TCP

TCP Services TCP is designed to provide reliable communication between pairs of processes (TCP users) across a variety of reliable and unreliable networks and internets. TCP provides two useful facilities for labeling data: push and urgent.

• **Data stream push:** Ordinarily, TCP decides when sufficient data have accumulated to form a segment for transmission. The TCP user can require TCP to transmit all outstanding data up to and including that labeled with a push flag. On the receiving end, TCP will deliver these data to the user in the same manner. A user might request this if it has come to a logical break in the data.

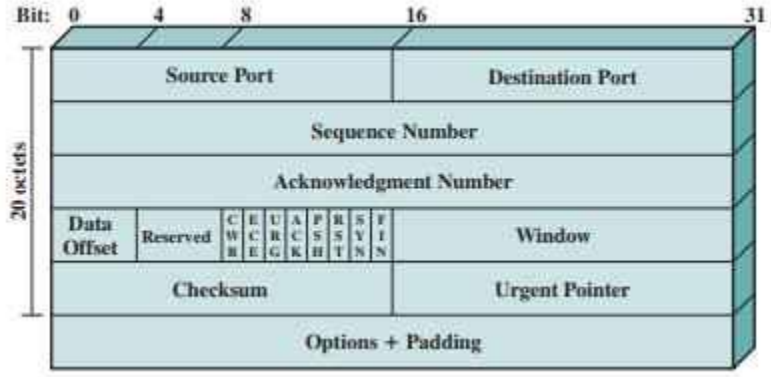
• **Urgent data signaling:** This provides a means of informing the destination TCP user that significant or "urgent" data is in the upcoming data stream. It is up to the destination user to determine appropriate action.

TCP Header Format

TCP uses only a single type of protocol data unit, called a TCP segment. The header is shown in Figure 15.10. Because one header must serve to perform all protocol mechanisms, it is rather large, with a minimum length of 20 octets. The fields are as follows:

- Source Port (16 bits): Source TCP user. Example values are Telnet = 23; TFTP = 69; HTTP = 80.
- Destination Port (16 bits): Destination TCP user.
- Sequence Number (32 bits): Sequence number of the first data octet in this segment except when the SYN flag is set. If SYN is set, this field contains the initial sequence number (ISN) and the first data octet in this segment has sequence number ISN + 1.
- Acknowledgment Number (32 bits): Contains the sequence number of the next data octet that the TCP entity expects to receive.
- Data Offset (4 bits): Number of 32-bit words in the header.
- Reserved (4 bits): Reserved for future use.
- Flags (8 bits): For each flag, if set to 1, the meaning is CWR: congestion window reduced. ECE: ECN-Echo; the CWR and ECE bits, defined in RFC 3168, are used for the explicit congestion notification function; a discussion of this function is beyond our scope.
    - URG: urgent pointer field significant.
    - ACK: acknowledgment field significant.
    - PSH: push function.
    - RST: reset the connection.
    - SYN: synchronize the sequence numbers.
    - FIN: no more data from sender.

- Window (16 bits): Flow control credit allocation, in octets. Contains the number of data octets, beginning with the sequence number indicated in the acknowledgment field that the sender is willing to accept.
- Checksum (16 bits): The ones complement of the ones-complement sum of all the 16-bit words in the segment plus a pseudoheader, described subsequently.2
- Urgent Pointer (16 bits): This value, when added to the segment sequence number, contains the sequence number of the last octet in a sequence of urgent data. This allows the receiver to know how much urgent data is coming.
- Options (Variable): An example is the option that specifies the maximum segment size that will be accepted



TCP Implementation Policy Options

The TCP standard provides a precise specification of the protocol to be used between TCP entities. However, certain aspects of the protocol admit several possible implementation options. Although two implementations that choose alternative options will be interoperable, there may be performance implications. The design areas for which options are specified are the following:

• Send policy

• Deliver policy

• Accept policy

• Retransmit policy

• Acknowledge policy

Send Policy

In the absence of both pushed data and a closed transmission window a sending TCP entity is free to transmit data at its own convenience, within its current credit allocation. As data are issued by the user, they are buffered in the transmit buffer. TCP may construct a segment for each batch of data provided by its user or it may wait until a certain amount of data accumulates before constructing and sending a segment. The actual policy will depend on performance considerations. If transmissions are infrequent and large, there is low overhead in terms of segment generation and processing. On the other hand, if transmissions are frequent and small, the system is providing quick response.

Deliver Policy

In the absence of a Push, a receiving TCP entity is free to deliver data to the user at its own convenience. It may deliver data as each in-order segment is received, or it may buffer data from a number of segments in the receive buffer before delivery. The actual policy will depend on performance considerations. If deliveries are infrequent and large, the user is not receiving data as promptly as may be desirable. On the other hand, if deliveries are frequent and small, there may be unnecessary processing both in TCP and in the user software, as well as an unnecessary number of operating system interrupts.

Accept Policy

When all data segments arrive in order over a TCP connection, TCP places the data in a receive buffer for delivery to the user. It is possible, however, for segments to arrive out of order. In this case, the receiving TCP entity has two options:

• In-order: Accept only segments that arrive in order; any segment that arrives out of order is discarded.

• In-window: Accept all segments that are within the receive window.

Retransmit Policy

TCP maintains a queue of segments that have been sent but not yet acknowledged. The TCP specification states that TCP will retransmit a segment if it fails to receive an acknowledgment within a given time. A TCP implementation may employ one of three retransmission strategies:

• First-only: Maintain one retransmission timer for the entire queue. If an acknowledgment is received, remove the appropriate segment or segments from the queue and reset the timer. If the timer expires, retransmit the segment at the front of the queue and reset the timer.

• Batch: Maintain one retransmission timer for the entire queue. If an acknowledgment is received, remove the appropriate segment or segments from the queue and reset the timer. If the timer expires, retransmit all segments in the queue and reset the timer.

• Individual: Maintain one timer for each segment in the queue. If an acknowledgment is received, remove the appropriate segment or segments from the queue and destroy the corresponding timer or timers. If any timer expires, retransmit the corresponding segment individually and reset its timer.

Acknowledge Policy

When a data segment arrives that is in sequence, the receiving TCP entity has two options concerning the timing of acknowledgment:

• Immediate: When data are accepted, immediately transmit an empty (no data) segment containing the appropriate acknowledgment number.

• Cumulative: When data are accepted, record the need for acknowledgment, but wait for an outbound segment with data on which to piggyback the acknowledgment.