

Name : Ansari M.Saeem M.Saleem

Uid : 2019430001

Subject : NAD

Expt no : 1

Aim : Write a program to transfer the content of a requested file from server to client using TCP/IP sockets.

Date Of Performance : 14/01/2020

Aim:

Write a program to transfer the content of a requested file from server to client using TCP/IP sockets.

Objectives:

- To Create and Bind sockets to provide end to end data transfer from client to server.
- To use socket in TCP/IP for transfer of requested file.
- To securely transfer data packets from client to server.

Theory:

The Internet protocol suite is the conceptual model and set of communications protocols used in the Internet and similar computer networks. It is commonly known as TCP/IP because the foundational protocols in the suite are the Transmission Control Protocol (TCP) and the Internet Protocol (IP). During its development, versions of it were known as the Department of Defense (DoD) model because the development of the networking method was funded by the United States Department of Defense through DARPA.

There are four layers in TCP/IP Protocol as follows :

1. Network Access Layer :

It looks out for hardware addressing and the protocols present in this layer allows for the physical transmission of data. This layer corresponds to the combination of Data Link Layer and Physical Layer of the OSI model.

2. Internet Layer :

It defines the protocols which are responsible for logical transmission of data over the entire network. The main protocols residing at this layer are :

- I. **IP** – stands for Internet Protocol and it is responsible for delivering packets from the source host to the destination host by looking at the IP addresses in the packet headers. IP has 2 versions: IPv4 and IPv6. IPv4 is the one that most of the websites are using currently. But IPv6 is growing as the number of IPv4 addresses are limited in number when compared to the number of users.
- II. **ICMP** – stands for Internet Control Message Protocol. It is encapsulated within IP datagrams and is responsible for providing hosts with information about network problems.
- III. **ARP** – stands for Address Resolution Protocol. Its job is to find the hardware address of a host from a known IP address. ARP has several types: Reverse ARP, Proxy ARP, Gratuitous ARP and Inverse ARP.

3. Host-to-Host Layer :

It is responsible for end-to-end communication and error-free delivery of data. It shields the upper-layer applications from the complexities of data. This layer is analogous to the transport layer of the OSI model. The two main protocols present in this layer are :

- I. **Transmission Control Protocol (TCP)** – It is known to provide reliable and error-free communication between end systems. It performs sequencing and segmentation of data. It also has acknowledgment feature and controls the flow of the data through flow control mechanism. It is a very effective protocol but has a lot of overhead due to such features. Increased overhead leads to increased cost.
- II. **User Datagram Protocol (UDP)** – On the other hand does not provide any such features. It is the go-to protocol if your application does not require reliable transport as it is very cost-effective. Unlike TCP, which is connection-oriented protocol, UDP is connectionless.

4. Process Layer

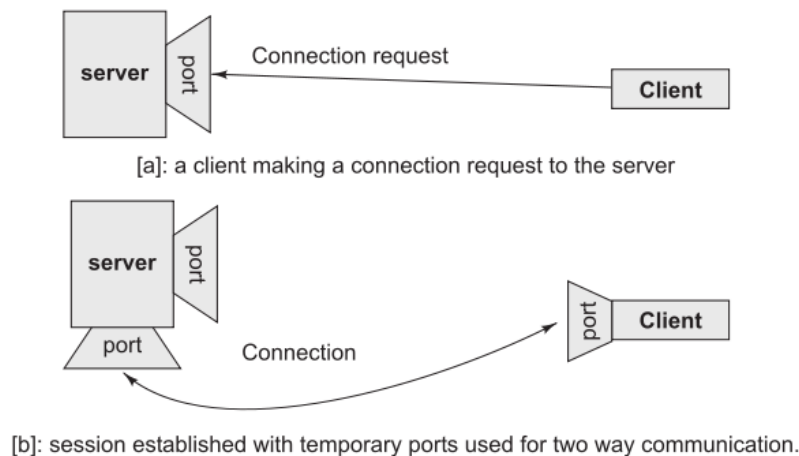
It is responsible for node-to-node communication and controls user-interface specifications. Some of the protocols present in this layer are: HTTP, HTTPS, FTP, TFTP, Telnet, SSH, SMTP, SNMP, NTP, DNS, DHCP, NFS, X Window, LPD. Have a look at Protocols in Application Layer for some information about these protocols. This layer performs the functions of top three layers of the OSI model: Application, Presentation and Session Layer.

Methodology:

Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request. On the client-side: The client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection. This is usually assigned by the system.

If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to the same local port and also has its remote endpoint set to the address and port of the client. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client.

On the client side, if the connection is accepted, a socket is successfully created and the client can use the socket to communicate with the server. The client and server can now communicate by writing to or reading from their sockets.



Steps:

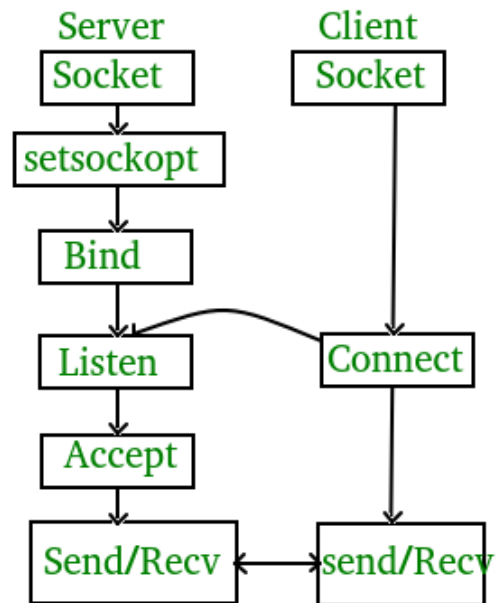
The entire process can be broken down into following steps:

TCP Server –

1. using `create()`, Create TCP socket.
2. using `bind()`, Bind the socket to server address.
3. using `listen()`, put the server socket in a passive mode, where it waits for the client to approach the server to make a connection
4. using `accept()`, At this point, connection is established between client and server, and they are ready to transfer data.
5. Go back to Step 3.

TCP Client –

1. Create TCP socket.
2. connect newly created client socket to server.

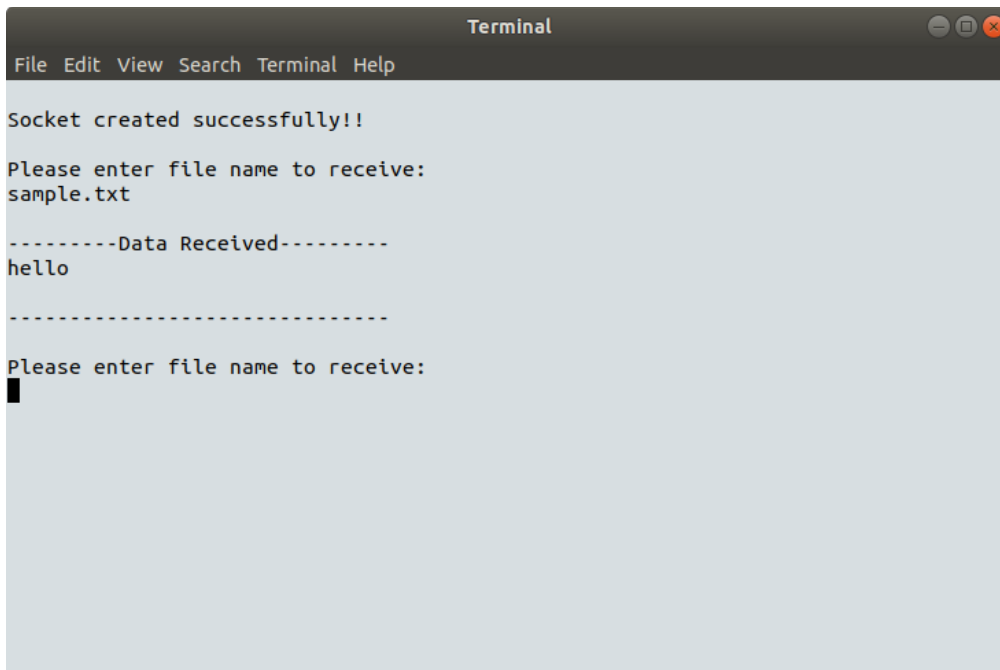


Results:

Server :

```
Terminal
File Edit View Search Terminal Help
Socket created successfully!!
Successfully binded!
Waiting for file name...
File Name Received: sample.txt
File Successfully opened!
Waiting for file name...
█
```

Client :

A screenshot of a macOS Terminal window titled "Terminal". The window has a dark title bar with standard window controls (minimize, maximize, close) on the right. Below the title bar is a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal is light gray and contains the following text: "Socket created successfully!!", "Please enter file name to receive:", "sample.txt", "-----Data Received-----", "hello", "-----", "Please enter file name to receive:", and a cursor. The terminal window has a red vertical bar on the right side.

```
Terminal
File Edit View Search Terminal Help

Socket created successfully!!

Please enter file name to receive:
sample.txt

-----Data Received-----
hello

-----
Please enter file name to receive:
█
```

Conclusion:

Here we can conclude that TCP/IP used to communicate and transfer data between client and server. A server creates a socket, binds the socket to an IP address and port number and then listens for incoming connections. When a client connects to the server, a new socket is created for communication with the client.