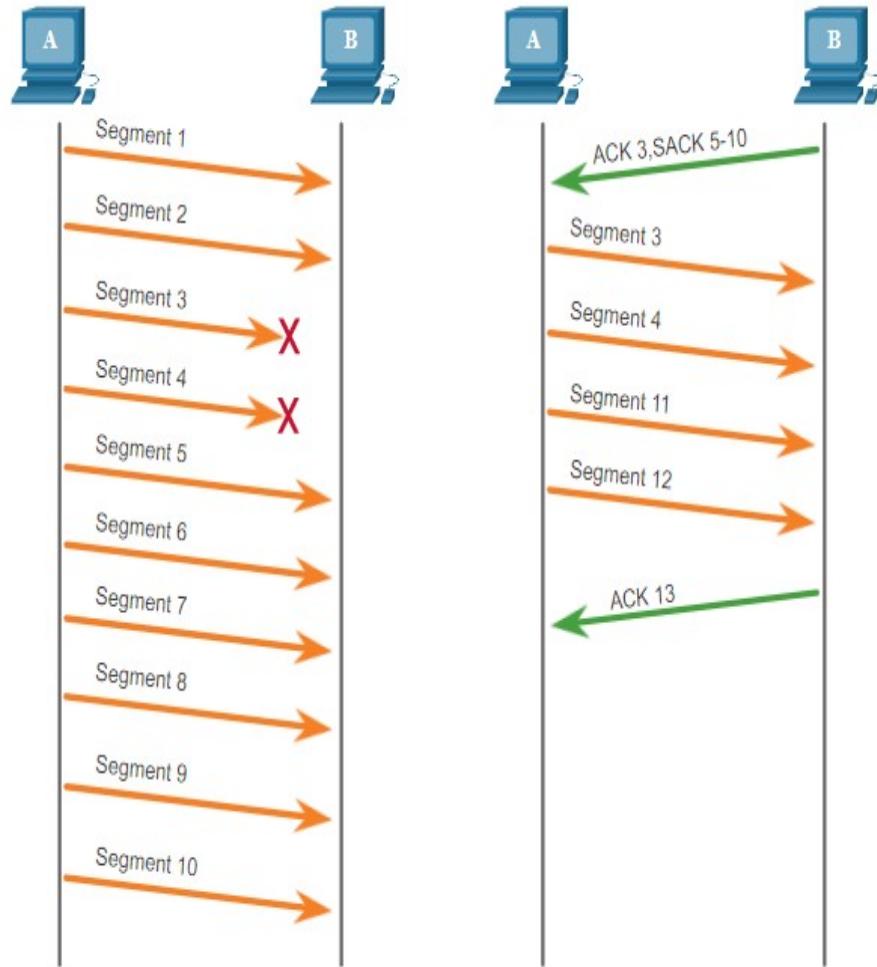
This is Go-Back N



Data Loss and Retransmission (Cont.)

Host operating systems today typically employ an optional TCP feature called **selective acknowledgment (SACK)**, negotiated during the three-way handshake.

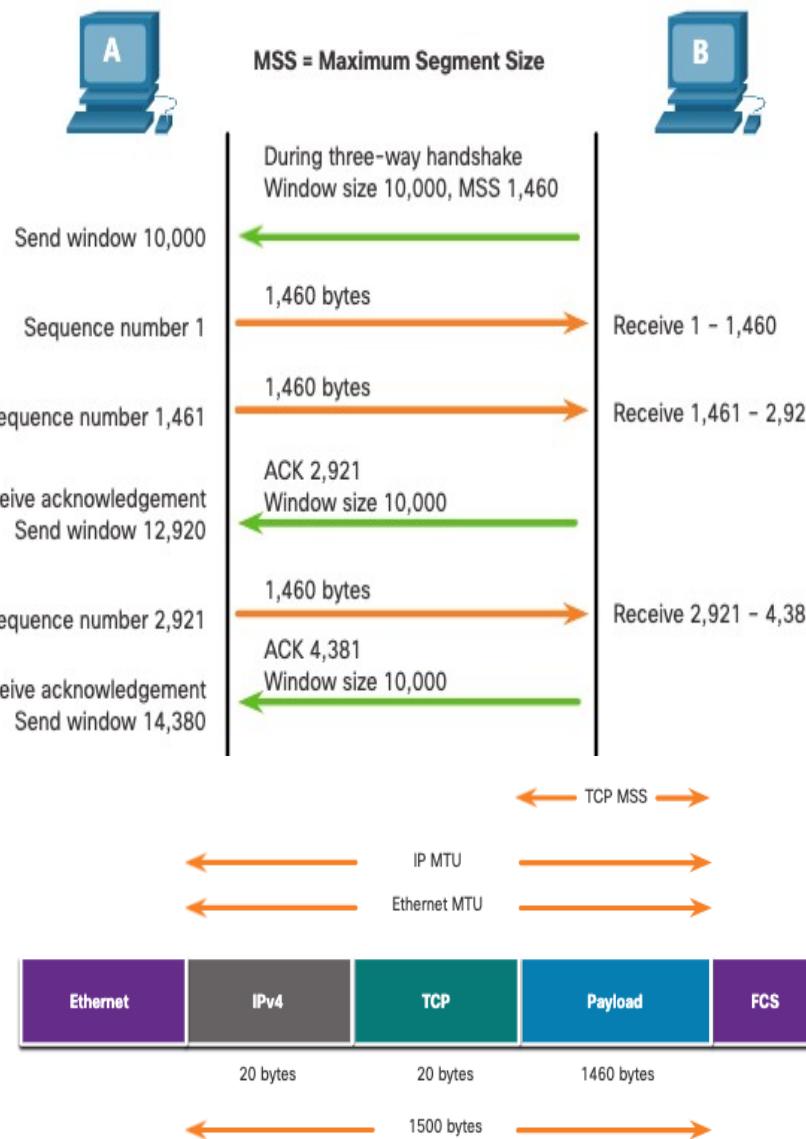
If both hosts support SACK, the receiver can explicitly acknowledge which segments (bytes) were received including any discontinuous segments.



TCP Flow Control – Maximum Segment Size

Maximum Segment Size (MSS) is the maximum amount of data that the destination device can receive.

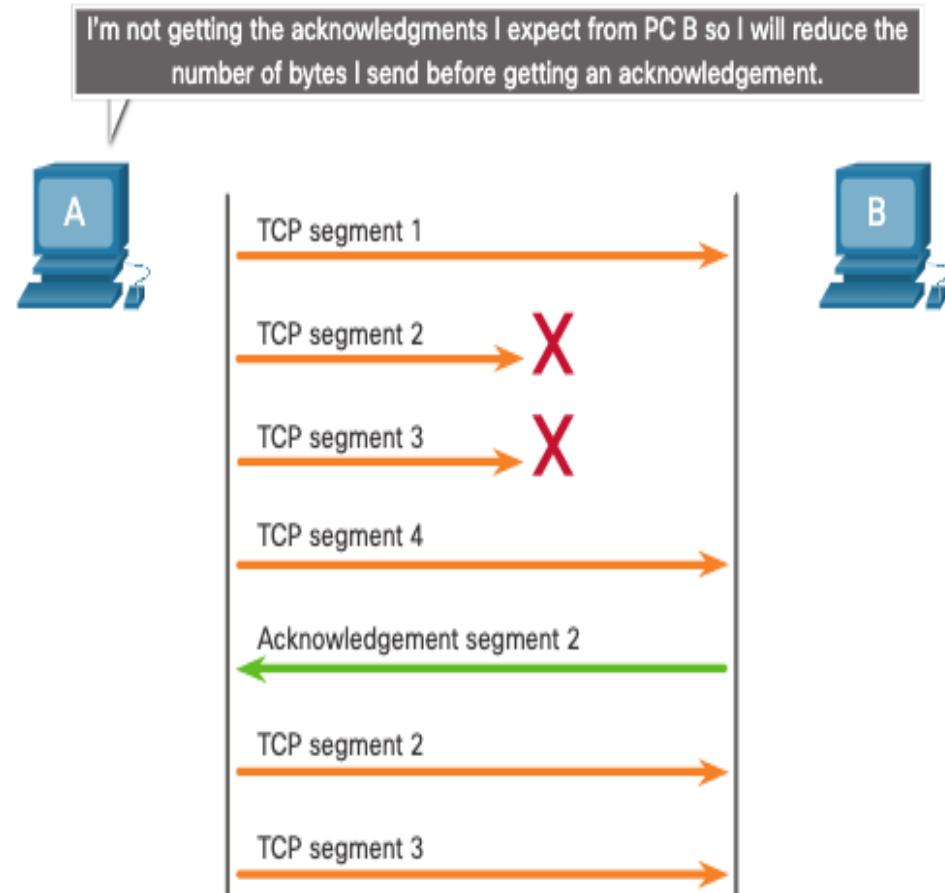
- A common MSS is 1,460 bytes when using IPv4.
- A host determines the value of its MSS field by subtracting the IP and TCP headers from the Ethernet maximum transmission unit (MTU), which is 1500 bytes by default.
- 1500 minus 60 (20 bytes for the IPv4 header and 20 bytes for the TCP header) leaves 1460 bytes.



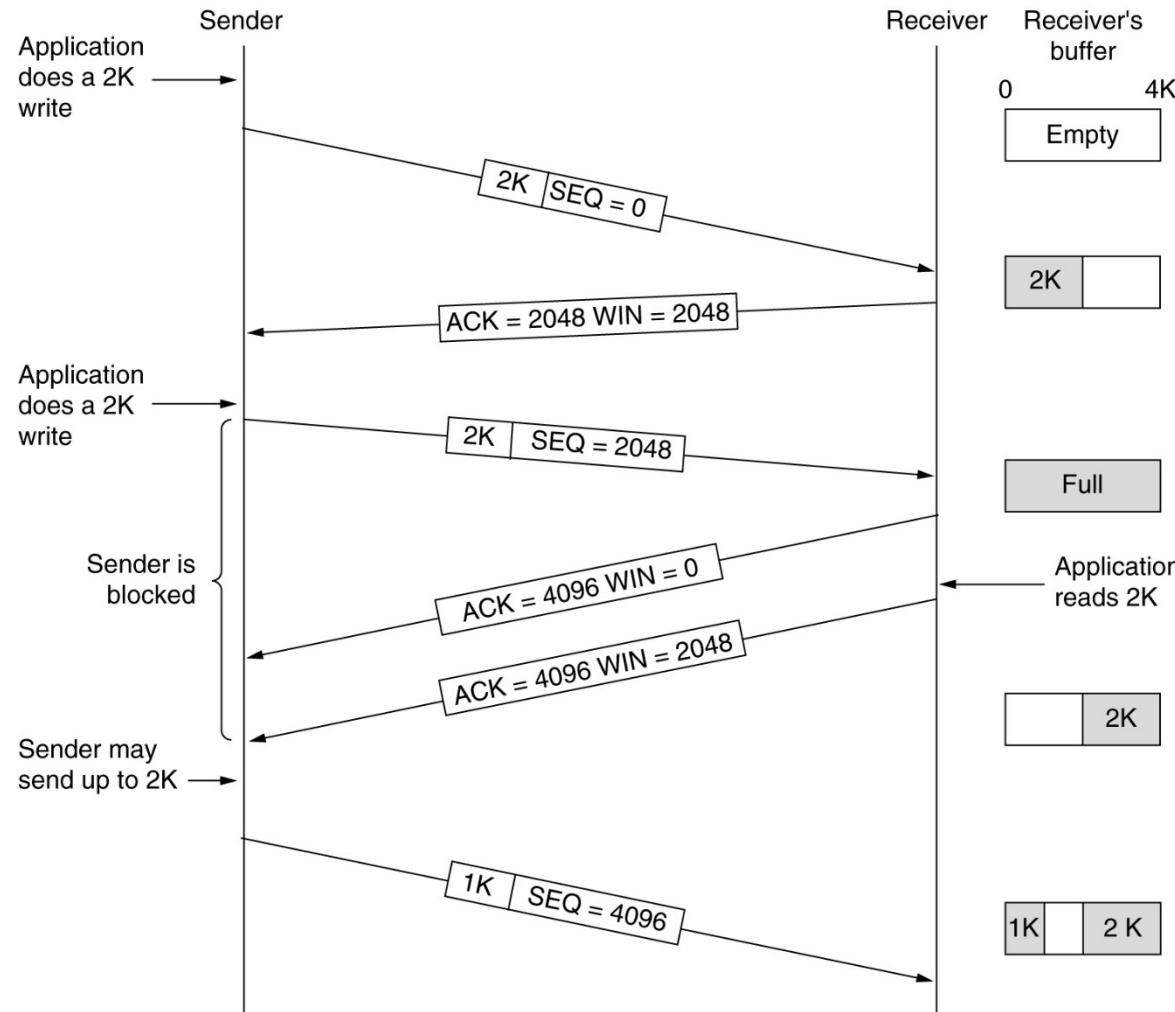
TCP Flow Control – Congestion Avoidance

When congestion occurs on a network, it results in packets being discarded by the overloaded router.

To avoid and control congestion, TCP employs several congestion handling mechanisms, timers, and algorithms.



TCP Transmission Policy



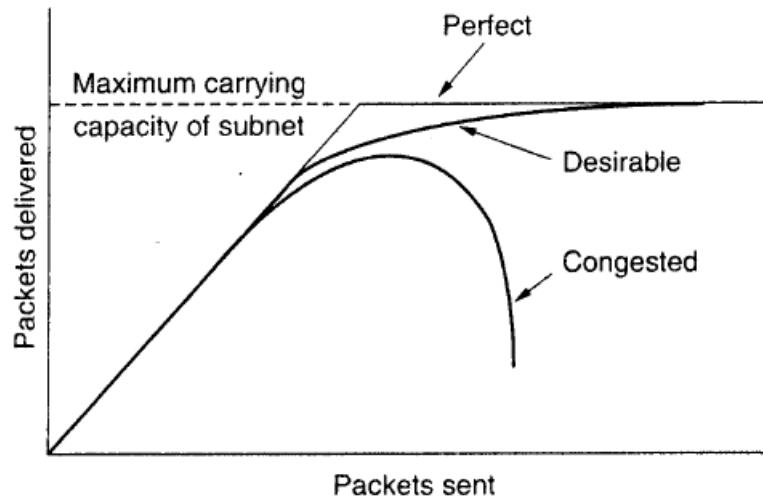
Window management in TCP.

Congestion

Congestion arises when the **total load on the network becomes too large**. This leads to queues building up and to long delays

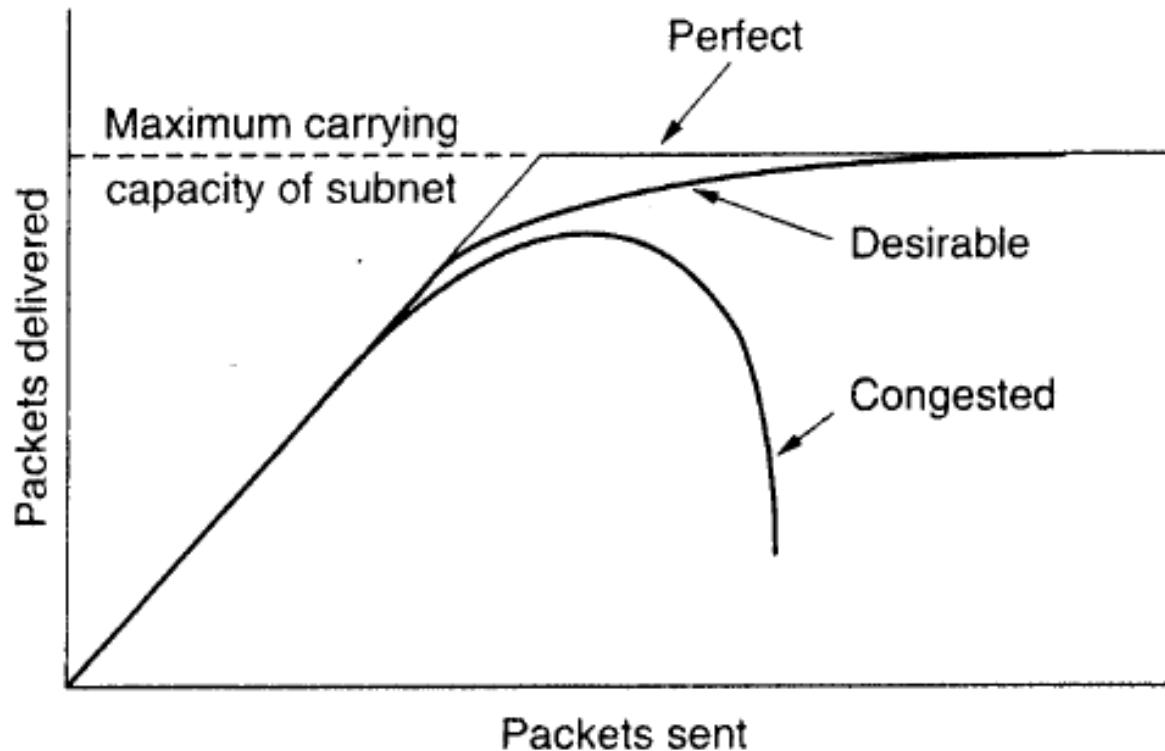
If sources retransmit messages, then this can lead to even more congestion and eventually to **congestion collapse**.

Notice, as the offered load increase, the number of packets delivered at first increases, but at high enough loads, this rapidly decreases.



Congestion

Notice, as the offered load increase, the number of packets delivered at first increases, but at high enough loads, this rapidly decreases.



Approaches to Congestion Control

Congestion control may be addressed at both the network level and the transport layer.

At the network layer possible approaches include:

Packet dropping → when a buffer becomes full a router can drop waiting packets - if not coupled with some other technique, this can lead to greater congestion through retransmissions.

Packet scheduling → certain scheduling policies may help in avoiding congestion - in particular scheduling can help to isolate users that are transmitting at a high rate.

Approaches to Congestion Control

Dynamic routing → when a link becomes congested, change the routing to avoid this link - this only helps up to a point (eventually all links become congested) and can lead to instabilities

Admission control/Traffic policing - Only allow connections in if the network can handle them and make sure that admitted sessions do not send at too high of a rate - only useful for connection-oriented networks.

Approaches to Congestion Control

An approach that can be used at either the network or transport layers is

Rate control → this refers to techniques where the source rate is explicitly controlled based on feedback from either the network and/or the receiver.

For example, routers in the network may send a source a "choke packet" upon becoming congested. When receiving such a packet, the source should lower its rate.

Approaches to Congestion Control

These approaches can be classified as either "congestion avoidance" approaches, if they try to prevent congestion from ever occurring, or as "congestion recovery" approaches, if they wait until congestion occurs and then react to it. In general, "better to prevent than to recover."

Different networks have used various combinations of all these approaches.

Traditionally, rate control at the transport layer has been used in the Internet, but new approaches are beginning to be used that incorporate some of the network layer techniques discussed above.

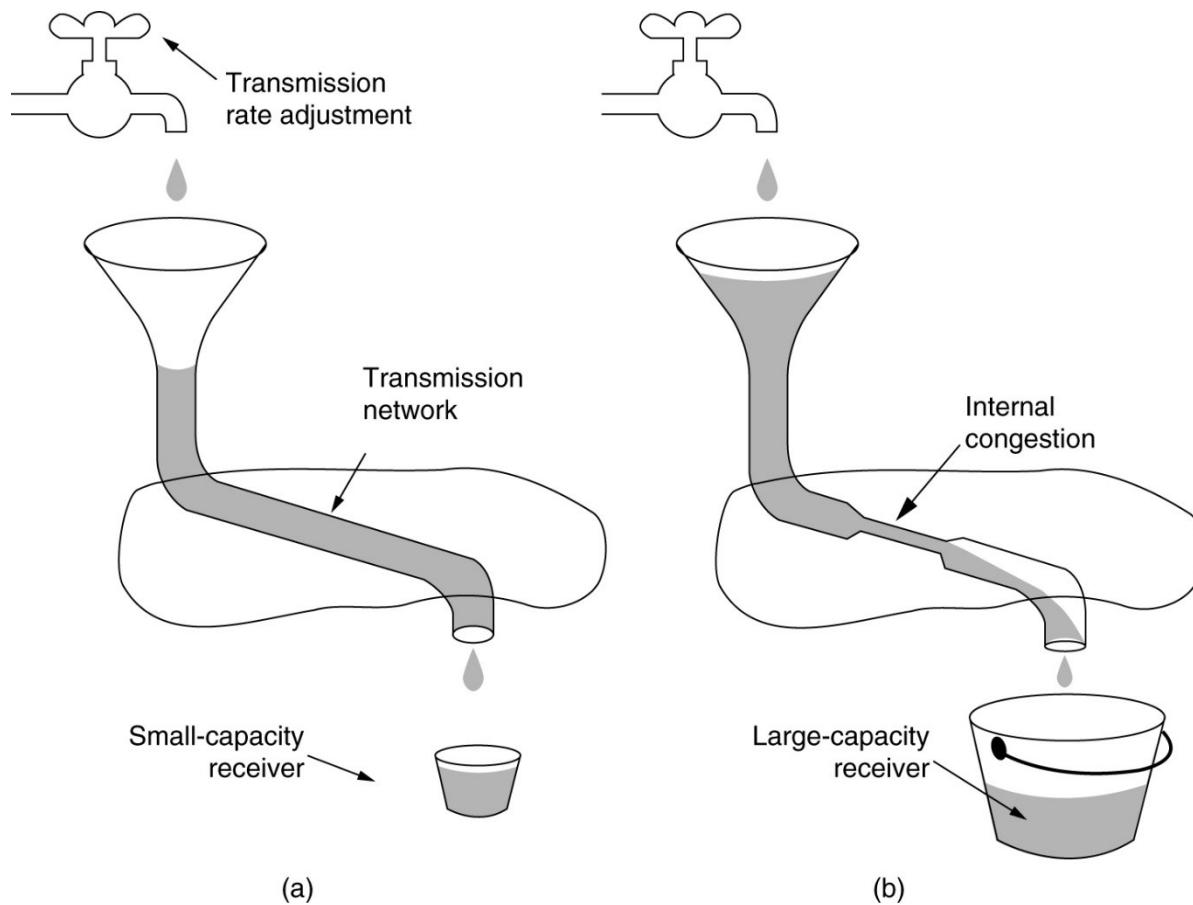
Congestion Control in TCP

TCP implements end-to-end congestion control. TCP detects congestion via the ACK's from the sliding-window ARQ algorithm used for providing reliable service.

When the source times out before receiving an ACK, the most likely reason is because a link became congested. TCP uses this as an indication of congestion. In this case, TCP will slow down the transmission rate.

TCP controls the transmission rate of a source by varying the window size used in the sliding window protocol.

TCP Flow control versus Congestion Control



Slow Start

Initial CW = 1.

After each ACK, CW += 1;

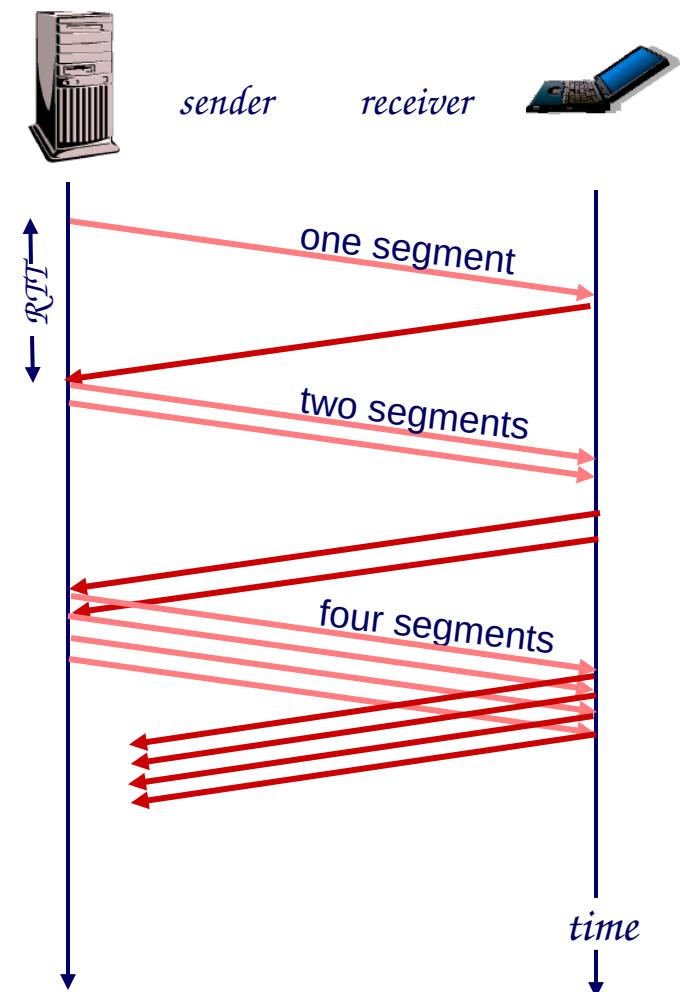
Continue until:

- Loss occurs OR
- CW > slow start threshold

Then switch to congestion avoidance

If we detect loss, cut CW in half

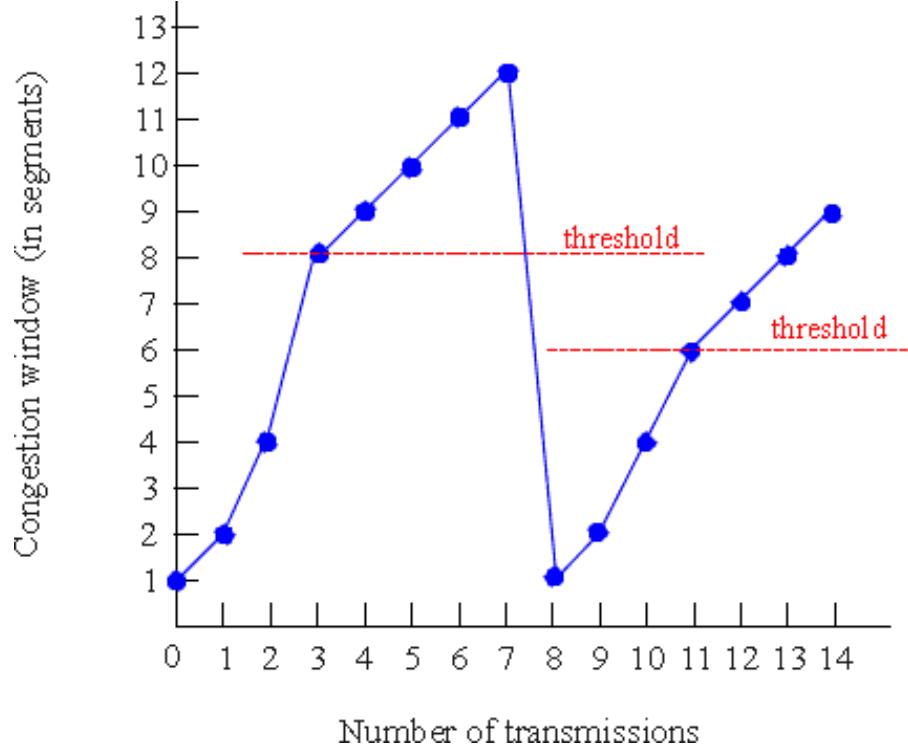
Exponential increase in window size per RTT



Congestion Avoidance

```
Until (loss) {  
    after CW packets ACKed:  
        CW += 1;  
    }  
    ssthresh = CW/2;  
Depending on loss type:  
    SACK/Fast Retransmit:  
        CW/= 2; continue;  
    Course grained timeout:  
        CW = 1; go to slow start.
```

TCP Reno: $CW = CW/2$ after loss



TCP Tahoe: $CW=1$ after a loss

How are losses recovered?

Say packet is lost (data or ACK!)

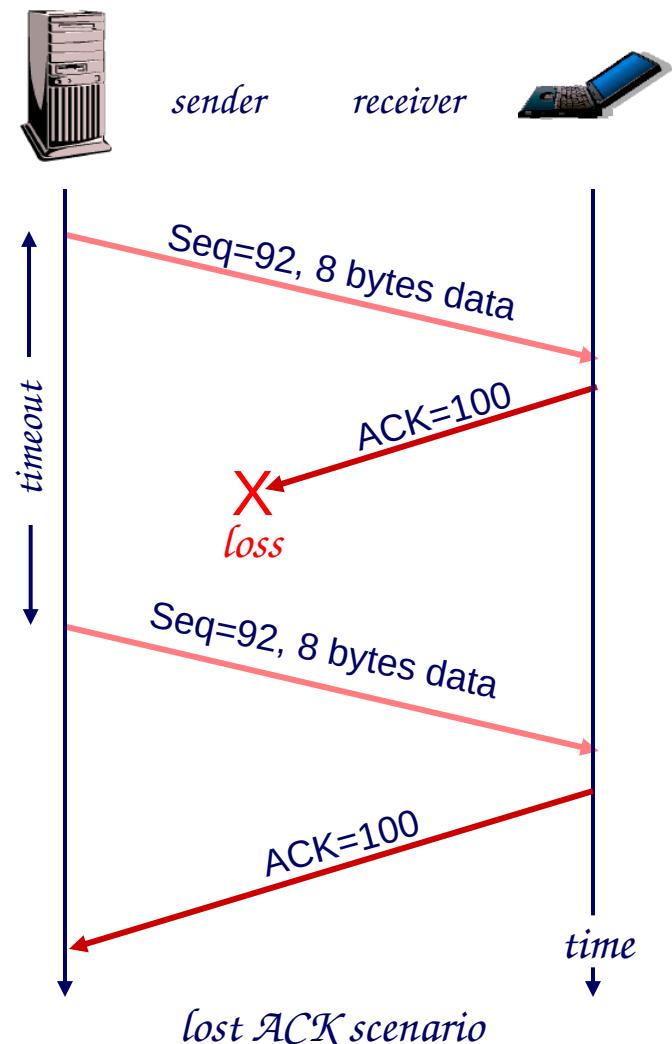
Coarse-grained Timeout:

- Sender does not receive ACK after some period of time
- Event is called a retransmission time-out (RTO)
- RTO value is based on estimated round-trip time (RTT)
- RTT is adjusted over time using exponential weighted moving average:

$$\text{RTT} = (1-x) * \text{RTT} + (x) * \text{sample}$$

(x is typically 0.1)

First done in TCP Tahoe



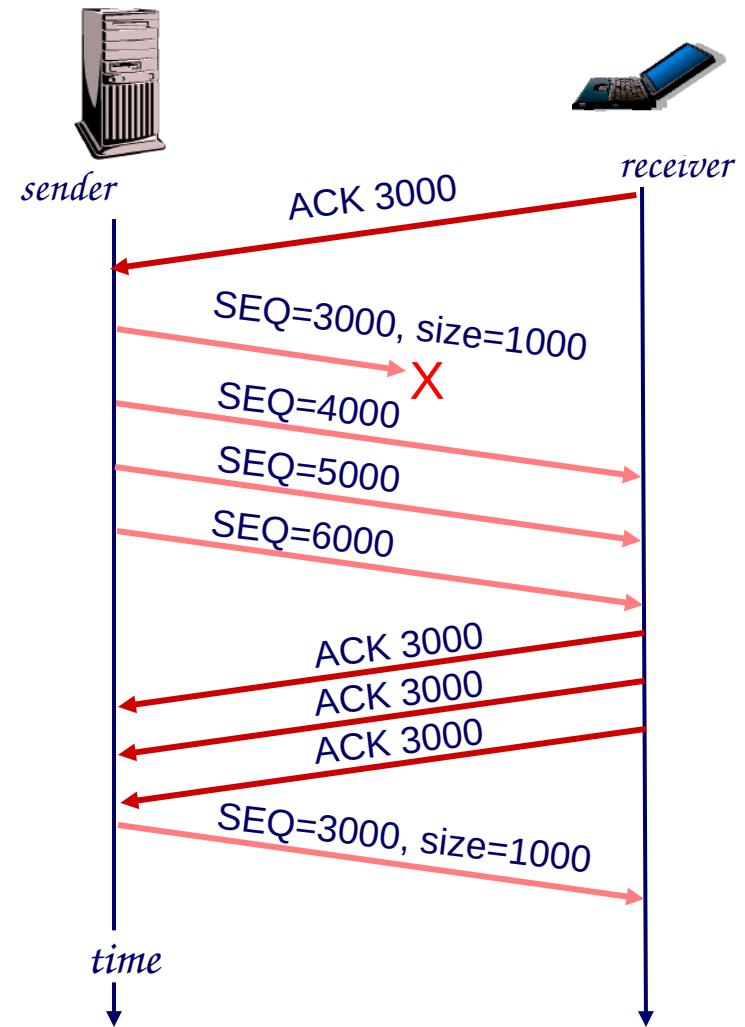
Fast Retransmit (TCP Reno)

Receiver expects N, gets N+1:

- Immediately sends ACK(N)
- This is called a **duplicate ACK**
- Does NOT delay ACKs here!
- Continue sending dup ACKs for each subsequent packet (not N)

Sender gets 3 duplicate ACKs:

- Infers N is lost and resends
- 3 chosen so out-of-order packets don't trigger Fast Retransmit accidentally
- Called "fast" since we don't need to wait for a full RTT



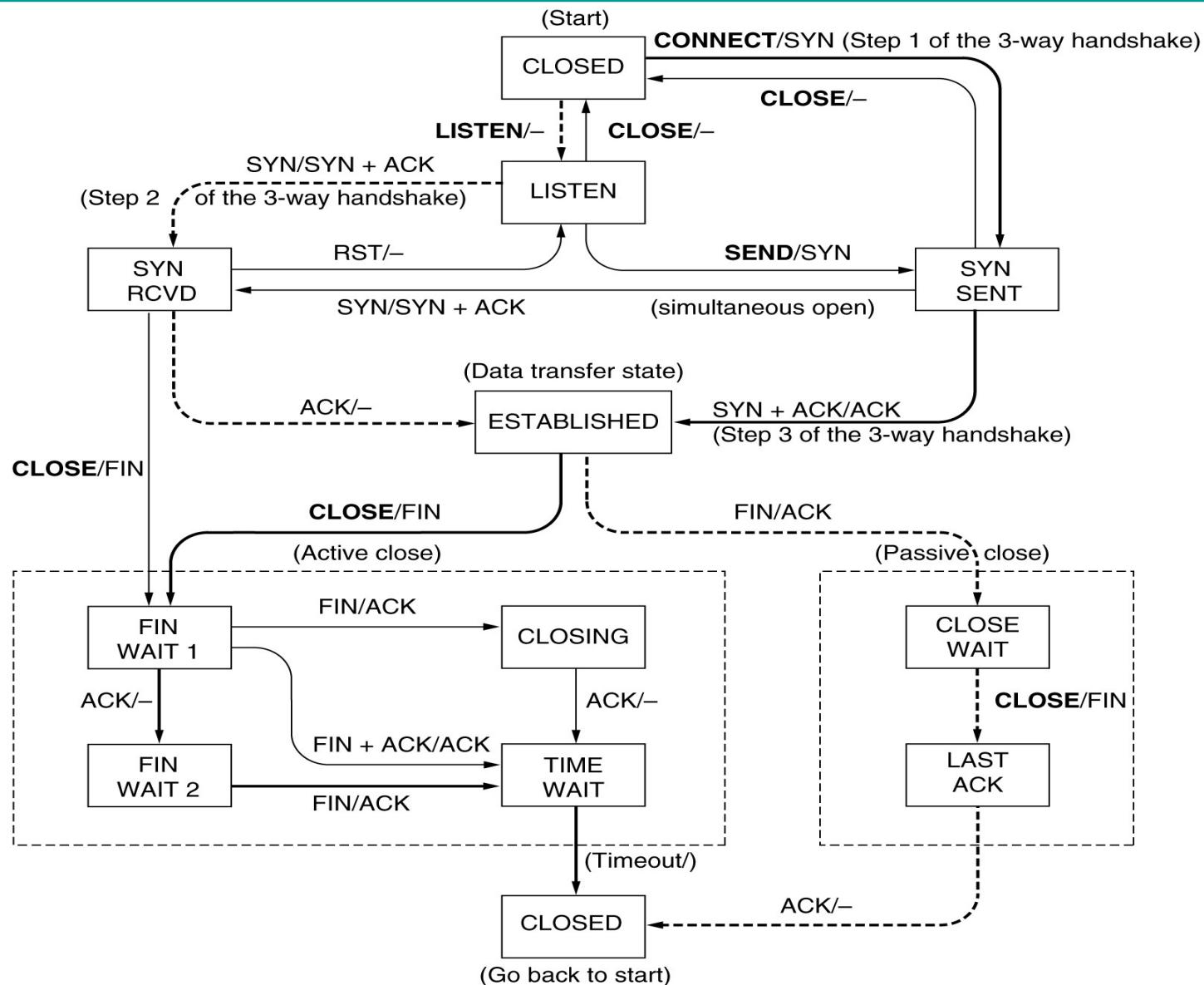
Introduced in TCP Reno

TCP Connection Management Modeling

The states used in the TCP connection management finite state machine.

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

TCP Connection Management Modeling (2)



PART D: UDP

- **Unreliable**
- **Connectionless**
- **No TCP's flow control;**
- **Applications where prompt delivery more important than accurate delivery (speech, video, ...)**

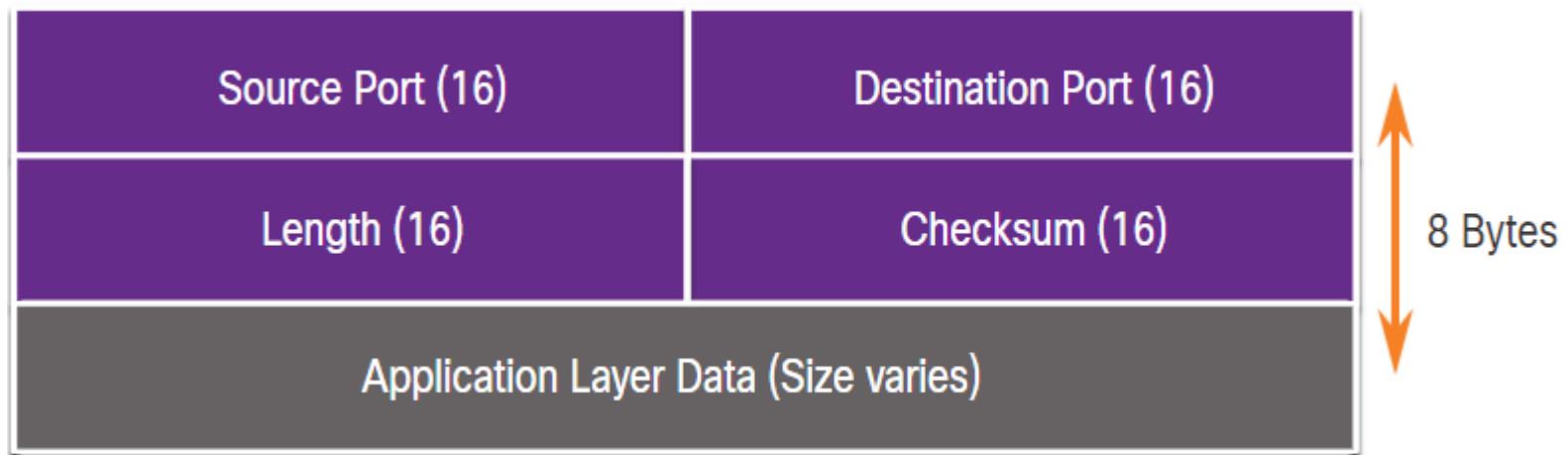
UDP Features

UDP features include the following:

- Data is reconstructed in the order that it is received.
- Any segments that are lost are not resent.
- There is no session establishment.
- The sending is not informed about resource availability.

UDP Header

The UDP header is far simpler than the TCP header because it only has four fields and requires 8 bytes (i.e. 64 bits).



UDP Header Fields

The table identifies and describes the four fields in a UDP header.

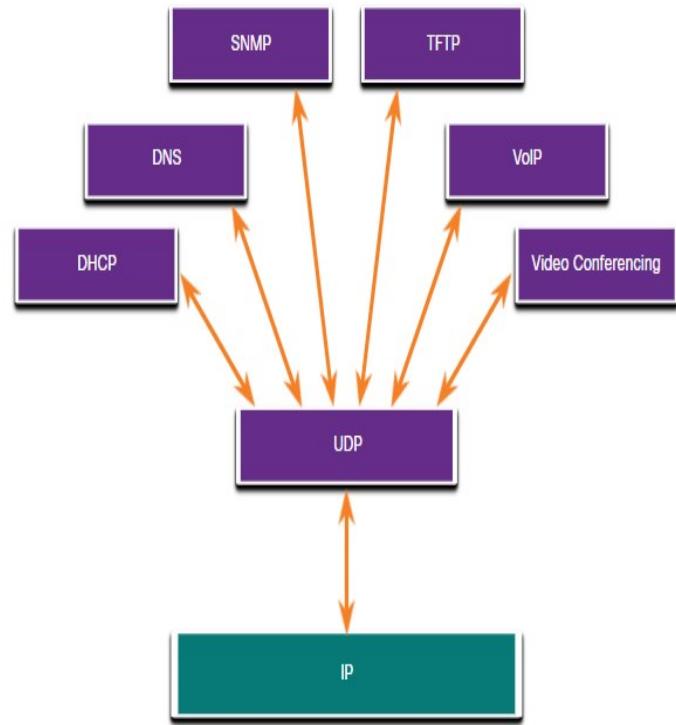
UDP Header Field	Description
Source Port	A 16-bit field used to identify the source application by port number.
Destination Port	A 16-bit field used to identify the destination application by port number.
Length	A 16-bit field that indicates the length of the UDP datagram header.
Checksum	A 16-bit field used for error checking of the datagram header and data.

Applications that use UDP

Live video and multimedia applications - These applications can tolerate some data loss but require little or no delay. Examples include VoIP and live streaming video.

Simple request and reply applications - Applications with simple transactions where a host sends a request and may or may not receive a reply. Examples include DNS and DHCP.

Applications that handle reliability themselves - Unidirectional communications where flow control, error detection, acknowledgments, and error recovery is not required, or can be handled by the application. Examples include SNMP and TFTP.



UDP

Datagram protocol → it does not have to establish a connection to another machine before sending data.

1. Takes the data an application provides
2. Packs it into a UDP packet
3. Hands it to the IP layer.
4. The packet is then put on the wire, and that is where it ends.

There is no way to guarantee that the packet will reach its destination.

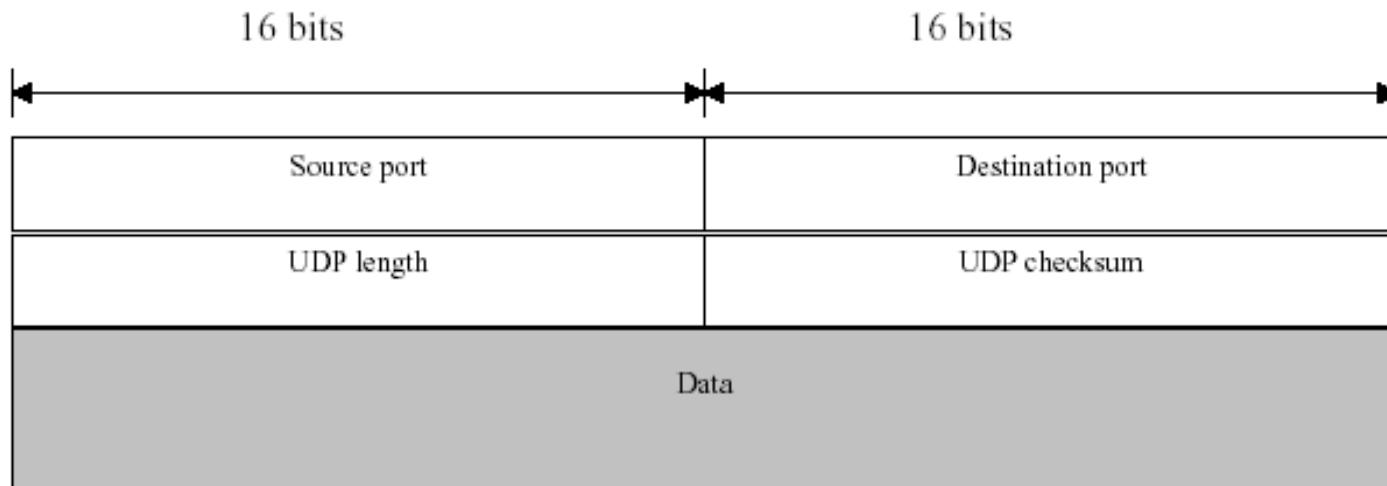
UDP Packet

Basically all UDP provides is a multiplexing capability on top of IP.

The length field gives the length of the entire packet including the header.

The checksum is calculated on the entire packet (and also on part of the IP header).

Calculating the checksum is optional, and if an error is detected the packet may be discarded or may be passed on to the application with an error flag.



APPLICATION LAYER

Standard Application Layer Protocols

2

DNS

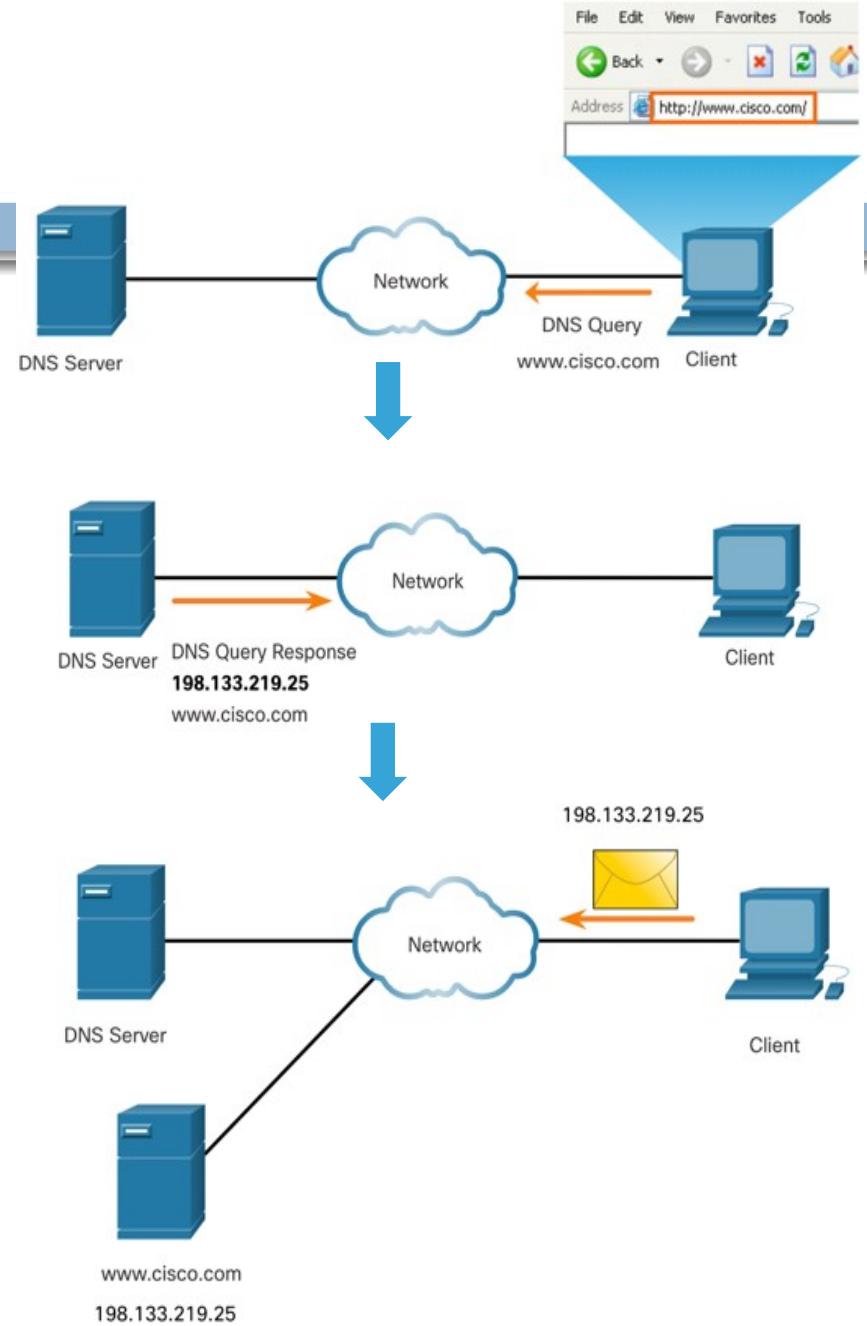
DNS

3

- Stands for Domain Name System
- Devices on the network has IP addresses
- But Human always prefers names instead of numeric addresses
- DNS maps the human readable addresses to corresponding IP addressees
- DNS uses UDP

DNS

- Domain names were created to convert the numeric IP addresses into a simple, recognizable name.
- Fully-qualified domain names (FQDNs), such as `http://www.cisco.com`, are much easier for people to remember than `198.133.219.25`.
- The DNS protocol defines an automated service that matches resource names with the required numeric network address. It includes the format for queries, responses, and data.



The nslookup Command

- Nslookup is a computer operating system utility that allows a user to manually query the DNS servers configured on the device to resolve a given host name.
- This utility can also be used to troubleshoot name resolution issues and to verify the current status of the name servers.
- When the **nslookup** command is issued, the default DNS server configured for your host is displayed.
- The name of a host or domain can be entered at the **nslookup** prompt.

```
C:\Users> nslookup
Default Server: dns-sj.cisco.com
Address: 171.70.168.183
> www.cisco.com
Server: dns-sj.cisco.com
Address: 171.70.168.183
Name: origin-www.cisco.com
Addresses: 2001:420:1101:1::a
          173.37.145.84
Aliases: www.cisco.com
> cisco.netacad.net
Server: dns-sj.cisco.com
Address: 171.70.168.183
Name: cisco.netacad.net
Address: 72.163.6.223
>
```

DNS Process Overview

6

1. User machine runs client side of the DNS application
2. The browser (or any other program that uses URL) extracts the hostname to the client side of the DNS application
3. The DNS client sends a query containing the hostname to a DNS application
4. The DNS client eventually receives a reply, which includes the ip address of the hostname

Name Space

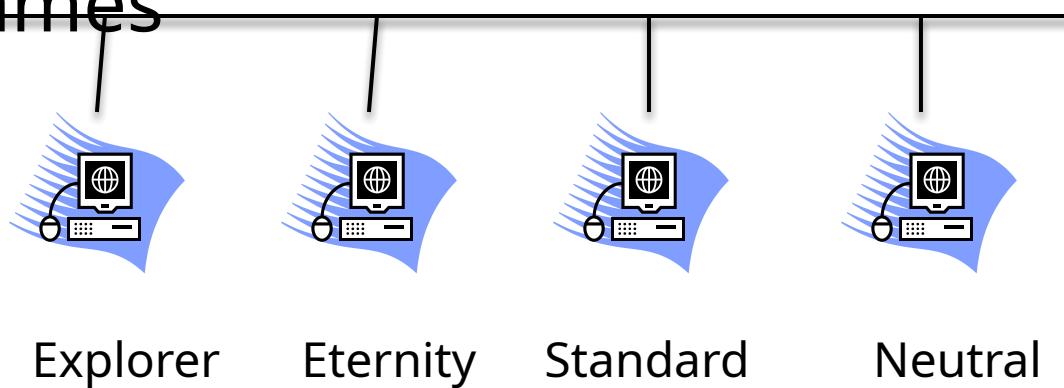
7

- Maps each address to a unique name
- Organized in two ways
 - Flat Name Space
 - Hierarchical Name Space

Flat Name Space

8

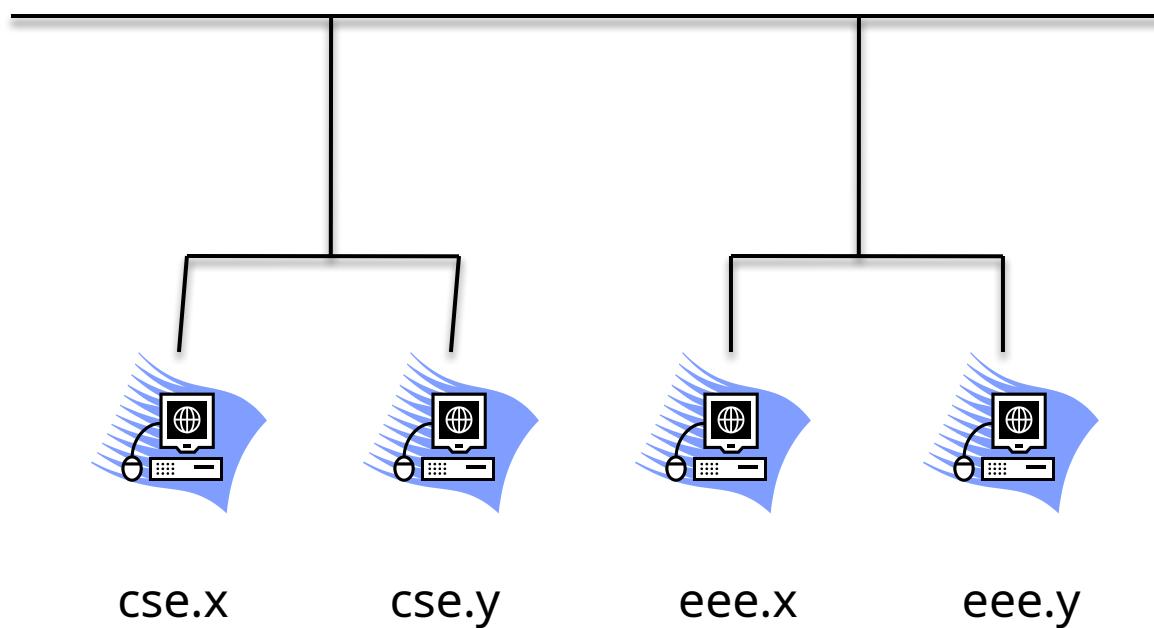
- Names are interpreted as a single, whole label
- No internal structure
- No clear relationship between any two names



Hierarchical Name Space

9

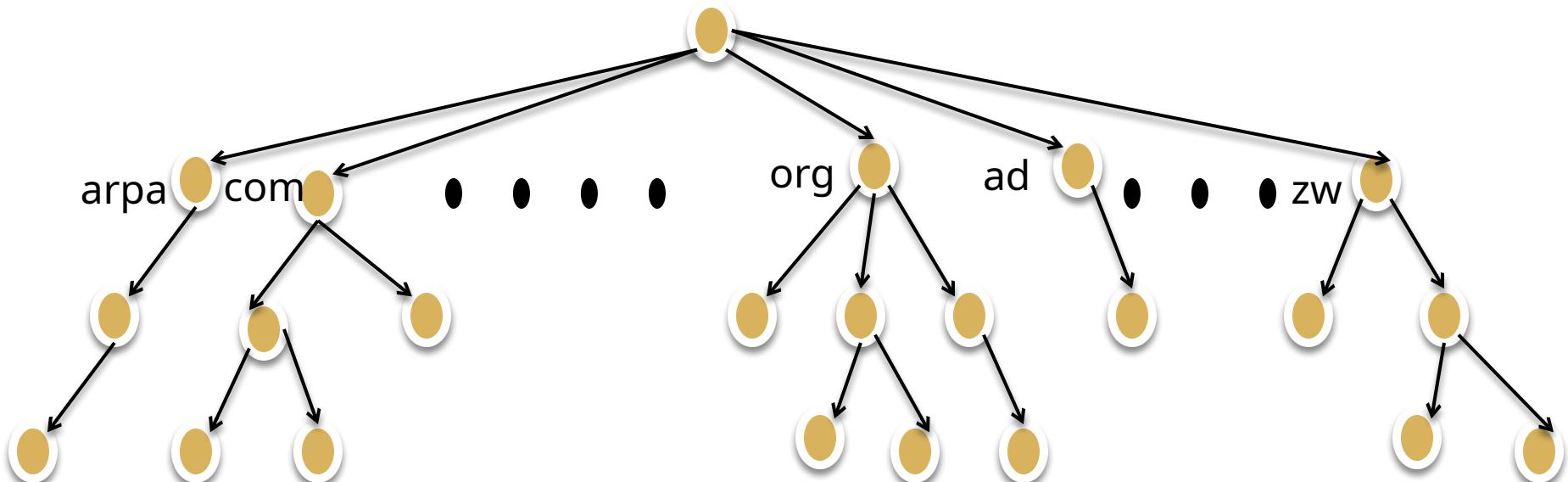
- Logical grouping is possible based on the names



Domain Name Space

10

- Names are defined in an inverted-tree structure with the root at the top
- Tree can have at most 128 levels



Labels & Domain Name

11

□ Label

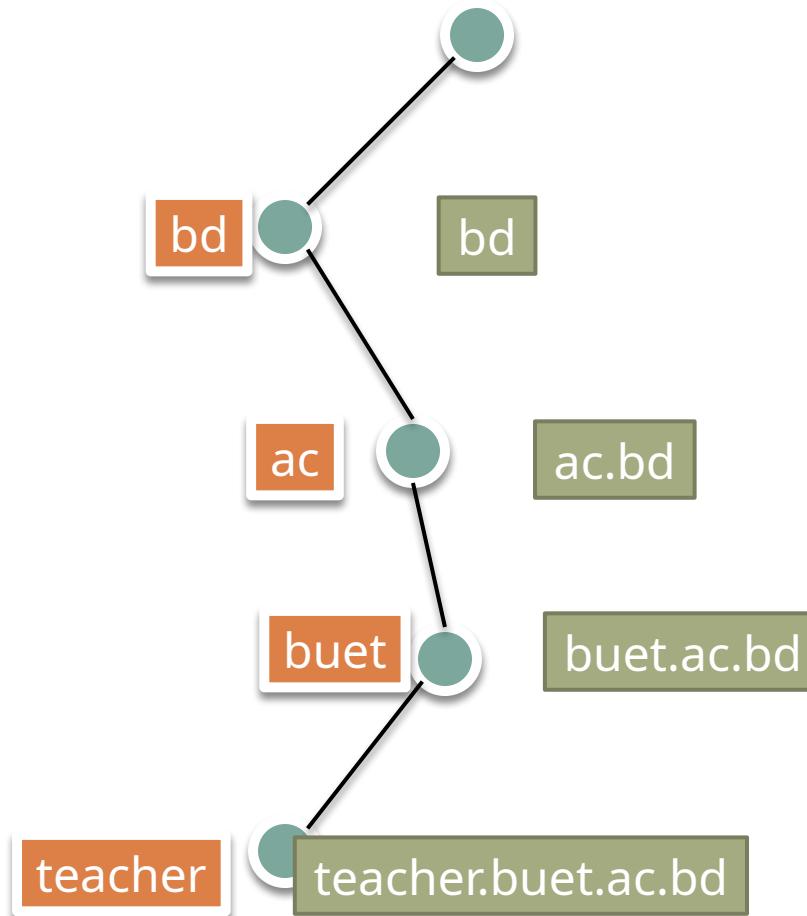
- Each node has a label of maximum 63 characters
- Root is labeled with null string
- Children of a node must have different labels for uniqueness

□ Domain Name

- Each node of a tree has a domain name
- A full domain name is a sequence of labels separated by dots(.)
- Read from node up to the root.

Label & Domain Name (Explained)

12



Fully Qualified Domain Name

13

- An FQDN is a domain name that contains the full name of a host
- Starts from a node and read up to the root
- It uniquely defines the host machine irrespective of the location of client

teacher.buet.ac.bd

Partially Qualified Domain Name

14

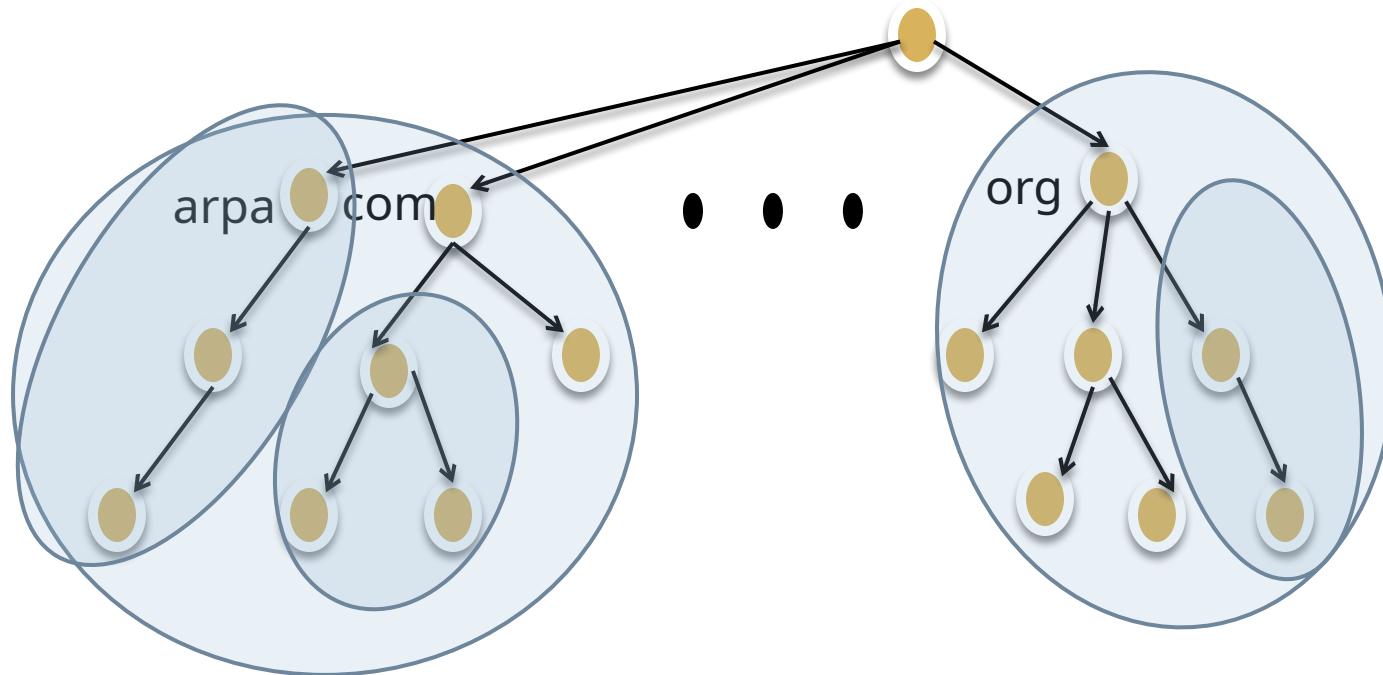
- PQDN starts from a node but does not reach the root
- Used when the name to be resolved belongs to the same site as the client
- Resolver can supply the suffix to create an FQDN



Domain

15

- Subtree of the DNS
- The name of the domain is the Domain Name of the node at the top of the subtree
- Domain is divided in subdomains



DNS Information storing

16

- In a single server
 - Inefficient
 - Unreliable
- Multiple and Distributed
 - Efficient
 - Reliable
 - But Processing Requires structure and protocols

Inefficiency of Single Server

17

1. Single point of failure
2. Traffic Volume high. All requests have to be handled by a single DNS server
3. A single DNS server cannot be close to all querying clients
4. It would be updated frequently for every new host .

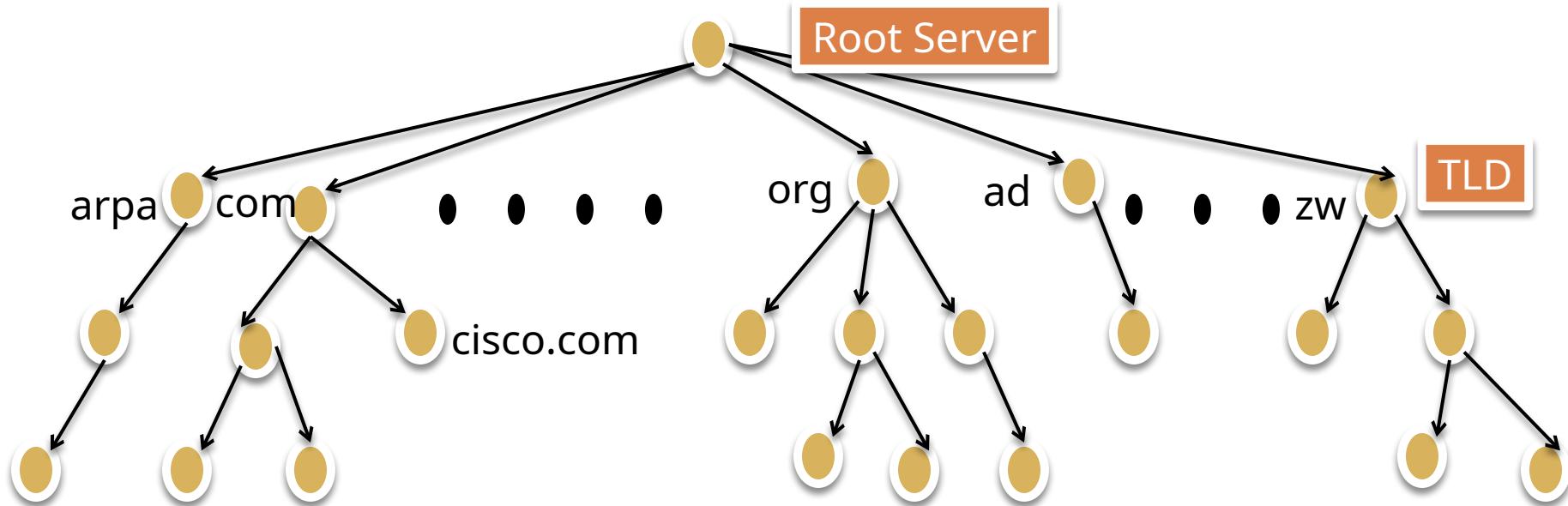
Hierarchy of Name Server

18

- Each server known as DNS server
- Each server is responsible(authoritative) for a domain
- Three classes of Server
 - Root DNS Server
 - Top-level Domain (TLD) Server
 - Authoritative DNS Server

Types of DNS Server

19



Root DNS Server

20

□ Root DNS Server

- Number of Root servers distributed throughout the world
- Some servers are in fact cluster



TLD Server & Authoritative DNS Server

21

- TLD Server
 - These servers are responsible for **Generic Top-level Domains** such as com, org, net, edu and gov and all of the **Country Top-level Domains** such as uk, fr, ca, bd
- Authoritative DNS Server
 - Every organization with publicly accessible host must provide publicly accessible DNS records that map names of those host to IP Addresses
 - An organization authoritative DNS houses these DNS records

Local DNS Server

22

- Does not strictly belong to the hierarchy of servers
- Each ISP has a local DNS server
- When a host connects to the ISP, the ISP provides IP addresses one or more of its local DNS servers
- When a host makes a DNS query, the query is sent to the local DNS server which acts as a proxy, forwarding the query into the DNS server hierarchy

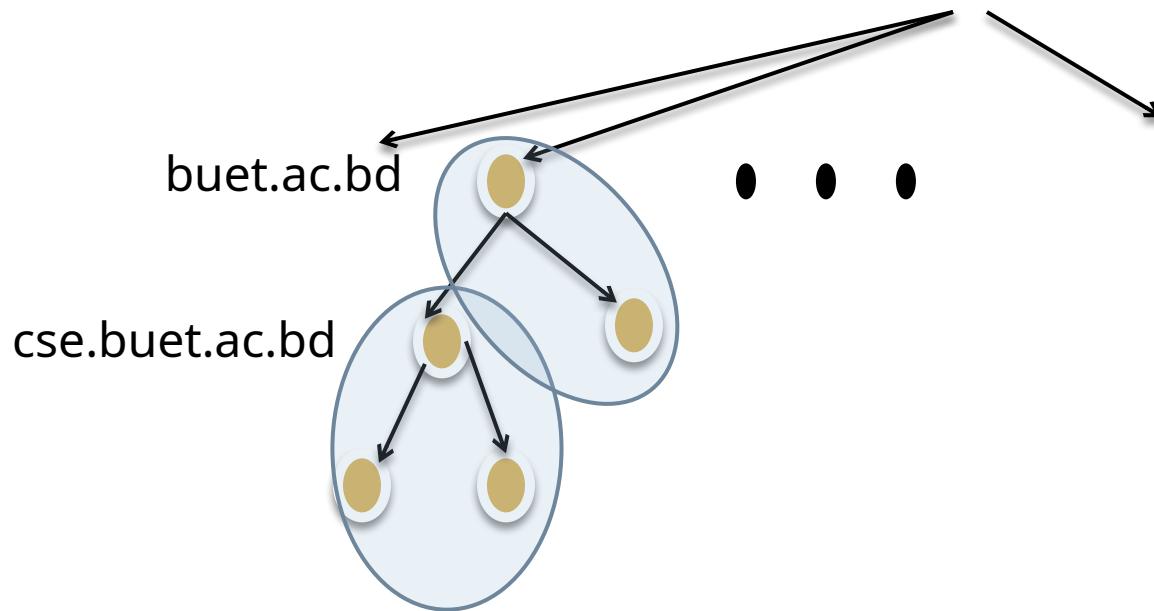
Zone

23

- Refers to a certain portion or administrative space within the global DNS
- All DNS zones form the DNS namespace
- Depending on the administrative rights delegated to a certain entity, DNS zones may consist of only one domain, or of many domains and sub-domains.
- Further authority over a sub-space could be delegated to other servers, if necessary.
- **Zone File** keeps all information for every node under the domain

Zone Explained

24



Authoritative DNS Server Types

25

- Primary DNS Server
 - Stores a file about the zone for which it is **an authority**
 - Responsible for creating, maintaining and updating Zone file
 - Stores zone file in a local disk
- Secondary DNS Server
 - Transfers the complete information about a zone from another server (primary or secondary) and stores the file in the local disk

Primary and Secondary Servers

26

- Both are **authoritative** for the zone that they serve
- The method of downloading information by secondary from primary is known as **Zone Transfer**
- Utility
 - Create Redundancy to **mitigate failure**

Resolver

27

- Host that needs to map an address to a name or a name to an address calls a DNS client called a Resolver
- The resolver accesses the closest DNS server with a mapping request.
- If the server has the information, it satisfies the resolver, otherwise, it either refers the resolver to other servers or other servers to provide information
- It also checks the occurrence of any error

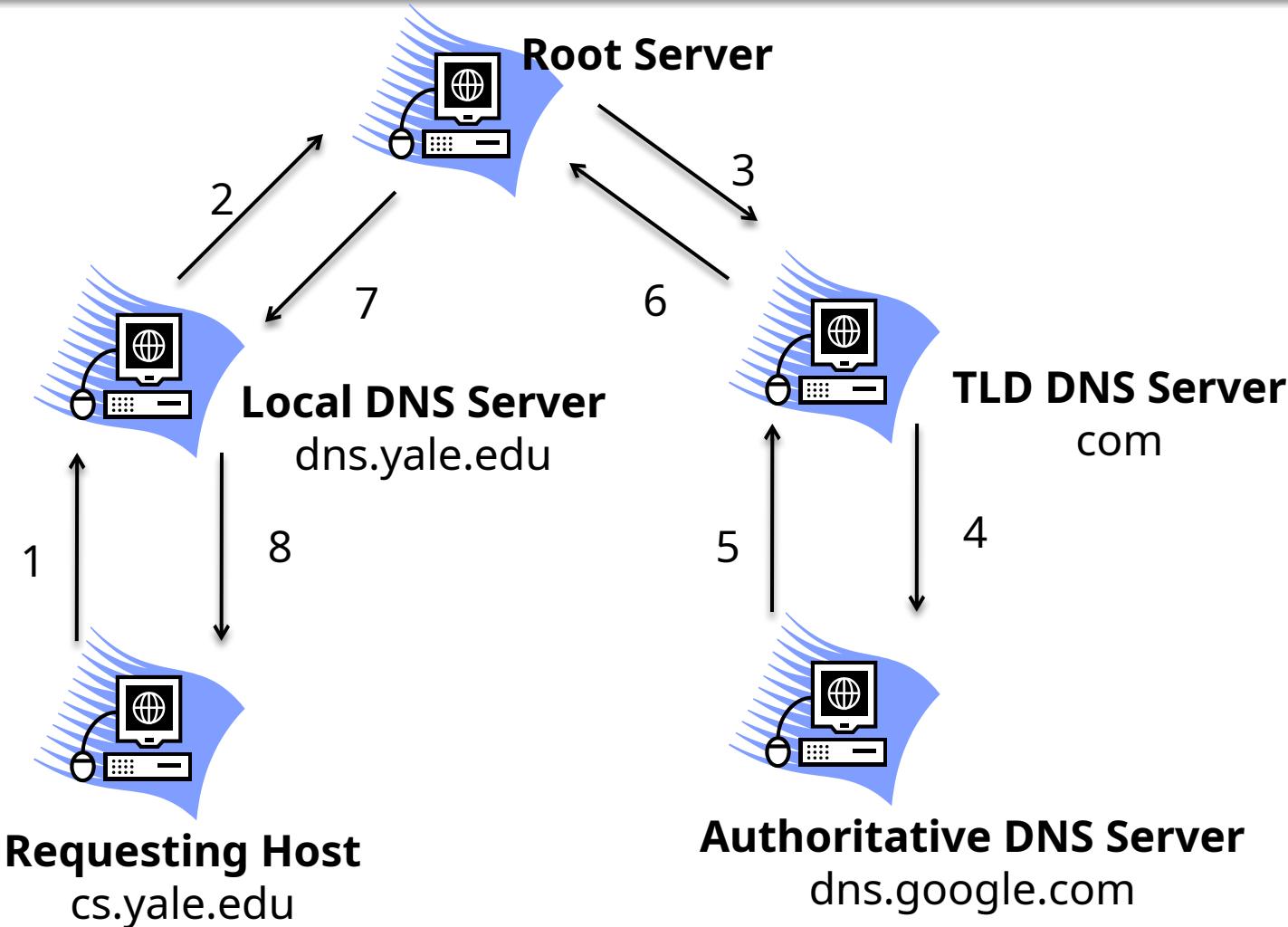
Recursive Resolution

28

- Resolver accepts the server to supply the final answer
- If the server is the authority of the DNS, it supplies the answer
- If the server is not authority, it send the request to other server (usually parent) and waits for the response.
- If the parent is authority, it responds, otherwise, it sends request to another one
- When the query is resolved, then the response travel backs to the client

Recursive Resolution Explained

29



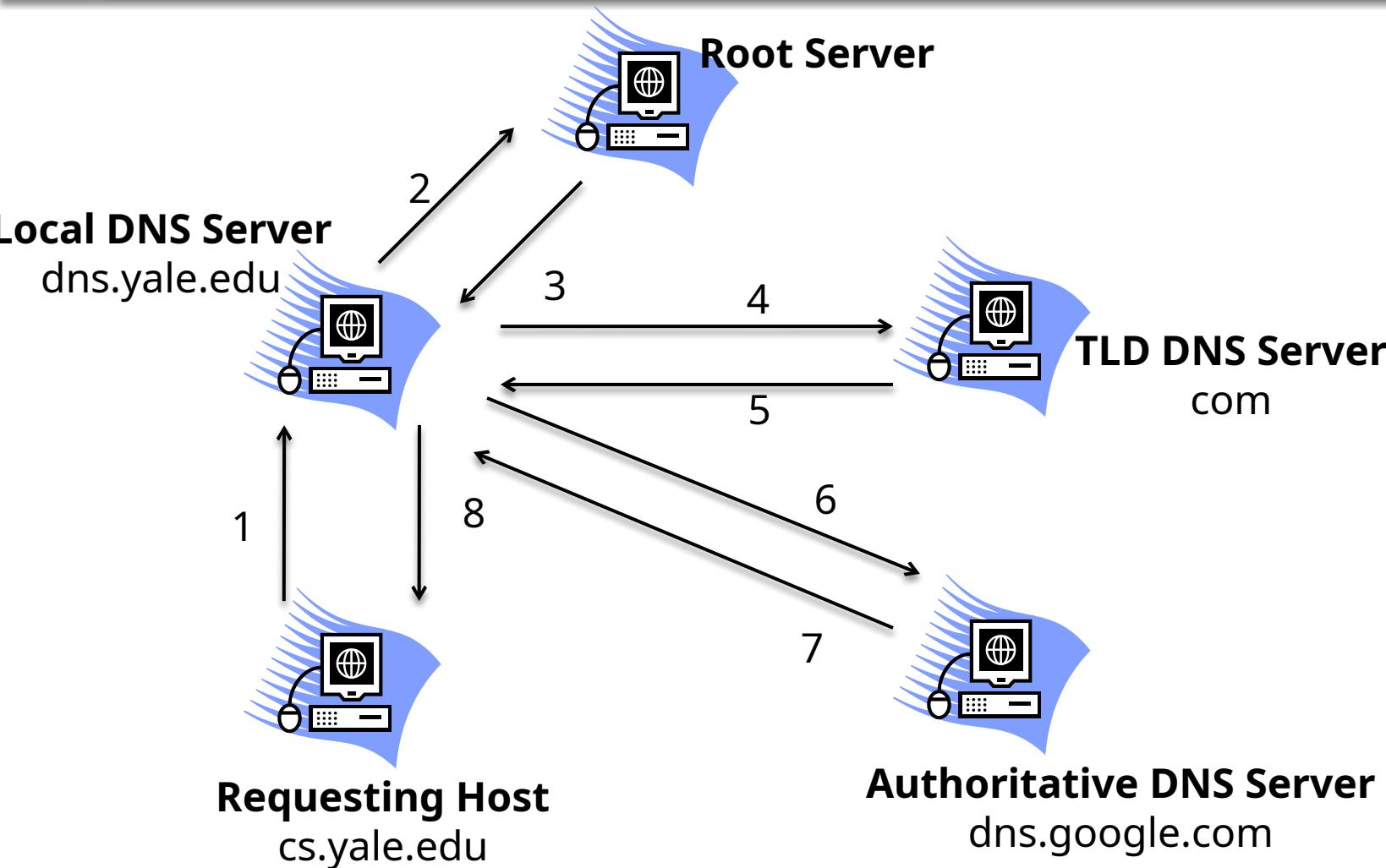
Iterative Resolution

30

- The client requests the local DNS.
- Local DNS then request another DNS Server
- If the server is an authority for the name, it sends the answer
- If the server is not an authority, it returns the ip addresses of the server which it thinks can resolve the query.
- The local DNS then repeats the query to the second server and so on
- Finally Client receives the resolution from Local DNS server

Iterative Resolution Explained

31



Caching

32

- Reduction in DNS lookup can be reduced at a greater scale if the mapping is stored locally
- When a server asks for a mapping from another server and receives response, it stores the information in its cache memory before sending it to client to satisfy other clients afterwards without further query
- The response coming from the cache is marked as **Non-authoritative**
- **TTL** is used to keep track of older records. The records are **purged periodically**.

DNS Records

33

- DNS servers store information in **Resource Records**
- A Records provide hostname-to-ip mapping
- Resource Record is a four-tuple
 - (Name, **Value**, Type, TTL)
- TTL
 - Determines when a resource record should be removed from the cache
- The meaning of Name and Value depend on Type

Type Explained

34

- Type=A
 - Name- Hostname
 - Value – IP Address
 - Provides standard hostname to IP Address Mapping
 - (sites.google.com, **142.23.23.1**, A, 200)
- Type=NS
 - Name – Domain
 - Value – Hostname of an authoritative DNS server that may know the information about the domain
 - Provides routing the DNS query
 - (sites.google.com, dns.google.com, NS, 125)

Type Explained

35

- Type = CNAME
 - Name – Alias Domain Name
 - Value – Main Domain Name
 - Specifies that the domain name is an alias of another
 - This helps when running multiple services (like an FTP *and* a web server; each running on different ports) from a single IP address. Each service can then have its own entry in DNS.
 - (www.cs.mit.edu, star.cs.mit.edu, CNAME, 120)

Type Explained

36

- Type=MX
 - Name - Hostname
 - Value – The address of the Mail Server
 - A company can use the same aliased name for all its mail server and for one of its other server
 - (buet.ac.bd, mail.buet.ac.bd, MX, 125)

A and NS Record Explained

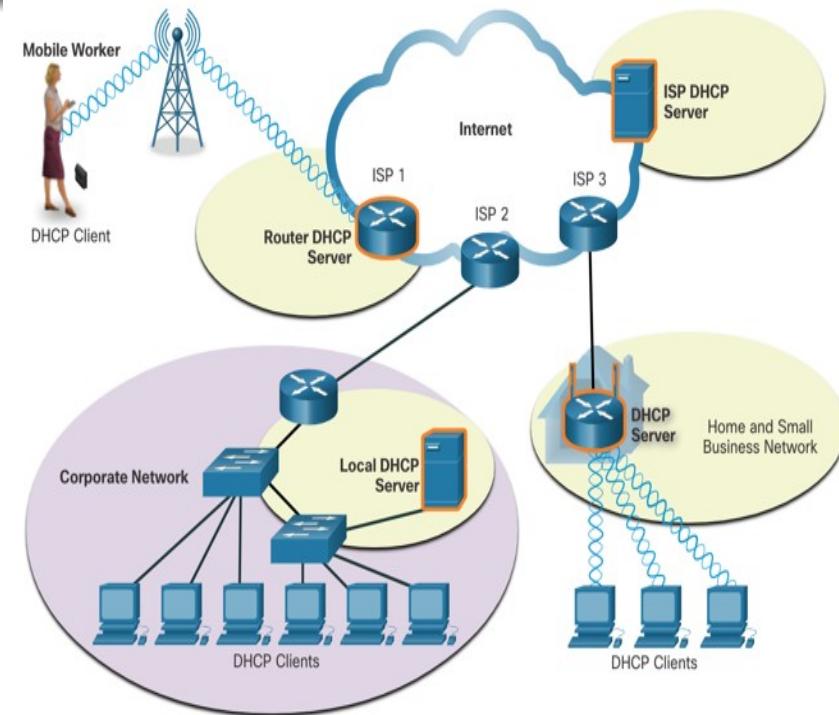
37

- If a DNS server is authoritative for a particular domain, it contains a Type A record
- If a server is not authoritative for a hostname, then it contains a type NS record for the domain that includes the hostname; it will also contain a Type A record that provides the ip address of the DNS server in the value field of the NS record
- TLD is not authoritative for cs.mit.edu
 - (mit.edu, dns.mit.edu, NS, 123)
 - (dns.mit.edu, 123.22.32.21, A, 134)

DHCP

Dynamic Host Configuration Protocol

- DHCP service automates the assignment of IPv4 addresses, subnet masks, gateways, and other IPv4 networking parameters.
- DHCP is considered dynamic addressing compared to static addressing. Static addressing is manually entering IP address information.



DHCP

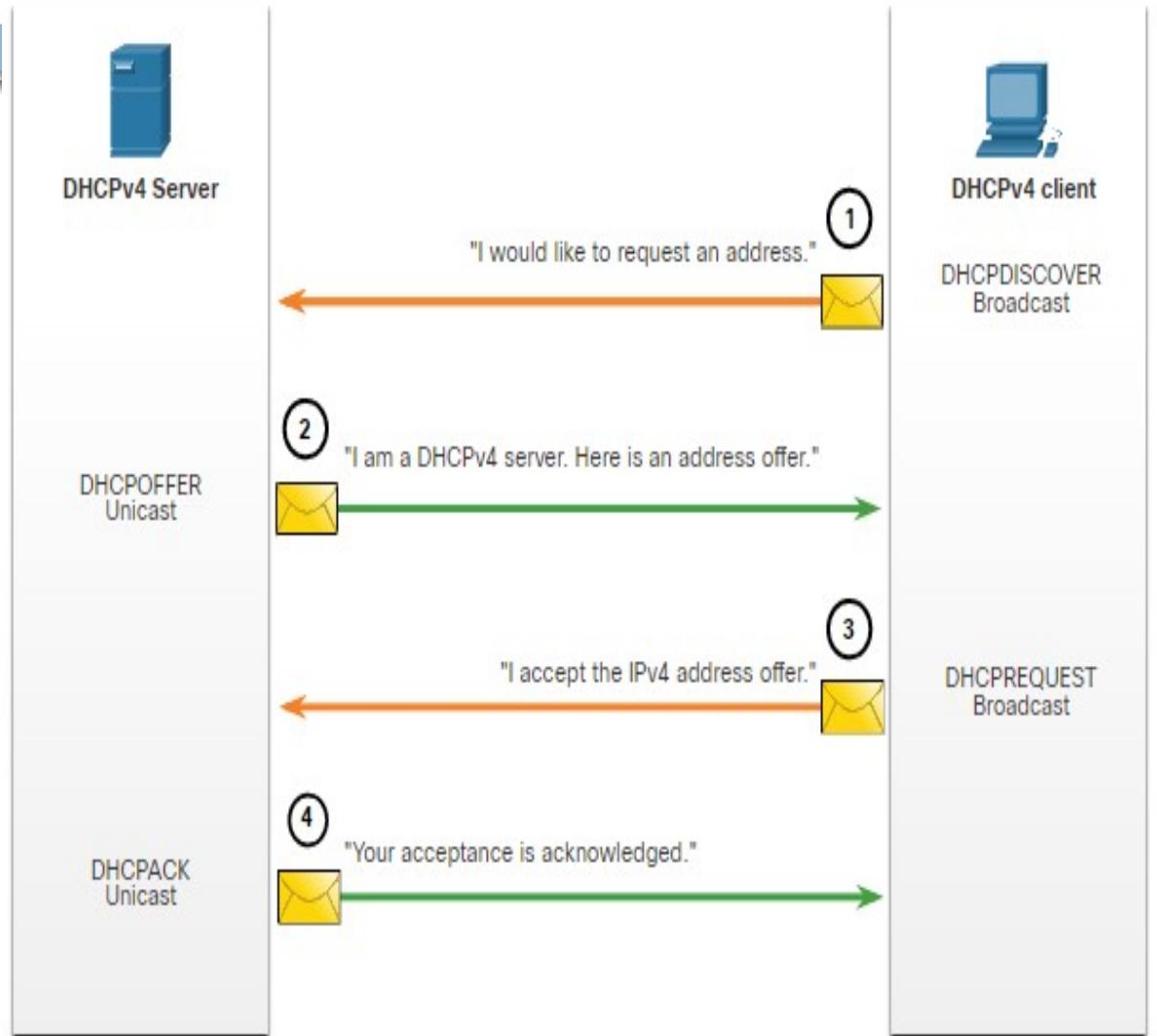
40

- When a host connects to the network, the DHCP server is contacted, and an address is requested. The DHCP server chooses an address from a configured range of addresses called a pool and assigns (leases) it to the host.
- Many networks use both DHCP and static addressing. DHCP is used for general purpose hosts, such as end user devices. Static addressing is used for network devices, such as gateway routers, switches, servers, and printers.

Steps to Obtain a Lease

When the client boots (or otherwise wants to join a network), it begins a four-step process to obtain a lease:

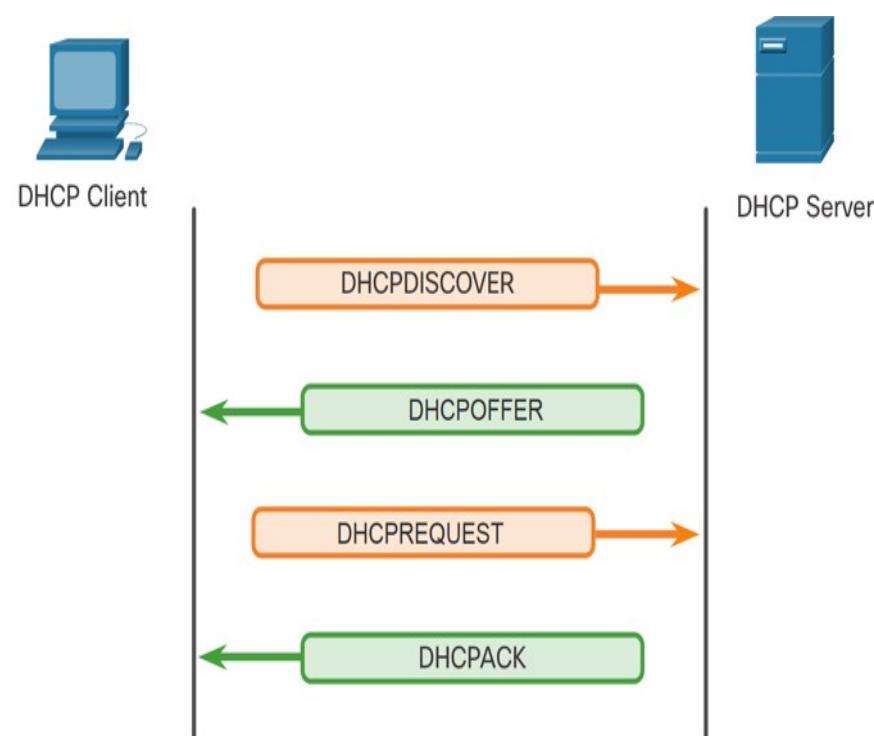
1. DHCP Discover (DHCPDISCOVER)
2. DHCP Offer (DHCPOFFER)
3. DHCP Request (DHCPREQUEST)
4. DHCP Acknowledgment (DHCPACK)



DHCP Operation

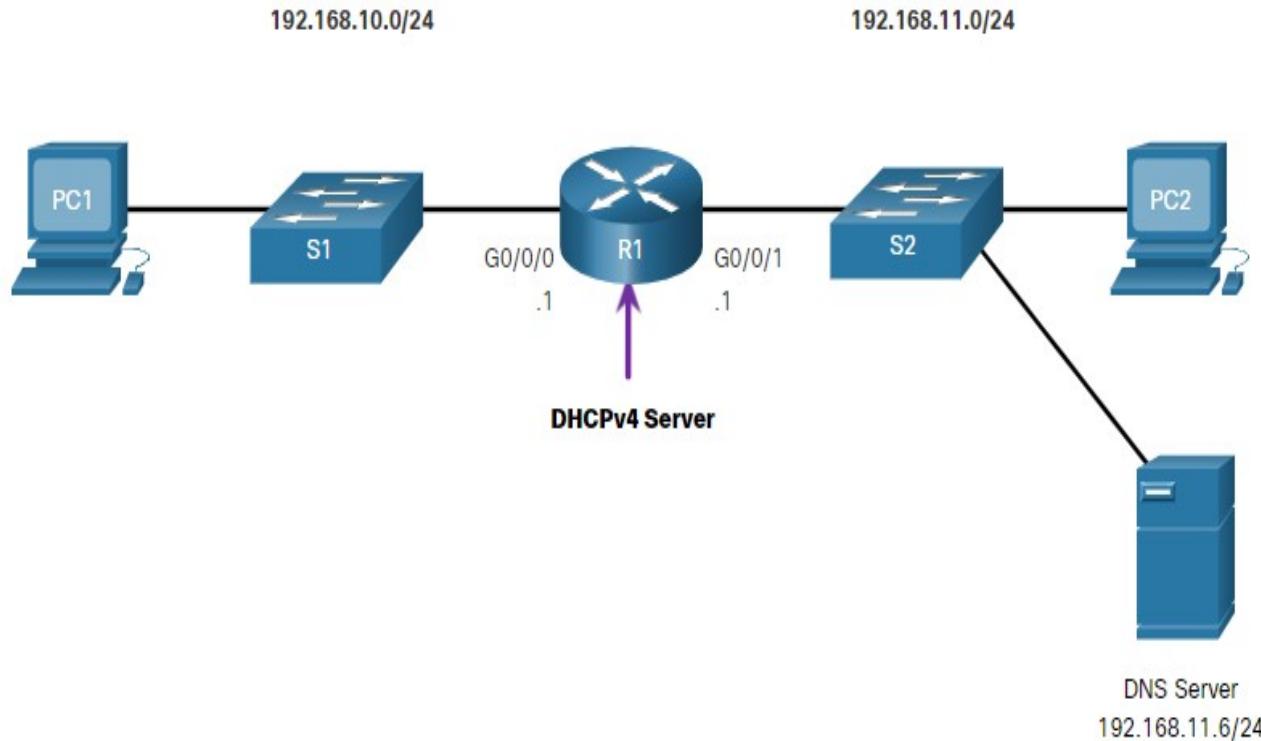
The DHCP Process:

- When an IPv4, DHCP-configured device boots up or connects to the network, the client broadcasts a DHCP discover (DHCPDISCOVER) message to identify any available DHCP servers on the network.
- A DHCP server replies with a DHCP offer (DHCPOFFER) message, which offers a lease to the client. (If a client receives more than one offer due to multiple DHCP servers on the network, it must choose one.)
- The client sends a DHCP request (DHCPREQUEST) message that identifies the explicit server and lease offer that the client is accepting.
- The server then returns a DHCP acknowledgment (DHCPACK) message that acknowledges to the client that the lease has been finalized.
- If the offer is no longer valid, then the selected server responds with a DHCP negative acknowledgment (DHCPNAK) message and the process must begin with a new DHCPDISCOVER message.



Configuring a Router as DHCPv4 Server

Now you have a basic understanding of how DHCPv4 works and how it can make your job a bit easier. A Cisco router running Cisco IOS software can be configured to act as a DHCPv4 server. The Cisco IOS DHCPv4 server assigns and manages IPv4 addresses from specified address pools within the router to DHCPv4 clients.

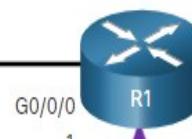


Steps to Configure a Cisco IOS DHCPv4 Server (Cont.)

Task	IOS Command
Define the address pool.	network <i>network-number [mask /prefix-length]</i>
Define the default router or gateway.	default-router <i>address [address2....address8]</i>
Define a DNS server.	dns-server <i>address [address2...address8]</i>
Define the domain name.	domain-name <i>domain</i>
Define the duration of the DHCP lease.	lease { <i>days [hours [minutes]]</i> infinite }
Define the NetBIOS WINS server.	netbios-name-server <i>address [address2...address8]</i>

Configuration Example

192.168.10.0/24



192.168.11.0/24



DNS Server
192.168.11.6/24

```
R1(config)# ip dhcp excluded-address 192.168.10.1 192.168.10.9
R1(config)# ip dhcp excluded-address 192.168.10.254
R1(config)# ip dhcp pool LAN-POOL-1
R1(dhcp-config)# network 192.168.10.0 255.255.255.0
R1(dhcp-config)# default-router 192.168.10.1
R1(dhcp-config)# dns-server 192.168.11.5
R1(dhcp-config)# domain-name example.com
R1(dhcp-config)# end
R1#
```

Configure a Cisco IOS DHCPv4 Server

DHCPv4 Verification

Use the commands in the table to verify that the Cisco IOS DHCPv4 server is operational.

Command	Description
show running-config section dhcp	Displays the DHCPv4 commands configured on the router.
show ip dhcp binding	Displays a list of all IPv4 address to MAC address bindings provided by the DHCPv4 service.
show ip dhcp server statistics	Displays count information regarding the number of DHCPv4 messages that have been sent and received

Verify DHCPv4 is Operational

Verify the DHCPv4 Configuration: As shown in the example, the **show running-config | section dhcp** command output displays the DHCPv4 commands configured on R1. The **| section** parameter displays only the commands associated with DHCPv4 configuration.

```
R1# show running-config | section dhcp
ip dhcp excluded-address 192.168.10.1 192.168.10.9
ip dhcp excluded-address 192.168.10.254
ip dhcp pool LAN-POOL-1
  network 192.168.10.0 255.255.255.0
  default-router 192.168.10.1
  dns-server 192.168.11.5
  domain-name example.com
```

Verify DHCPv4 is Operational (Cont.)

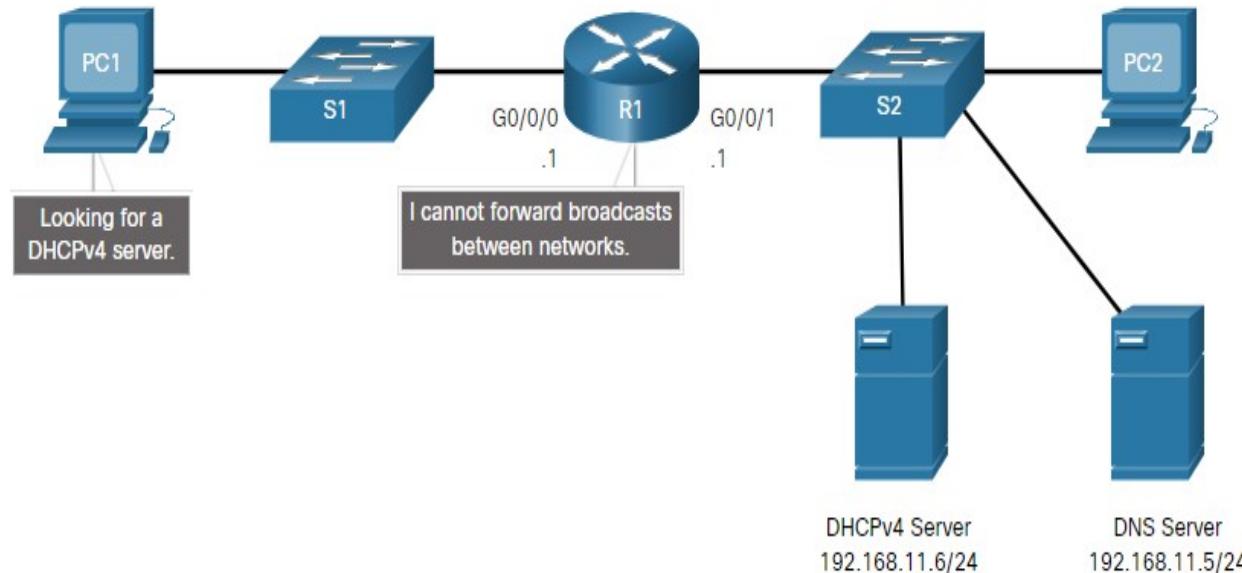
Verify DHCPv4 Bindings: As shown in the example, the operation of DHCPv4 can be verified using the **show ip dhcp binding** command. This command displays a list of all IPv4 address to MAC address bindings that have been provided by the DHCPv4 service.

```
R1# show ip dhcp binding
Bindings from all pools not associated with VRF:
IP address      Client-ID/          Lease expiration      Type      State      Interface
                  Hardware address/
                  User name
192.168.10.10   0100.5056.b3ed.d8    Sep 15 2019 8:42 AM  Automatic  Active
GigabitEthernet0/0/0
```

Configure a Cisco IOS DHCPv4 Server

DHCPv4 Relay

- In a complex hierarchical network, enterprise servers are usually located centrally. These servers may provide DHCP, DNS, TFTP, and FTP services for the network. Network clients are not typically on the same subnet as those servers. In order to locate the servers and receive services, clients often use broadcast messages.
- In the figure, PC1 is attempting to acquire an IPv4 address from a DHCPv4 server using a broadcast message. In this scenario, R1 is not configured as a DHCPv4 server and does not forward the broadcast. Because the DHCPv4 server is located on a different network, PC1 cannot receive an IP address using DHCP. R1 must be configured to relay DHCPv4 messages to the DHCPv4 server.



DHCPv4 Relay Configuration

- Configure R1 with the **ip helper-address address** interface configuration command. This will cause R1 to relay DHCPv4 broadcasts to the DHCPv4 server. As shown in the example, the interface on R1 receiving the broadcast from PC1 is configured to relay DHCPv4 address to the DHCPv4 server at 192.168.11.6.
- When R1 has been configured as a DHCPv4 relay agent, it accepts broadcast requests for the DHCPv4 service and then forwards those requests as a unicast to the IPv4 address 192.168.11.6. The network administrator can use the **show ip interface** command to verify the configuration.

```
R1(config)# interface g0/0/0
R1(config-if)# ip helper-address 192.168.11.6
R1(config-if)# end
R1#
```

```
R1# show ip interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Internet address is 192.168.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is 192.168.11.6
  (output omitted)
```

HTTP

HTTP

52

- Stands for **Hyper Text Transfer Protocol**
- Uses the services of TCP on well-known port 80
- HTTP messages are read and interpreted by the HTTP server and HTTP client(browser)
- HTTP is a stateless protocol as HTTP server does not store any information for the clients
- Uses TCP

URL

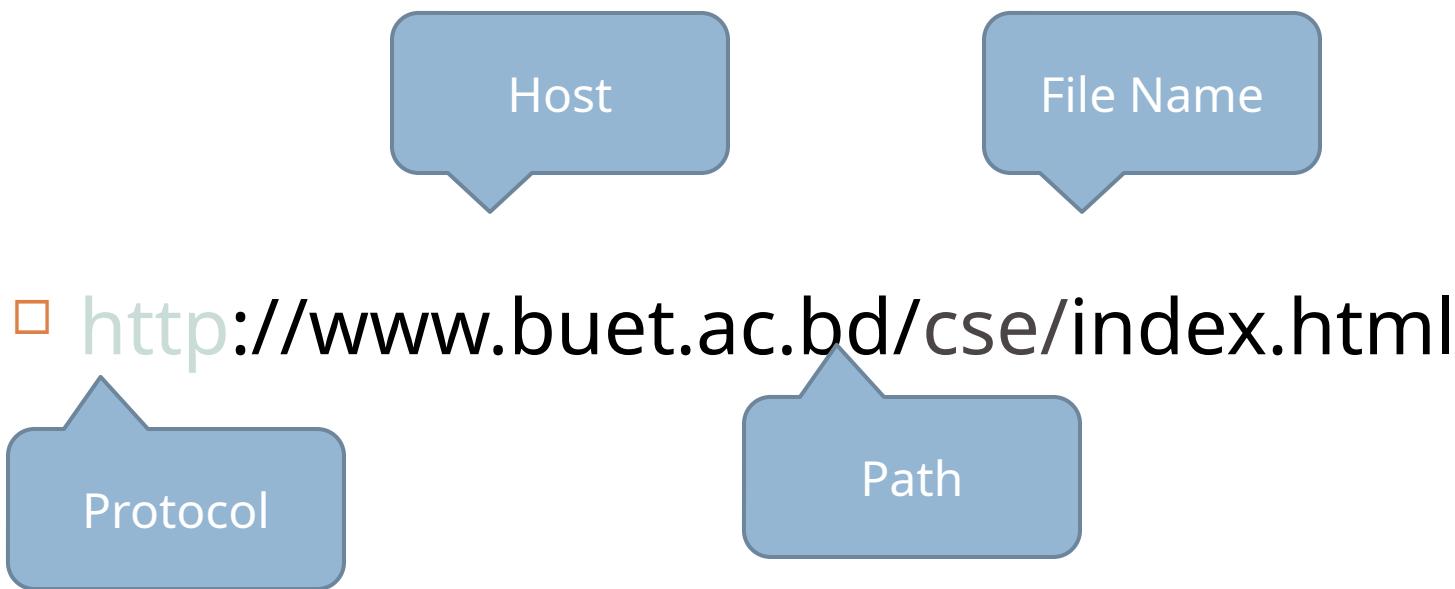
- Abbreviated Form of
 - Uniform Resource Locator
- Components
 - Protocol
 - Host
 - Port (**Optional**)
 - Path
 - File Name

URL (Explained)

54

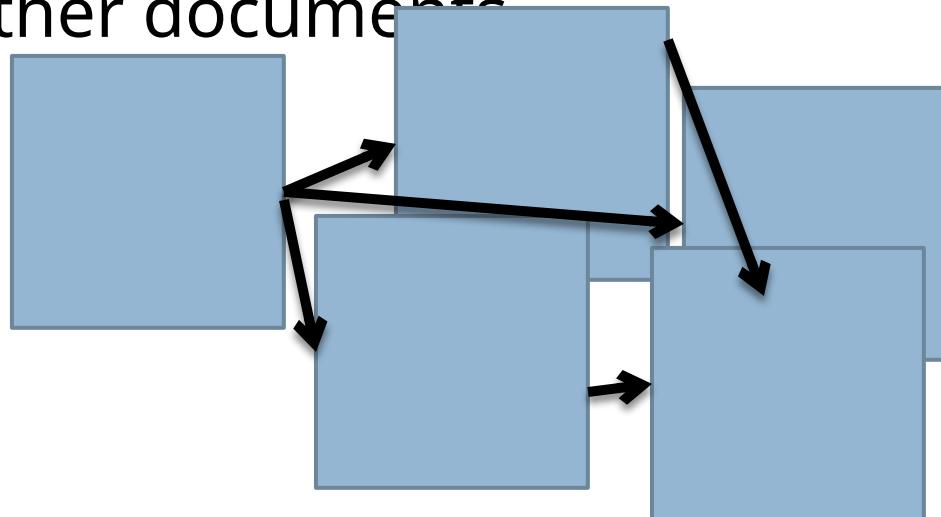
- Protocol
 - The client/server program used to retrieve the document
- Host
 - The computer on which the information is located
- Port
 - Optionally contains the port number of the server
- Path
 - The pathname of the file where the information is located
 - Slashes separates the directories

URL (Explained)



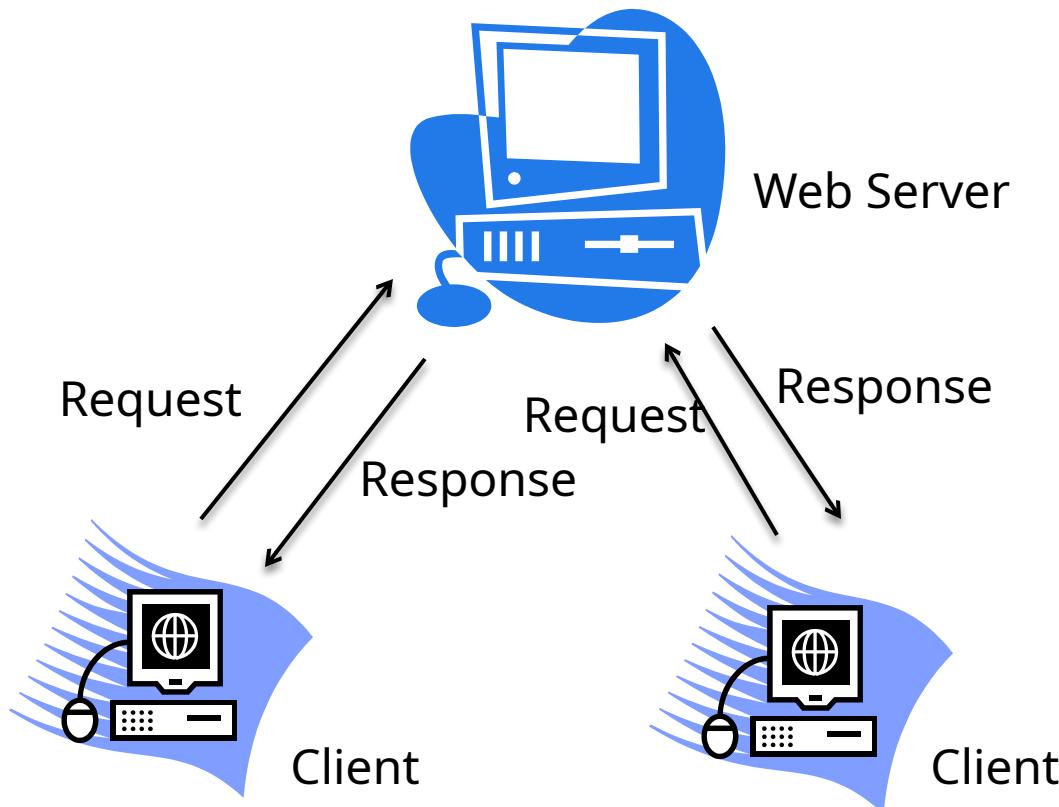
Hypertext

- Obviously, Hypertext file is a Collection of Text, like Document File.
- What is Extra?
 - Certain words within the text are marked as *links* to other areas of the current document or to other documents.



HTTP Communication

57



Non-persistent Connection

58

- One TCP connection is made for each request/response
- Strategy
 - The client opens a TCP connection and sends a request
 - The server sends the response and closes the connection
 - The client reads the data until it encounters an end-of-file marker; it then closes the connection
- HTTP prior to version 1.1 specified non-persistent connection
- Disadvantages
 - High overhead for server to maintain separate connections for all separate requests between a specific host.
 - Slow start

Persistent Connection

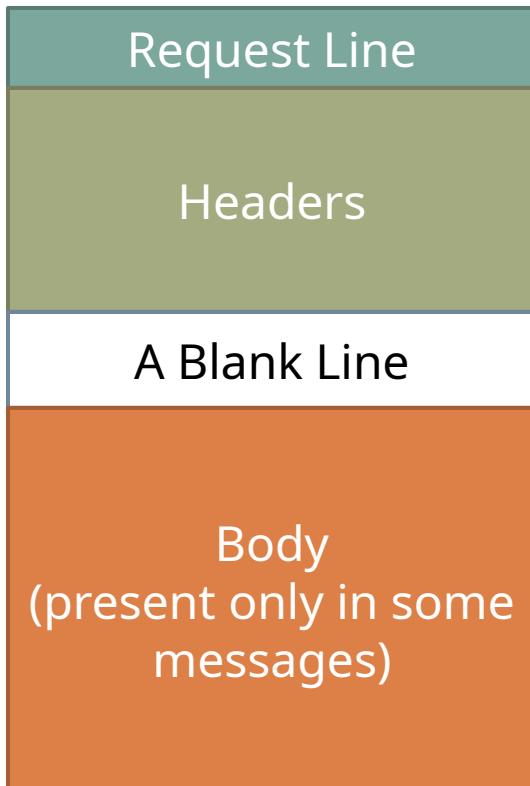
59

- Server leaves the connection open for more requests after sending a response
- The server can close the connection at the request of a client or if a time-out has been reached
- HTTP version 1.1 specifies persistent connection as default

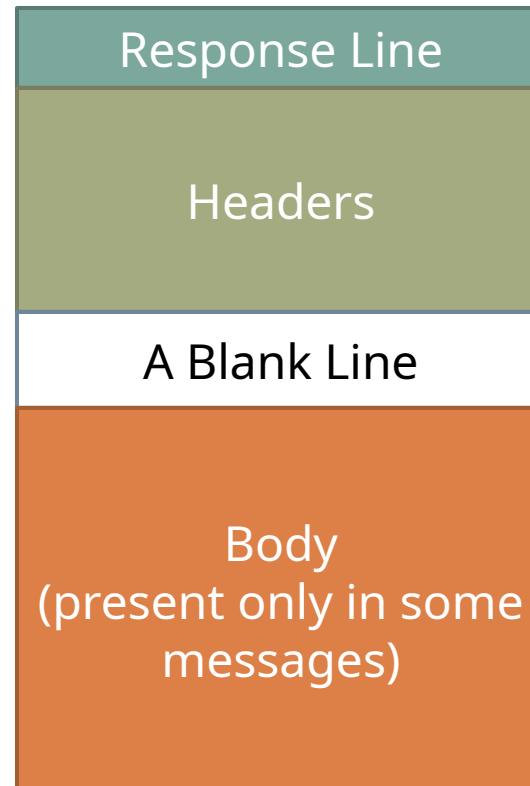
HTTP Message Format

60

HTTP Request



HTTP Response



Request Line

61

Request
Type

URL

HTTP
Version

- Request Type- Identifies the request type
- URL – The URL of the requested object

Method	Action
GET	Requests a document from the server
HEAD	Requests information about a document in the server, but not the document
POST	Sends some information from the client to server
PUT	Sends a document from the server to client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	For client to check availability of resource

Status Line

62

HTTP Version			Status Code	Status Phrase
Code	Phrase	Description		
100	Continue	The initial part of the request is received, and the client may continue its request		
200	OK	The request is successful		
202	Accepted	The request is accepted, but it is not immediately acted upon		
204	No content	There is no content in the body		
400	Bad Request	There is a syntax error in the request		
401	Unauthorized	The request lacks authorization		
404	Not found	The document is not found		
500	Internal Server Error	There is error, such a crash, at the server side		

HTTP Headers

63

Header Name:

Value

- Three types of Headers
 - General Header
 - Gives general information about the message
 - Can be present in both response and request
 - Request Header
 - Present only in the request message
 - Specifies the client configuration and client's preferred document format
 - Response Header
 - Present only in the response message
 - Provides server configuration and special information about the request

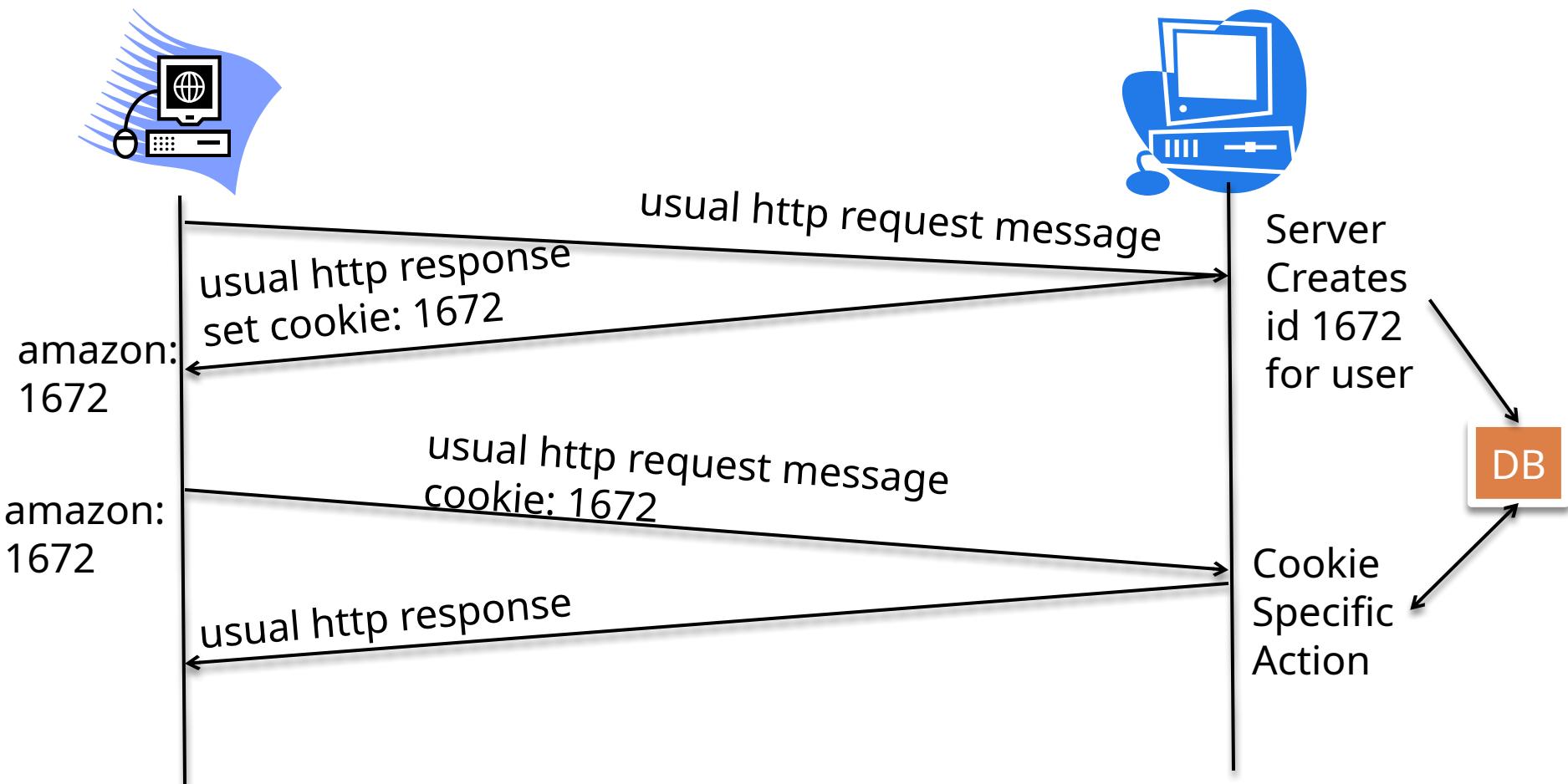
Cookies

64

- HTTP server is stateless
 - But identification is useful restricting access, personalized content
- Components
 - Cookie header line in HTTP response message
 - Cookie header line in HTTP request message
- Cookie file kept on user's host and managed by user's browser
- Back-end database at Web site

Cookies Explained

65



Advantages & Disadvantages

66

□ Advantages

- Authorization, shopping carts
recommendation, user session state

□ Disadvantages

- Websites learns a lot about user

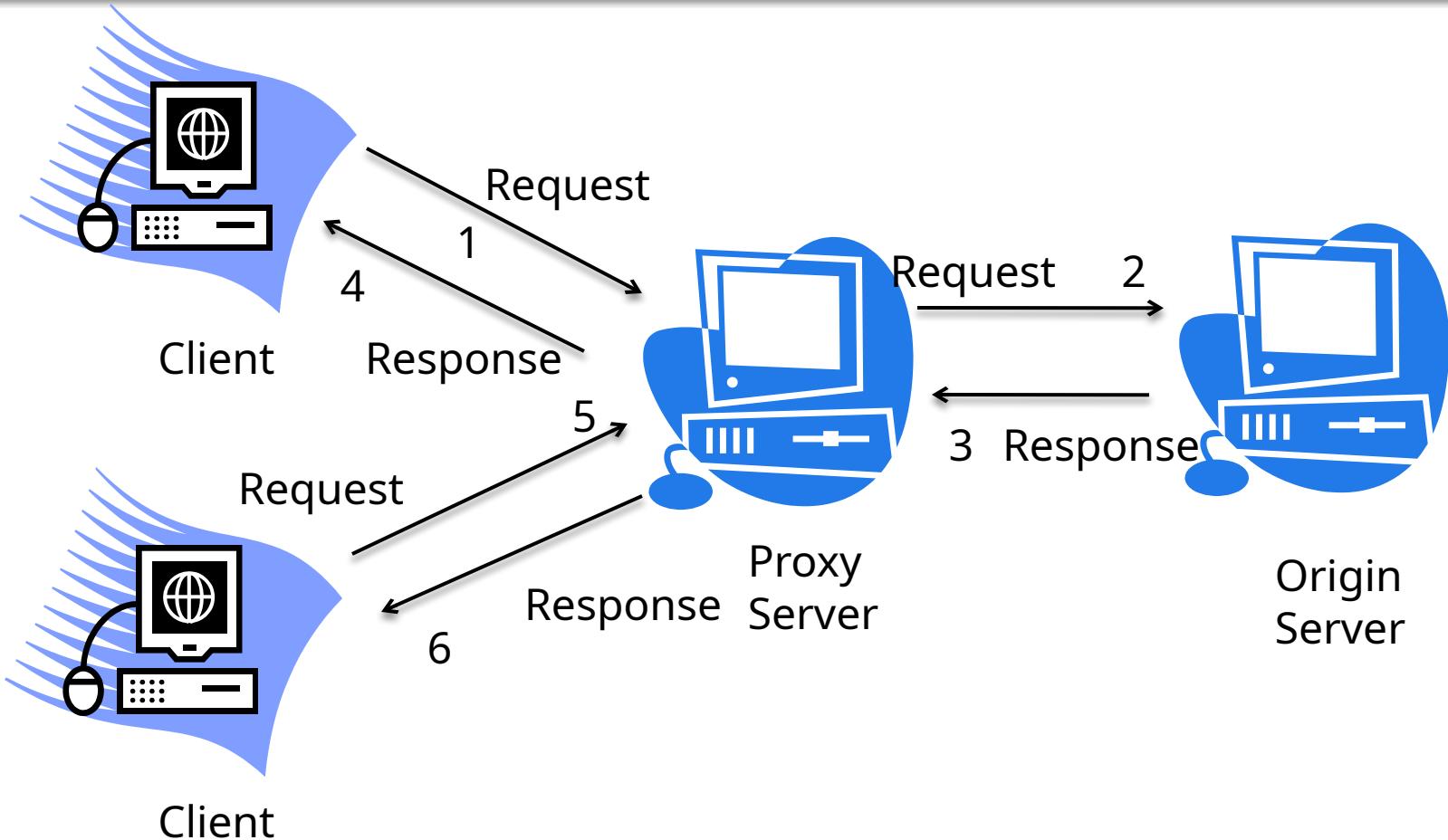
Proxy Server

67

- User sets browser: all Web accesses via proxy
- Browser sends all HTTP requests to proxy
- If object in proxy, proxy returns the object
- If the object is not there, proxy requests object from origin server, returns object to client, stores
- Reduces
 - Delay
 - Network Traffic

Proxy Server Explained

68

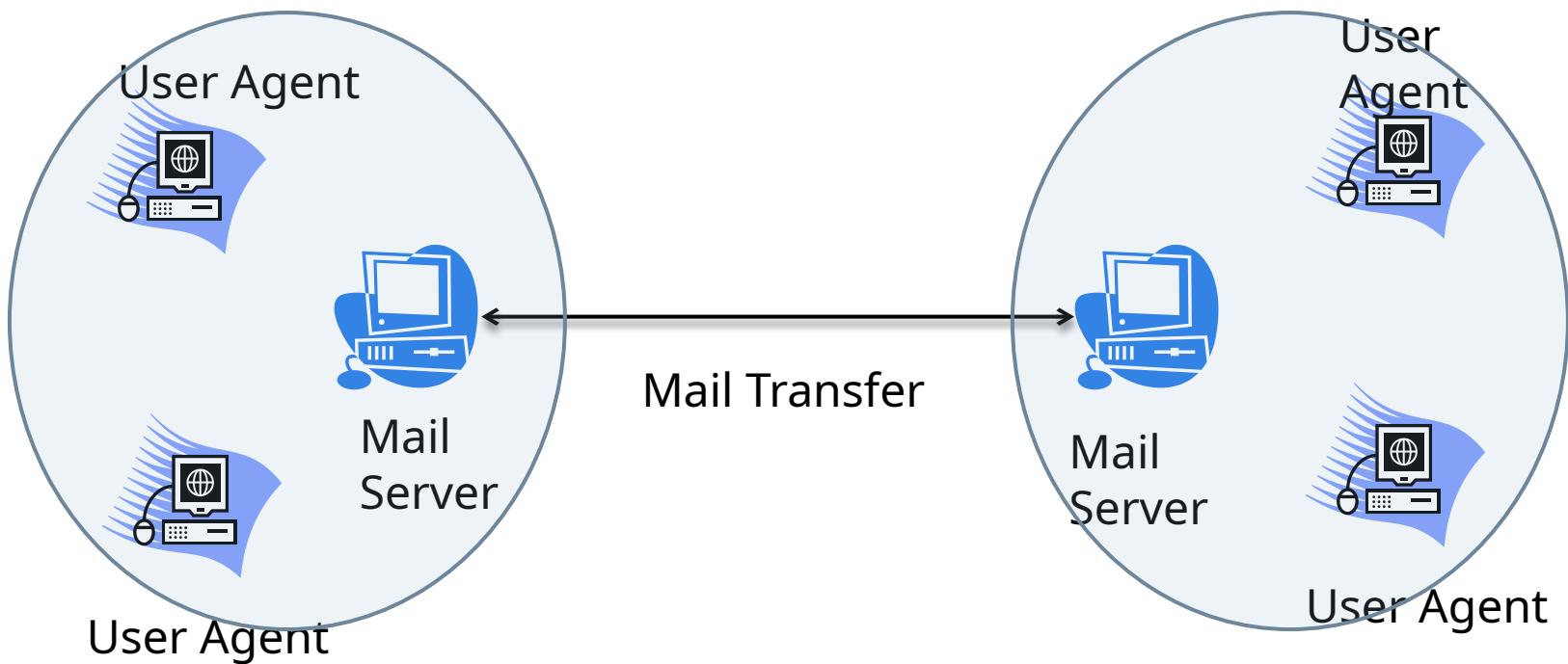


Electronic Mail

Architecture

70

- System comprises of
 - User Agents
 - Mail Servers



User Agent

71

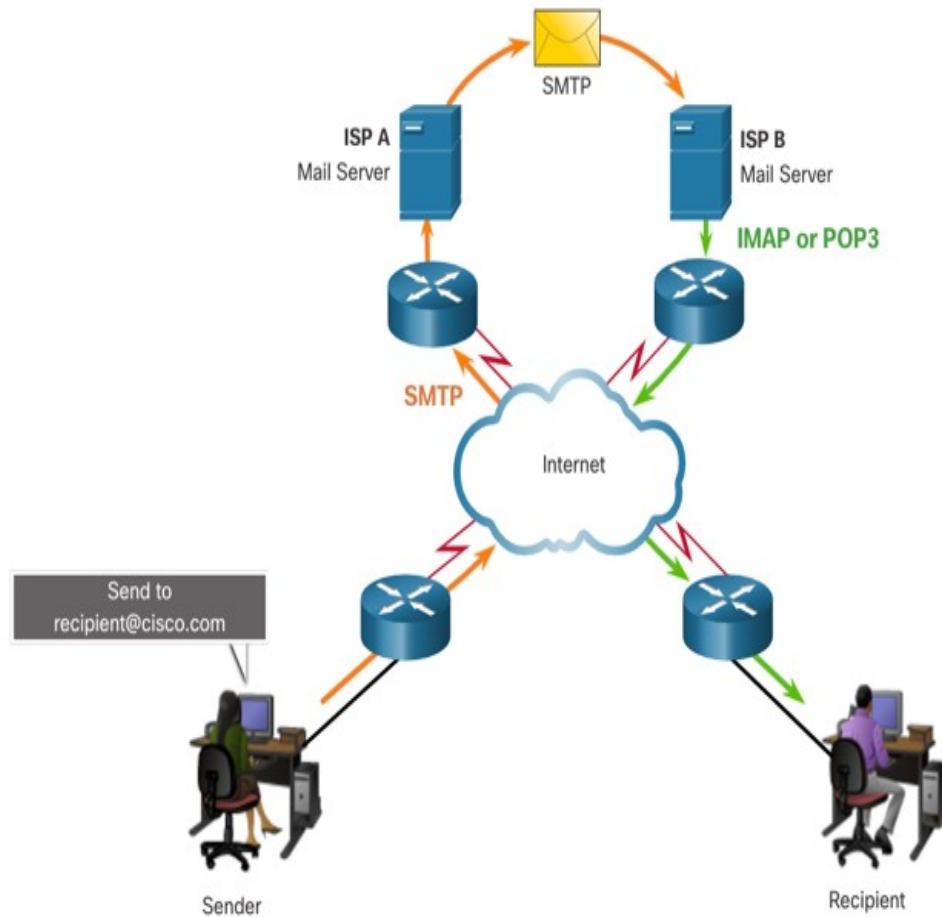
- User Agent
 - Mail Client
 - Purpose - Composing, editing, reading mail messages
 - Eudora, Outlook
- Mail Server
 - Mailbox contains incoming messages for user
 - Message queue of outgoing mail messages
 - SMTP protocol between mail servers to send email messages

Email Protocols

Email is a store-and-forward method of sending, storing, and retrieving electronic messages across a network. Email messages are stored in databases on mail servers. Email clients communicate with mail servers to send and receive email.

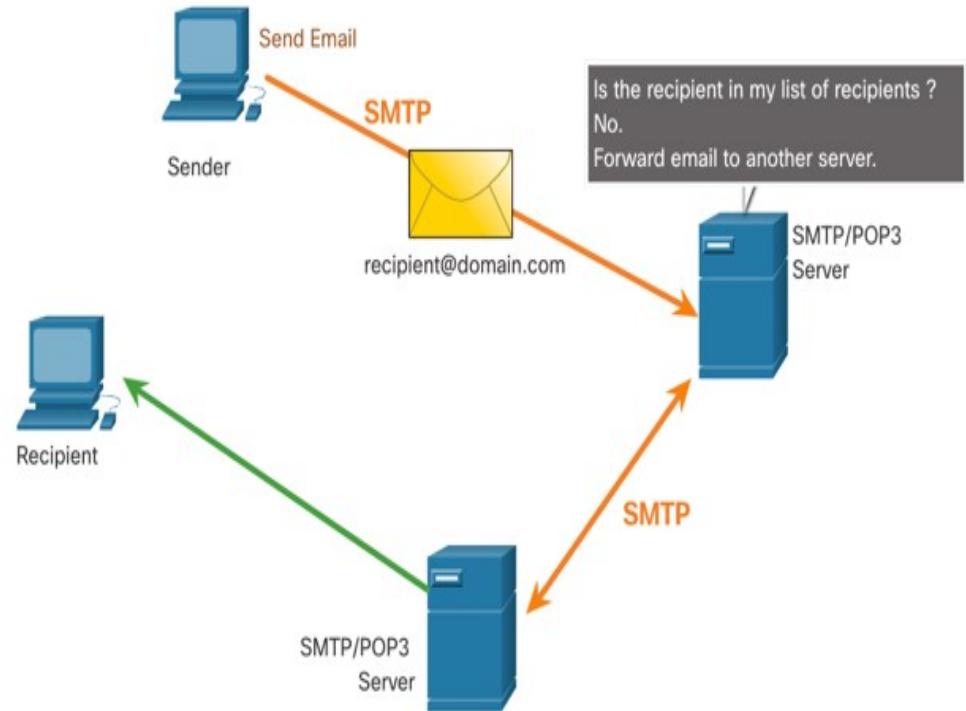
The email protocols used for operation are:

- Simple Mail Transfer Protocol (SMTP) – used to send mail.
- Post Office Protocol (POP) & IMAP – used for clients to receive mail.



SMTP

- When a client sends email, the client **SMTP process connects with a server SMTP process** on well-known port 25.
- After the connection is made, the client attempts to send the email to the server across the connection.
- When the server receives the message, it either places the message in a local account, if the recipient is local, or forwards the message to another mail server for delivery.
- The destination email server may not be online or may be busy. If so, SMTP spools messages to be sent at a later time.



Note: SMTP message formats require a message header (recipient email address & sender email address) and a message body.

Mail Routing

74

- Sender uses UA to compose message and send
- Sender's UA sends message to her mail server; message placed in message queue
- Client side of SMTP opens TCP connection with receiver's mail server
- SMTP client sends sender's message over the TCP connection
- Receiver's mail server places the message in receiver's mailbox
- Receiver invokes his user agent to read message

SMTP

75

- Simple Mail Transfer Protocol
- A simple ASCHII protocol
- After establishing a TCP connection to port 25, the sending machine(client) waits for the receiving machine(server)
- The server starts by sending a line of text giving its identity and telling whether it is prepared to receive mail
- If the server is not ready, the client releases the connection and tries again

SMTP

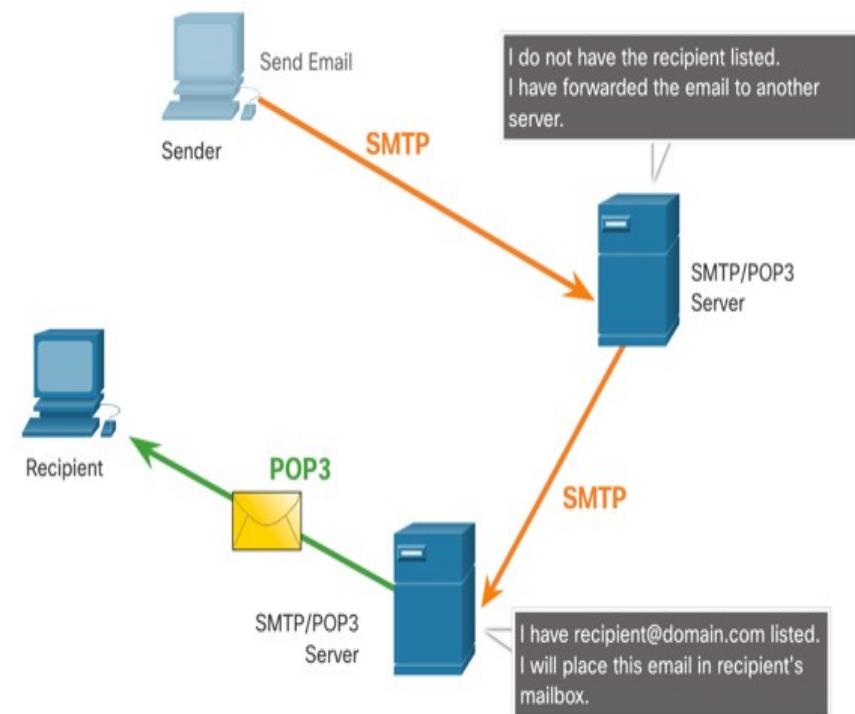
76

- If the server is willing to accept e-mail, the client announces whom the e-mail is coming from and whom it is going to.
- If such recipients exists at the destination, the server gives the client to go-ahead to send the message
- Then the client sends the message and server acknowledges it.

POP

POP is used by an application to retrieve mail from a mail server. When mail is downloaded from the server to the client using POP the messages are then deleted on the server.

- The server starts the POP service by passively listening on TCP port 110 for client connection requests.
- When a client wants to make use of the service, it sends a request to establish a TCP connection with the server.
- When the connection is established, the POP server sends a greeting.
- The client and POP server then exchange commands and responses until the connection is closed or aborted.



Note: Since POP **does not store messages**, it is **not recommended** for small businesses that need a centralized backup solution.

POP3

78

- Post Office 3 Protocol
- Begins when the UA start their mail reader
- The mail reader calls up the mail server and establishes a TCP connection with the mail transfer agent at port 110
- Once the connection is established, POP3 goes through 3 states
 - Authorization
 Authorize the user
 - Transactions
 - Collect mail and mark them for deletion
 - Update
 - Causes mails to be deleted

IMAP

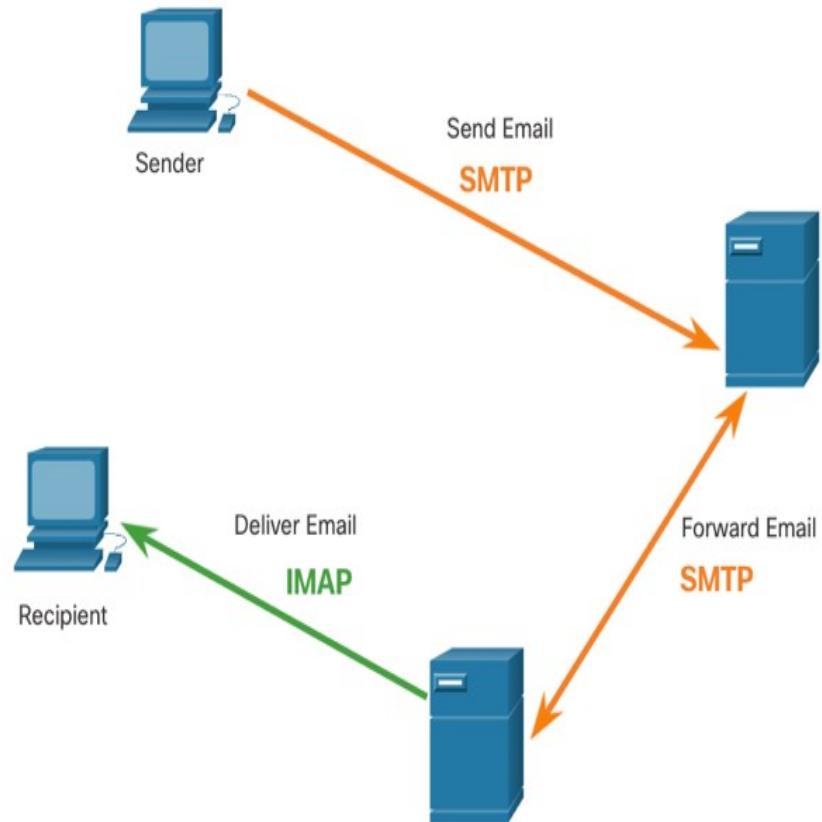
79

- Internet Message Access Protocol
- Assumes all **e-mail will remain on the server** indefinitely In multipart mailboxes
- Provides mechanism for reading messages or even part of messages
- Provides mechanism of creating, destroying and manipulating multiple mailboxes on the server

IMAP

IMAP is another protocol that describes a method to retrieve email messages.

- Unlike POP, when a user connects to an IMAP server, copies of the messages are downloaded to the client application. The original messages are kept on the server until manually deleted.
- When a user decides to delete a message, the server synchronizes that action and deletes the message from the server.



MIME

81

- Multipurpose Internet Mail Extensions
- Supplementary protocol to allow non-ASCII data to be sent through e-mail
- Different types of contents are supported by MIME

Type	Subtype	Description
Text	Plain	Unformatted
	HTML	HTML format
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but not ordered
Image	JPEG	Image in JPEG
	GIF	Image in GIF
Video	MPEG	Video in MPEG
Audio	Basic	Single-channel encoding of voice at 8kHz
Application	Postscript	Adobe Postscript
	Octet-	General binary data (8-bit bytes)

IPv6

Source: Cisco networking

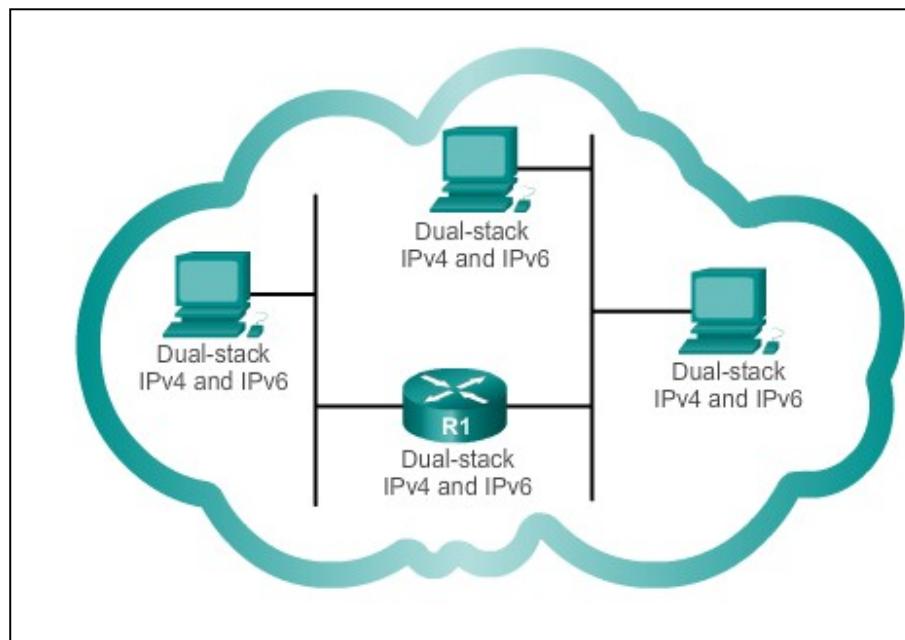
The Need for IPv6

- IPv6 is designed to be the successor to IPv4.
- Depletion of IPv4 address space has been the motivating factor for moving to IPv6.
- Projections show that all five RIRs will run out of IPv4 addresses between 2015 and 2020.
- With an increasing Internet population, a limited IPv4 address space, issues with NAT and an Internet of things, the time has come to begin the transition to IPv6!
- IPv4 has a theoretical maximum of 4.3 billion addresses, plus private addresses in combination with NAT.
- IPv6 larger 128-bit address space provides **for 340 undecillion addresses.**
- IPv6 fixes the limitations of IPv4 and includes additional enhancements, such as ICMPv6.

IPv4 and IPv6 Coexistence

The migration techniques can be divided into three categories:
Dual-stack, Tunnelling, and Translation.

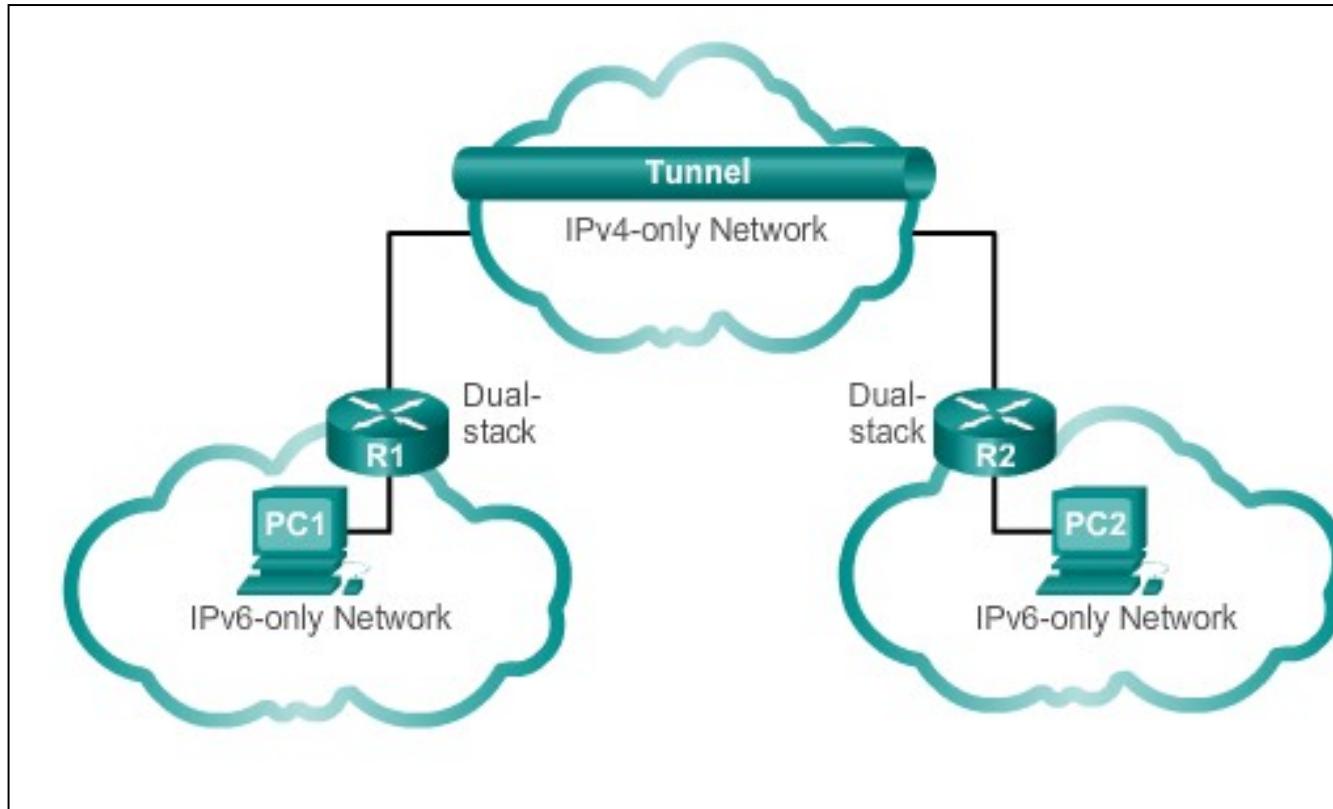
Dual-stack



Dual-stack: Allows IPv4 and IPv6 to coexist on the same network.
Devices run both IPv4 and IPv6 protocol stacks simultaneously.

IPv4 and IPv6 Coexistence (cont.)

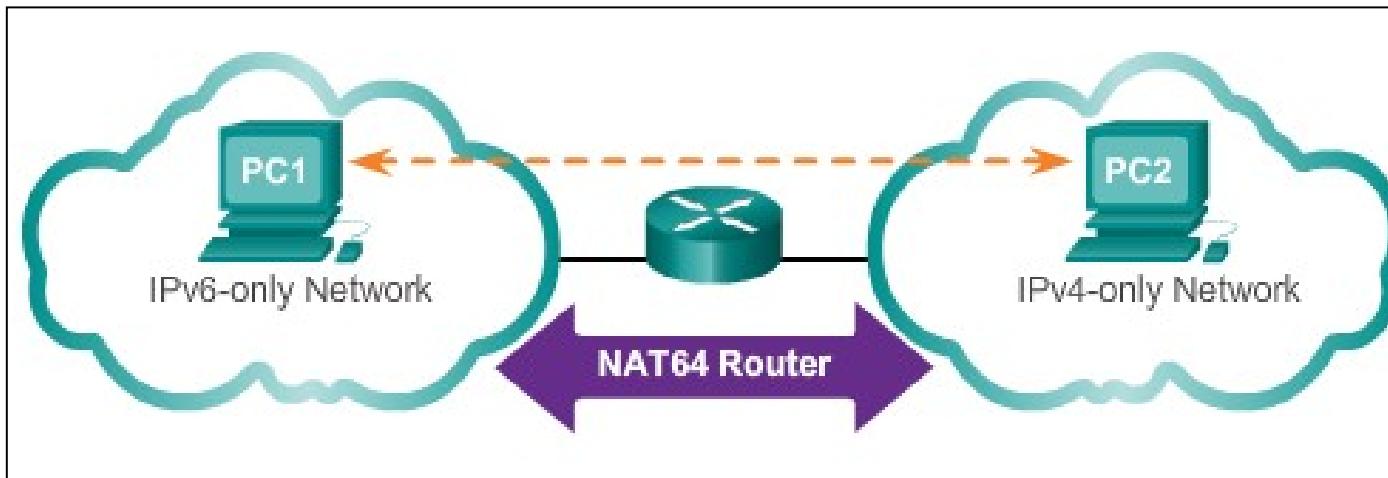
Tunnelling



Tunnelling: A method of transporting an IPv6 packet over an IPv4 network. The IPv6 packet is encapsulated inside an IPv4 packet.

IPv4 and IPv6 Coexistence (cont.)

Translation



Translation: The Network Address Translation 64 (NAT64) allows IPv6-enabled devices to communicate with IPv4-enabled devices using a translation technique similar to NAT for IPv4. An IPv6 packet is translated to an IPv4 packet, and vice versa.

IPv6 Address Representation

- **128 bits** in length and written as a string of hexadecimal values
- In IPv6, 4 bits represents a single hexadecimal digit, 32 hexadecimal value = IPv6 address

2001:0DB8:0000:1111:0000:0000:0000:0200

FE80:0000:0000:0000:0123:4567:89AB:CDEF

- Hextet used to refer to a segment of 16 bits or four hexadecimals
- Can be written in either lowercase or uppercase

IPv6 Addressing

Rule 1- Omitting Leading 0s

- The first rule to help reduce the notation of IPv6 addresses is any leading 0s (zeros) in any 16-bit section or hextet can be omitted.
- 01AB can be represented as 1AB.
- 09F0 can be represented as 9F0.
- 0A00 can be represented as A00.
- 00AB can be represented as AB.

Preferred	2001: 0 DB8: 000 A: 1000 : 000 0: 000 0: 0000 : 0100
No leading <u>0s</u>	2001: DB8: A:1000: 0: 0: 0: 100
Compressed	2001:DB8:A:1000:0:0:0:100

Rule 2 - Omitting All 0 Segments

- A double colon (::) can replace any single, contiguous string of one or more 16-bit segments (hextets) consisting of all 0's.
- **Double colon (::)** can only be used once within an address otherwise the address will be ambiguous.
- Known as the *compressed format*.
- Incorrect address - 2001:0DB8::ABCD::1234.

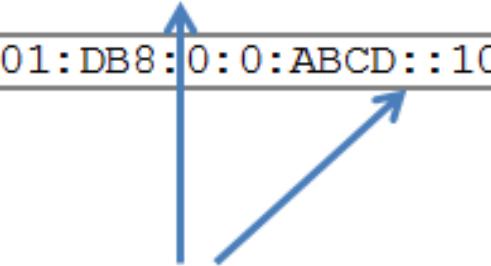
IPv6 Addressing

Rule 2 - Omitting All 0 Segments (cont.)

Example #1

Preferred	2001: 0 DB8: 0000 : 0000 :ABCD: 0000 : 0000 : 0100
Omit leading 0s	2001: DB8: 0 : 0 :ABCD: 0 : 0 : 100
Compressed	2001:DB8::ABCD:0:0:100
OR	
Compressed	2001:DB8: 0 : 0 :ABCD:: 100

Only one :: may be used.



Example #2

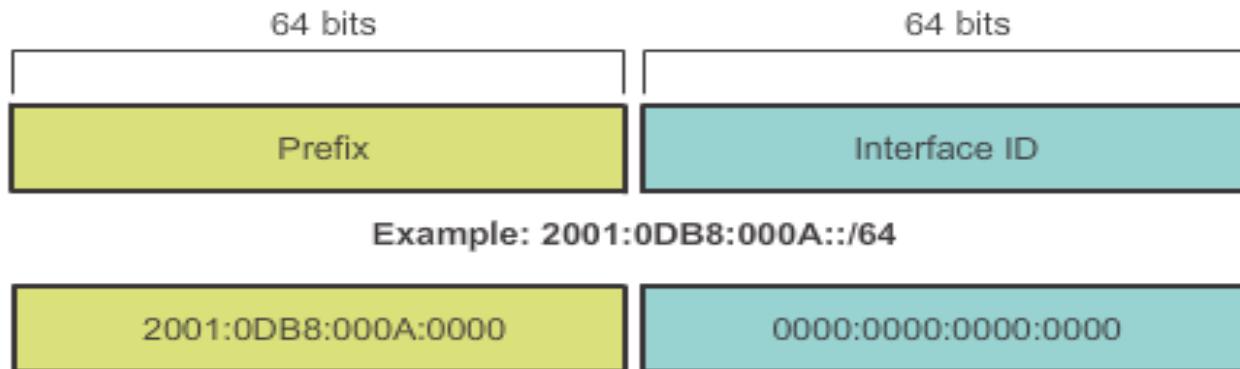
Preferred	FE80: 0000 : 0000 : 0000 : 0123 :4567:89AB:CDEF
Omit leading 0s	FE80: 0 : 0 : 0 : 123 :4567:89AB:CDEF
Compressed	FE80: : 123 :4567:89AB:CDEF

Types of IPv6 Addresses

IPv6 Prefix Length

- IPv6 does not use the dotted-decimal subnet mask notation
- Prefix length indicates the network portion of an IPv6 address using the following format:
 - IPv6 address/prefix length
 - Prefix length can range from 0 to 128
 - Typical prefix length is /64

/64 Prefix



Types of IPv6 Addresses

IPv6 Address Types

There are three types of IPv6 addresses:

- Unicast
- Multicast
- Anycast.

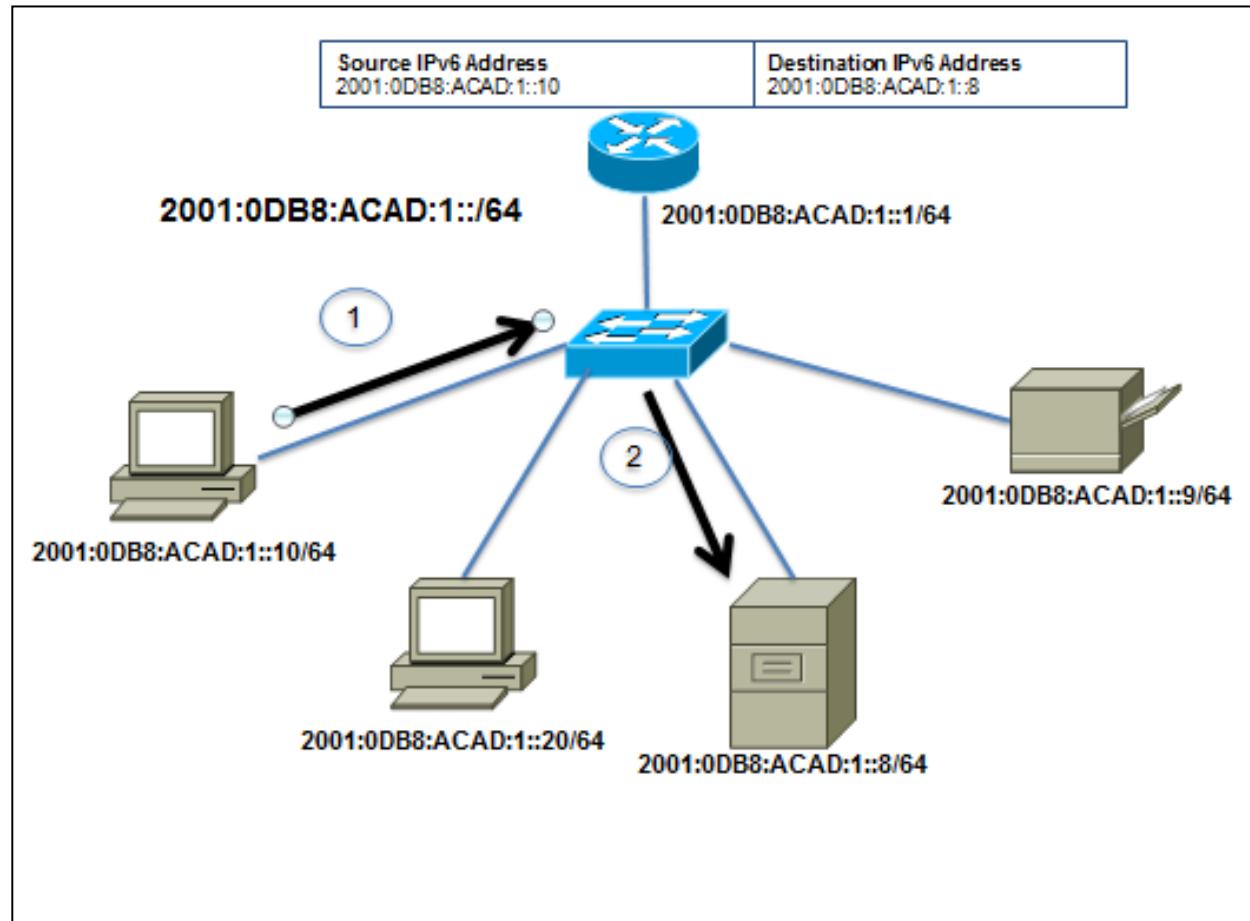
Note: IPv6 does not have broadcast addresses.

Types of IPv6 Addresses

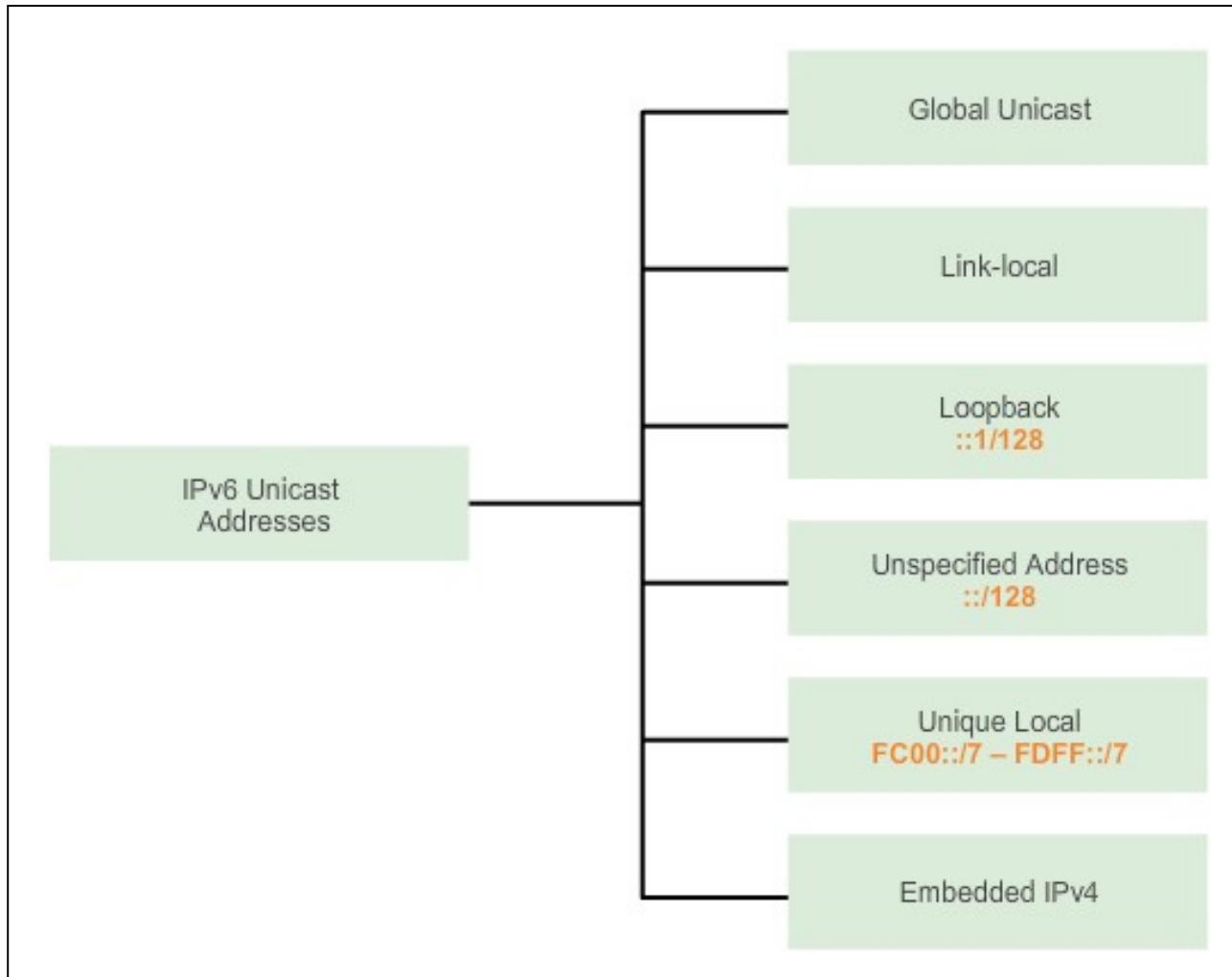
IPv6 Unicast Addresses

Unicast

- Uniquely identifies an interface on an IPv6-enabled device.
- A packet sent to a unicast address is received by the interface that is assigned that address.



IPv6 Unicast Addresses (cont.)



IPv6 Unicast Addresses (cont.)

Global Unicast

- Similar to a public IPv4 address
- Globally unique
- Internet routable addresses
- Can be configured statically or assigned dynamically

Link-local

- Used to communicate with other devices on the same local link
- Confined to a single link; not routable beyond the link

IPv6 Unicast Addresses (cont.)

Loopback

- Used by a host to send a packet to itself and cannot be assigned to a physical interface.
- Ping an IPv6 loopback address to test the configuration of TCP/IP on the local host.
- All-0s except for the last bit, represented as ::1/128 or just ::1.

Unspecified Address

- All-0's address represented as ::/128 or just ::
- Cannot be assigned to an interface and is only used as a source address.
- An unspecified address is used as a source address when the device does not yet have a permanent IPv6 address or when the source of the packet is irrelevant to the destination.

IPv6 Unicast Addresses

Unique Local

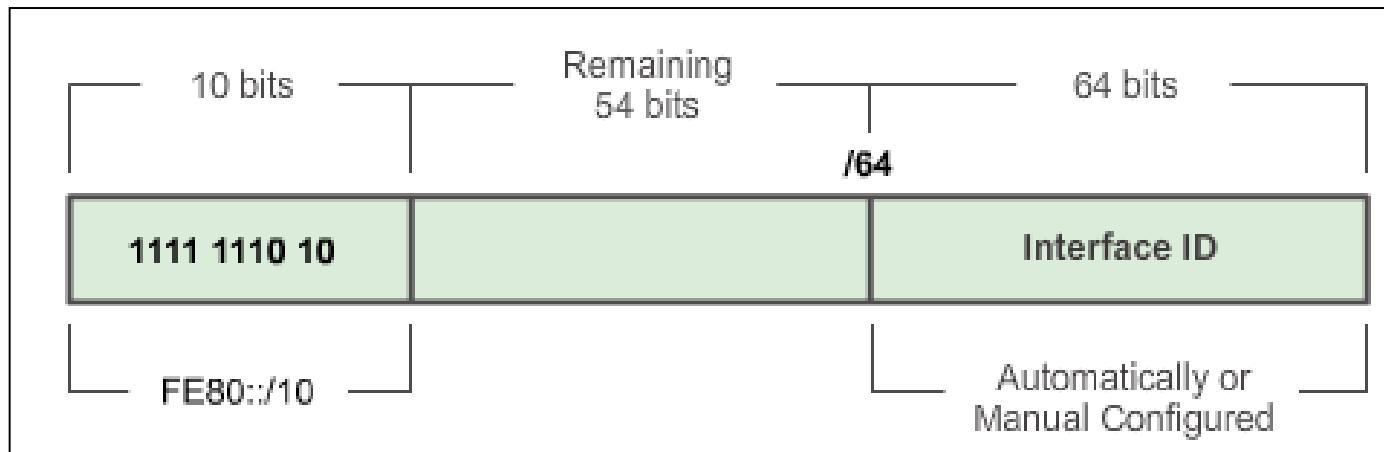
- Similar to private addresses for IPv4.
- Used for local addressing within a site or between a limited number of sites.
- In the range of FC00::/7 to FDFF::/7.

IPv4 Embedded

- Used to help transition from IPv4 to IPv6.

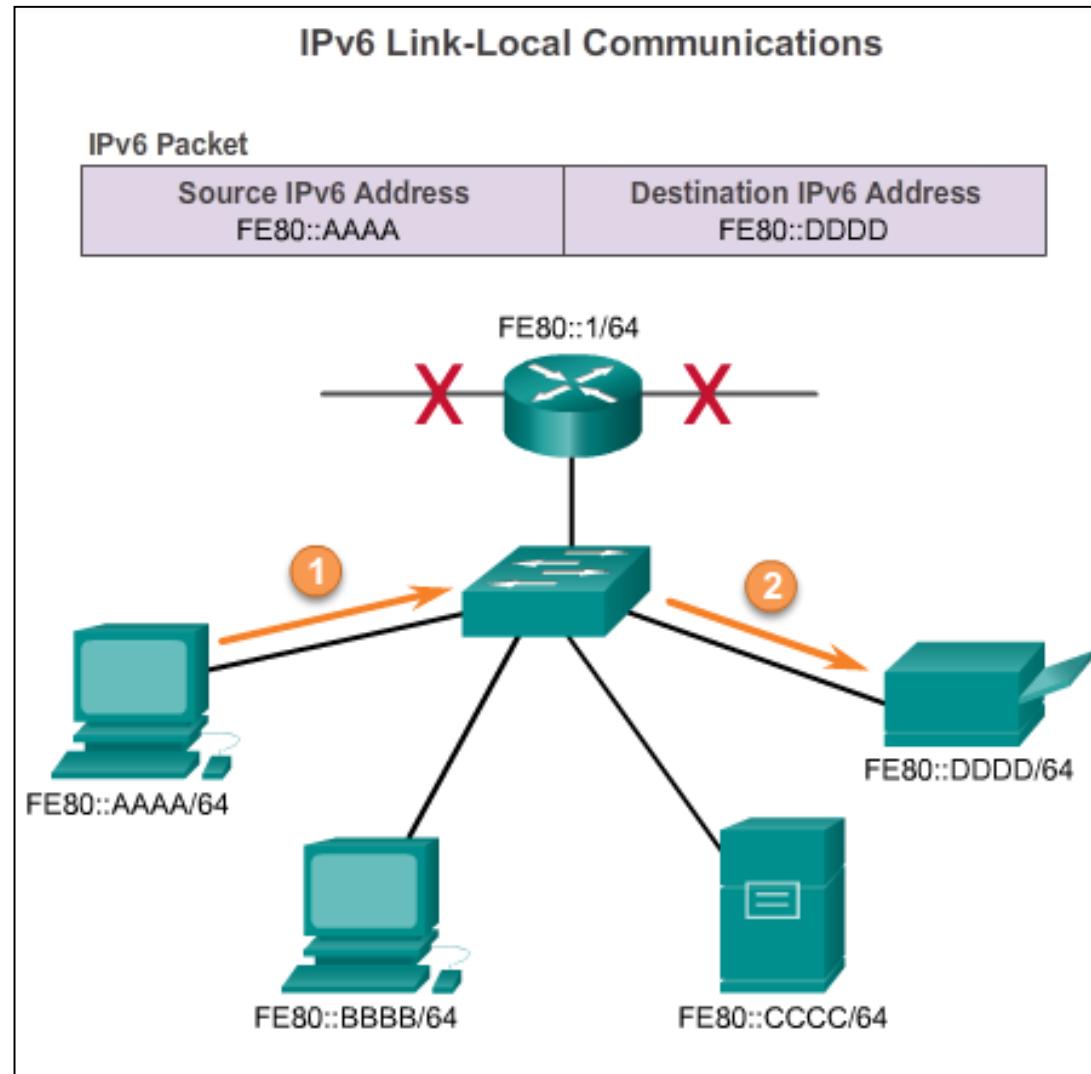
IPv6 Link-Local Unicast Addresses

- Every IPv6-enabled network interface is REQUIRED to have a link-local address
- Enables a device to communicate with other IPv6-enabled devices on the same link and only on that link (subnet)
- FE80::/10 range, first 10 bits are 1111 1110 10xx xxxx
- 1111 1110 10**00 0000** (FE80) - 1111 1110 10**11 1111** (FEBF)



IPv6 Link-Local Unicast Addresses (cont.)

Packets with a source or destination link-local address cannot be routed beyond the link from where the packet originated.



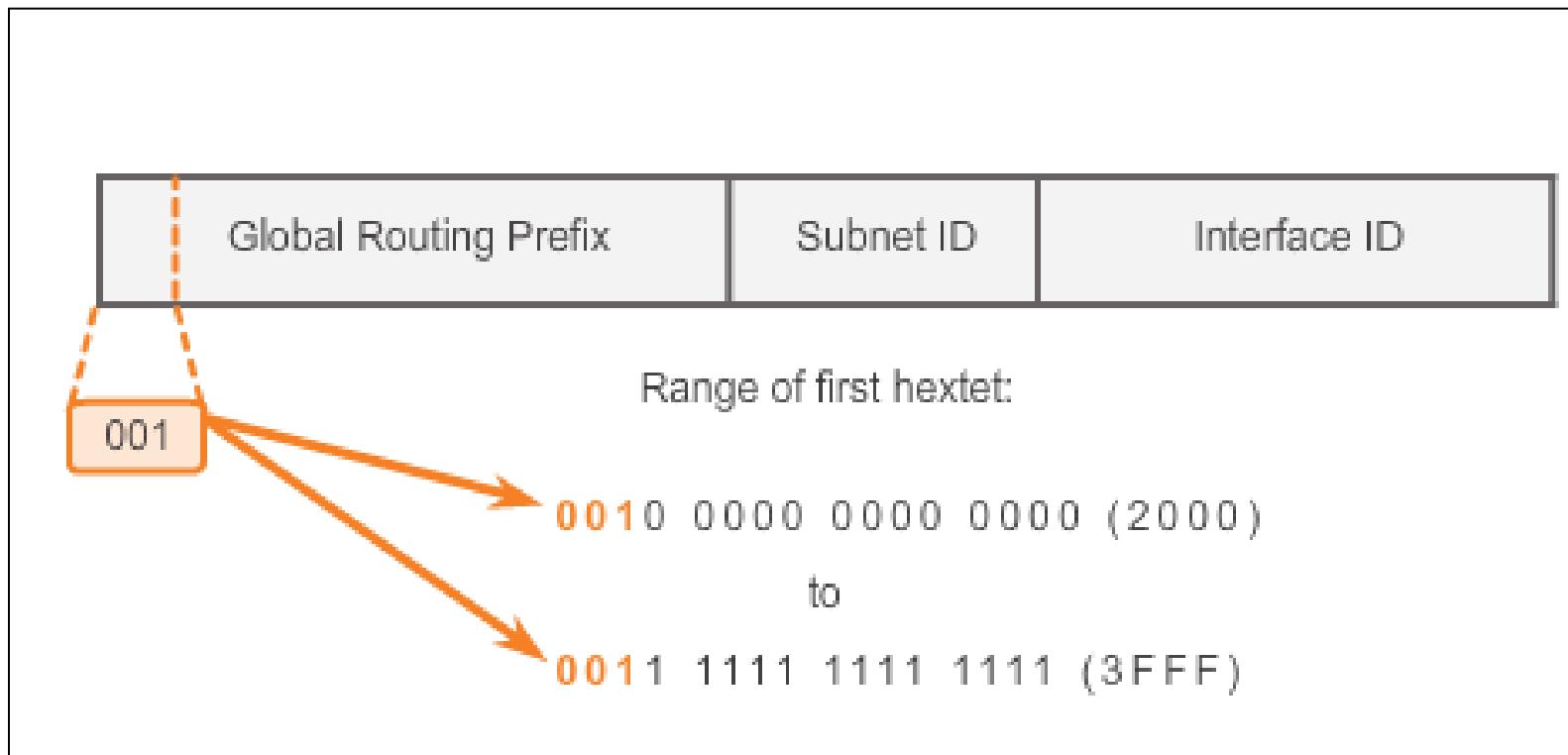
IPv6 Unicast Addresses

Structure of an IPv6 Global Unicast Address

- IPv6 global unicast addresses are globally unique and routable on the IPv6 Internet
- Equivalent to public IPv4 addresses
- **ICANN** allocates IPv6 address blocks to the five RIRs

Structure of an IPv6 Global Unicast Address (cont.)

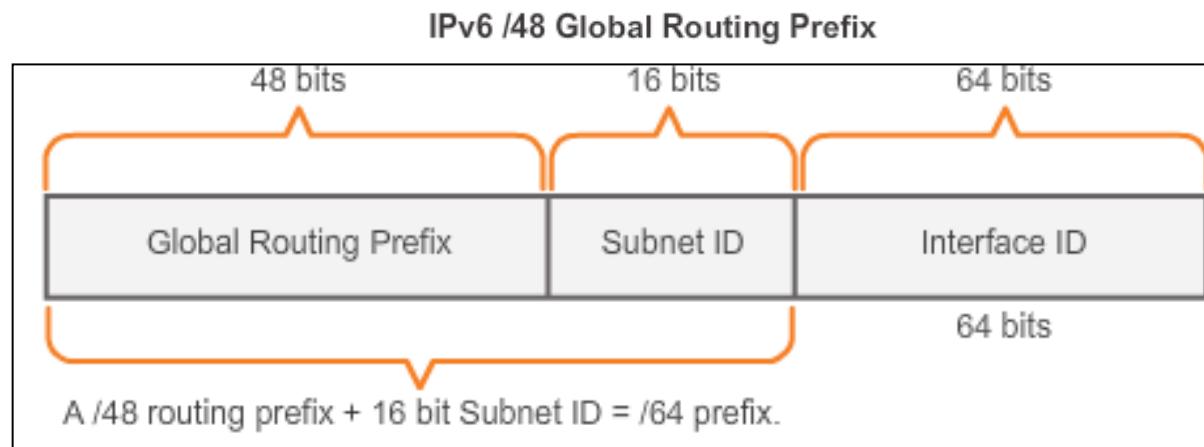
Currently, only **global unicast addresses** with the first three bits of 001 or 2000::/3 are being assigned



Structure of an IPv6 Global Unicast Address (cont.)

A global unicast address has three parts: Global Routing Prefix, Subnet ID, and Interface ID.

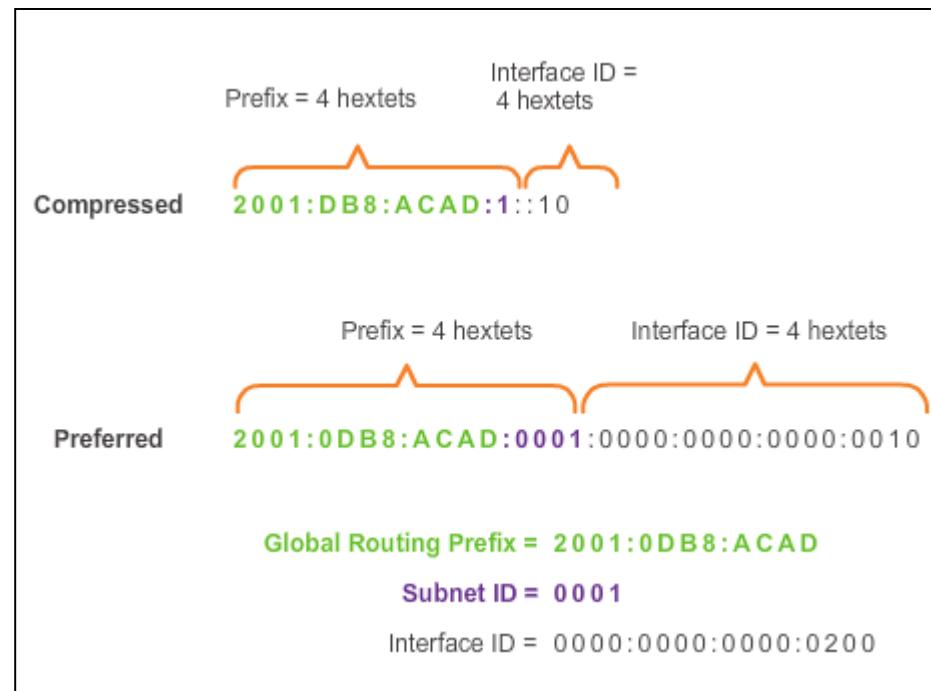
- **Global Routing Prefix** is the prefix or network portion of the address assigned by the provider, such as an ISP, to a customer or site, currently, RIR's assign a /48 global routing prefix to customers.
- 2001:0DB8:ACAD::/48 has a prefix that indicates that the first 48 bits (2001:0DB8:ACAD) is the prefix or network portion.



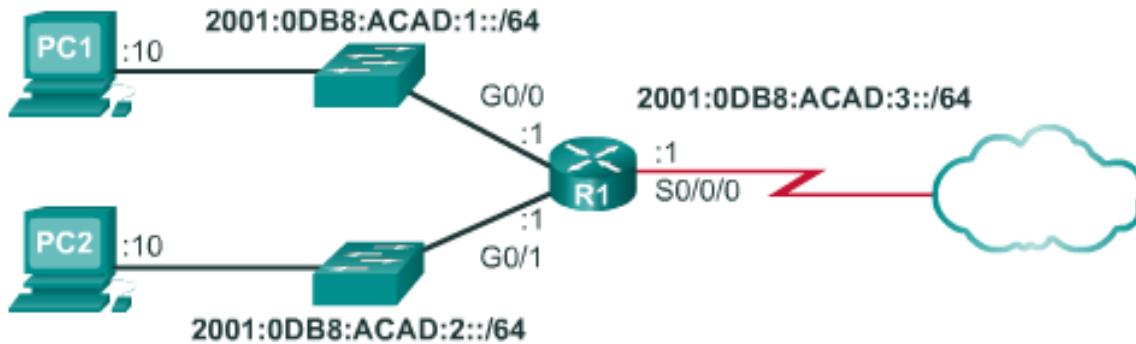
Structure of an IPv6 Global Unicast Address (cont.)

- **Subnet ID** is used by an organization to identify subnets within its site
- **Interface ID**
 - Equivalent to the host portion of an IPv4 address.
 - Used because a single host may have multiple interfaces, each having one or more IPv6 addresses.

Reading a Global Unicast Address



Static Configuration of a Global Unicast Address

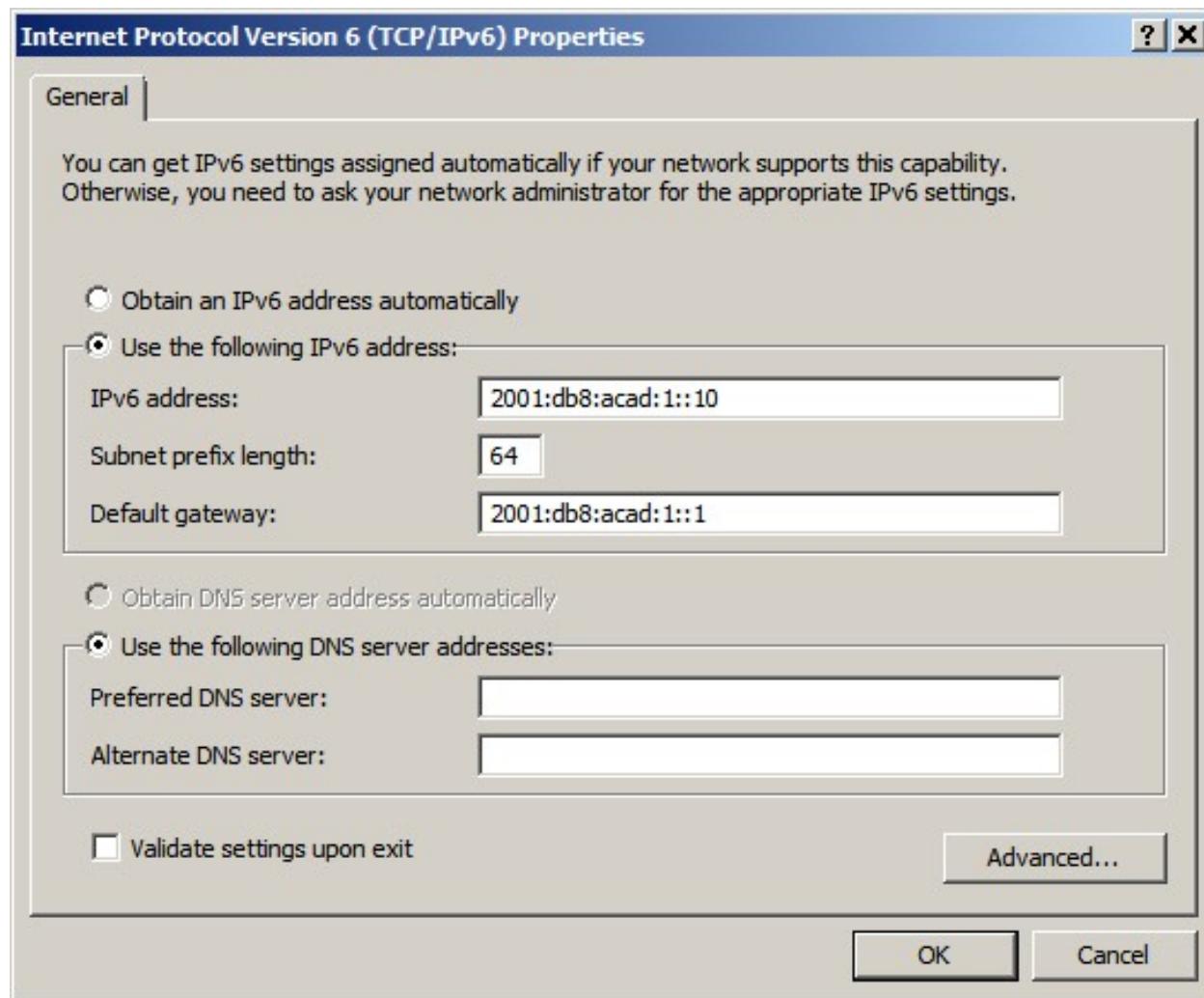


```
R1(config)#interface gigabitethernet 0/0
R1(config-if)#ipv6 address 2001:db8:acad:1::1/64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface gigabitethernet 0/1
R1(config-if)#ipv6 address 2001:db8:acad:2::1/64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ipv6 address 2001:db8:acad:3::1/64
R1(config-if)#clock rate 56000
R1(config-if)#no shutdown
```

IPv6 Unicast Addresses

Static Configuration of an IPv6 Global Unicast Address (cont.)

Windows IPv6 Setup



IPv6 Unicast Addresses

Dynamic Configuration of a Global Unicast Address using SLAAC

Stateless Address Autoconfiguraton (SLAAC)

- A method that allows a device to obtain its prefix, prefix length and default gateway from an IPv6 router
- No DHCPv6 server needed
- Rely on ICMPv6 Router Advertisement (RA) messages

IPv6 routers

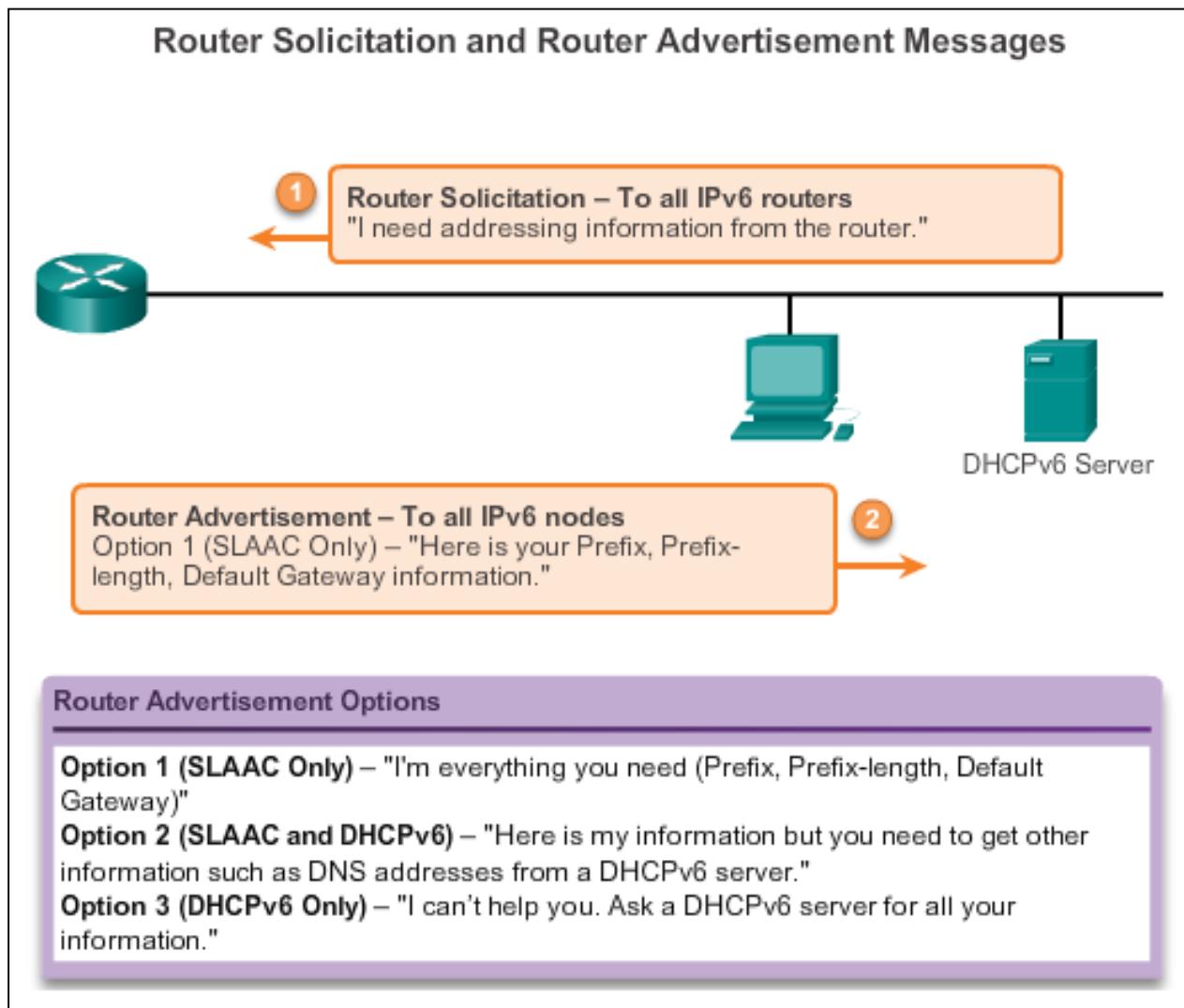
- Forwards IPv6 packets between networks
- Can be configured with static routes or a dynamic IPv6 routing protocol
- Sends ICMPv6 RA messages

IPv6 Unicast Addresses

Dynamic Configuration of a Global Unicast Address using SLAAC (cont.)

- The **IPv6 unicast-routing** command **enables IPv6 routing**.
- RA message can contain one of the following three options:
 - **SLAAC Only** – Uses the information contained in the RA message.
 - **SLAAC and DHCPv6** – Uses the information contained in the RA message and get other information from the DHCPv6 server, stateless DHCPv6 (for example, DNS).
 - **DHCPv6 only** – The device should not use the information in the RA, stateful DHCPv6.
- Routers send ICMPv6 RA messages using the link-local address as the source IPv6 address

Dynamic Configuration of a Global Unicast Address using SLAAC (cont.)

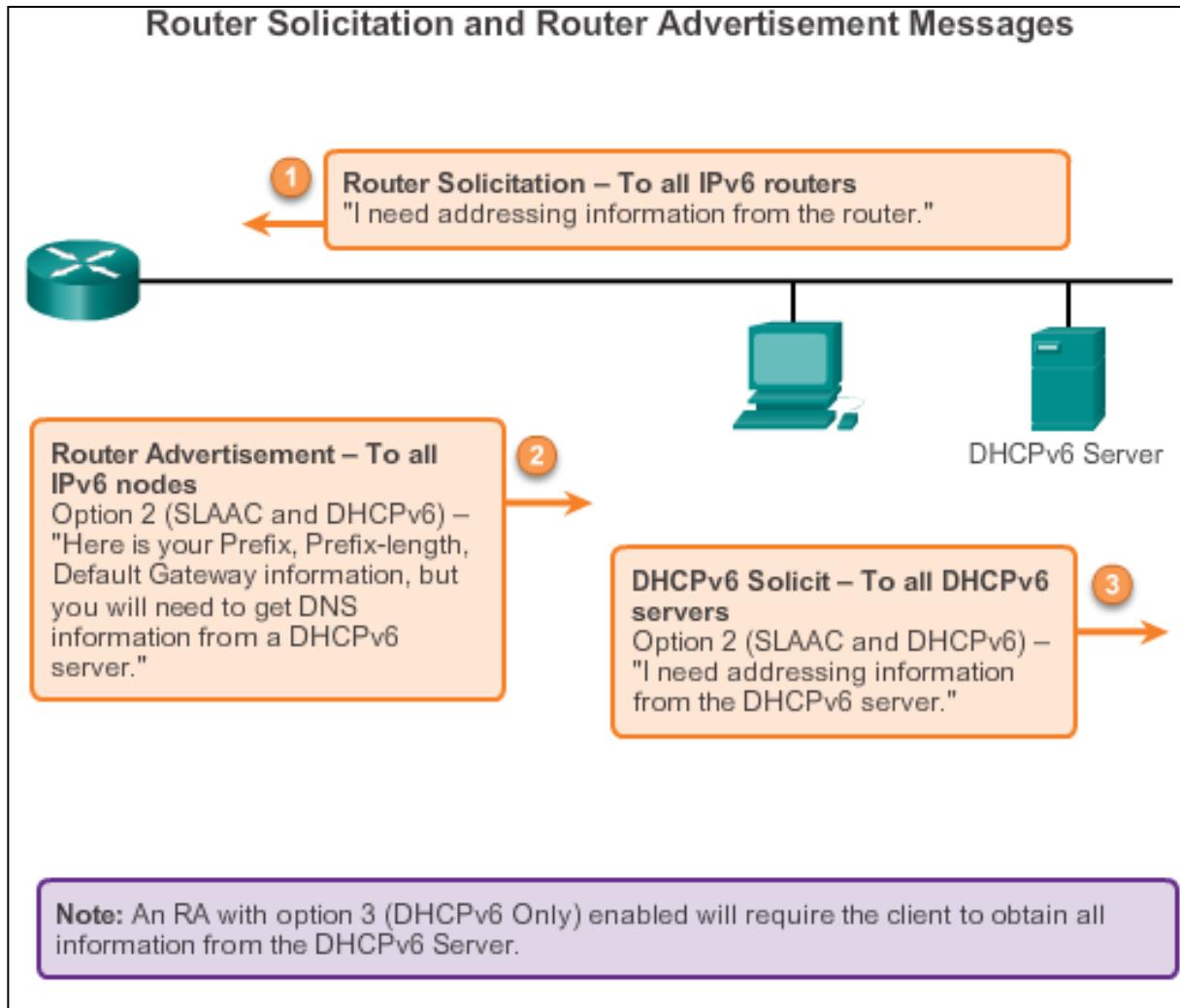


Dynamic Configuration of a Global Unicast Address using DHCPv6 (cont.)

Dynamic Host Configuration Protocol for IPv6 (DHCPv6)

- Similar to IPv4
- Automatically receives addressing information, including a global unicast address, prefix length, default gateway address and the addresses of DNS servers using the services of a DHCPv6 server.
- Device may receive all or some of its IPv6 addressing information from a DHCPv6 server depending upon whether option 2 (SLAAC and DHCPv6) or option 3 (DHCPv6 only) is specified in the ICMPv6 RA message.
- Host may choose to ignore whatever is in the router's RA message and obtain its IPv6 address and other information directly from a DHCPv6 server.

dynamic Configuration of a Global Unicast Address using DHCPv6 (cont.)



EUI-64 Process

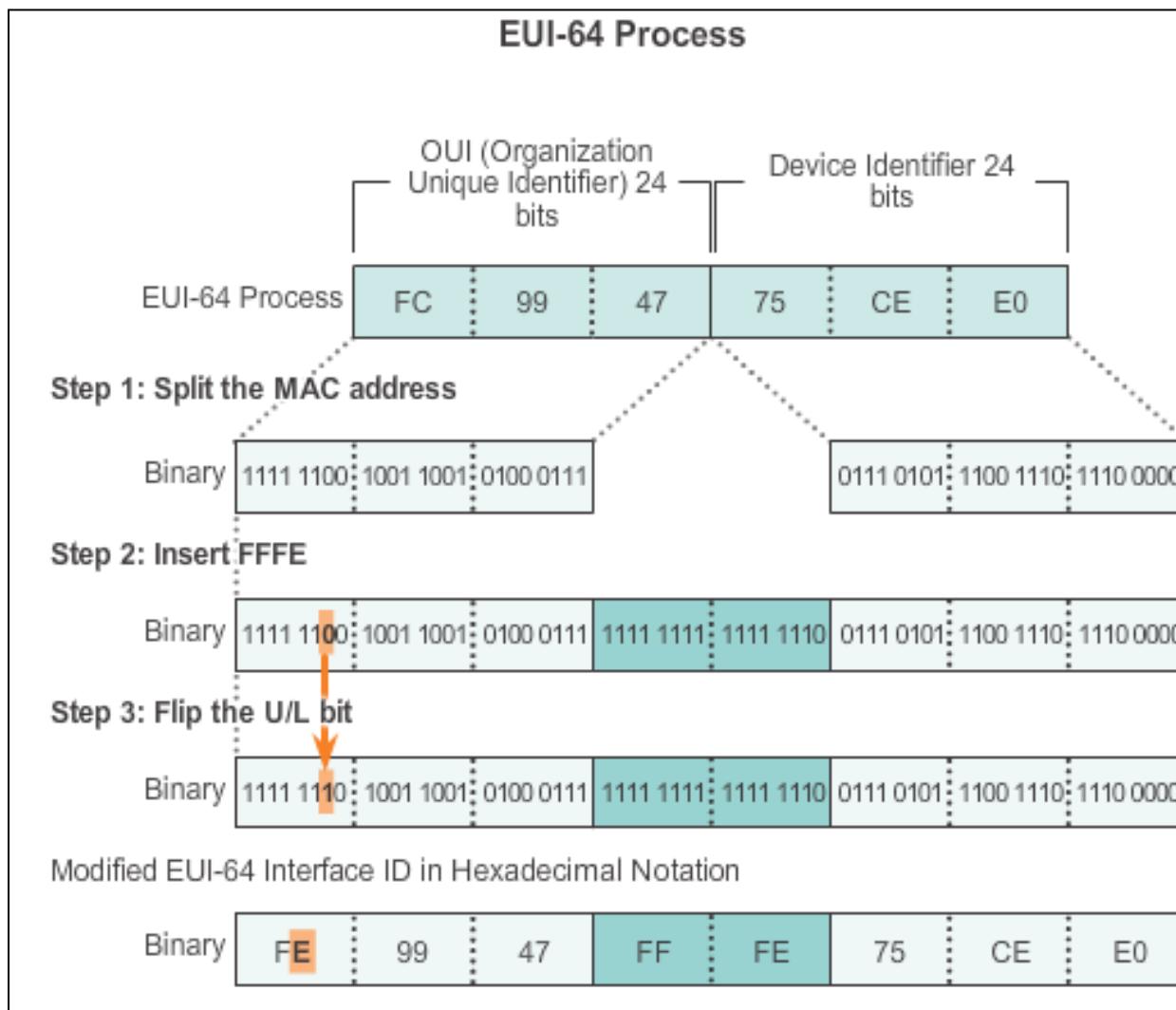
EUI-64 Process

- Uses a client's 48-bit Ethernet MAC address and inserts another 16 bits in the middle of the 46-bit MAC address to create a 64-bit Interface ID.
- Advantage is that the Ethernet MAC address can be used to determine the interface; is easily tracked.

EUI-64 Interface ID is represented in binary and comprises three parts:

- 24-bit OUI from the client MAC address, but the **7th bit** (the Universally/Locally bit) is reversed (0 becomes a 1).
- Inserted as a 16-bit value **FFFE**.
- 24-bit device identifier from the client MAC address.

EUI-64 Process



EUI-64 Process

```
R1#show interface gigabitethernet 0/0
GigabitEthernet0/0 is up, line protocol is up
    Hardware is CN Gigabit Ethernet, address is fc99.4775.c3e0
(bia fc99.4775.c3e0)
<Output Omitted>
```

```
R1#show ipv6 interface brief
GigabitEthernet0/0      [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:1::1
GigabitEthernet0/1      [up/up]
  FE80::FE99:47FF:FE75:C3E1
  2001:DB8:ACAD:2::1
Serial0/0/0             [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:3::1
Serial0/0/1              [administratively down/down]
  unassigned
R1#
```

Link-local addresses using
EUI-64

Static Link-local Addresses

Configuring Link-local

```
R1(config)#interface gigabitethernet 0/0
R1(config-if)#ipv6 address fe80::1 ?
    link-local  Use link-local address

R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#exit
R1(config)#interface gigabitethernet 0/1
R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#

```

Static Link-local Addresses (cont.)

Configuring Link-local

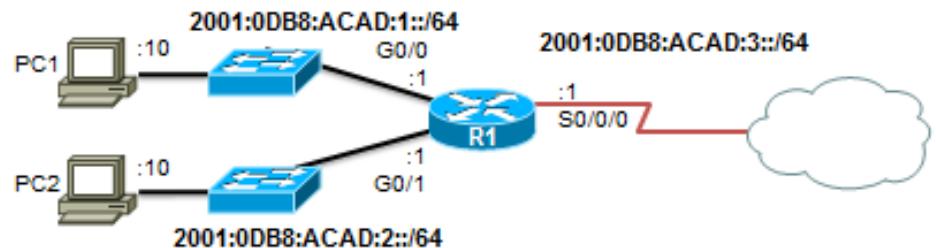
```
R1#show ipv6 interface brief
GigabitEthernet0/0      [up/up]
  FE80::1
  2001:DB8:ACAD:1::1
GigabitEthernet0/1      [up/up]
  FE80::1
  2001:DB8:ACAD:2::1
Serial0/0/0              [up/up]
  FE80::1
  2001:DB8:ACAD:3::1
Serial0/0/1              [administratively down/down]
  unassigned
R1#
```

Statically configured link-local addresses

Verifying IPv6 Address Configuration

Each interface has two IPv6 addresses -

1. global unicast address that was configured
2. one that begins with FE80 is automatically added as a link-local unicast address



```
R1#show ipv6 interface brief
GigabitEthernet0/0      [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:1::1
GigabitEthernet0/1      [up/up]
  FE80::FE99:47FF:FE75:C3E1
  2001:DB8:ACAD:2::1
Serial0/0/0             [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:3::1
Serial0/0/1             [administratively down/down]
  unassigned
R1#
```

Verifying IPv6 Address Configuration (cont.)

```
R1#show ipv6 route
IPv6 Routing Table - default - 7 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user
Static

<output omitted>

C 2001:DB8:ACAD:1::/64 [0/0]
    via GigabitEthernet0/0, directly connected
L 2001:DB8:ACAD:1::1/128 [0/0]
    via GigabitEthernet0/0, receive
C 2001:DB8:ACAD:2::/64 [0/0]
    via GigabitEthernet0/1, directly connected
L 2001:DB8:ACAD:2::1/128 [0/0]
    via GigabitEthernet0/1, receive
C 2001:DB8:ACAD:3::/64 [0/0]
    via Serial0/0/0, directly connected
L 2001:DB8:ACAD:3::1/128 [0/0]
    via Serial0/0/0, receive
L FF00::/8 [0/0]
    via Null0, receive
R1#
```

Assigned IPv6 Multicast Addresses

- IPv6 multicast addresses have the prefix FF00::/8
- There are two types of **IPv6 multicast addresses**:
 - Assigned multicast
 - Solicited node multicast

Assigned IPv6 Multicast Addresses (cont.)

Two common IPv6 assigned multicast groups include:

- **FF02::1 All-nodes multicast group –**
 - All IPv6-enabled devices join
 - Same effect as an IPv4 broadcast address
- **FF02::2 All-routers multicast group**
 - All IPv6 routers join
 - A router becomes a member of this group when it is enabled as an IPv6 router with the **ipv6 unicast-routing** global configuration mode command.
 - A packet sent to this group is received and processed by all IPv6 routers on the link or network.

Assigned IPv6 Multicast Addresses (cont.)

