Back To Course

Learn

Classroom

Theory

Quiz

Learn Quiz Filter

We have combined Classroom and Theory tab and created a new Learn tab for easy access. You can access Classroom and Theory from the left panel.

## - Transport Layer

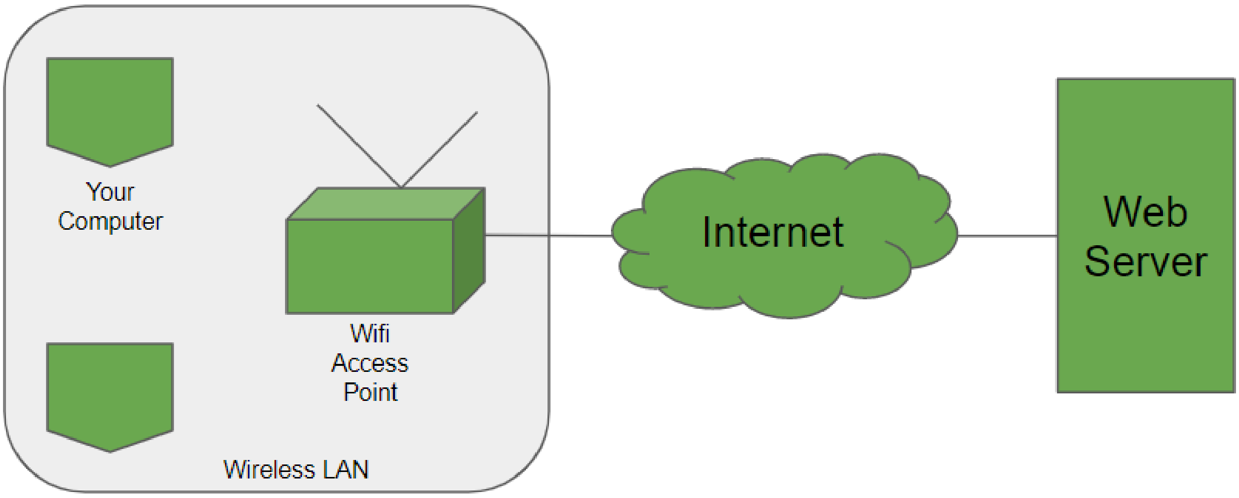
Transport Layer is the layer which lies just above the Network layer and is responsible for **end-to-end connectivity**. It is so-called because it provides point-to-point rather than hop-to-hop. The unit of transmission at the transport layer is called segmentation. **TCP** (Transmission Control Protocol), **UDP** (User Datagram Protocol) and **DCCP** (Datagram Congestion Control Protocol) are some of the protocols running in the transport layer. The transport layer also provides the acknowledgement of the successful data transmission and re-transmits the data if an error is found.

**At sender's side:** Transport layer receives the formatted data from the upper layers, performs **Segmentation** and also implements **Flow & Error control** to ensure proper data transmission. It also adds Source and Destination port number in its header and forwards the segmented data to the Network Layer.

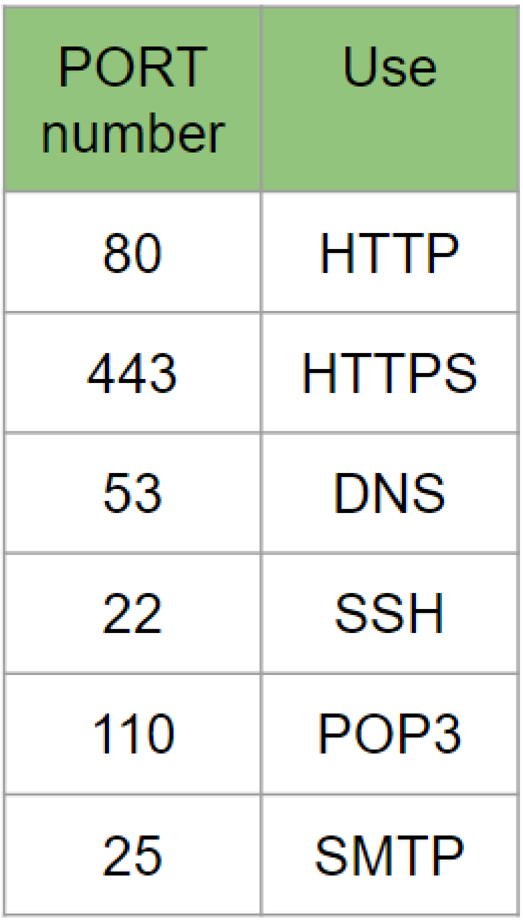
Note: The sender need to know the port number associated with the receiver's application. Generally, this destination port number is configured, either by default or manually. For example, when a web application makes a request to a web server, it typically uses port number 80, because this is the default port assigned to web applications. Many applications have de ▲ t assigned.

**At receiver’s side:** Transport Layer reads the port number from its header and forwards the Data which it has received to the respective application. It also performs sequencing and reassembling of the segmented data.

Let's look at the diagrammatic representation of the working at the transport layer:



Here is a list of few important port numbers and there uses:



| PORT number | Use |
|---|---|
| 80 | HTTP |
| 443 | HTTPS |
| 53 | DNS |
| 22 | SSH |
| 110 | POP3 |
| 25 | SMTP |

Transport Layer has the following responsibilities:

- **Process to process delivery -** While Data Link Layer requires the MAC address (48 bits address contained inside the N Interface Card of every host machine) of source-destination hosts to correctly deliver a frame and Network layer requires the IP address for appropriate routing of packets , in a similar way Transport Layer requires a Port number to correctly deliver the segments of data to the correct process amongst the multiple processes running on a particular host. A **port number** is a 16 bit address used to identify any client-server program uniquely.

- **End-to-end Connection between hosts -** The transport layer is also responsible for creating the end-to-end Connection between hosts for which it mainly uses TCP and UDP. TCP is a secure, connection- orientated protocol which uses a handshake protocol to establish a robust connection between two end- hosts. TCP ensures reliable delivery of messages and is used in various applications. UDP, on the other hand, is a stateless and unreliable protocol which ensures best-effort delivery. It is suitable for the applications which have little concern with flow or error control and requires to send the bulk of data like video conferencing. It is often used in multicasting protocols.

- **Multiplexing and Demultiplexing -** Multiplexing allows simultaneous use of different applications over a network which is running on a host. The transport layer provides this mechanism which enables us to send packet streams from various applications simultaneously over a network. The transport layer accepts these packets from different processes differentiated by their port numbers and passes them to the network layer after adding proper headers. Similarly, Demultiplexing is required at the receiver side to obtain the data coming from various processes. Transport receives the segments of data from the network layer and delivers it to the appropriate process running on the receiver's machine.

- **Congestion Control -** Congestion is a situation in which too many sources over a network attempt to send data and the router buffers start overflowing due to which loss of packets occur. As a result retransmission of packets from the sources increases the congestion further. In this situation, the Transport layer provides Congestion Control in different ways. It uses **open loop** congestion control to prevent the congestion and **closed loop** congestion control to remove the congestion in a network once it occurred. TCP provides AIMD- additive increase multiplicative decrease, leaky bucket technique for congestion control.

- **Data integrity and Error correction -** Transport layer checks for errors in the messages coming from application layer by using error detection codes, computing checksums, it checks whether the received data is not corrupted and uses the ACK and NACK services to inform the sender if the data has arrived or not and checks for the integrity of data.

- **Flow control -** The transport layer provides a flow control mechanism between the adjacent layers of the TCP/IP mode ▲ lso prevents data loss due to a fast sender and slow receiver by imposing some flow control techniques. It uses the method of sliding window protocol which is accomplished by the receiver by sending a window back to the sender informing the size of data it can receive.

## − Differences between TCP and UDP

Prerequisite - Transport Layer responsibilities, TCP, UDP

| Transmission control protocol (TCP) | User datagram protocol (UDP) |
|---|---|
| TCP is a connection-oriented protocol. Connection-orientation means that the communicating devices should establish a connection before transmitting data and should close the connection after transmitting the data. | UDP is the Datagram oriented protocol. This is because there is no overhead for opening a connection, maintaining a connection, and terminating a connection. UDP is efficient for broadcast and multicast type of network transmission. |
| TCP is reliable as it guarantees delivery of data to the destination router. | The delivery of data to the destination cannot be guaranteed in UDP. |
| TCP provides extensive error checking mechanisms. It is because it provides flow control and acknowledgment of data. | UDP has only the basic error checking mechanism using checksums. |
| Sequencing of data is a feature of Transmission Control Protocol (TCP). this means that packets arrive in-order at the receiver. | There is no sequencing of data in UDP. If ordering is required, it has to be managed by the application layer. |
| TCP is comparatively slower than UDP. | UDP is faster, simpler and more efficient than TCP. |
| Retransmission of lost packets is possible in TCP, but not in UDP. | There is no retransmission of lost packets in User Datagram Protocol (UDP). |
| TCP has a (20-80) bytes variable length header. | UDP has a 8 bytes fixed length header. |
| TCP is heavy-weight. | UDP is lightweight. |
| TCP doesn't supports Broadcasting. | UDP supports Broadcasting. |
| TCP is used by HTTP, HTTPs, FTP, SMTP and Telnet. | UDP is used by DNS, DHCP, TFTP, SNMP, RIP, and VoIP. |

A short example to understand the differences clearly :

Suppose there are two houses, H1 and H2 and a letter has to be sent from H1 to H2. But there is a river in between those two houses. Now how can we send the letter?

Solution 1: Make a bridge over the river and then it can be delivered.

Solution 2: Get it delivered through a pigeon.

Consider the first solution as TCP. A connection has to made ( bridge ) to get the data (letter) delivered.

The data is reliable because it will directly reach another end without loss in data or error.

And the second solution is UDP. No connection is required for sending the data.

The process is fast as compare to TCP, where we need to set up a connection(bridge). But the data is not reliable: we don't know whether the pigeon will go in the right direction, or it will drop the letter on the way or some issue is encountered in mid-travel.