

Transport Level

Lenuta Alboaie
adria@info.uaic.ro

Content

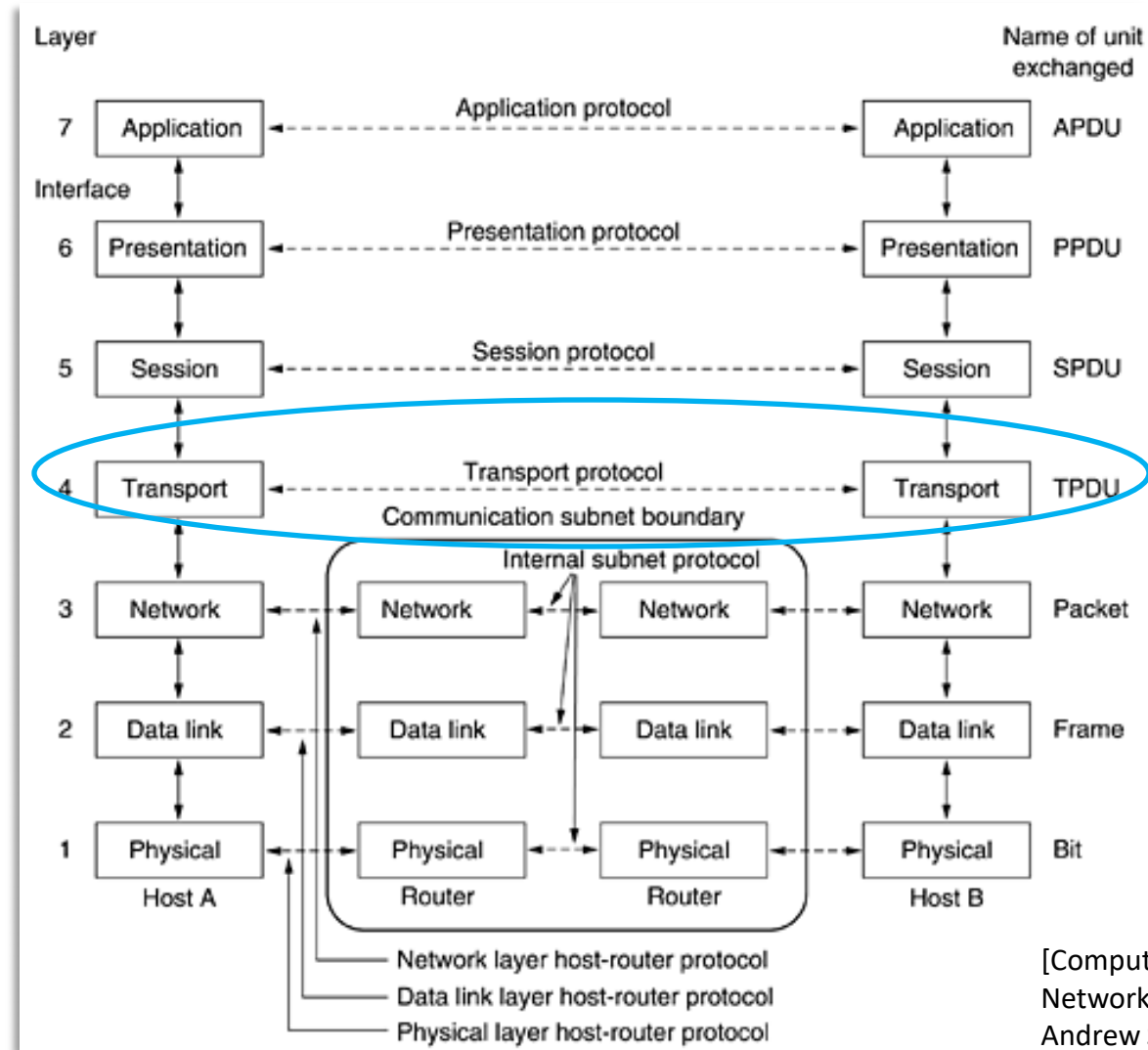
Transport Level

- Preliminary
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- TCP versus UDP

Transport Level | Preliminary

They are
using
Transport
services

They provide
Transport
services

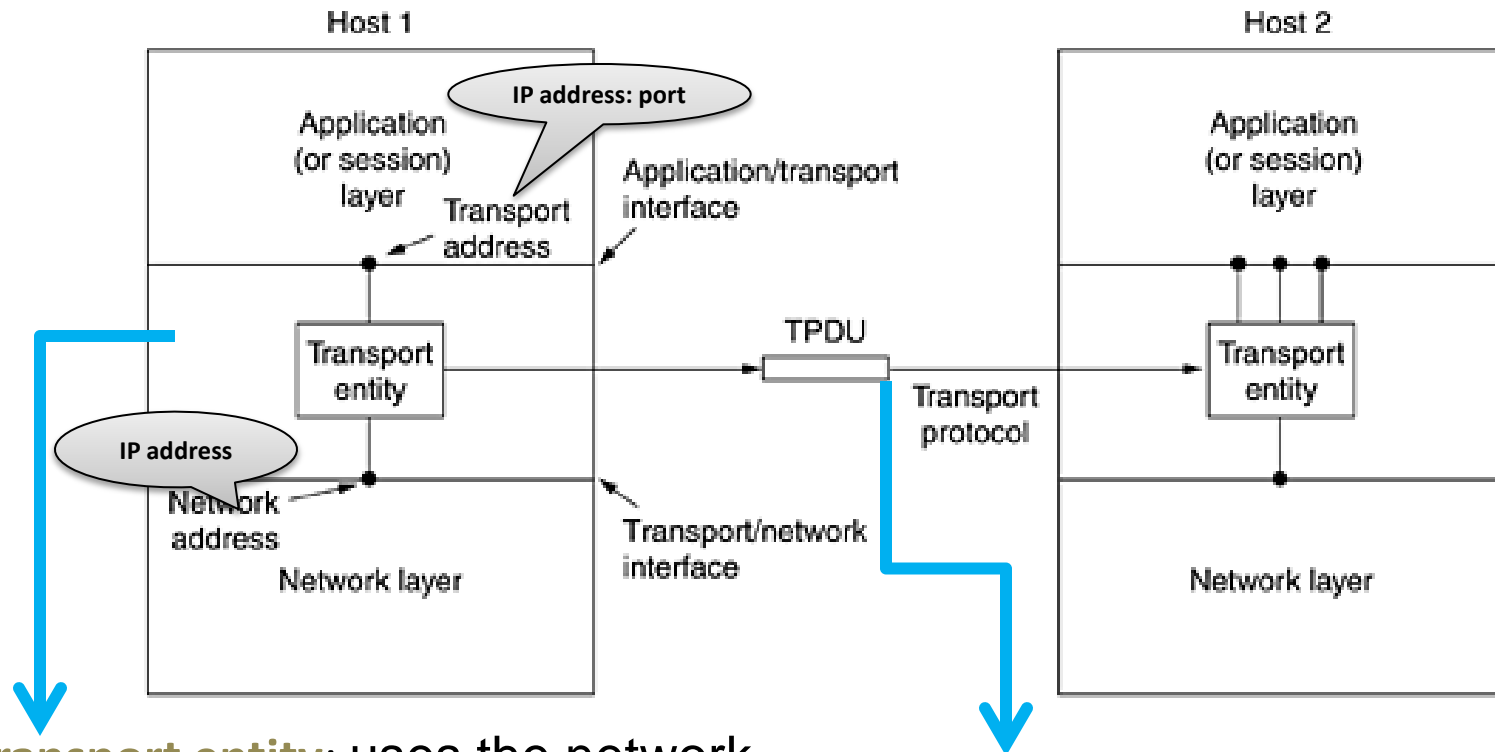


[Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

Figure. OSI Model

Transport Level

- The relationship between levels: network-**transport**-application



Transport entity: uses the network services level and provides services to a superior level

TPDU (Transport Protocol Data Unit) – the data transmission unit

[conform Computer Networks, 2010
– Andrew S. Tanenbaum, et.al.]

Transport Level

- Offers much more reliable services than those at the network layer (e.g. packets lost at the network layer can be detected and the situation can be fixed at the transport level)

QoS (*Quality of Service*)

- The delay in establishing the connection
- Productivity or transfer rate
- Delay in transfer
- The residual error rate
- Protection
- Priority
- Resilience

Transport Level

- Provides logical communication between processes running on different hosts (*end-to-end communication*)
 - (Previous Course!) Network level provides logical communication between **hosts**

PORT

- Add a new dimension of IP addresses from the network level
- It is associated to an application (service) and not a host
- A process can provide more services => can use multiple ports
- A service can correspond to several processes

Transport Level

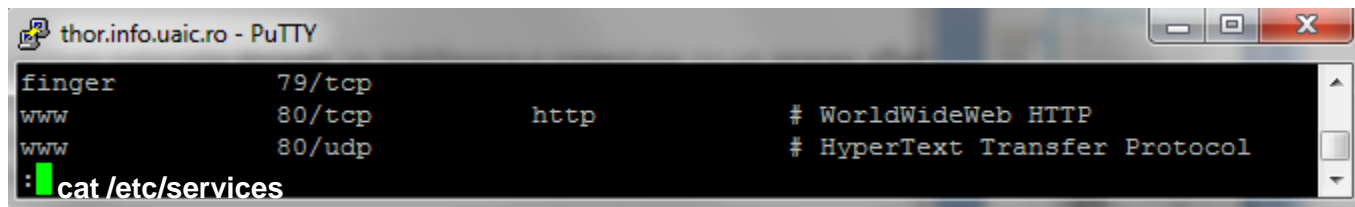
PORT

- It is a 16-bit number and can be set between 0-65535
 - 1-1024 – reserved values (*well-known*), 1-512 system services (IANA – Internet Assigned Number Authority: RFC 1700)

Examples:

/etc/services : system services are associated with ports

HTTP – 80; SMTP – 25; telnet – 23; SSH - 22



The screenshot shows a PuTTY terminal window titled 'thor.info.uaic.ro - PuTTY'. The terminal displays the output of the command 'cat /etc/services'. The output lists several services and their associated ports and protocols:

```
finger      79/tcp
www         80/tcp      http        # WorldWideWeb HTTP
www         80/udp      # HyperText Transfer Protocol
```

The prompt is a green cursor followed by ': cat /etc/services'.

Transport Level

General primitives

- Allow to transport layer users (e.g. application programs) to access services

Primitive	TPDU	Explanation
LISTEN	(nothing)	It blocks until a process tries to connect
CONNECT	CONNECTION REQUEST	Try to establish the connection
SEND	DATA	Send information
RECEIVE	(nothing)	It blocks until it receives sent data
DISCONNECT	DISCONNECTION REQ.	Sent by the party that wants to disconnect

Transport Level

- The most important transport layer protocols:
 - **TCP (Transmission Control Protocol)** – connection-oriented transport protocol; RFC 793 (1122), 1323
 - **UDP (User Datagram Protocol)** – connectionless transport protocol; RFC 768

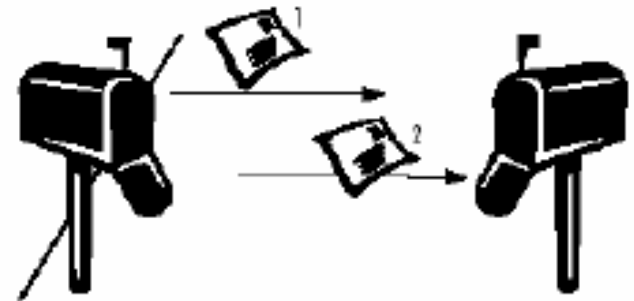
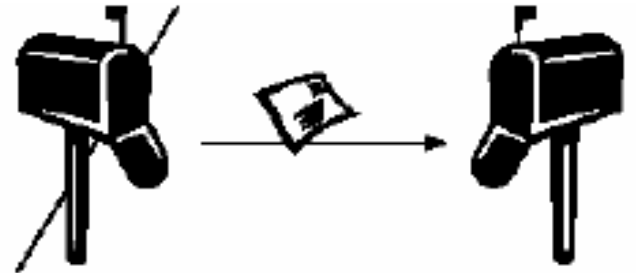
UDP

- Connectionless transport protocol, unsafe, minimal
- Does not offer any additional quality to services
- There is no negotiation regarding receiving data or data confirmation

UDP

- **Analogy: UDP** – similar to **post system**

- Sending a letter
- Does not guarantee the receiving order
- The message may be lost



[conform Retele de calculatoare – curs
2007-2008, Sabin Buraga]

UDP

- Uses IP
- Offers communication services between processes using **ports**
- UDP sends packages: header (8 bytes) + content

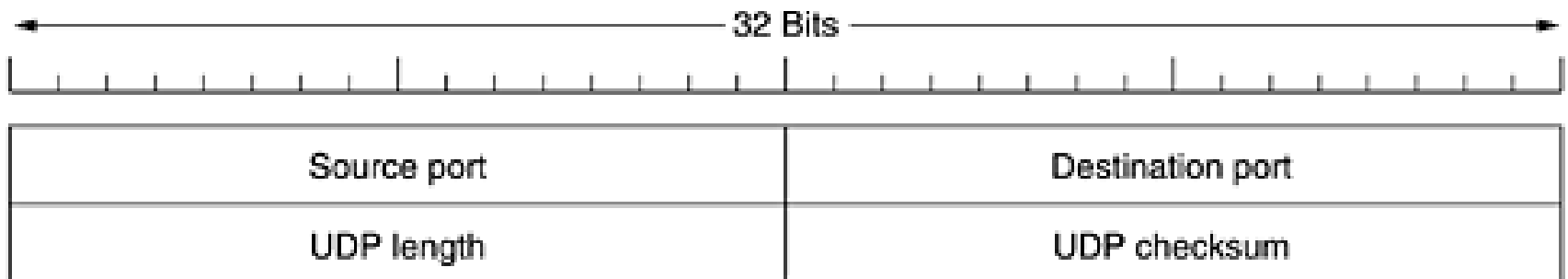


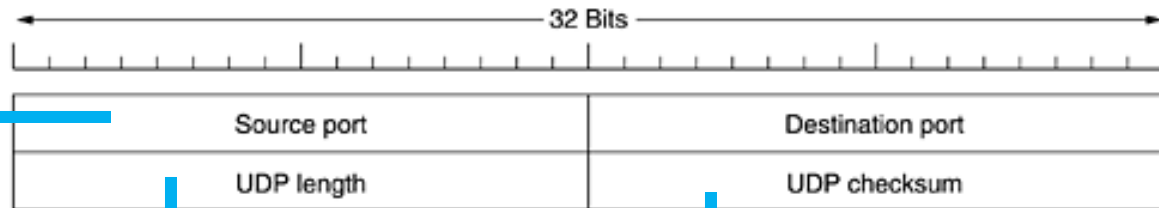
Figure: UDP Header

[Computer Networks, 2010 –
Andrew S. Tanenbaum, et.al.]

UDP

Figure: UDP Header

- *Source port* and *Destination port*
- identify "end-point" from the source and destination machines



UDP length = 8
bytes +
content size

- *UDP checksum*
(is not required)

UDP

- Examples of uses:

- **DNS** (Domain Name System)

Use-case: A needs the host's IP which has the name
www.info.uaic.ro

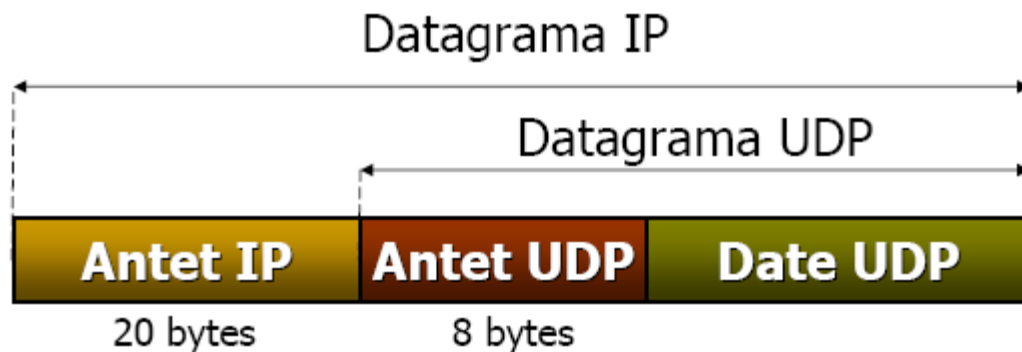
Pas 1. A sends a UDP packet containing hostname:
www.info.uaic.ro

Pas 2. DNS server sends an UDP packet containing the
host's IP address: 85.122.23.146 as response

- **RPC** (Remote Procedure Call)

UDP

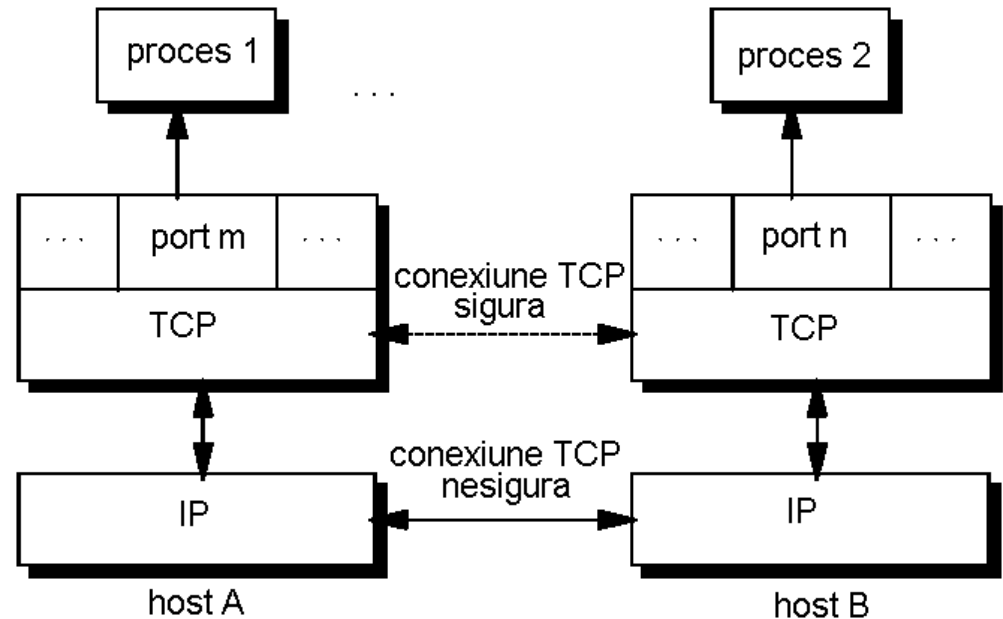
- What doesn't it offer?
 - flow control
 - error control
 - Retransmission of an error package
- What does it offer?
 - Using **ports** expands IP protocol for communication between processes running on different hosts (and not only between host as in the IP case)



[conform Retele de calculatoare – curs
2007-2008, Sabin Buraga]

TCP - Transmission Control Protocol

- Transport connection-oriented protocol without information loss
- Aimed at providing maximum quality of service
- Integrates mechanisms of establishing and releasing connections
- Controls the flow of data (stream-oriented)
- Used by many application protocols: HTTP, SMTP, ...

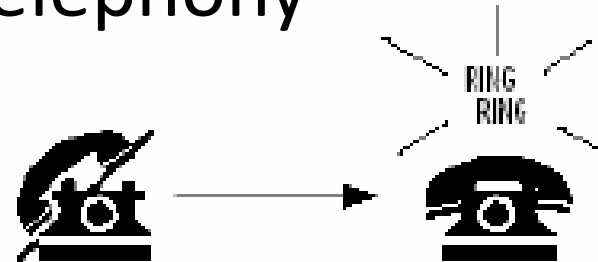


[conform IBM – TCP/IP Tutorial and Technical Overview, 2006]

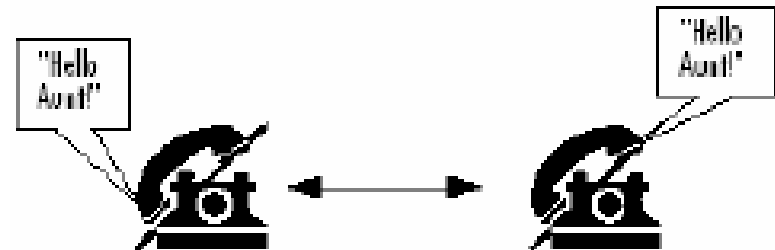
TCP

- **Analogy: TCP** – similar to telephony

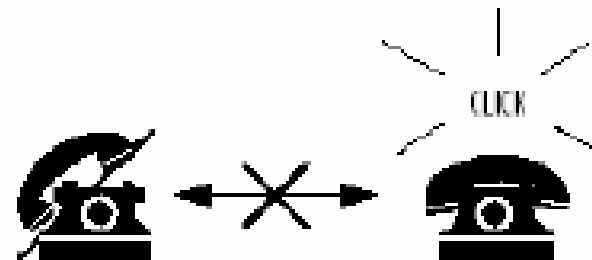
➤ Initiating a call



➤ Dialogue between parties



➤ Call ends



[conform Retele de calculatoare – curs
2007-2008, Sabin Buraga]

TCP

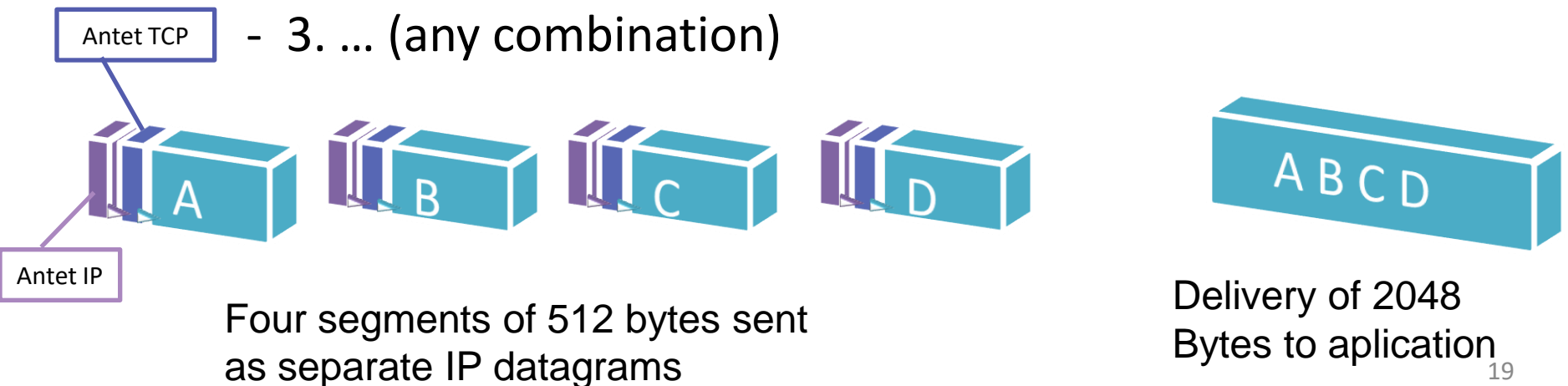
- Uses **connections**, not ports as fundamental abstractions
- Both parties (sender, recipient) must participate to the connection
- Connections are identified by pairs represented by **IP adress :PORT** (*socket*)
 - Example:
(85.122.23.146: 12345, 85.122.23.146: 22)
- One party provides a **passive open** - expect occurrence of connection requests; the communication partner accomplishes an **active open**
- A socket can be shared by multiple connections from the same machine

TCP

- TCP connections are full-duplex
- A TCP is a stream of bytes and not a message flow

Example:

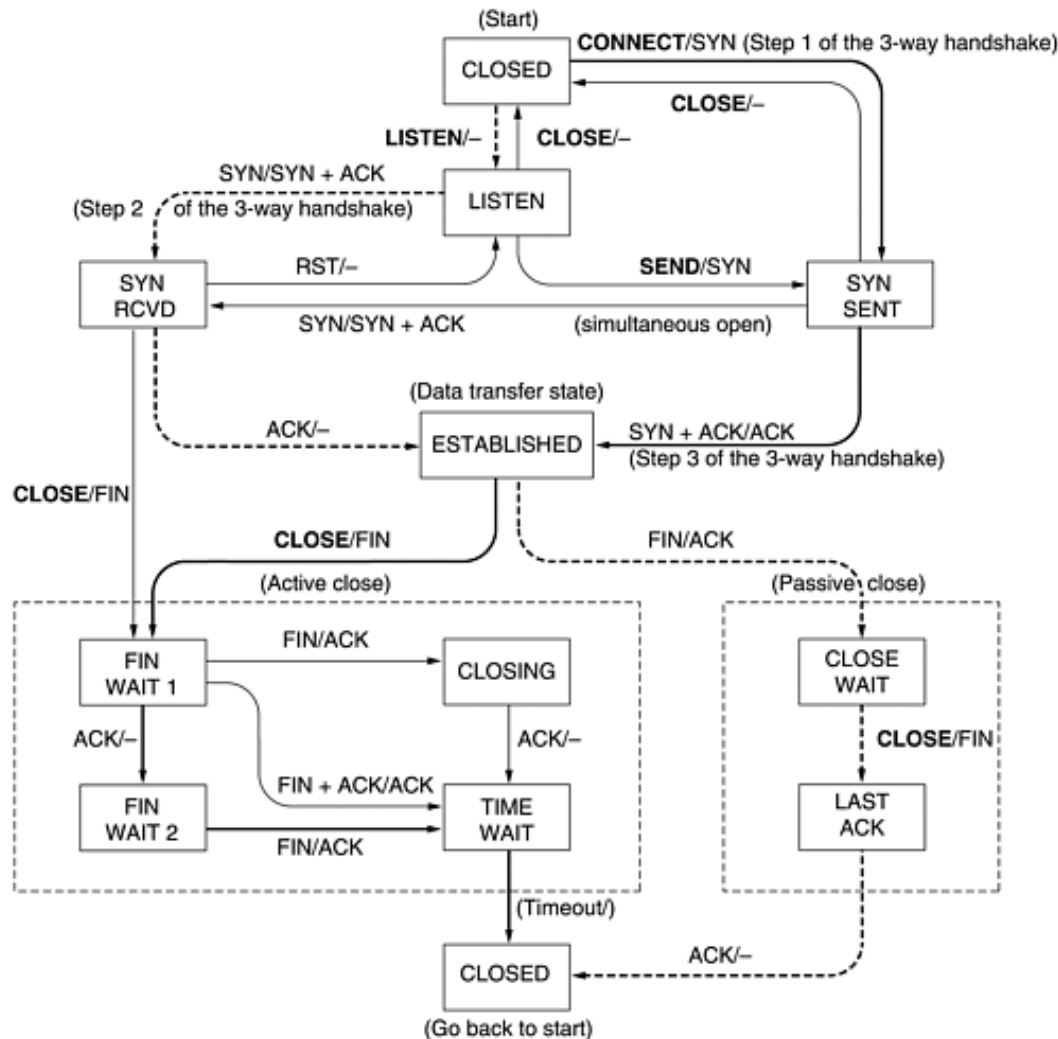
- The transmitter sends four fragments of 512 bytes
- The receiver may receive:
 - 1. Two fragments of 1024 bytes
 - 2. A fragment of one byte and one of 2047 bytes
 - 3. ... (any combination)



TCP

TCP finite state machine

- Shape protocol behavior
- The states are used for connection management



Legend:

———— Client
----- Server

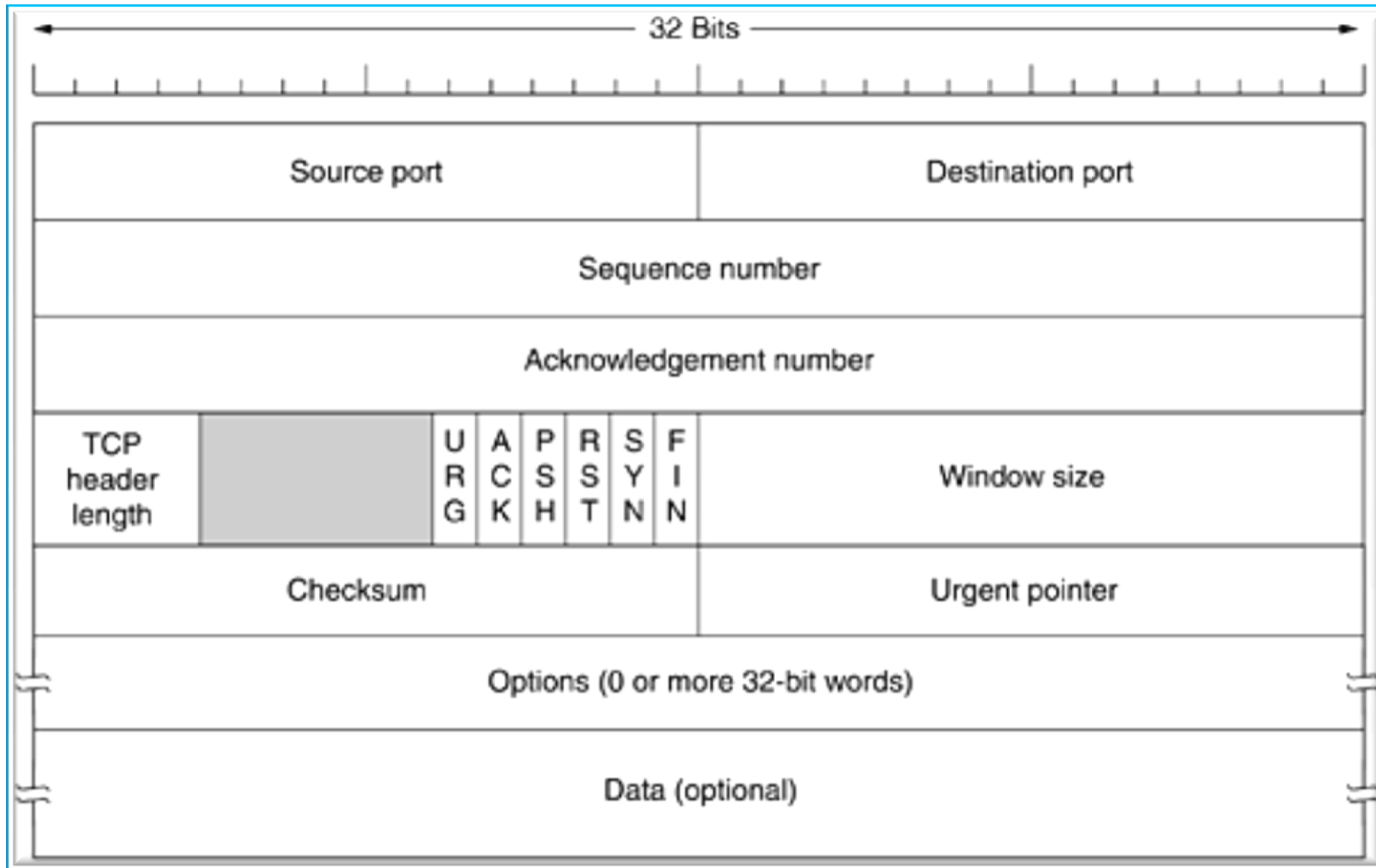
-Each line is labeled with a pair event / action

-Example: ACK/-

[Computer Networks, 2010
– Andrew S. Tanenbaum,
et.al.]

TCP

TCP Header



[Computer Networks, 2010
– Andrew S. Tanenbaum, et.al.]

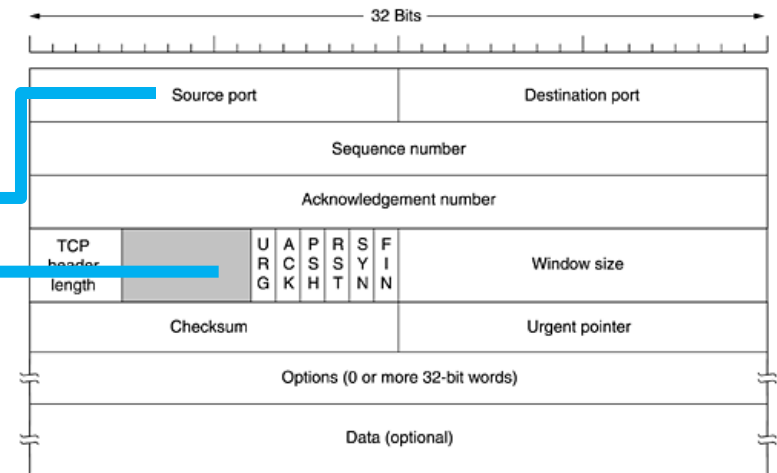
TCP

TCP Header

Source Port and *Destination Port* fields identify endpoints connection

Control Flags

- **URG** (URGence) (1 bit) – if it is set, the *Urgent Pointer* field is used
- **ACK** (ACKnowledgment)
 - If it is set the *Acknowledge Number* field is used;
 - It is set in all acknowledgments packages

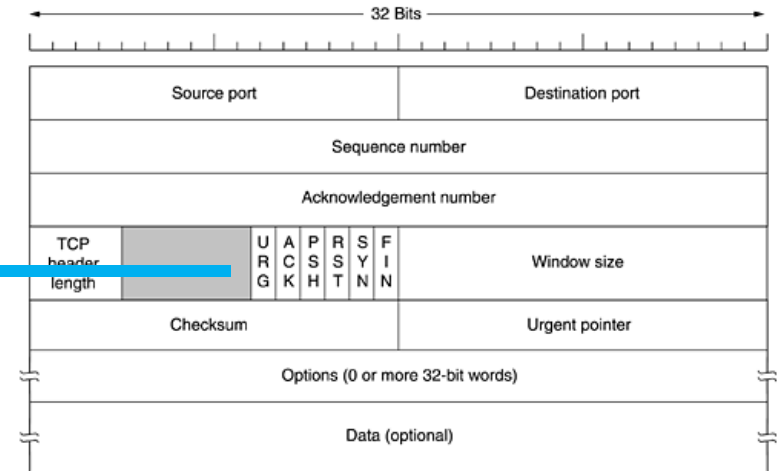


TCP

TCP Header

- **Control Flags**

- **PSH** (PuSH) – data will be transmitted immediately to the recipient's application
- **RST** (ReSeT) – signal connection errors (e.g. opening a connection is refused)



TCP

TCP Header

- **Control Flags**

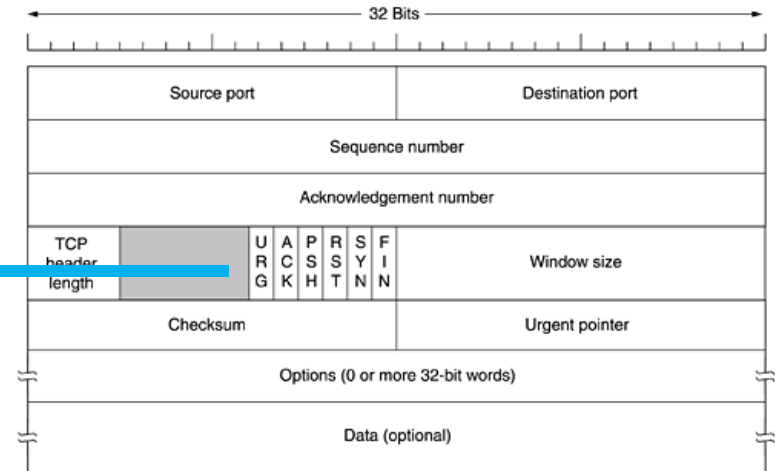
- **SYN** (SYNchronize)

- It is used to establish the connection

(The request to start the connection: SYN=1, ACK=0;
The answer to the request: SYN=1, ACK=1;)

- Indicates the **connection request** and the **accepted connection**

- **FIN** (FINish)- indicates the closing of the connection

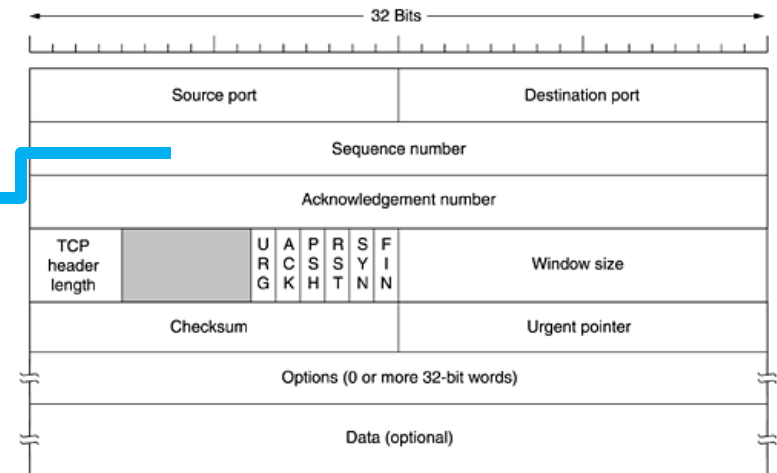


TCP

TCP Header

Sequence Number field(**SEQ**)

- If SYN flag is set => SEQ has as value the initial value of order number;



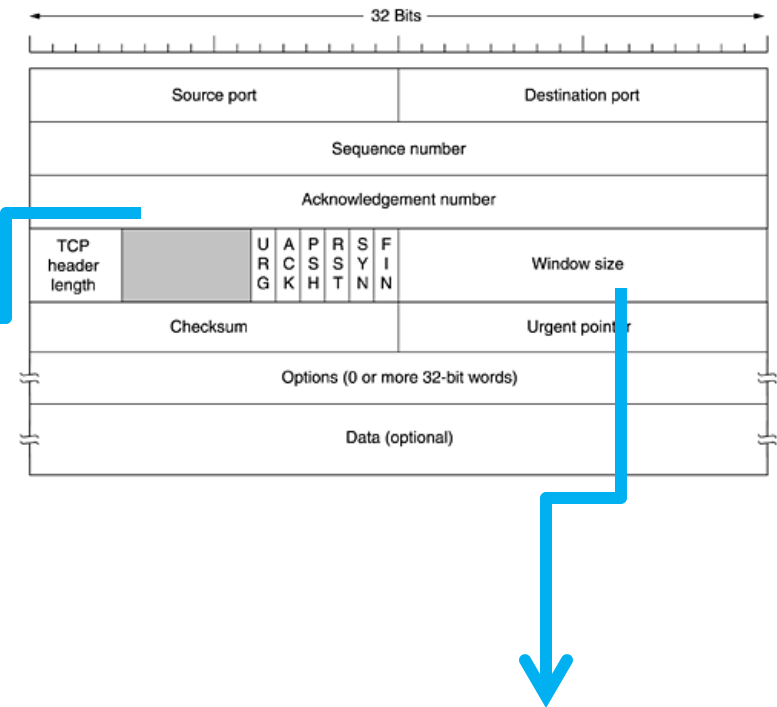
- If SYN flag is not set => SEQ has as value the order number corresponding to the first byte sent in this package;

TCP

TCP Header

Acknowledge Number field

- If ACK flag is set => it is the order number corresponding to the next byte expected to be send;

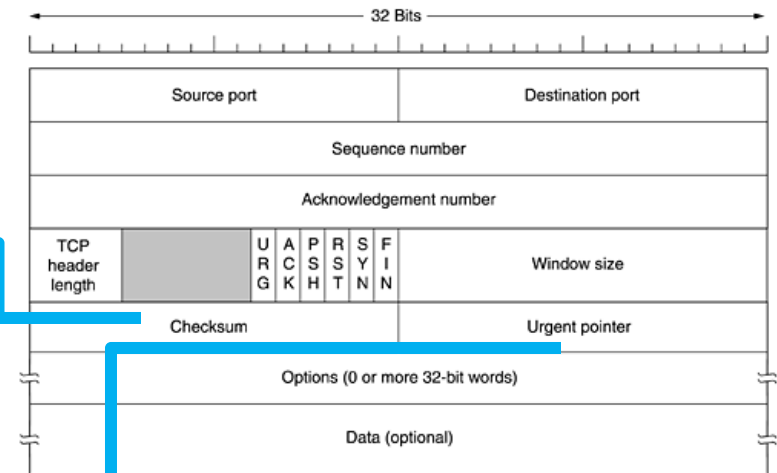


Window size field – the number of bytes that the receiver wants to receive (flow control)

TCP

TCP Header

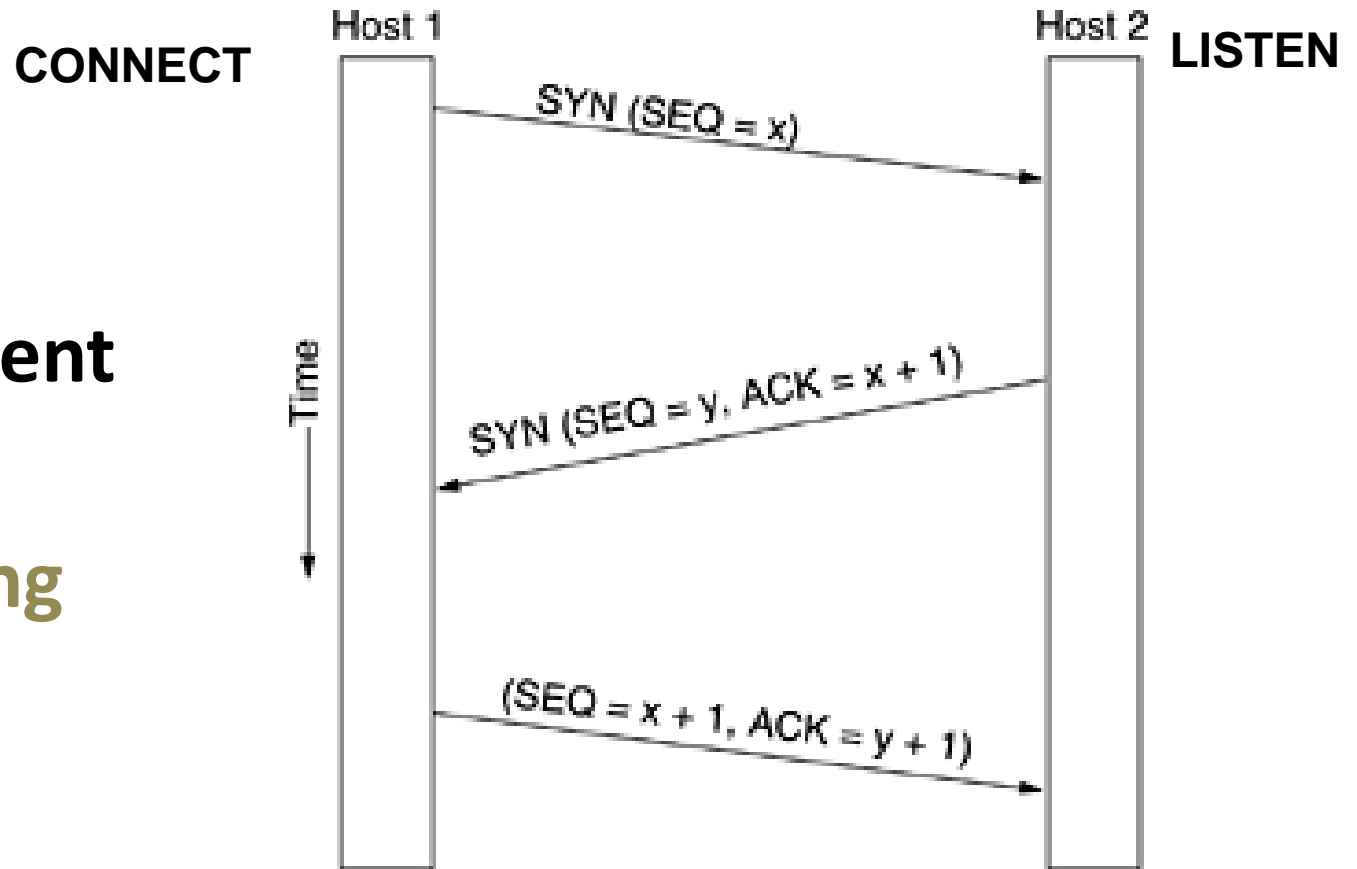
Checksum field – used for verification of TCP packet (header and data)



Urgent Pointer field – if URG flag is set, it indicates the displacement of the number of current orders where emergency information is to be found

TCP connection management

Connection establishment Three-way handshaking

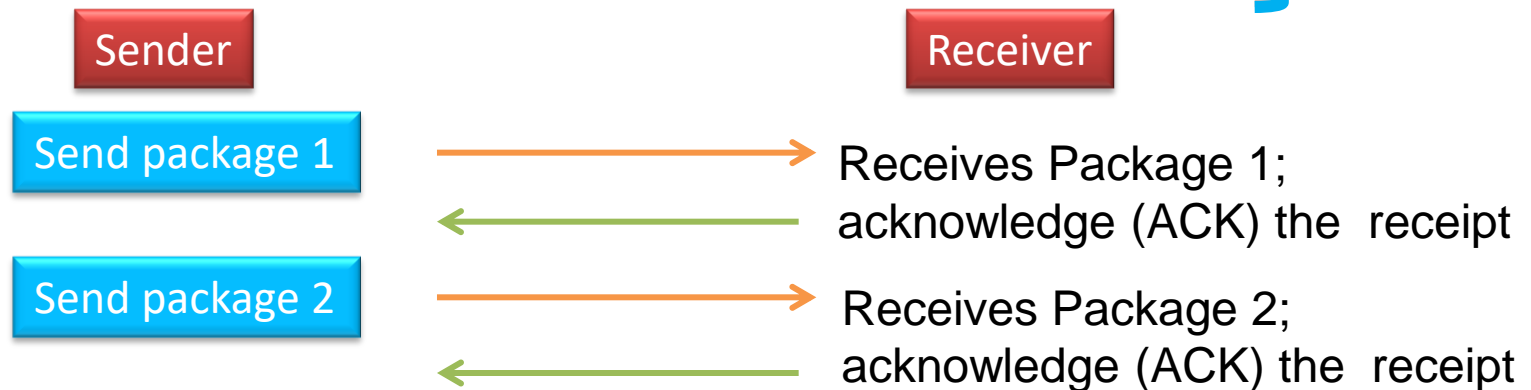


[Computer Networks, 2010
– Andrew S. Tanenbaum,
et.al.]

Flow Control

General mechanism:

- Simple communication protocol: sending a package and then expect confirmation of receipt from the recipient before sending the next packet.
- If acknowledgment (ACK) is not received within a predetermined time, the packet is sent again



The problem: communication mechanism ensures safety but determines the use of only a portion of available network bandwidth

TCP connection management

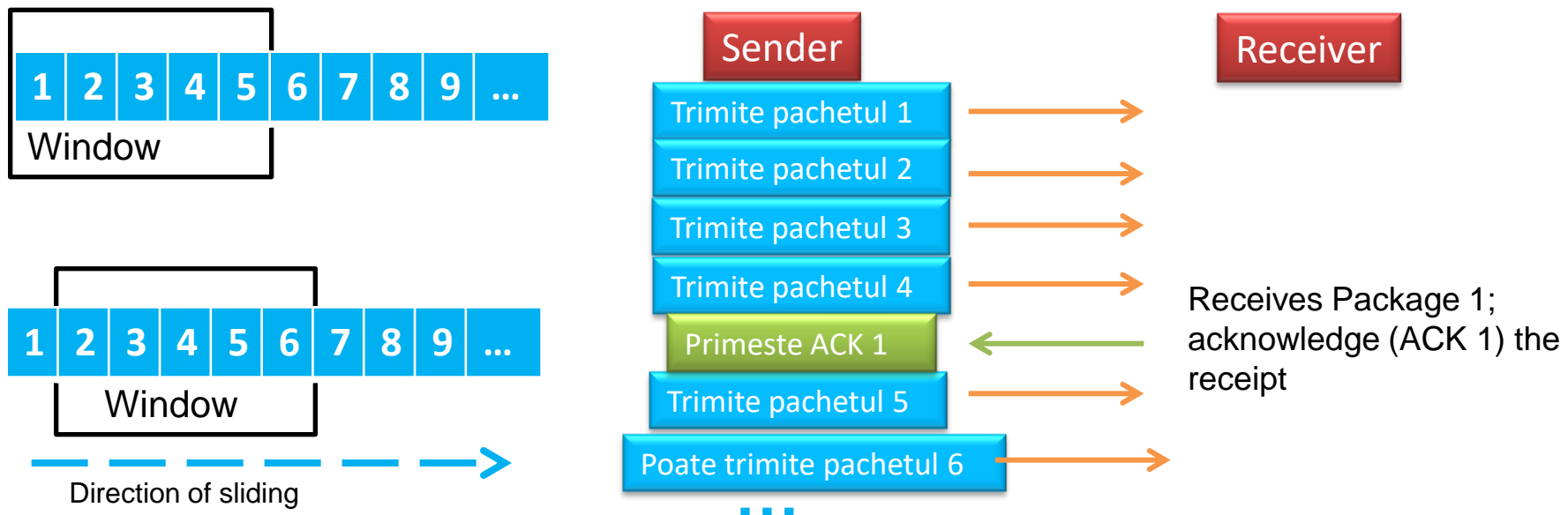
Sliding window - context:

- TCP provides a stream of bytes => sequence numbers are assigned to each byte of stream
- TCP stream is split into TCP segments; a segment (the sending and the acknowledgments) will carry sequence numbers (see slide 19)
- The window size is expressed in bytes and not the number of packages
- The window size is determined by the receiver when the connection is established and it is variable during data transfer
- Each ACK message will include the window size that the receiver is able to operate in the time of communication

TCP connection management

Flow Control

- Communication protocol with **sliding window**:
 - The sender groups packages in a *buffer*
 - the transmitter can send packages from the window, without receiving an acknowledgment (ACK), but starts a timer (which will signal exceeding a predetermined time) for each package
 - receiver must acknowledge each package received, indicating the sequence number of the last byte correctly received
 - After confirmation, the sender slides the window



TCP connection management

Flow control using sliding window – situations

- **Package 2 was lost**
 - the sender does not receive ACK 2, so the window remains on position 1
 - receiver does not receive the package 2 and it will not confirm 3,4,5 packages
 - The timer will signal that the time is elapsed and the package is retransmitted
 - The receiver receives package 2 and will send a ACK 5 confirmation (sequence of 2-5 packets has been received successfully) => sliding window moves four positions after receiving ACK 5
- **Package 2 reached its destination but confirmation was lost**
 - The transmitter does not receive ACK 2 but receives ACK 3 (all packages up to 3 successfully reached) => the sliding window moves to package 4

TCP connection management

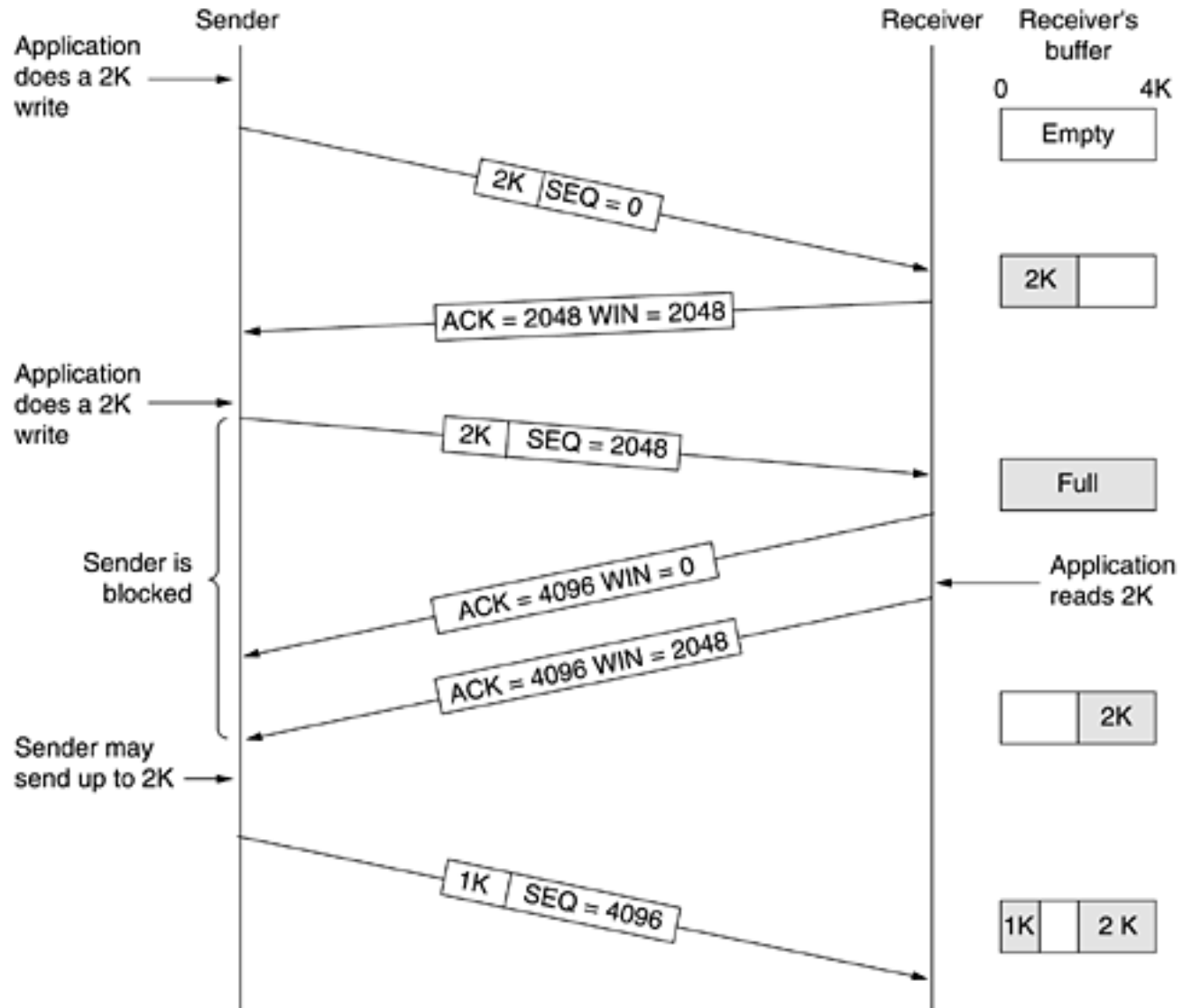
Sliding Windows assures:

- Safe transmission
- Better use of bandwidth => better transmission speed
- Transmission control (receiver may delay data confirmation, taking into account the free space and the windows size)

TCP connection management

Example:

TCP policy for
data
transmission
and windows
size
management



[Computer Networks,
2010 – Andrew S.
Tanenbaum, et.al.]

TCP connection management

- Failure detection & data retransmission
 - Each segment sent contains a sequence number (*Sequence Number*) indicating the position of the bytes transmitted in the data stream
 - The recipient checks sequence number for each segment (it is tested whether certain segments are lost, duplicated, or are not in order) and sends back a confirmation number to each segment (*Acknowledgment Number* specifying the sequence number of the next byte that is expected to be received)
 - Lost segments are detected using a timer
 - Checksums are used to detect errors

TCP connection management

- Connection Reset

- Sometimes abnormal conditions are forcing network software or applications to destroy the connection
- To reset the connection, an end-point initiate the communication ending by sending a TCP segment with RST bit set
- The other side abandons the connection without transmitting any data remained as undelivered
- Transfer in both directions is stopped, the buffers are emptied

TCP connection management

- Forcing the data transmission
 - TCP data stream may be divided into segments of different sizes, different by packages handled by applications => efficient transmission
 - Sometimes there are situations when data must be transmitted, without waiting for the buffers to be filled(e.g., interactive applications)
 - Forcing transmission is achieved by push: PSH bit is set and force the transmission of segments, regardless of filling buffers

TCP connection management

- Closing the connection
 - TCP connections are full-duplex, when an application signals that there is no data to send, TCP connection will close only one direction
 - Communication partner can send a TCP segment with the FIN bit set; confirmation means that on that direction, the communication is effectively stopped;
 - The connection is closed down when both directions are stopped

TCP

- **TCP finite state machine**

- Shapes protocol behavior
- The states are used for connection management

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

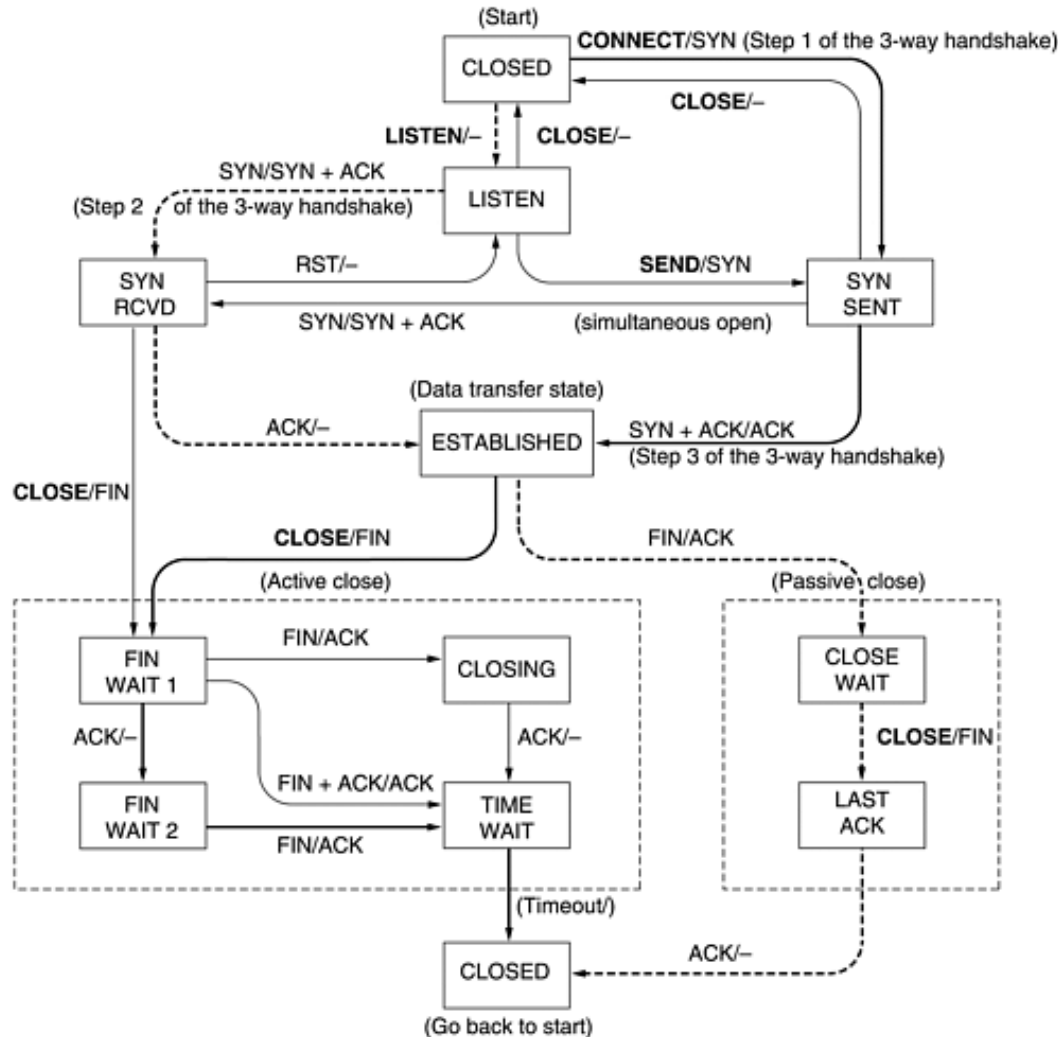
Figure. States of TCP finite state machine

[conform Computer Networks, 2010 – Andrew S. Tanenbaum, et.al.]

TCP

TCP finite state machine

- Shape protocol behavior
- The states are used for connection management



Legend:

———— Client
----- Server

-Each line is labeled with a pair event / action

-Example: ACK/-

[Computer Networks, 2010
– Andrew S. Tanenbaum,
et.al.]

TCP

- TCP finite state machine

- Connection Establishment:

- **CLOSED** – from this state an active open can be requested (it passes in **SYN_SENT**) or active open (**SYN_RCVD**)
 - **LISTEN** – – from this state an active open can be requested (it passes in **SYN_SENT**) or active open (**SYN_RCVD\0**)

- Established connection:

- **ESTABLISHED** – The data can be sent (from this state it can be passed in **CLOSE_WAIT** or **FIN_WAIT_1**)

- Disconnecting initiated by the partner

- **CLOSE_WAIT, LAST_ACK, CLOSE**

- States that intervene in the process of disconnection

- **FIN_WAIT_1, FIN_WAIT_2, CLOSING, TIME_WAIT**

TCP

Example:
netstat -t

```
adria@thor:~/html/teach/courses/net$ netstat -t
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 thor.info.uaic.ro:smtp 186.122.246.108:33374   SYN_RECV
tcp        0      0 localhost:60000         localhost:45740         ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps mail.traveltech.c:55156 ESTABLISHED
tcp        0      0 localhost:45740         localhost:60000         ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps 81.253.57.112:51462     ESTABLISHED
tcp        0      0 thor.info.uaic.ro:smtp  fenrir.info.uaic.:59806 TIME_WAIT
tcp        0      0 thor.info.uaic.ro:imaps  ia.info.uaic.ro:51058   ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  77-58-250-39.dcli:56424 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:59605 fenrir.info.uaic.ro:ssh ESTABLISHED
tcp        0      0 localhost:57572         localhost:spamd         TIME_WAIT
tcp        0      0 thor.info.uaic.ro:pop3s  79-112-42-154.iasi:2019 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:pop3s  info-c-117.info.ua:1302 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:ssh    mail.traveltech.c:52266 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:pop3s  info-c-59.info.ua:50149 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  mail.traveltech.c:55152 ESTABLISHED
tcp        0      0 localhost:58053         localhost:10025         TIME_WAIT
tcp        0      0 thor.info.uaic.ro:imaps  info-c-70.info.uai:1266 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  info-c-91.info.uai:4285 ESTABLISHED
tcp        0      0 192.168.103.2:39107     pdc:65022              ESTABLISHED
tcp        0      0 thor.info.uaic.ro:pop3s  info-c-59.info.ua:55896 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  77-58-250-39.dcli:56476 ESTABLISHED
tcp        0      0 192.168.103.2:39082     pdc:65022              ESTABLISHED
tcp        0      0 thor.info.uaic.ro:smtp   main.dntis.ro:52854     ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  ws70-228.unine.ch:35907 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  info-c-70.info.uai:1364 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:ssh    ia.info.uaic.ro:40542   ESTABLISHED
tcp        0    396 thor.info.uaic.ro:ssh    79-112-21-031.ias:54540 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  ia.info.uaic.ro:39325   ESTABLISHED
tcp        0      0 thor.info.uaic.ro:pop3s  79-112-42-179.iasi:1146 ESTABLISHED
tcp        0      0 thor.info.uaic.ro:imaps  81.253.57.112:51461     ESTABLISHED
```

TCP

- Examples of TCP uses:
 - Most application protocols :
 - HTTP
 - TELNET
 - SMTP
 - SSH
 - ...

TCP versus UDP

- Both rely on IP uses ports
- Transmission unit
 - TCP -> TCP Segment
 - UDP -> UDP Packet

TCP versus UDP

- UDP offers minimal transport services (minimum effort transmission)
- TCP provides connection-oriented, full-duplex, safe - transport streams of bytes (-> complex transmission mechanism)
- Using TCP or UDP depends on the application: e-mail, file transfer, operating in real-time multimedia transmissions in real time chat, ...

Summary

Transport Level

- Preliminary
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- TCP versus UDP

Bibliography

- Andrew S. Tanenbaum, David J. Wetherall, Computer Networks (5th Edition), ISBN-10: 0132126958 , Publication Date: October 7, 2010
- A. Tanenbaum, Computer Networks. 4th Edition. Prentice Hall. 2003
- James F. Kurose, Keith W. Ross; Computer Networking: A Top-Down Approach (5th Edition), ISBN-10: 0136079679
- Lydia Parziale, David T. Britt, Chuck Davis, Jason Forrester, Wei Liu, Carolyn Matthews, Nicolas Rosselot , IBM – TCP/IP Tutorial and Technical Overview, 2006
- Tamara Dean, Network +Guide to Networks (Editia 5), ISBN-10: 1-423-90245-9, 2009



Questions?