INTRODUCTION TO TRANSPORT LAYER

INTRODUCTION:

- The transport layer is located between the application layer and the network layer. It provides a process-to-process communication between two application layers.

Transport-Layer Services:

Process-to-Process Communication:

- The first duty of a transport-layer protocol is to provide **process-to-process communication.**

Addressing: Port Numbers:

- Although there are a few ways to achieve process-to-process communication, the most common is through the **client-server paradigm.**
- A process on the local host, called a *client,* needs services from a process usually on the remote host, called a *server.*
- The local host and the remote host are defined using IP addresses (discussed in Chapter 18). To define the processes, we need second identifiers, called ***port numbers.*** In the TCP/IP protocol suite, the port numbers are integers between 0 and 65,535 (16 bits).
- The client program defines itself with a port number, called the ***ephemeral port number.*** The word *ephemeral* means "short-lived".
- TCP/IP has decided to use universal port numbers for servers; these are called ***well-known port numbers.***

Fig: ICANN ranges.

- ***Well-known ports.*** The ports ranging from 0 to 1023 are assigned and controlled by ICANN. These are the well-known ports.
- ***Registered ports*.** The ports ranging from 1024 to 49,151 are not assigned or controlled by ICANN. They can only be registered with ICANN to prevent duplication.
- ***Dynamic ports*.** The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used as temporary or private port numbers.

Encapsulation and Decapsulation:

- To send a message from one process to another, the transport-layer protocol encapsulates and decapsulates messages.
- When a process has a message to send, it passes the message to the transport layer along with a pair of socket addresses and some other pieces of information, which depend on the transport-layer protocol.
- The transport layer receives the data and adds the transport-layer header. The packets at the transport layer in the Internet are called *user datagrams, segments,* or *packets,* depending on what transport-layer protocol we use.
- Decapsulation happens at the receiver site. When the message arrives at the destination transport layer, the header is dropped and the transport layer delivers the message to the process running at the application layer.
- The sender socket address is passed to the process in case it needs to respond to the message received.

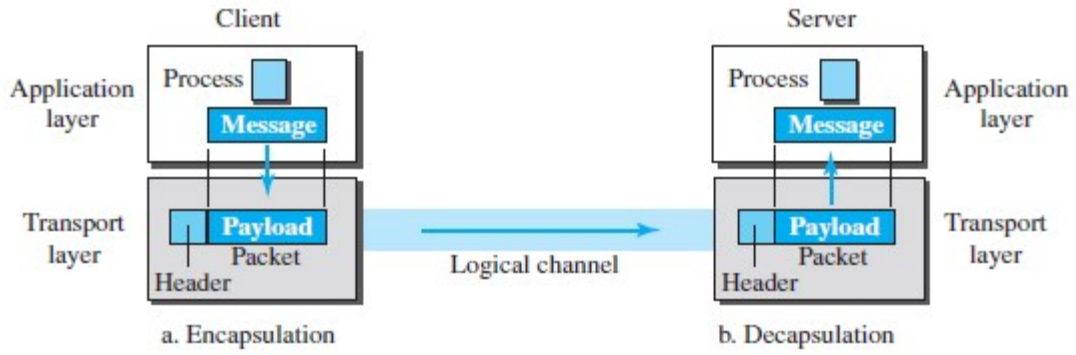

Fig: Encapsulation and decapsulation.

Multiplexing and Demultiplexing:

- Whenever an entity accepts items from more than one source, this is referred to as ***multiplexing*** (many to one); whenever an entity delivers items to more than one source, this is referred to as ***demultiplexing*** (one to many).
- The transport layer at the source performs multiplexing; the transport layer at the destination performs demultiplexing.

Flow Control:

- Whenever an entity produces items and another entity consumes them, there should be a balance between production and consumption rates.
- If the items are produced faster than they can be consumed, the consumer can be overwhelmed and may need to discard some items.
- If the items are produced more slowly than they can be consumed, the consumer must wait, and the system becomes less efficient.
- Flow control is related to the first issue. We need to prevent losing the data items at the consumer site.

Pushing or Pulling:

- Delivery of items from a producer to a consumer can occur in one of two ways: *pushing* or *pulling*.
- If the sender delivers items whenever they are produced--without a prior request from the consumer-- the delivery is referred to as *pushing*.
- If the producer delivers the items after the consumer has requested them, the delivery is referred to as *pulling*.

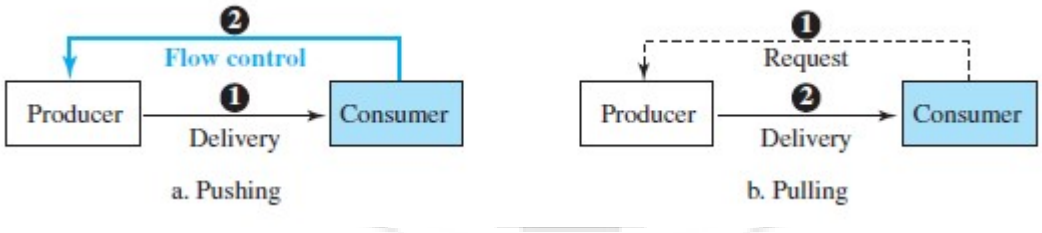

Fig: Pushing or pulling.

When the producer *pushes* the items, the consumer may be overwhelmed and there is a need for flow control.