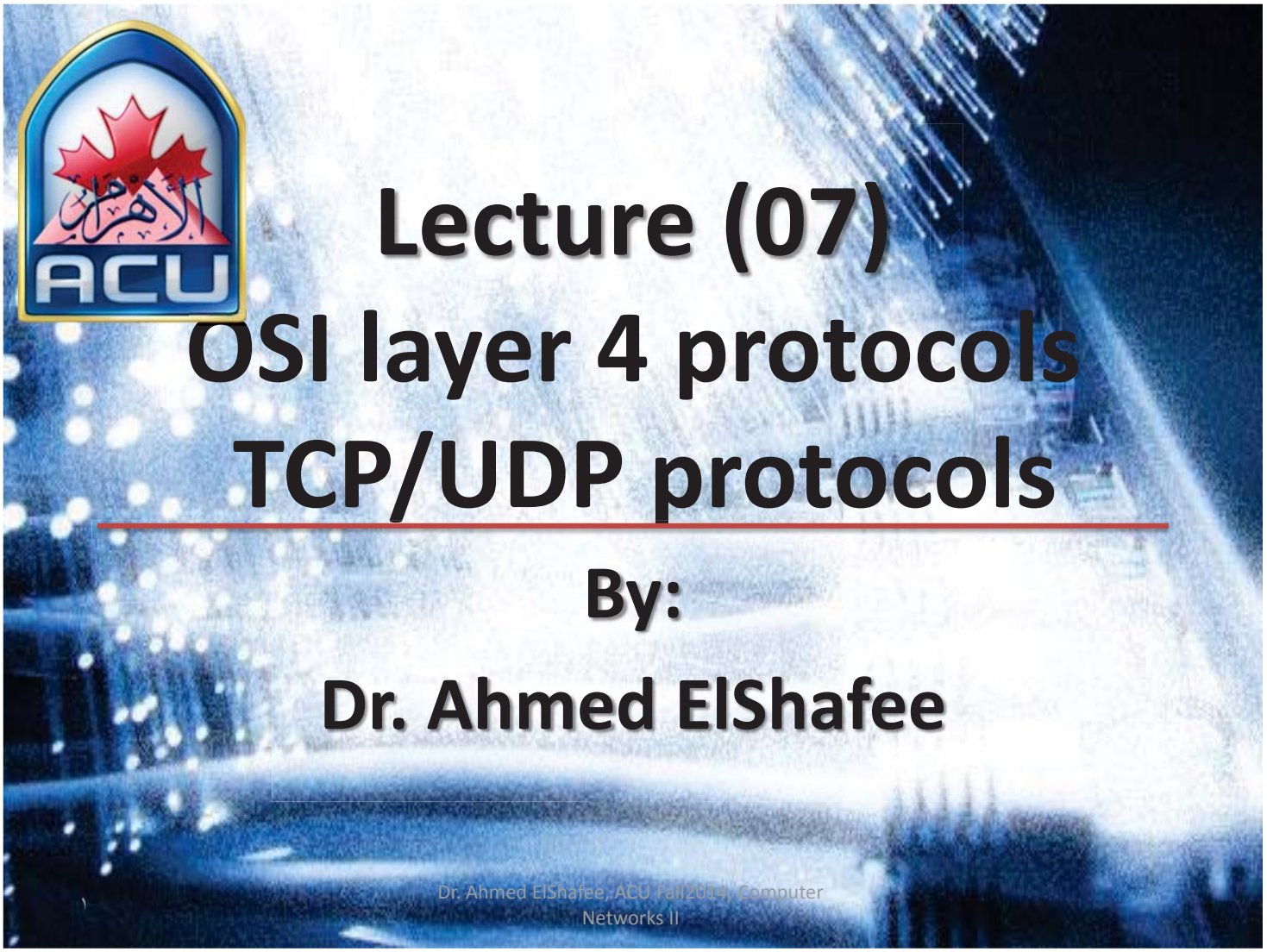


Lecture (07)
OSI layer 4 protocols
TCP/UDP protocols

By:

Dr. Ahmed ElShafee

Agenda

- Introduction
- Typical Features of OSI Layer 4
- Connectionless and Connection-Oriented Protocols
- OSI Layer 4 Common feature: Multiplexing Using Port Numbers
- The Transmission Control Protocol (TCP)
- TCP main feature
- The User Datagram Protocol

Introduction

- Most data-link protocols notice errors then discard frames that have errors
- The OSI transport layer (layer 4) might provide for retransmission (error recovery) and help to avoid congestion (flow control)
- These functions depend on the particular protocol used in OSI transport layer (layer 4)
- There are 2 main protocols
  - TCP (Transmission Control Protocol)
  - UDP (User Datagram Protocol)

Typical Features of OSI Layer 4

Feature	Explanation
Connection-oriented or connectionless	Defines whether the protocol establishes some correlation between two endpoints before any user data is allowed to be transferred (connection oriented), or not (connectionless).
Error recovery	The process of noticing errored or lost segments and causing them to be resent.
Reliability	Another term for error recovery.
Flow control	Processes that control the rates at which data is transferred between two endpoints.
Segmenting application data	Application layer protocols may need to send large chunks of data—much larger than can fit inside one IP packet. The transport layer is responsible for segmenting the larger data into pieces, called segments, that can fit inside a packet.

Connectionless and Connection-Oriented Protocols

- **Connection-oriented protocol—A protocol either that requires an exchange of messages** before data transfer begins or that has a required pre-established correlation between two endpoints
- **Connectionless protocol—A protocol that does not require an exchange of messages and** that does not require a pre-established correlation between two endpoints

Connectionless and Connection-Oriented Protocols (2)

- TCP & IPX are connection oriented protocols, as they exchange few message between communicated parties to establish connection before sending data.
- permanent virtual circuits (PVCs), Frame Relay does not require any messages to be sent ahead of time, but it does require predefinition in the Frame Relay switches, establishing a connection between two Frame Relay–attached devices.
- ATM PVCs are also connection oriented, for similar reasons.

Connectionless and Connection-Oriented Protocols (3)

- Many people confuse the real meaning of *connection-oriented with the definition of a* reliable, or error-recovering,
- If a Protocol is a connection-oriented does not mean that it also performs error recovery

Protocol Characteristics: Recovery and Connections

Connected?	Reliable?	Examples
Connection-oriented	Yes	LLC Type 2 (802.2), TCP, Novell SPX
Connection-oriented	No	Frame Relay VCs, ATM VCs, PPP
Connectionless	Yes	TFTP, NetWare NCP (no Packet Burst)
Connectionless	No	UDP, IP, most Layer 3 protocols

OSI Layer 4 Common feature: Multiplexing Using Port Numbers

- TCP and UDP both use a concept called *multiplexing.*
- computer might be running many applications, such as a web browser, an e-mail package, or an FTP client.
- TCP and UDP multiplexing enables the receiving computer to know which application to give the data to.

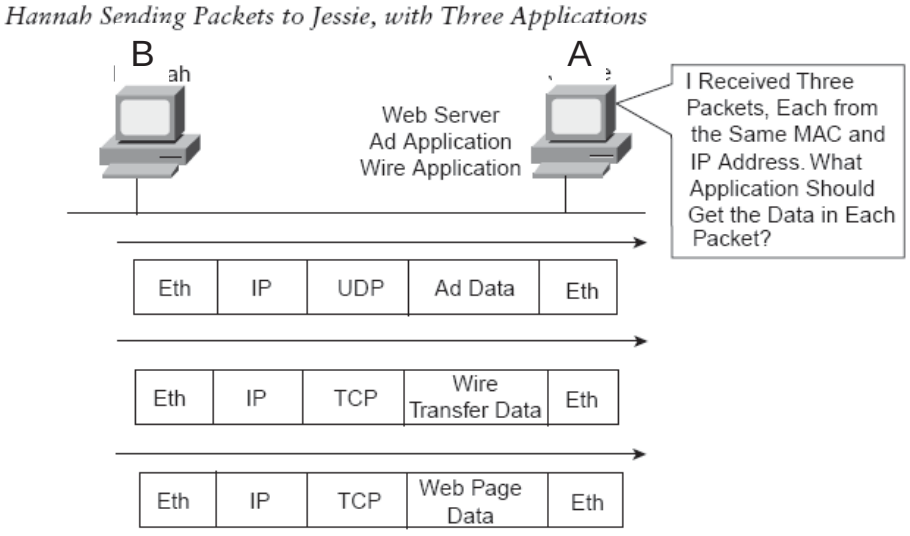
OSI Layer 4 Common feature: Multiplexing Using Port Numbers (2)

Example

- sample network consists of two PCs, labeled "A" and "B".
- "A" uses an application that wrote to send advertisements that display on "B"'s screen.
- The application sends a new ad to "B" every 10 seconds.
- "A" uses a second application, a wire-transfer application, to send "B" some money.
- Finally, "A" uses a web browser to access the web server that runs on "B"'s PC.

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (3)

- Now "B" runs three applications:
  - A UDP-based ad application
  - A TCP-based wire-transfer application
  - A TCP web server application

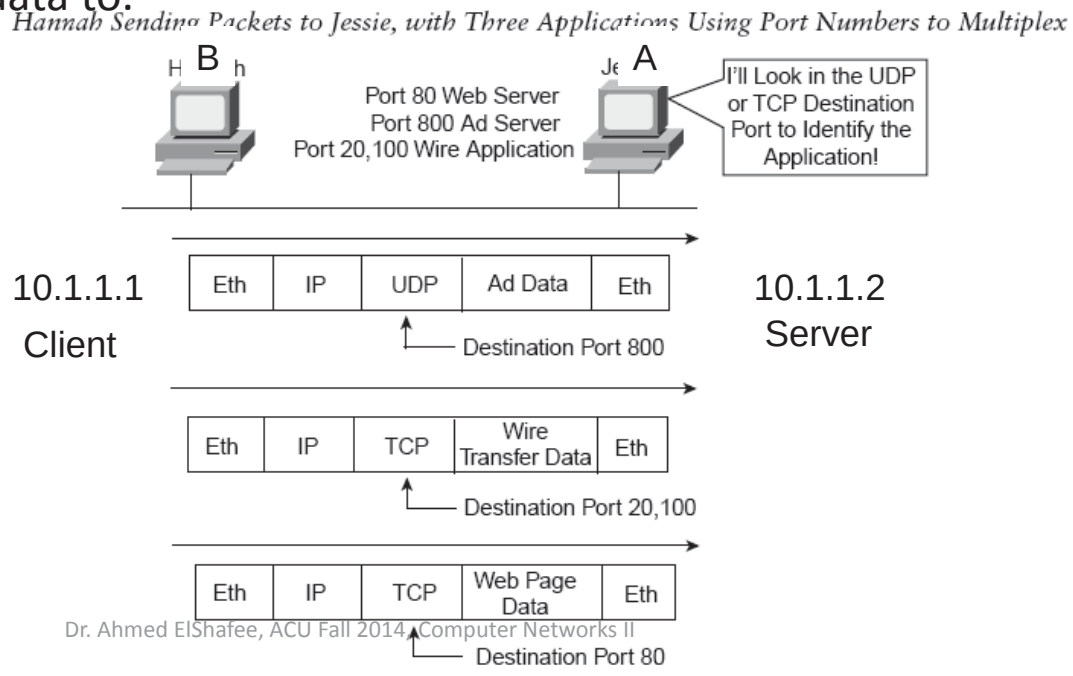


OSI Layer 4 Common feature: Multiplexing Using Port Numbers (4)

- "B" needs to know which application to give the data to, but all three packets are from the same Ethernet and IP address.
- You might think that "B" could look at whether the packet contains a UDP or a TCP header, but, as you see in the figure, two applications (wire transfer and web) both are using TCP.
- TCP and UDP solve this problem by using a port number field in the TCP or UDP header, respectively.

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (5)

- Each of "A"'s TCP and UDP segments uses a different destination port number so that "B" knows which application to give the data to.



OSI Layer 4 Common feature: Multiplexing Using Port Numbers (6)

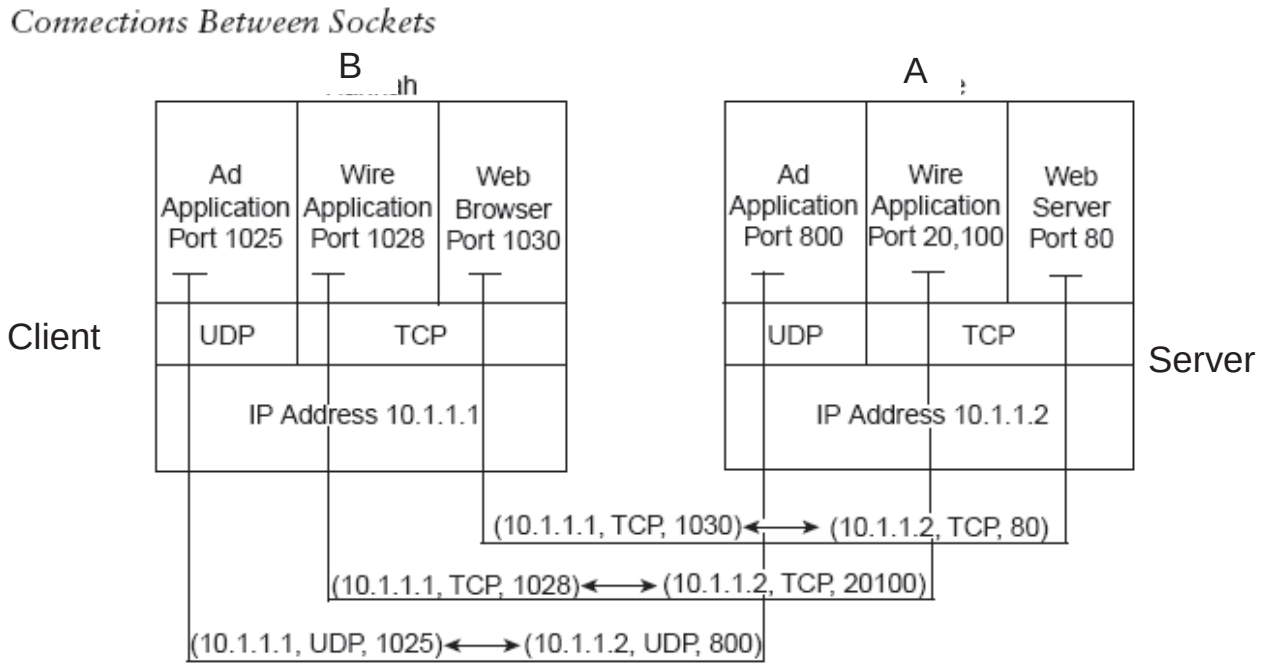
- Multiplexing relies on the use of a concept called a *socket.*

A socket consists of three things:

  - an IP address,
  - a transport protocol, and
  - a port number.

- Web server application uses socket (10.1.1.2, TCP, port 80)
- Web browser connected to the web server, uses a socket (10.1.1.1, TCP, 1030).

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (7)



OSI Layer 4 Common feature: Multiplexing Using Port Numbers (8)

- Port numbers are a vital part of the socket concept. Well-known port numbers are used by servers; other port numbers are used by clients.
- Hosts typically allocate dynamic port numbers starting at 1024 because the ports below 1024 are reserved for well-known applications,
- connection requests from clients are required to include both the source and the destination port numbers, the port numbers used by the servers must be well known.
- Therefore, each server has a hard-coded, well-known port number
- On client machines, where the requests originate, any unused port number can be allocated.

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (9)

Popular Applications and Their Well-Known Port Numbers

Port Number	Protocol	Application
20	TCP	FTP data
21	TCP	FTP control
23	TCP	Telnet
25	TCP	SMTP
53	UDP, TCP	DNS
67, 68	UDP	DHCP
69	UDP	TFTP
80	TCP	HTTP (WWW)
110	TCP	POP3
161	UDP	SNMP

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (10)

Popular Applications

1. The World Wide Web (WWW) application uses web browsers to surf internet.

   WWW may be used to manage a router or switch by enabling a web server function in the router or switch, and using a browser to access the router or switch.

2. Domain Name System (DNS) allows users to use names to refer to computers,
3. Simple Network Management Protocol (SNMP) is an application layer protocol used specifically for network device management.

OSI Layer 4 Common feature: Multiplexing Using Port Numbers (11)

4. Trivial File Transfer Protocol (TFTP) defines a protocol for basic file transfer, used by routers and switched to upload configuration.
5. Simple Mail Transport Protocol (SMTP)
6. Post Office Protocol version 3 (POP3), both used for transferring mail,

The Transmission Control Protocol (TCP)

- TCP provides a variety of useful features, including error recovery
  - Multiplexing using port numbers
  - Error recovery (reliability)
  - Flow control using windowing
  - Connection establishment and termination
  - End-to-end ordered data transfer
  - Segmentation

The Transmission Control Protocol (TCP) (2)

- TCP accomplishes these functions through mechanisms at the endpoint computers
- Regardless of whether two computers are on the same Ethernet, or are separated by the entire Internet, TCP performs its functions the same way.

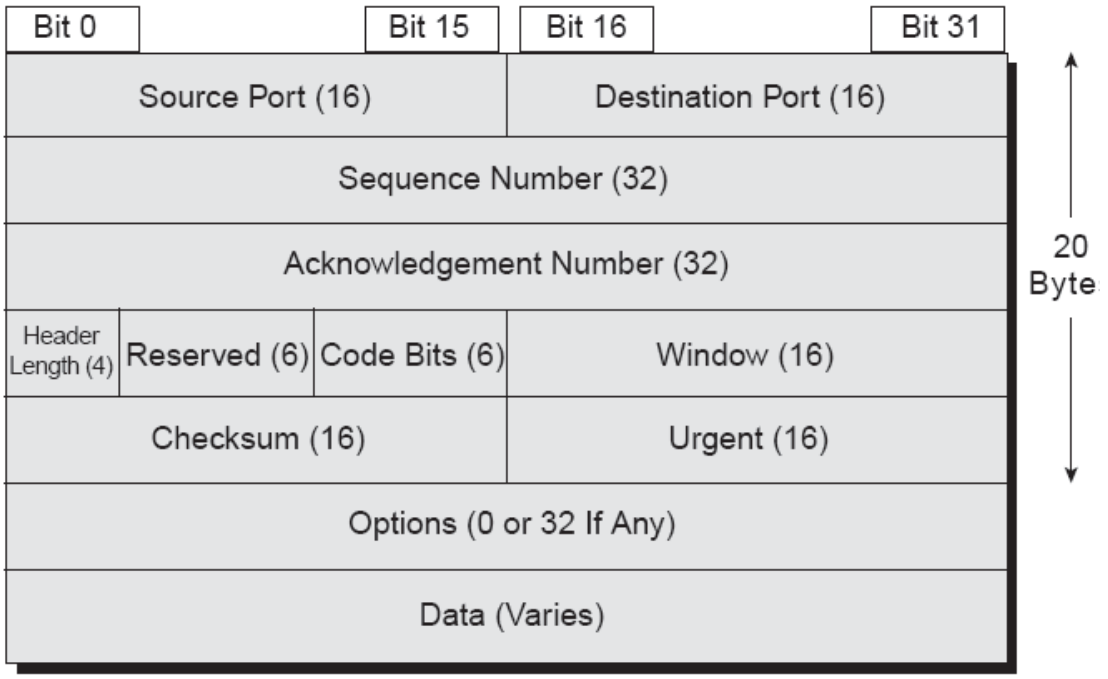
The Transmission Control Protocol (TCP) (3)

TCP & IP interworking

- TCP relies on IP for end-to-end delivery of the data, including routing issues.
- TCP performs only part of the functions necessary to deliver the data between applications

The Transmission Control Protocol (TCP) (4)

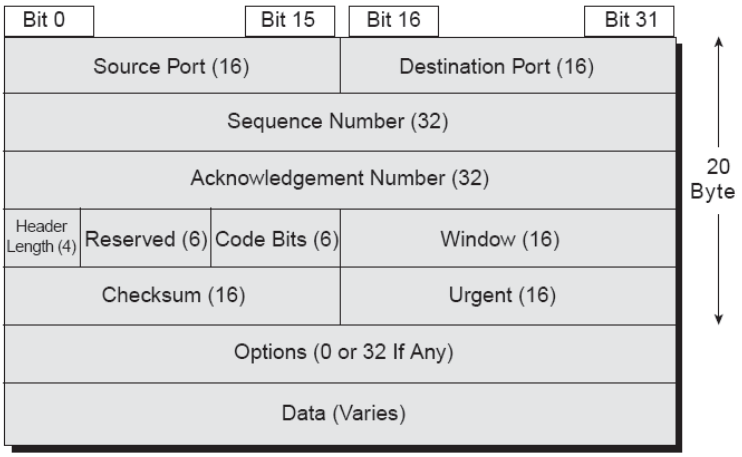
TCP Header Fields



TCP main feature

1. Error Recovery (Reliability)

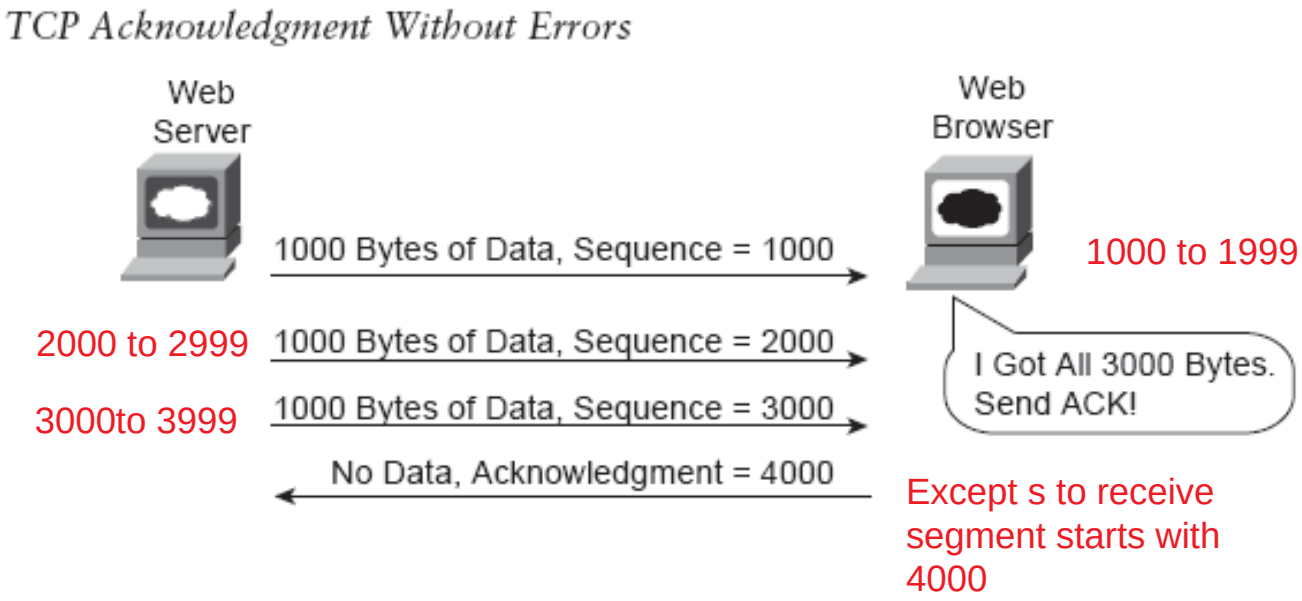
- To accomplish reliability, TCP numbers data bytes using the Sequence and Acknowledgment fields in the TCP header.
- TCP achieves reliability in both directions, using the Sequence Number field of one direction combined with the Acknowledgment field in the opposite direction.



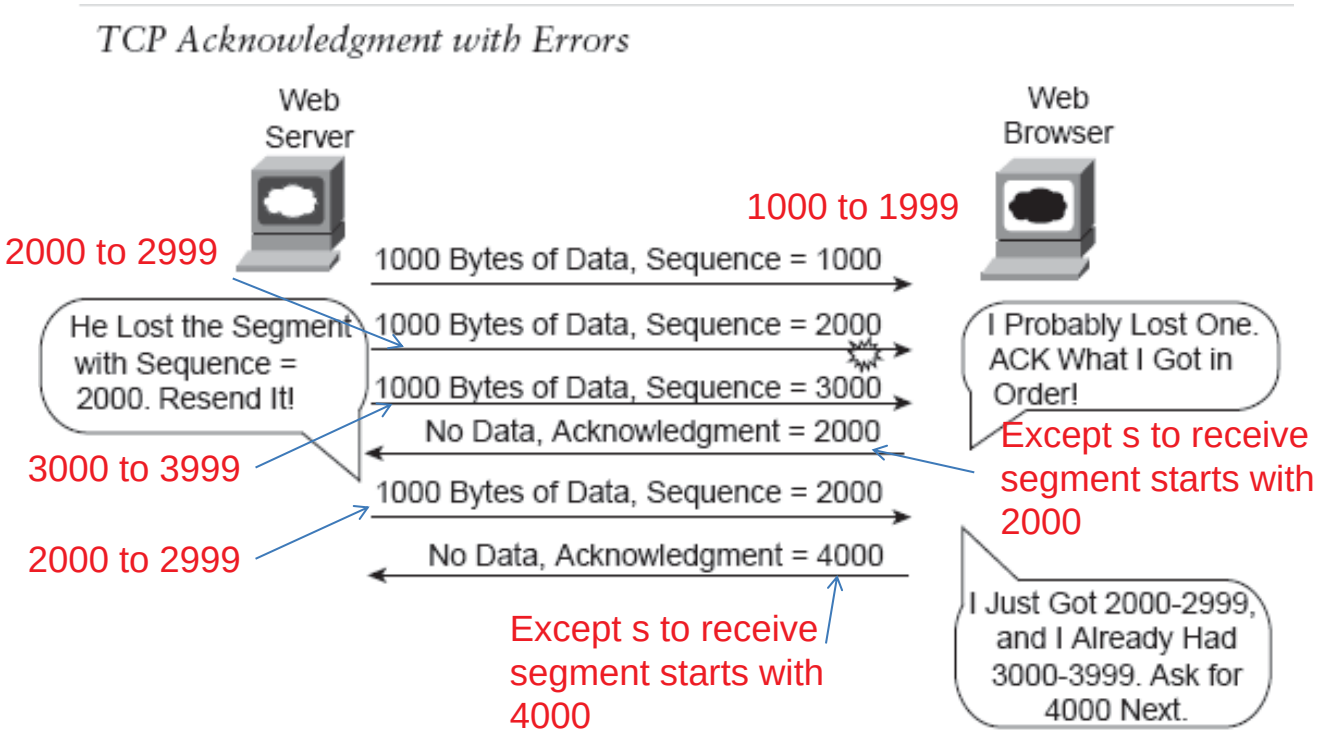
TCP main feature (2)

- The sequence number contains the number of the first byte in the segment. (by sender)
- Acknowledgment field contains the next byte to be received this is called *forward acknowledgment* (by receiver)

TCP main feature (3)



TCP main feature (4)



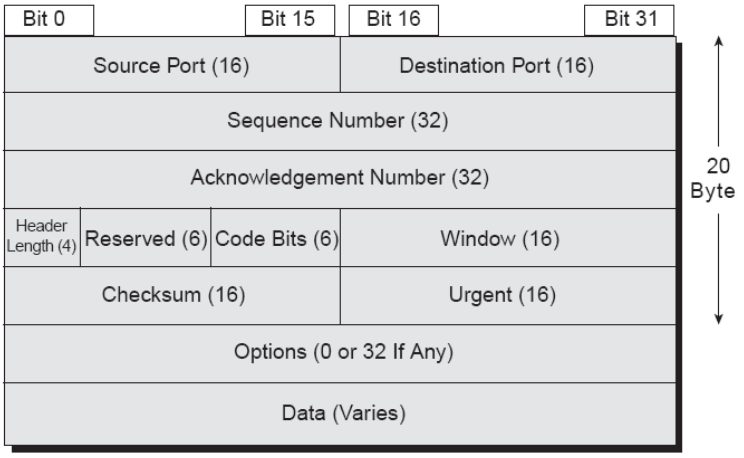
TCP main feature (5)

- (Although not shown, the sender also sets a re-transmission timer, awaiting acknowledgment, just in case the acknowledgment is lost, or in case all transmitted segments are lost.
- If that timer expires, the TCP sender sends all segments again.)

TCP main feature (6)

2. **Flow Control Using Windowing**

- Window field contains the maximum number of unacknowledged bytes allowed by receiver,
- When the window is full, (receiver didn't acknowledged the bytes sent by sender) the sender will stop sending any new packets, which controls the flow of data

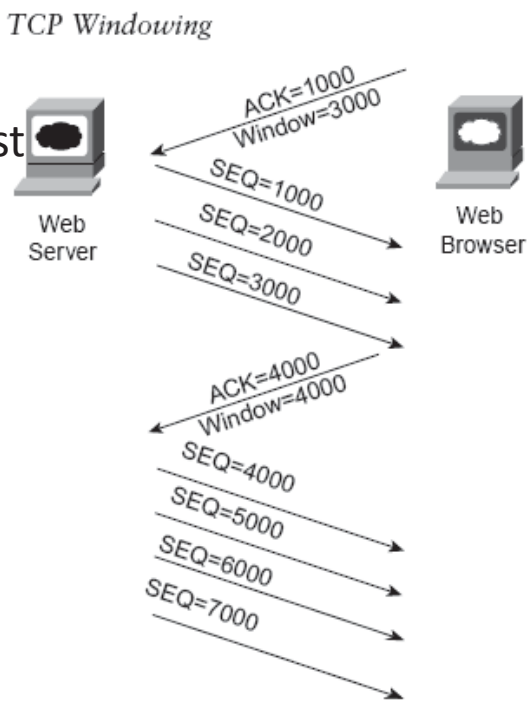


TCP main feature (7)

- Windowing does not require that the sender stop sending in all cases.
- If an acknowledgment is received before the window is exhausted, a new window begins and the sender continues to send data until the current window is exhausted.
- The window starts small and then grows until errors occur. The window then "slides" up and down based on network performance, so it is sometimes called a *sliding window.*

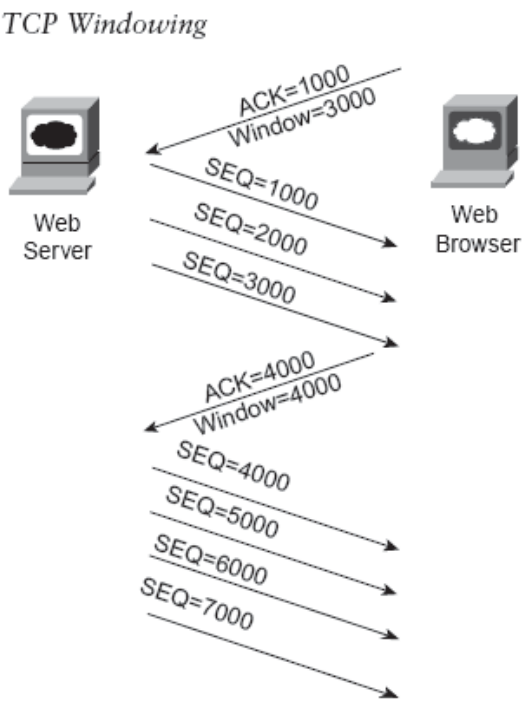
TCP main feature (8)

- Firstly web browser send Ack:=1000, and windows=3000
- Which means that the expected first byte in received segment contains byte number 1000,
- Sender are allowed to send 3000 bytes, and will not continue before receiving Ack=4000 from receiver.
- Receiver may send Ack=2000, or 3000 which means a new window will start from bytes



TCP main feature (9)

- After sender receive Ack=4000,
- Because there have been no errors, the web client grants a larger window to the server, so now 4000 bytes can be sent before an acknowledgment is received by the server.



TCP main feature (10)

In other words,

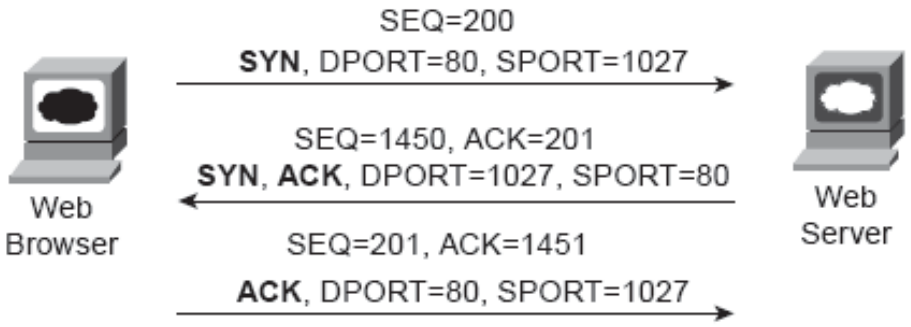
- the Window field is used by the receiver to tell the sender how much data it can send before it must stop and wait for the next acknowledgment.

TCP main feature (11)

3. **Connection Establishment and Termination**

- Connection establishment refers to the process of initializing sequence and acknowledgment fields and agreeing to the port numbers used.
- Which occurs before any of the other TCP features can begin their work.

TCP Connection Establishment



TCP main feature (12)

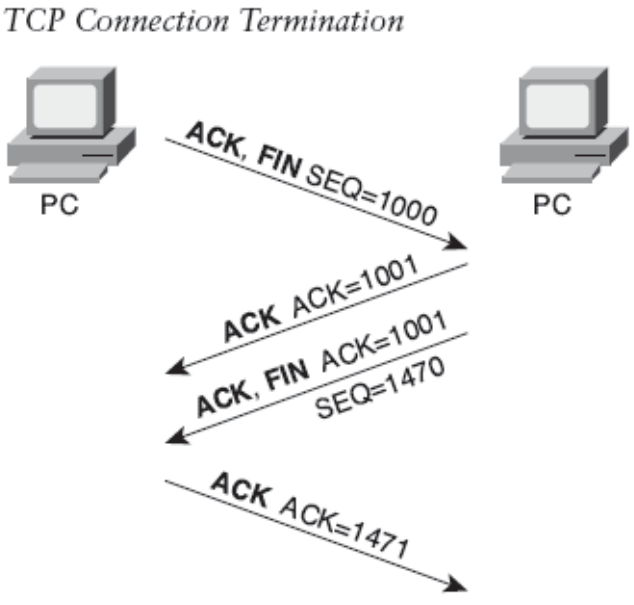
- This three-way connection-establishment flow must complete before data transfer can begin.
- The initialization contains a single byte of data
- TCP signals connection establishment using 2 bits inside the flag fields of the TCP header.
- Called the SYN and ACK flags
- SYN means “synchronize the sequence numbers,” which is one necessary component in initialization for TCP.
- The ACK field means “the acknowledgment field is valid in this header.”
- the initial TCP segment doesn’t contain acknowledgment number; this is because that number is not valid yet.

TCP main feature (13)

- Because the ACK field must be present in all the ensuing segments, the ACK bit continues to be set until the connection is terminated.
- TCP initializes the Sequence Number and Acknowledgment Number fields to any number that fits into the 4-byte fields;

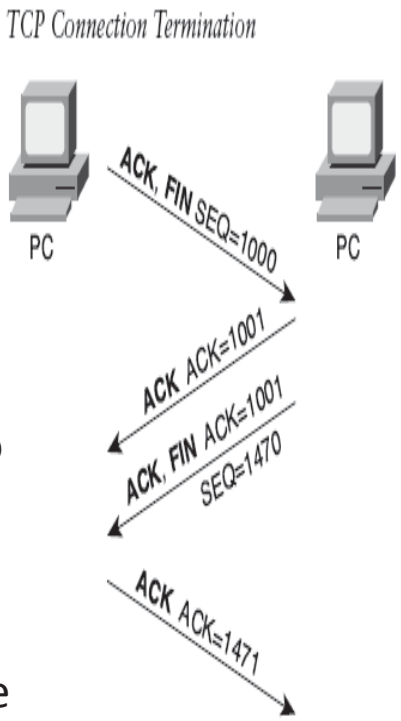
TCP main feature (14)

- TCP connection termination has a four-way termination sequence is straightforward and uses an additional flag, called the *FIN bit. (FIN is short for "finished,"*.)



TCP main feature (15)

- Before the device on the right sends the third TCP segment in the sequence, it notifies the application that the connection is coming down.
- It then waits on an acknowledgment from the application before sending the third segment in the figure.
- Just in case the application takes some time to reply, the PC on the right sends the second flow in the figure, acknowledging that the other PC wants to take down the connection.
- Otherwise, the PC on the left might resend the first segment over and over.



TCP main feature (16)

4. **Data Segmentation and Ordered Data Transfer**

- Applications need to send data.
- Sometimes the data is small –single byte- and in other cases might be millions of bytes –file transfer-.
- MTU refers to the size of the "data," according to the data link layer—
- For many data link protocols, Ethernet included, the MTU is 1500 bytes.
- because IP and TCP headers are 20 bytes each, TCP typically segments large data into 1460 byte (or smaller) segments.

TCP main feature (17)

- *The TCP sender segments* the data into smaller pieces, called *segments,* The TCP receiver performs re-ssembly when it receives the segments
- Usually IP routing can choose to balance traffic across multiple links, the actual segments may be delivered out of order.
- To solve that, the TCP receiver reorders segments that arrive out of sequence.
- How? If segments arrive with the sequence numbers 1000, 3000, and 2000, each with 1000 bytes of data, the receiver can reorder them and no retransmissions are required.

TCP main feature (18)

- That's why The TCP header, along with the data field, together are called a *TCP segment.*
- This term is similar to a data link frame and an IP packet,
- The term *L4PDU also can be used instead of* the term *TCP segment because TCP is a Layer 4 protocol.*

TCP main feature (19)

TCP Function Summary

Function	Description
Multiplexing	Function that allows receiving hosts to decide the correct application for which the data is destined, based on the port number
Error recovery (reliability)	Process of numbering and acknowledging data with Sequence and Acknowledgment header fields
Flow control using windowing	Process that uses window sizes to protect buffer space and routing devices
Connection establishment and termination	Process used to initialize port numbers and Sequence and Acknowledgment fields
Ordered data transfer and data segmentation	Continuous stream of bytes from upper-layer process that is "segmented" for transmission and delivered to upper-layer processes at the receiving device, with the bytes in the same order

The User Datagram Protocol

- UDP provides some functions of TCP, such as
  - data transfer,
  - segmentation, and
  - multiplexing using port numbers,
- it does so with fewer bytes of overhead and with less processing required.
- UDP data transfer differs from TCP data transfer in that
  - no reordering or
  - No recovery
- So that Applications that use UDP are tolerant of the lost data, or they have some application mechanism to recover lost data.

The User Datagram Protocol (2)

Examples:

- DNS requests use UDP because the user will retry an operation if the DNS resolution fails.
- The Network File System (NFS), a remote file system application, performs recovery with application layer code, so UDP features are acceptable to NFS.

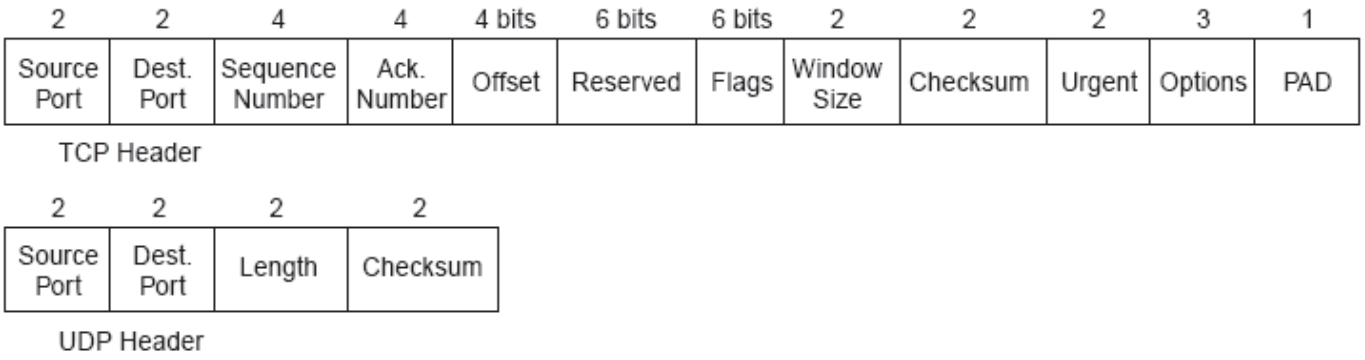
The User Datagram Protocol (3)

TCP and UDP Functional Comparison

Function	Description (TCP)	Description (UDP)
Ordered data transfer	This involves a continuous stream of ordered data.	Does not reorder received data.
Multiplexing using ports	Receiving hosts decide the correct application for which the data is destined, based on the port number.	Same as TCP.
Reliable transfer	Acknowledgment of data uses the Sequence and Acknowledgment fields in the TCP header.	This is not a feature of UDP.
Flow control	This process is used to protect buffer space and routing devices.	This is not a feature of UDP.
Connections	This process is used to initialize port numbers and other TCP header fields.	UDP is connectionless.

The User Datagram Protocol (4)

TCP and UDP Headers



* Unless Specified, Lengths Shown Are the Numbers of Bytes

The User Datagram Protocol (5)

- Note the existence of both Source Port and Destination Port fields in the TCP and UDP headers,
- Note absence of Sequence Number and Acknowledgment Number fields in the UDP header.
- UDP gains some advantages over TCP
  - there are fewer bytes of overhead.
  - UDP does not require waiting on acknowledgments
  - or holding the data in memory until it is acknowledged.
  - UDP applications are not artificially slowed by the acknowledgment process,
  - memory is freed more quickly.

The User Datagram Protocol (6)

- UDP loses some features like
  - Not a reliable protocol (no recovery for lost segments)
  - No re-ordering for received segments
  - No flow control (sender doesn't detect receivers over flow)