Computer networks

Instructor:
Assistant Professor, PhD Akzhibek Amirova
akzhibek.amirova@astanait.edu.kz

Lecture 7

Transport Layer

Objectives

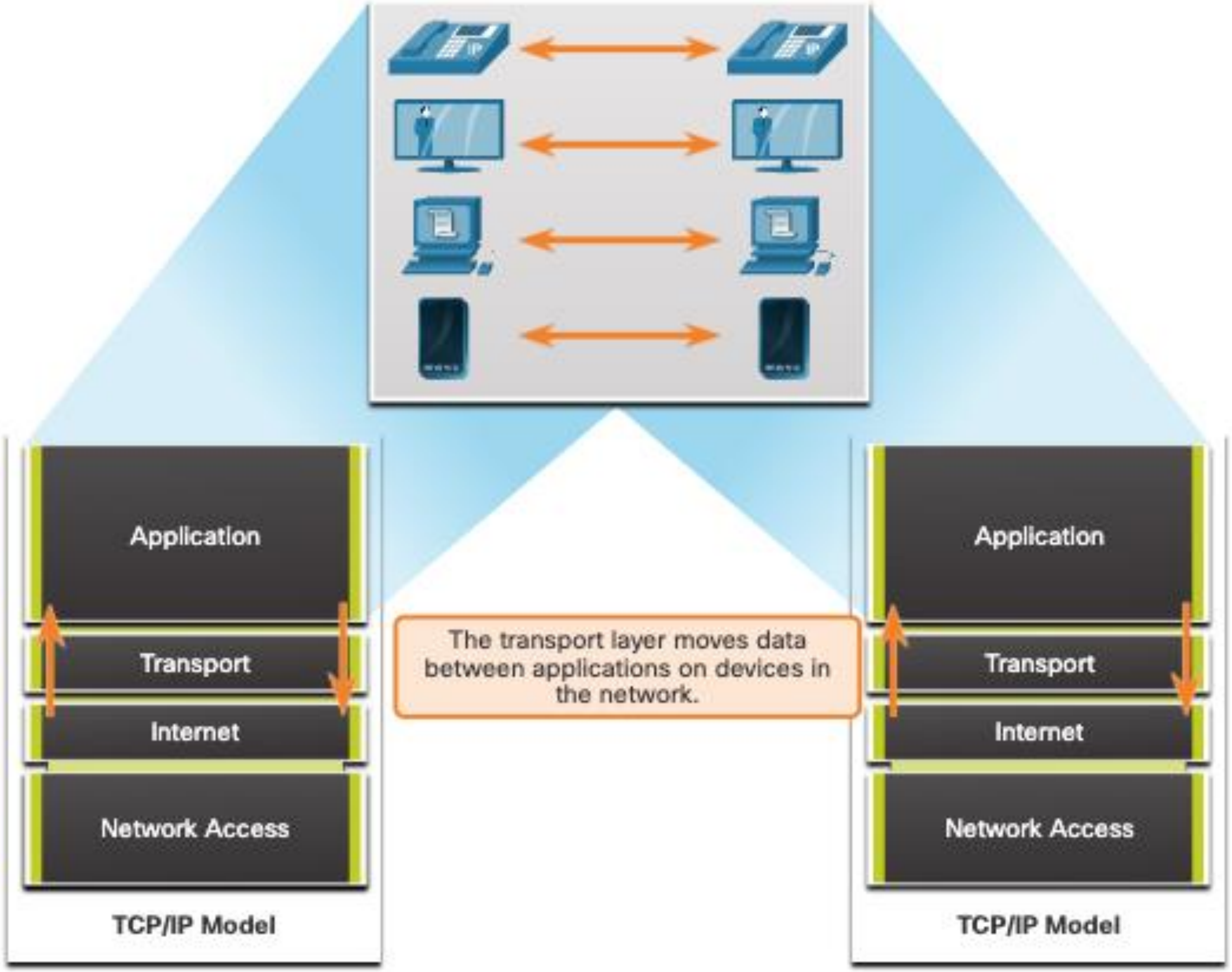
Transportation of Data	TCP Overview	UDP Overview

Transportation of Data

Role of the Transport Layer

The transport layer is:

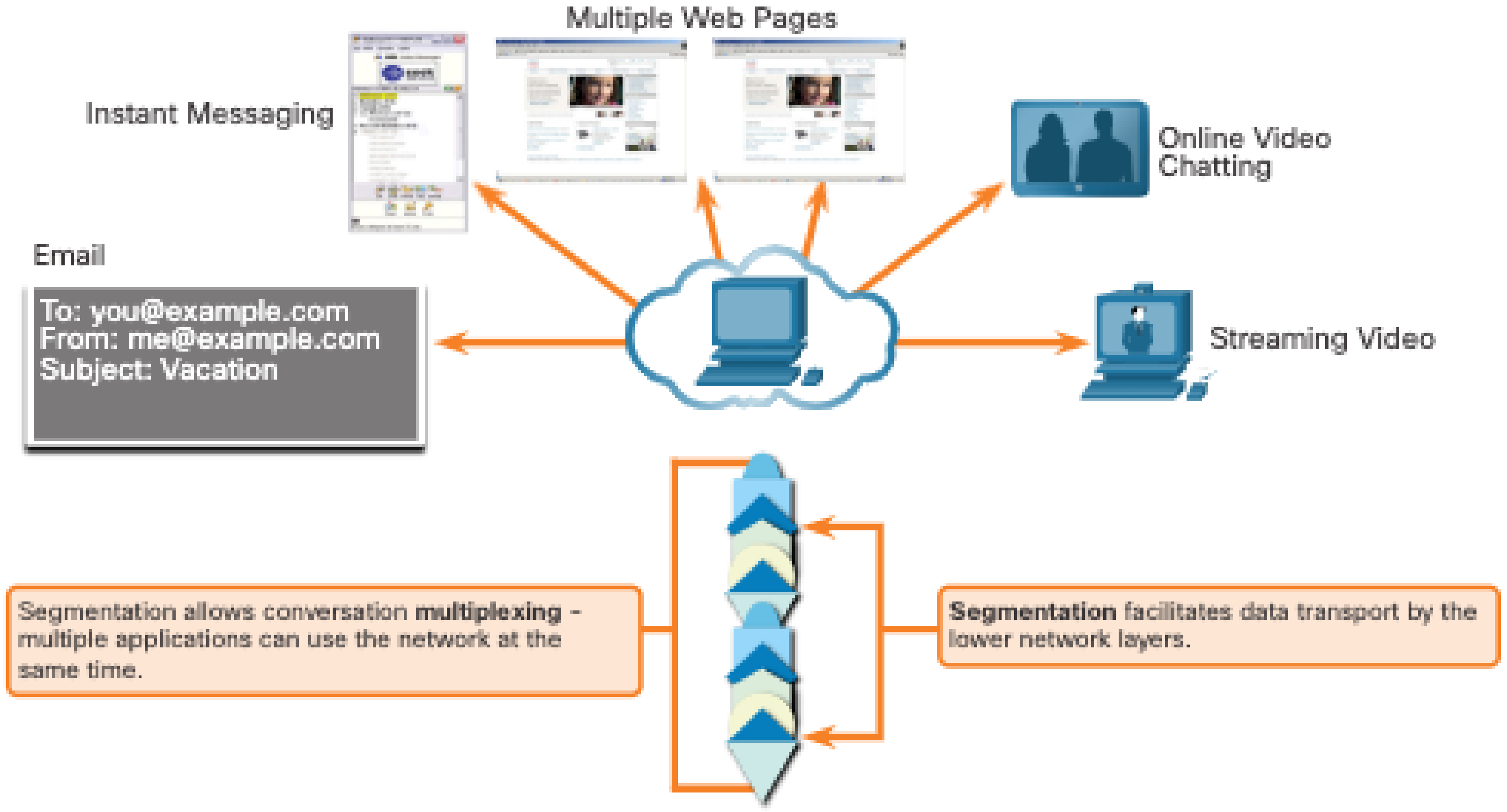
- responsible for logical communications between applications running on different hosts.
- The link between the application layer and the lower layers that are responsible for network transmission.



Transport Layer Responsibilities

The transport layer has the following responsibilities:

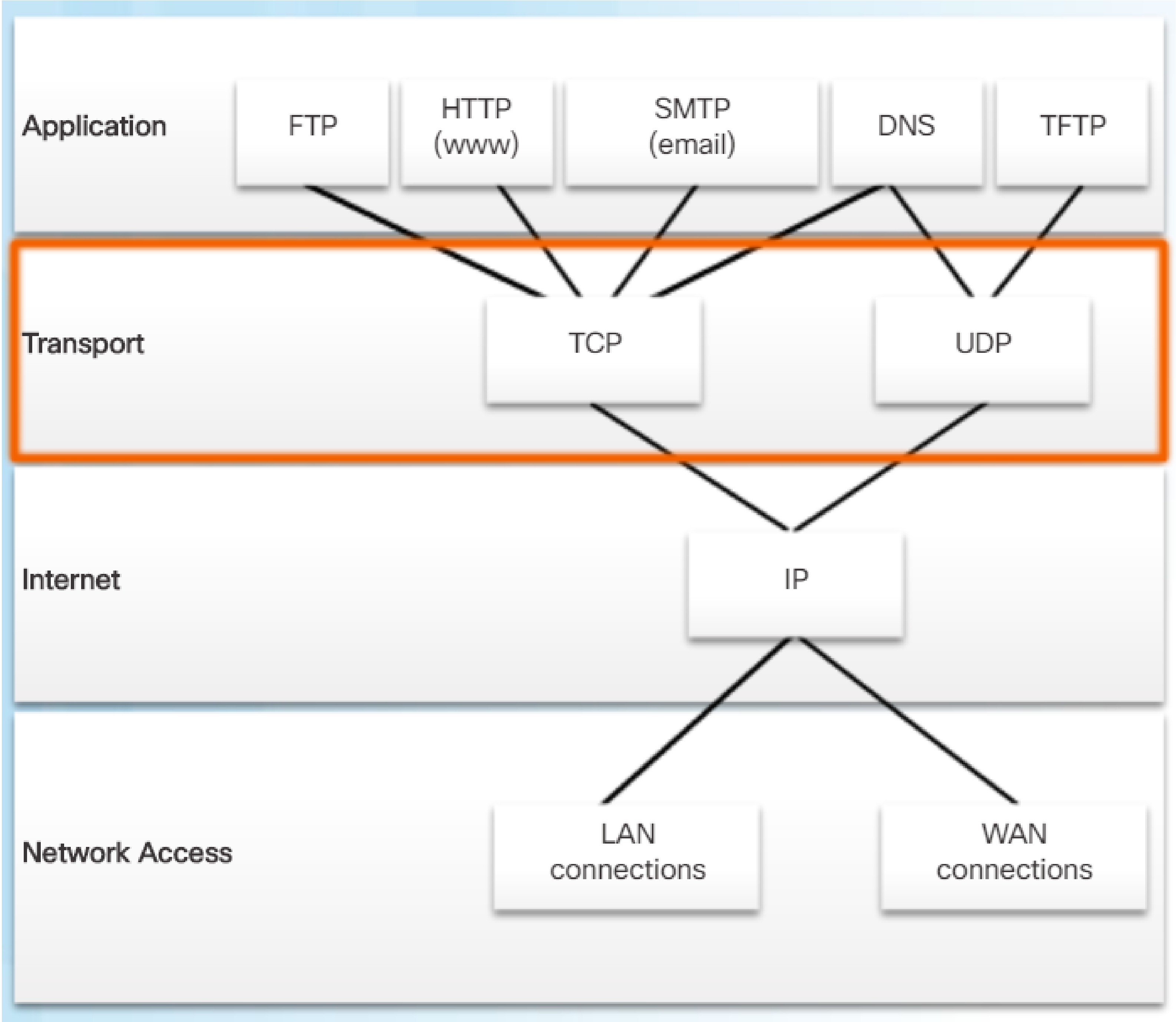
- Tracking individual conversations
- Segmenting data and reassembling segments
- Adds header information
- Identify, separate, and manage multiple conversations
- Uses segmentation and multiplexing to enable different communication conversations to be interleaved on the same network



Transportation of Data

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.

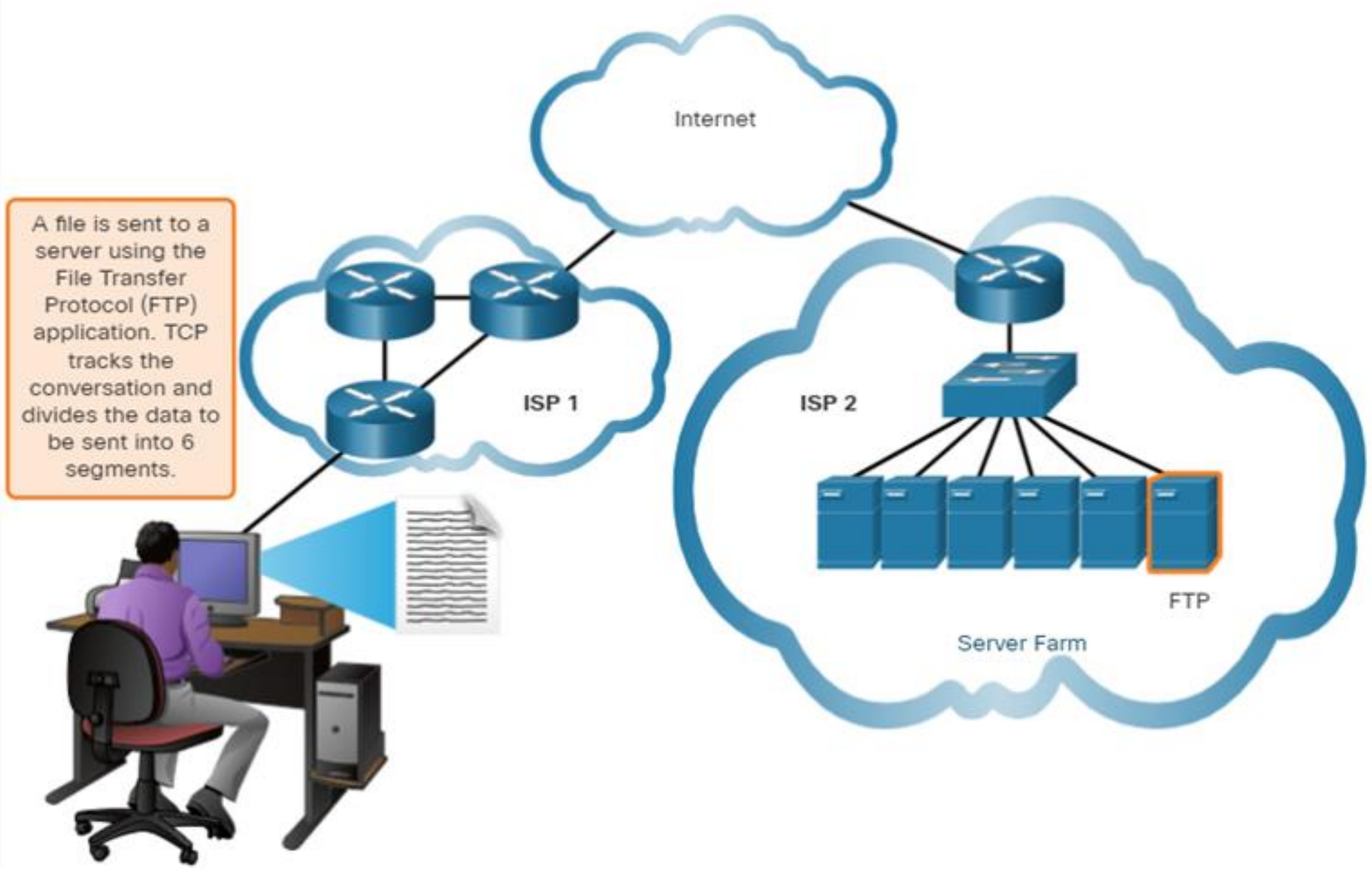


Transmission Control Protocol

TCP provides reliability and flow control.

TCP basic operations:

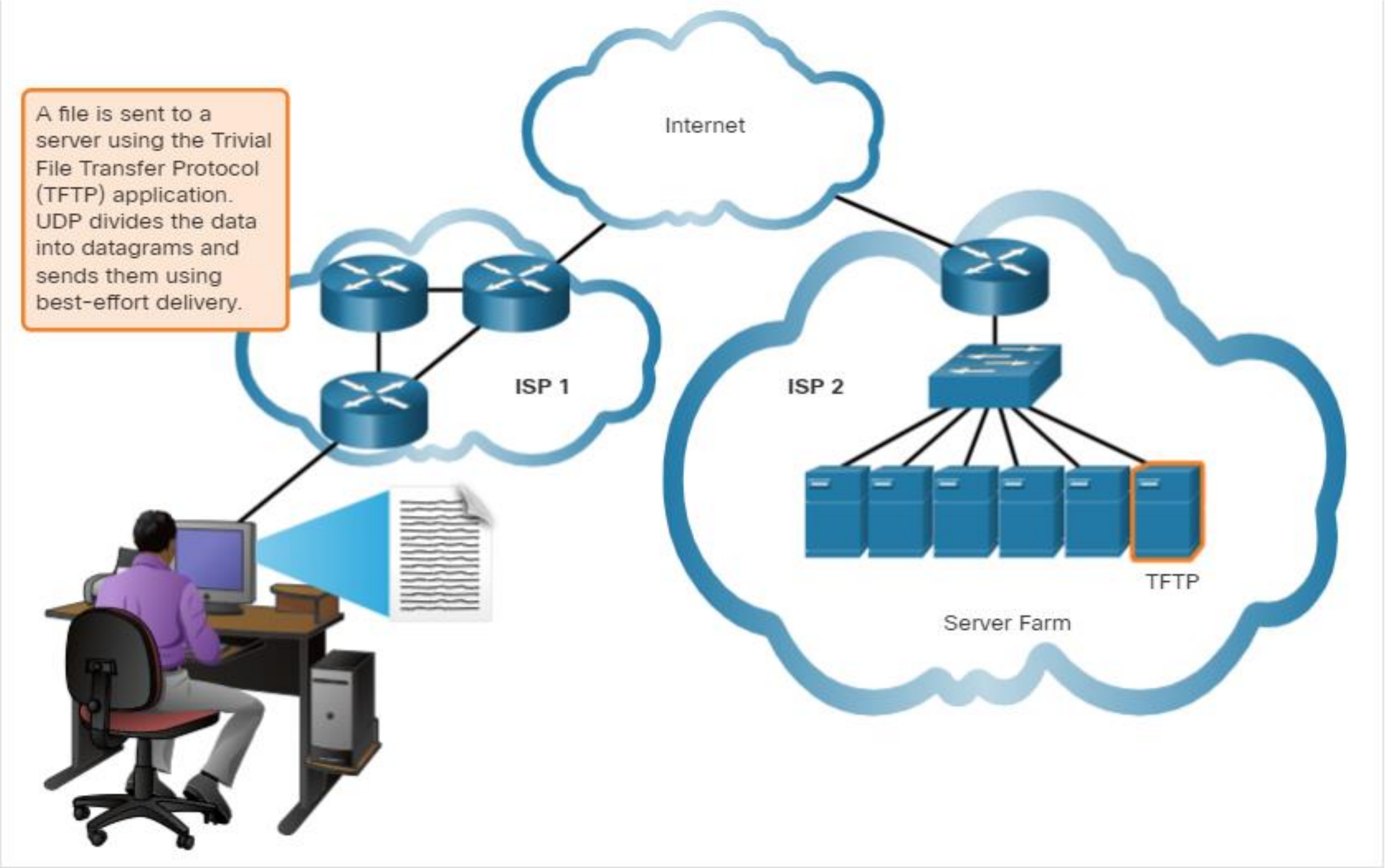
- Number and track data segments transmitted to a specific host from a specific application
- Acknowledge received data
- Retransmit any unacknowledged data after a certain amount of time
- Sequence data that might arrive in wrong order
- Send data at an efficient rate that is acceptable by the receiver



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.

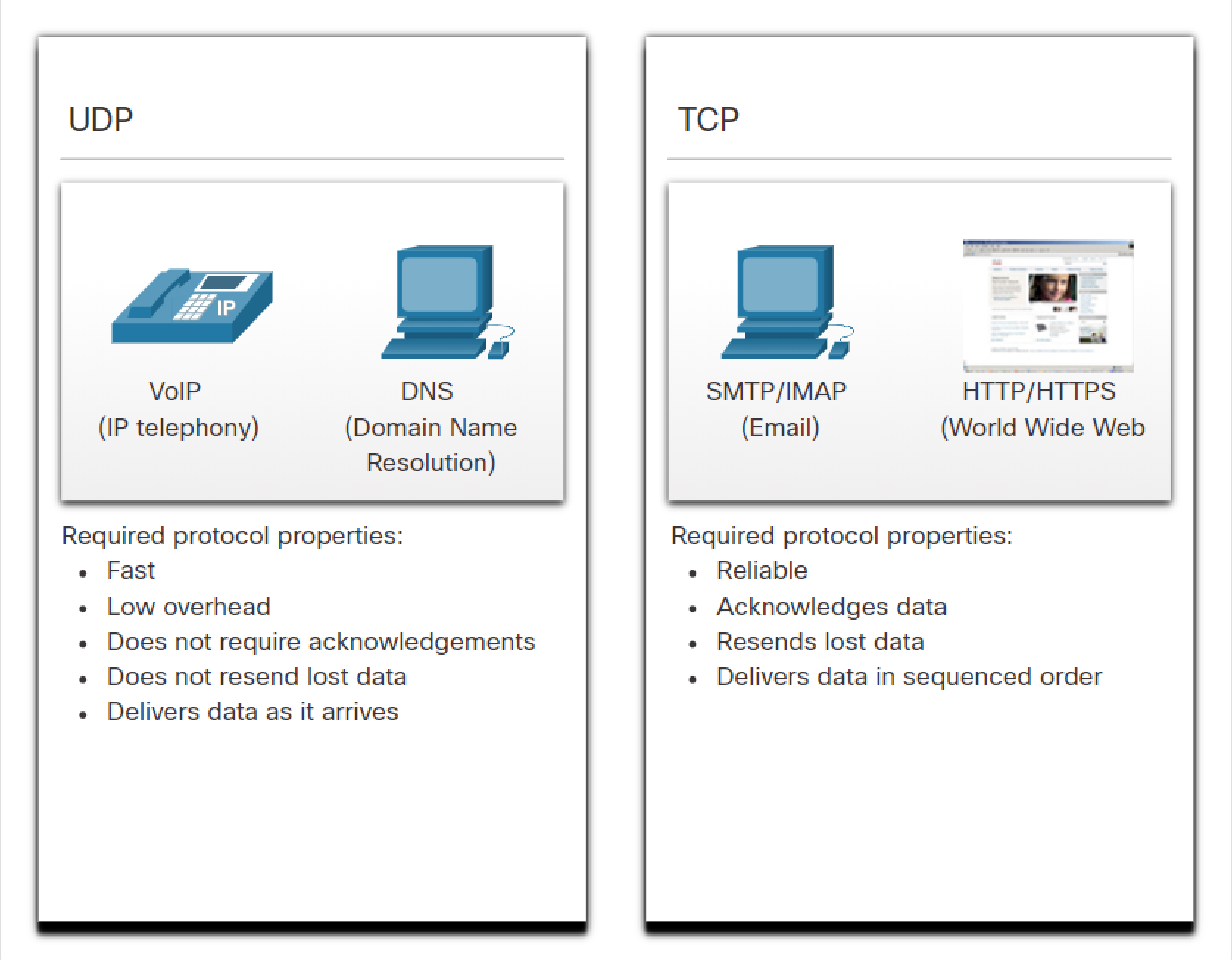


Transportation of Data

The 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.



TCP Overview

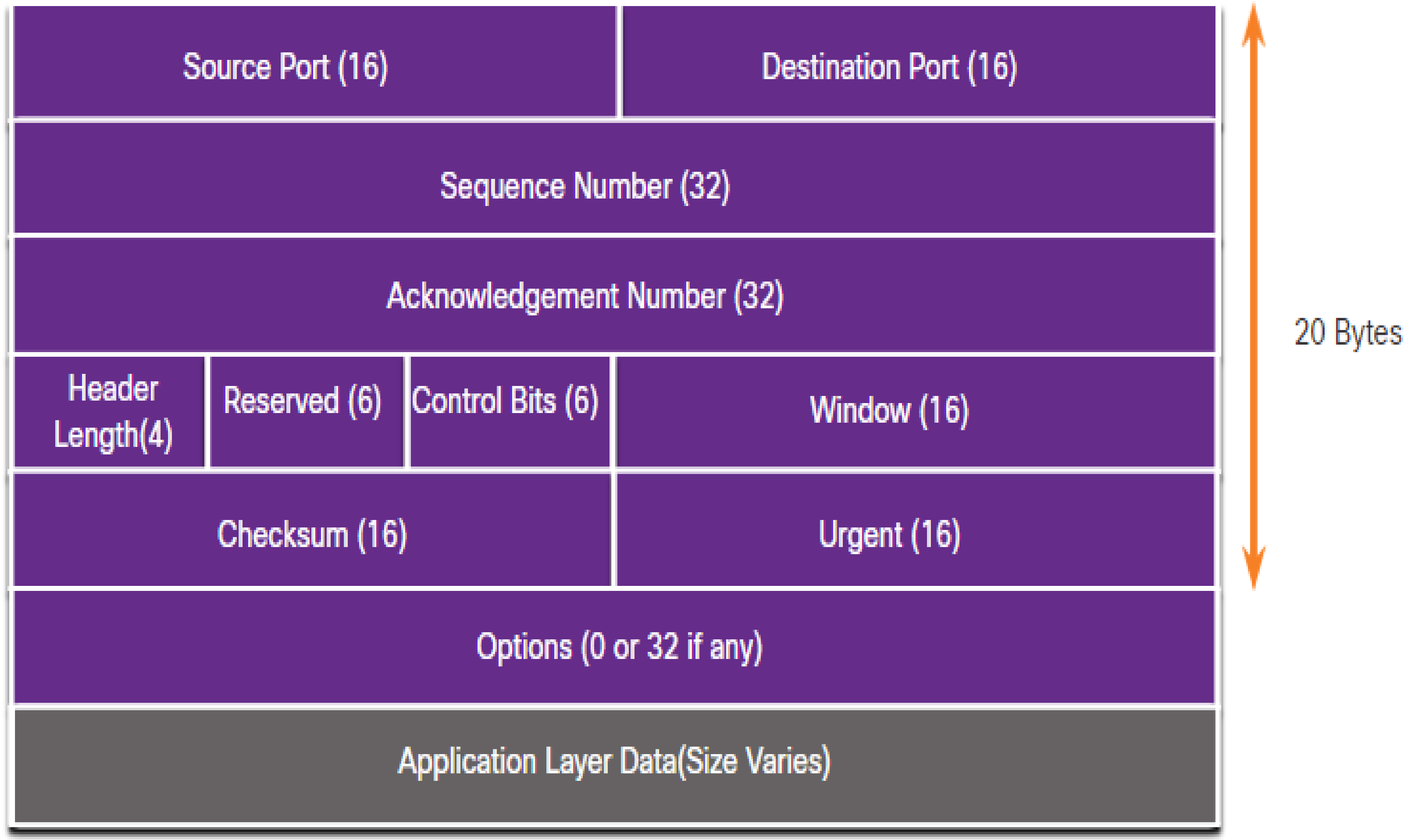
TCP Features

- **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.

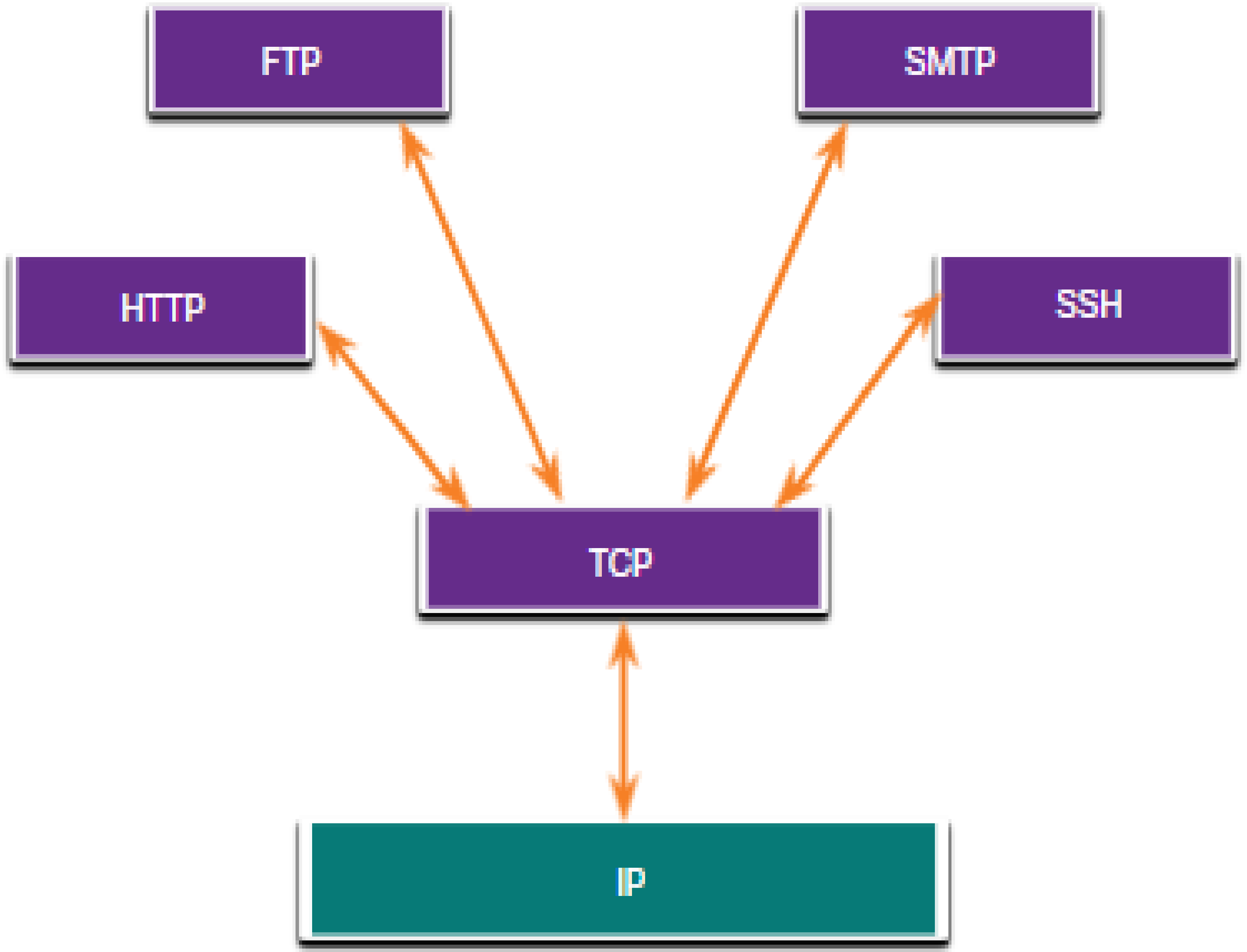


TCP Header Fields

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.

Applications that use TCP

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



UDP Overview

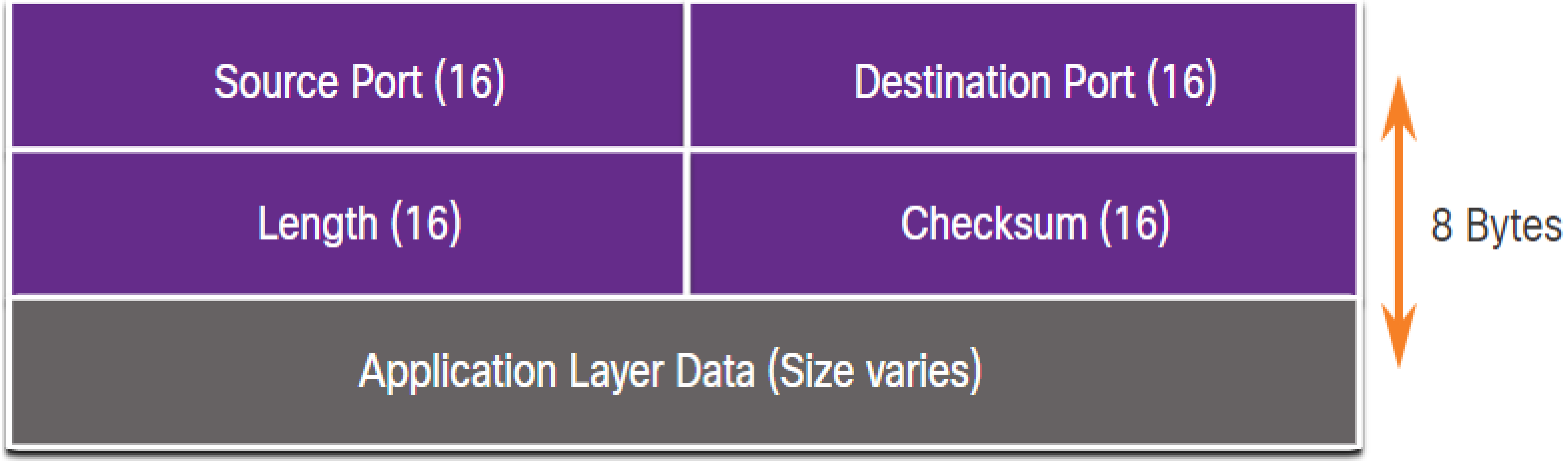
UDP Features

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

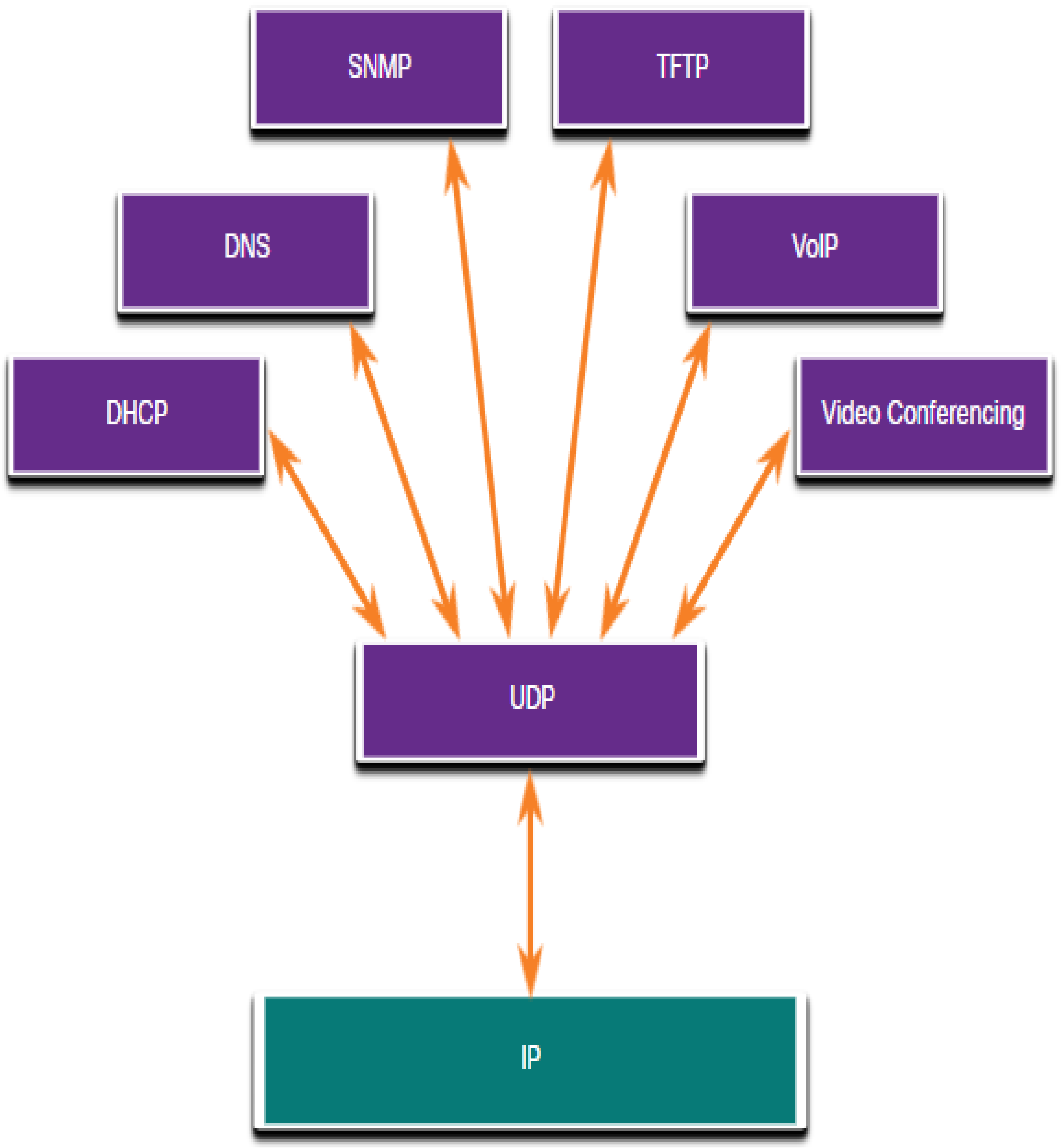
The table identifies and describes the four fields in a UDP header.

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.

UDP Overview

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.



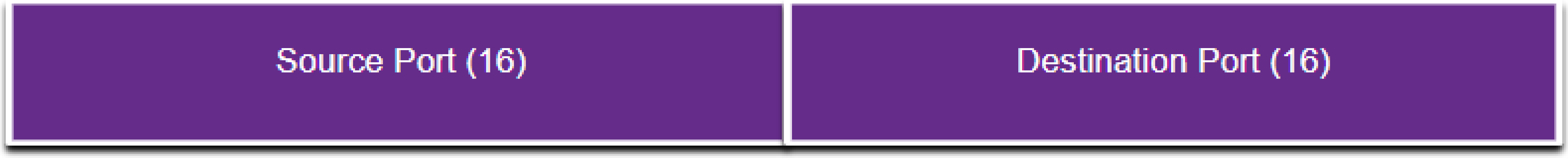
Port Numbers

Port Numbers

Multiple Separate Communications

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

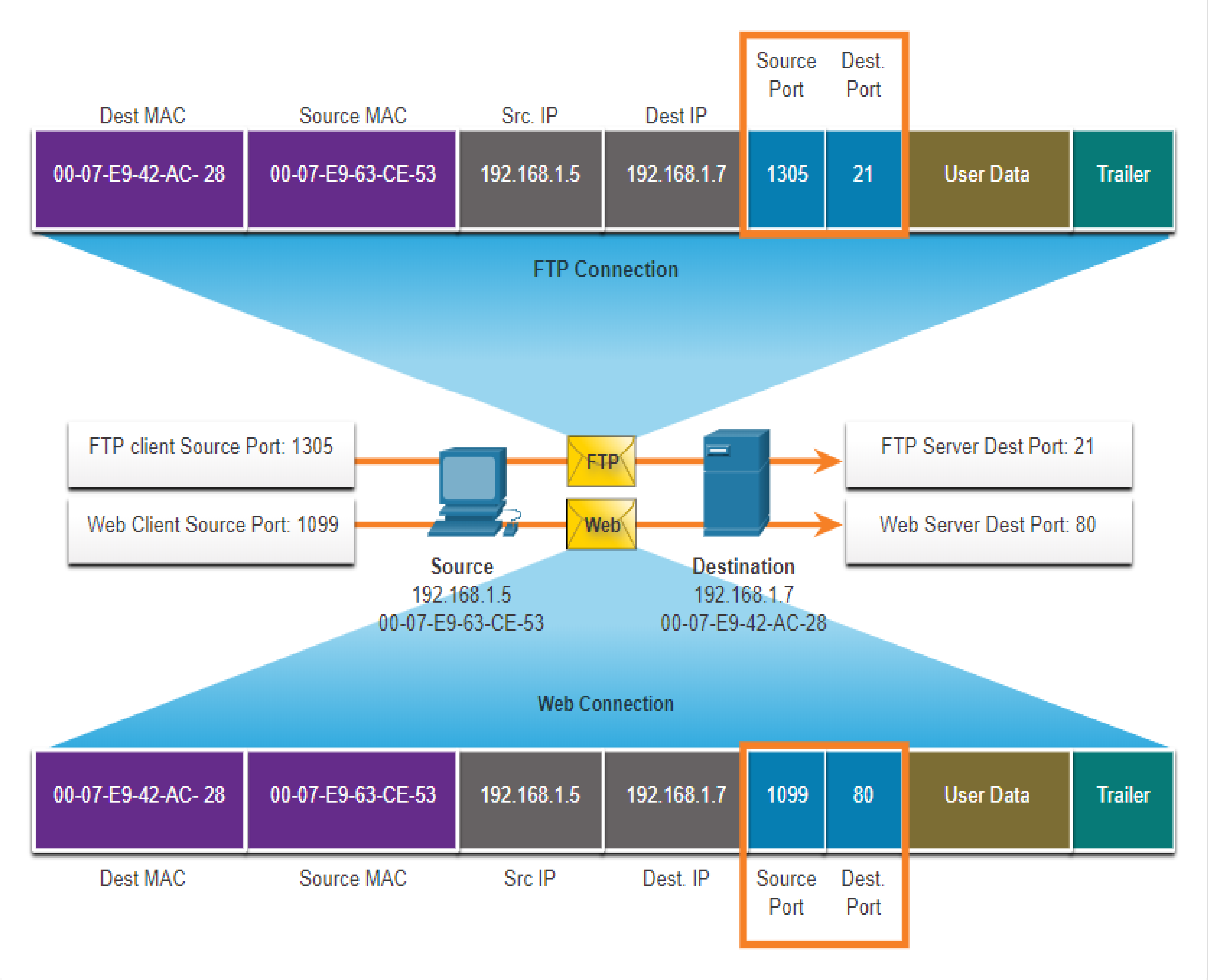
The source port number is associated with the originating application on the local host whereas the destination port number is associated with the destination application on the remote host.



Port numbers

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**	•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**	•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 **D ynamic Ports**	**49,152 to 65,535**	•These ports are also known as *ephemeral ports*. •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.

Port Number Groups (Cont.)

Well-Known Port Numbers

Port Number	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.

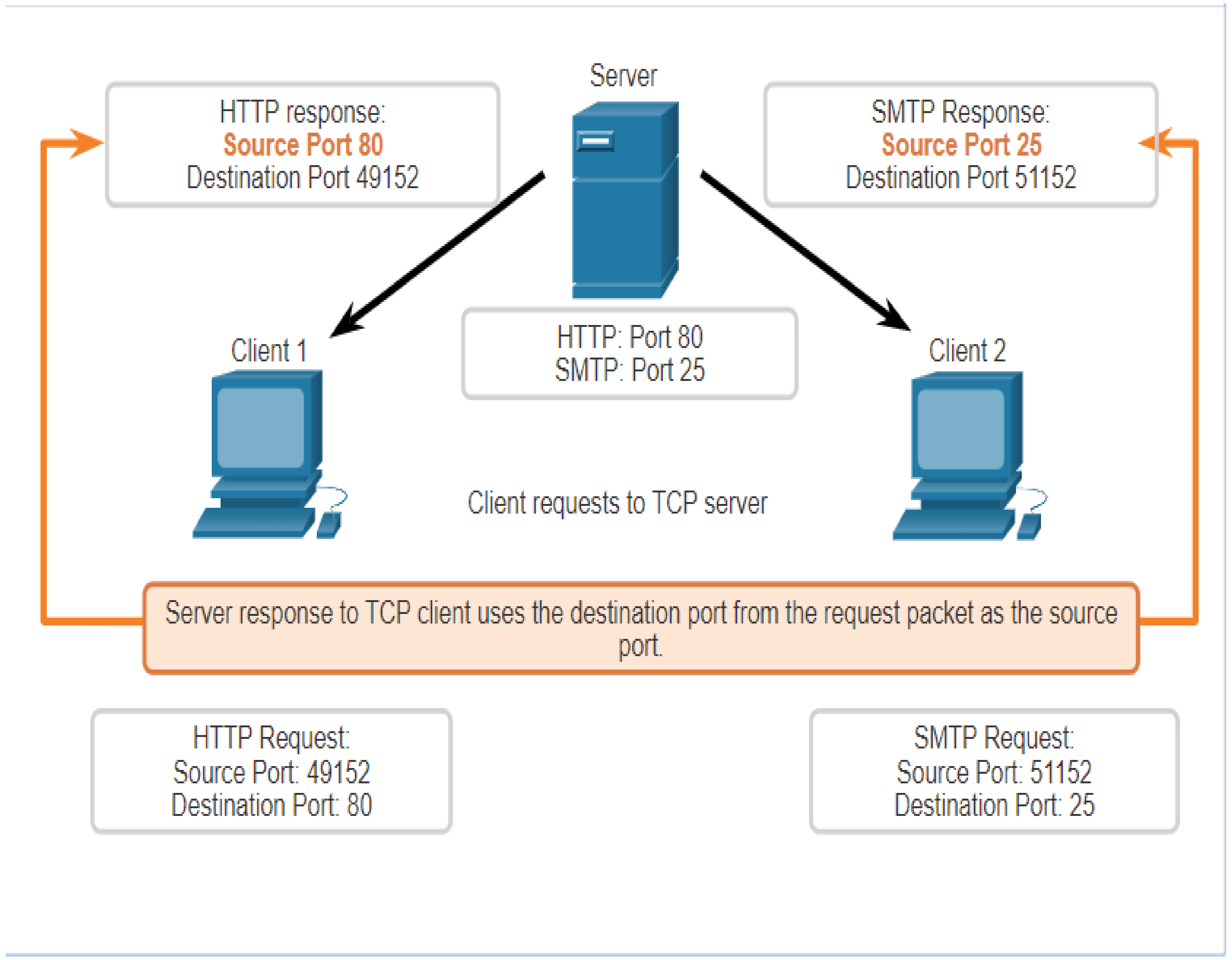
```
C:\> netstat
Active Connections
Proto Local Address          Foreign Address          State
TCP   192.168.1.124:3126     192.168.0.2:netbios-ssn  ESTABLISHED
TCP   192.168.1.124:3158     207.138.126.152:http     ESTABLISHED
TCP   192.168.1.124:3159     207.138.126.169:http     ESTABLISHED
TCP   192.168.1.124:3160     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
```

TCP Communication Process

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.

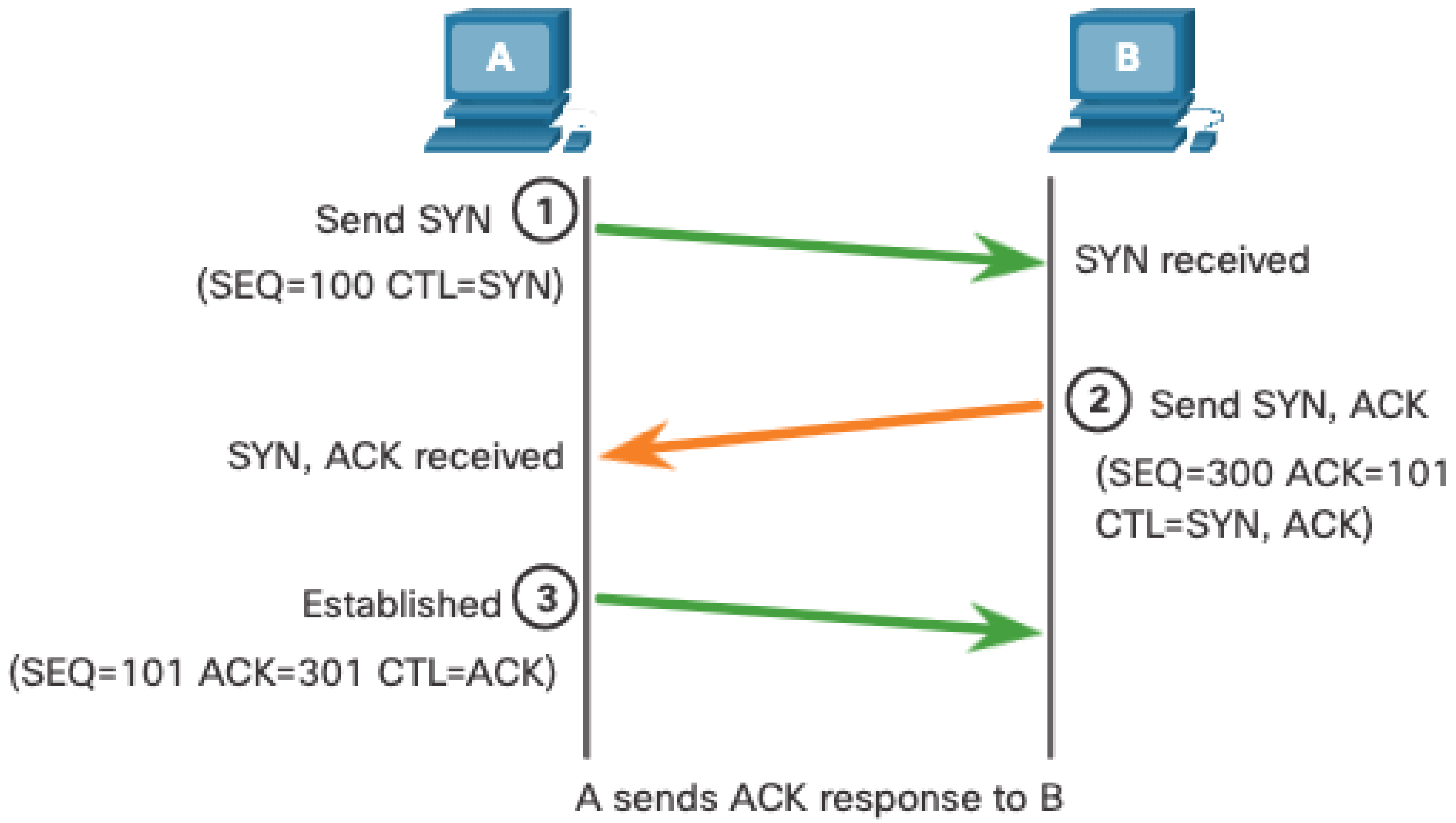


TCP Connection Establishment

Step 1: The initiating client requests a client-to-server communication session with the server.

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

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



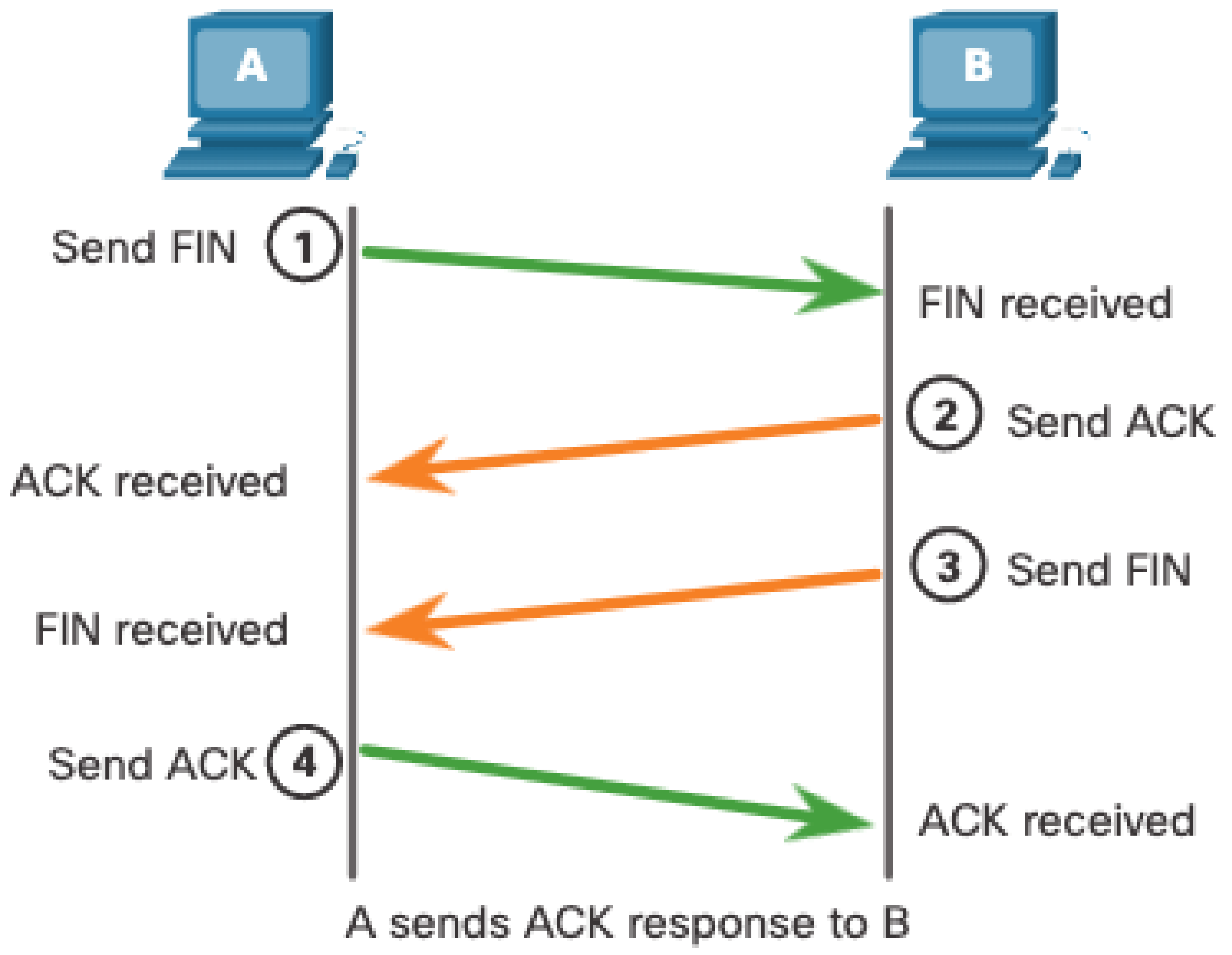
Session Termination

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

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

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

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



TCP Three-Way Handshake Analysis

Functions of the Three-Way Handshake:

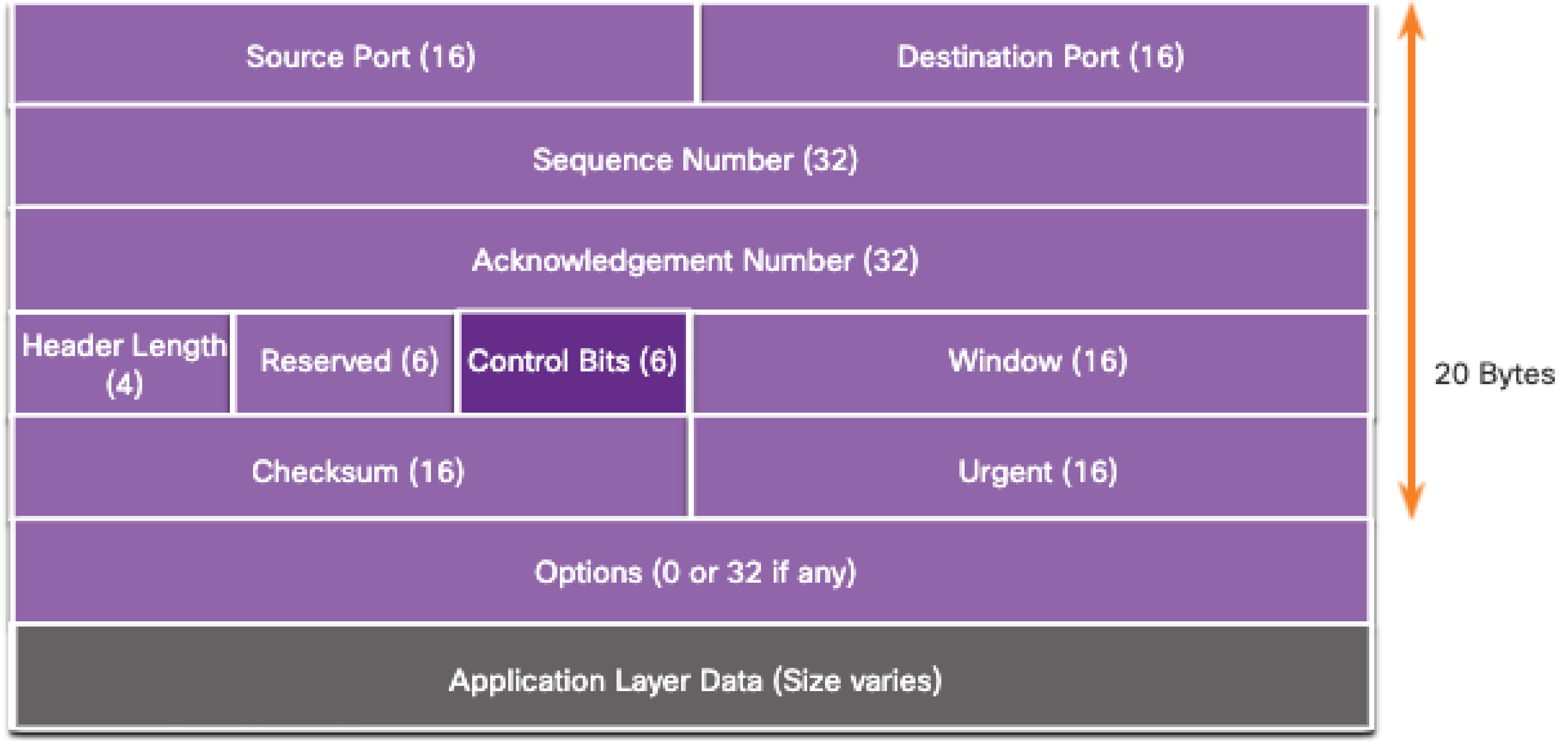
- It establishes that the destination device is present on the network.
- It verifies that the destination device has an active service and is accepting requests on the destination port number that the initiating client intends to use.
- It informs the destination device that the source client intends 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 (Cont.)

The six control bit flags are as follows:

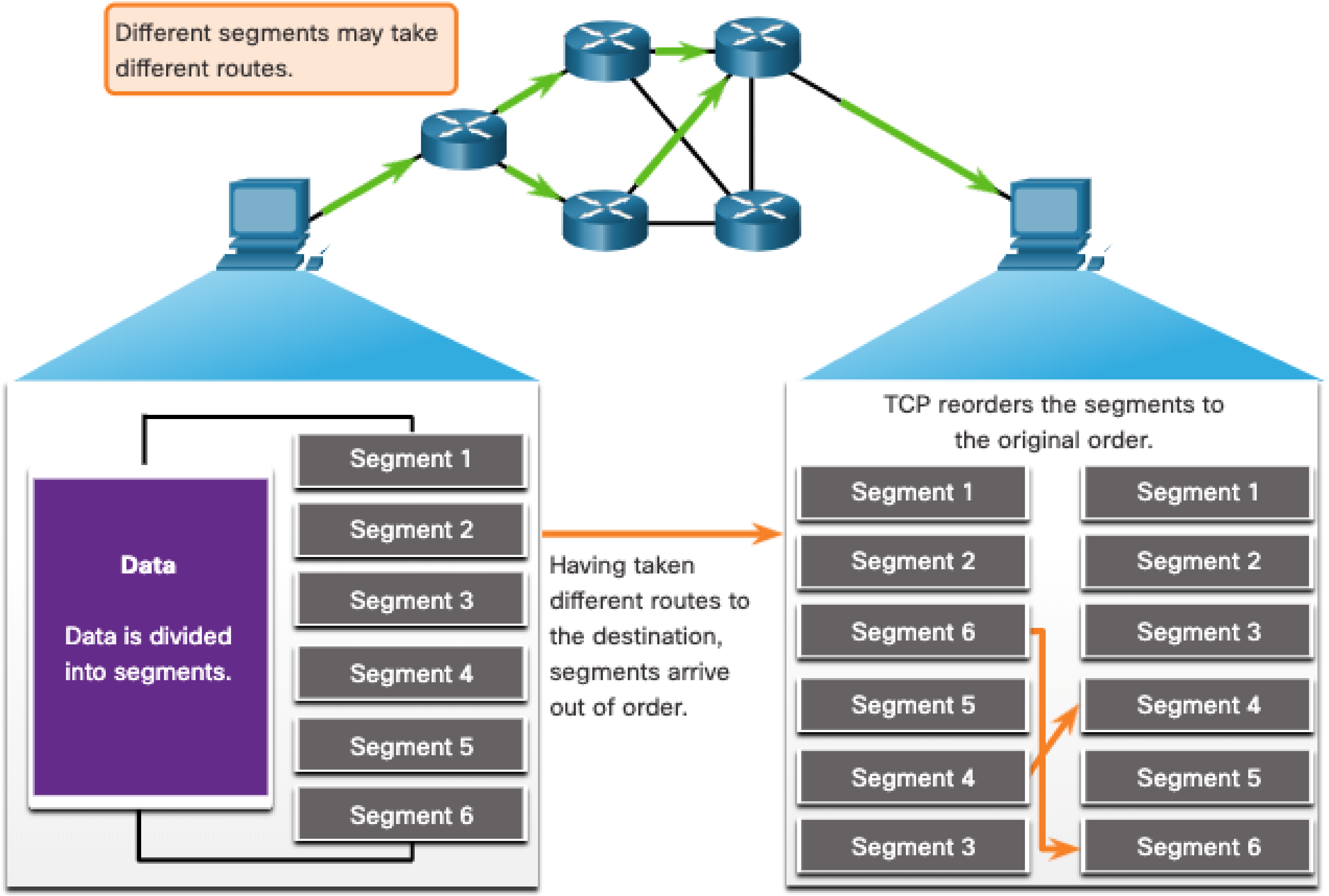
- **URG** - Urgent pointer field significant
- **ACK** - Acknowledgment flag used in connection establishment and session termination
- **PSH** - Push function
- **RST** - Reset the connection when an error or timeout occurs
- **SYN** - Synchronize sequence numbers used in connection establishment
-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



Reliability and Flow Control

TCP Reliability- Guaranteed and Ordered Delivery

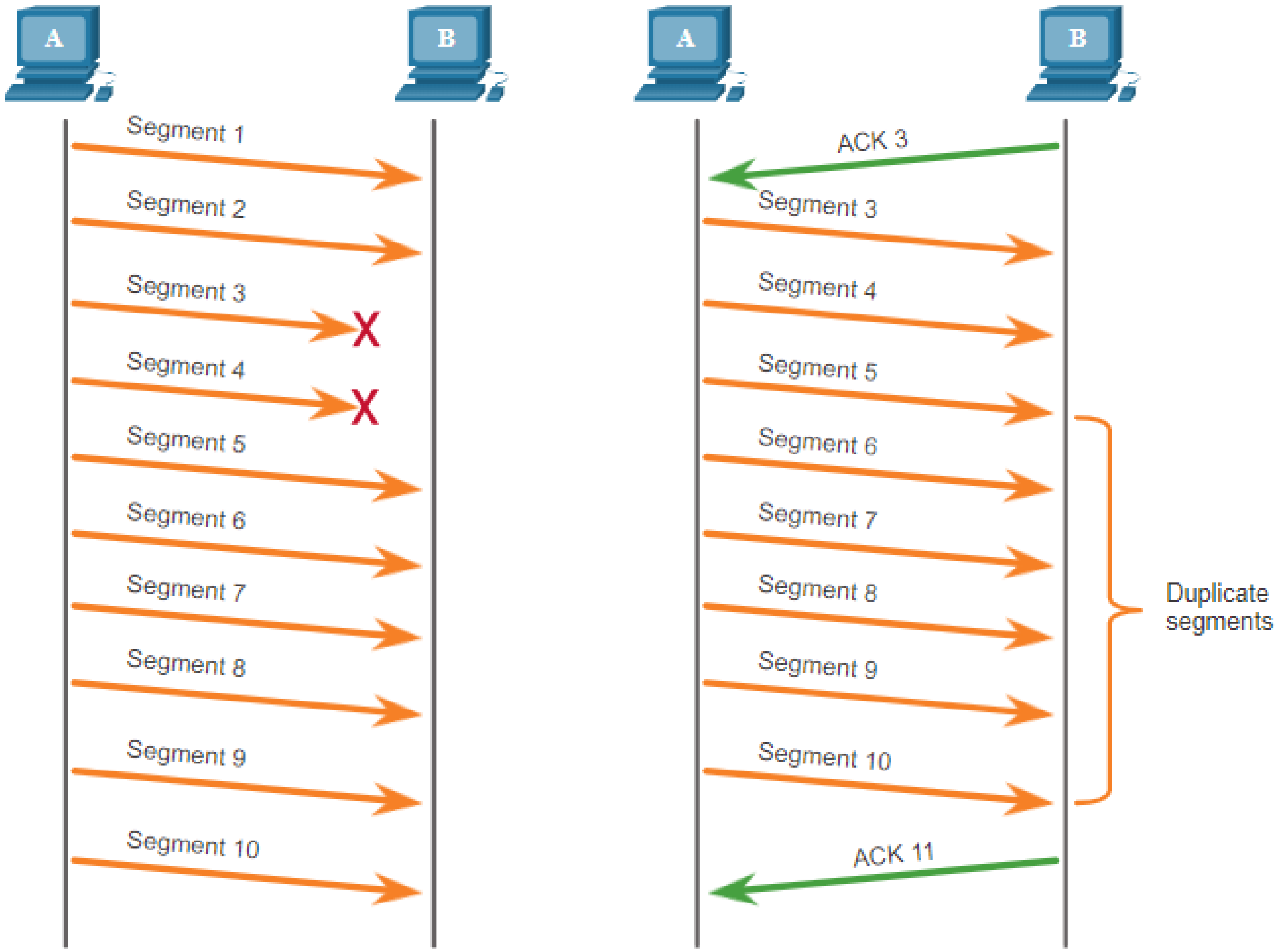
- TCP can also help maintain the flow of packets so that devices do not become overloaded.
- There may be times when TCP segments do not arrive at their destination or arrive out of order.
- All the data must be received and the data in these segments must be reassembled into the original order.
- Sequence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



TCP Reliability – Data Loss and Retransmission

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

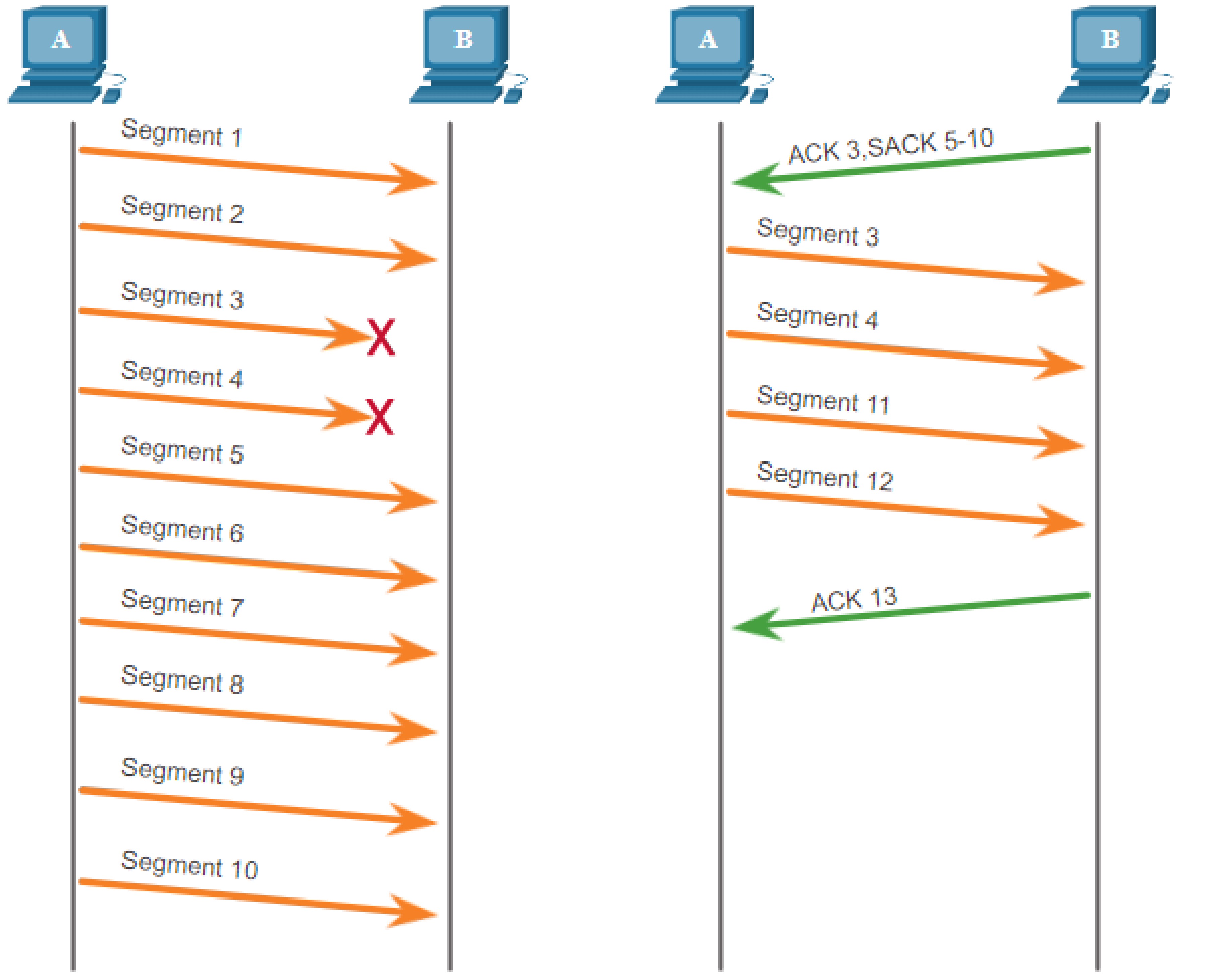
TCP provides methods of managing these segment losses. Among these is a mechanism to retransmit segments for unacknowledged data.



TCP Reliability – 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.

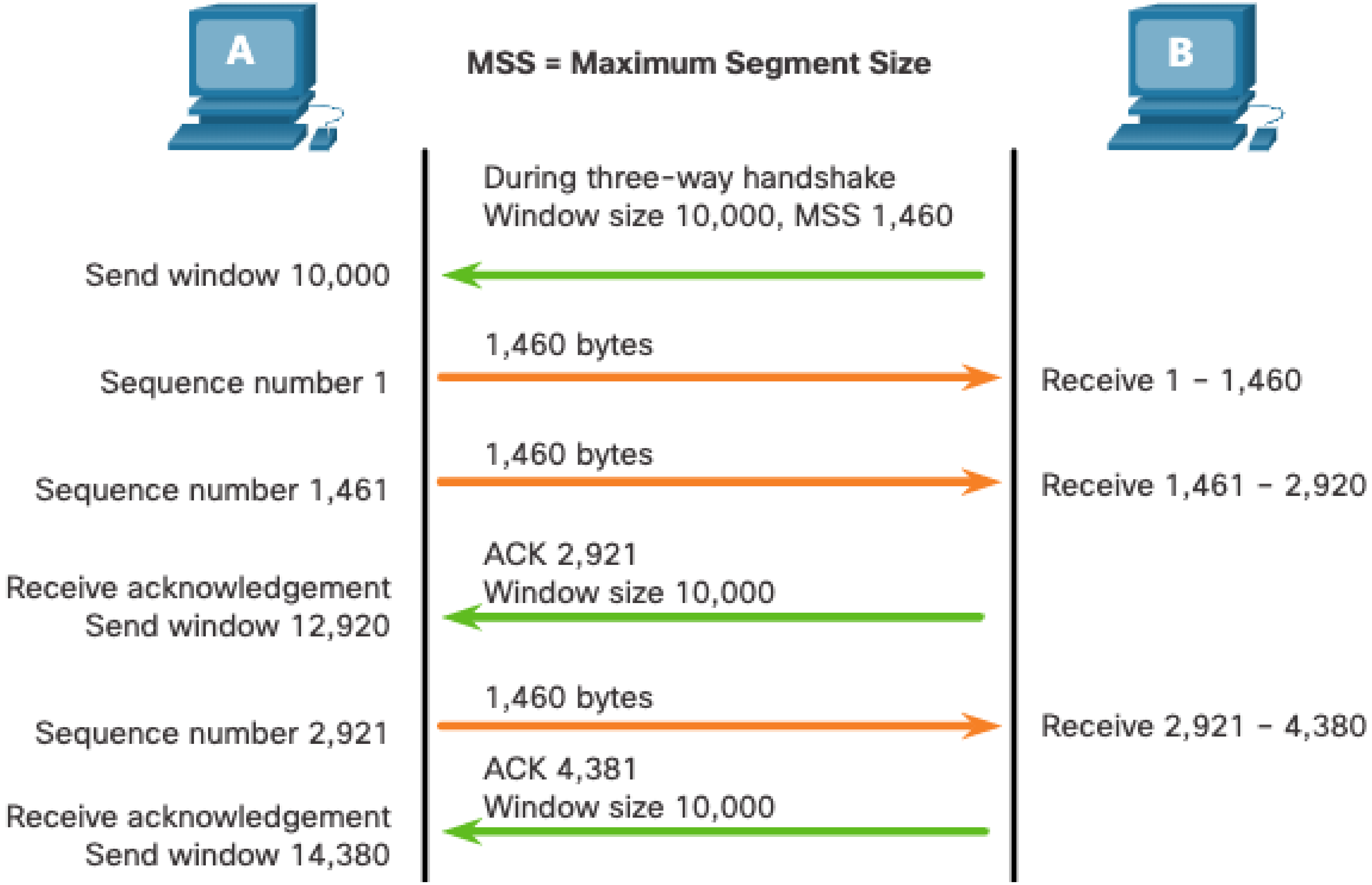
If both hosts support SACK, the receiver can explicitly acknowledge which segments (bytes) were received including any discontinuous segments.



TCP Flow Control – Window Size and Acknowledgments

TCP also provides mechanisms for flow control as follows:

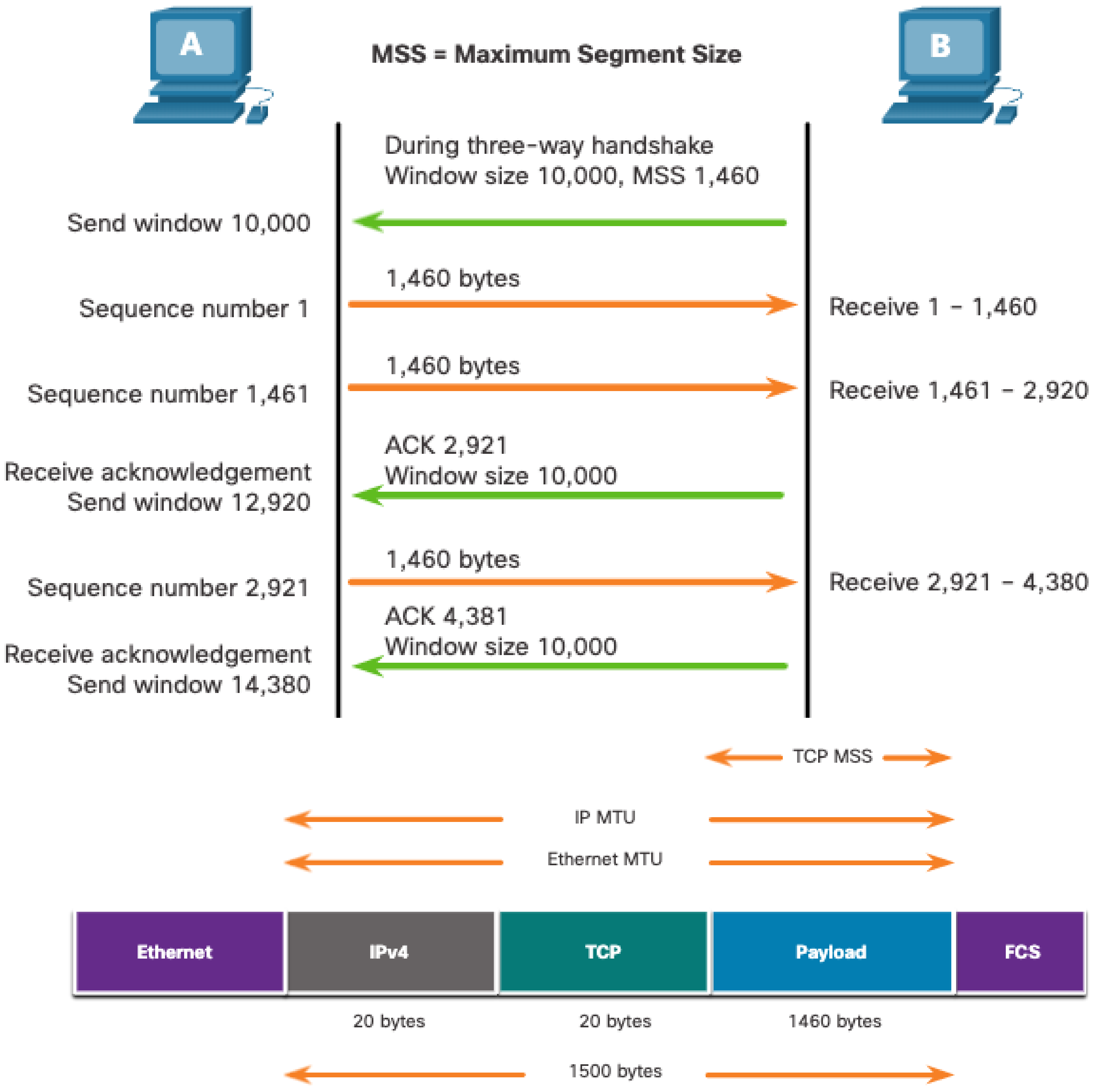
- Flow control is the amount of data that the destination can receive and process reliably.
- Flow control helps maintain the reliability of TCP transmission by adjusting the rate of data flow between source and destination for a given session.



TCP Flow Control – Maximum Segment Size

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

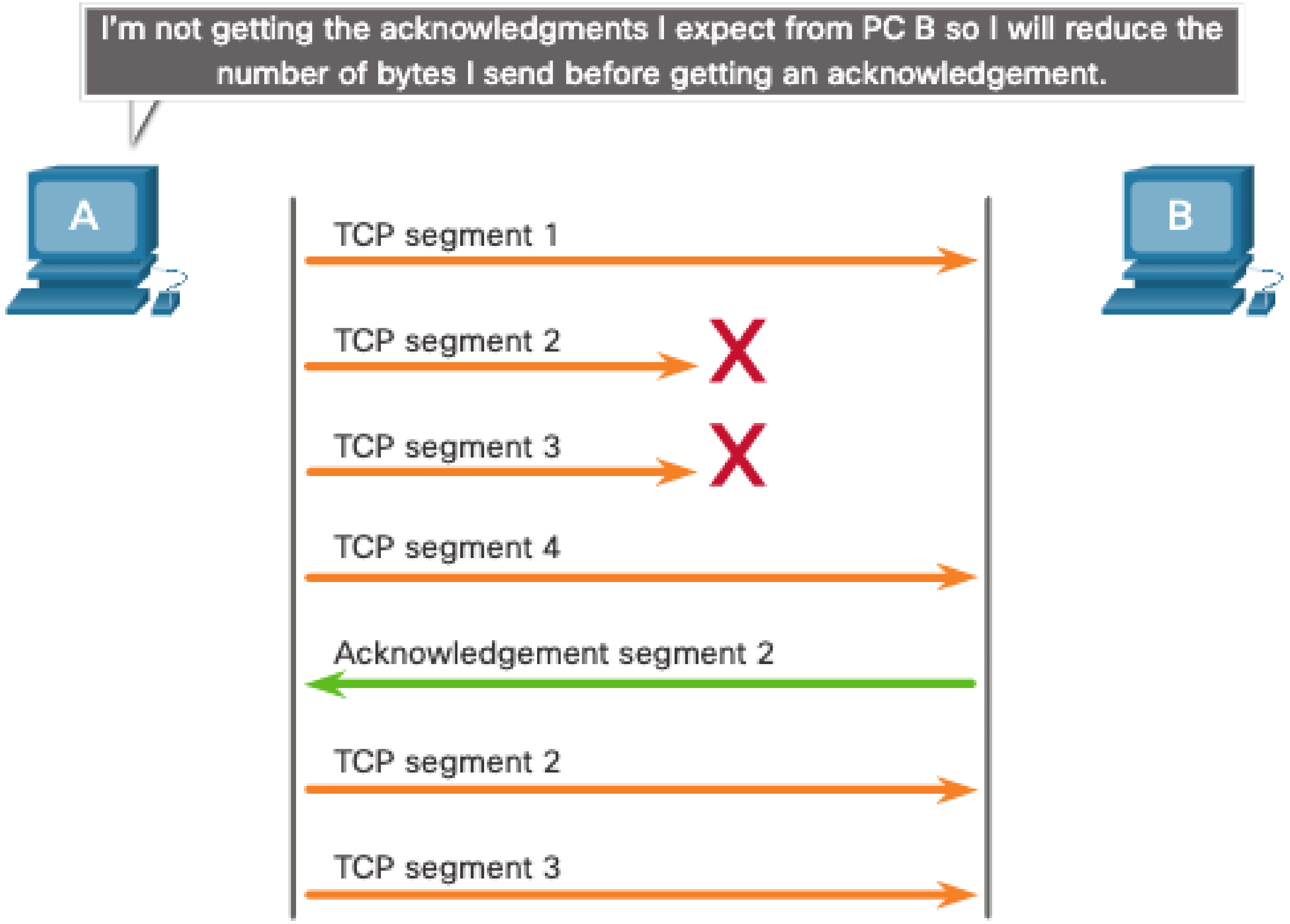
- A common MSS is 1,460 bytes when using IPv4.
- A host determines the value of its MSS field by subtracting the IP and TCP headers from the Ethernet maximum transmission unit (MTU), which is 1500 bytes be default.
- 1500 minus 40 (20 bytes for the IPv4 header and 20 bytes for the TCP header) leaves 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.

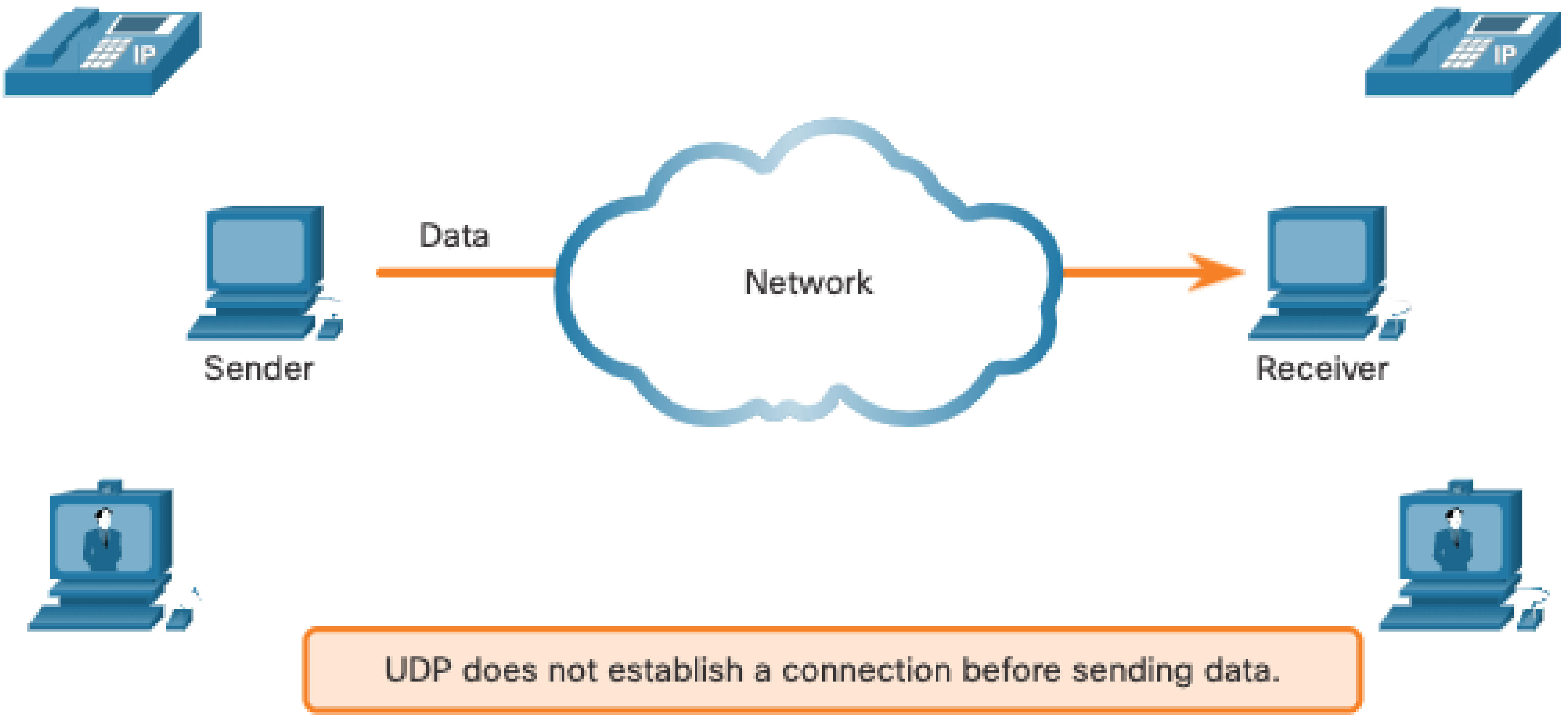


UDP Communication

ASTANA IT UNIVERSITY

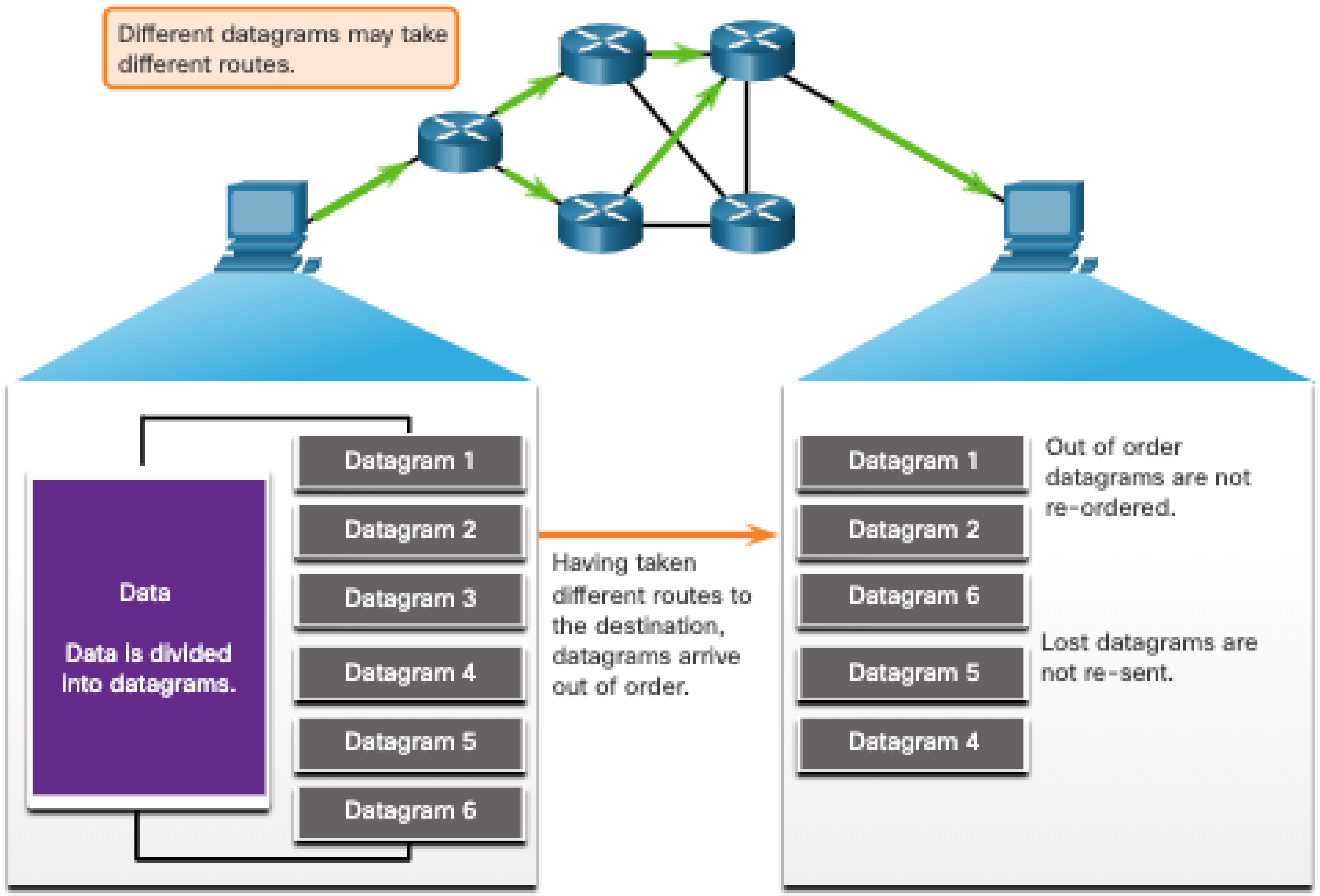
UDP Low Overhead versus Reliability

UDP does not establish a connection. UDP provides low overhead data transport because it has a small datagram header and no network management traffic.



UDP Datagram Reassembly

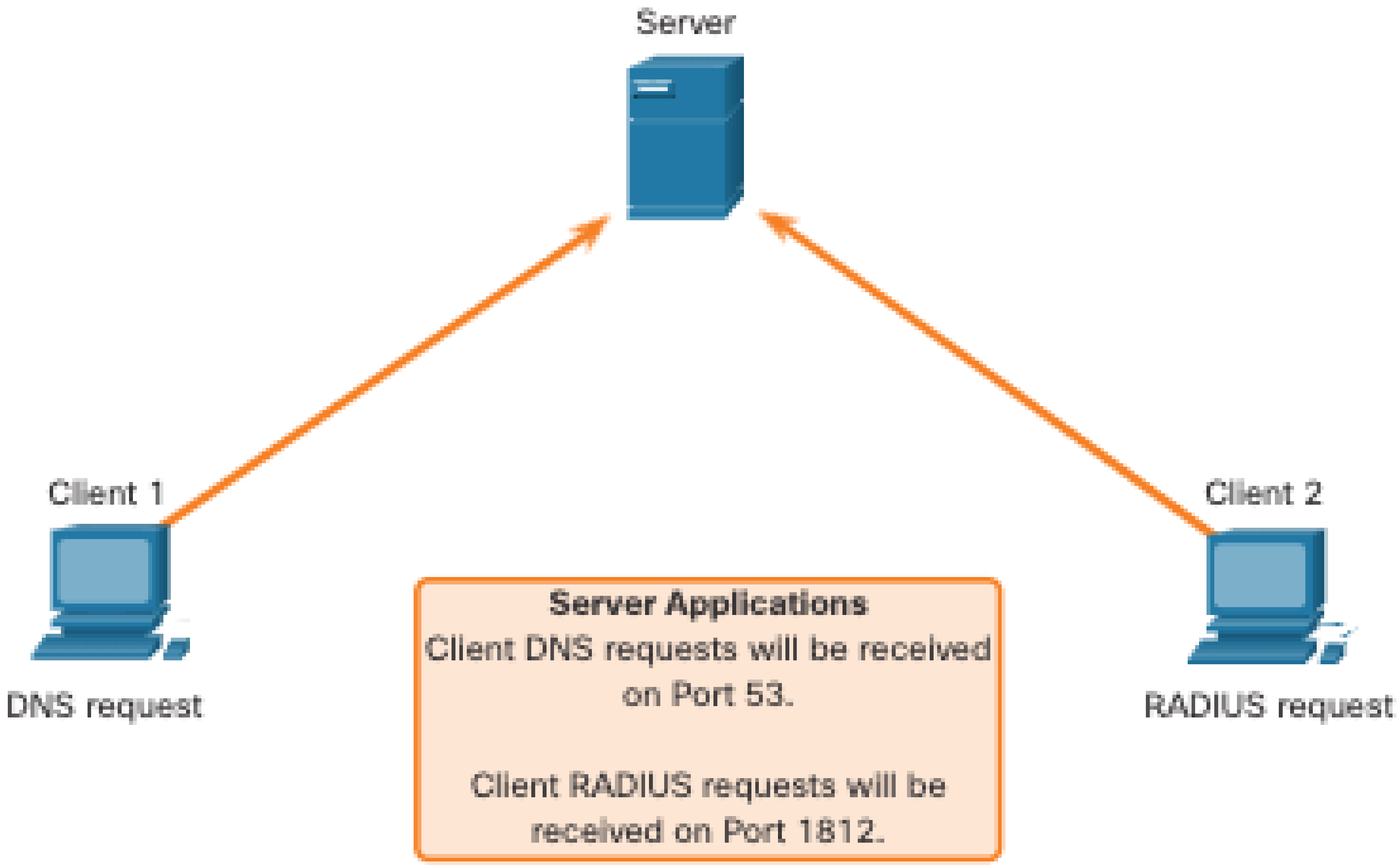
- UDP does not track sequence numbers the way TCP does.
- UDP has no way to reorder the datagrams into their transmission order.
- UDP simply reassembles the data in the order that it was received and forwards it to the application.



UDP Server Processes and Requests

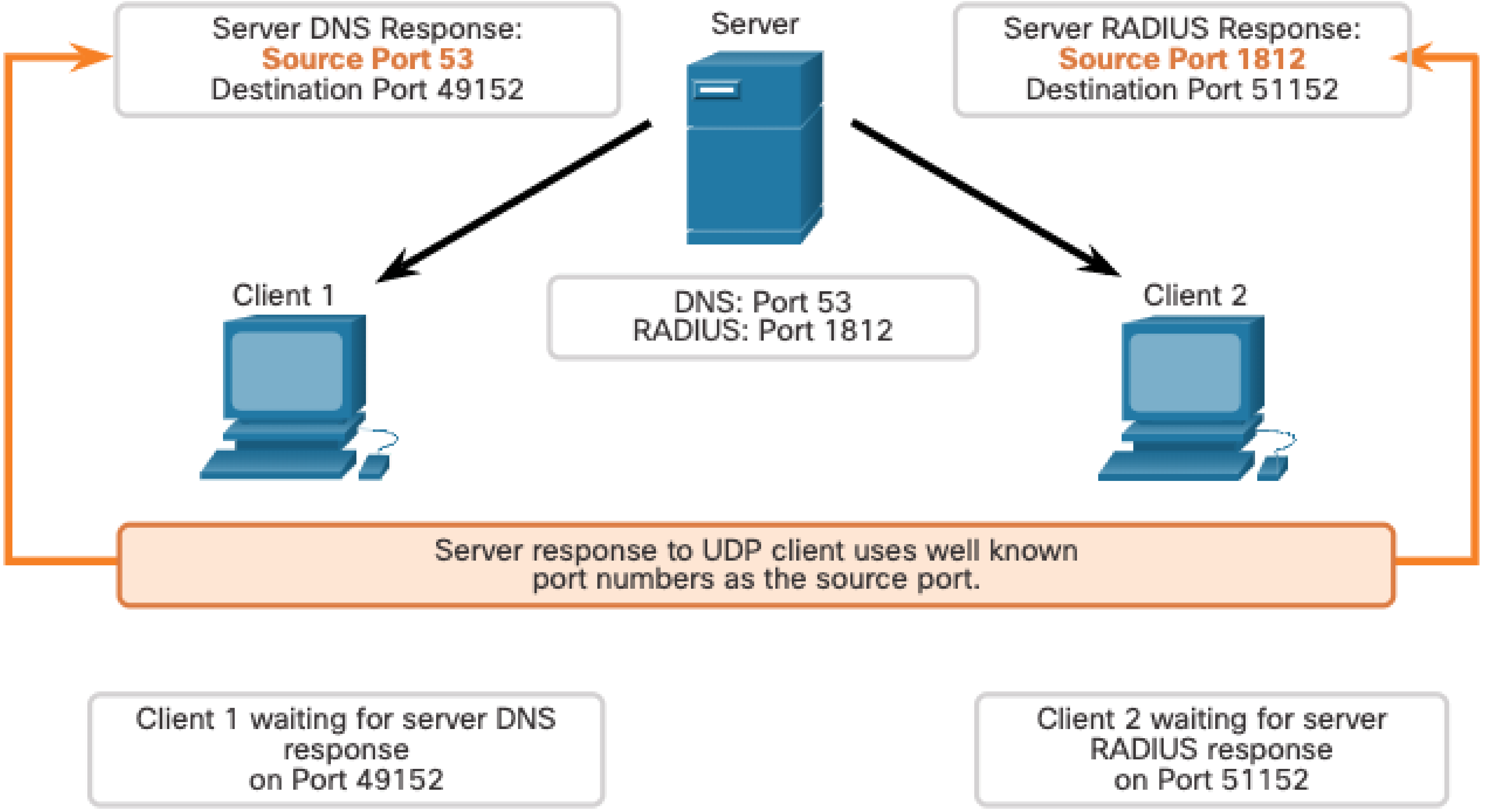
UDP-based server applications are assigned well-known or registered port numbers.

UDP receives a datagram destined for one of these ports, it forwards the application data to the appropriate application based on its port number.



UDP Client Processes

- The UDP client process dynamically selects a port number from the range of port numbers and uses this as the source port for the conversation.
- The destination port is usually the well-known or registered port number assigned to the server process.
- After a client has selected the source and destination ports, the same pair of ports are used in the header of all datagrams in the transaction.



THANK YOU!