**Sockets:**

We have different types of sockets which works based on different protocols.

They are:

**Datagram Sockets: works on UDP protocol.(un reliable, connection-less)** SOCK\_DGRAM

**Stream Sockets: works on TCP protocol(reliable, connection oriented)** SOCK\_STREAM

**Raw Sockets: works on ICMP protocol(un-reliable, connection-less)** SOCK\_RAW.

**Sequenced Packet Sockets: works on IDP protocol( similar to stream socket, sequence packets.** SOCK\_SEQPACKET.

Address families:

**UNIX:** Provides socket communication between processes running on the same operating system when an address family of **AF\_UNIX** is specified. A socket name in the UNIX domain is a string of ASCII characters whose maximum length depends on the machine in use.

**Internet:** Provides socket communication between a local process and a process running on a remote host when an address family of **AF\_INET** is specified. The Internet domain requires that Transmission Control Protocol/Internet Protocol (TCP/IP) be installed on your system. A socket name in the Internet domain is an Internet address, made up of a 32-bit IP address and a 16-bit port address.

**NDD:** Provides socket communication between a local process and a process running on a remote host when an address family of **AF\_NDD** is specified. The NDD domain enables applications to run directly on top of physical networks. This is in contrast to the Internet domain, in which applications run on top of transport protocols such as TCP, or User Datagram Protocol (UDP). A socket name in the NDD domain consists of operating system NDD name and a second part that is protocol dependent.

**AF\_UNIX:**

Its address structure format:

A UNIX domain socket address is represented in the following

structure:

struct sockaddr\_un {

sa\_family\_t sun\_family; /\* AF\_UNIX \*/

char sun\_path[108]; /\* Pathname \*/

};

AF\_INET4: It is possible for these type of sockets.

tcp\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

udp\_socket = socket(AF\_INET, SOCK\_DGRAM, 0);

raw\_socket = socket(AF\_INET, SOCK\_RAW, protocol);

Address structure:

struct sockaddr\_in {

sa\_family\_t sin\_family; /\* address family: AF\_INET \*/

in\_port\_t sin\_port; /\* port in network byte order \*/

struct in\_addr sin\_addr; /\* internet address \*/

};

/\* Internet address \*/

struct in\_addr {

uint32\_t s\_addr; /\* address in network byte order \*/

};

protocol is the IP protocol in the IP header to be received or sent. Valid values for protocol include:

• 0 and IPPROTO\_TCP for tcp(7) stream sockets;

• 0 and IPPROTO\_UDP for udp(7) datagram sockets;

• IPPROTO\_SCTP for sctp(7) stream sockets; and

• IPPROTO\_UDPLITE for udplite(7) datagram sockets.

AF\_INET6:

IpV6:

tcp6\_socket = socket(AF\_INET6, SOCK\_STREAM, 0);

raw6\_socket = socket(AF\_INET6, SOCK\_RAW, protocol);

udp6\_socket = socket(AF\_INET6, SOCK\_DGRAM, protocol);

Address format

struct sockaddr\_in6 {

sa\_family\_t sin6\_family; /\* AF\_INET6 \*/

in\_port\_t sin6\_port; /\* port number \*/

uint32\_t sin6\_flowinfo; /\* IPv6 flow information \*/

struct in6\_addr sin6\_addr; /\* IPv6 address \*/

uint32\_t sin6\_scope\_id; /\* Scope ID (new in 2.4) \*/

};

struct in6\_addr {

unsigned char s6\_addr[16]; /\* IPv6 address \*/

};