Chapter 7

The Transport Layer

7.1 The User Datagram Protocol (UDP)

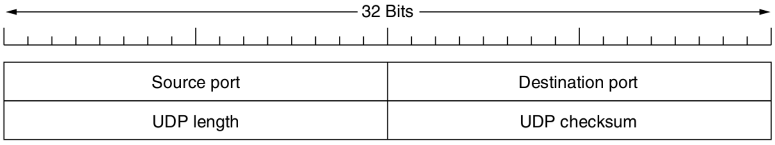

Figure 7.1: The UDP Header (Courtesy: Computer Networks by Tanenbaum, 5th Ed.)

- The UDP header is shown in figure 7.1
- UDP is a connection-less transport protocol.
- UDP transmits segments consisting of an 8-byte header followed by the payload (data).
- The two ports serve to identify the end points within the source and the destination machines. When a UDP packet arrives, it's payload is handed to the process attached to the destination port.
- The source port is primarily needed when a reply must be sent back to the source.
- The UDP length field includes the 8-byte header and the data.
- The UDP checksum is optional.
- UDP *does not do:*
  - Flow Control.
  - Error Control.
  - Re-transmission.
- UDP *does:*
  - Provide an interface to the IP protocol with the added feature of demultiplexing multiple processes using *ports.*
- Examples of some protocols using UDP:
  - DHCP
  - DNS
  - RPC
  - RTSP

7.2 The Transmission Control Protocol (TCP)

- TCP is designed to provide a reliable end-to-end byte stream over an unreliable internetwork.
- TCP service is obtained by both the sender and the receiver creating end-points called *sockets*.
- Each socket has a socket number (address) consisting of the IP address of the host and a 16-bit number local to that host, called a port.
- For TCP service to be obtained, a connection must be explicitly established between a socket on the sending and the receiving machines.
- A socket may be used for multiple connections at the same time.
- Port numbers below 1024 are called *well-known ports* and are reserved for standard services like: port 21 for FTP, 22 for SSH, 25 for SMTP, 80 for HTTP, and so on.
- All TCP connections are full duplex and point-to-point (point-to-point means each connection has exactly two end points).
- TCP does not support multicasting or broadcasting.
- A TCP connection is a byte stream, not a message stream.
- When an application passes data to TCP, TCP may send it immediately or buffer it.
- TCP service uses a *URGENT* flag, which it uses to stop accumulating data and transmit everything it has for that connection immediately.

7.2.1 The TCP Header

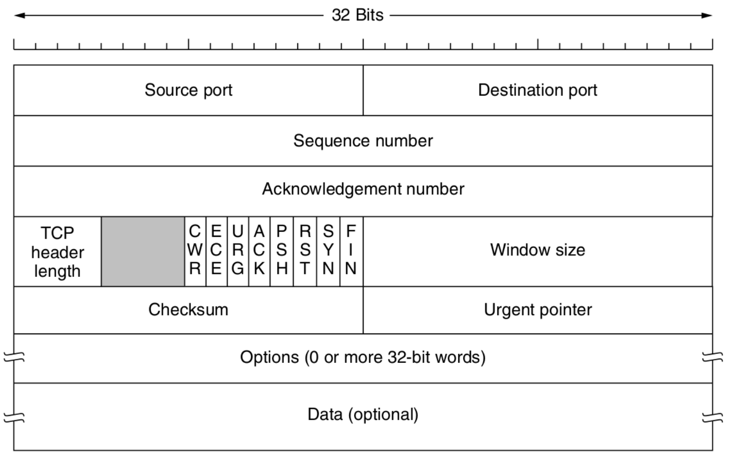

Figure 7.2: The TCP Header (Courtesy: Computer Networks by Tanenbaum, 5th Ed.)

- The TCP header is shown in figure 7.2
- The sending and the receiving TCP entities exchange data in the form of segments.
- A TCP segment consists of a fixed 20-byte header (plus an optional part) followed by zero or more data bytes.
- Each segment, including the TCP header, must fit in the 65,515 byte IP payload.
- Each network has a *maximum transfer unit* or MTU, and each segment must fit in the MTU.

- The various fields of the header has the following meaning:
  - *Source & Destination Ports:* Identify the local end points of the connection. A TCP port plus it's host's IP address forms a 48-bit unique end-point. The source and the destination end-points together identify the connection.
  - *Sequence Number & Acknowledgment Number:* Usual functions.
  - *TCP Header Length:* Total size of the header in 32-bit words.
  - *CWR*[1] *& ECE*[2]*:* Used to signal congestion when ECN[3] is used.
  - *URG:* Is set to 1, if "Urgent pointer" is in use (rarely used).
  - *ACK:* Set to 1 to indicate that the "Acknowledgment number" is valid (true for most of the packets).
  - *PSH:* Indicates PUSH-ed data, i.e. do not buffer the data but deliver it immediately upon arrival.
  - *RST:* Reset a connection, in case of problem.
  - *SYN:* Used to establish a connection.
  - *FIN:* Used to release a connection.
  - *Window size:* For handling flow control using variable-sized sliding window.
  - *Checksum:* For error checking.
  - *Urgent pointer:* Indicates a byte offset from the current sequence number at which urgent data are to be found.
  - *Options:* Provides for a way to add extra facilities not covered by the regular header. Some of the option names are:
    - Maximum Segment Size (MSS).
    - Window scale.
    - Timestamp.
    - Selective Acknowledgment (SACK).

7.2.2 TCP Connection Establishment

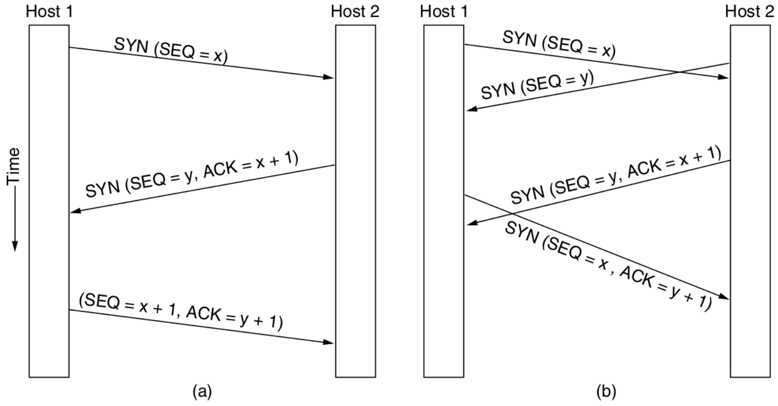
Host 1
Host 2
SYN (SEQ = x)
Time
SYN (SEQ = y, ACK = x + 1)
(SEQ = x + 1, ACK = y + 1)
(a)
Host 1
Host 2
SYN (SEQ = x)
SYN (SEQ = y)
SYN (SEQ = y, ACK = x + 1)
SYN (SEQ = x, ACK = y + 1)
(b)

Figure 7.3: TCP Connection Establishment (Courtesy: Computer Networks by Tanenbaum, 5th Ed.)

- The TCP connection establishment is also known as *TCP 3-way handshake* (SYN, SYN-ACK, ACK).

[1] Congestion Window Reduced

[2] ECN-Echo

[3] Explicit Congestion Notification

  - This mechanism allows two nodes to negotiate the socket parameters before communication starts (figure 7.3(a)).
  - This also allows to negotiate multiple TCP socket connections in both directions at the same time (figure 7.3(b)).

- For setting up the connection:
  - Host A sends a TCP packet to host B with SYN bit set (with sequence number SEQ $= x$).
  - Host B receives A's SYN.
  - Host B sends a SYN-ACK to A (SEQ $= y$, ACK $= x + 1$).
  - Host A receives B's SYN-ACK.
  - Host A sends B a packet with ACK bit set (SEQ $= x + 1$, ACK $= y + 1$).
  - Host B receives A's ACK.
  - TCP socket connection is ESTABLISHED.

7.2.3 TCP Connection Release

- TCP connections are released by each hosts independently of the other i.e. when a connection is closed by one host, that direction is shut-down for new data.
- To release a connection, either party can send a TCP segment with the FIN bit set, which means that it has no more data to transmit.
- When the other party acknowledges the FIN (i.e. FIN-ACK), that direction is shut-down for new data.
- However, data may continue to flow indefinitely in the other direction.
- When both directions have shut-down, the connection is released.
- As such, the connection may be released in a *3-way* fashion (similar to the connection establishment process but using FIN instead of SYN), or in a *4-way* fashion (using independent FIN followed by ACK by both the sides).