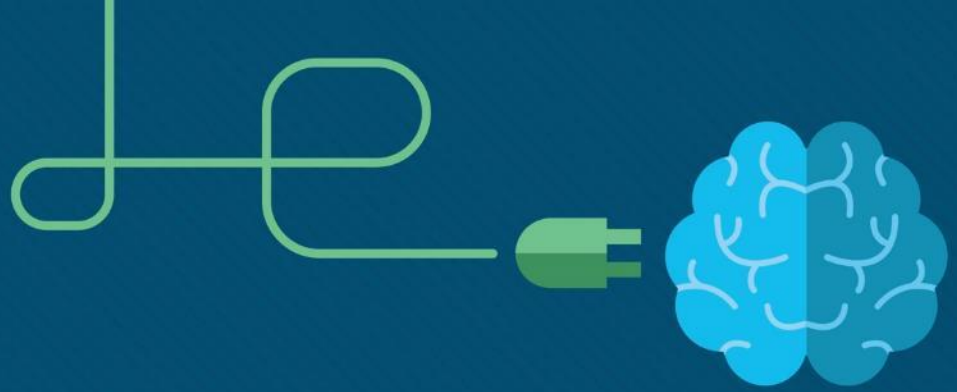


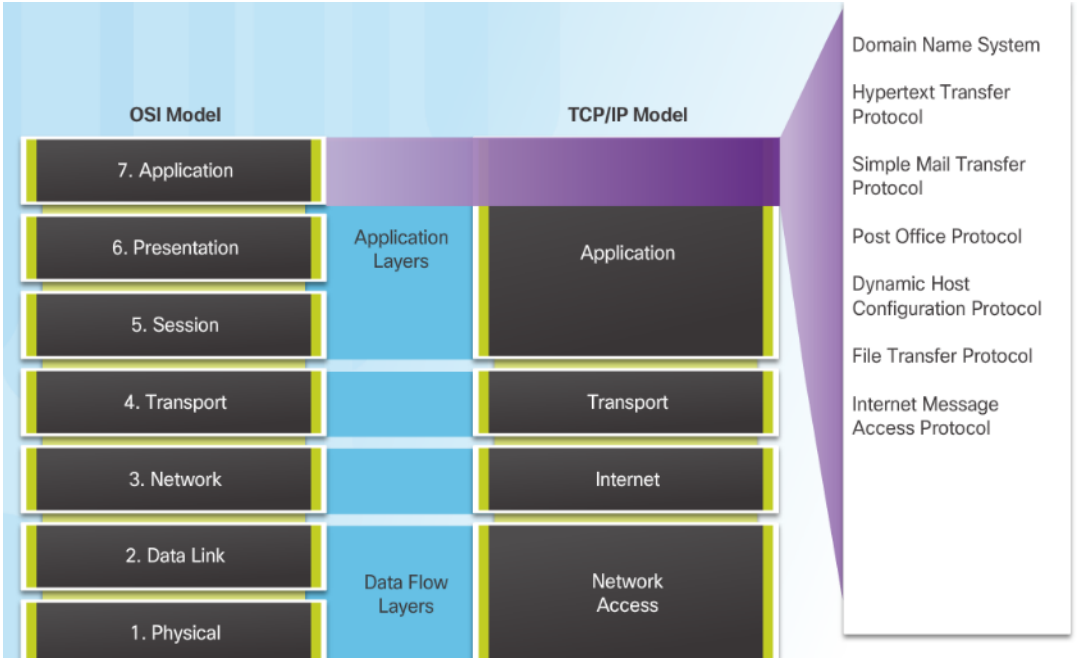
Application Layer

Sections & Objectives

- Application Layer Protocols
  - Explain the operation of the application layer in providing support to end-user applications.
    - Explain how the functions of the application layer, session layer, and presentation layer work together to provide network services to end user applications
    - Explain how common application layer protocols interact with end user applications.
- Well-Known Application Protocols and Services
  - Explain how well-known TCP/IP application layer protocols operate.
    - Explain how web and email protocols operate.
    - Explain how DNS and DHCP operate.
    - Explain how file transfer protocols operate.

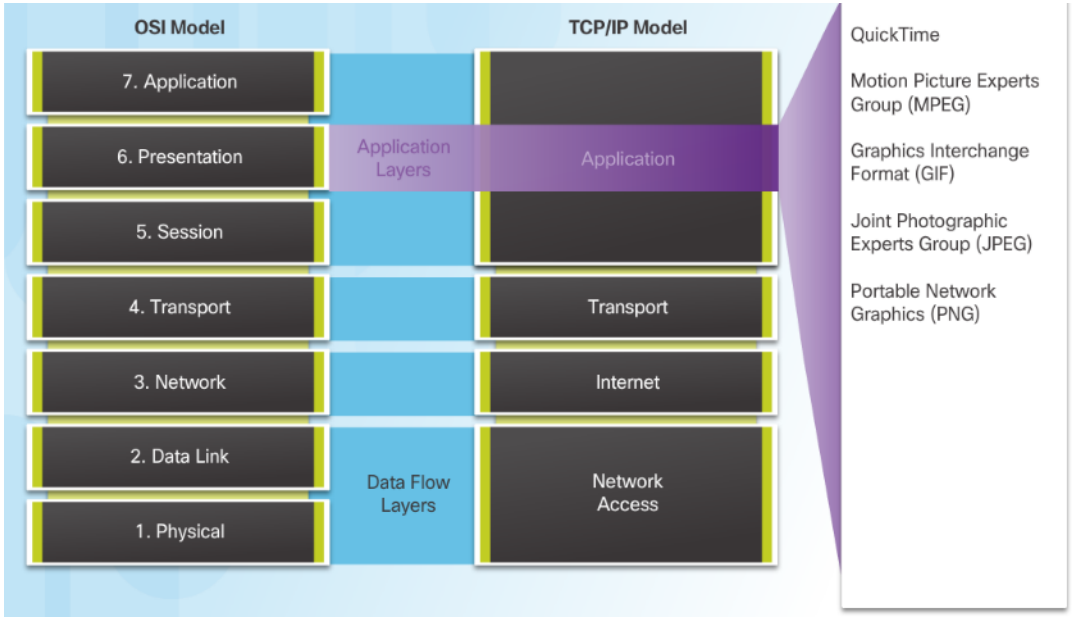
Application Layer Protocols

Application Layer



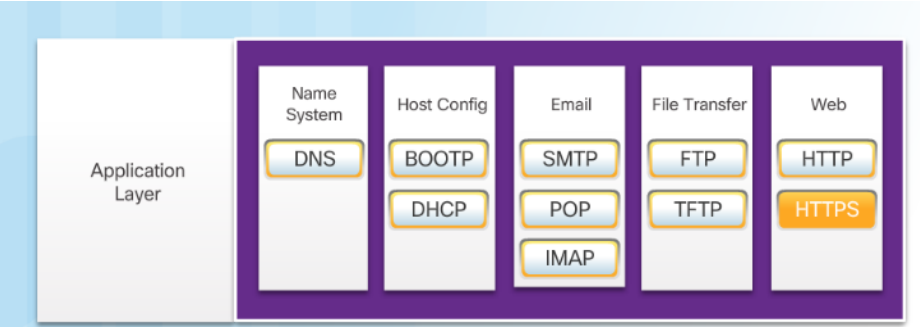
- Application Layer:
  - Closest to the end user.
  - Used to exchange data between programs running on the source and destination hosts.

Presentation and Session Layer



- Presentation Layer function:
  - Formatting data at the source device into a compatible form for the receiving device.
  - Compressing data.
  - Encrypting data.
- Session Layer Function
  - Create and maintain dialogs between source and destination applications.

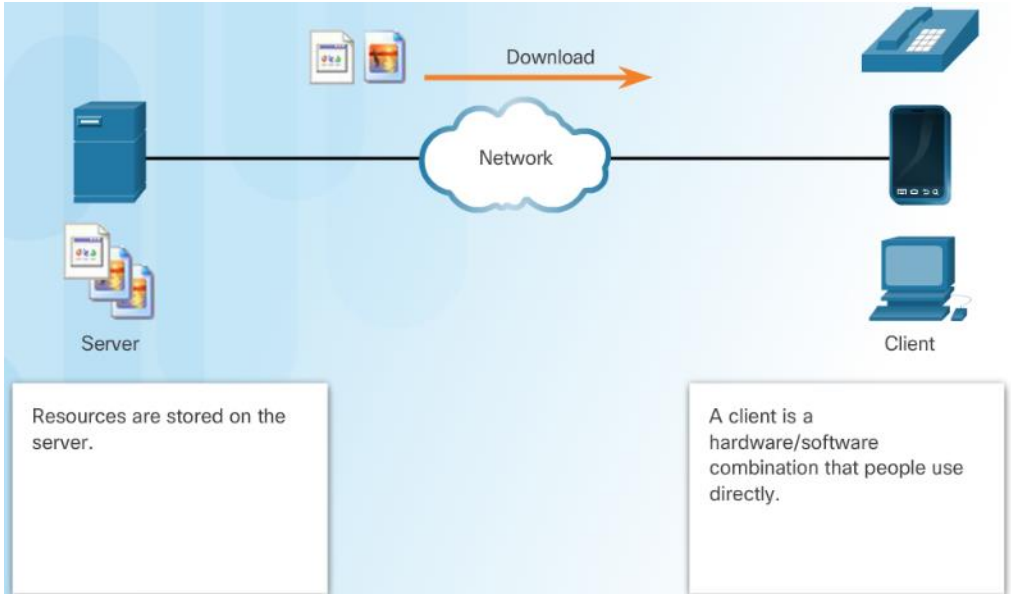
TCP/IP Application Layer Protocols



- Domain Name Server (DNS) TCP,UDP 53 - Translates domain names, such as cisco.com, into IP addresses.
- (BOOTP) – Bootstrap Protocol - BOOTP is being superseded by DHCP.
- Dynamic Host Configuration Protocol (DHCP) UDP client 68, server 67 – Dynamically assigns IP addresses to client stations at start-up.
- Simple Mail Transport Protocol (SMTP) TCP 25 - Enables clients to send email to a mail server.
- Post Office Protocol (POP) TCP 110 - Enables clients to retrieve email from a mail server.
- Internet Message Access Protocol (IMAP) TCP 143 - Enables clients to retrieve email from a mail server, maintains email on server.
- File Transfer Protocol (FTP) TCP 20 and 21 - Reliable, connection-oriented, and acknowledged file delivery protocol.
- Trivial File Transfer Protocol (TFTP) UDP 69 – simple connectionless file transfer protocol.
- Hypertext Transfer Protocol (HTTP) TCP 80, 8080 - Set of rules for exchanging text, graphic images, etc. on the World Wide Web.
- Hypertext Transfer Protocol Secure (HTTPS) TCP, UDP 443 – Uses encryption and authentication to secure communication.

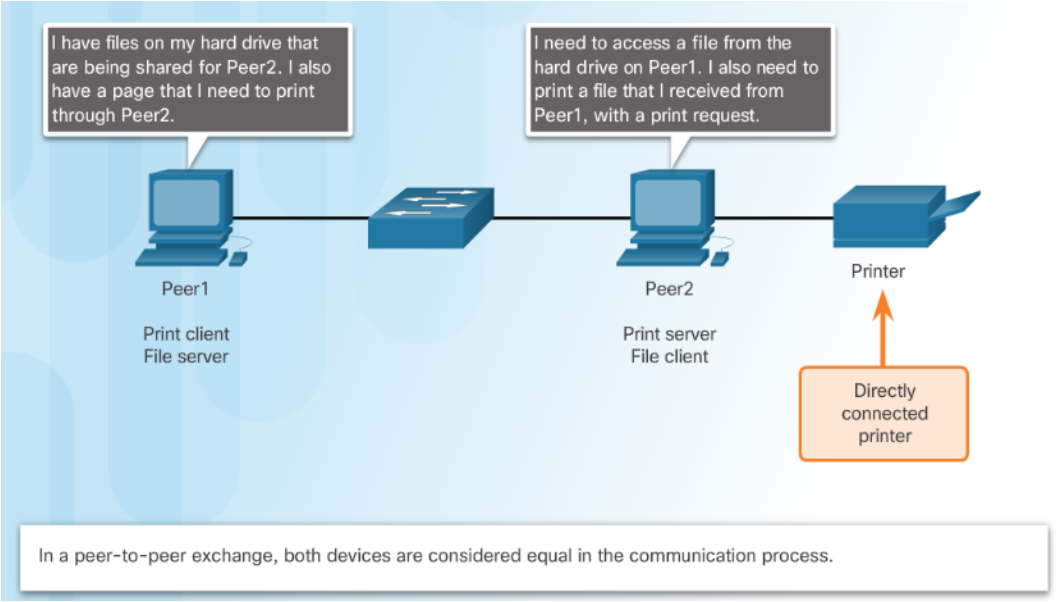
Client-Server Model

- Client and server processes are considered to be in the application layer.
- Application layer protocols describe the format of the requests and responses between clients and servers.
- Example of a client-server network is using an ISP's email service to send, receive and store email.



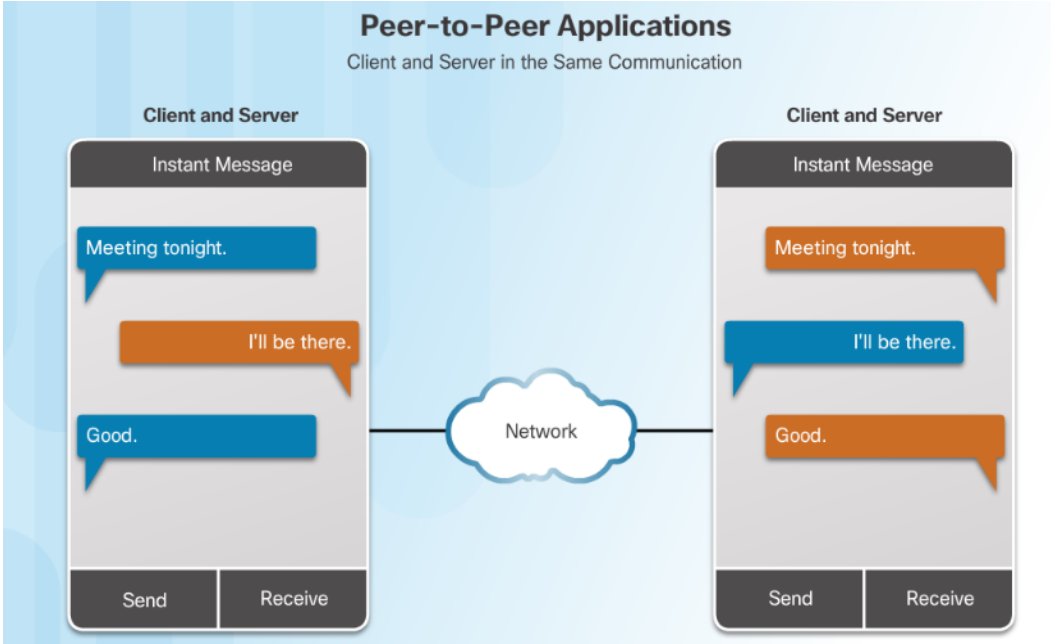
Peer-to-Peer Networks

- Data is accessed from a peer device without the use of a dedicated server.
- Each device (known as a peer) can function as both a server and a client.

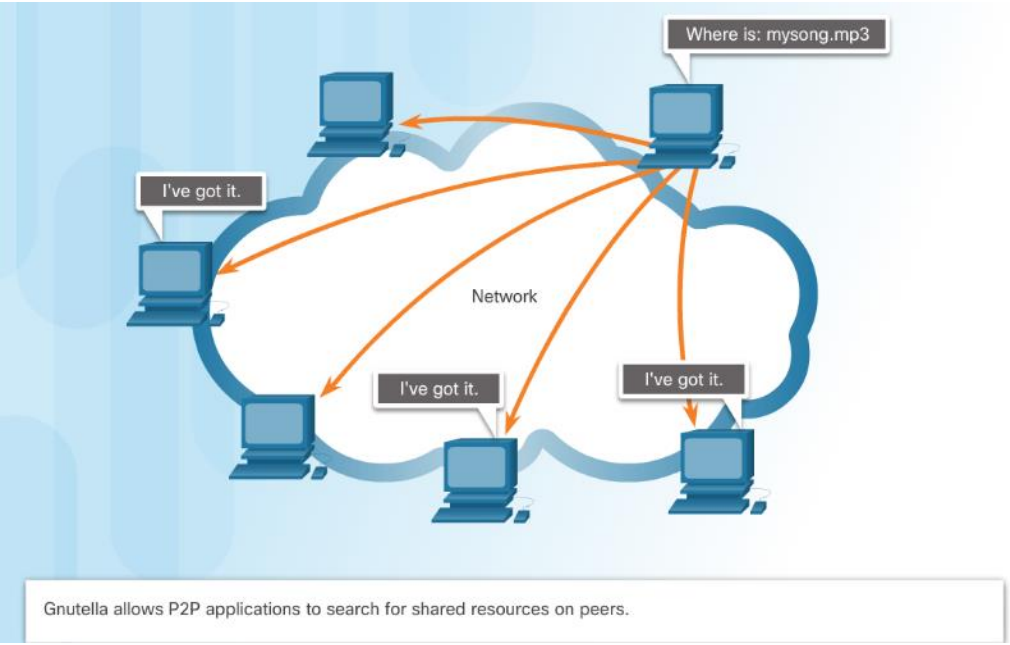


Peer-to-Peer Applications

- A P2P application allows a device to act as both a client and a server within the same communication.
- P2P applications require that each end device provide a user interface and run a background service.



Common P2P Applications



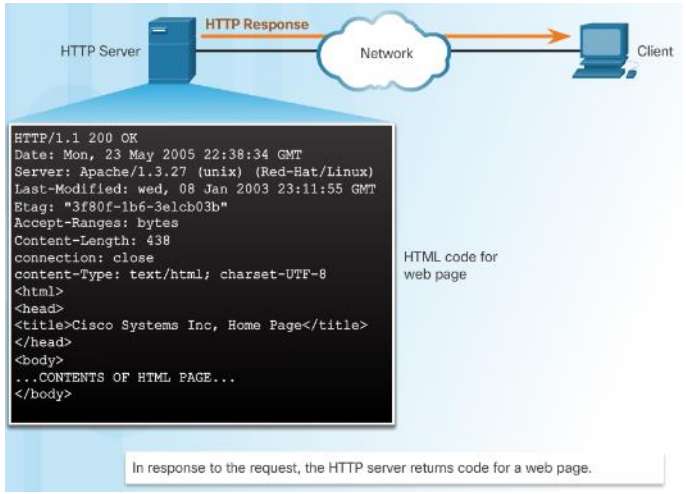
Gnutella allows P2P applications to search for shared resources on peers.

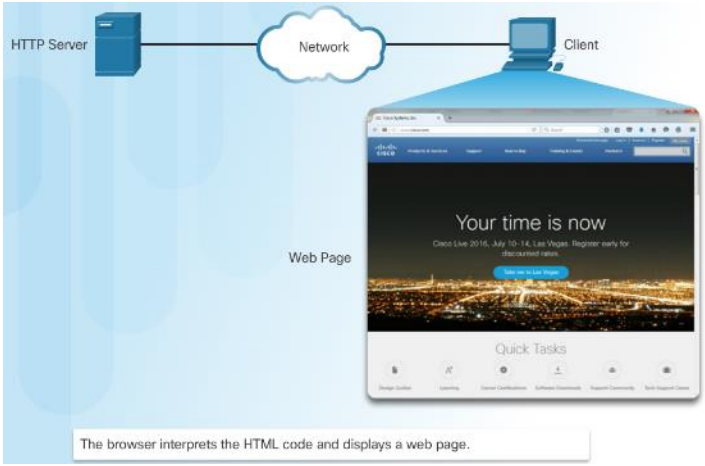
- Common P2P networks include:
  - G2
  - Bitcoin
  - BitTorrent
  - eDonkey
- Some P2P applications are based on the Gnutella protocol, where each user shares whole files with other users.
- Many P2P applications allow users to share pieces of many files with each other at the same time –this is BitTorrent technology.

Well-Known Application Layer Protocols and Services

Hypertext Transfer Protocol and Hypertext Markup Language

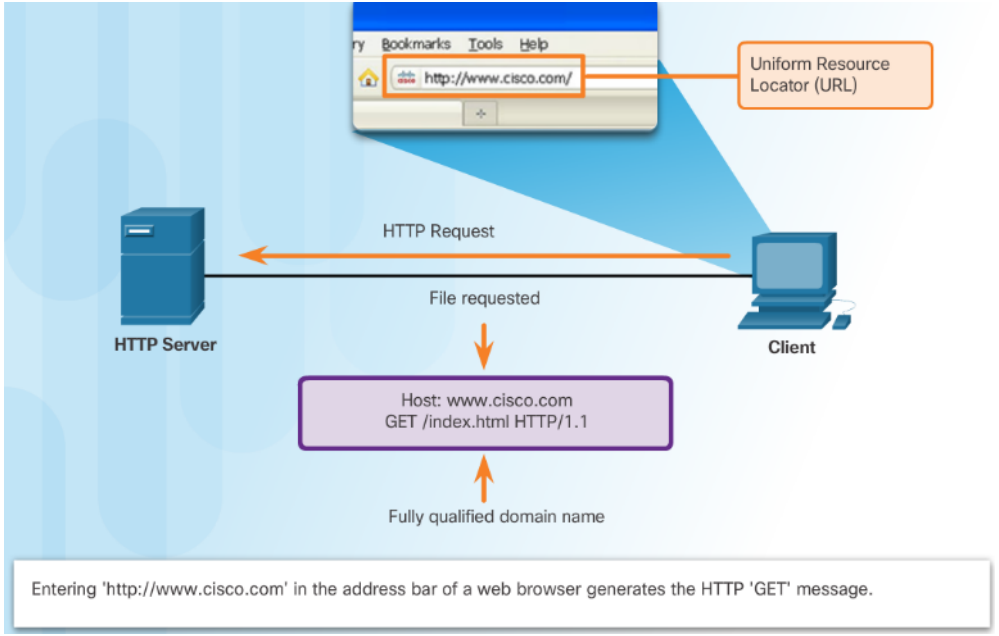
- When a web address or uniform resource locator (URL) is typed into a web browser, the web browser establishes a connection to the web service running on the server, using the HTTP protocol.



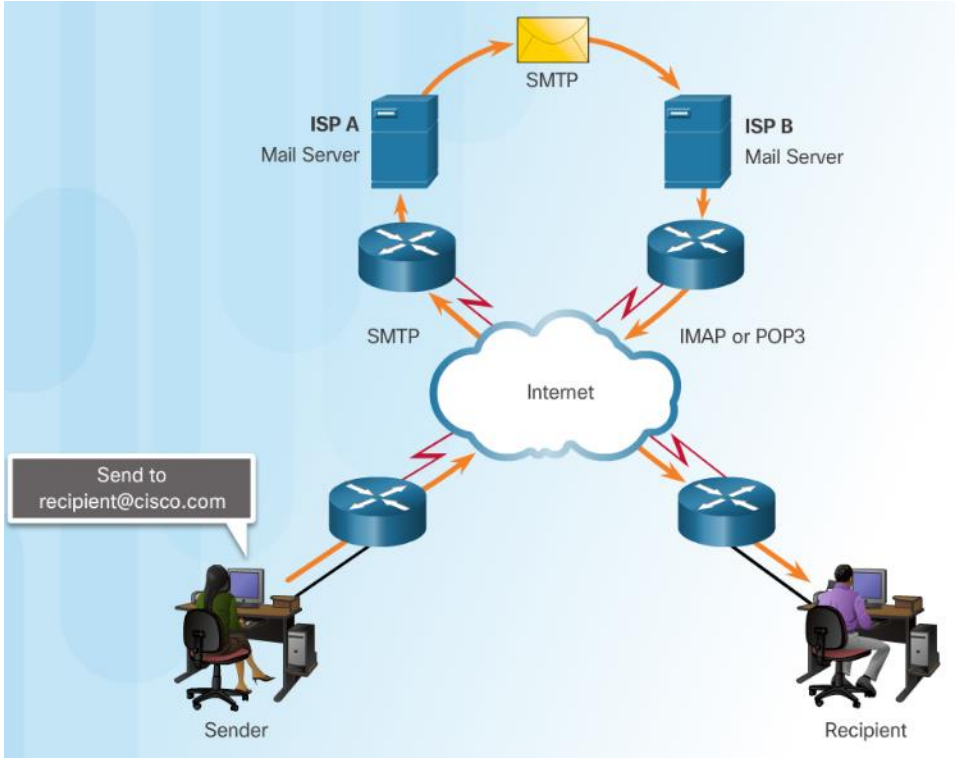


HTTP and HTTPS

- HTTP is a request/response protocol.
- Three common HTTP message types are:
  - GET - A client request for data.
  - POST - Uploads data files to the web server.
  - PUT - Uploads resources or content to the web server.
- HTTP Secure (HTTPS) protocol uses encryption and authentication to secure data.



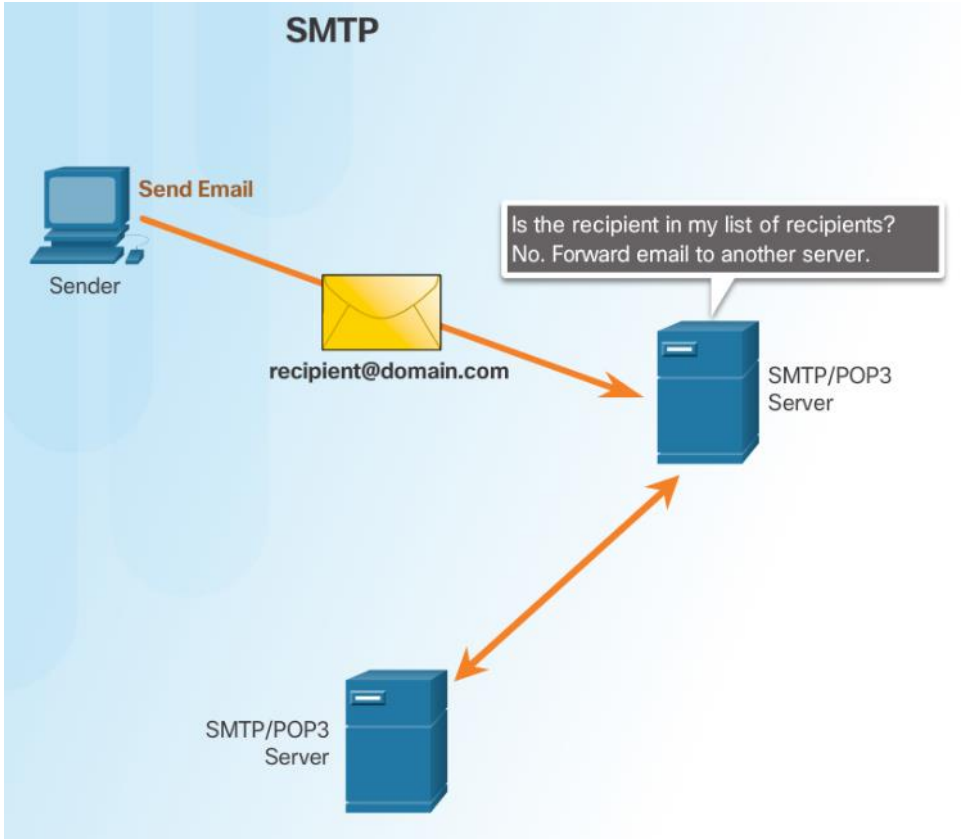
Email Protocols



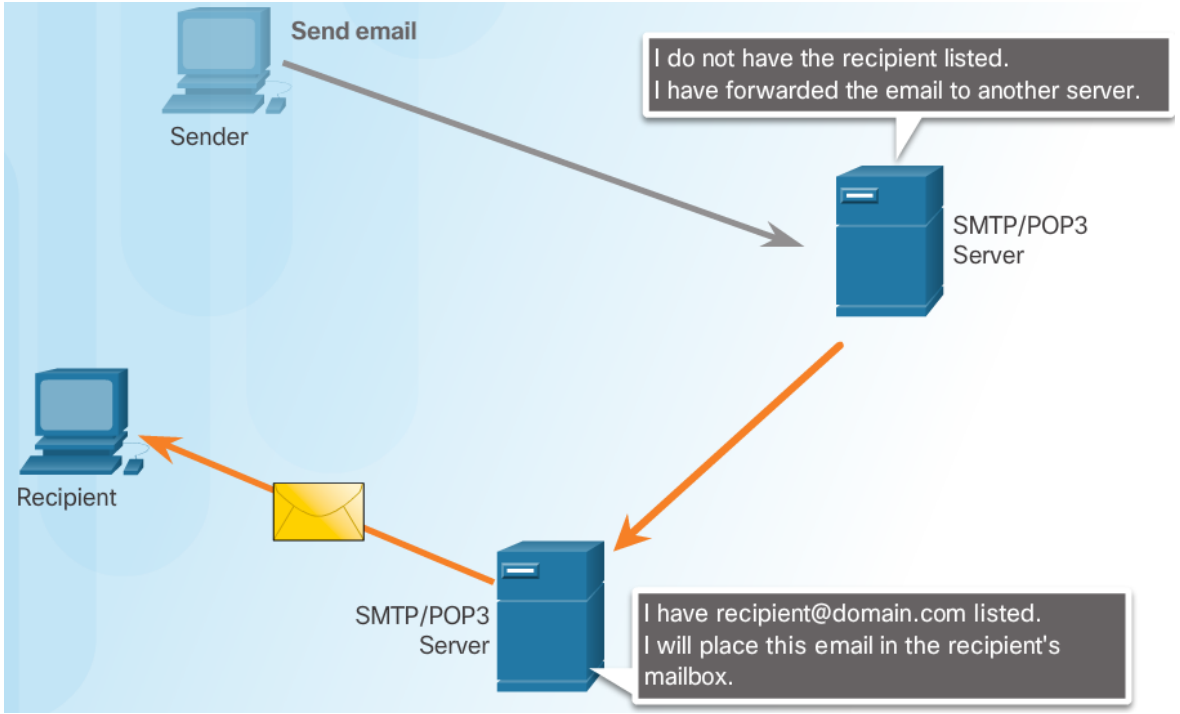
- Email clients communicate with mail servers to send and receive email.
- Mail servers communicate with other mail servers to transport messages from one domain to another.
- Three protocols for email:
  - Simple Mail Transfer Protocol (SMTP) to send email.
  - Post Office Protocol (POP) to retrieve email.
  - Internet Message Access Protocol (IMAP) to retrieve email.

SMTP Operation

- SMTP is used to send email

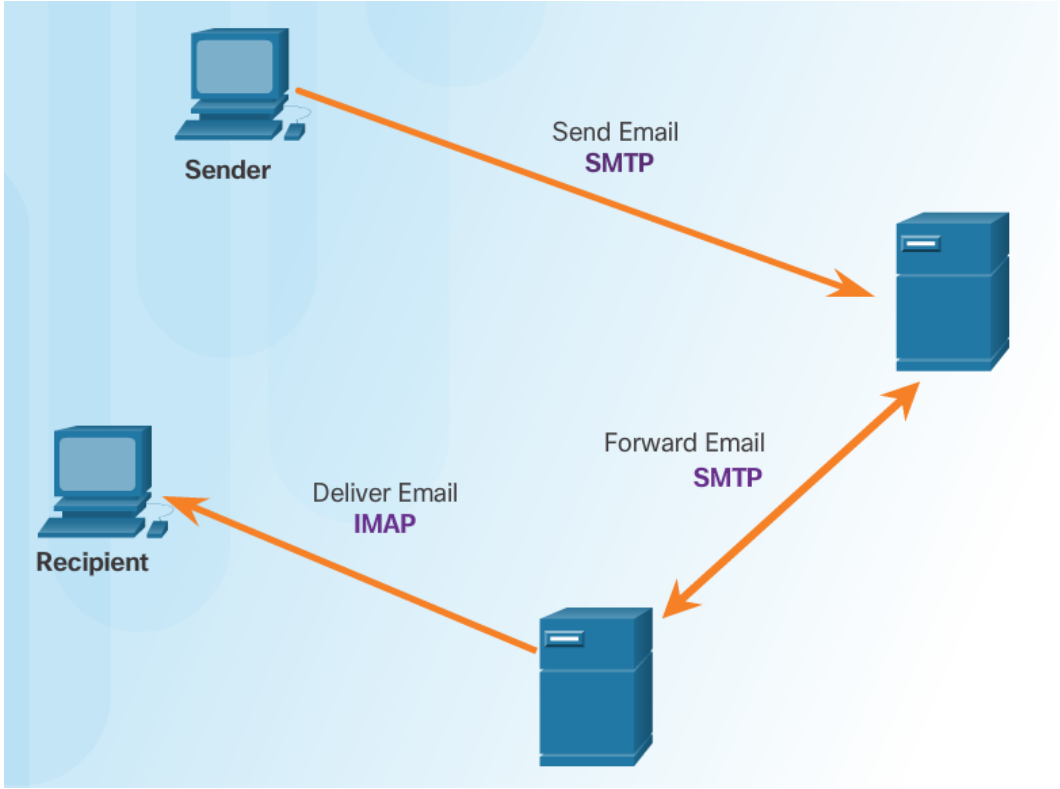


POP Operation



- POP is used to retrieve email from a mail server.
- Email is downloaded from the server to the client and then deleted on the server.

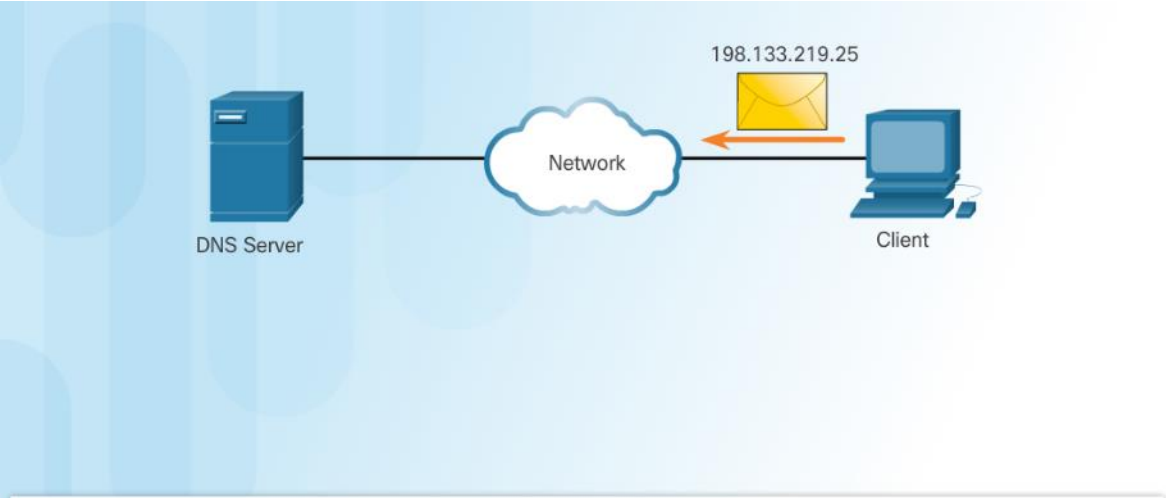
IMAP Operation



- IMAP is used to retrieve mail from a mail server.
- Copies of messages are downloaded from the server to the client and the original messages are stored on the server.

Domain Name Service

- Domain names convert the numeric address into a simple, recognizable name.
- The DNS protocol defines an automated service that matches resource names with the required numeric network address.



A domain name is resolved to its numeric network device address by the DNS protocol.

DNS Message Format

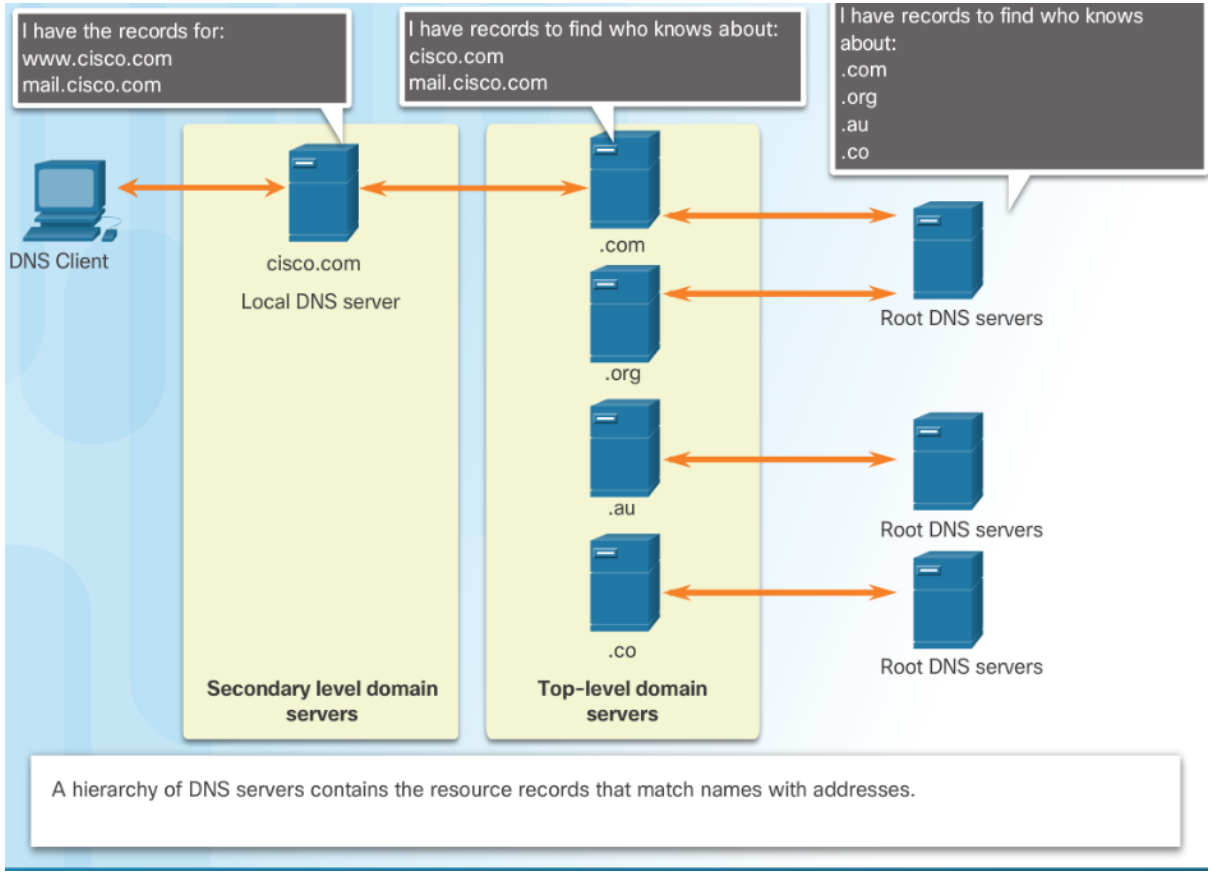
DNS uses the same message format for:

- all types of client queries and server responses
- error messages
- the transfer of resource record information between servers

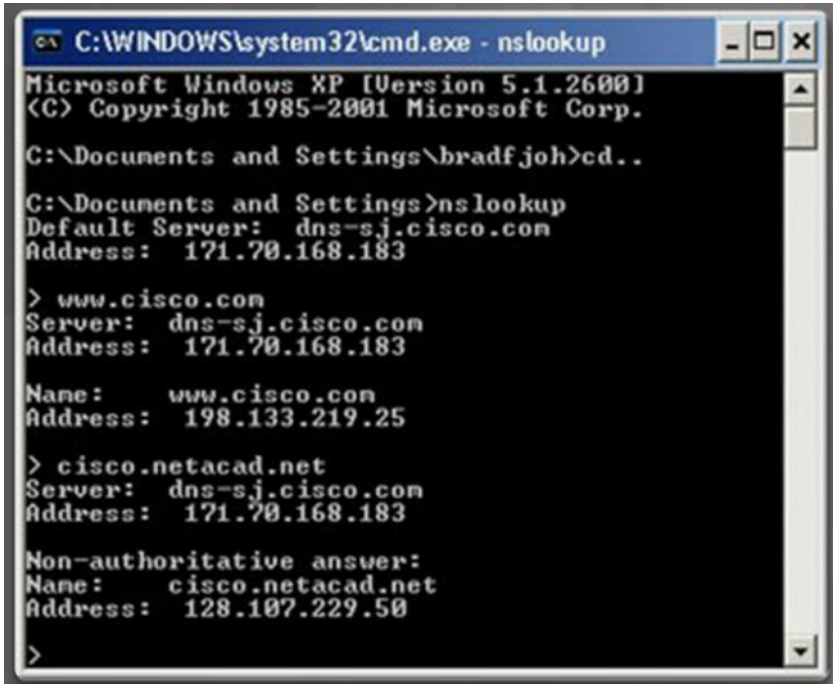
Header	
Question	The question for the name server
Answer	Resource Records answering the question
Authority	Resource Records pointing toward an authority
Additional	Resource Records holding additional information

- When a client makes a query, the server's DNS process first looks at its own records to resolve the name.
- If unable to resolve, it contacts other servers to resolve the name.
- The server temporarily stores the numbered address in the event that the same name is requested again.
- The **ipconfig /displaydns** command displays all of the cached DNS entries on a Windows PC.

DNS Hierarchy



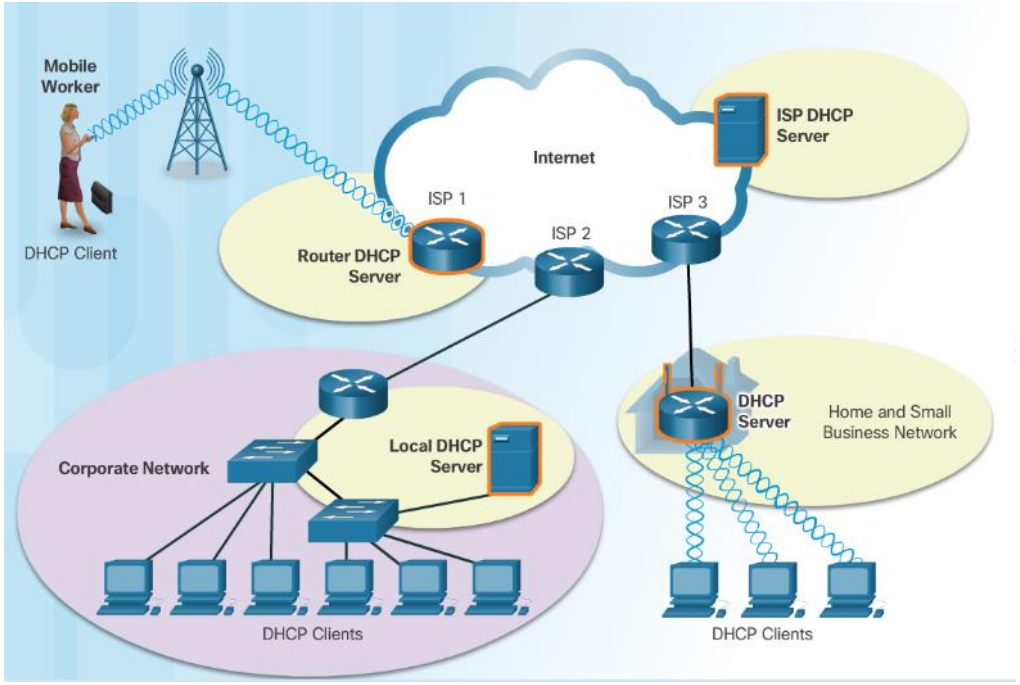
The nslookup Command



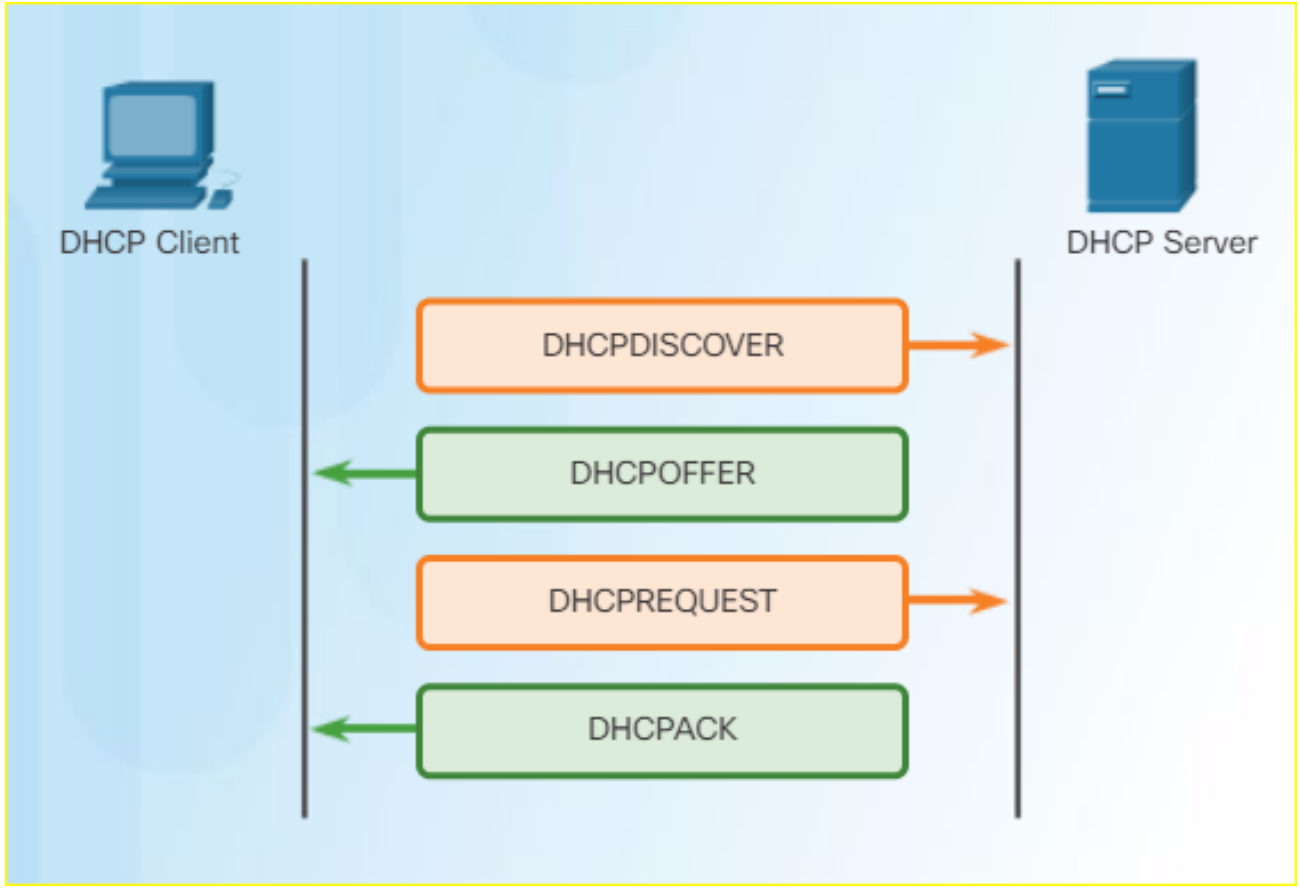
- **Nslookup** - a utility that allows a user to manually query the name servers to resolve a given host.
  - Can also be used to troubleshoot name resolution issues and to verify the current status of the name servers.

Dynamic Host Configuration Protocol

- The Dynamic Host Configuration Protocol (DHCP) for IPv4 automates the assignment of IPv4 addresses, subnet masks, gateways, and other parameters.
- DHCP-distributed addresses are leased for a set period of time, then returned to pool for reuse.
- DHCP is usually employed for end user devices. Static addressing is used for network devices, such as gateways, switches, servers, and printers.
- DHCPv6 (DHCP for IPv6) provides similar services for IPv6 clients.

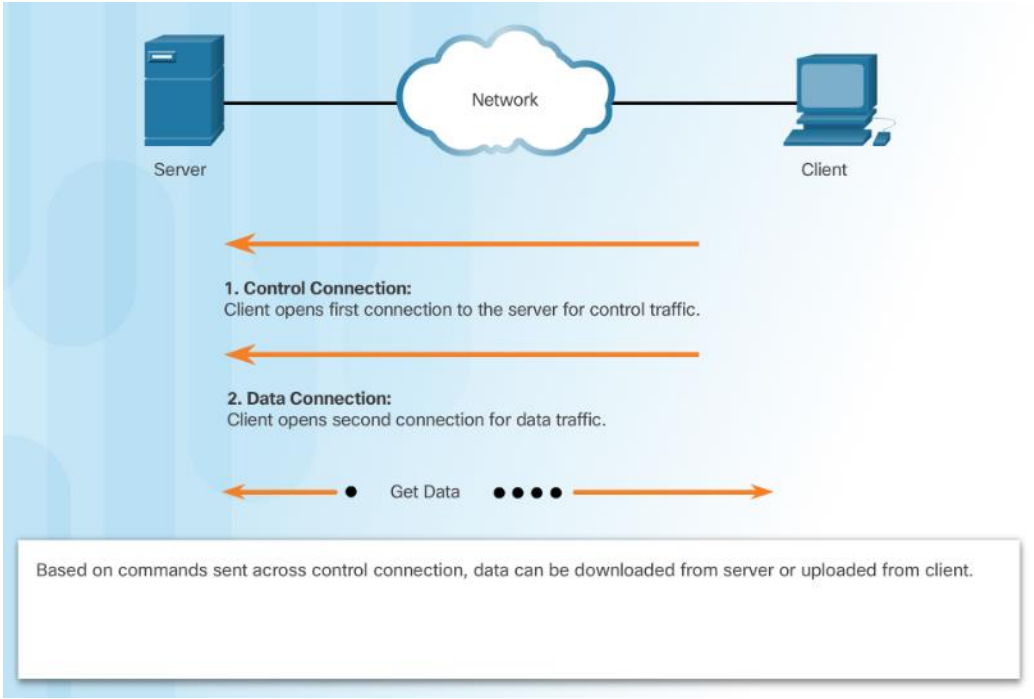


DHCP Operation



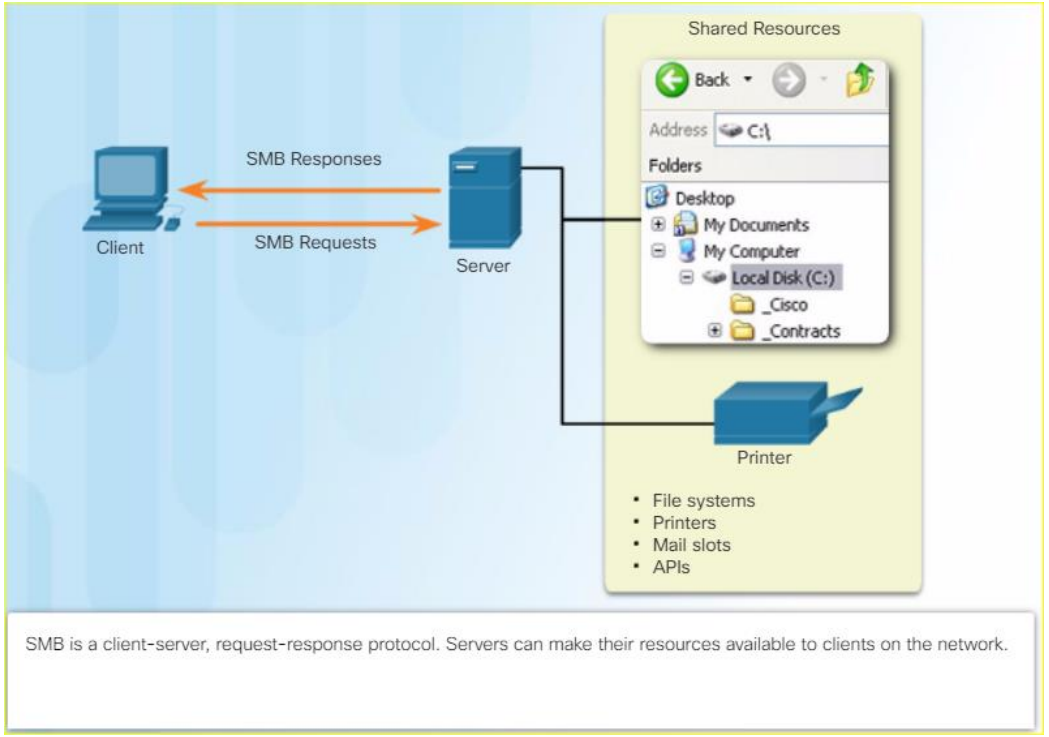
File Transfer Protocol

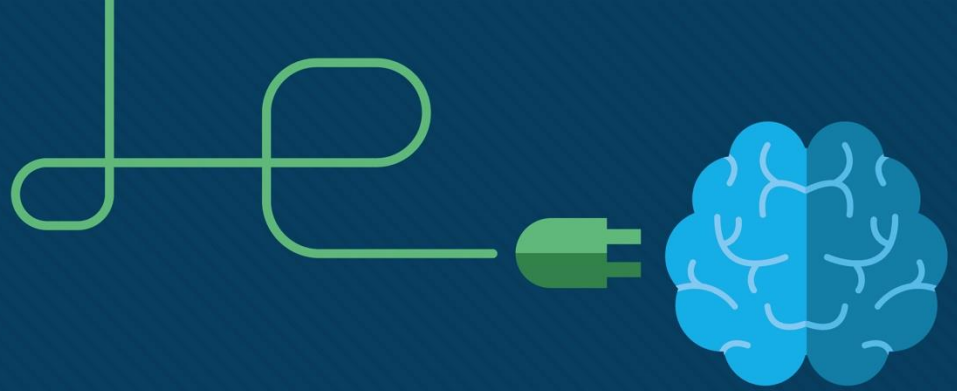
- FTP requires two connections between the client and the server, one for commands and replies, the other for the actual file transfer:
  - The client establishes the first connection to the server for control traffic using TCP port 21.
  - The client establishes the second connection to the server for the actual data transfer using TCP port 20.



Server Message Block

- The Server Message Block (SMB) is a client/server file sharing protocol:
  - SMB file-sharing and print services have become the mainstay of Microsoft networking.
  - Clients establish a long-term connection to servers and can access the resources on the server as if the resource is local to the client host.





The Transport Layer

Objectives

The Transport Layer

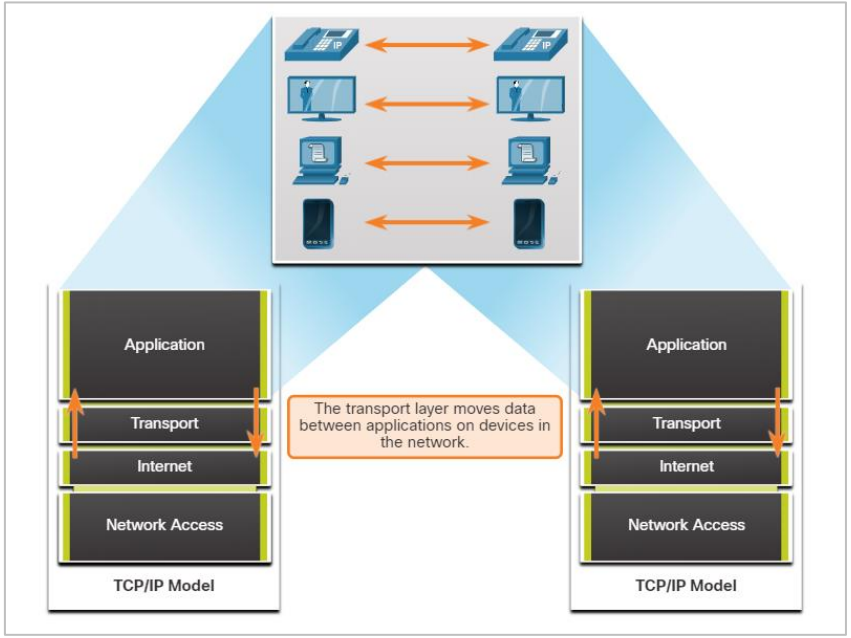
Explain how transport layer protocols support network functionality.

Topic Title	Topic Objective
Transport Layer Characteristics	Explain how transport layer protocols support network communication.
Transport Layer Session Establishment	Explain how the transport layer establishes communication sessions.
Transport Layer Reliability	Explain how the transport layer establishes reliable communications.

Transport Layer Characteristics

Role of the Transport Layer

- The transport layer is responsible for logical communications between applications running on different hosts.
- As shown in the figure, the transport layer is the link between the application layer and the lower layers that are responsible for network transmission.
- The transport layer has no knowledge of the destination host type, the type of media for which the data must travel, the path taken by the data, the congestion on a link, or the size of the network.
- The transport layer includes two protocols, Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

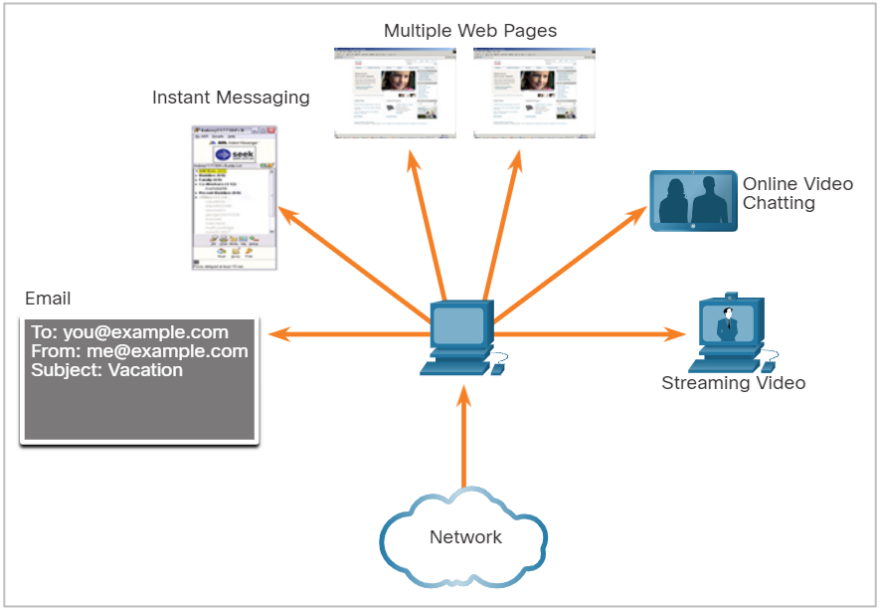


Transport Layer Responsibilities

The transport layer has many responsibilities.

Tracking Individual Conversations

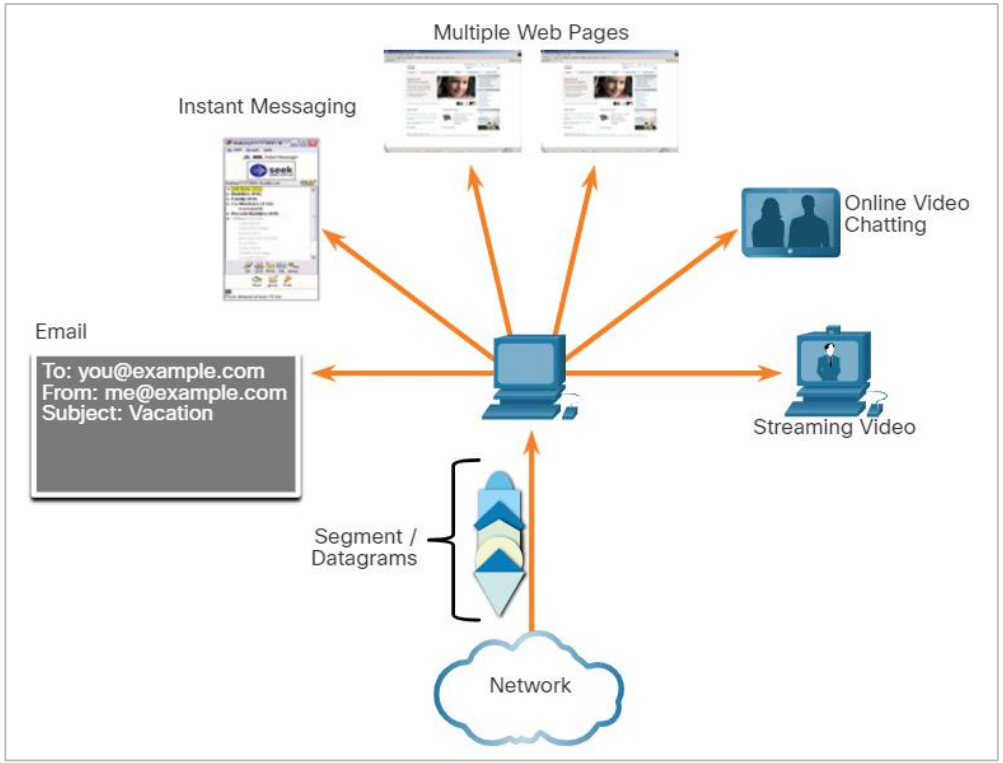
- Each set of data flowing between a source application and a destination application is known as a conversation and is tracked separately.
- It is the responsibility of the transport layer to maintain and track these multiple conversations.
- As shown in the figure, a host may have multiple applications that are communicating across the network simultaneously.
- Most networks have a limitation on the amount of data that can be included in a single packet. Data must be divided into manageable pieces.



Transport Layer Responsibilities (Contd.)

Segmenting Data and Reassembling Segments

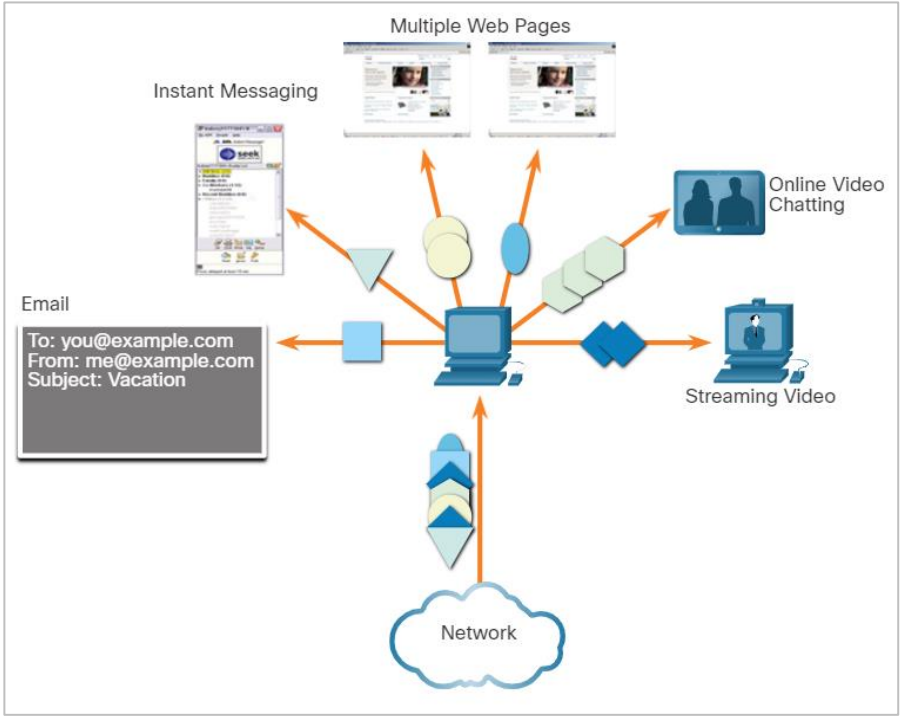
- It is the transport layer responsibility to divide the application data into appropriately sized blocks.
- Depending on the transport layer protocol used, the transport layer blocks are called either segments or datagrams.
- The figure shows the transport layer using different blocks for each conversation.
- The transport layer divides the data into smaller blocks (segments or datagrams) that are easier to manage and transport.



Transport Layer Responsibilities (Contd.)

Add Header Information

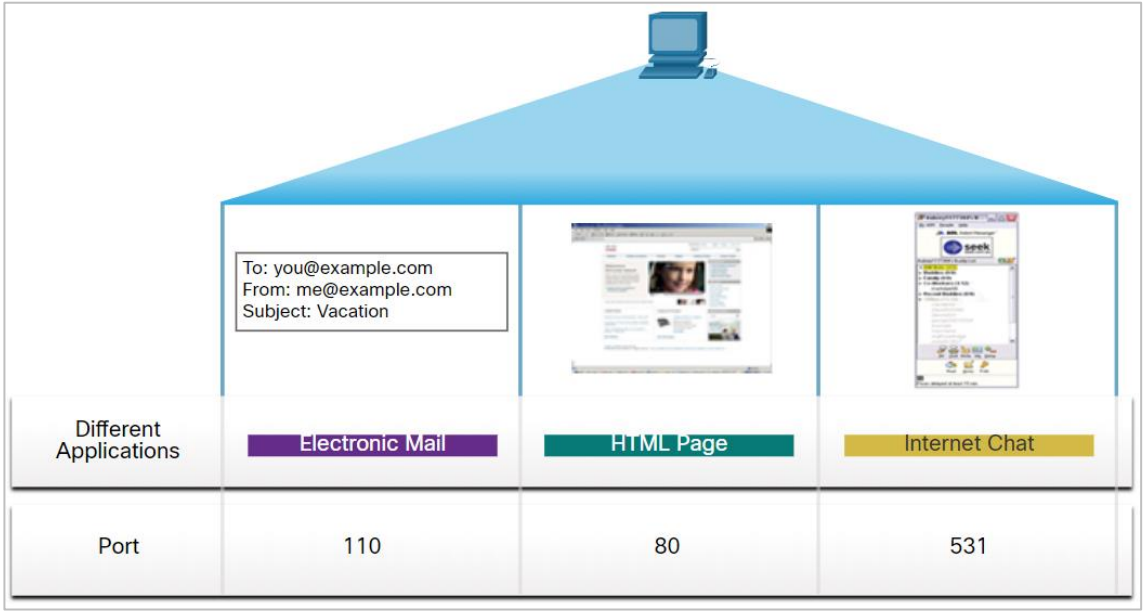
- The transport layer protocol also adds header information containing binary data organized into several fields to each block of data.
- The values in these fields enable various transport layer protocols to perform different functions in managing data communication.
- The header information is used by the receiving host to reassemble the blocks of data into a complete data stream for the receiving application layer program.
- The transport layer ensures that even with multiple application running on a device, all applications receive the correct data.



Transport Layer Responsibilities (Contd.)

Identifying the Applications

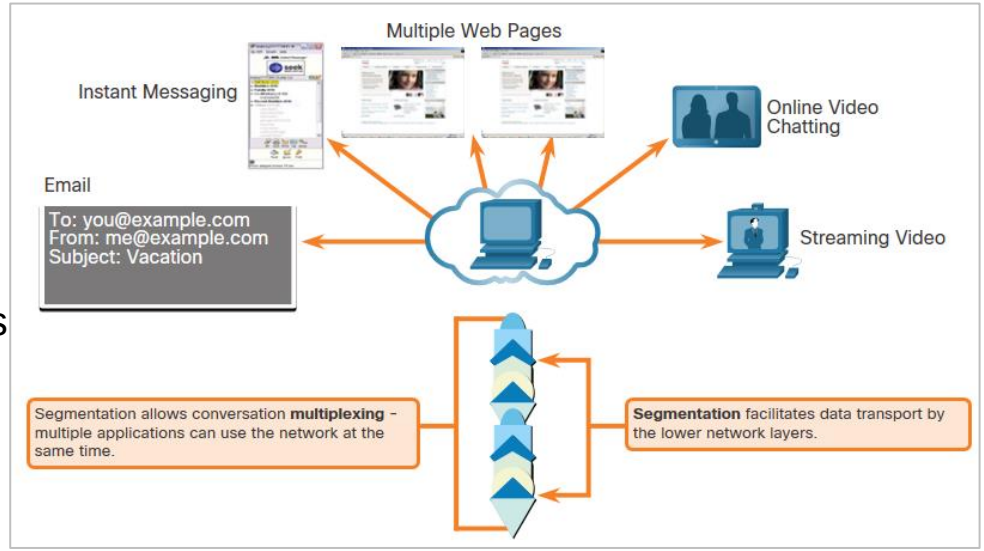
- The transport layer must be able to separate and manage multiple communications with different transport requirement needs.
- To pass data streams to the proper applications, the transport layer identifies the target application using an identifier called a port number.
- As shown in the figure, each software process that needs to access the network is assigned a port number unique to that host.



Transport Layer Responsibilities (Contd.)

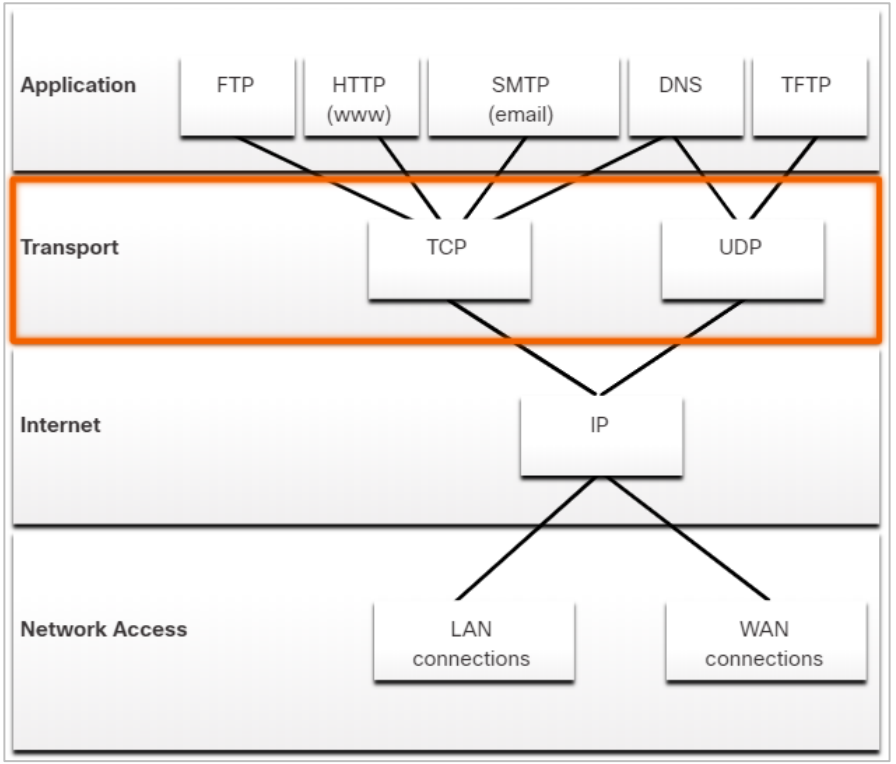
Conversation Multiplexing

- Sending some types of data across a network, as one complete communication stream, can consume all the available bandwidth.
- This prevents other communication conversations from occurring at the same time and also make error recovery and retransmission of damaged data difficult.
- As shown in the figure, the transport layer uses segmentation and multiplexing to enable different communication conversations to be interleaved on the same network.
- Error checking can be performed on the data in the segment, to determine if the segment was altered during transmission.



Transport Layer Protocols

- IP is concerned only with the structure, addressing, and routing of packets.
- IP does not specify how the delivery or transportation of the packets takes place.
- Transport layer protocols (TCP and UDP) 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.
- Different applications have different transport reliability requirements. Therefore, TCP/IP provides two transport layer protocols, as shown in the figure.



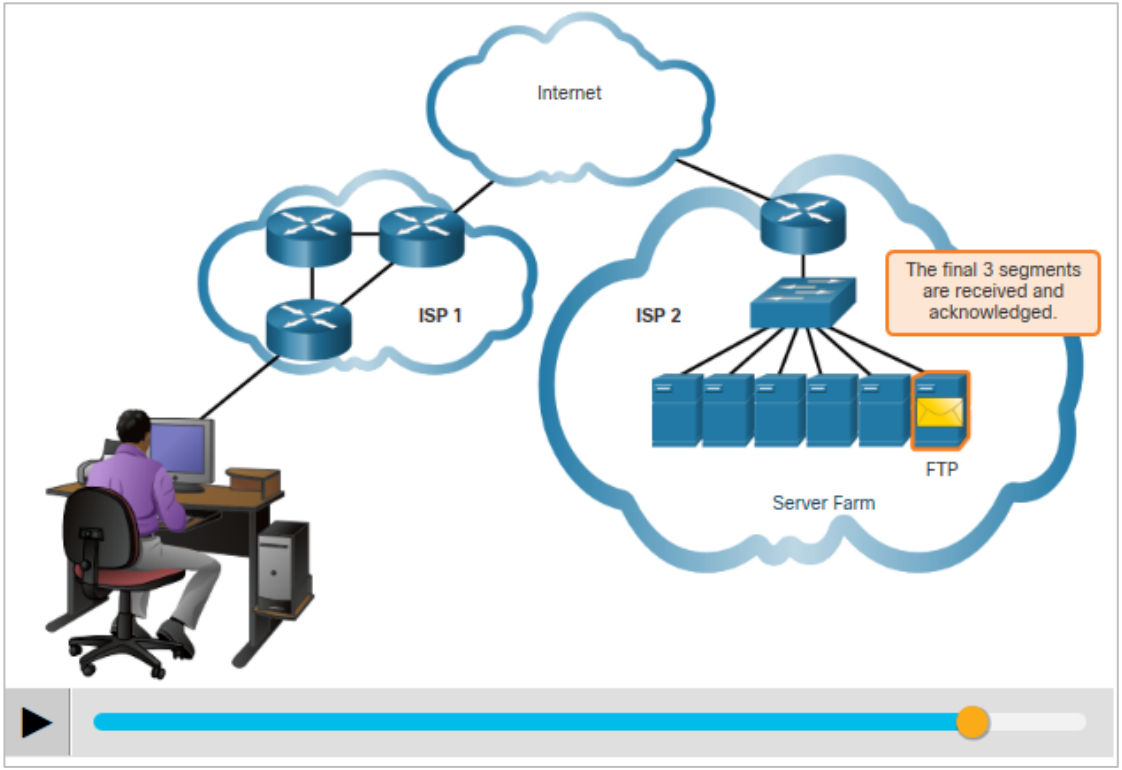
Transmission Control Protocol (TCP)

- TCP is considered a reliable, full-featured transport layer protocol, which ensures that all of the data arrives at the destination.
- TCP includes fields which ensure the delivery of the application data. These fields require additional processing by the sending and receiving hosts.
- TCP transport is analogous to sending packages that are tracked from source to destination.
- TCP provides reliability and flow control using these 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

***Note**: TCP divides data into segments.*

Transmission Control Protocol (TCP) (Contd.)

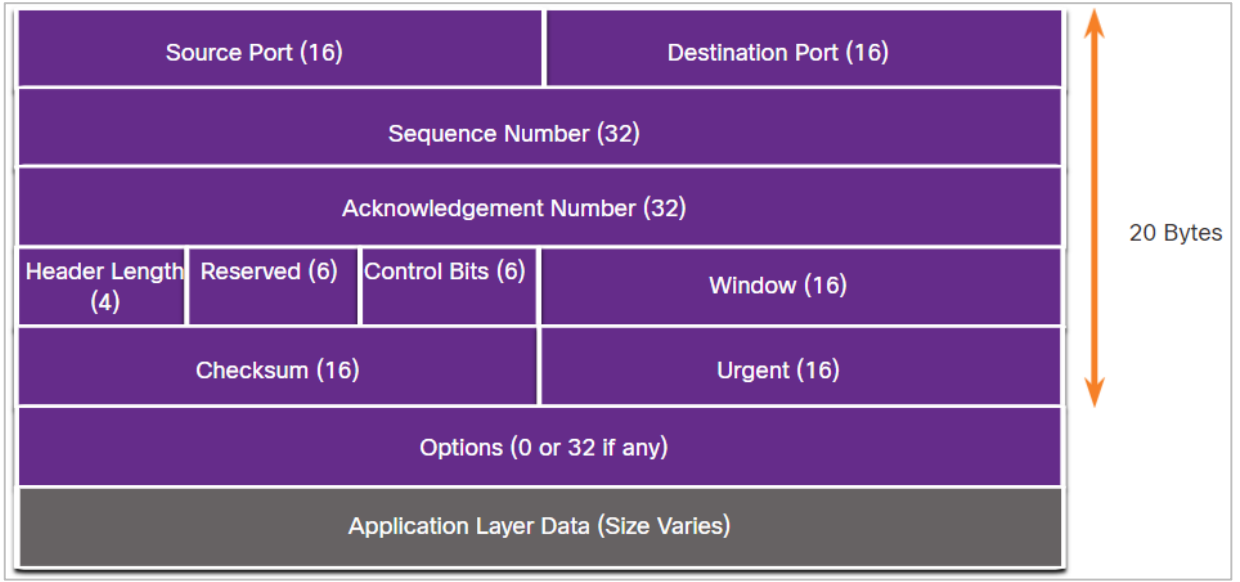
In order to maintain the state of a conversation and track the information, TCP must first establish a connection between the sender and the receiver. This is why TCP is known as a connection-oriented protocol.



TCP Header

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

- To track the state of a session, TCP records which information it has sent and which information has been acknowledged.
- The stateful session begins with the session establishment and ends with the session termination.
- A TCP segment adds 20 bytes (160 bits) of overhead when encapsulating the application layer data. The figure shows the fields in a TCP header.



TCP Header Fields

The table identifies and describes the ten fields in a TCP header.

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 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.

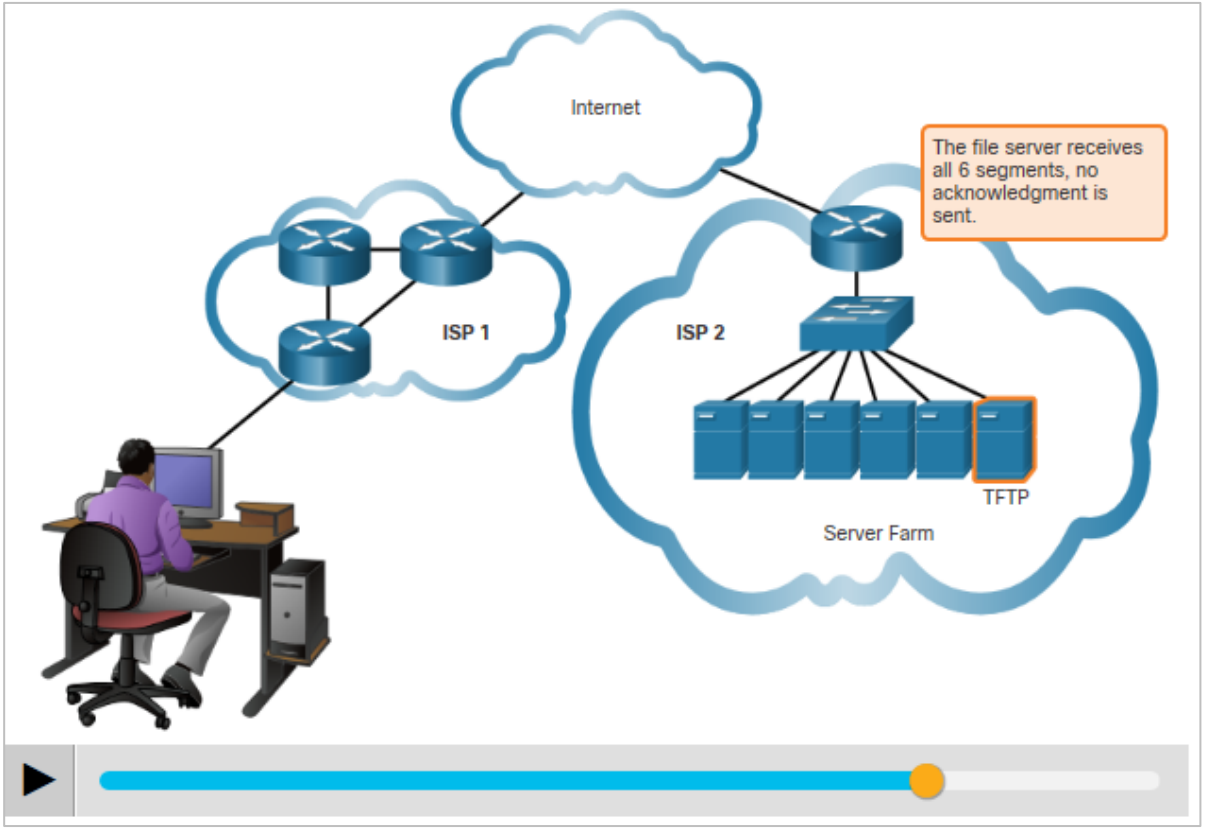
User Datagram Protocol (UDP)

- UDP is a simpler transport layer protocol than TCP.
- It does not provide reliability and flow control, which means it requires fewer header fields.
- The sender and the receiver UDP processes do not have to manage reliability and flow control, this means UDP datagrams can be processed faster than TCP segments.
- UDP provides the basic functions for delivering datagrams between the appropriate applications, with very little overhead and data checking.
- UDP is a connectionless protocol. Because UDP does not provide reliability or flow control, it does not require an established connection.
- UDP is also known as a stateless protocol. Because UDP does not track information sent or received between the client and server.

***Note**: UDP divides data into datagrams that are also referred to as segments.*

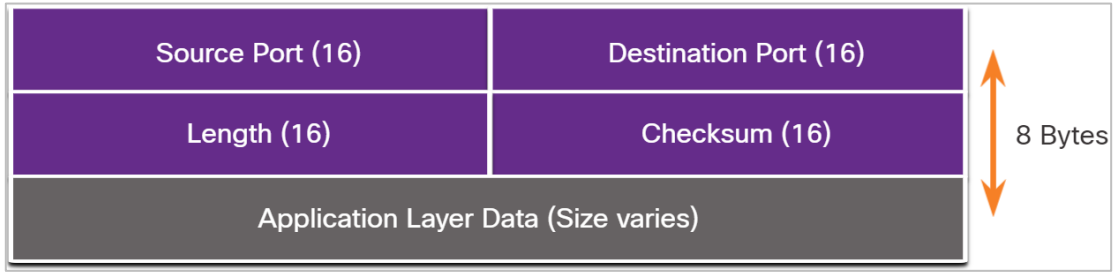
User Datagram Protocol (UDP) (Contd.)

- UDP is also known as a best-effort delivery protocol because there is no acknowledgment that the data is received at the destination.
- UDP is like placing a regular, nonregistered, letter in the mail. The sender of the letter is not aware of the availability of the receiver to receive the letter. Nor is the post office responsible for tracking the letter or informing the sender if the letter does not arrive at the final destination.



UDP Header

- UDP is a stateless protocol meaning neither the client, nor the server, tracks the state of the communication session. If reliability is required when using UDP as the transport protocol, it must be handled by the application.
- The requirements for delivering live video and voice over the network is the data continues to flow quickly. Live video and voice applications can tolerate some data loss and are perfectly suited to UDP.
- The blocks of communication in UDP are called datagrams, or segments. These datagrams are sent as best effort by the transport layer protocol.
- The UDP header is only has four fields and requires 8 bytes (64 bits). The figure shows the fields in a UDP header.



UDP Header Fields

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

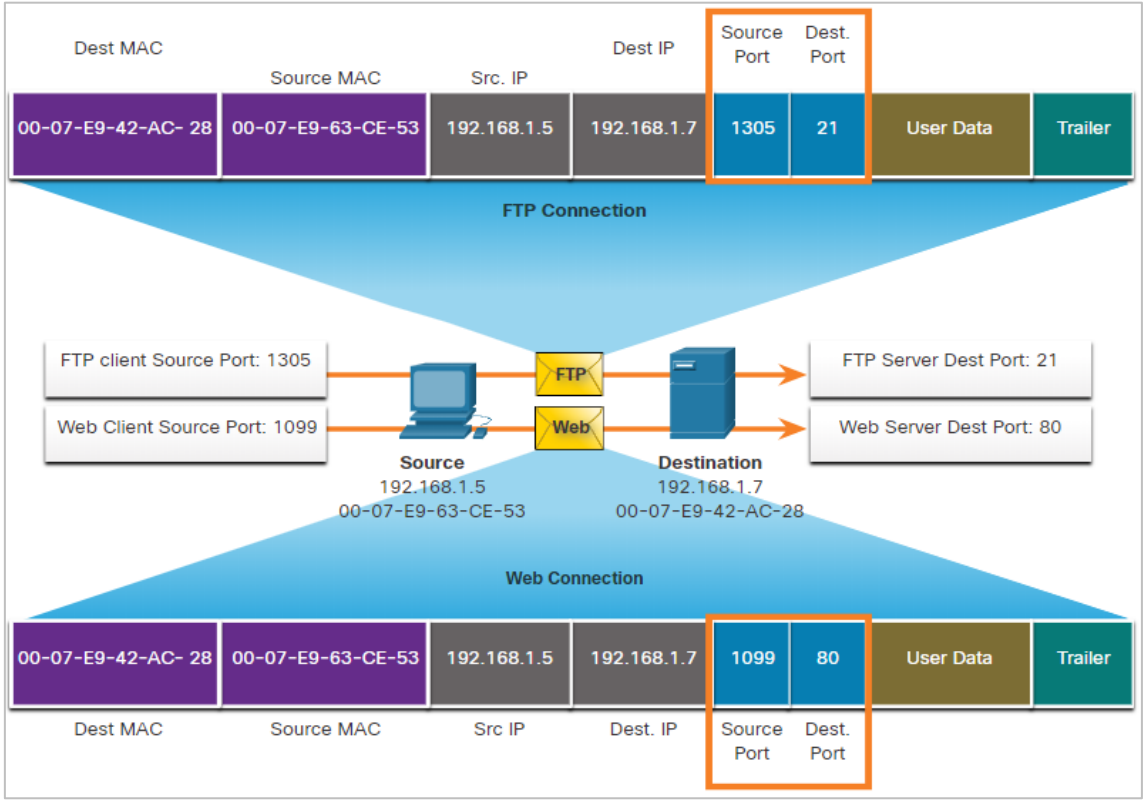
UDP Header Field	Description
Source Port	A 16-bit field used to identify the source application by port number.
Destination Port	A 16-bit field used to identify the destination application by port number.
Sequence Number	A 32-bit field used for data reassembly purposes.
Length	A 16-bit field that indicates the length of the UDP datagram header.
Checksum	A 16-bit field used for error checking of the datagram header and data.

Socket Pairs

- The source and destination ports are placed within the segment. The segments are then encapsulated within an IP packet.
- The IP packet contains the IP address of the source and destination. 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.
- The source port number acts as a return address for the requesting application.
- The transport layer keeps track of this port and the application that initiated the request so that when a response is returned, it can be forwarded to the correct application.

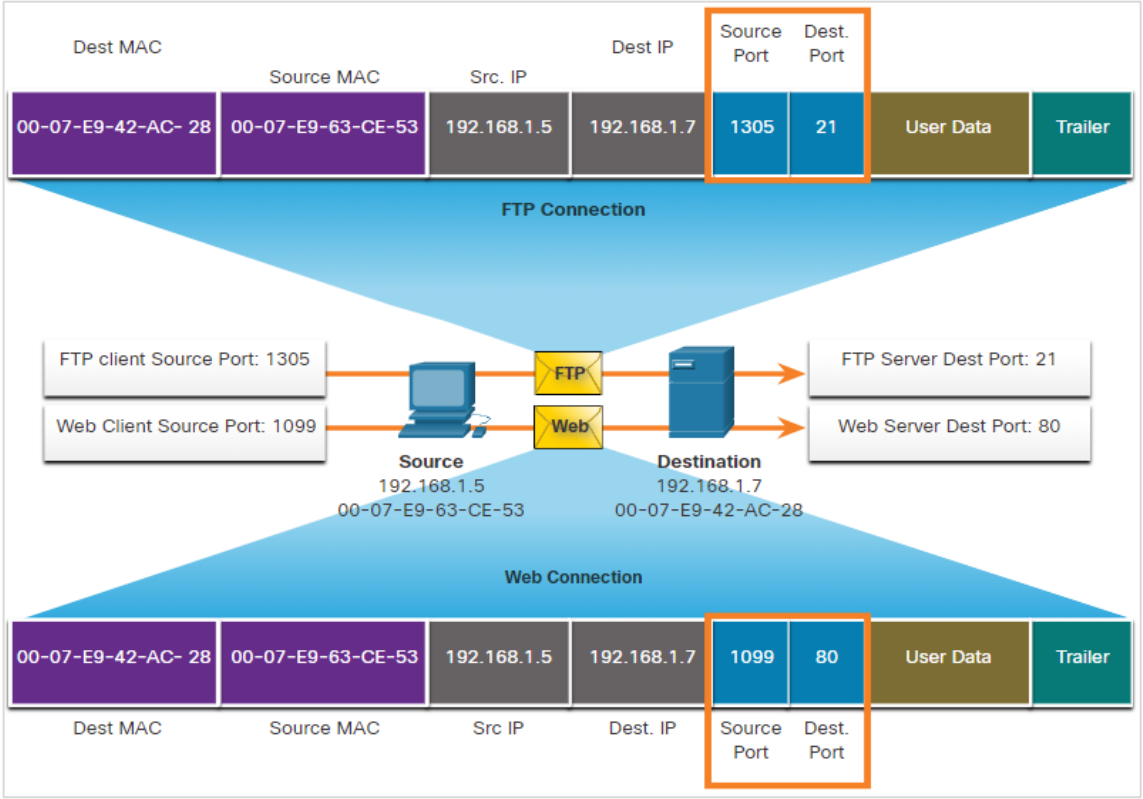
Socket Pairs (Contd.)

- In the figure, the PC is simultaneously requesting FTP and web services from the destination server.
- The FTP request generated by the PC includes the Layer 2 MAC addresses and the Layer 3 IP addresses. The request also identifies the source port number 1305 and destination port, identifying the FTP services on port 21.
- The host also has requested a web page from the server using the same Layer 2 and Layer 3 addresses.



Socket Pairs (Contd.)

- It is using the source port number 1099 and destination port identifying the web service on port 80.
- The socket is used to identify the server and service being requested by the client.
- A client socket with 1099 representing the source port number might be 192.168.1.5:1099. The socket on a web server might be 192.168.1.7:80. Together, these two sockets combine to form a *socket pair*: 192.168.1.5:1099, 192.168.1.7:80



Transport Layer Session Establishment

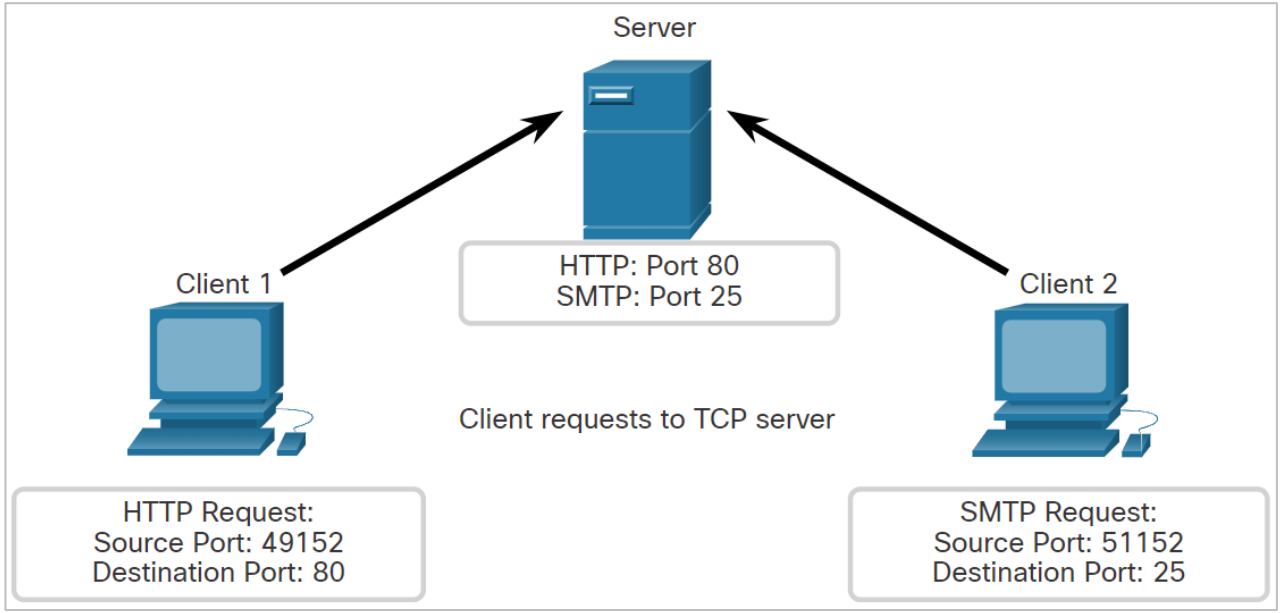
TCP Server Processes

- Each application process running on a server is configured to use a port number. The port number is either automatically assigned or configured manually by a system administrator.
- An individual server cannot have two services assigned to the same port number within the same transport layer services.
- A host running a web server application and a file transfer application cannot have both configured to use the same port, such as TCP port 80.
- 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.
- There can be many ports open simultaneously on a server, one for each active server application.

TCP Server Processes (Contd.)

Clients Sending TCP Requests

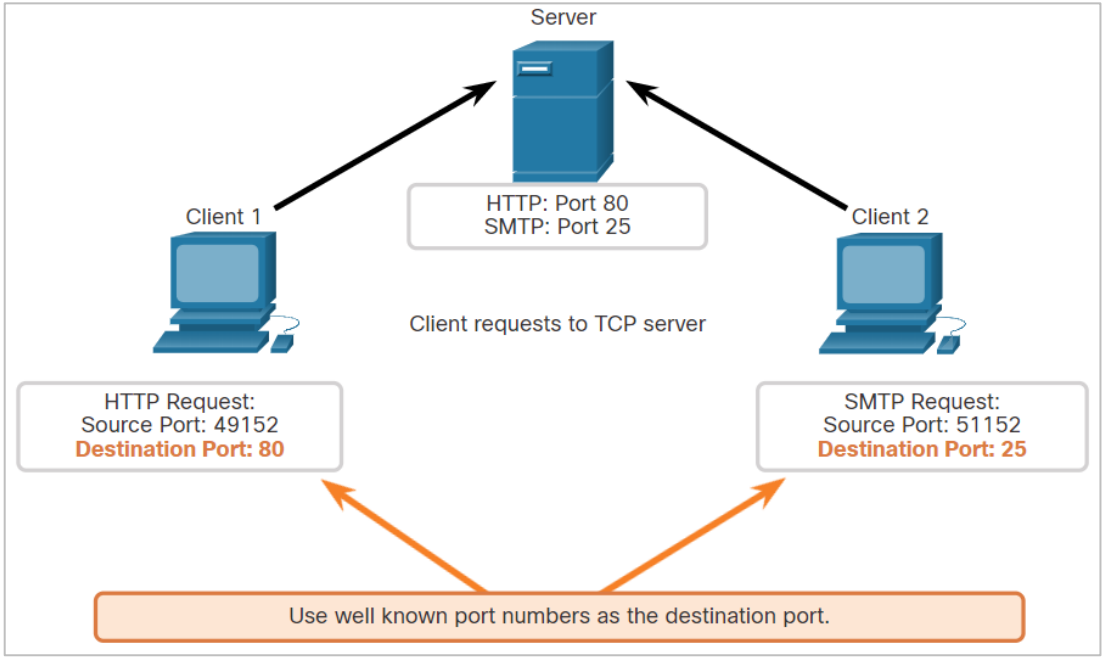
Client 1 is requesting web services and Client 2 is requesting email service of the same sever.



TCP Server Processes (Contd.)

Request Destination Ports

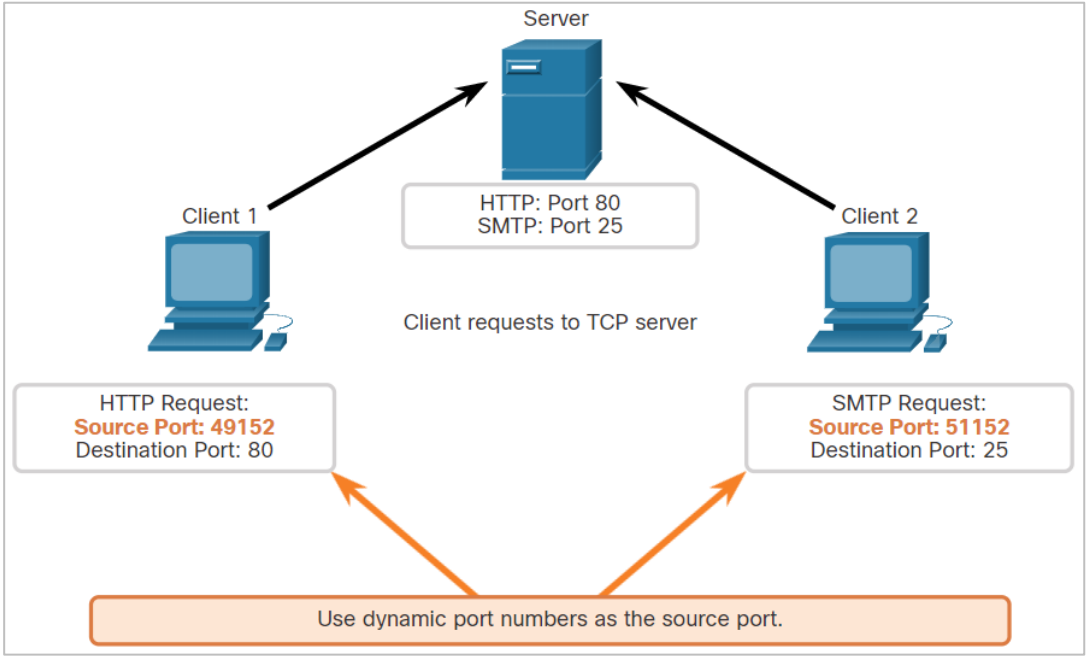
Client 1 is requesting web services using well-known destination port 80 (HTTP) and Client 2 is requesting email service using well-known port 25 (SMTP).



TCP Server Processes (Contd.)

Request Source Ports

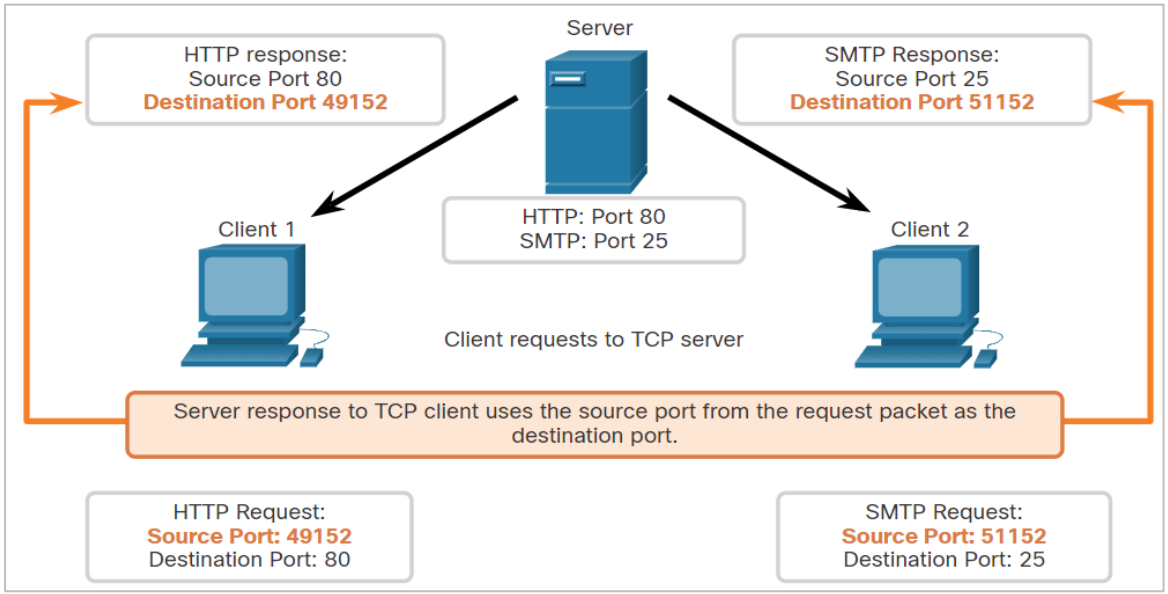
Client requests dynamically generate a source port number. In this case, Client 1 is using source port 49152 and Client 2 is using source port 51152.



TCP Server Processes (Contd.)

Response Destination Ports

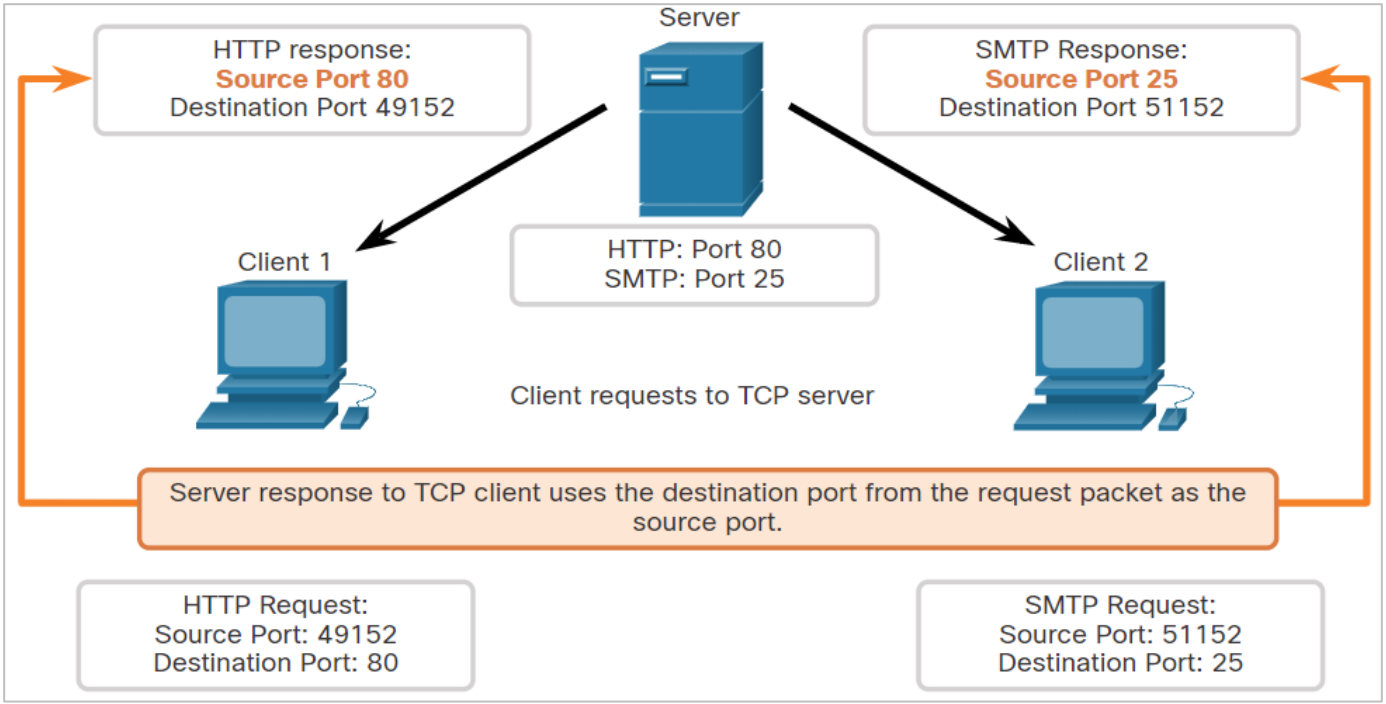
When the server responds to the client requests, it reverses the destination and source ports of the initial request. Notice that the Server response to the web request now has destination port 49152 and the email response now has destination port 51152.



TCP Server Processes (Contd.)

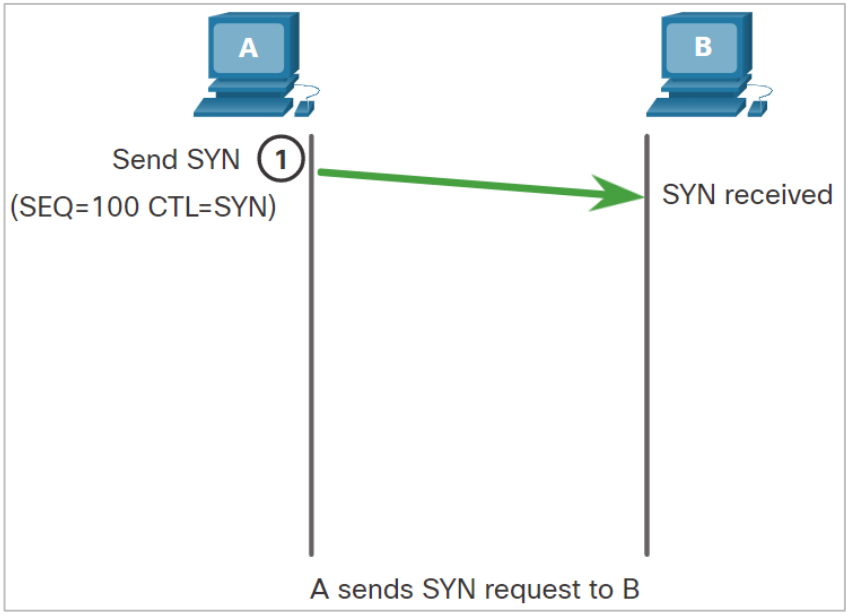
Response Source Ports

The source port in the server response is the original destination port in the initial requests.



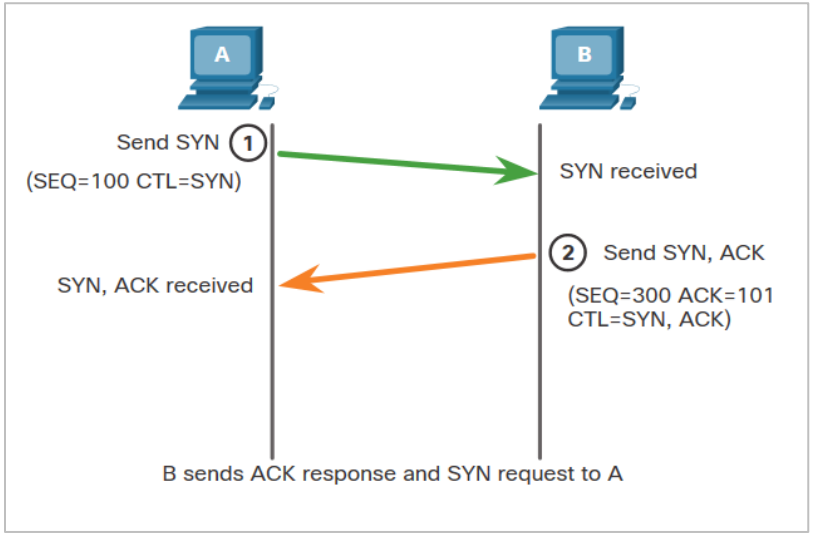
TCP Connection Establishment

- In TCP connections, the host client establishes the connection with the server using the three-way handshake process.
- The three-way handshake validates that the destination host is available to communicate.
- The TCP connection establishment steps are:
  - **Step 1. SYN:** The initiating client requests a client-to-server communication session with the server.

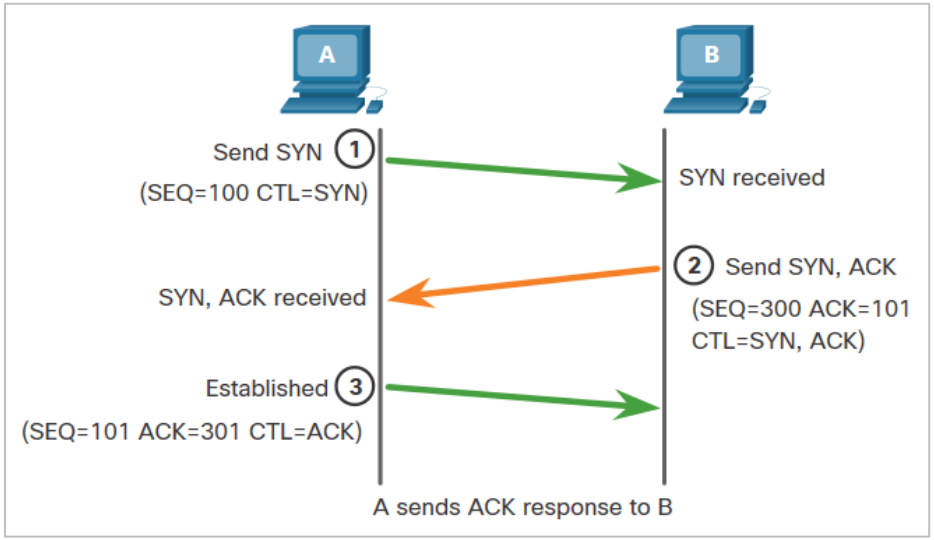


TCP Connection Establishment (Contd.)

Step 2. ACK and SYN: The server acknowledges the client-to-server communication session and requests a server-to-client communication session.



Step 3. ACK: The initiating client acknowledges the server-to-client communication session.



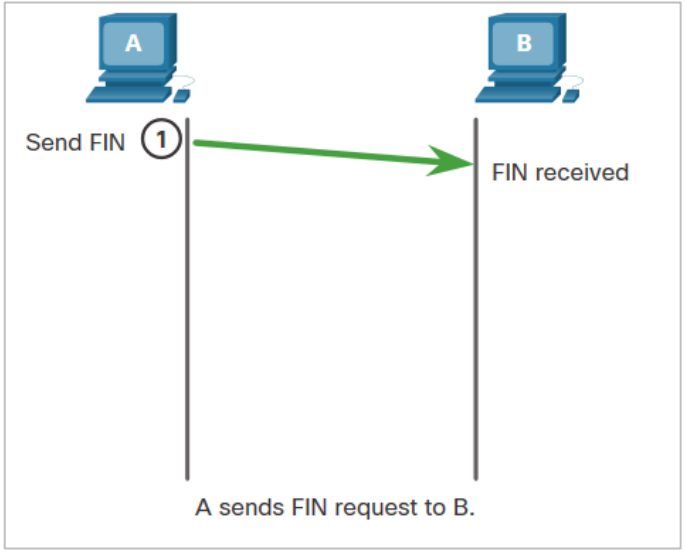
Session Termination

- To close a connection, the Finish (FIN) control flag must be set in the segment header.
- To end each one-way TCP session, a two-way handshake, consisting of a FIN segment and an Acknowledgment (ACK) segment, is used.
- Therefore, to terminate a single conversation supported by TCP, four exchanges are needed to end both sessions. Either the client or the server can initiate the termination.
- The terms client and server are used as a reference for simplicity, but any two hosts that have an open session can initiate the termination process.
- When all segments have been acknowledged, the session is closed.

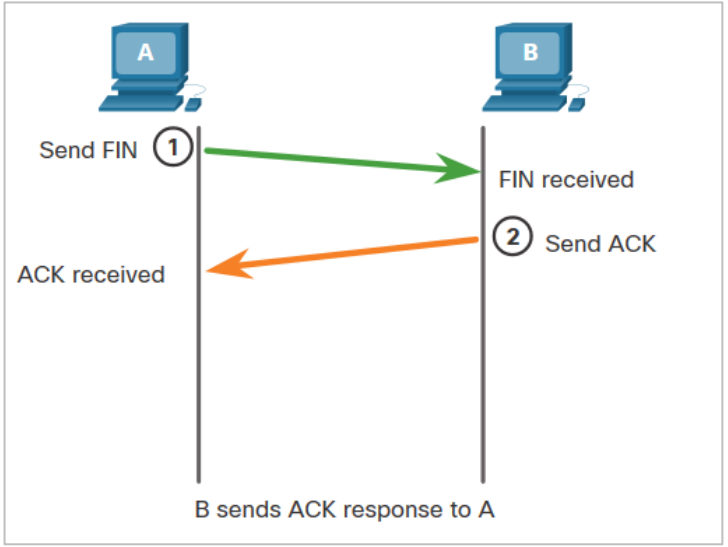
Session Termination (Contd.)

The session termination steps are:

Step 1. FIN: When the client has no more data to send in the stream, it sends a segment with the FIN flag set.

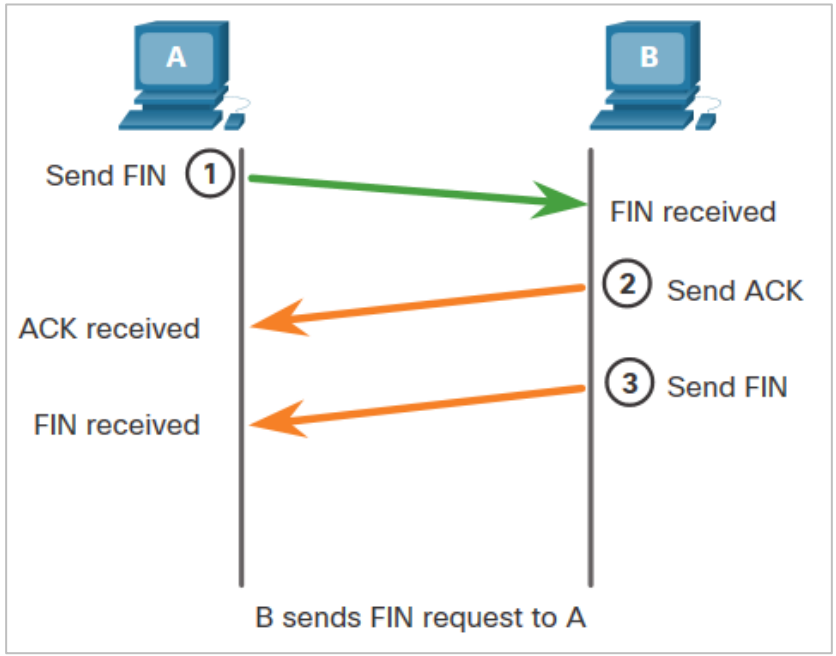


Step 2. ACK: The server sends an ACK to acknowledge the receipt of the FIN to terminate the session from client to server.

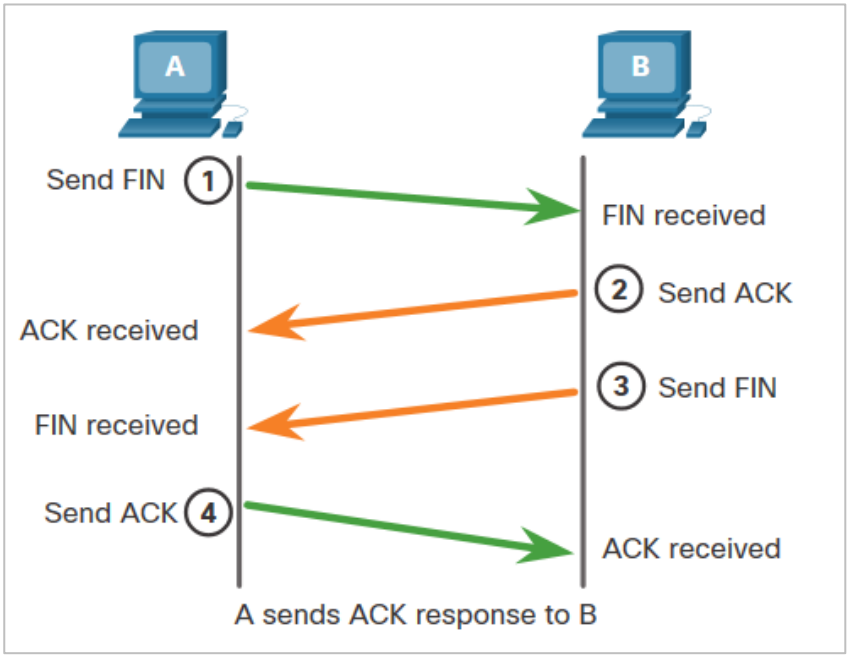


Session Termination (Contd.)

Step 3. FIN: The server sends a FIN to the client to terminate the server-to-client session.



Step 4. ACK: The client responds with an ACK to acknowledge the FIN from the server.



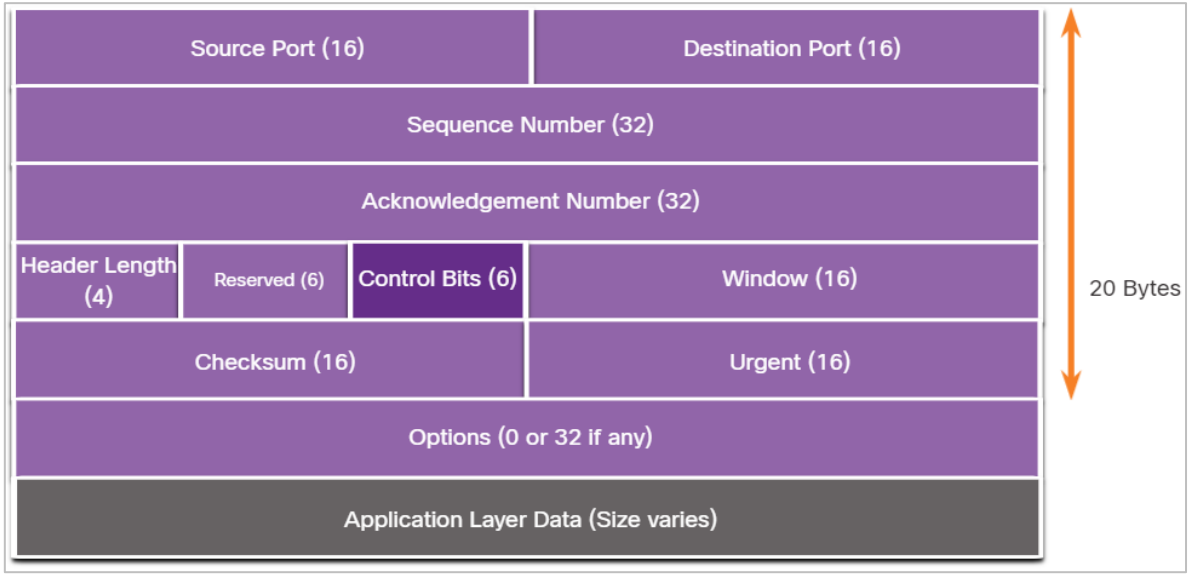
TCP Three-way Handshake Analysis

- Hosts maintain state, track each data segment within a session, and exchange information about the data is received using the information in the TCP header.
- TCP is a full-duplex protocol, where each connection represents two one-way communication sessions. To establish the connection, the hosts perform a three-way handshake. As shown in the figure, control bits in the TCP header indicate the progress and status of the connection.
- The functions of the three-way handshake are:
  - It establishes that the destination device is present on the network.
  - It verifies that the destination device has an active service and is accepting requests on the destination port number that the initiating client intends to use.
  - It informs the destination device that the source client intends to establish a communication session on that port number.
- After the communication is completed the sessions are closed, and the connection is terminated. The connection and session mechanisms enable TCP reliability function.

TCP Three-way Handshake Analysis (Contd.)

The six bits in the Control Bits field of the TCP segment header are also known as flags. A flag is a bit that is set to either on or off. The six control bits 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



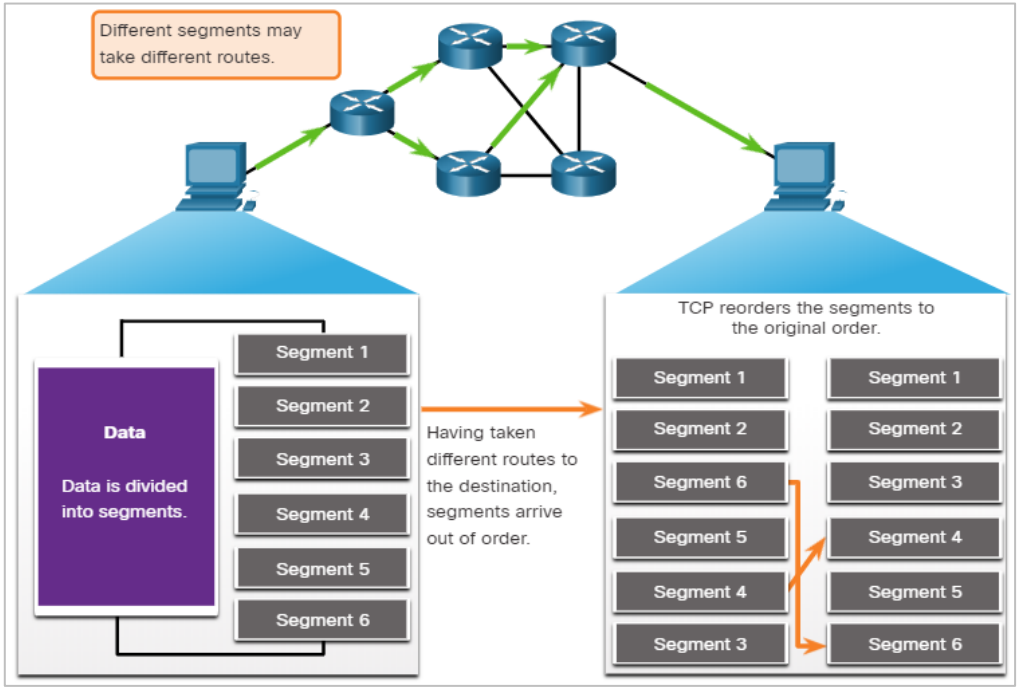
Transport Layer Reliability

TCP Reliability - Guaranteed and Ordered Delivery

- There may be times when either TCP segments do not arrive at their destination or arrive out of order.
- For the original message to be understood by the recipient, 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 for each packet to achieve this goal. The sequence number represents the first data byte of the TCP segment.
- During session setup, an initial sequence number (ISN) is set, which represents the starting value of the bytes that are transmitted to the receiving application.
- As data is transmitted during the session, the sequence number is incremented by the number of bytes that have been transmitted.
- This data byte tracking enables each segment to be uniquely identified and acknowledged. Missing segments can then be identified.
- The ISN is effectively a random number which prevents certain types of malicious attacks.

TCP Reliability - Guaranteed and Ordered Delivery (Contd.)

- Segment sequence numbers indicate how to reassemble and reorder received segments, as shown in the figure.
- The receiving TCP process places the data from a segment into a receiving buffer.
- Segments are then placed in the proper sequence order and passed to the application layer when reassembled.
- Any segments that arrive with sequence numbers that are out of order are held for later processing.
- Then, when the segments with the missing bytes arrives, these segments are processed in order.

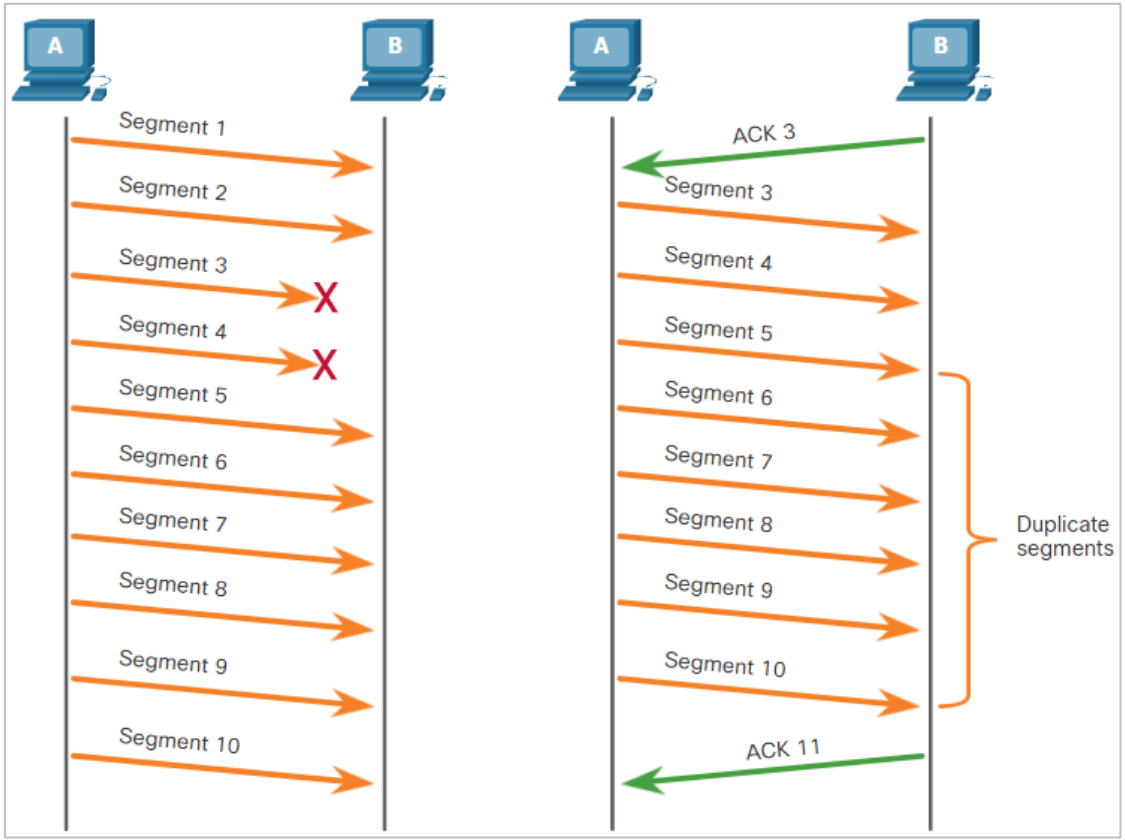


TCP Reliability - Data Loss and Retransmission

- TCP provides methods of managing the segment losses by retransmitting the segments for unacknowledged data.
- The sequence (SEQ) number and acknowledgement (ACK) number are used together to confirm receipt of the bytes of data contained in the transmitted segments.
- The SEQ number identifies the first byte of data in the segment being transmitted.
- TCP uses the ACK number sent back to the source to indicate the next byte that the receiver expects to receive. This is called expectational acknowledgement.
- Prior to later enhancements, TCP could only acknowledge the next byte expected.

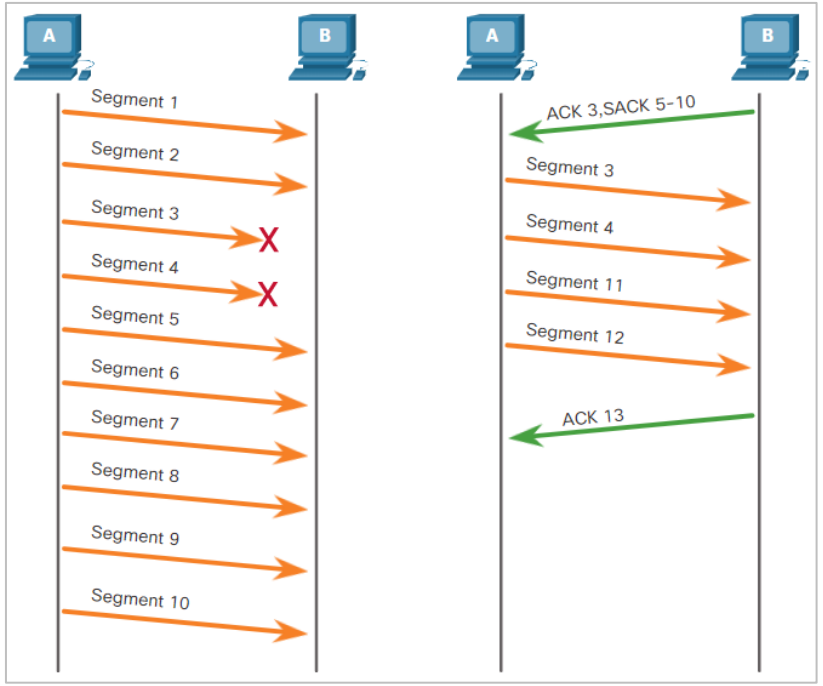
TCP Reliability - Data Loss and Retransmission (Contd.)

- In the figure, Host A sends segments 1 through 10 to host B. If all the segments arrive except segments 3 and 4, host B would reply with acknowledgment specifying that the next segment expected is segment 3.
- Host A has no idea if any other segments arrived or not. It would resend segments 3 through 10.
- If all the resent segments arrived successfully, segments 5 through 10 would be duplicates. This can lead to delays, congestion, and inefficiencies.



TCP Reliability - Data Loss and Retransmission (Contd.)

- Host operating systems employ an optional TCP feature called selective acknowledgment (SACK), negotiated during the three-way handshake.
- If both hosts support SACK, the receiver can acknowledge which segments (bytes) were received including any discontinuous segments.
- The sending host would only need to retransmit the missing data.
- In the figure, host A sends segments 1 through 10 to host B.
- If all the segments arrive except for segments 3 and 4, host B can acknowledge that it has received segments 1 and 2 (ACK 3), and selectively acknowledge segments 5 through 10 (SACK 5-10). Host A would only need to resend segments 3 and 4.

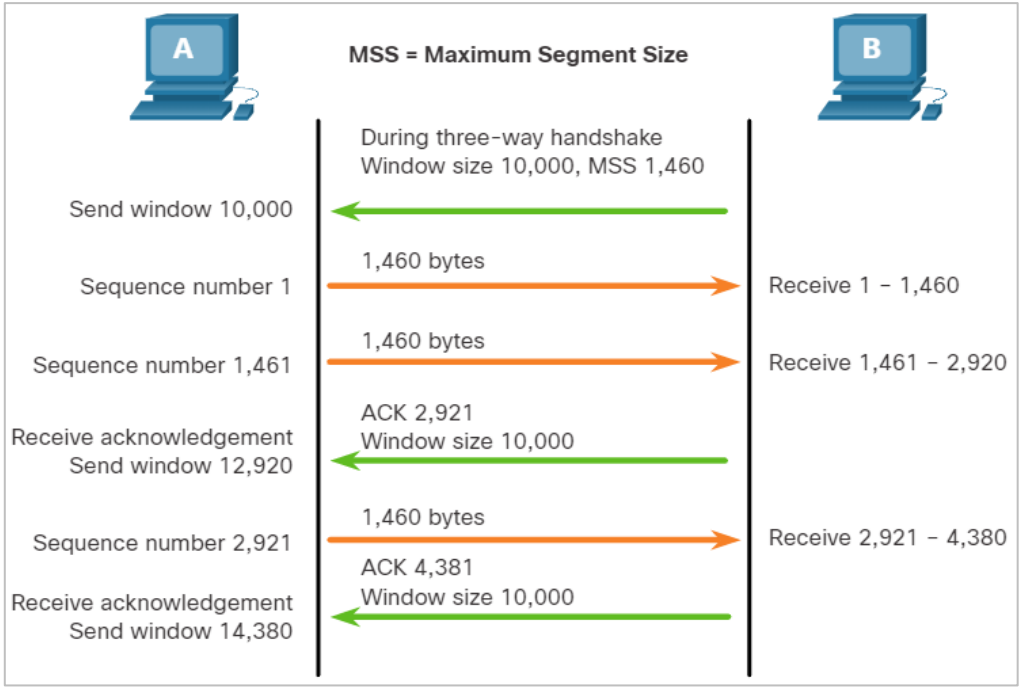


TCP Flow Control - Window Size and Acknowledgments

- TCP also provides mechanisms for flow control. Flow control is the amount of data that the destination can receive and process reliably.
- Flow control helps maintain the reliability of TCP transmission by adjusting the rate of data flow between source and destination for a given session.
- To accomplish this, the TCP header includes a 16-bit field called the window size.
- The window size that determines the number of bytes that can be sent before expecting an acknowledgment.
- The acknowledgment number is the number of the next expected byte.
- The window size is the number of bytes that the destination device of a TCP session can accept and process at one time.

TCP Flow Control - Window Size and Acknowledgments (Contd.)

- The figure shows an example of window size and acknowledgments.
- The window size is included in every TCP segment so the destination can modify the window size at any time depending on buffer availability.
- The initial window size is agreed upon when the TCP session is established during the three-way handshake.
- The source device must limit the number of bytes sent to the destination device based on the window size of the destination. Only after the source receives an acknowledgment, it can continue sending more data for the session.



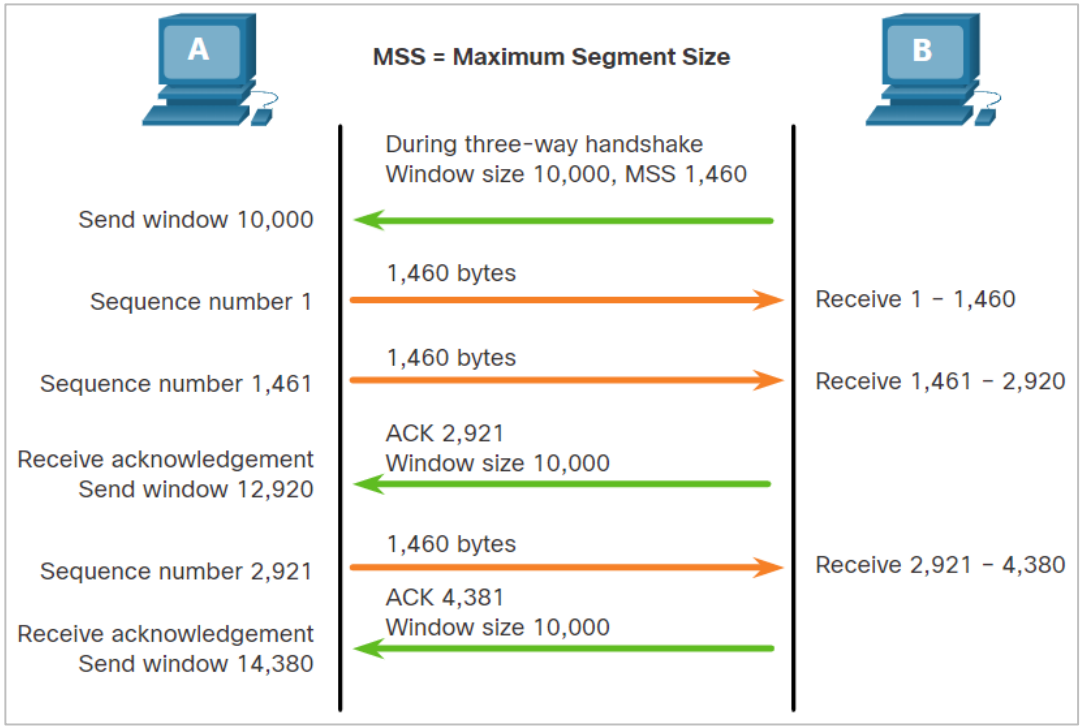
TCP Flow Control - Window Size and Acknowledgments (Contd.)

- The destination will not wait for all the bytes for its window size to be received before replying with an acknowledgment.
- As the bytes are received and processed, the destination will send acknowledgments to inform the source that it can continue to send additional bytes.
- A destination sending acknowledgments as it processes bytes received, and the continual adjustment of the source send window, is known as sliding windows.
- If the availability of the destination's buffer space decreases, it may reduce its window size to inform the source to reduce the number of bytes it should send without receiving an acknowledgment.

***Note**: Devices today use the sliding windows protocol. The receiver sends an acknowledgment after every two segments it receives. The advantage of sliding windows is that it allows the sender to continuously transmit segments, as long as the receiver is acknowledging previous segments.*

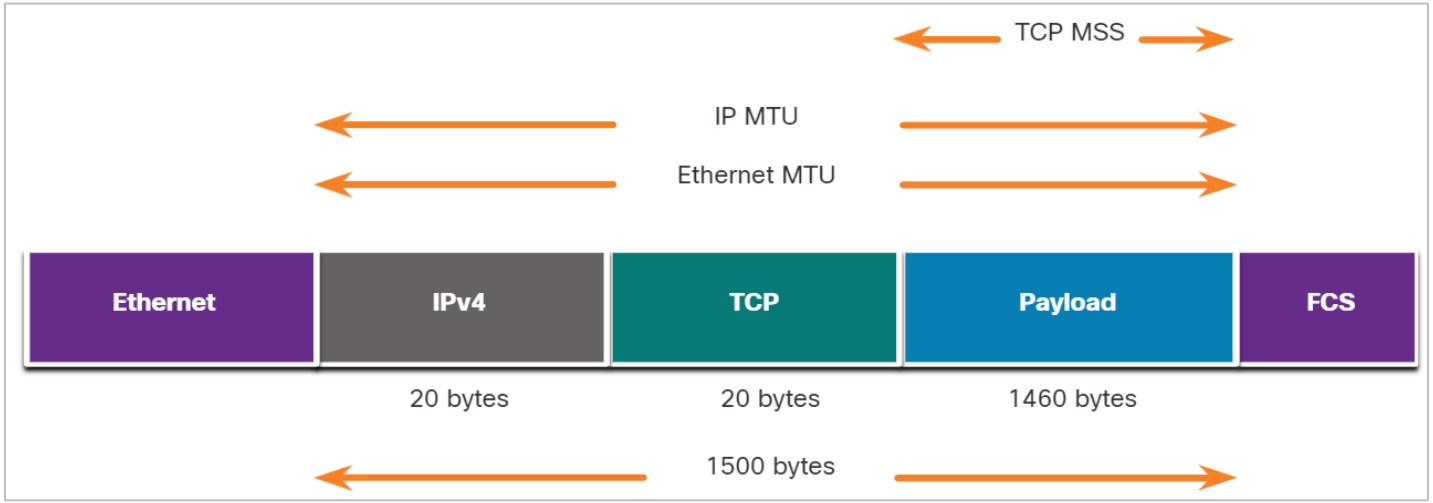
TCP Flow Control - Maximum Segment Size (MSS)

- In the figure, the source is transmitting 1,460 bytes of data within each TCP segment. This is the Maximum Segment Size (MSS) that the destination device can receive.
- The MSS is part of the options field in the TCP header that specifies the largest amount of data, in bytes, that a device can receive in a single TCP segment.
- The MSS size does not include the TCP header.
- The MSS is included during the three-way handshake.



TCP Flow Control - Maximum Segment Size (MSS) (Contd.)

- 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).
- On an Ethernet interface, the default MTU is 1500 bytes. Subtracting the IPv4 header of 20 bytes and the TCP header of 20 bytes, the default MSS size will be 1460 bytes, as shown in the figure.

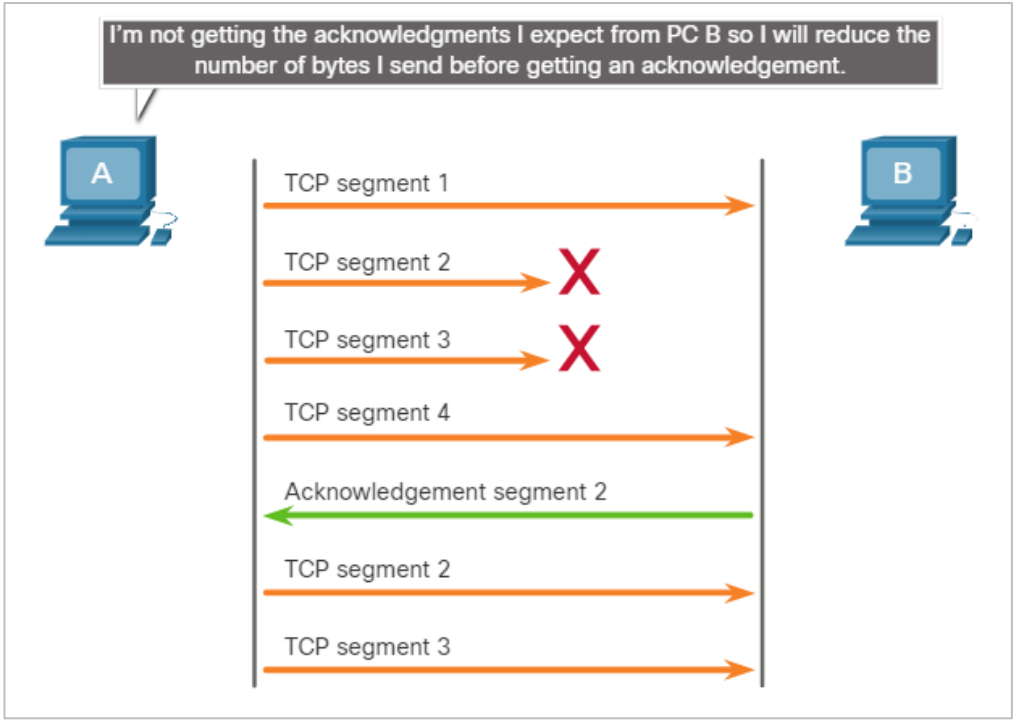


TCP Flow Control - Congestion Avoidance

- When congestion occurs on a network, it results in packets being discarded by the overloaded router.
- When packets containing TCP segments do not reach their destination, they are left unacknowledged.
- By determining the rate at which TCP segments are sent but not acknowledged, the source can assume a certain level of network congestion.
- Whenever there is congestion, retransmission of lost TCP segments from the source will occur.
- If the retransmission is not properly controlled, the additional retransmission of the TCP segments can make the congestion even worse.
- Not only are new packets with TCP segments introduced into the network, but the feedback effect of the retransmitted TCP segments that were lost will also add to the congestion.
- To avoid and control congestion, TCP employs several congestion handling mechanisms, timers, and algorithms.

TCP Flow Control - Congestion Avoidance (Contd.)

- If the source determines that the TCP segments are either not being acknowledged or not acknowledged in a timely manner, then it can reduce the number of bytes it sends before receiving an acknowledgment.
- As shown in the figure, PC A senses there is congestion and therefore, reduces the number of bytes it sends before receiving an acknowledgment from PC B.
- Acknowledgment numbers are for the next expected byte and not for a segment. The segment numbers used are simplified for illustration purposes.



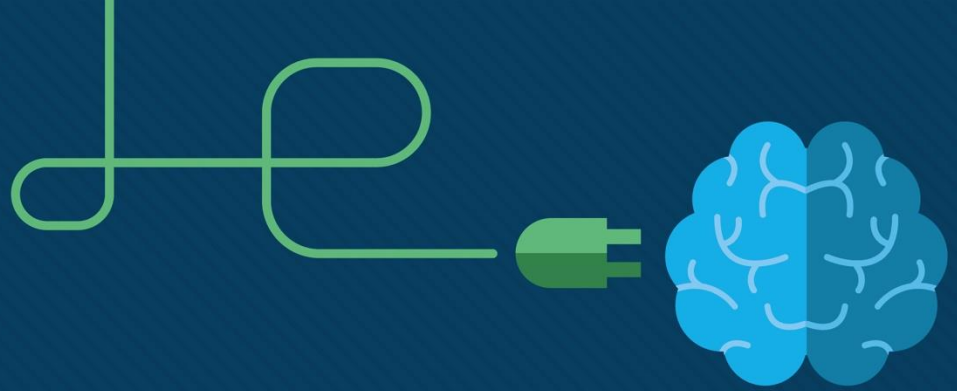
The Transport Layer Summary

The Transport Layer Summary

- The transport layer is the link between the application layer and the lower layers of the OSI model that are responsible for network transmission.
- The transport layer includes TCP and UDP. Transport layer protocols specify how to transfer messages between hosts and is responsible for managing reliability requirements of a conversation.
- The transport layer is responsible for tracking conversations (sessions), segmenting data and reassembling segments, adding segment header information, identifying applications, and conversation multiplexing.
- TCP is stateful and reliable. It acknowledges data, resends lost data, and delivers data in sequenced order. TCP is used for email and the web.
- UDP is stateless and fast. It has low overhead, does not requires acknowledgments, does not resend lost data, and processes data in the order in which it arrives. UDP is used for VoIP and DNS.

The Transport Layer Summary

- The TCP and UDP transport layer protocols use port numbers to manage multiple simultaneous conversations. This is why the TCP and UDP header fields identify a source and destination application port number.
- The three-way handshake establishes that the destination device is present on the network. It verifies that the destination device has an active service that is accepting requests on the destination port number that the initiating client intends to use.
- The six control bits flags are: URG, ACK, PSH, RST, SYN, and FIN and are used to identify the function of TCP messages that are sent.
- For the original message to be understood by the recipient, all the data must be received and the data in these segments must be reassembled into the original order.
- Host operating systems today typically employ an optional TCP feature called selective acknowledgment (SACK), which is negotiated during the three-way handshake.
- Flow control helps maintain the reliability of TCP transmission by adjusting the rate of data flow between source and destination.



Network Layer

Sections & Objectives

- Network Layer Protocols
  - Explain how network layer protocols and services support communications across data networks
    - Describe the purpose of the network layer in data communication.
    - Explain why the IPv4 protocol requires other layers to provide reliability.
    - Explain the role of the major header fields in the IPv4 packet.
    - Explain the role of the major header fields in the IPv6 packet.
- Routing
  - Explain how routers enable end-to-end connectivity in a small to medium-sized business network.
    - Explain how network devices use routing tables to direct packets to a destination network.
    - Compare a host routing table to a routing table in a router.

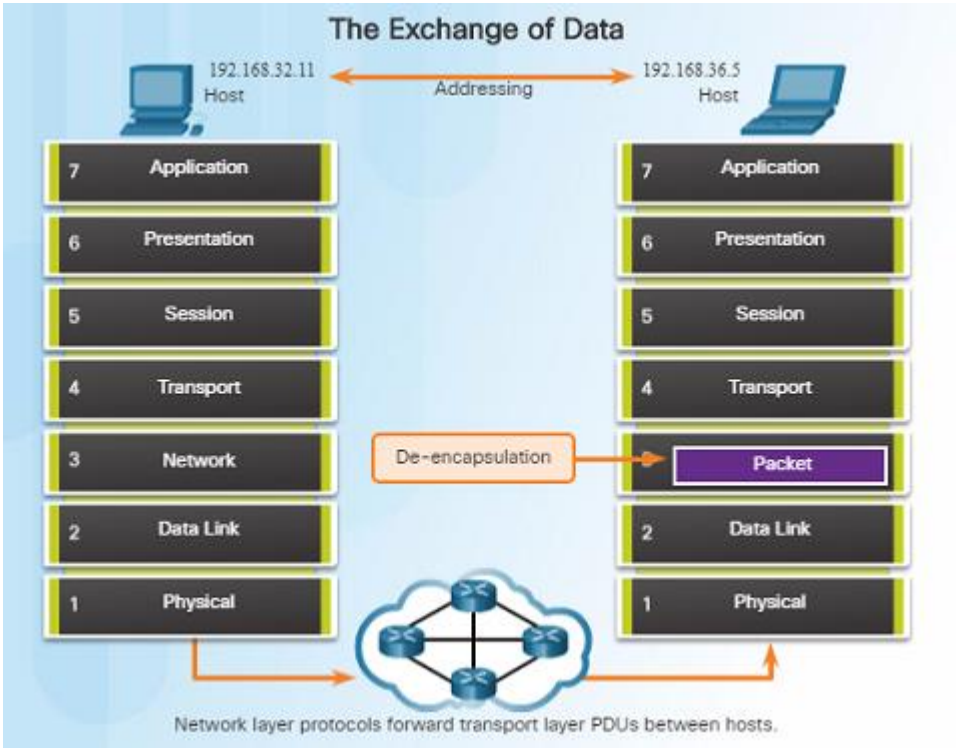
Sections & Objectives (Cont.)

- Routers
  - Explain how devices route traffic in a small to medium-sized business network
    - Describe the common components and interface of a router.
    - Describe the boot-up process of a Cisco IOS router.
- Configuring a Cisco Router
  - Configure a router with basic configurations.
    - Configure initial settings on a Cisco IOS router.
    - Configure two active interfaces on a Cisco IOS router.
    - Configure devices to use the default gateway

Network Layer Protocols

The Network Layer

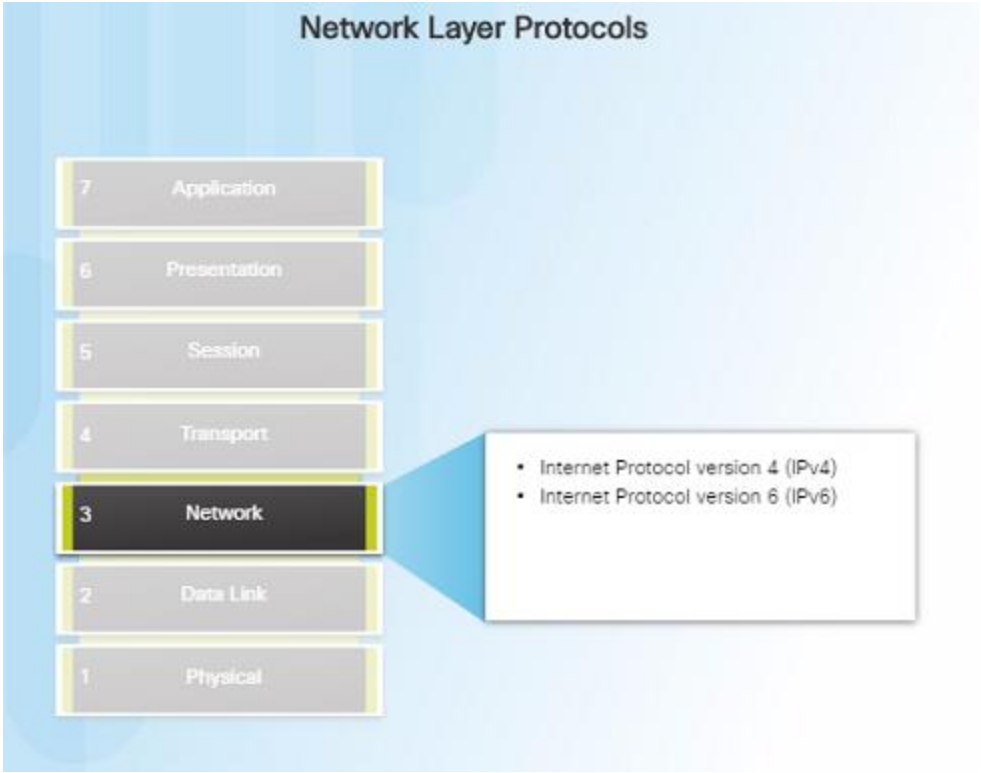
- The network layer, which resides at OSI Layer 3, provides services that allow end devices to exchange data across a network.
- The network layer uses four processes in order to provide end-to-end transport:
  - Addressing of end devices – IP addresses must be unique for identification purposes.
  - Encapsulation – The protocol data units from the transport layer are encapsulated by adding IP header information including source and destination IP addresses.
  - Routing – The network layer provides services to direct packets to other networks. Routers select the best path for a packet to take to its destination network.
  - De-encapsulation – The destination host de-encapsulates the packet to see if it matches its own.



Network Layer Protocols

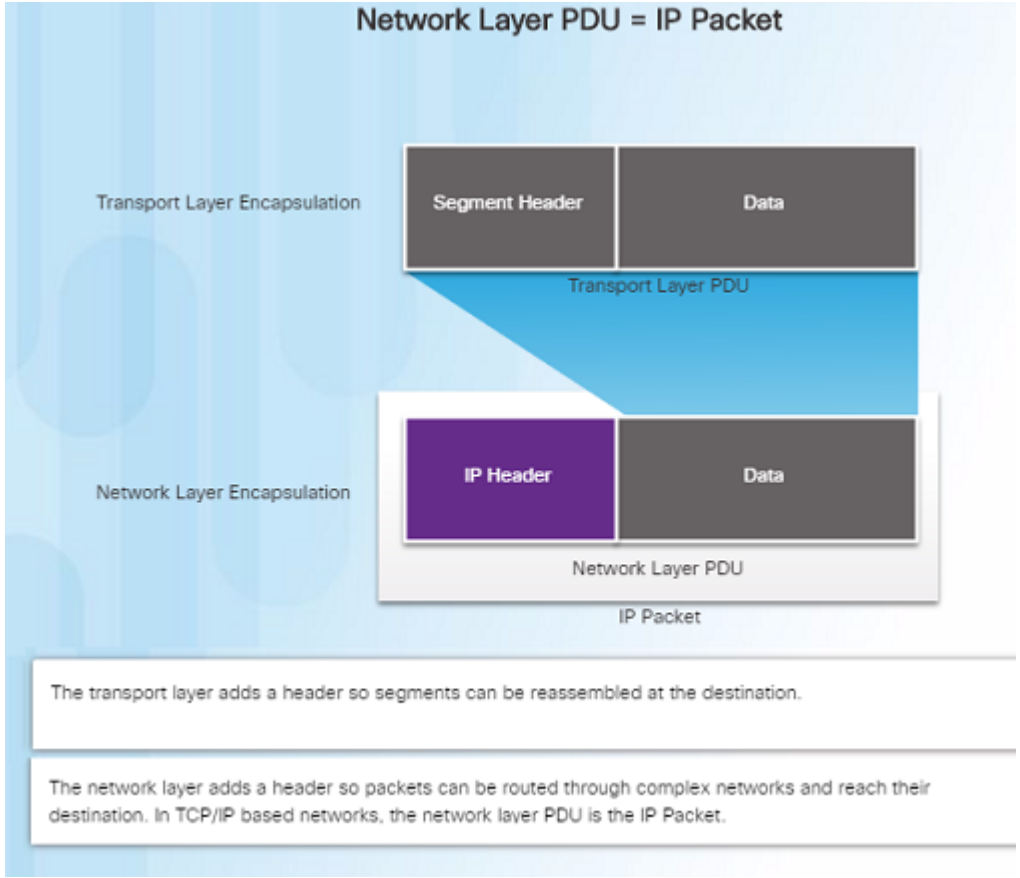
- There are several network layer protocols in existence; however, the most commonly implemented are:
  - Internet Protocol version 4 (IPv4)
  - Internet Protocol version 6 (IPv6)

Note: Legacy network layer protocols are not discussed in this course.



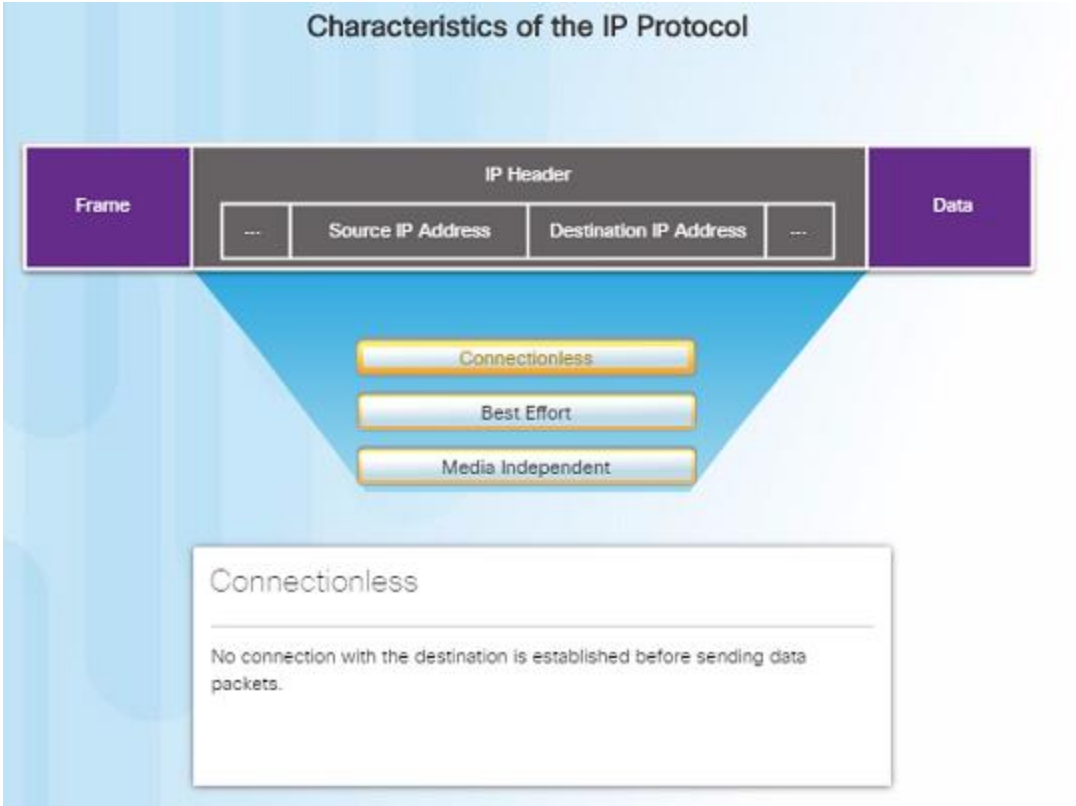
Encapsulating IP

- At the network layer, IP encapsulates the transport layer segment by adding an IP header for the purpose of delivery to the destination host.
- The IP header stays the same from the source to the destination host.
- The process of encapsulating data layer by layer enables the services at different layers to scale without affecting other layers.
- Routers implement different network layer protocols concurrently over a network and use the network layer packet header for routing.



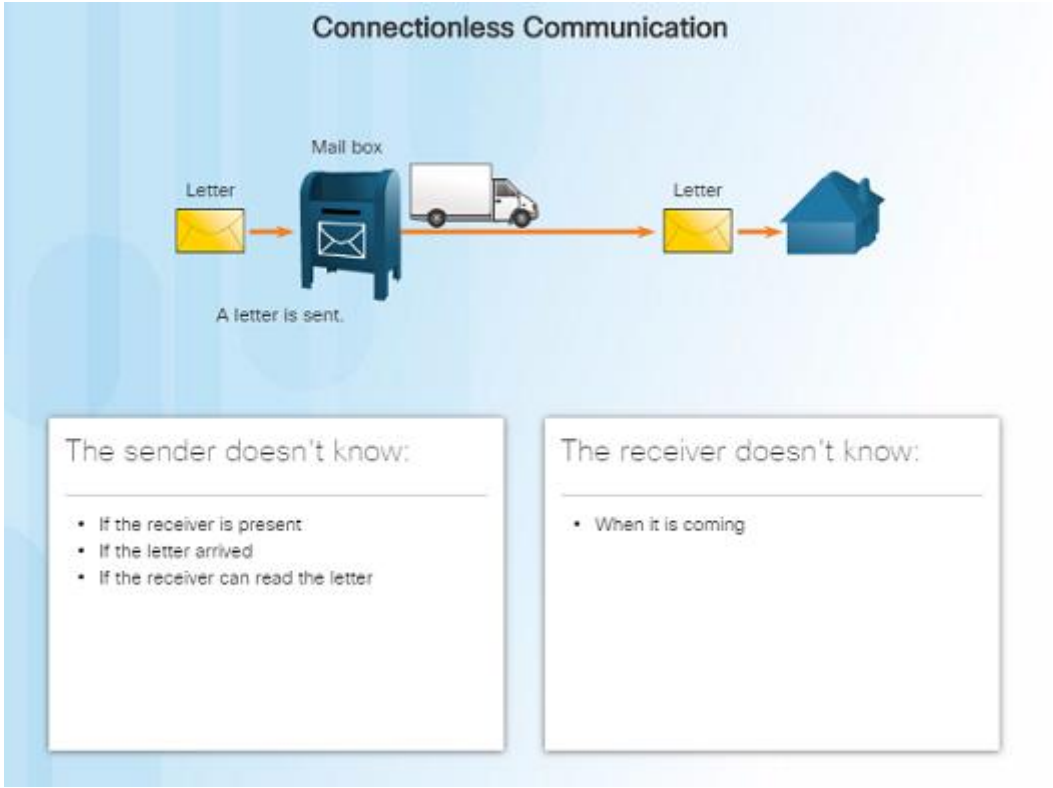
Characteristics of IP

- IP was designed as a protocol with low overhead – it provides only the functions required to deliver a packet from the source to a destination.
- An IP packet is sent to the destination without prior establishment of a connection
- IP was not designed to track and manage the flow of packets.
  - These functions, if required, are performed by other layers – primarily TCP



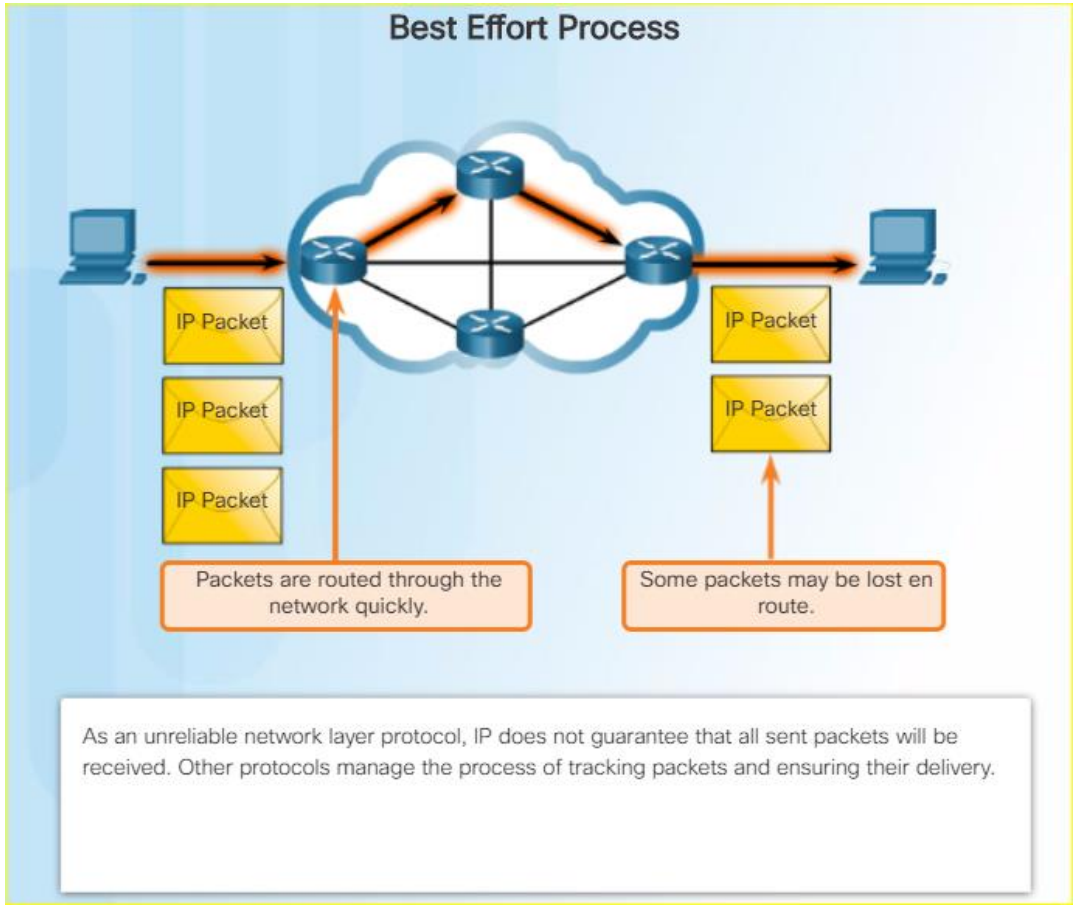
IP - Connectionless

- IP is a connectionless protocol:
  - No dedicated end-to-end connection is created before data is sent.
  - Very similar process as sending someone a letter through snail mail.
  - Senders do not know whether or not the destination is present, reachable, or functional before sending packets.
  - This feature contributes to the low overhead of IP.



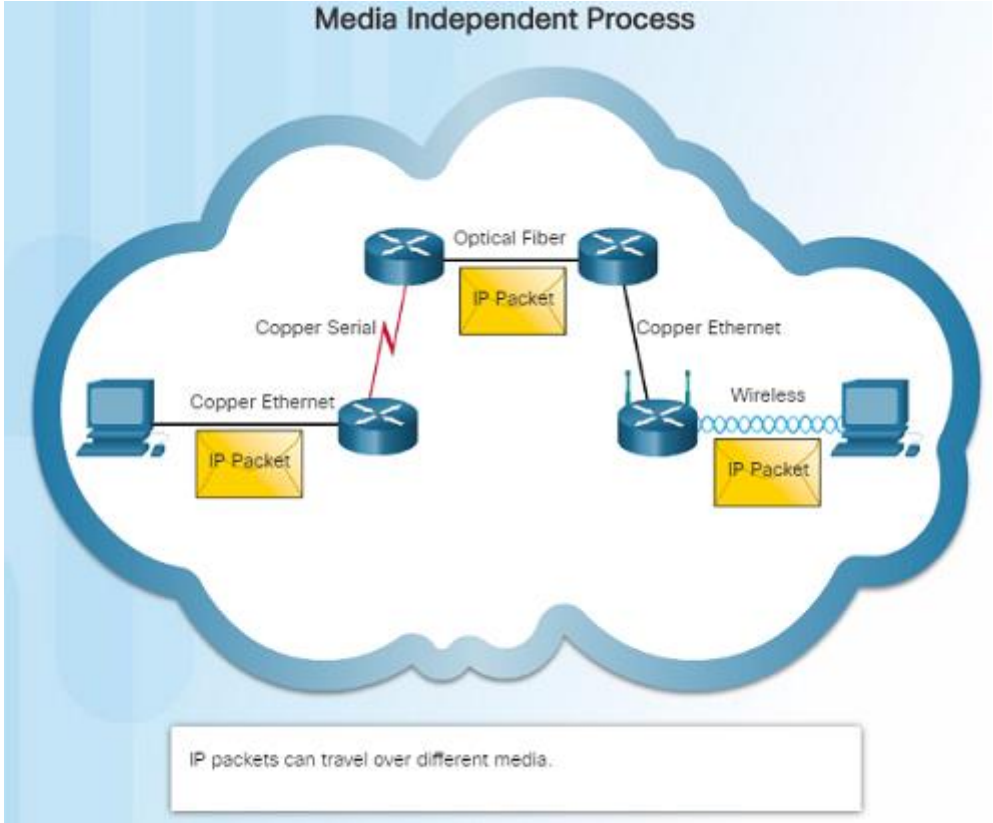
IP – Best Effort Delivery

- IP is a Best Effort Delivery protocol:
  - IP is considered “unreliable” because it does not guarantee that all packets that are sent will be received.
  - Unreliable means that IP does not have the capability to manage and recover from undelivered, corrupt, or out of sequence packets.
  - If packets are missing or not in the correct order at the destination, upper layer protocols/services must resolve these issues.



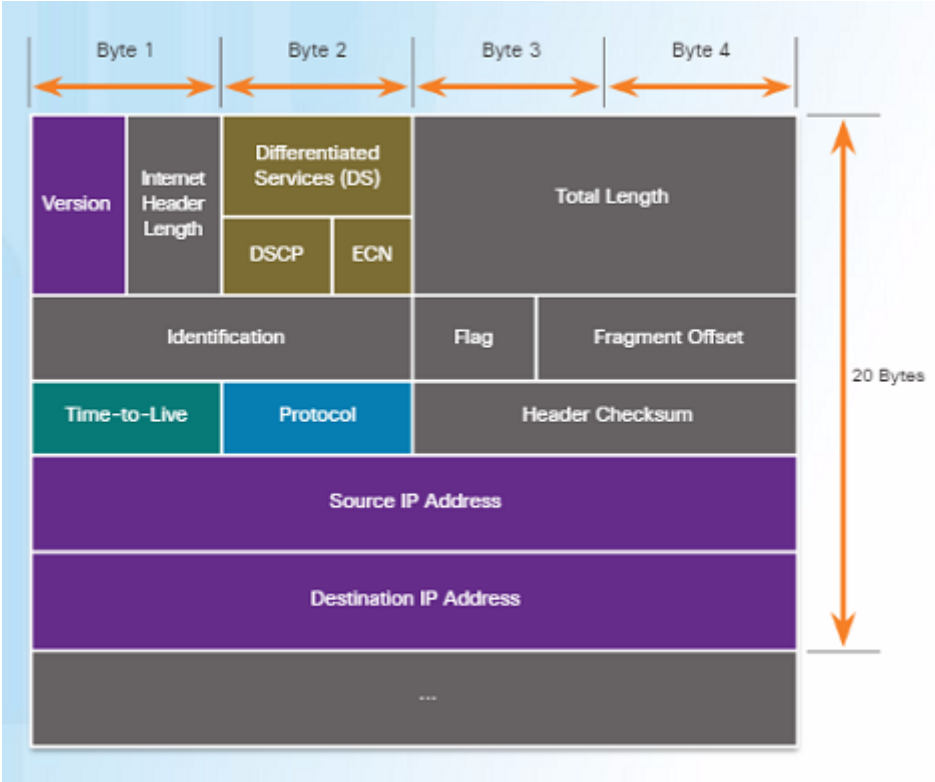
IP – Media Independent

- IP operates independently from the media that carries the data at lower layers of the protocol stack – it does not care if the media is copper cables, fiber optics or wireless.
- The OSI data link layer is responsible for taking the IP packet and preparing it for transmission over the communications medium.
- The network layer does have a maximum size of the PDU that can be transported – referred to as MTU (maximum transmission unit).
- The data link layer tells the network layer the MTU.



IPv4 Packet Header

- An IPv4 packet header consists of the fields containing binary numbers. These numbers identify various settings of the IP packet which are examined by the Layer 3 process.
- Significant fields include:
  - Version – Specifies that the packet is IP version 4
  - Differentiated Services or DiffServ (DS) – Used to determine the priority of each packet on the network.
  - Time-to-Live (TTL) – Limits the lifetime of a packet – decreased by one at each router along the way.
  - Protocol – Used to identify the next level protocol.
  - Source IPv4 Address – Source address of the packet.
  - Destination IPv4 Address – Address of destination.

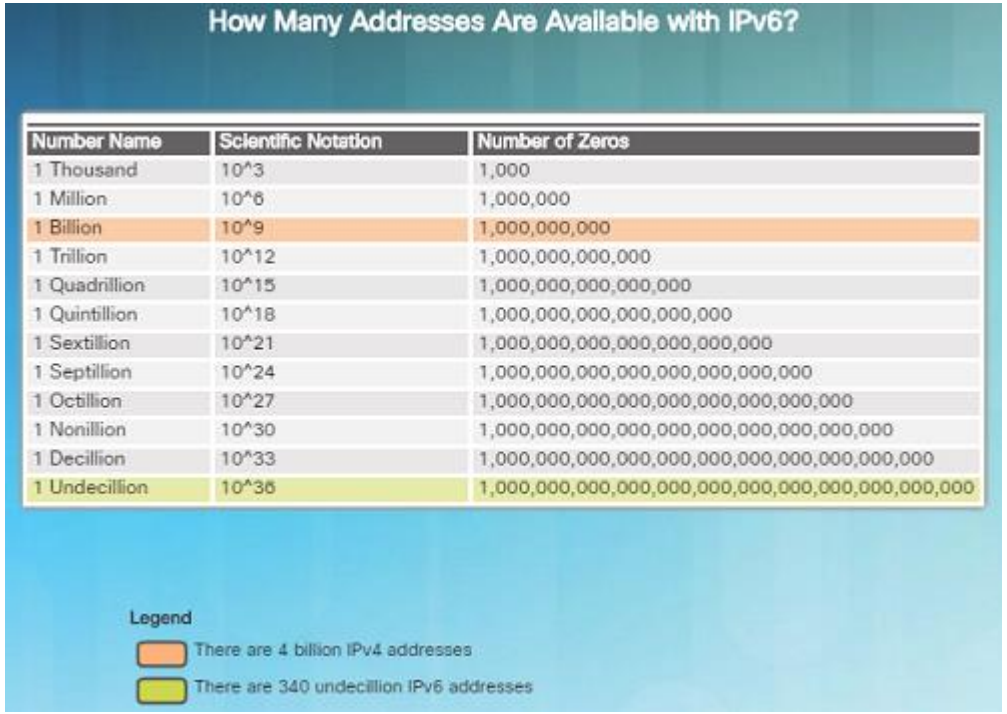


Limitations of IPv4

- IPv4 has been updated to address new challenges.
- Three major issues still exist with IPv4:
  - IP address depletion – IPv4 has a limited number of unique public IPv4 addresses available. Although there are about 4 billion IPv4 addresses, the exponential growth of new IP-enabled devices has increased the need.
  - Internet routing table expansion – A routing table contains the routes to different networks in order to make the best path determination. As more devices and servers are connected to the network, more routes are created. A large number of routes can slow down a router.
  - Lack of end-to-end connectivity – Network Address Translation (NAT) was created for devices to share a single IPv4 address. However, because they are shared, this can cause problems for technologies that require end-to-end connectivity.

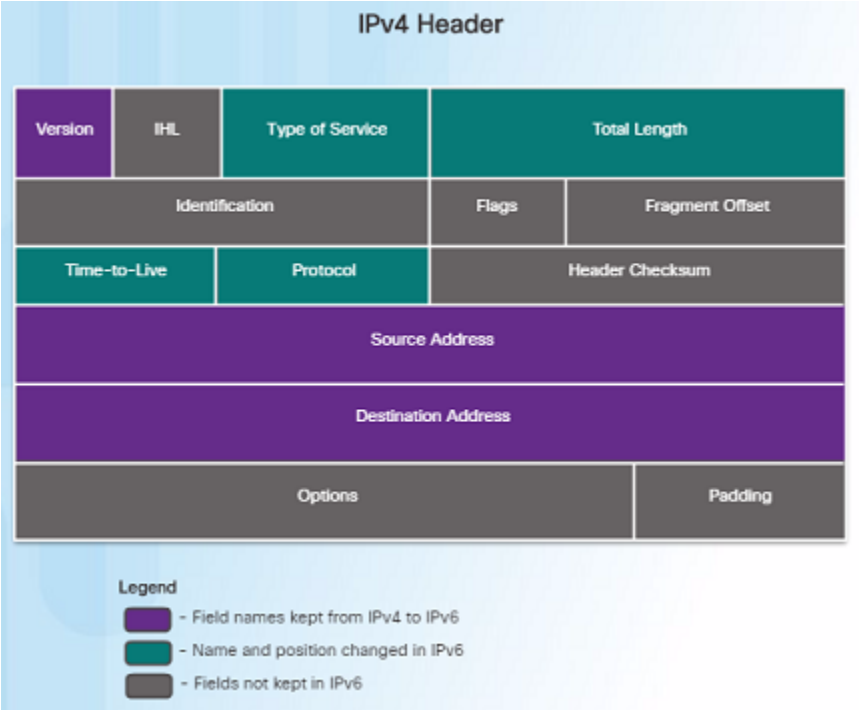
Introducing IPv6

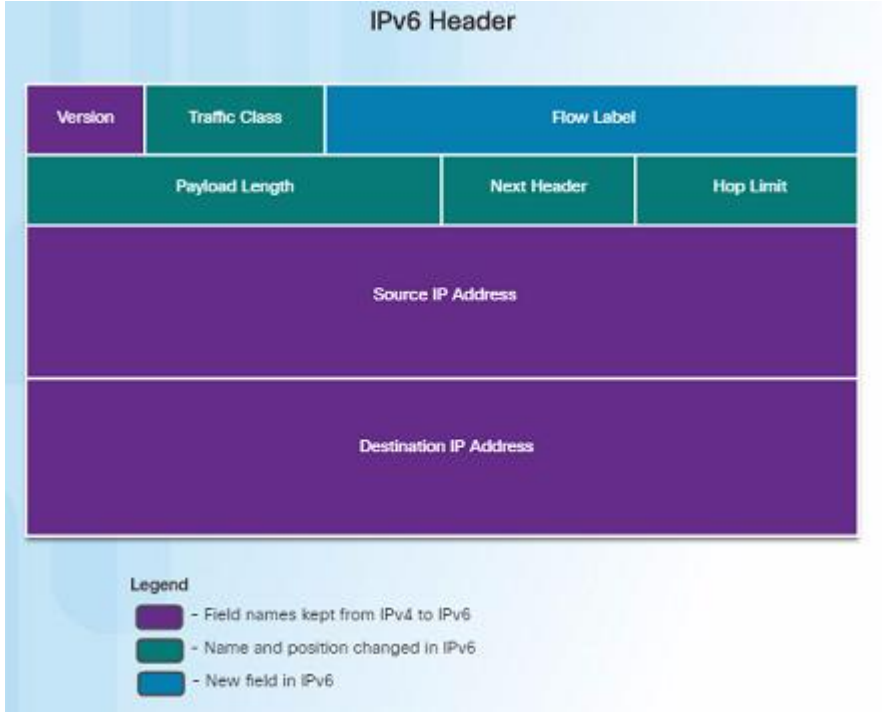
- In the early ‘90s, the IETF started looking at a replacement for IPv4 – which led to IPv6.
- Advantages of IPv6 over IPv4 include:
  - Increased address space – based on 128-bit addressing vs. 32-bit with IPv4
  - Improved packet handling – fewer fields with IPv6 than IPv4
  - Eliminates the need for NAT – no need to share addresses with IPv6
- There are roughly enough IPv6 addresses for every grain of sand on Earth.



Encapsulating IPv6

- The IPv6 header is simpler than the IPv4 header.

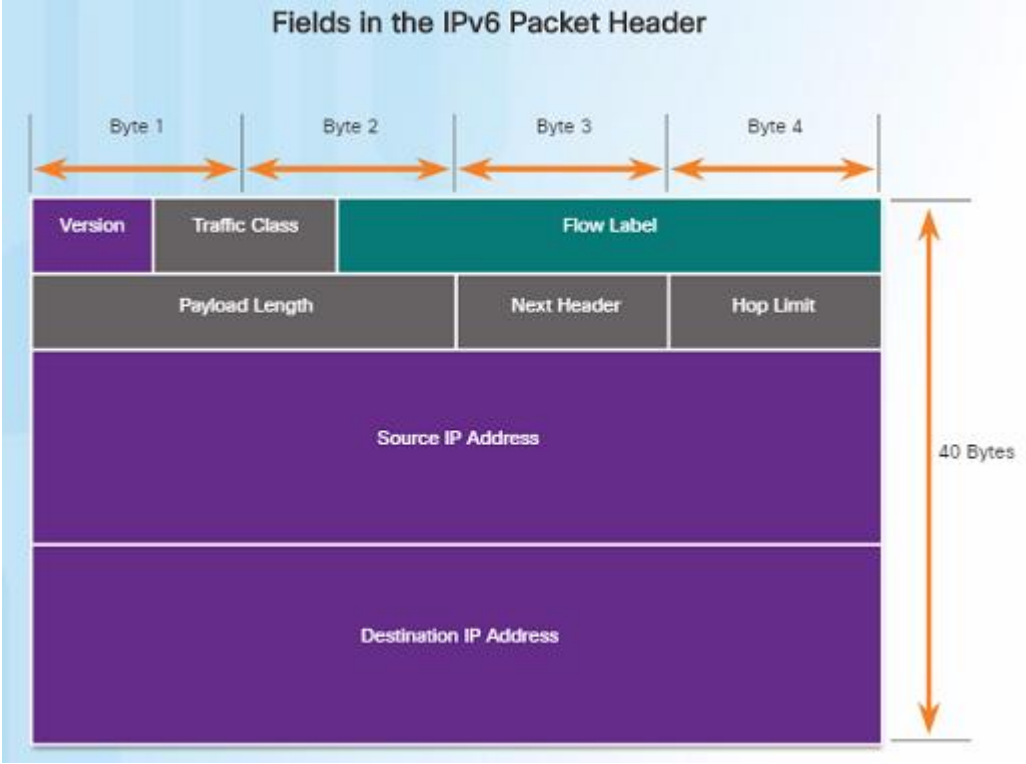




Encapsulating IPv6 (Cont.)

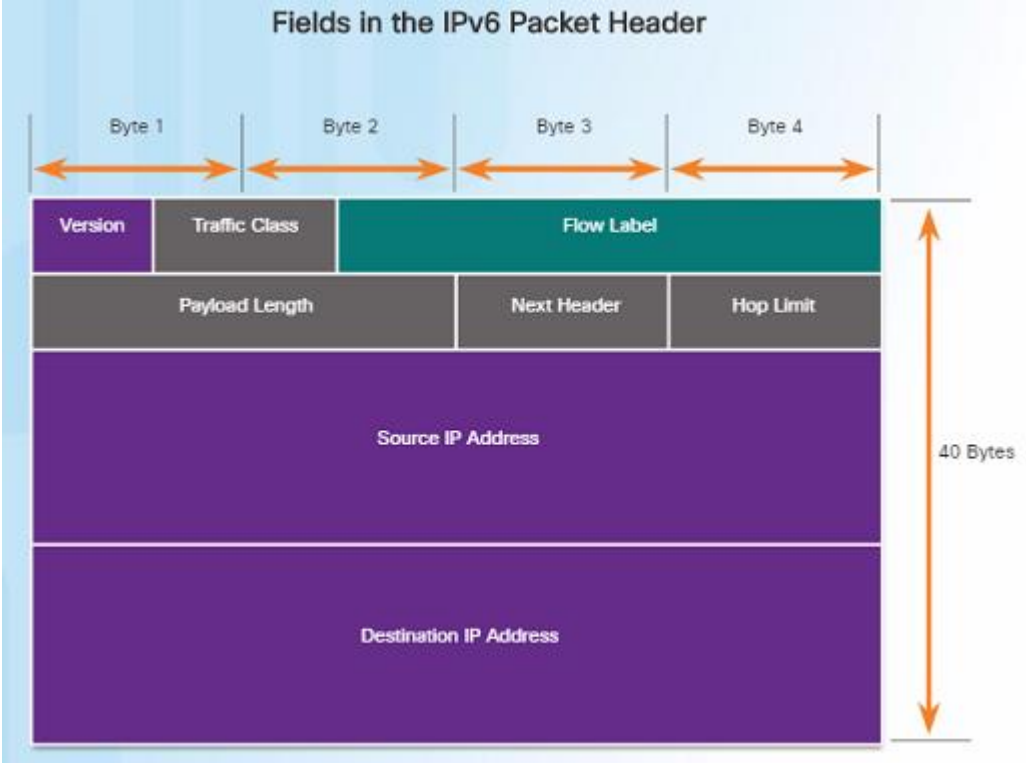
- Advantages of IPv6 over IPv4 using the simplified header:
  - Simplified header format for efficient packet handling
  - Hierarchical network architecture for routing efficiency
  - Autoconfiguration for addresses
  - Elimination of need for network address translation (NAT) between private and public addresses

IPv6 Packet Header



- IPv6 packet header fields:
  - Version – Contains a 4-bit binary value set to 0110 that identifies it as a IPv6 packet.
  - Traffic Class – 8-bit field equivalent to the IPv4 Differentiated Services (DS) field.
  - Flow Label – 20-bit field suggests that all packets with the same flow label receive the same type of handling by routers.
  - Payload Length – 16-bit field indicates the length of the data portion or payload of the packet.
  - Next Header – 8-bit field is equivalent to the IPv4 Protocol field. It indicates the data payload type that the packet is carrying.

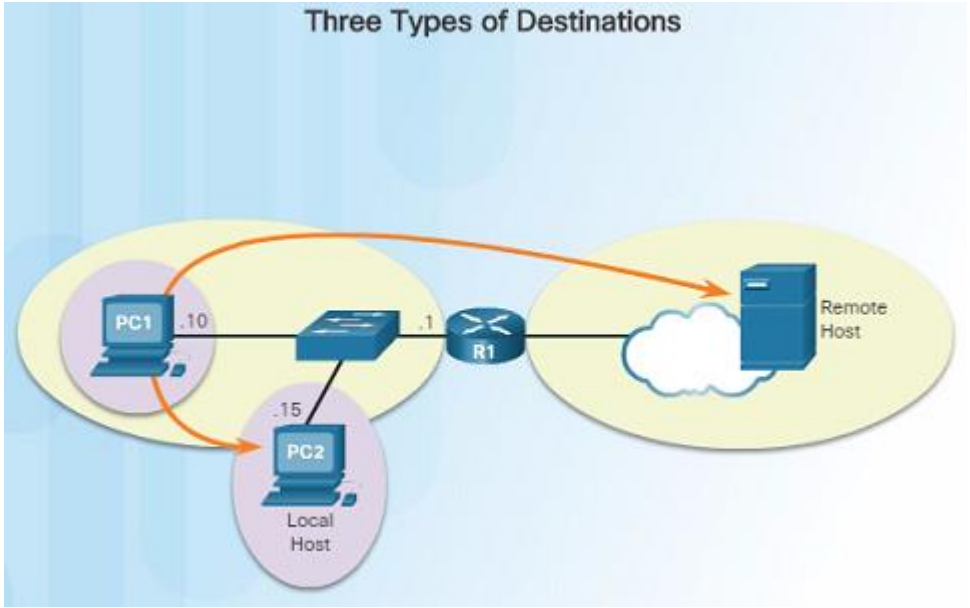
IPv6 Packet Header (Cont.)



- IPv6 packet header fields:
  - Hop Limit – 8-bit field replaces the IPv4 TTL field. This value is decremented by 1 as it passes through each router. When it reaches zero, the packet is discarded.
  - Source IPv6 Address – 128-bit field that identifies the IPv6 address of the sending host.
  - Destination IPv6 Address – 128-bit field that identifies the IPv6 address of the receiving host.

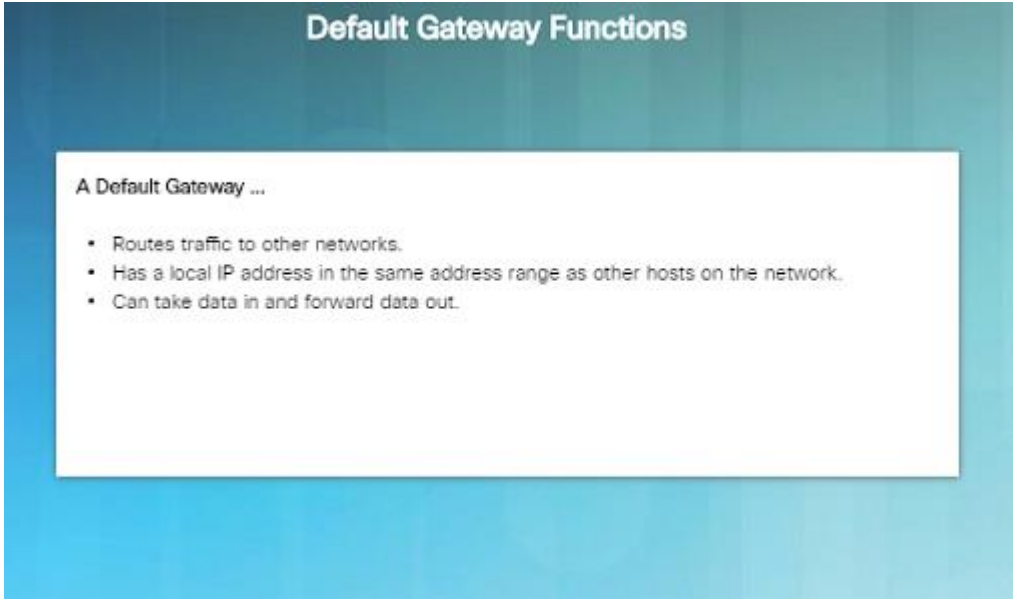
Routing

Host Forwarding Decision



- An important role of the network layer is to direct packets between hosts. A host can send a packet to:
  - Itself – A host can ping itself for testing purposes using 127.0.0.1 which is referred to as the loopback interface.
  - Local host – This is a host on the same local network as the sending host. The hosts share the same network address.
  - Remote host – This is a host on a remote network. The hosts do not share the same network address.
- The source IPv4 address and subnet mask is compared with the destination address and subnet mask in order to determine if the host is on the local network or remote network.

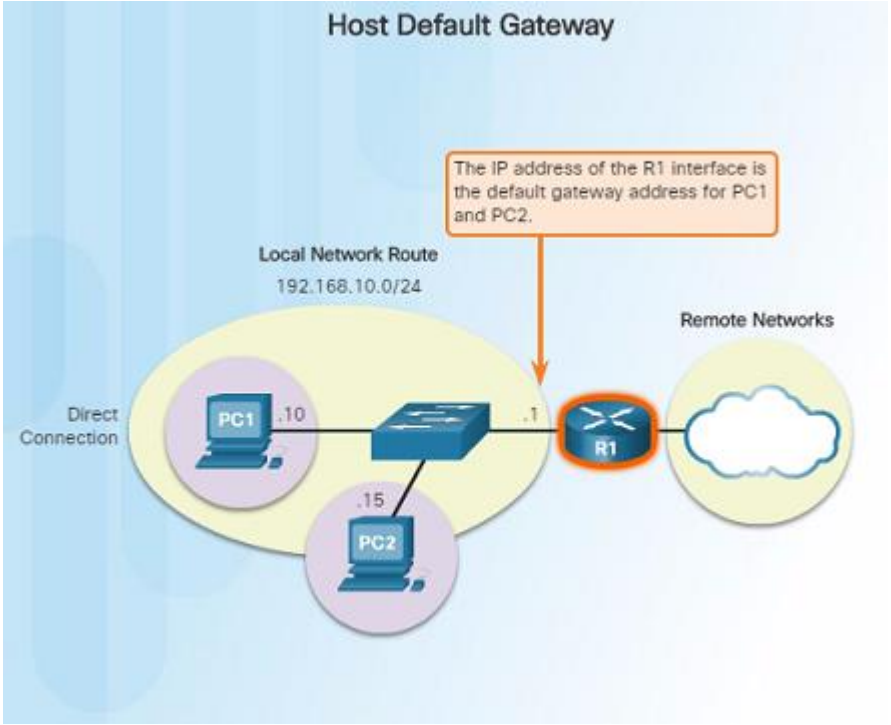
Default Gateway



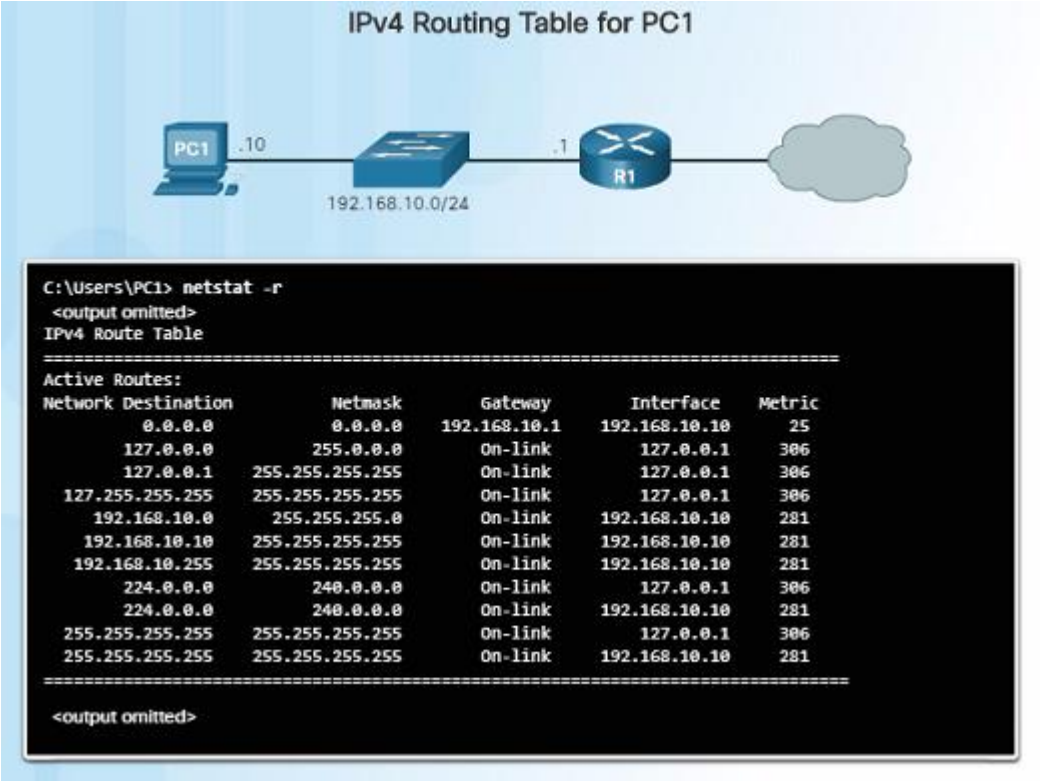
- The default gateway is the network device that can route traffic out to other networks. It is the router that routes traffic out of a local network.
- This occurs when the destination host is not on the same local network as the sending host.
- The default gateway will know where to send the packet using its routing table.
- The sending host does not need to know where to send the packet other than to the default gateway – or router.

Using the Default Gateway

- A host's routing table usually includes a default gateway address – which is the router IP address for the network that the host is on.
- The host receives the IPv4 address for the default gateway from DHCP, or it is manually configured.
- Having a default gateway configured creates a default route in the routing table of a host - which is the route the computer will send a packet to when it needs to contact a remote network.

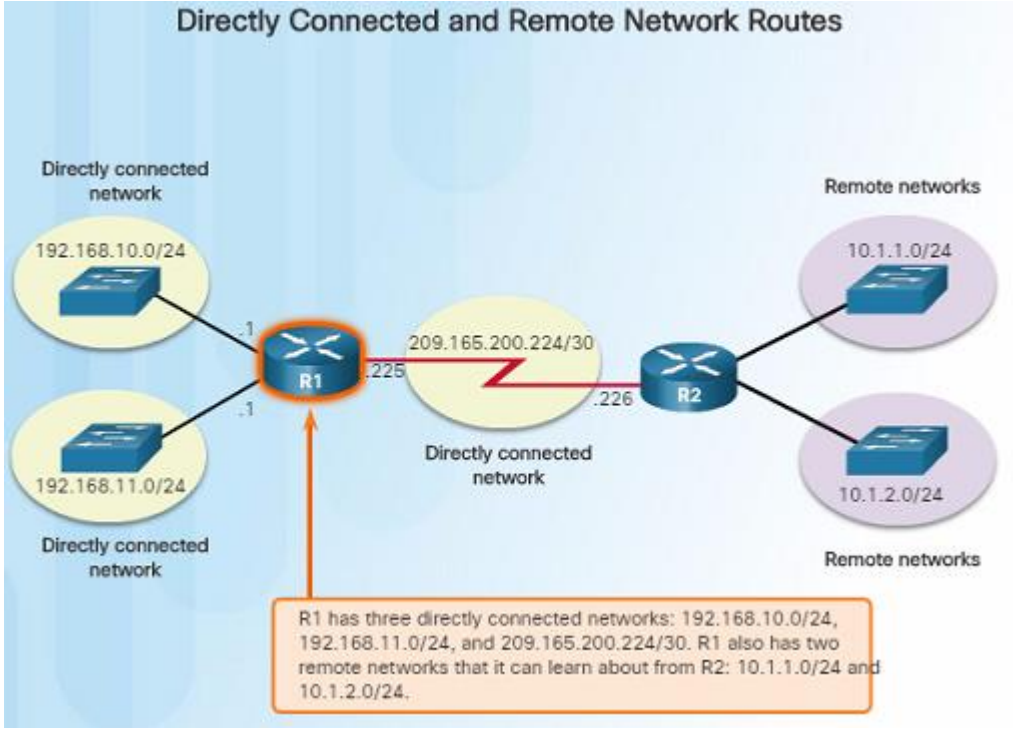


Host Routing Tables



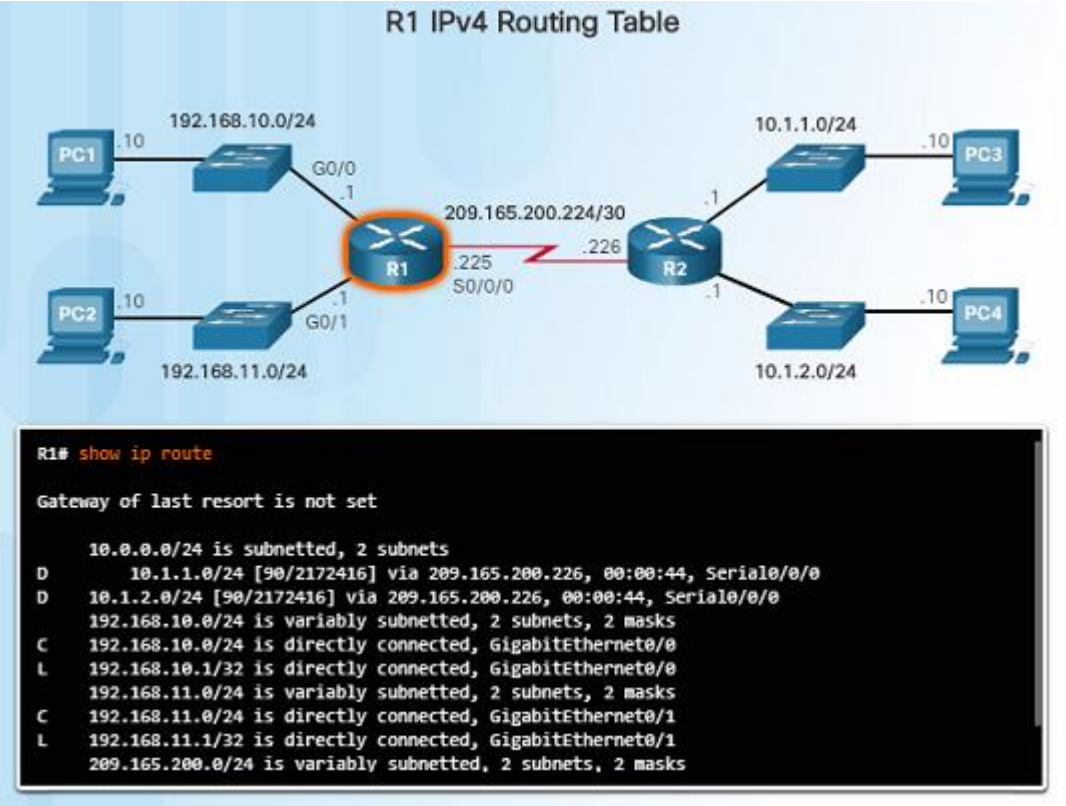
- On a Windows host, you can display the routing table using:
  - **route print**
  - **netstat -r**
- Three sections will be displayed:
  - Interface List – Lists the Media Access Control (MAC) address and assigned interface number of network interfaces on the host.
  - IPv4 Route Table – Lists all known IPv4 routes.
  - IPv6 Route Table – Lists all known IPv6 routes.

Router Packet Forwarding Decision



- When a router receives a packet destined for a remote network, the router has to look at its routing table to determine where to forward the packet. A router's routing table contains:
  - Directly-connected routes – These routes come from the active router interfaces configured with IP addresses.
  - Remote routes – These routes come from remote networks connected to other routers. They are either configured manually or learned through a dynamic routing protocol.
  - Default route – This is where the packet is sent when a route does not exist in the routing table.

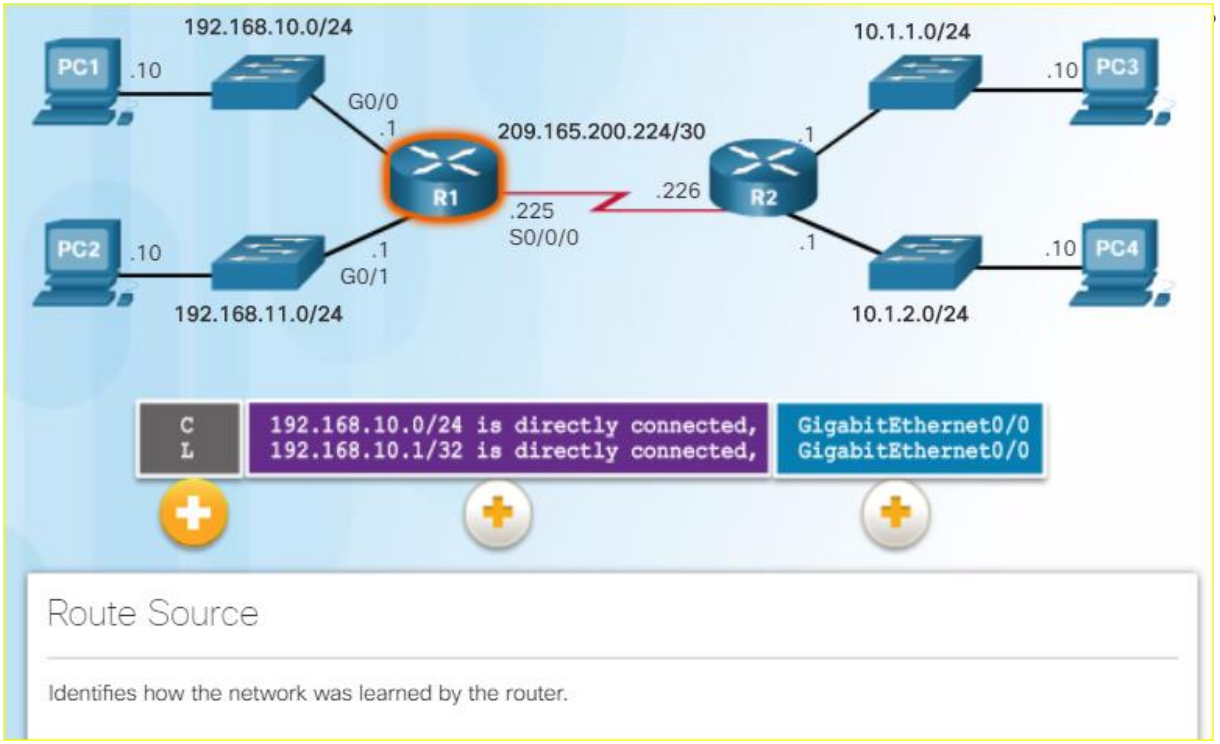
IPv4 Router Routing Table



- On a Cisco IOS router, the **show ip route** command is used to display the router's IPv4 routing table. The routing table shows:
  - Directly connected and remote routes
  - How each route was learned
  - Trustworthiness and rating of the route
  - When the route was last updated
  - Which interface is used to reach the destination

- A router examines an incoming packet's header to determine the destination network. If there's a match, the packet is forwarded using the specified information in the routing table.

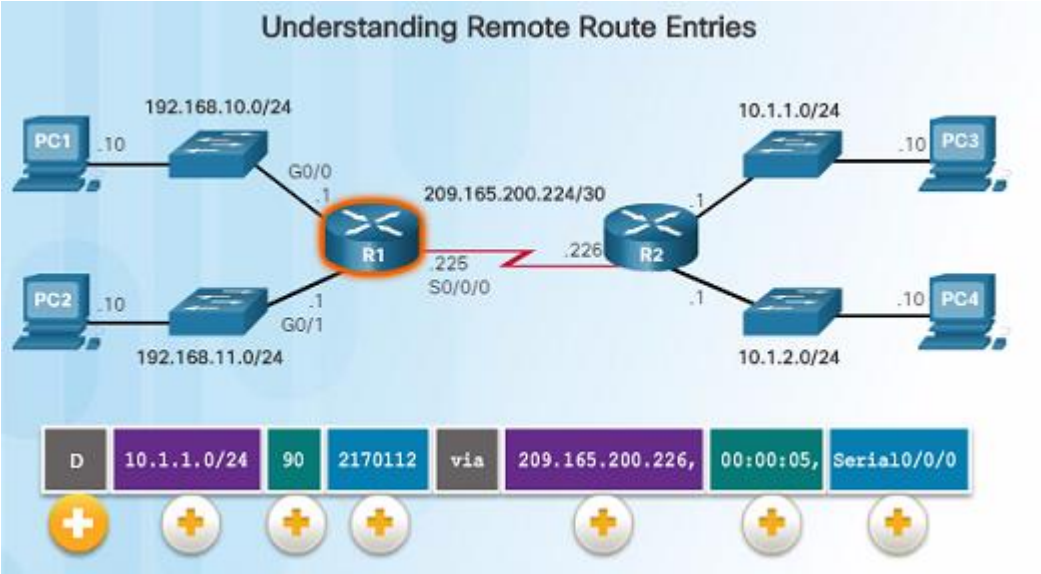
Directly Connected Routing Table Entries



When a router interface is configured and activated, the following two routing table entries are created automatically:

- **C** – Identifies that the network is directly connected and the interface is configured with an IP address and activated.
- **L** – Identifies that it is a local interface. This is the IPv4 address of the interface on the router.

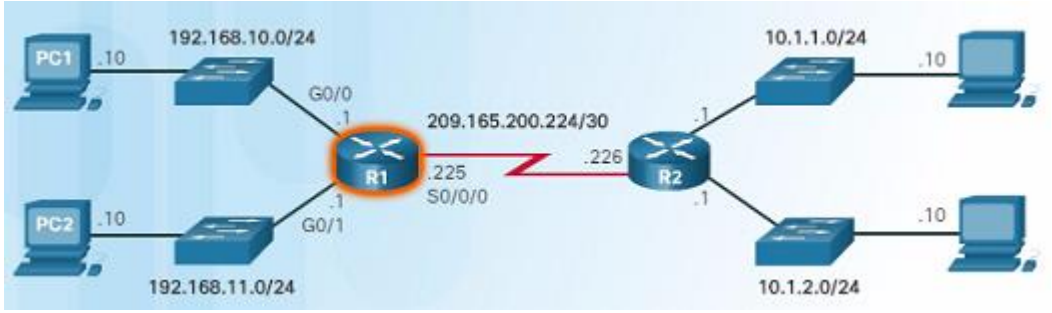
Understanding Remote Route Entries



- The **D** represents the Route Source which is how the network was learned by the router. **D** identifies the route as an EIGRP route or (Enhanced Interior Gateway Routing Protocol)

- **10.1.1.0/24** identifies the destination network.
- **90** is the administrative distance for the corresponding network – or the trustworthiness of the route. The lower the number, the more trustworthy it is.
- **2170112** – represents the metric or value assigned to reach the remote network. Lower values indicate preferred routes.
- **209.165.200.226** – Next-hop or IP address of the next router to forward the packet.
- **00:00:05** - Route Timestamp identifies when the router was last heard from.
- **Serial/0/0/0** – Outgoing Interface

Next-Hop Address




```
R1# show ip route
<output omitted>
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
D     10.1.1.0/24 [90/2170112] via 209.165.200.226, 00:00:05,
      Serial0/0/0
D     10.1.2.0/24 [90/2170112] via 209.165.200.226, 00:00:05,
      Serial0/0/0
      192.168.10.0/24 is variably subnetted, 2 subnets, 3 masks
C     192.168.10.0/24 is directly connected, GigabitEthernet0/0
L     192.168.10.1/32 is directly connected, GigabitEthernet0/0
      192.168.11.0/24 is variably subnetted, 2 subnets, 3 masks
C     192.168.11.0/24 is directly connected, GigabitEthernet0/1
L     192.168.11.1/32 is directly connected, GigabitEthernet0/1
      209.165.200.0/24 is variably subnetted, 2 subnets, 3 masks
C     209.165.200.224/30 is directly connected, Serial0/0/0
L     209.165.200.225/32 is directly connected, Serial0/0/0
R1#
```

- When a packet arrives at a router destined for a remote network, it will send the packet to the next hop address corresponding to the destination network address in its routing table.
- For example, if the R1 router in the figure to the left receives a packet destined for a device on the 10.1.1.0/24 network, it will send it to the next hop address of 209.165.200.226.
- Notice in the routing table, a default gateway address is not set – if the router receives a packet for a network that isn't in the routing table, it will be dropped.