Network Programming:

* Network application programming is all about writing application programs using Network API’s and Network protocol stack.
* This program will communicate with programs either on the same machine or program on the different machine on remote location.
* LAN works on broadcast approach, no intermediate switches are present, due to which the data rates is faster in LAN as compared to WAN.
* LAN is divided into two parts

1. CSN (Circuit Switching Network):

* A dedicated path is applied between two nodes, a path is a collection of sequence of physical link between the nodes. (Cable Lines).
* If a sender wants to transmit the packet the packet goes through this physical links and rapidly as it can.
* Ex: Telephone communication

1. PSN (Packet Switching Network)

* It uses network protocols that divides the data into smaller packets called as network packets.
* Network packets are transmitted over the network.
* In Digital format it sends Packets.
* Ex: Internet, WIFI.

**Networking Devices:**

1. Router and Switches:

They are Used to connecting 1 or more devices to other computers or other networking devices or to connect other networks.

1. HUB:

It is used to connects devices in LAN.

1. Bridges:

It connects two LAN’s on a same Network.

**OSI Model:**

Application Layer:

* User interface Layer.
* Also known as desktop layer.

Presentation Layer:

* Encryption and Decryption of data is implemented in this layer.

Session Layer:

* It creates connection between sender and receiver, which maintaining a session between sender and receiver.
* It must maintain a session until receiver receives a packet.
* It maintains synchronization and terminate the session.
* While transmitting a packet from the session layer because of the events session layer stop transmitting in next cycle session layer did not to start from the beginning, it continues transmitting from the point of interruption. (This is synchronization) .

Transport Layer:

* It receives data from session layer and divide the data into small segment.

Network Layer:

* Network layers take data from the transport layer and converts into the network packets.
* Network layer has IP addresses of source and destination.
* All the routing process starts from the network layer, then transmitted to datalink layer.
* Network layer knows the sender and receiver of the packet, will decide which path the data should transmit to destination.

Datalink Layer:

* It checks for error message in data. (i.e., checking if data is error free or not, I there is any error it removes the error and transfer the error free packet forward.)
* Convert packet into frames.
* Control the data rate speed between the sender and receiver.
* Will also find the physical addressing job.

Physical Link Layer:

* It converts the frame into digit format i.e., 0’s and 1’s.
* At physical link layer we have devices: router, switches, hub that will convert the digital data into electrical signal or the radio waves.

**Ifconfig** (interface configuration) is a tool that display active network interfaces of a system. Ifconfig tool is used to view and change the network interface configurations of a system.

**MTU** (Maximum transferring unit): Maximum size of packet in a transmission.

**TXQ**: Maximum no of packet in a transmission. Txq length is the interface of the Linux kernel that deals with the network devices.

**Loopback address**:

* Each device has loop back address 127.0.0.1.
* It is also called as Local Host. When we send the data using loopback address the data never reaches the network, data is in the loop in the network card itself.
* Loopback address is used for testing TCP/IP internet protocol flow path.
* Loopback address will help the device to send and receives the packet on its own.

**Socket system Calls:**

* Socket system calls many system network protocols, we can say that socket system calls are design to support network communication protocols.
* Because of the same reason socket system called parameters are generic in nature.
* As we are using generic parameters all the system calls use same socket structure as argument (struct Sockaddr).
* Socket system calls also take the size as a parameter; this will identify the size of socket structure.
* Socket is an end point or a final point or a end link in any network communication.
* Basically, we need 2 sockets one for sender and one for receiver.

**Socket Parameters:**

Sockets need five different parameters:

1. Protocols
2. IP addresses of source
3. IP addresses of destination
4. Port address of source
5. Port address of destination

* Network application programming is all about implementing Client and server model program.
* Http client program connects Http server program example: google
* POP client connecting POP server: Downloading documents in mail

**Network engineer Jobs**

* Customizing WAN
* Writing routing protocol
* Optimizing server data center
* Write own network driver (Network driver operator)

**Software system calls:**

1. Socket:
2. Binding:
3. Listen:
4. API:

* At this stage server is listening for the client until it gets the client, server is said to be in while loop (server is in blocking operation).
* After server gets the clients, server should accept the clients.
* Once the connection establishes between client and server the communication starts between client and server.

1. **Socket structure:**

**{**

**Short Sin\_family;** (decide Network protocol type)

**Ushort Sin\_port;** (it talks about 16-bit port order in network byte order)

**Struct in\_addr Sin\_Addr;** (it talks about 32 addresses in network byte order)

**}**

**Int Socket (int family, int type, int protocol);**

* We use family **(AF\_INET =** address format internet protocol)
* Int type:
  + - Connection oriented (TCP= SOCK\_STREAM)
    - Connectionless (UDP = SOCK\_DGRAM)

In IP protocol header the protocol value is zero

* On success full execution of a socket system calls return small integer called as socket file descriptor.
* In failure -1.
* Sockets are internally nothing but a special file.

1. **Bind**

Binding binds the IP to the server.

**Bind( int Socket, struct Sockaddr \*Serv, int addlen)**

1. **Listen:**

**Listen(int socket, int Queuelen)**

(Queuelength=No of client connection request)

1. **Accept**

**accept(int sockfd, struct sockaddr \*cli, int \*addrcli)**

**connect(int sockfd, Struct Sockaddr \*ser, int addrlen)**

***Big and Little endian***

* In a network there are different types of computers which are using different CPU’s and Microcontrollers few CPU follows big endian format and few follow little endian format to store integer byte in the memory.
* TCP/IP specifies all the Protocol Headers of a network packet to follow Network byte order. (Big endian format)
* Every machine in the network should be aware of this and convert host order to network byte order.
* Again, at the receiver side while receiving the incoming packets should convert network byte order to Host byte order.
* Source code used the API to convert host network order to network to host order to match respective CPU requirement.
* This API will work for both little- and big-endian Machine.

Host to network long function

**htonl (long hostlong)**

Long hostlong = takes 32-bit byte address and convert into 32-bit network byte order

***Address conversion API’s:***

**Inet\_addr(Char \*str);**

Return type unsigned long

**Inet\_ntoa ( unlong n/wlong);**

Return char\*

**Iterator server:**

* Iterator server are one that server one connection (client) at a time.
* Second client in the queue waits till client one has terminated, only than server serves the client.
* After accept system call a new client socket is created of same type now enters the while loop and server receives the client data and process the data.
* Now the program will come out of inner while loop only when client closes the connection(n=0).
* Once client closes server is again executing a access system calls and waiting for a new client.
* Iterators server has a simple interface.
* If client transaction duration is less than iterators server is good.

**Concurrent servers:**

* They can handle multiple clients simultaneously.

**Network statics:**

* It talks about how your computer communicating with other computer or other networking devices.
* It provides detail information of induvial network communication. Also, protocol information.
* The information is used for troubleshooting and diagnose the errors.

A file called services in ETC folder display all the services that are used by client application from user computer.

A function called get servant will read the next line of services file in etc. folder and return a pointer to next entry in services file.

* PF\_Packet family used to print packet socket
* Data transmission is from the physical link layer i.e., below the IP Layer.
* Type of communication can be SOCK\_RAW or SOCK\_DGRAM
* When including SOCK\_RAW packet including link layer header is passed to the specific user.
* When using type as a SOCK\_DGRAM packet without link layer header is passed to specific user.
* Protocol value is define by IEEE people, a standard Ethernet protocol(ETH\_P\_ALL).

**int setsockopt(int *socket*, int *level*, int *option\_name*, const void \**option\_value*, socklen\_t *option\_len*);**

SetSockopt is used by the application program to change behaviour, we can set any action of socket.

SetRefusePort: 1 for set and 0 for default.

* While implementing terminal emulate program, it reads from multiple devices like keyboard and mouse, if a configured a process to read from keyboard when there is no data from the keyboard the process will block, even if there is a data in the serial port process cannot read.
* To overcome this situation, we can use multiple threads to read from multiple input sources.
* Other operation is to used file descriptor using select system call.

**Select system calls:**

* Select system call makes a programmer to read from multiple file descriptor, when file descriptor are ready for I/O operation.

int select (int nfds, fd\_set \*readfds, fd\_set \*writefds,

fd\_set \*exceptfds, struct timeval \*timeout);

* the first parameter nfds tells the select system call the number of file descriptor to be checked.
* The first pointer of type fd\_set is for testing the readies of file descriptors for read operation.
* The second pointer of type fd\_set is for testing readies of file descriptors for write operation.
* If any of file descriptor have error pending will be verified by exceptfds pointer.
* Timeout is a variable or a object of type struct timevar of struct.
* This timeout make system call to block until a fd is ready for I/O operation.
* If no fd is set by any pointer than it returns 0.
* On successful read or write they return successful no of fd set by either read ,write or error.
* And if there is error than it returns -1.

**SSH Protocol:**

* $ssh usernamey@IPaddressY
* Using open SSH client can connect to remote machines and control the machines.
* Open SSH is an open-source software i.e is used to provide a secure and encrypted communication in a computer network using SSH protocol.
* SSH protocol is one that establish a connection between SSH client and SSH server.
* Generally used by system administrator to access the remote machine in a unsecure network.
* SSH is a TCP/IP protocol that uses dedicate port number 22.
* Once one has provided username and ip address of y, the y system shares a key to X if it really wants to establish connection.

**SCP: secure copy protocol:**

* Used to connect to machine over the network and to secure the transfer of file over network.

***Wireshark:***

* It is open-source software tool that captures incoming and outgoing network data packets in real time and the captured information is used for network troubleshooting and diagnosing and used for packet analysis, software development and communication protocol development.
* With Wireshark we can see the things around the network, and we can get the detail packet movement information.
* Wireshark supports 100 of packets of different protocols, they only logic operates is capturing online and examining them offline.
* It can read and write different captured file format.
* Main toolbars give 4 functionalities of Wireshark
* Filter bar is used to filter the required packet
* Status bar the bottom line will show us the complete summary the lost packets and the received packets.
* The red button is used to stop the connection.

**IP META Data:**

IP packets carry IP header (20-24) bytes + data, IP has source and destination IP address and the routing information with the actual data is called IP meta data.

**Promiscuous mode**

* It captures network data packets of other network devices on the same network i.e. capturing the packets or traffic going through all other devices of the network.
* By default, the network card is listening and monitoring the packets coming to your MAC address.
* Once this mode is started allows the network card to listen and monitor the packets of other networking devices.

**Filters:**

* If you want to capture the packets with respect to protocols or port number, we need to inform the Wireshark before starting the filtration.
* Even using IP address, we can filter.
* Capture filters: The capture filter is used to filtering during a live session