

Advanced Networking

TCP

Renato Lo Cigno
Renato.LoCigno@dit.unitn.it

Content

- Some details on window protocols
- TCP headers and formats
- TCP Options
- TCP flow control
- TCP Congestion control (most bulky!)



Basic Selective Repeat

- Requires 1 ACK per packet
- Positive ACK if the packet is received in order or it is received out-of-order
- Negative ACK if the packet is missing
 - Problem: lost ACKs block the protocol
- Implicit negative ACK by repeating the ACK of the last in-order packet
- Transmitter builds a local copy of the receiver window and retransmit only lost packets
- Same effect can be obtained with cumulative ACKs, with the limit of recovering 1pkt per RTT

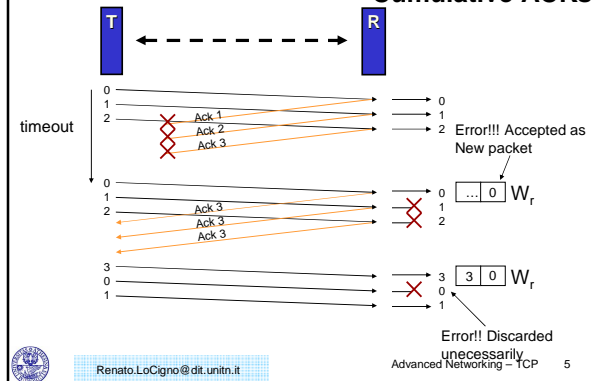


Window relations in SR

- W = size of the counting space (bytes, packets, ...)
- W_t = Transmitter window size
- W_r = Receiver window size
- Must be $W_t + W_r < W$ to ensure working correctly
- Relation holds for both cumulative and selective ACKs



Example: $W=4$, $W_t=3$, $W_r=3$ Cumulative ACKs



TCP: Bibliography

- Richard Stevens: TCP Illustrated, vol.1
- William Stallings:
- RFC 793 (1981)
 - Transmission Control Protocol
- RFC 1122/1123: (1989)
 - Requirements for Internet Hosts
- RFC 1323: (1992)
 - TCP Extensions for High Performance
- RFC 2018: (1996)
 - TCP Selective Acknowledgment Options



TCP: riferimenti bibliografici

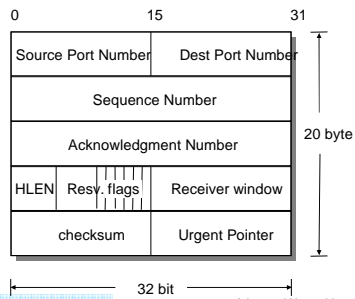
- RFC 2581:
 - TCP Congestion Control (PRP STD)
- RFC 2582:
 - The NewReno Modification to TCP's Fast Recovery Algorithm
- RFC 2883:
 - An Extension to the Selective Acknowledgement (SACK) Option for TCP
- RFC 2988:
 - Computing TCP's Retransmission Timer
-



Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 7

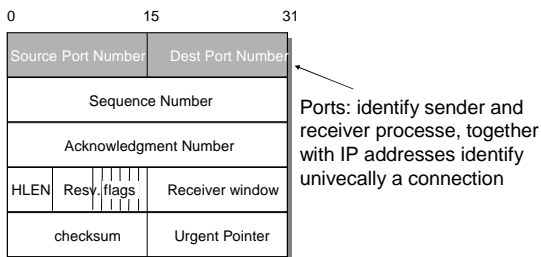
TCP header (no options)



Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 8

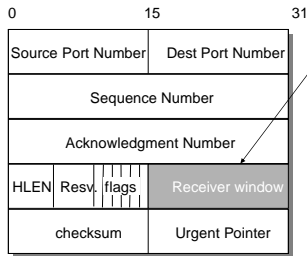
TCP header



Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 9

TCP Header



Number of bytes, starting and including the one in the ACK field that the receiver can accept; implements flow control. 16 bits, the maximum value for rwnd is 65535 byte, unless the *window scaling option* is enabled (more later on)



Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 16

The receiver window drives throughput

- Throughput is given by W/RTT
- Maximum data per RTT is max RWND:
 - 16-bit rwnd = 64kB max
- Given $RTT=100ms$ the following windows are required to exploit the relative channels

Channel (capacity)	banda x ritardo
T1 (1.5Mbps)	18kB
Ethernet (10Mbps)	122kB
T3 (45Mbps)	549kB
FDDI (100Mbps)	1.2MB
STS-3 (155Mbps)	1.8MB
STS-12 (622Mbps)	7.4MB
STS-24 (1.2Gbps)	14.8MB

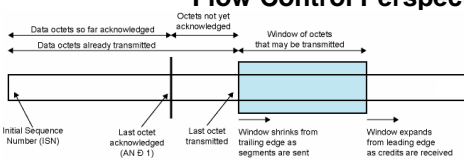
- These limits can be overcome using the window scale option



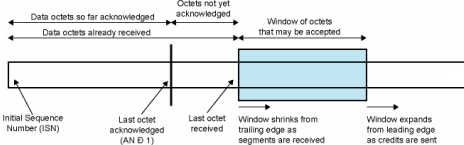
Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 17

Sending and Receiving Flow Control Perspectives



(a) Send sequence space



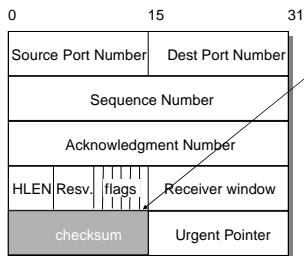
(b) Receive sequence space



Renato.LoCigno@dit.unitn.it

Advanced Networking – TCP 18

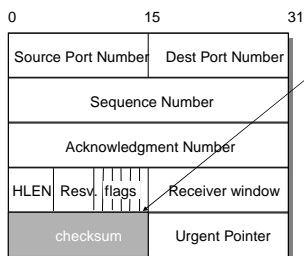
TCP header



Checksum is compulsory and is computed on header and data plus the pseudo-header including IP address and protocol type. This is a layering violation, but a useful one!



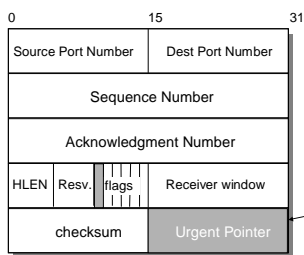
Intestazione TCP



- Checksum algorithm
 - align header, data and pseudo-header to 16 bits
 - sum every line in 1s complement algebra
 - the result is a 32 bit number that is divided in two 16 bits parts
 - sum in 1s complement the two parts including the overflow
 - The result is the checksum inserted in the header



Intestazione TCP



It's the pointer to what is the "urgent data" in the data field (e.g. ctrl-C in a telnet session).

It's expressed as offset wrt the seq. no.

Valid only if URG is set



TCP options

- It's an extension to the header, used to add features to the protocol, many options exist
- Comes before data and it's in multiple of four bytes
- Most used are:
 - MSS (Maximum Segment Size), sent in the SYN segment to define the "optimal" size of segments to be received, not negotiated; default is 536 byte
 - Timestamping of packet to improve RTT calculation



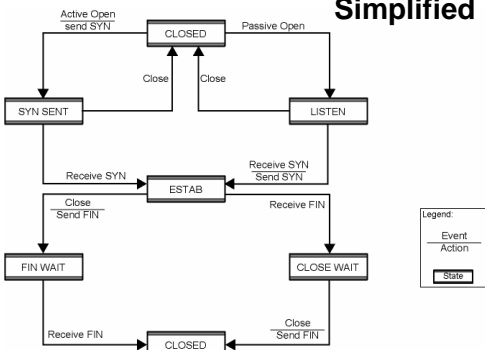
TCP options

- Window scale
 - Included in SYN segment
 - Window field gives credit allocation in octets
 - With Window Scale value in Window field multiplied by 2^F
 - F is the value of window scale option
- Sack-permitted
 - Selective acknowledgement allowed
- Sack
 - Receiver can inform sender of all segments received successfully
 - Sender retransmit segments not received SACK, to enable



Both must be issued for successful negotiation

State Diagram for TCP connections Simplified FSM



Side Initiating Termination

- TS user Close request
- Transport entity sends FIN, requesting termination
- Connection placed in FIN WAIT state
 - Continue to accept data and deliver data to user
 - Not send any more data
- When FIN received, inform user and close connection



Side Not Initiating Termination

- FIN received
- Inform TS user Place connection in CLOSE WAIT state
 - Continue to accept data from TS user and transmit it
- TS user issues CLOSE primitive
- Transport entity sends FIN
- Connection closed

- All outstanding data is transmitted from both sides
- Both sides agree to terminate



Unreliable Network Service

- E.g.
 - internet using IP,
 - frame relay using LAPF
 - IEEE 802.3 using unacknowledged connectionless LLC
- Segments may get lost
- Segments may arrive out of order
- ... we know this all but

- **What are the consequences on a reliable transport layer?**



Problems

- Ordered Delivery
- Retransmission strategy
- Duplication detection
- Flow control
- Connection establishment
- Connection termination
- Crash recovery



Ordered Delivery

- Segments may arrive out of order
- Number segments sequentially
- TCP numbers each octet sequentially
- Segments are numbered by the first octet number in the segment



Retransmission Strategy

- Segment damaged in transit
- Segment fails to arrive
- Transmitter does not know of failure
- Receiver must acknowledge successful receipt
- Use cumulative acknowledgement
- Time out waiting for ACK triggers re-transmission



Timer Value

- Fixed timer
 - Based on understanding of network behavior
 - Can not adapt to changing network conditions
 - Too small leads to unnecessary re-transmissions
 - Too large and response to lost segments is slow
 - Should be a bit longer than round trip time
- Adaptive scheme
 - May not ACK immediately
 - Can not distinguish between ACK of original segment and re-transmitted segment
 - Conditions may change suddenly



Duplication Detection

- If ACK lost, segment is re-transmitted
- Receiver must recognize duplicates
- Duplicate received prior to closing connection
 - Receiver assumes ACK lost and ACKs duplicate
 - Sender must not get confused with multiple ACKs
 - Sequence number space large enough to not cycle within maximum life of segment
- Duplicate received after closing connection



Flow Control

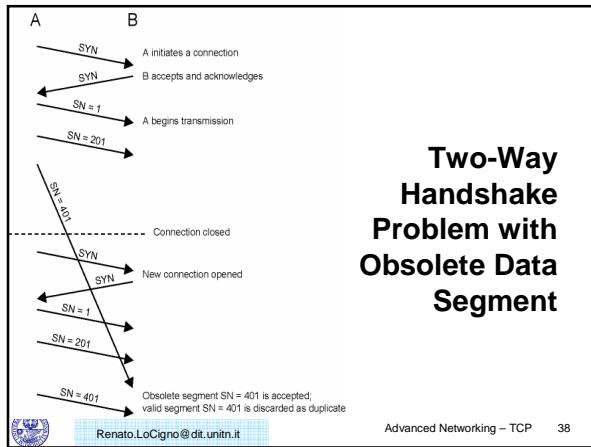
- Credit allocation
- Problem if $AN=i$, $W=0$ closing window
- Send $AN=i$, $W=j$ to reopen, but this is lost
- Sender thinks window is closed, receiver thinks it is open
- Use window timer
- If timer expires, send something
 - Could be re-transmission of previous segment

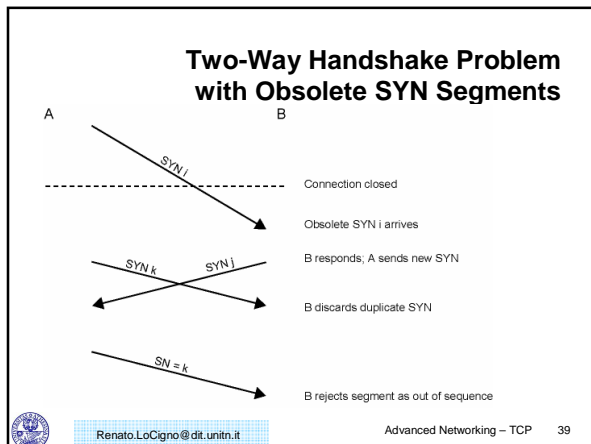


Connection Establishment

- Two way handshake
 - A send SYN, B replies with SYN
 - Lost SYN handled by re-transmission
 - Can lead to duplicate SYNs
 - Ignore duplicate SYNs once connected
- Lost or delayed data segments can cause connection problems
 - Segment from old connections
 - Start segment numbers far removed from previous connection
 - Use SYN i
 - Need ACK to include i
- Solved using Three Way Handshake







Graceful Close

- Composition of the two half close
 - Send FIN i and receive AN i
 - Receive FIN j and send AN j
- Wait twice maximum expected segment lifetime
- Guarantees that all data in both directions is correctly sent
- Ensures proper freeing of logical resources on both sides

- Is slow and requires cooperation ...



Failure Recovery

- After restart all state info is lost
- Connection is half open
 - Side that did not crash still thinks it is connected
- Close connection using persistence timer
 - Wait for ACK for (time out) * (number of retries)
 - When expired, close connection and inform user
- Send RST i in response to any i segment arriving
- User must decide whether to reconnect
 - Problems with lost or duplicate data



Data Transport

- Full duplex
- Timely
 - Associate timeout with data submitted for transmission
 - If data not delivered within timeout, user notified of service failure and connection abruptly terminates
- Ordered
- Labelled
 - Establish connection only if security designations match
 - If precedence levels do not match higher level used
- Flow controlled
- Error controlled
 - Simple checksum
 - Delivers data free of errors within probabilities supported by checksum



Special Capabilities

- Data stream push
 - TCP decides when enough data available to form segment
 - Push flag requires transmission of all outstanding data up to and including that labelled
 - Receiver will deliver data in same way
- Urgent data signalling
 - Tells destination user that significant or "urgent" data is in stream
 - Destination user determines appropriate action
- Error Reporting
 - TCP will report service failure due to internetwork conditions for which TCP cannot compensate

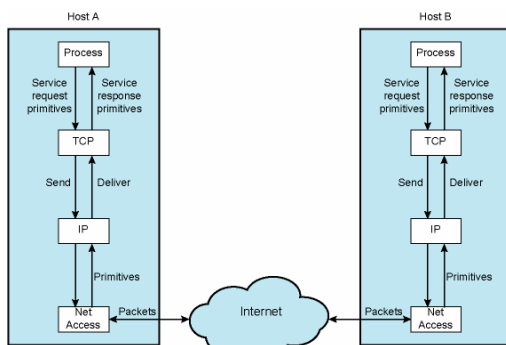


TCP Service Primitives

- Services defined in terms of primitives and parameters
- Primitive specifies function to be performed
- Parameters pass data and control information
- These defines the so-called socket programming



Use of TCP and IP Service Primitives

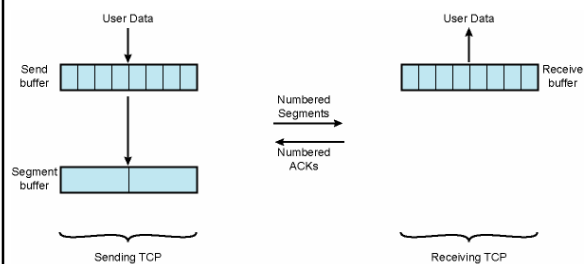


Basic Operation

- Data transmitted in segments
 - TCP header and portion of user data
 - Some segments carry no data
 - For connection management
- Data passed to TCP by user in sequence of Send primitives
- Buffered in send buffer
- TCP assembles data from buffer into segment and transmits
- Segment transmitted by IP service
- Delivered to destination TCP entity
- Strips off header and places data in receive buffer
- TCP notifies its user by Deliver primitive that data are available



Basic TCP Operation



Items Passed to IP

- TCP passes some parameters down to IP
 - Precedence
 - Normal delay/low delay
 - Normal throughput/high throughput
 - Normal reliability/high reliability
 - Security



TCP Mechanisms (1)

- Connection establishment
 - Three way handshake
 - Between pairs of ports
 - One port can connect to multiple destinations
- Data transfer
 - Logical stream of octets
 - Octets numbered modulo 2^{32}
 - Flow control by credit allocation of number of octets
 - Data buffered at transmitter and receiver



Implementation Policy Options

- Send
- Deliver
- Accept
- Retransmit
- Acknowledge



Send

- If no push or close TCP entity transmits at its own convenience
- Data buffered at transmit buffer
- May construct segment per data batch
- May wait for certain amount of data



Deliver

- In absence of push, deliver data at own convenience
- May deliver as each in order segment received
- May buffer data from more than one segment



Accept

- Segments may arrive out of order
- In order
 - Only accept segments in order
 - Discard out of order segments
- In windows
 - Accept all segments within receive window



Retransmit

- TCP maintains queue of segments transmitted but not acknowledged
- TCP will retransmit if not ACKed in given time
 - First only
 - Batch
 - Individual



Acknowledgement

- Immediate
 - send one ACK per packet
- Delayed
 - send ACKs with delay to allow data piggybacking or every 2 segments received
- Cumulative
 - Always ACK all the data received in order, allows for quasi-selective repeat without (almost) any overhead