Chapter 4

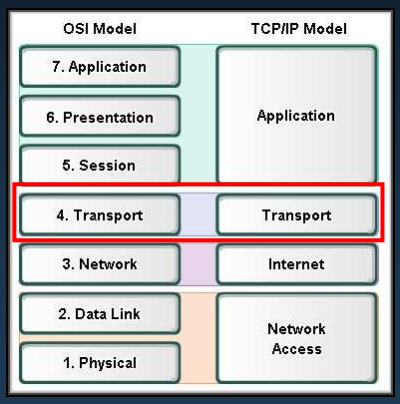
OSI Transport Layer

Note for Instructors

- These presentations are the result of a collaboration among the instructors at St. Clair College in Windsor, Ontario.
- Thanks must go out to Rick Graziani of Cabrillo College. His material and additional information was used as a reference in their creation.
- If anyone finds any errors or omissions, please let me know at:
  - tdame@stclaircollege.ca.

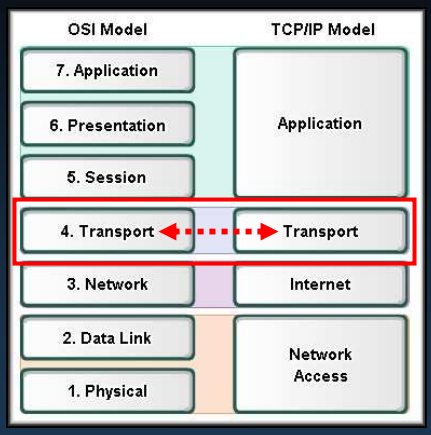
OSI Transport Layer

Roles of the Transport Layer

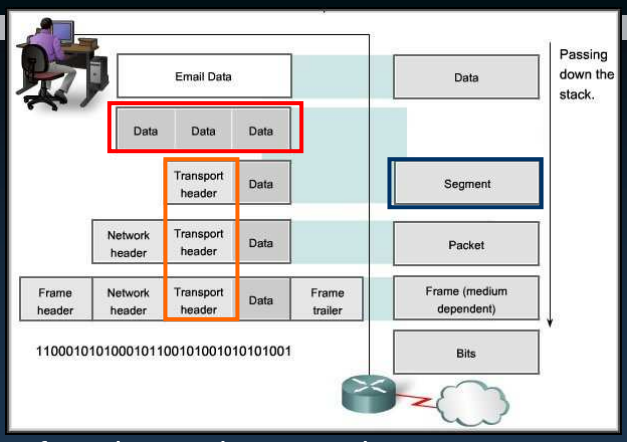


Purpose of the Transport Layer

- The Layer 4 data stream is a **logical connection between the endpoints** of a network.
- It **provides transport services** from a host to a destination.
- This service is sometimes referred to as an **end-to-end service**.
- Provides two protocols:
  - **TCP** – Transmission Control Protocol
  - **UDP** – User Datagram Protocol



Purpose of the Transport Layer



- We will be focusing on the Layer that:
  - Segments the data.
  - Creates and inserts the header for either the TCP or the UDP protocol.

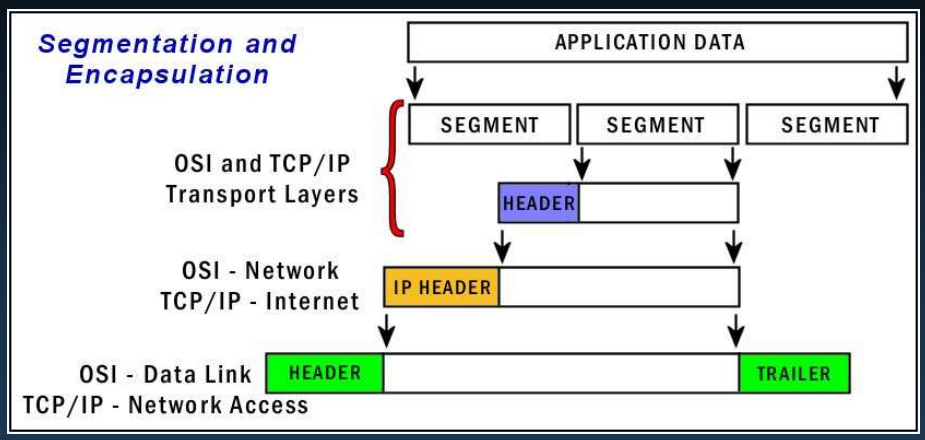
Purpose of the Transport Layer

- Primary responsibilities:
  - Tracking the individual communications between applications on the source and destination hosts.
  - Segmenting the data and managing each piece.
  - Reassembling the segments into streams of application data.
  - Identifying the different applications.
  - Performing flow control between end users.
  - Enabling error recovery.
  - Initiating a session.

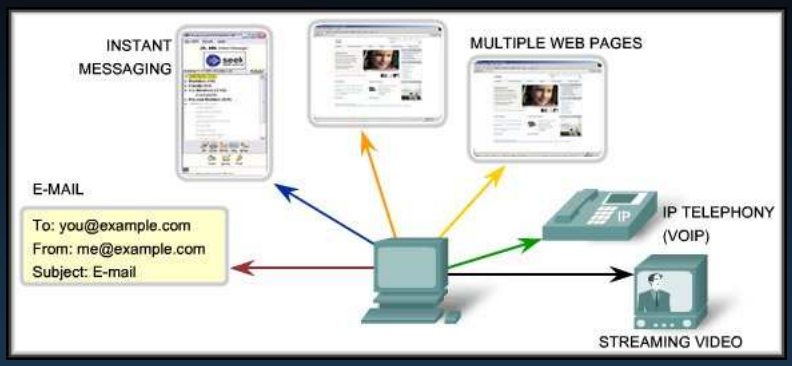
Segmentation and Reassembly

- An Ethernet frame has a maximum frame size or **Maximum Transmission Unit (MTU)** of 1,518 bytes.
  - When a larger message must be sent, the application data must be segmented into sections that ***will not exceed the maximum size***.
  - The segment size must also take into account the encapsulation process that must take place before the frame can be transmitted.

Segmentation and Reassembly

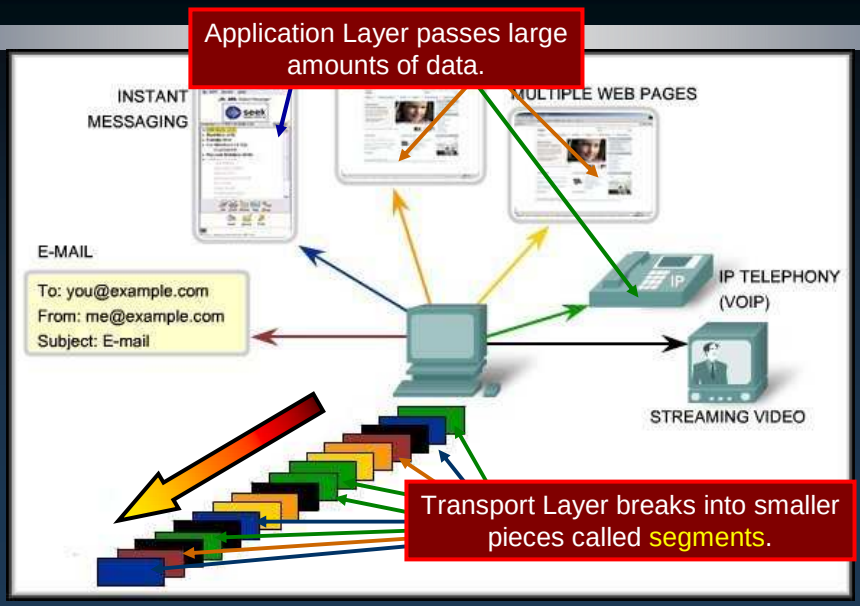


Tracking Individual Conversations

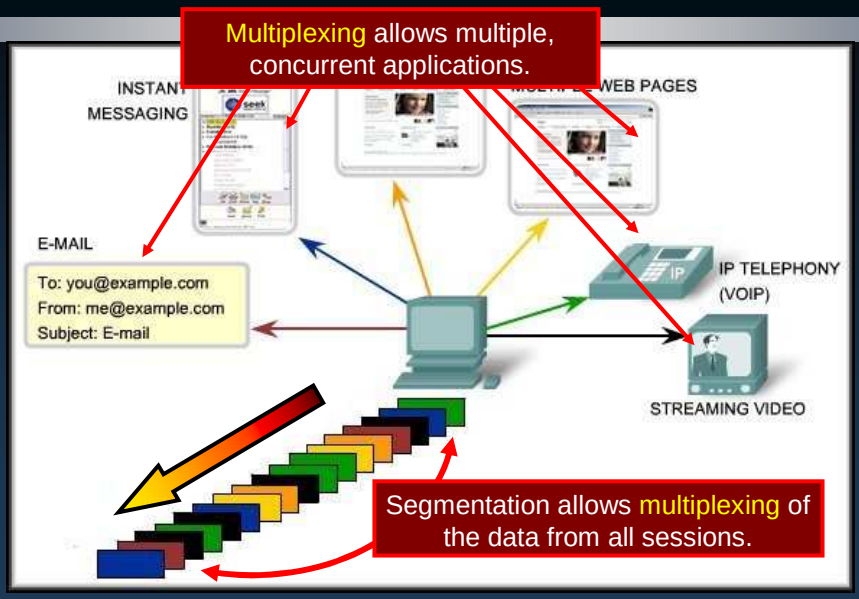


- Any host can have multiple applications running simultaneously.
- Transport Layer maintains these separate data streams.
  - For example, it makes sure that Instant Messaging data does not appear on the E-mail application.

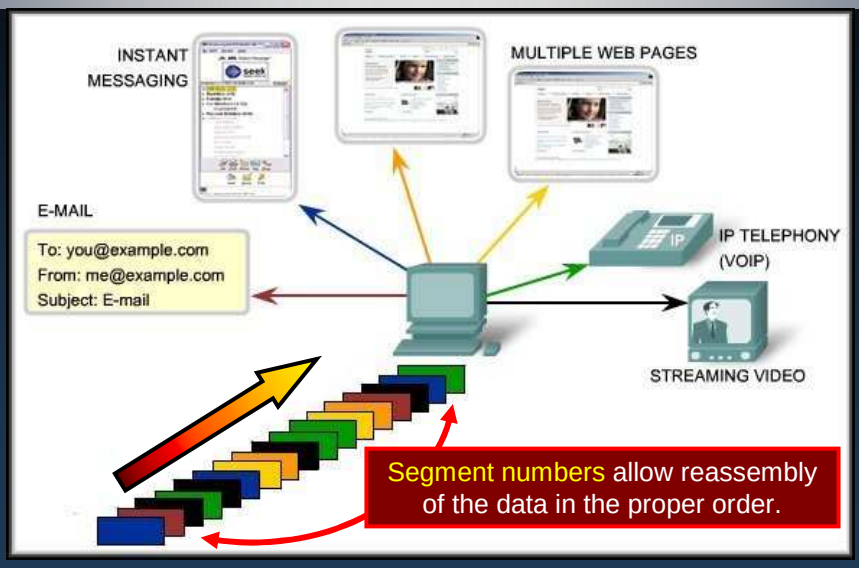
Segmenting Data



Segmenting Data

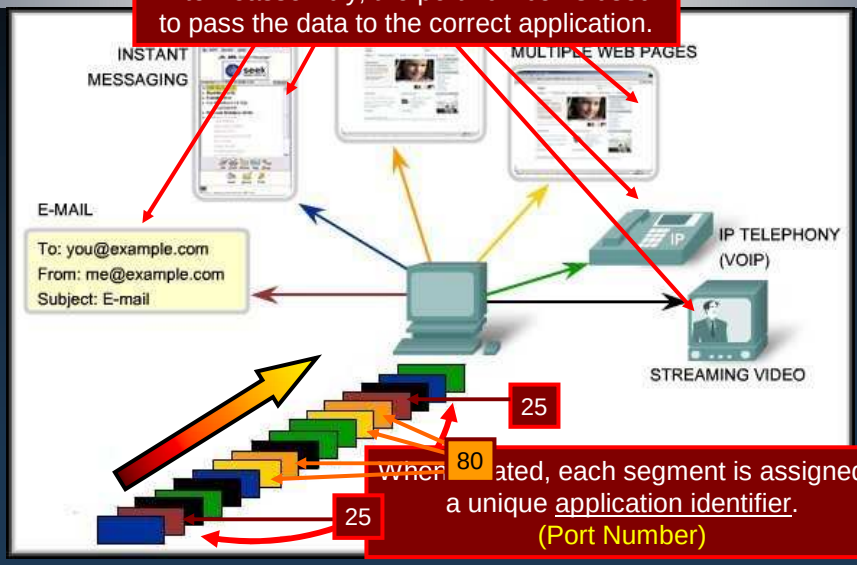


Reassembling Segments



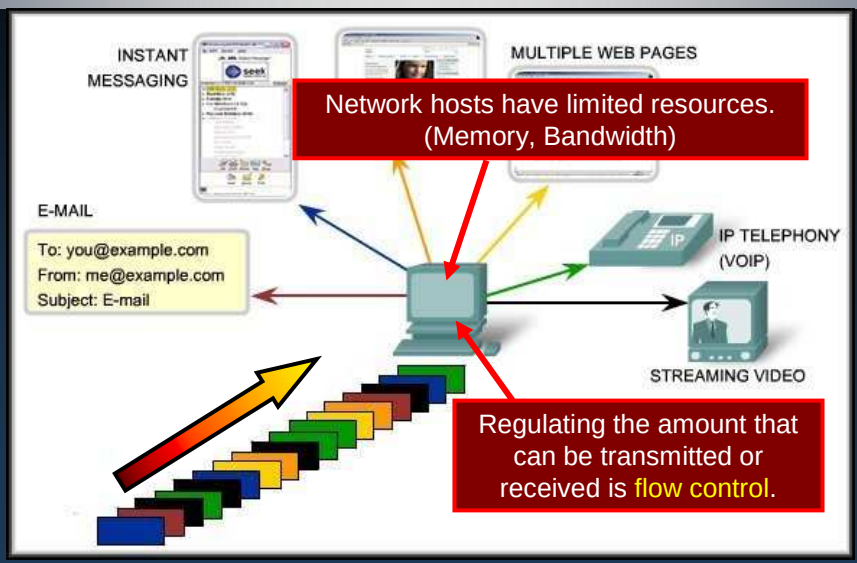
Reassembling Segments

After reassembly, the port number is used to pass the data to the correct application.

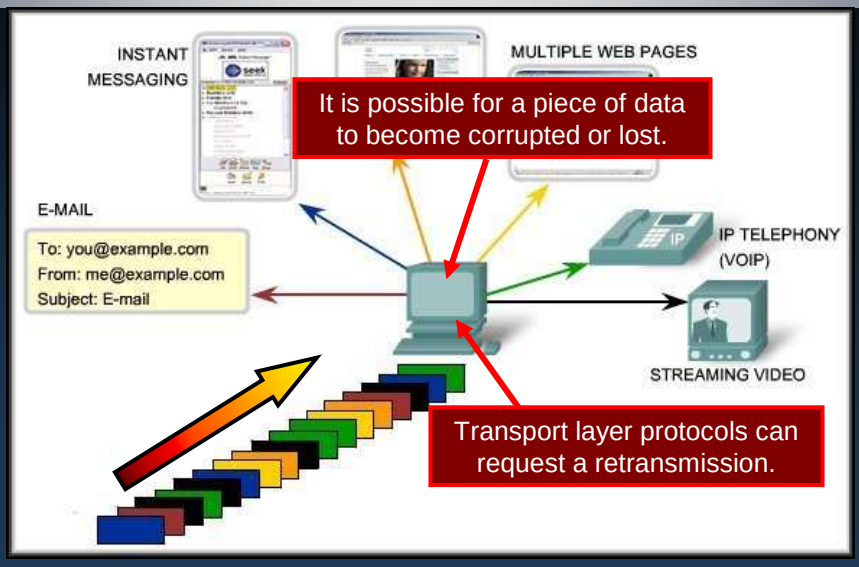


When created, each segment is assigned a unique application identifier. (Port Number)

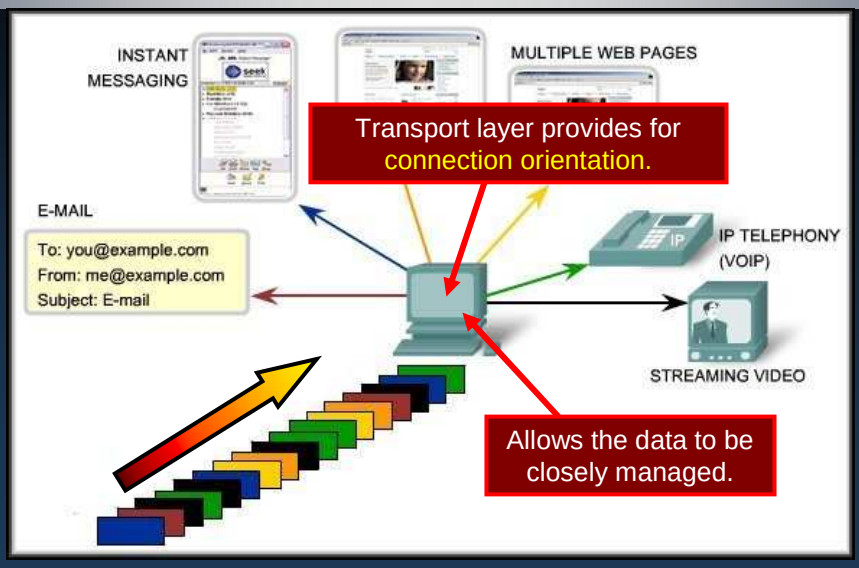
Flow Control



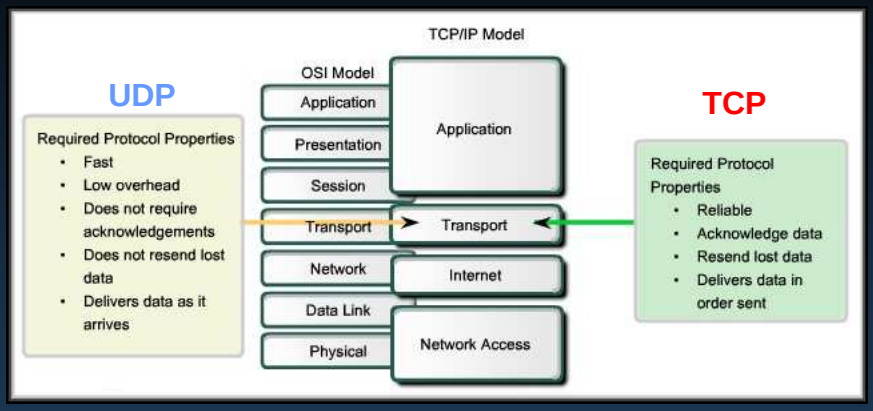
Error Recovery



Initiating a Session



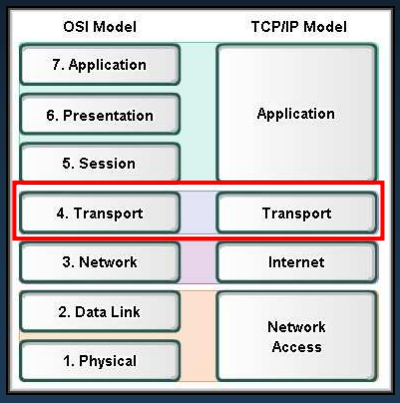
Supporting Reliable Communication



- Different applications have different requirements.
- Different protocols have been developed to meet them.

OSI Transport Layer

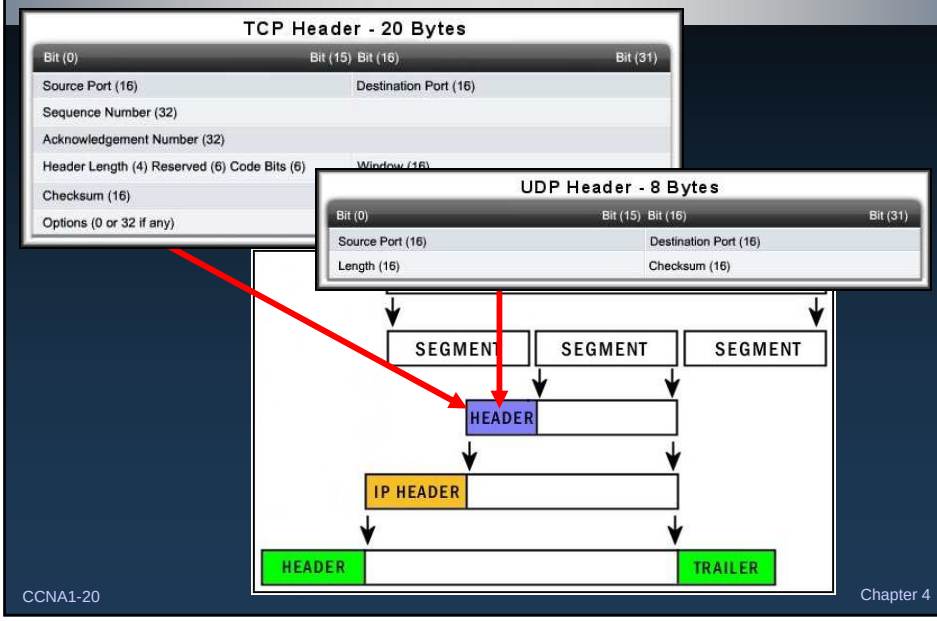
TCP and UDP



TCP and UDP

- Two most common Transport Layer protocols

TCP and UDP



User Datagram Protocol (UDP)

UDP Datagram

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Length (16)	Checksum (16)	
APPLICATION LAYER DATA (Size varies)		

8 Bytes

- Connectionless
- "Best Effort" delivery
- Low overhead

Example Applications

Domain Name System (DNS)
Online Games
Voice over IP (VoIP)
Dynamic Host Configuration Protocol (DHCP)
Trivial File Transfer Protocol (TFTP)

Transmission Control Protocol (TCP)

TCP Segment

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Sequence Number (32)		
Acknowledgement Number (32)		
Header Length (4) Reserved (6) Code Bits (6)	Window (16)	
Checksum (16)	Urgent (16)	
Options (0 or 32 if any)		
APPLICATION LAYER DATA (Size varies)		

20 Bytes

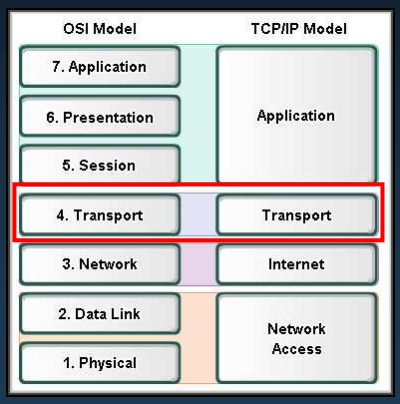
- Connection-oriented
- Reliable delivery
- Error checking
- Flow control

Example Applications

Hypertext Transfer Protocol (HTTP)
File Transfer Protocol (FTP)
Telnet
Simple Message Transfer Protocol (SMTP)

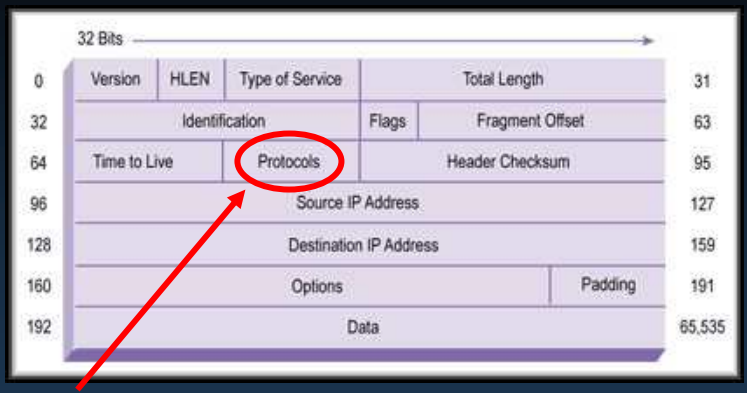
OSI Transport Layer

Port Addressing



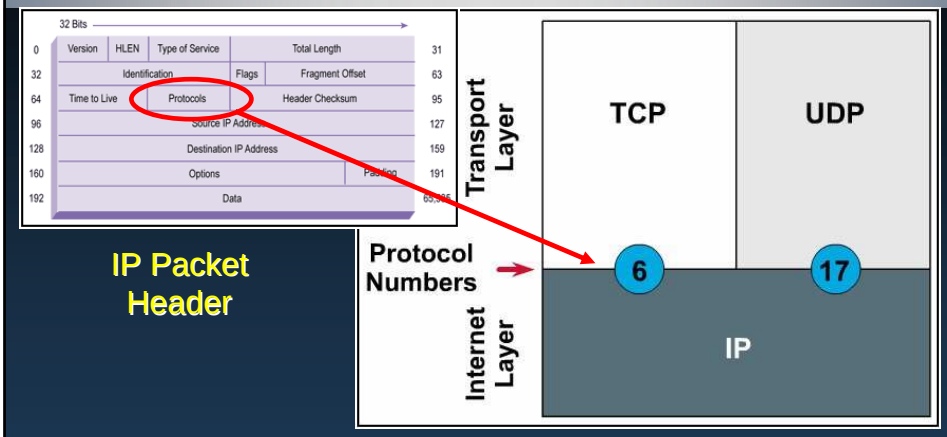
Identifying the Conversations

IP Packet Header



- *At the TCP/IP Internet Layer*:
  - The IP Packet Header has a Protocol field that specifies whether the segment is TCP *or* UDP.

Identifying the Conversations



- When a packet is encapsulated at the Network Layer, it is coded to identify the source of the packet (TCP *or* UDP).

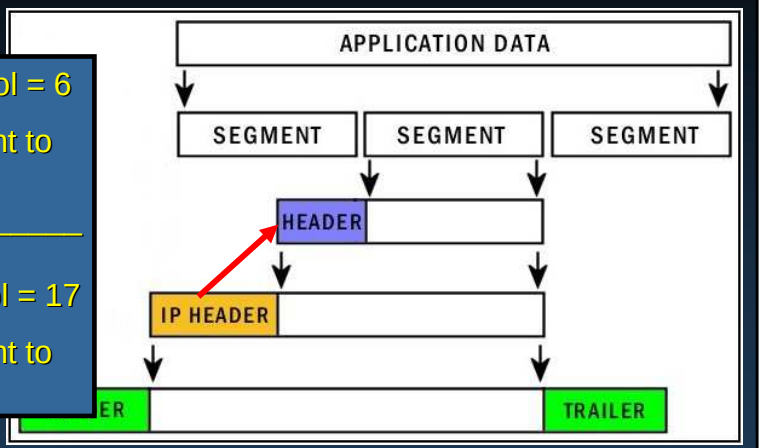
Identifying the Conversations

IP Protocol = 6

Data sent to TCP

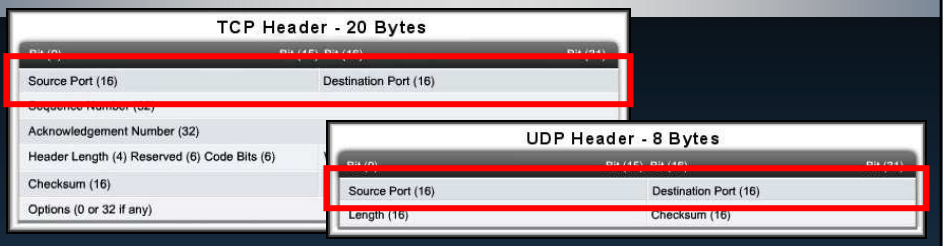
IP Protocol = 17

Data sent to UDP



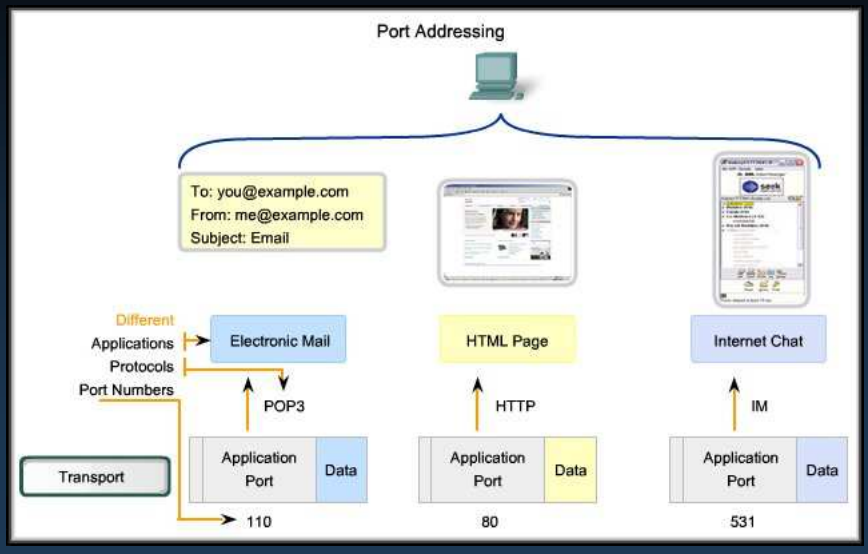
- When a packet is decapsulated at the destination, the code is used to send the packet to the proper protocol (TCP *or* UDP).

Identifying the Conversations



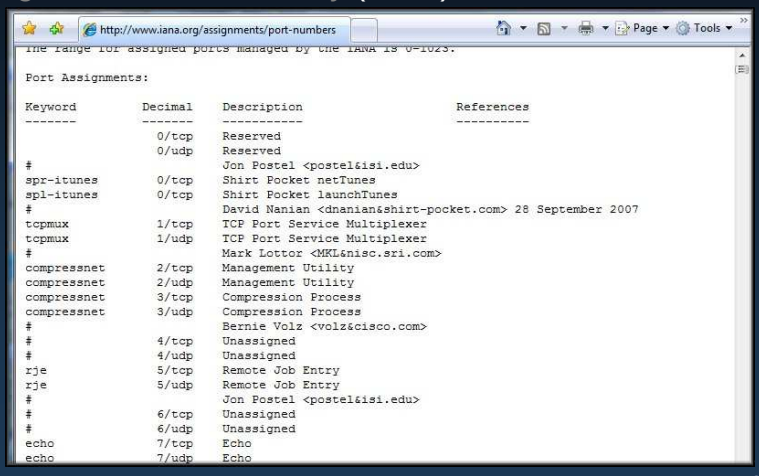
- Both TCP and UDP use port numbers to pass information to the upper layers.
- These ports are actually termed sockets.
  - A socket is simply the combination of the device's IP address and the source/destination port for the data, separated by a colon.
  - e.g. 207.134.65.2:80 references an HTTP socket.

Identifying the Conversations



Port Addressing Types and Tools

- Port numbers are managed and assigned by the Internet Assigned Number Authority (IANA).



Port Addressing Types and Tools

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

- Some ports are reserved in both TCP and UDP, although applications might not be written to support them.
- Three groupings of port types:
  - Well-known or Contact Ports (0 to 1023).
  - Registered Ports (1024 through 49151).
  - Private and/or Dynamic Ports (49152 through 65535).

Port Addressing Types and Tools

- **Well-Known Ports:**
  - Reserved for common services and applications.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

20 – FTP Data

21 – FTP Control

23 – Telnet

25 – SMTP

110 – POP3

194 – IRC

443 – HTTPS

69 – TFTP

520 – RIP

Port Addressing Types and Tools

- **Registered Ports:**
  - Optional user processes and applications.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

1863 – MSN Messenger

8008 – Alternate HTTP

8080 – Alternate HTTP

1812 – RADIUS

2000 – Cisco VoIP

5004 – RTP

5060 – SIP (VoIP)

Port Addressing Types and Tools

- Dynamic Ports:
  - Assigned to a user application at connect time.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

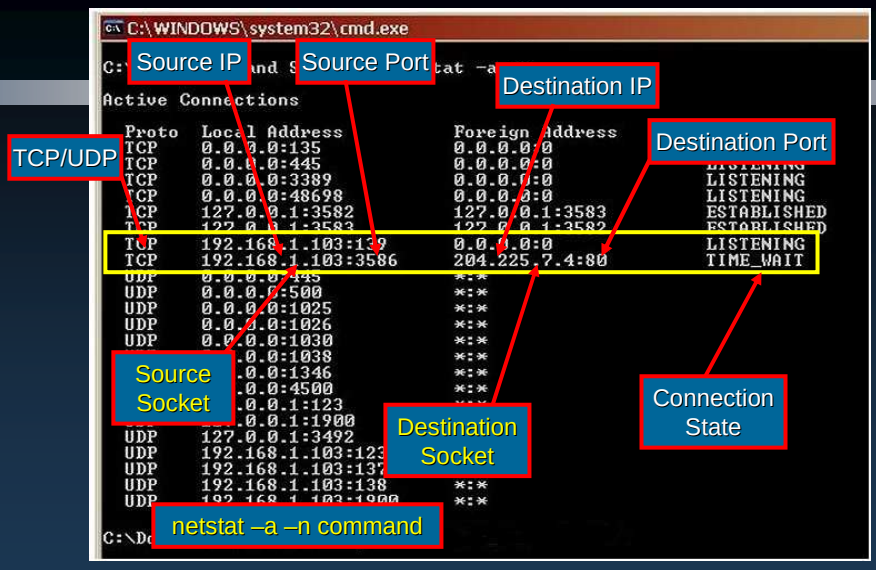
Dynamic port usage will become clearer as we move through the material.

STAY TUNED!

Port Addressing Types and Tools

- Using both TCP and UDP:
  - Some applications may use both TCP and UDP.
  - For example, the low overhead of UDP enables DNS to serve many client requests very quickly.
    - *Sometimes, however, sending the requested information may require the reliability of TCP. In this case, the well known port number of 53 is used by both protocols with this service.*

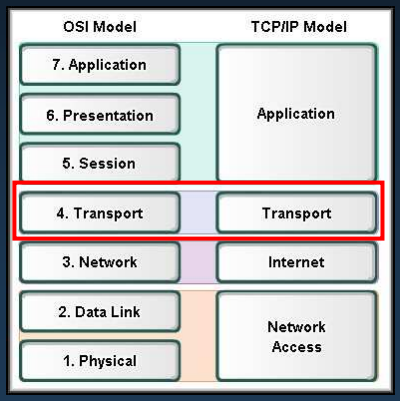
Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports



- Actually, when you open up a single web page, there are usually several TCP sessions created, not just one.

OSI Transport Layer

TCP
Communicating with Reliability



Transmission Control Protocol (TCP)

TCP Segment

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Sequence Number (32)		
Acknowledgement Number (32)		
Header Length (4) Reserved (6) Code Bits (6)	Window (16)	
Checksum (16)	Urgent (16)	
Options (0 or 32 if any)		
APPLICATION LAYER DATA (Size varies)		

20 Bytes

- Connection-oriented
- Reliable delivery
- Error checking
- Flow control

Example Applications

Hypertext Transfer Protocol (HTTP)
File Transfer Protocol (FTP)
Telnet
Simple Message Transfer Protocol (SMTP)

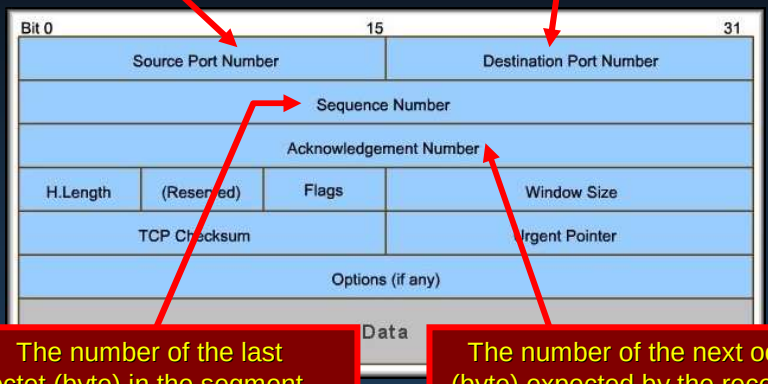
Making Conversations Reliable

- *The key difference between TCP and UDP is **reliability**.*
- TCP uses **connection-oriented** sessions.
  - Before any data is exchanged, the Transport Layer initiates a connection to the destination.
  - This connection allows the tracking of the session.
    - Sequence Numbers
    - Acknowledgments
  - Creates the **overhead** of TCP.
  - Reliability is achieved by having fields in the TCP header that have specific functions.

Making Conversations Reliable

TCP session that opened a connection. Usually a random value above 1023.

Upper Layer application on the remote site.



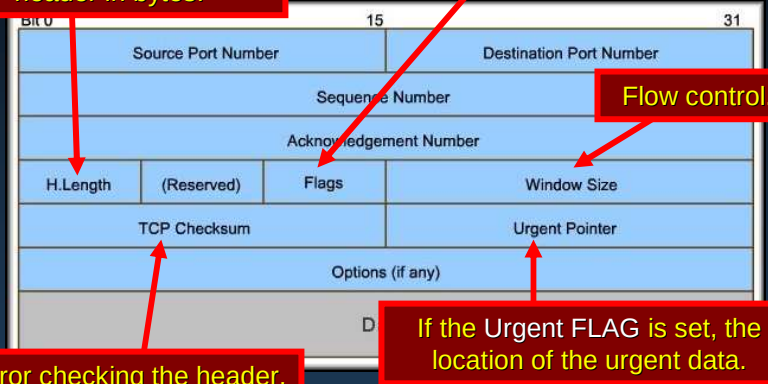
The number of the last octet (byte) in the segment.

The number of the next octet (byte) expected by the receiver.

Making Conversations Reliable

The length of the segment header in bytes.

Individual 1 bit fields used in session management.

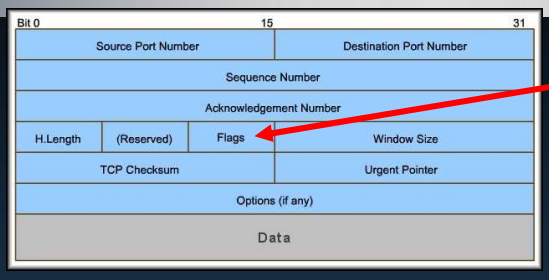


Flow control.

Error checking the header.

If the Urgent FLAG is set, the location of the urgent data.

Making Conversations Reliable - FYI

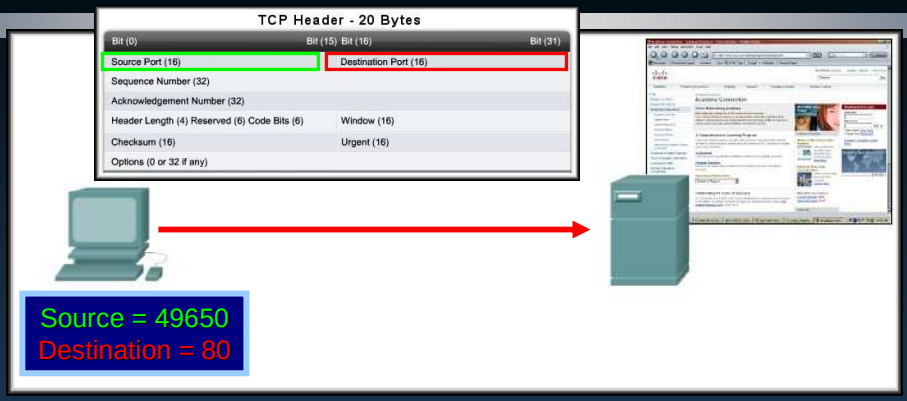


U A P R S F
R C S S Y I
G K H T N N

6 Bits
0 = OFF
1 = ON

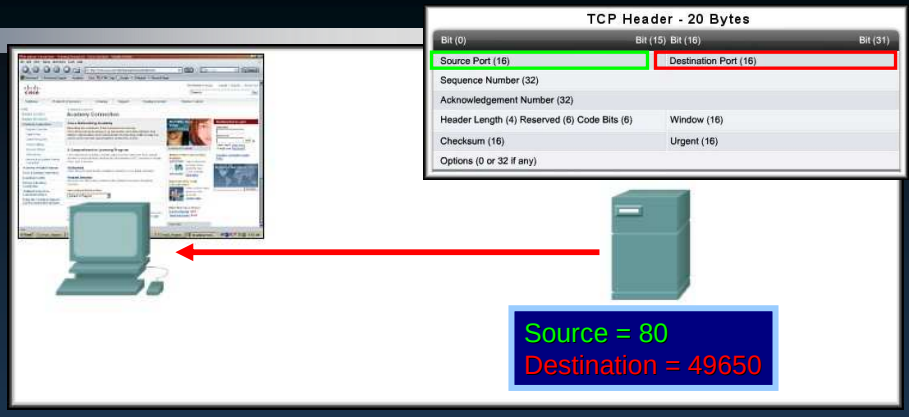
- URG – the Urgent Pointer Field is significant.
- ACK – the Acknowledgement Number field is significant
- PSH – push function
- RST – reset connection
- SYN – synchronize sequence numbers
- FIN – no more data from sender

TCP Server Processes



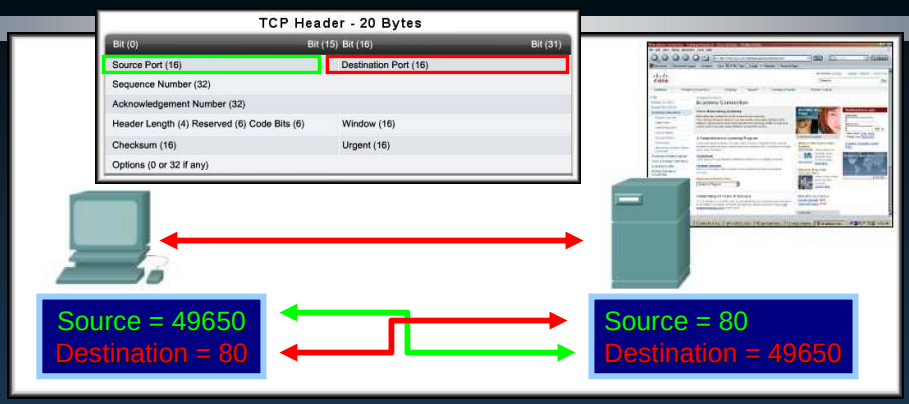
- Server is listening on Port 80 for HTTP connections.
- The client sets the destination port to 80 and uses a dynamic port as its source.

TCP Server Processes



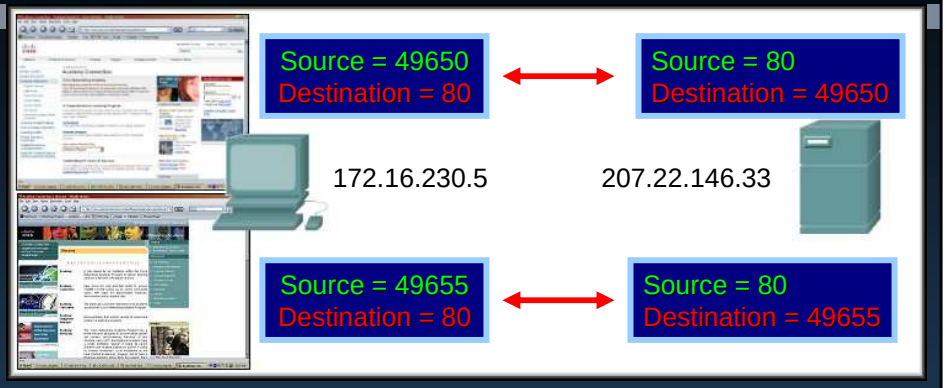
- Server replies with the web page.
  - Sets the source port to 80 and uses the client's source port as the destination.

TCP Server Processes



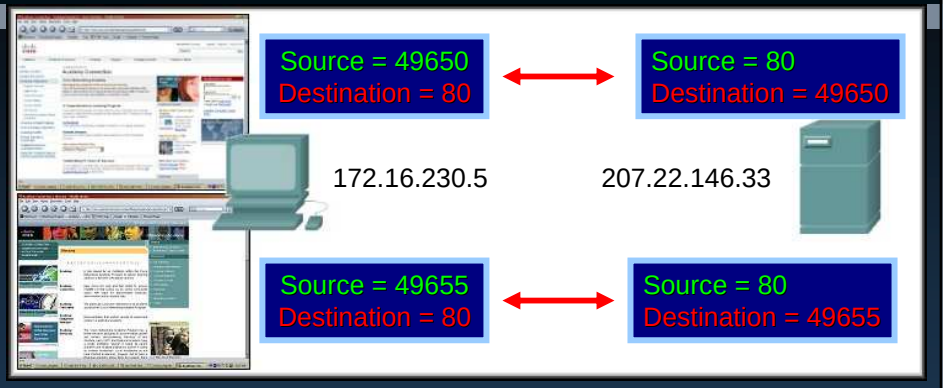
- Notice how the source and destination ports are used.

TCP Server Processes



- What if there are two sessions to the same server?
  - The client uses another dynamic port as its source and the destination is still port 80.
  - Different source ports keep the sessions unique on the server.

TCP Server Processes

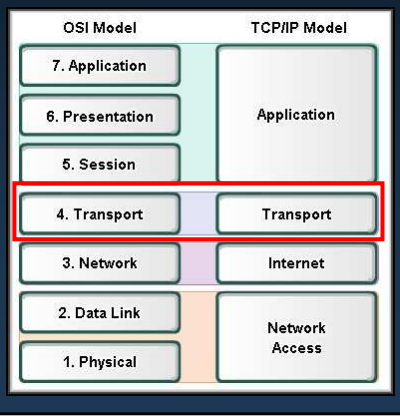


- How does the Transport Layer keep them separate?
  - The socket (IP Address:Port)

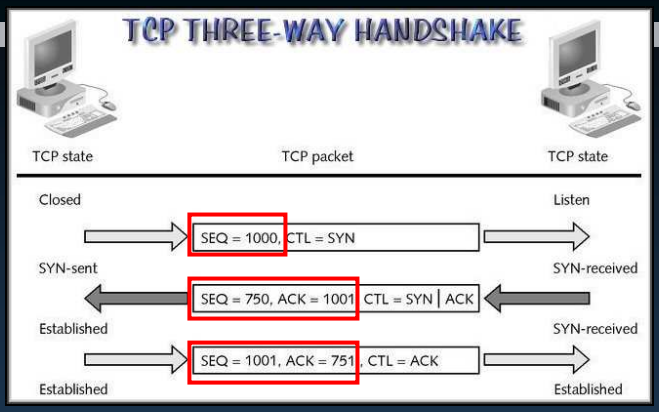
172.16.230.5:49650 ⟷ 207.22.146.33:49650
172.16.230.5:49655 ⟷ 207.22.146.33:49655

OSI Transport Layer

TCP
Connection Establishment and Termination

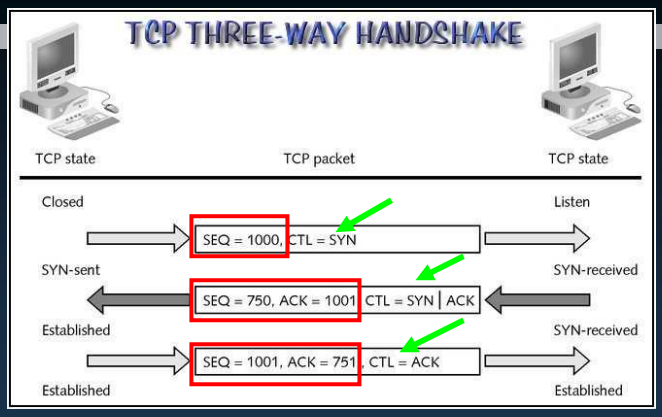


TCP Three-Way Handshake



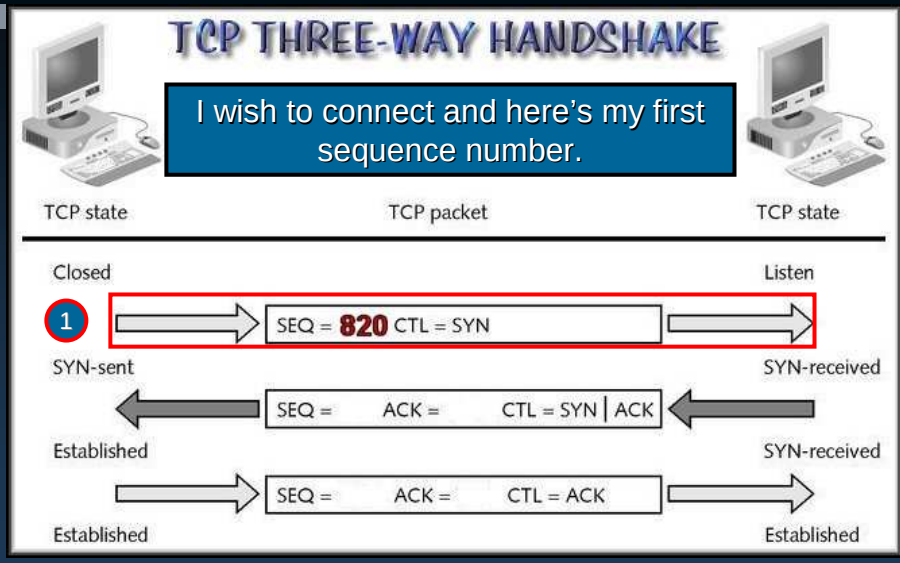
- For a connection to be established, the two end stations must **synchronize** on each other's initial sequence numbers **(ISNs)**.
- The ISN is the **starting sequence number** used when a TCP connection is established.

TCP Three-Way Handshake

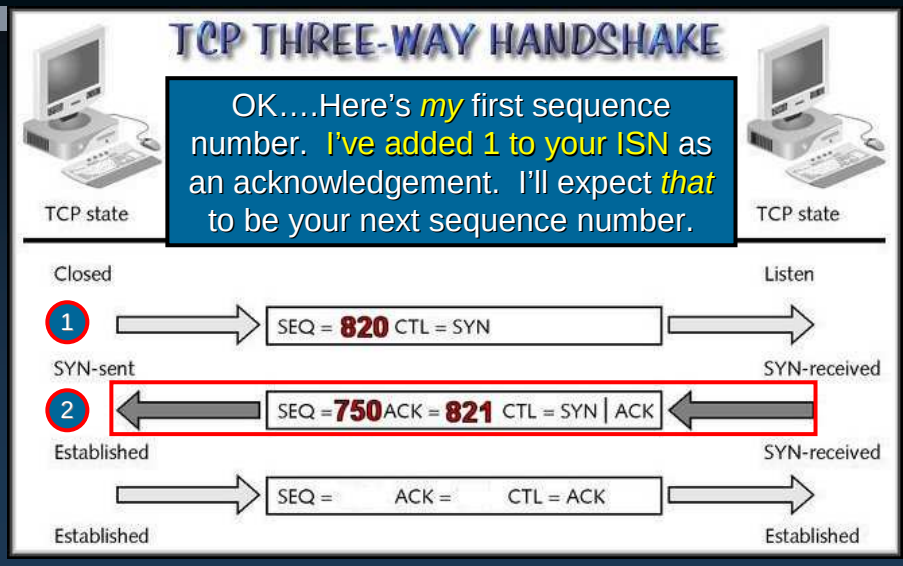


- **Sequence numbers** are used to track the order of segments and to ensure that no segments are lost in transmission.
- The **Flag fields** are used to identify the type of segment.

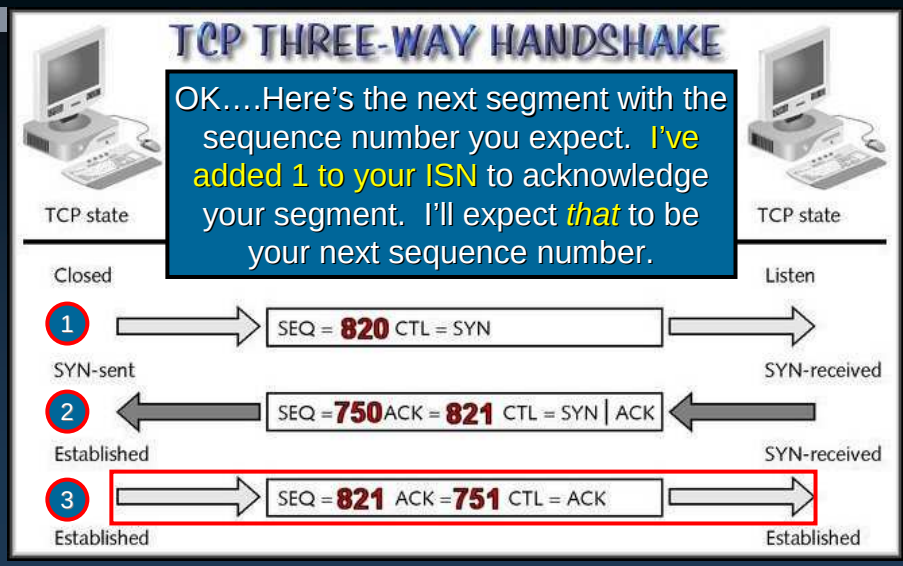
TCP Three-Way Handshake



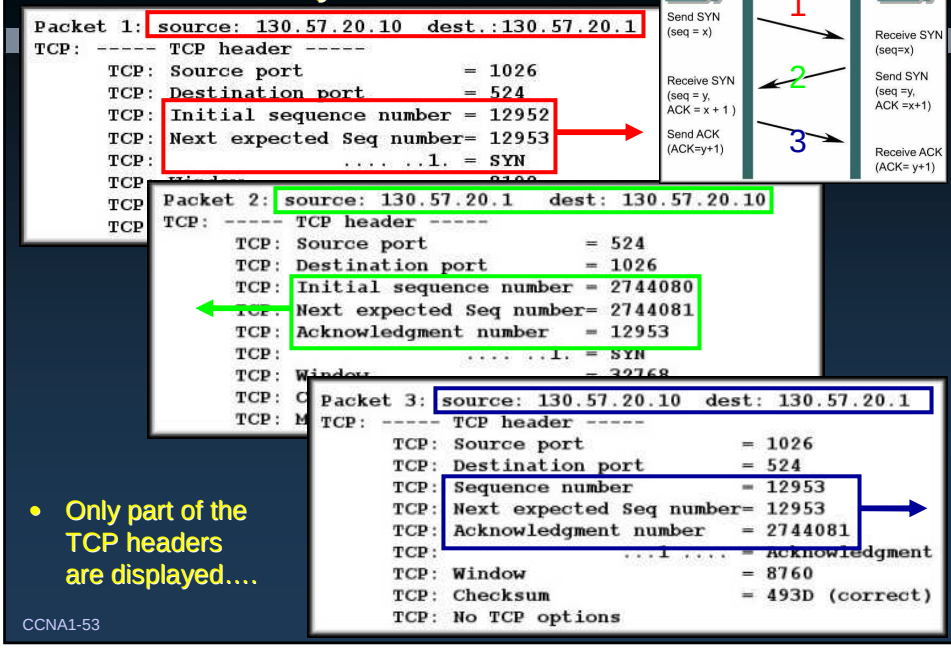
TCP Three-Way Handshake



TCP Three-Way Handshake

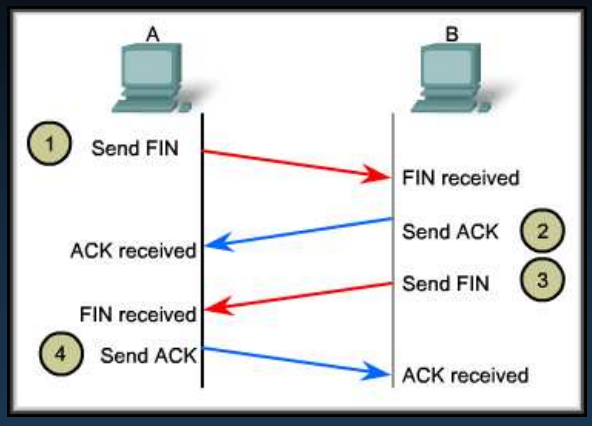


TCP Three-Way Handshake



- Only part of the TCP headers are displayed….

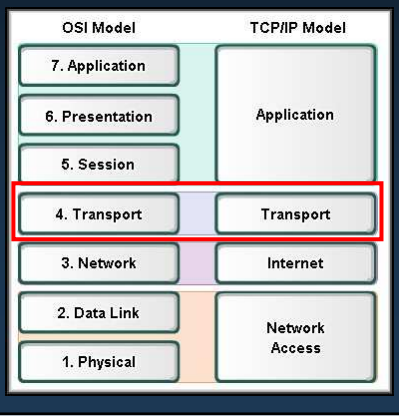
TCP Session Termination



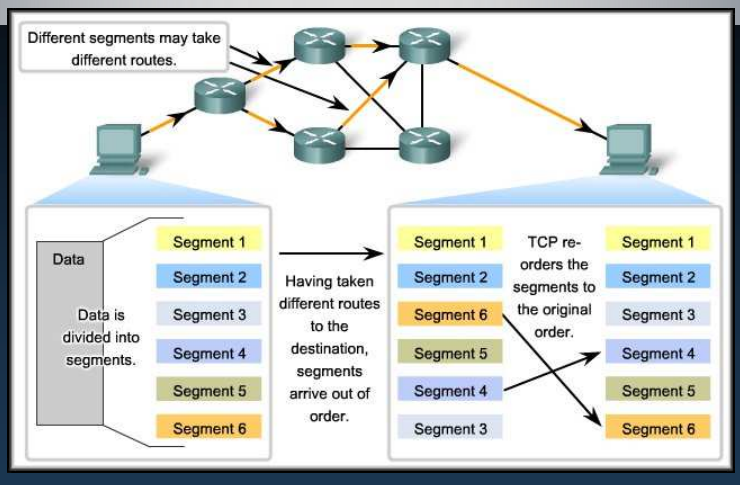
- Four step process using the Flag and sequence number fields.

OSI Transport Layer

TCP
Acknowledgements and Windowing

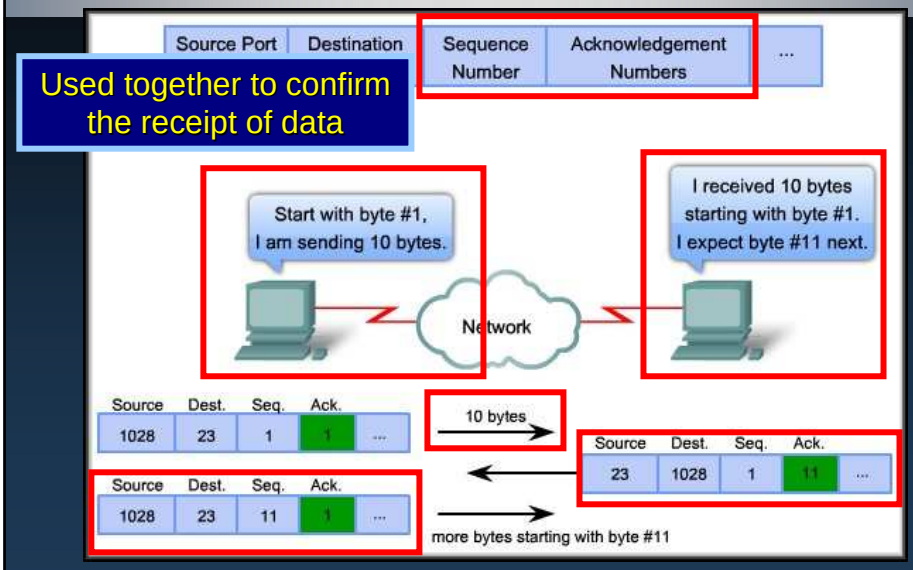


TCP Acknowledgements and Windowing

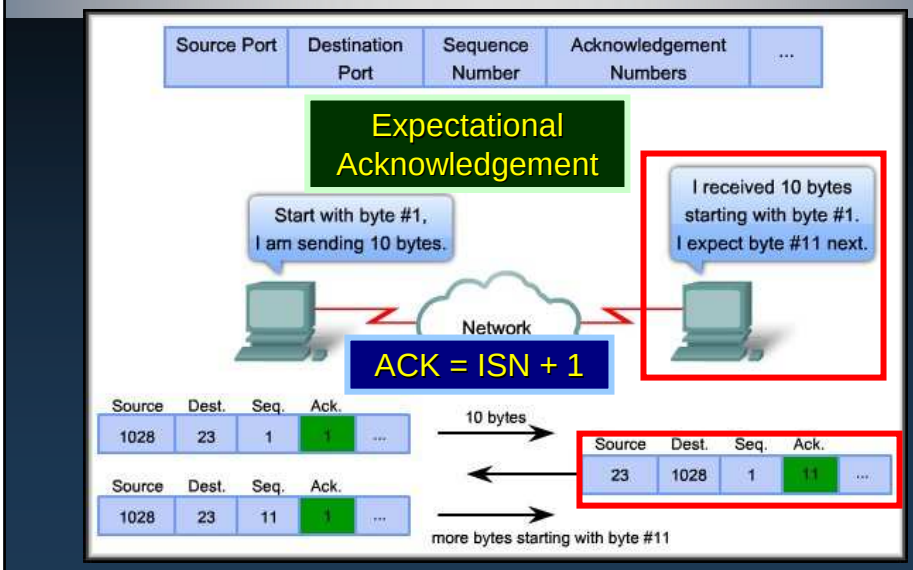


- Remember that the Transport Layer must reassemble the segments in the correct order.

TCP Acknowledgements and Windowing



TCP Acknowledgements and Windowing



TCP Acknowledgements and Windowing

- With a window size of 10, each segment carries only ten bytes of data and must be acknowledged before another segment is transmitted.

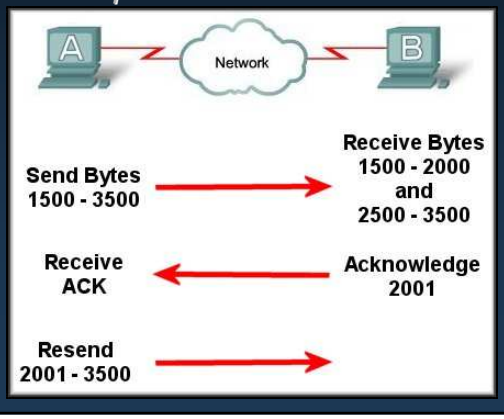
TCP Header - 20 Bytes

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Sequence Number (32)		
Acknowledgement Number (32)		
Header Length (4) Reserved (6) Code Bits (6)	Window (16)	
Checksum (16)	Urgent (16)	
Options (0 or 32 if any)		

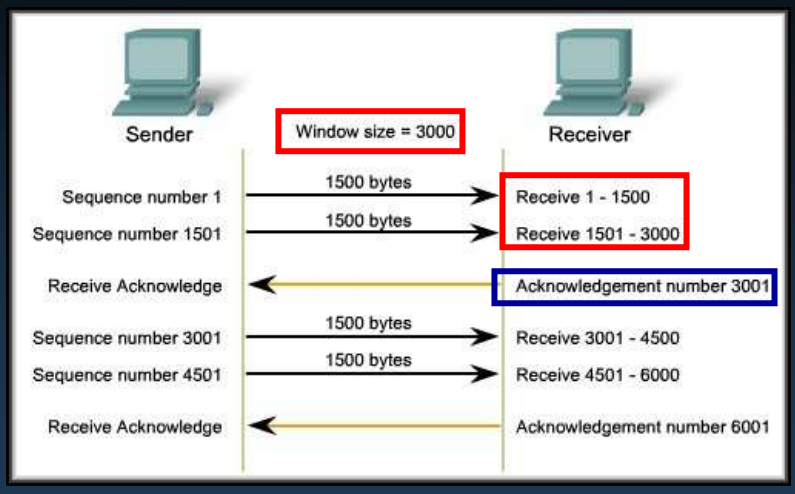
- Window Size:
  - The amount of data that can be sent before requiring an acknowledgement.
  - Determined by the Window field in the header.

TCP Retransmission

- A destination host service using TCP usually only acknowledges data for contiguous sequence bytes.
- If one or more segments are missing, *only the data in the segments that complete the stream are acknowledged*.



TCP Flow Control



TCP Flow Control

```
Packet 1: source: 130.57.20.10  dest.:130.57.20.1
TCP: ----- TCP header -----
     TCP: Source port              = 1026
     TCP: Destination port         = 524
     TCP: Initial sequence number = 12952
     TCP: Next expected Seq number= 12953
     TCP:                     ....  ..1. = SYN
     TCP: Window                   = 8192
     TCP: Checksum                 = 1303 (correct)
     TCP:
```

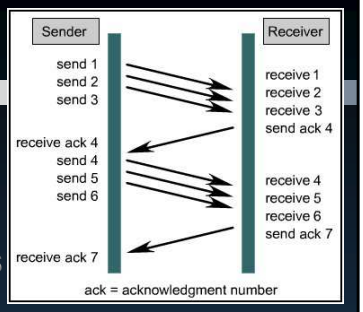
```
Packet 2: source: 130.57.20.1   dest: 130.57.20.10
TCP: ----- TCP header -----
     TCP: Source port              = 524
     TCP: Destination port         = 1026
     TCP: Initial sequence number = 2744080
     TCP: Next expected Seq number= 2744081
     TCP: Acknowledgment number    = 12953
     TCP:                     ....  ..1. = SYN
     TCP: Window                   = 32768
     TCP: Checksu                  = B3B7 (correct)
     TCP: Maximum
```

```
Packet 3: source: 130.57.20.10  dest: 130.57.20.1
TCP: ----- TCP header -----
     TCP: Source port              = 1026
     TCP: Destination port         = 524
     TCP: Sequence number          = 12953
     TCP: Next expected Seq number= 12953
     TCP: Acknowledgment number    = 2744081
     TCP:                     ....  ..1. = Acknowledgment
     TCP: Window                   = 8760
     TCP: Checksum                 = 493D (correct)
     TCP: No TCP options
```

Amount of data that can be sent before an acknowledgement.

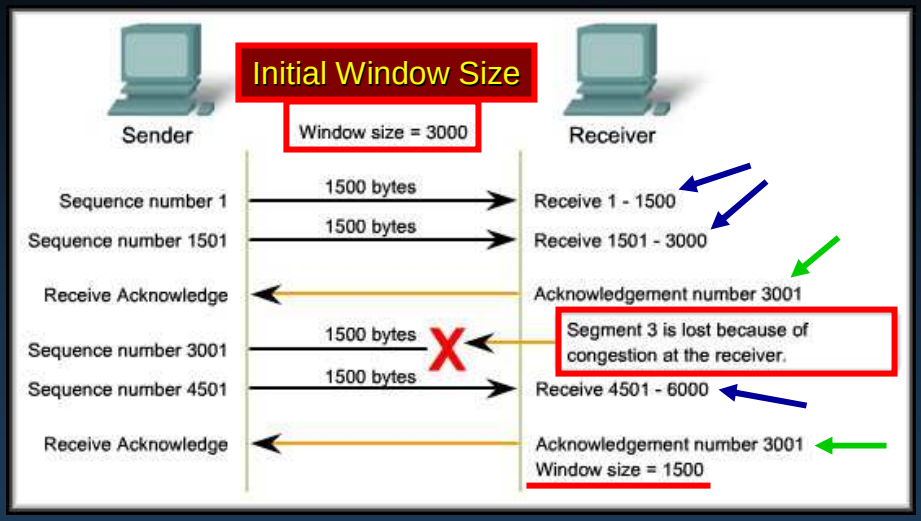
The initial window size is determined during the three-way handshake.

TCP Dynamic Window Sizes



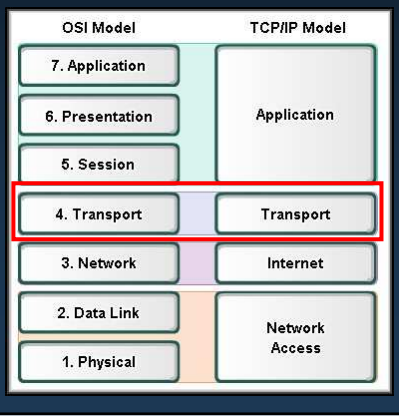
- TCP Full-duplex Service:
  - Means data can be flowing in each direction, simultaneously.
  - Window sizes, sequence numbers and acknowledgment numbers are independent of each other's data flow.
- The receiver sends an acceptable window size to the sender during each segment transmission.
  - If too much is data being sent, the acceptable window size is reduced.
  - if more data can be handled, the acceptable window size is increased.
- This is known as a *Stop-and-Wait* windowing protocol.

Dynamic Window Sizes

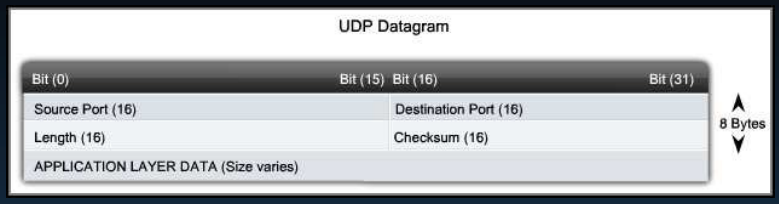


OSI Transport Layer

UDP
Communicating with Low Overhead



User Datagram Protocol (UDP)



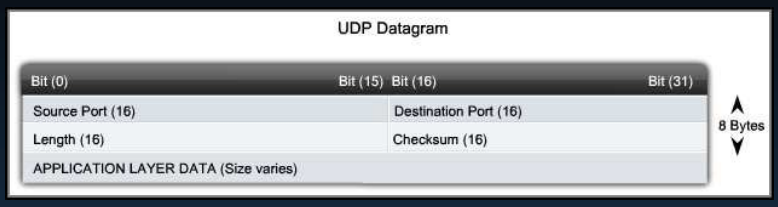
- Connectionless
- "Best Effort" delivery
- Low overhead

NO THREE-WAY HANDSHAKE

Example Applications

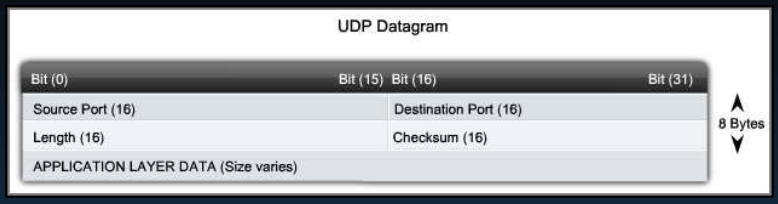
Domain Name System (DNS)
Online Games
Voice over IP (VoIP)
Dynamic Host Configuration Protocol (DHCP)
Trivial File Transfer Protocol (TFTP)

User Datagram Protocol (UDP)



- Low Overhead:
  - Connectionless:
    - No connection establishment as with TCP.
  - Unreliable or "Best Effort" delivery:
    - No error checking
    - No flow control
    - No congestion control
    - No sequence numbers for ordered delivery

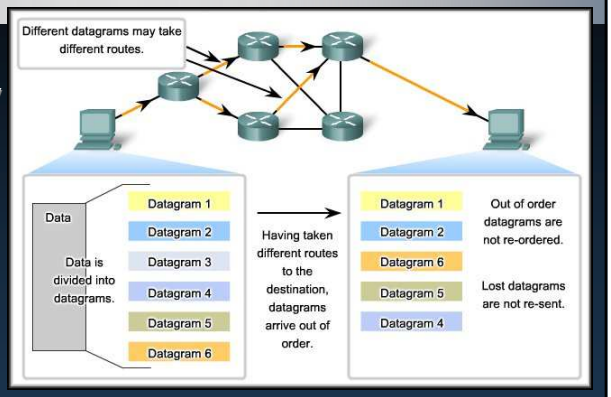
User Datagram Protocol (UDP)



- UDP is said to be transaction-based.
  - When an application has data to send, it simply sends the data.
- The UDP protocol data unit (PDU) is referred to as a datagram, although the terms *segment and datagram are sometimes used interchangeably* to describe a Transport layer PDU.

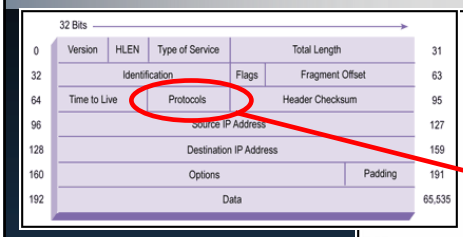
User Datagram Protocol (UDP)

- If datagrams take multiple paths, they will sometimes arrive in the wrong order.
- UDP does not sequence the datagrams as TCP does nor are there any acknowledgements.

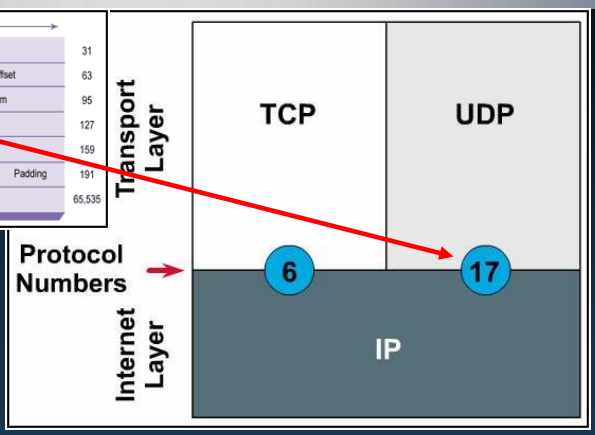


- Re-sequencing datagrams and handling missing data is up to the application.

User Datagram Protocol (UDP)

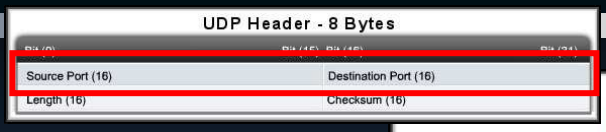


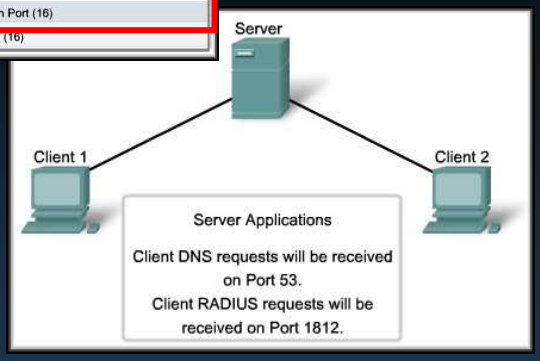
IP Packet Header



- When a packet is encapsulated at the Network Layer, it is coded to identify the source of the packet (TCP *or* UDP).

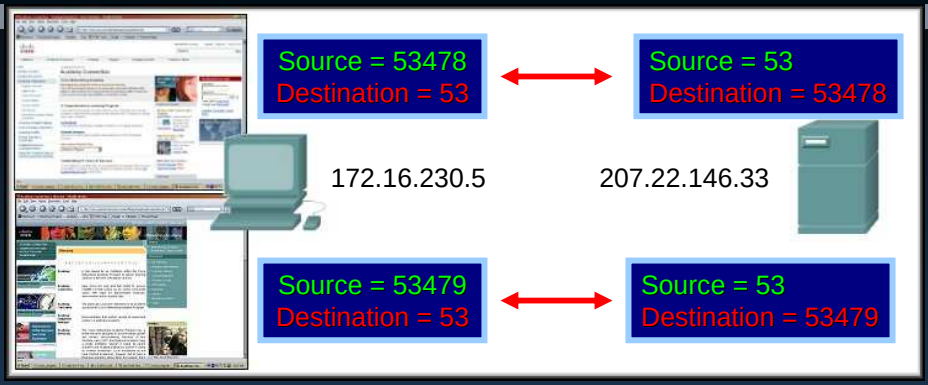
UDP Server Processes





- Like TCP, UDP applications use Well-Known or Registered ports specifically set aside for UDP.
- When UDP receives a datagram destined for one of these applications, it forwards the datagram based on its port number.

UDP Client Processes



- Uses the same methods as TCP for keeping multiple conversations separate (Socket - IP Address:Port).

172.16.230.5:53478 ⟷ 207.22.146.33:53478
172.16.230.5:53479 ⟷ 207.22.146.33:53479

Summary: Port Numbers

Port Number	Application	Layer 4 Protocol	Description
20	FTP	TCP	File Transfer Protocol – Data
21	FTP	TCP	File Transfer Protocol – Control Commands
23	TELNET	TCP	Terminal connection
25	SMTP	TCP	Simple Mail Transfer Protocol - Email
53	DNS	UDP	Domain Name System
67,68	DHCP	UDP	Dynamic Host Configuration Protocol
69	TFTP	UDP	Trivial File Transfer Protocol
80	HTTP	TCP	Hypertext Transfer Protocol

Summary: TCP vs UDP

- Transmission Control Protocol (TCP):
  - Connection-oriented
  - Reliable end-to-end delivery of messages
  - Error detection and recovery
  - Flow control
- User Datagram Protocol (UDP):
  - Connectionless
  - Best-effort datagram delivery
  - Applications that do not require full TCP services

Summary: Applications

- TCP:
  - File Transfer Protocol (FTP)
  - Telnet
  - Simple Mail Transfer Protocol (SMTP)
  - Post Office Protocol (POP3)
  - Hypertext Transfer Protocol (HTTP)
- UDP:
  - Trivial File Transfer Protocol (TFTP)
  - Domain Name System (DNS)
  - Simple Network Management Protocol (SNMP)
  - Dynamic Host Configuration Protocol (DHCP)

Am I Drivin' too fast for 'ya?

Time to do some STUFF!