Unit 3 (Part-4)

Transport Layer Services: UDP, TCP

Source: Data Communication and Networking by Fourozan

TRANSPORT LAYER SERVICES

The transport layer is responsible for process-to-process delivery—the delivery of a packet, part of a message, from one process to another. Two processes communicate in a client/server relationship, as we will see later.

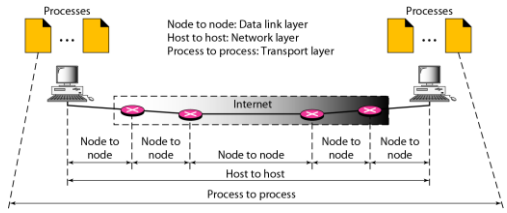
Working Protocols at Transport Layer:

Transmission Control Protocol (TCP)
User Datagram Protocol (UDP)
Transport for Real Time Applications (RTP)

TRANSPORT LAYER SERVICES

- Client/Server Paradigm
  - Process running on local-host called Client.
  - Process running on remote-host called Server.
- For communication, we have:
  - Local host
  - Local process
  - Remote host
  - Remote process

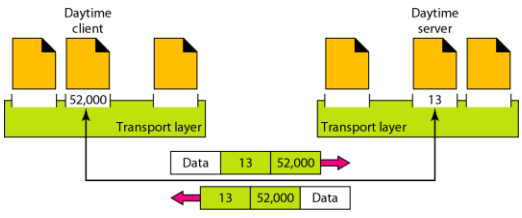
Figure 1 *Types of data deliveries*



TRANSPORT LAYER SERVICES

- Addressing-Port Number (16 bits integer)
- Categories of port numbers:
  - Well-known ports (0-1023)
  - Registered ports (1024 - 49,151)
  - Dynamic ports (49,151- 65,535)

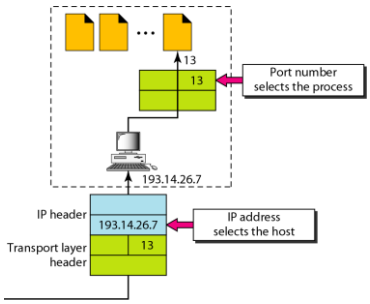
Figure 2 *Port numbers*



TRANSPORT LAYER SERVICES

- IP addresses and port-numbers play different roles in selecting the final destination of the data.
- IP-header contain IP addresses; and UDP/TCP headers contains the port-numbers.

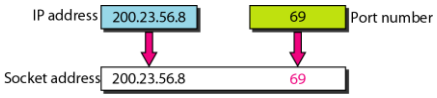
Figure 3 *IP addresses versus port numbers*



TRANSPORT LAYER SERVICES

- ***Socket*** address defines the host-process uniquely.

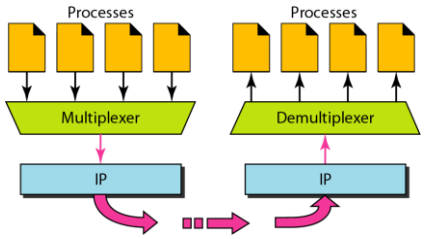
Figure 4 *Socket address*



TRANSPORT LAYER SERVICES

- The addressing mechanism allows multiplexing and demultiplexing by the transport layer.
  - Many-to-one relationship requires multiplexing.
  - One-to-many relationship requires demultiplexing.

Figure 5 *Multiplexing and Demultiplexing*



TRANSPORT LAYER SERVICES

- Connectionless Vs Connection-oriented.
  - No connection establishment process in connectionless for example UDP.
  - Packets follow the same path in connection-oriented for example TCP, SCTP.
- Reliable Vs Unreliable
  - Flow and error control is implemented to get reliable service.
  - No demand of flow and error control in unreliable service.

TRANSPORT LAYER SERVICES

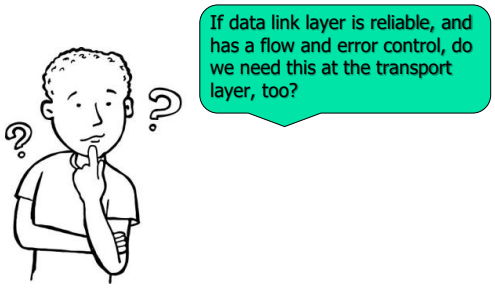
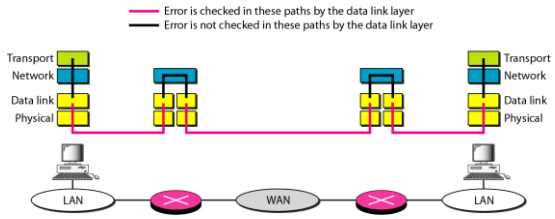


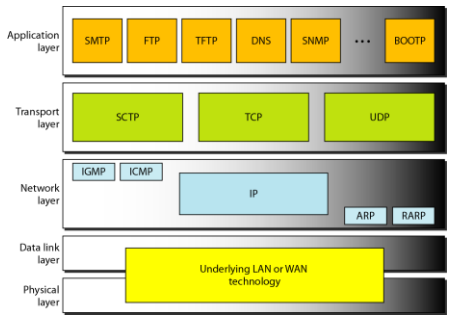
Figure 6 *Error control*



TRANSPORT LAYER SERVICES

- TCP/IP protocol suite provides two protocols for the transport layer:
  - UDP and
  - TCP.
- RTP is new one, which is designed for recently introduced internet-applications.

Figure 7 *Position of UDP, TCP, and SCTP in TCP/IP suite*



USER DATAGRAM PROTOCOL

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.

USER DATAGRAM PROTOCOL

Why UDP?:

Less overhead
Less interaction

Table 1 *Well-known ports used with UDP*

Port	Protocol	Description
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

Example 1

In UNIX, the well-known ports are stored in a file called /etc/services. Each line in this file gives the name of the server and the well-known port number. We can use the grep utility to extract the line corresponding to the desired application. The following shows the port for FTP. Note that FTP can use port 21 with either UDP or TCP.

```
$ grep    ftp    /etc/services
ftp       21/tcp
ftp       21/udp
```

Example 1 (Continued…)

SNMP uses two port numbers (161 and 162), each for a different purposes.

```
$ grep      snmp /etc/services
snmp        161/tcp     #Simple Net  Mgmt Proto
snmp        161/udp     #Simple Net  Mgmt Proto
snmptrap    162/udp     #Traps for SNMP
```

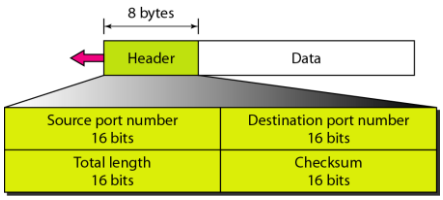
USER DATAGRAM PROTOCOL

The User Datagram is UDP packet, which has fixed-size header of 8 bytes.

UDP datagram header has following 4 fields:

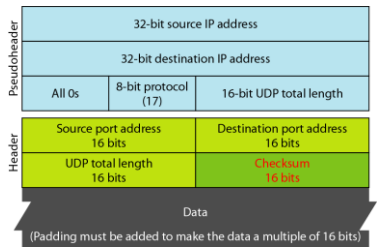
- **Source port-number**
- **Destination port-number**
- **Length**
- **Checksum**

Figure 8 *User datagram format*



UDP length = IP length – IP header's length

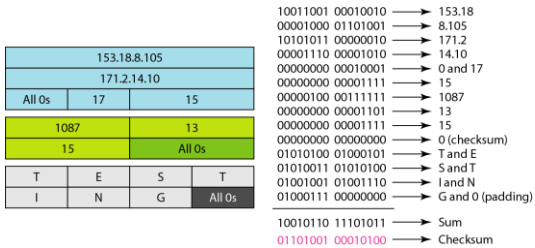
Figure 9 *Pseudoheader for checksum calculation*



Example2

Figure 10 shows the checksum calculation for a very small user datagram with only 7 bytes of data. Because the number of bytes of data is odd, padding is added for checksum calculation. The pseudoheader as well as the padding will be dropped when the user datagram is delivered to IP.

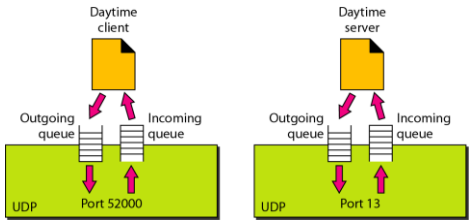
Figure 10 ***Checksum calculation of a simple UDP user datagram***



USER DATAGRAM PROTOCOL

- UDP Operations:
  - Connectionless services
  - Flow and Error control
  - Encapsulation and Decapsulation
  - Queuing

Figure 11 ***Queues in UDP***



USER DATAGRAM PROTOCOL

- Uses of UDP:
  - UDP is suitable for a process that requires simple request-response communication with little concern of flow and error control.
  - UDP is suitable for multicasting .
  - It is used for management processes such as SNMP.
  - It is used for some route updating protocols such as RIP.

Transmission Control Protocol

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPs to send data. In addition, TCP uses flow and error control mechanisms at the transport level.

Topics discussed in this section:

TCP Services
TCP Features
Segment
A TCP Connection
Flow Control
Error Control

Transmission Control Protocol

- TCP Services
  - Process to process communication
  - Stream delivery
  - Full-duplex communication
  - Connection-oriented
  - Reliable

Table 2 *Well-known ports used by TCP*

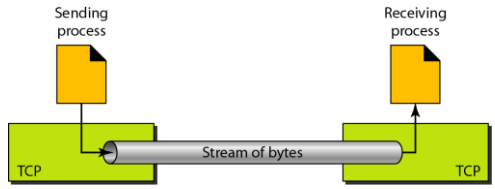
Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
20	FTP, Data	File Transfer Protocol (data connection)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	Bootstrap Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

Transmission Control Protocol

- Stream delivery service

TCP allows the sending process to deliver data as a stream of bytes and allows a receiving process to obtain data as a stream of bytes.

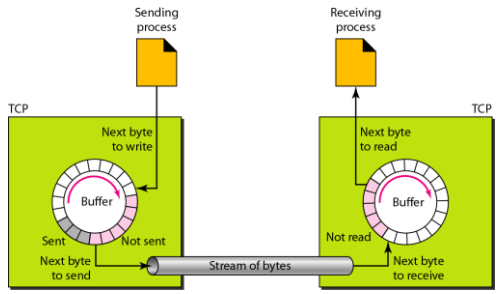
Figure 12 *Stream delivery*



TRANSMISSION CONTROL PROTOCOL

Do we need buffers to tackle the speed mismatch problem between the processes?

Figure 13 *Sending and receiving buffers*



TRANSMISSION CONTROL PROTOCOL

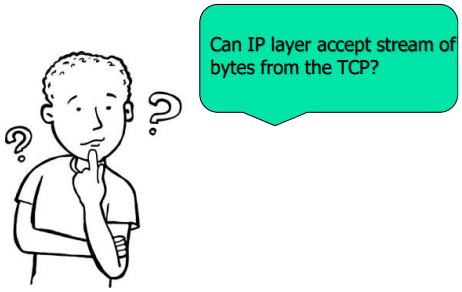
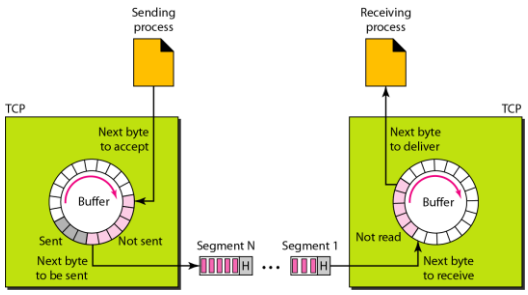
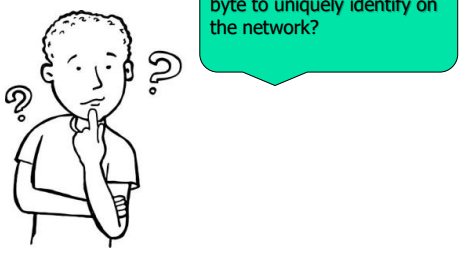


Figure 14 *TCP segments*



TRANSMISSION CONTROL PROTOCOL



Transmission Control Protocol

TCP Features

- *Numbering System*

  The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

  It generates a random number between 0 and 2^{32} - 1 for the number of first byte.

Example 3

The following shows the sequence number for each segment:

Segment 1	➡	Sequence Number: 10,001 (range: 10,001 to 11,000)
Segment 2	➡	Sequence Number: 11,001 (range: 11,001 to 12,000)
Segment 3	➡	Sequence Number: 12,001 (range: 12,001 to 13,000)
Segment 4	➡	Sequence Number: 13,001 (range: 13,001 to 14,000)
Segment 5	➡	Sequence Number: 14,001 (range: 14,001 to 15,000)

Transmission Control Protocol

TCP Features:

- *Sequence Number*

  The value in the sequence number field of a segment defines the number of the first data byte contained in that segment.

- *Acknowledgement Number*

  The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive. The acknowledgment number is cumulative.

Transmission Control Protocol

TCP Features:

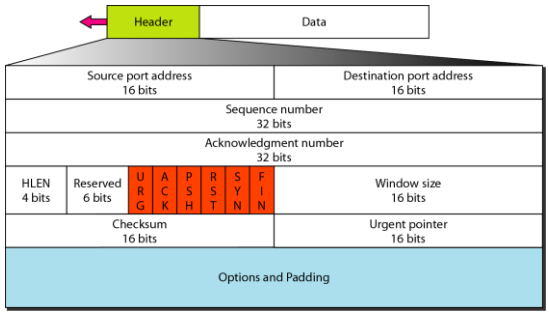
- *Flow Control*
  Numbering system allows TCP to use a byte-oriented flow control.
- *Error Control*
  To provide the reliable service, TCP implements an error control mechanism.
- *Congestion Control*
  Buffering also handles the congestion in the network.

Transmission Control Protocol

Segment format:

- *Segment consists of 20-60 bytes header.*
- *Header contains following fields:*
  - *Source port address*
  - *Destination port address*
  - *Sequence number*
  - *Acknowledge number*
  - *Header length*
  - *Reserved*
  - *Control*
  - *Window size*
  - *Checksum, Urgent pointer, Option*

Figure 15 ***TCP segment format***



Transmission Control Protocol

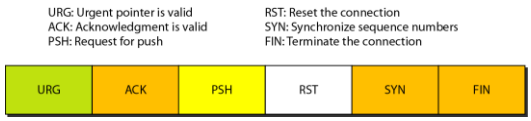
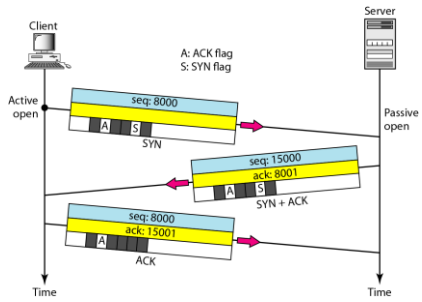


Figure 16 ***Control field***

Table 3 ***Description of flags in the control field***

Flag	*Description*
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledgment field is valid.
PSH	Push the data.
RST	Reset the connection.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

Figure 17 ***Connection establishment using three-way handshaking***



Transmission Control Protocol

- ***TCP Connection:***
  - ***A SYN segment cannot carry data, but it consumes one sequence number.***
  - ***A SYN + ACK segment cannot carry data, but does consume one sequence number.***
  - ***An ACK segment, if carrying no data, consumes no sequence number.***
- ***Simultaneous open***
- ***SYN Flooding***

Figure 18 *Data transfer*

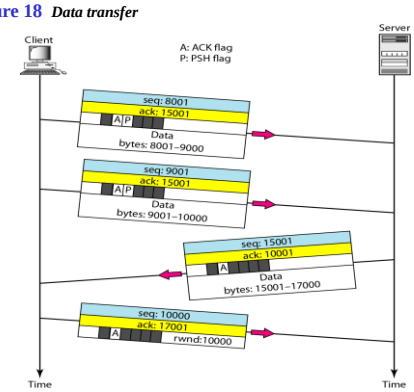
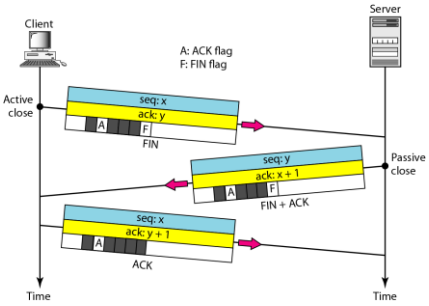


Figure 19 *Connection termination using three-way handshaking*

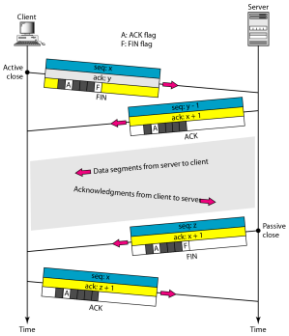


Transmission Control Protocol

Note:

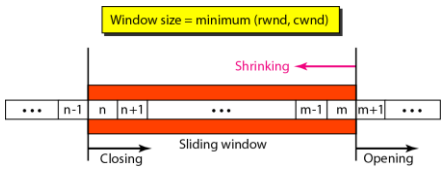
- ***The FIN segment consumes one sequence number if it does not carry data.***
- ***The FIN + ACK segment consumes one sequence number if it does not carry data.***

Figure 20 *Half-close*



Transmission Control Protocol

Figure 21 *Sliding window*



Transmission Control Protocol

- *A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data. TCP sliding windows are byte-oriented.*

Example 4

What is the value of the receiver window (rwnd) for host A if the receiver, host B, has a buffer size of 5000 bytes and 1000 bytes of received and unprocessed data?

Solution

The value of rwnd = 5000 − 1000 = 4000. Host B can receive only 4000 bytes of data before overflowing its buffer. Host B advertises this value in its next segment to A.

Example 5

What is the size of the window for host A if the value of rwnd is 3000 bytes and the value of cwnd is 3500 bytes?

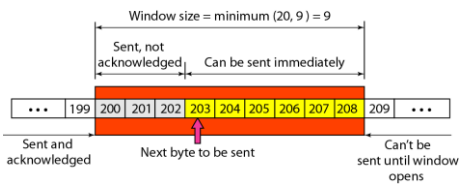
Solution

The size of the window is the smaller of rwnd and cwnd, which is 3000 bytes.

Example 6

Figure 21 shows an unrealistic example of a sliding window. The sender has sent bytes up to 202. We assume that cwnd is 20 (in reality this value is thousands of bytes). The receiver has sent an acknowledgment number of 200 with an rwnd of 9 bytes (in reality this value is thousands of bytes). The size of the sender window is the minimum of rwnd and cwnd, or 9 bytes. Bytes 200 to 202 are sent, but not acknowledged. Bytes 203 to 208 can be sent without worrying about acknowledgment. Bytes 209 and above cannot be sent.

Figure 21 ***Example 6***



Transmission Control Protocol

Some points about TCP sliding windows:

- The size of the window is the lesser of rwnd and cwnd.
- The source does not have to send a full window's worth of data.
- The window can be opened or closed by the receiver, but should not be shrunk.
- The destination can send an acknowledgment at any time as long as it does not result in a shrinking window.
- The receiver can temporarily shut down the window; the sender, however, can always send a segment of 1 byte after the window is shut down.

Transmission Control Protocol

Some points about TCP sliding windows:

- ACK segments do not consume sequence numbers and are not acknowledged.
- In modern implementations, a retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived.
- No retransmission timer is set for an ACK segment.
- Data may arrive out of order and be temporarily stored by the receiving TCP, but TCP guarantees that no out-of-order segment is delivered to the process.
- The receiver TCP delivers only ordered data to the process.

Thank you

Any Query?