UNIT 5 APPLICATION LAYER

5.1 APPLICATION LAYER PARADIGMS

The layer where all the applications are found is called Application layer.The traditional paradigm is called the client-server paradigm. In this paradigm, the service provider is an application program, called the server process; it runs continuously, waiting for another application program, called the client process, to make a connection through the Internet and ask for service.

Normally few server processes are available that can provide a specific type of service, but there are many clients that request service from any of these server processes.The server process must be running all the time; the client process is started when the client needs to receive service.

For example, a telephone directory center in any area can be a server; a subscriber that calls and asks for a specific telephone number can be thought of as a client.

The directory center must be ready and available all the time; the subscriber can call the center for a short period when the service is needed. Figure5.1.1 shows an example of a client-server communication in which three clients communicate with one server to receive the services provided by this server.

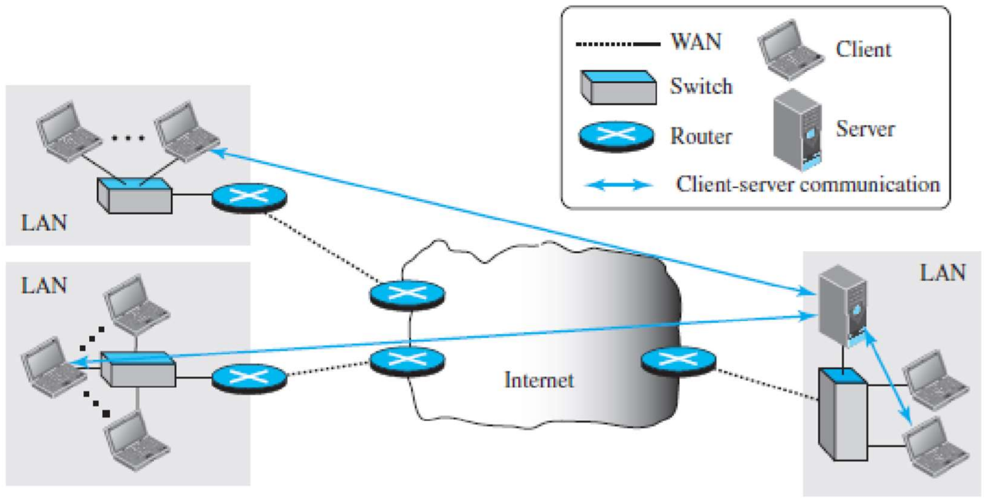

Fig5.1.1:Client server paradigm.

[Source :"Data Communications and Networking" by Behrouz A. Forouzan,Page-821]

Peer-to-Peer paradigm

In this paradigm, there is no need for a server process to be running all the time and waiting for the client processes to connect.The responsibility is shared between peers. A computer connected to the Internet can provide service at one time and receive service at another time. A computer can even provide and receive services at the same time.Figure 5.1.2 shows an example of communication in this paradigm.

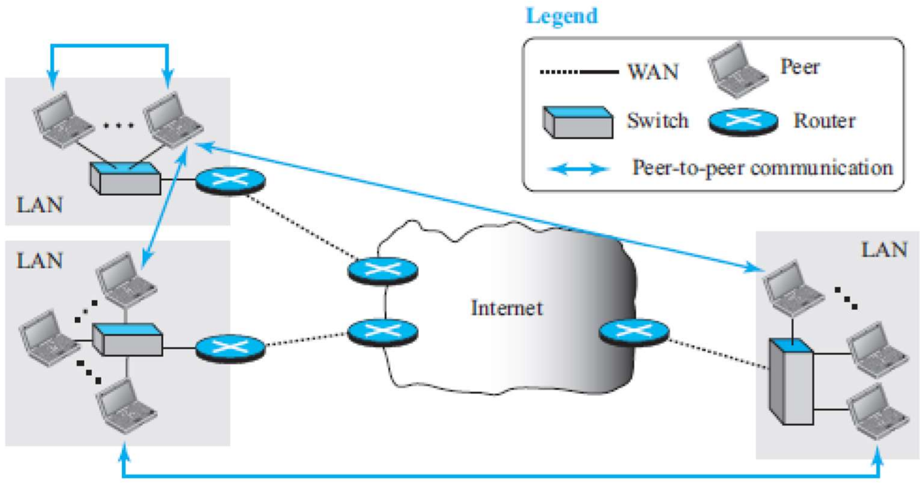

Fig5.1.2: Peer to peer paradigm.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-822]

Communication by phone is a peer-to-peer activity; no party needs to wait for the other party to call.The peer-to-peer paradigm can be used in a situation, when some computers connected to the Internet have something to share with each other.For example, if an Internet user has a file available to share with other Internet users, there is no need for the file holder to become a server and run a server process all the time waiting for other users to connect and to get the file.

Client-Server Programming

In a client-server paradigm, communication at the application layer is between two running application programs called processes: a client and a server. A client is a running program that initializes the communication by sending a request; a server is another application program that waits for a request from a client.The server handles the request received from a client, prepares a result, and sends the result back to the client.

The lifetime of a server is infinite: it should be started and run forever, waiting for the clients. The lifetime of a client is finite. It sends a finite number of requests to the corresponding server, receives the responses, and stops.

Application Programming Interface

If we need a process to be able to communicate with another process, we need a new set of instructions to tell the lowest four layers of the TCP/IP suite to open the connection, send and receive data from the other end, and close the connection.A set of instructions of this type is called as an application programming interface (API).An interface in programming is a set of instructions between two entities. In this case, one of the entities is the process at the application layer and the other is the operating system that encapsulates the first four layers of the TCP/IP protocol suite.

A computer manufacturer build the first four layers of the suite in the operating system and include an API. Here, the processes running at the application layer are able to communicate with the operating system when sending and receiving messages through the Internet.

Several APIs have been designed for communication. Three are common: socket interface, Transport Layer Interface (TLI), and STREAM.

Socket Interface

Socket interface started in the early 1980s at UC Berkeley as part of a UNIX environment.

The socket interface is a set of instructions that provide communication between the application layer and the operating system. It is a set of instructions that can be used by a process to communicate with another process. For example, in most computer languages, like C, C++, or Java, we have several instructions that can read and write data to other sources and sinks such as a keyboard (a source), a monitor (a sink), or a file (source and sink). Figure 5.1.3 shows the socket format.

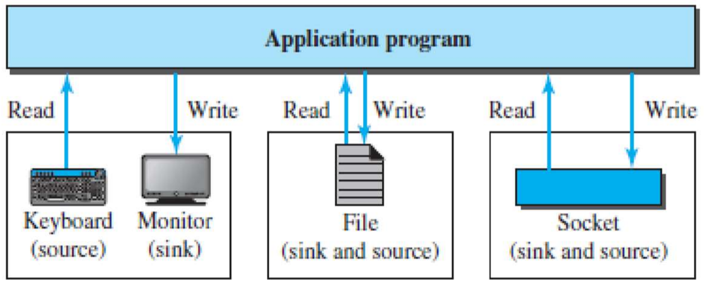

Fig5.1.3: Socket format.

[Source :"Data Communications and Networking" by Behrouz A. Forouzan,Page-824]

In application layer, communication between a client process and a server process is the communication between two sockets.As far as the application layer is concerned, communication between a client process and a server process is communication between two sockets, created at two ends, as shown in Figure 5.1.4. The client thinks that the socket is the entity that receives the request and gives the response; the server thinks that the socket is the one that has a request and needs the response. If we create two sockets, one at each end, and define the source and destination addresses correctly, we can use the available instructions to send and receive data. The rest is the responsibility of the operating system and the embedded TCP/IP protocol.

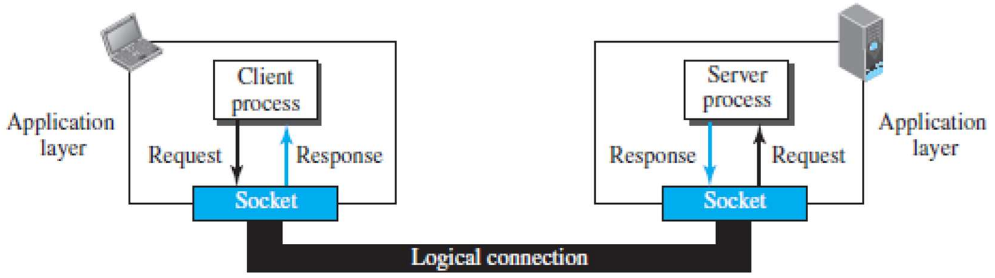

Fig5.1.4: Socket in process to process communication.
[Source :"Data Communications and Networking" by Behrouz A. Forouzan,Page-825]

Socket Addresses

The interaction between a client and a server is two-way communication. In a two-way communication, we need a pair of addresses: local (sender) and remote (receiver). The local address in one direction is the remote address in the other direction and vice versa.Since communication in the client-server paradigm is between two sockets, we need a pair of socket addresses for communication: a local socket address and a remote socket address.A socket address should first define the computer on which a client or a server is running. A computer in the Internet is defined by its IP address. a socket address should be a combination of an IP address and a port number as shown in Figure 5.1.5.

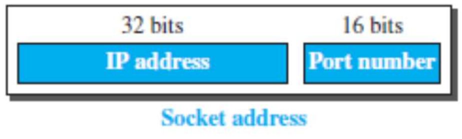

Fig5.1.5: A Socket address.
[Source :"Data Communications and Networking" by Behrouz A. Forouzan,Page-825]

Finding Socket Addresses

To find socket address, the situation is different for each site.

Local Socket Address The local (server) socket address is provided by the operating system. The operating system knows the IP address of the computer on which the server process is running.

The port number of a server process, however, needs to be assigned. If the server process is a standard one defined by the Internet authority, a port number is already assigned to it. For example, the assigned port number for a Hypertext Transfer Protocol (HTTP) is the integer 80, which cannot be used by any other process.

If the server process is not standard, the designer of the server process can choose a port number, in the range defined by the Internet authority, and assign it to the process. When a server starts running, it knows the local socket address.

Remote Socket Address

The remote socket address for a server is the socket address of the client that makes the connection. Since the server can serve many clients, it does not know previously, the remote socket address for communication.The server can find this socket address when a client tries to connect to the server. The client socket address, which is contained in the request packet sent to the server, becomes the remote socket address that is used for responding to the client.

ROHINI COLLEGE OF ENGINEERING AND TECHNOLOGY PALKULAM, KANYAKUMARI OBSERVE OPTIMIZE OUTSPREAD