Department of Computer science and Information Technology

Introduction to Transport Layer

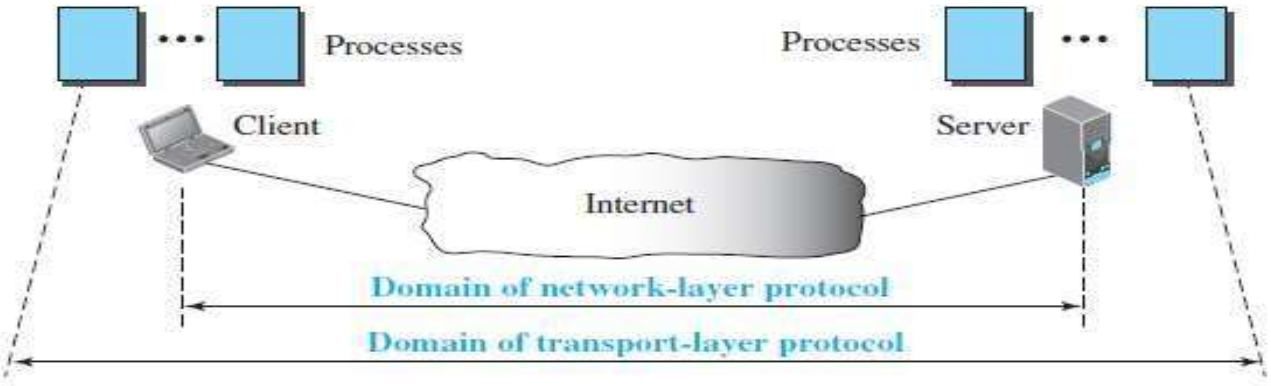
INTRODUCTION

The transport layer is located between the application layer and the network layer. It provides a process-to-process communication between two application layers, one at the local host and the other at the remote host.

Transport-Layer Services

Process-to-Process Communication: The first duty of a transport-layer protocol is to provide process-to-process communication. A process is an application-layer entity (running program) that uses the services of the transport layer.

Figure 23.2 *Network layer versus transport layer*



Addressing: Port Numbers

Although there are a few ways to achieve process-to-process communication, the most common is through the client-server paradigm.

we must define the local host, local process, remote host, and remote process. The local host and the remote host are defined using IP addresses. 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).

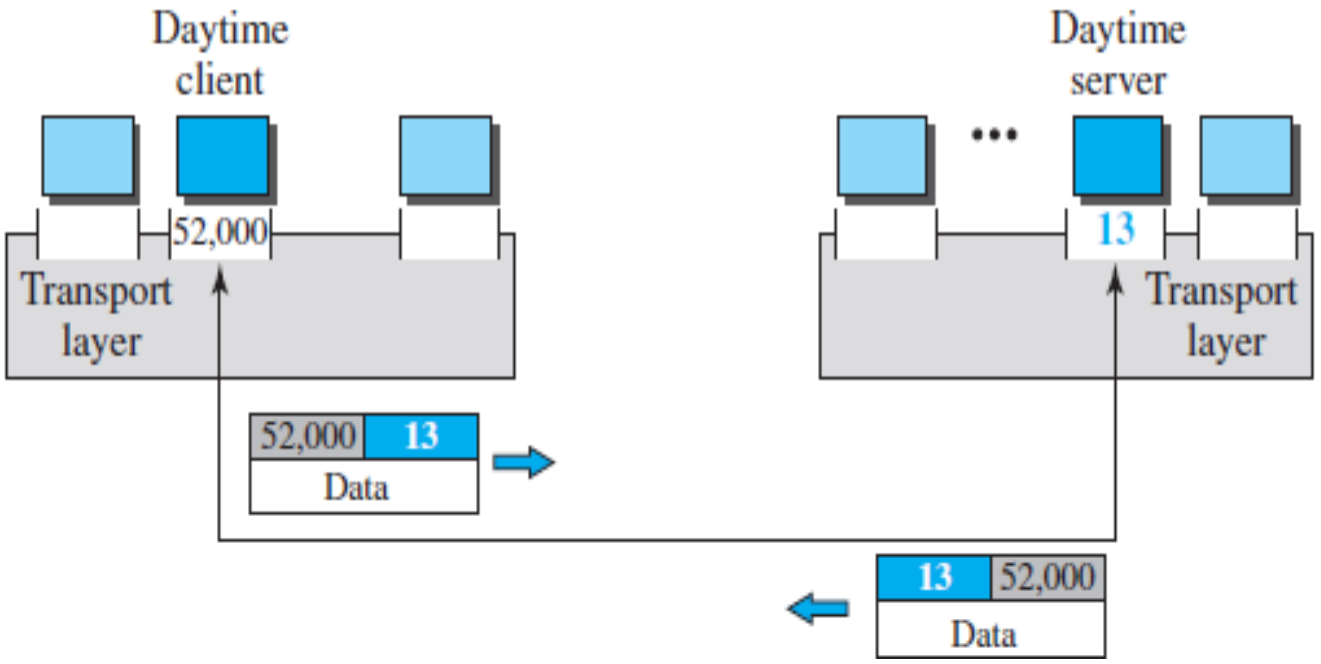
Continue..

The client program defines itself with a port number, called the ephemeral port number. The word ephemeral means "short-lived". An ephemeral port number is recommended to be greater than 1023 for some client/server programs to work properly.

The server process must also define itself with a port number. This port number, however, cannot be chosen randomly.

Continue..

Figure 23.3 *Port numbers*



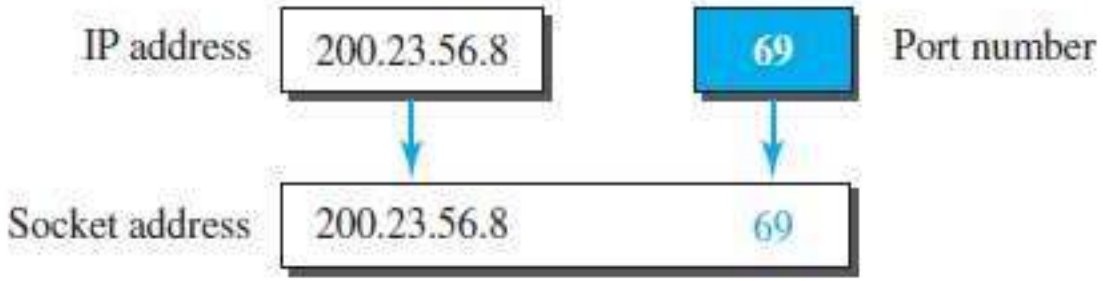
Internet Corporation for Assigned Names and Numbers (ICANN) Ranges

- **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.

Socket Addresses

A transport-layer protocol in the TCP suite needs both the IP address and the port number, at each end, to make a connection. The combination of an IP address and a port number is called a socket address.

Figure 23.6 *Socket address*



Encapsulation and Decapsulation

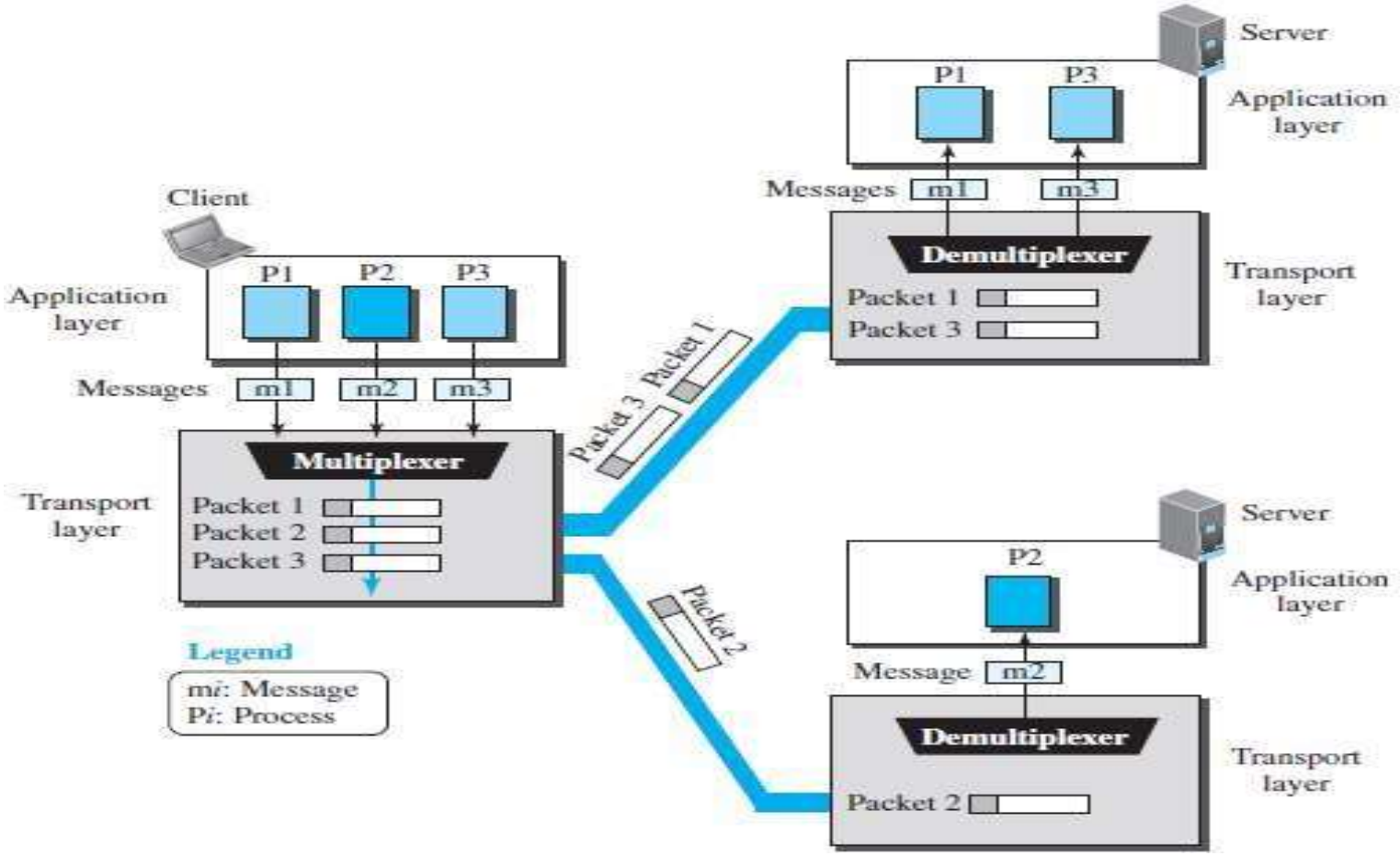
Encapsulation happens at the sender site. When a process has a message to send, it passes the message to the transport layer along with a pair of socket addresses and some other pieces of information, which depend on the transport-layer protocol. The packets at the transport layer in the Internet are called user datagrams, segments, or packets.

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.

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).

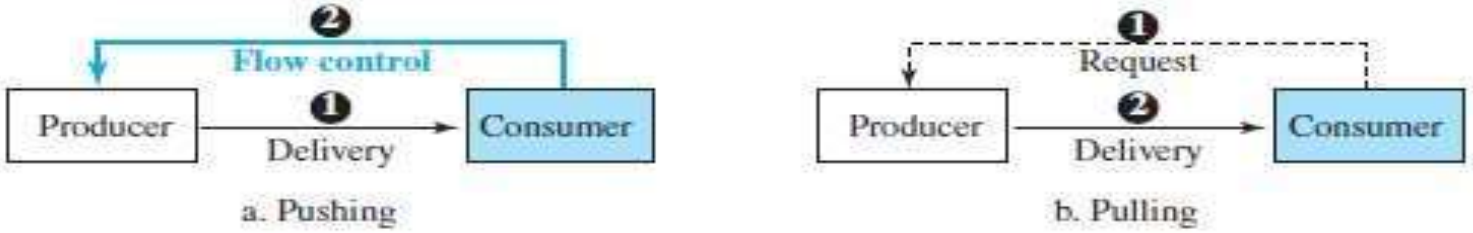
Multiplexing and Demultiplexing



Flow Control

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.

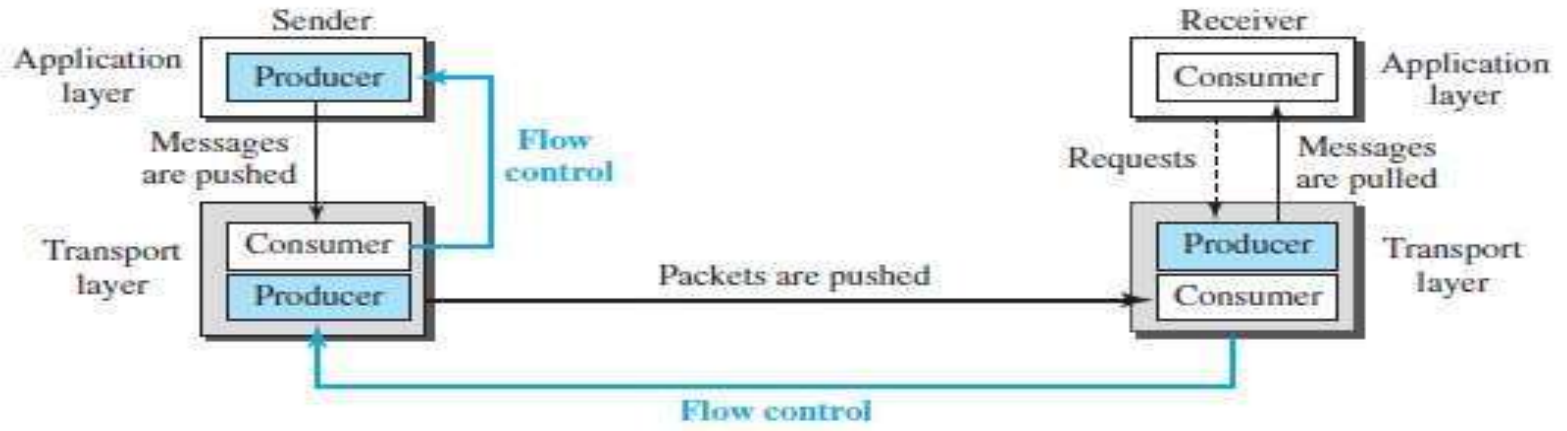
Figure 23.9 *Pushing or pulling*



Flow Control

Flow Control at Transport Layer: In communication at the transport layer, we are dealing with four entities: sender process, sender transport layer, receiver transport layer, and receiver process.

Figure 23.10 *Flow control at the transport layer*



Flow Control

Buffers: A buffer is a set of memory locations that can hold packets at the sender and receiver. When the buffer of the sending transport layer is full, it informs the application layer to stop passing chunks of messages; When the buffer of the receiving transport layer is full, it informs the sending transport layer to stop sending packets.

Error Control

Error control at the transport layer is responsible for

- Detecting and discarding corrupted packets.
- Keeping track of lost and discarded packets and resending them.
- Recognizing duplicate packets and discarding them.
- Buffering out-of-order packets until the missing packets arrive.

Error control, unlike flow control, involves only the sending and receiving transport layers.

Error Control

Sequence Numbers: We can add a field to the transport-layer packet to hold the sequence number of the packet.

When a packet is corrupted or lost, the receiving transport layer can somehow inform the sending transport layer to resend that packet using the sequence number.

If the header of the packet allows m bits for the sequence number, the sequence numbers range from 0 to $2^m - 1$.

Acknowledgment: When a packet is sent, the sender starts a timer. If an ACK does not arrive before the timer expires, the sender resends the packet.

Combination of Flow and Error Control

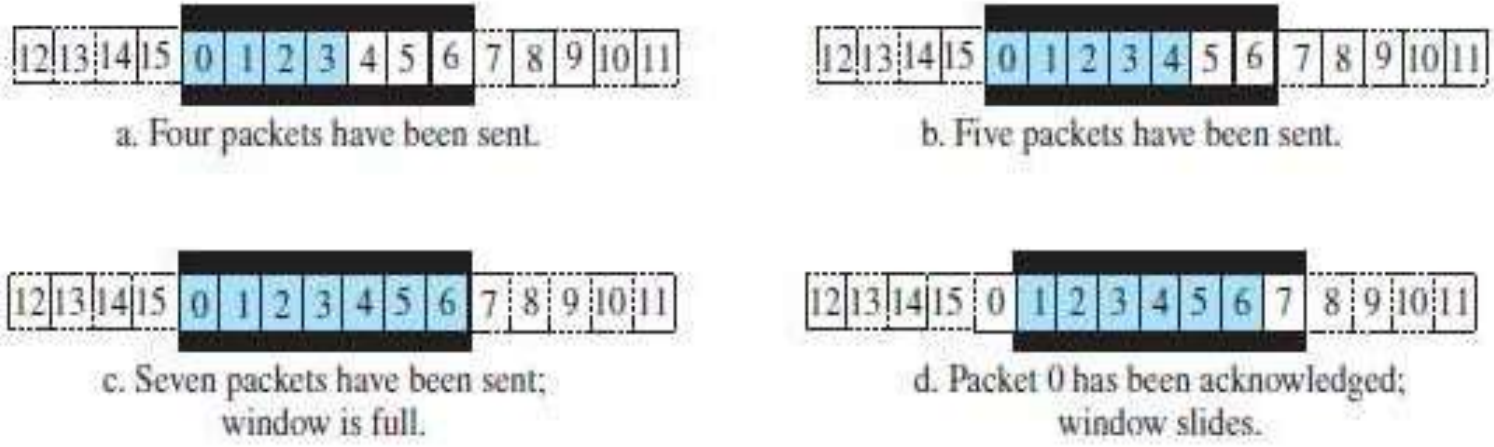
At the sender, when a packet is prepared to be sent, we use the number of the next free location, x, in the buffer as the sequence number of the packet. When the packet is sent, a copy is stored at memory location x, awaiting the acknowledgment from the other end. When an acknowledgment related to a sent packet arrives, the packet is purged and the memory location becomes free.

At the receiver, when a packet with sequence number y arrives, it is stored at the memory location y until the application layer is ready to receive it.

Combination of Flow and Error Control

Sliding Window:

Figure 23.13 *Sliding window in linear format*



Congestion Control

Congestion in a network may occur if the load on the network—the number of packets sent to the network—is greater than the capacity of the network.

Congestion control refers to the mechanisms and techniques that control the congestion and keep the load below the capacity.

Connectionless and Connection-Oriented Protocols

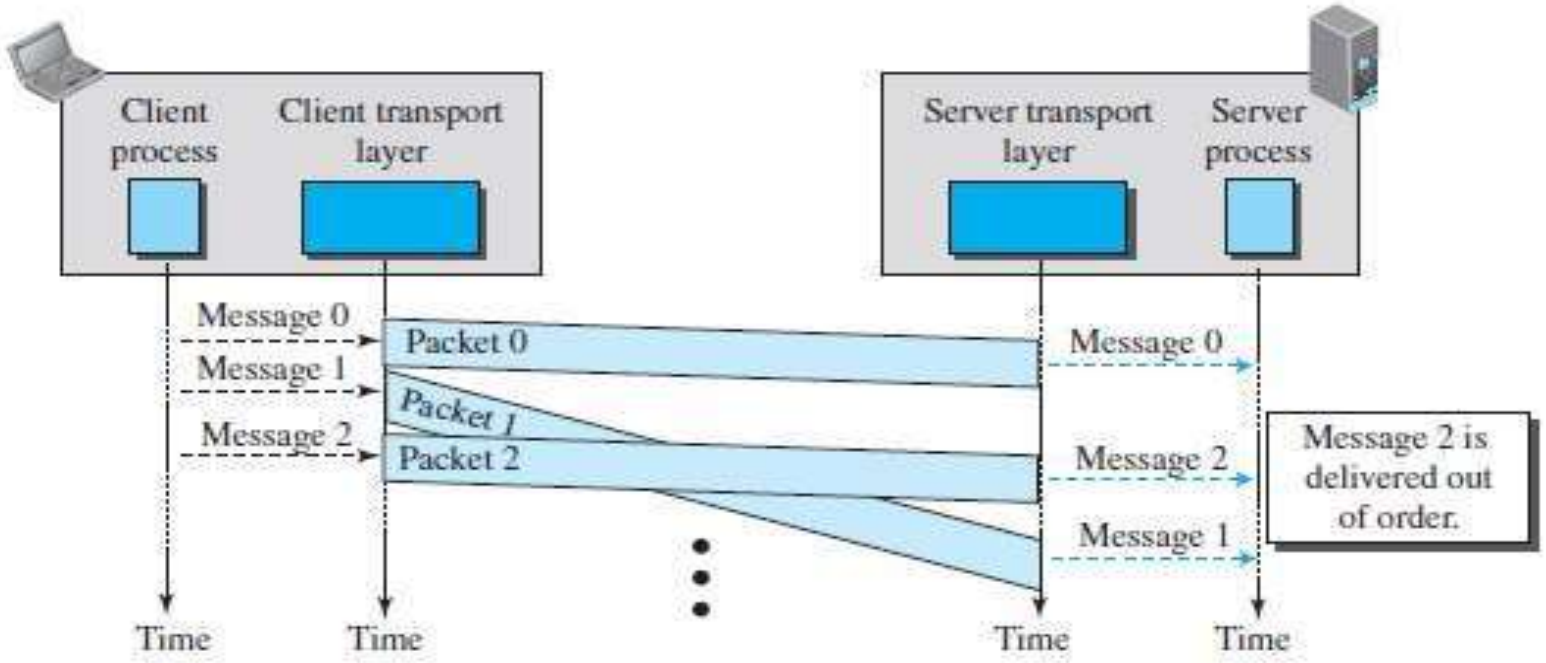
A transport-layer protocol, like a network-layer protocol, can provide two types of services: connectionless and connection-oriented. Connectionless service at the transport layer means independency between packets; connection-oriented means dependency.

Connectionless Service

assume that a client process has three chunks of messages to send to a server process. The chunks are handed over to the connectionless transport protocol in order. However, since there is no dependency between the packets at the transport layer, the packets may arrive out of order at the destination and will be delivered out of order to the server process.

Connectionless Service

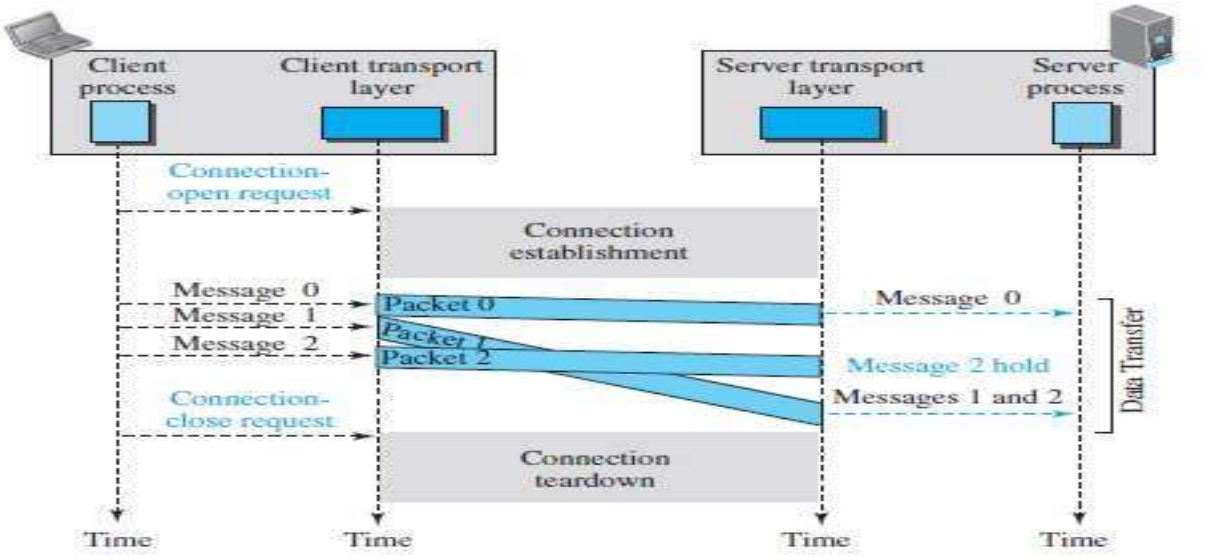
Figure 23.14 *Connectionless service*



Connection-Oriented Service

In a connection-oriented service, the client and the server first need to establish a logical connection between themselves.

Figure 23.15 *Connection-oriented service*



References

- Data Communications and Networking, Behrouz A. Forouzan, Fifth Edition, TMH, 2013.