Ex.No: 1	**Study of necessary header files with respect to socket programming**
Date:	

Aim:

To Study of necessary header files with respect to socket programming

Description:

1. stdio.h:

Has standard input and output library providing simple and efficient buffered stream IO interface.

2. unistd.h:

It is a POSIX standard for open system interface. [Portable Operating System Interface

3. string.h:

This header file is used to perform string manipulation operations on NULL terminated strings.(Bzero -0 the m/y)

4. stdlib.h:

This header file contains the utility functions such as string conversion routines, memory allocation routines, random number generator, etc.

5. sys/types.h:

Defines the data type of socket address structure in unsigned long.

6. sys/socket.h:

The socket functions can be defined as taking pointers to the generic socket address structure

called sockaddr.

7. netinet/in.h:

Defines the IPv4 socket address structure commonly called Internet socket address structure called sockaddr_in.

8. netdb.h:

Defines the structure hostent for using the system call gethostbyname to get the network host entry.

9. time.h:

Has structures and functions to get the system date and time and to perform time manipulation functions. We use the function ctime(), that is defined in this header file , to calculate the current date and time.

10. sys/stat.h:

Contains the structure stat to test a descriptor to see if it is of a specified type. Also it is used to display file or file system status.stat() updates any time related fields.when copying from 1 file to

another.

11. sys/ioctl.h:

Macros and defines used in specifying an ioctl request are located in this header file. We use the function ioctl() that is defined in this header file. ioctl() function is used to perform ARP cache operations.

12. pcap.h:

Has function definitions that are required for packet capturing. Some of the functions are pcap_lookupdev(),pcap_open_live() and pcap_loop(). pcap_lookupdev() is used to initialize the network device.The device to be sniffed is opened using the pcap_open_live(). Pcap_loop() determines the number of packets to be sniffed.

13. net/if_arp.h:

Contains the definitions for Address Resolution Protocol. We use this to manipulate the ARP request structure and its data members arp_pa,arp_dev and arp_ha. The arp_ha structure's data member sa_data[ ] has the hardware address.

14. errno.h:

It sets an error number when an error and that error can be displayed using perror function. It has symbolic error names. The error number is never set to zero by any library function.

15. arpa/inet.h:

This is used to convert internet addresses between ASCII strings and network byte ordered binary values (values that are stored in socket address structures). It is used for inet_aton, inet_addr, inet_ntoa functions

Result: Thus the Study of necessary header files with respect to socket programming has been done.

Ex.No: 2	**Study of Basic Functions of Socket Programming**
Date:	

Aim:
To discuss some of the basic functions used for socket programming.

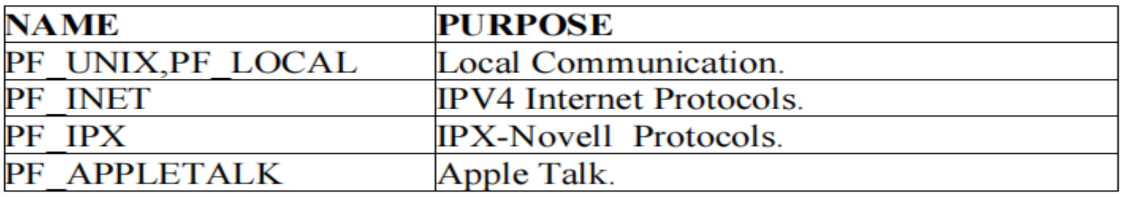
1.**man socket**

NAME:
Socket – create an endpoint for communication.

SYNOPSIS:
#include<sys/types.h>
#include<sys/socket.h>
int socket(int domain,inttype,int protocol);

DESCRIPTION:
➢ Socket creates an endpoint for communication and returns a descriptor.
➢ The domain parameter specifies a common domain this selects the protocol family
which will be used for communication.
➢ These families are defined in <sys/socket.h>.

FORMAT:



NAME	PURPOSE
PF_UNIX,PF_LOCAL	Local Communication.
PF_INET	IPV4 Internet Protocols.
PF_IPX	IPX-Novell Protocols.
PF_APPLETALK	Apple Talk.

➢ The socket has the indicated type, which specifies the communication semantics.

TYPES:
1.SOCK_STREAM:
➢ Provides sequenced , reliable, two-way , connection based byte streams.
➢ An out-of-band data transmission mechanism, may be supported.Page | 4
2.SOCK_DGRAM:
➢ Supports datagram (connectionless, unreliable messages of a fixed maximum length).
3.SOCK_SEQPACKET:
➢ Provides a sequenced , reliable, two-way connection based data transmission path for
datagrams of fixed maximum length.
4.SOCK_RAW:
➢ Provides raw network protocol access.

5.SOCK_RDM:
➢ Provides a reliable datagram layer that doesn't guarantee ordering.
6.SOCK_PACKET:
➢ Obsolete and shouldn't be used in new programs.

2.man connect:

NAME:
connect – initiate a connection on a socket.

SYNOPSIS:
#include<sys/types.h>
#include<sys/socket.h>
int connect(int sockfd,const (struct sockaddr*)serv_addr,socklen_taddrlen);

DESCRIPTION:
➢ The file descriptor sockfd must refer to a socket.
➢ If the socket is of type SOCK_DGRAM then the serv_addr address is the address to which datagrams are sent by default and the only addr from which datagrams are received.
➢ If the socket is of type SOCK_STREAM or SOCK_SEQPACKET , this call attempts to make a connection to another socket.
RETURN VALUE:
➢ If the connection or binding succeeds, zero is returned.
➢ On error , -1 is returned , and error number is set appropriately.

ERRORS:

EBADF	Not a valid Index.
EFAULT	The socket structure address is outside the user's address space.
ENOTSOCK	Not associated with a socket.
EISCONN	Socket is already connected.
ECONNREFUSED	No one listening on the remote address.

3.man accept
NAME:
accept/reject job is sent to a destination.
SYNOPSIS:
accept destination(s)
reject[-t] [-h server] [-r reason] destination(s)
DESCRIPTION:

➢ accept instructs the printing system to accept print jobs to the specified destination.
➢ The –r option sets the reason for rejecting print jobs.
➢ The –e option forces encryption when connecting to the server.

4. man send

NAME:
send, sendto, sendmsg - send a message from a socket.
SYNOPSIS:
#include<sys/types.h>
#include<sys/socket.h>
ssize_tsend(int s, const void *buf, size_tlen, int flags);
ssize_tsendto(int s, const void *buf, size_tlen, int flags, const struct sock_addr*to, socklen_ttolen);
ssize_tsendmsg(int s, const struct msghdr *msg, int flags);

DESCRIPTION:
➢ The system calls send, sendto and sendmsg are used to transmit a message to another
socket.
➢ The send call may be used only when the socket is in a connected state.
➢ The only difference between send and write is the presence of flags.
➢ The parameter is the file descriptor of the sending socket.

5.man recv
NAME:
recv, recvfrom, recvmsg – receive a message from a socket.
SYNOPSIS:
#include<sys/types.h>
#include<sys/socket.h>
ssize_trecv(int s, void *buf, size_tlen, int flags);
ssize_trecvfrom(int s, void *buf, size_tlen, int flags, struct sockaddr *from, socklen_t* from len);
ssize_trecvmsg(int s, struct msghdr *msg, int flags);
DESCRIPTION:
➢ The recvfrom and recvmsg calls are used to receive messages from a socket, and may
be used to recv data on a socket whether or not it is connection oriented.
➢ If from is not NULL, and the underlying protocol provides the srcaddr , this srcaddr is
filled in.
➢ The recv call is normally used only on a connection socket and is identical to recvfrom
with a NULL from parameter.

6.man read

NAME:
read, readonly, return

7.man write
NAME:
write- send a message to another user.
SYNOPSIS:
write user[ttyname]

DESCRIPTION:
➢ write allows you to communicate with other users, by copying lines from terminal to
………
➢ When you run the write and the user you are writing to get a message of the form:
Message from yourname @yourhost on yourtty at hh:mm:…
➢ Any further lines you enter will be copied to the specified user's terminal.
➢ If the other user wants to reply they must run write as well.

8. ifconfig
NAME:
ifconfig- configure a network interface.
SYNOPSIS:
Page | 6 ifconfig[interface]
ifconfig interface[aftype] options | address……
DESCRIPTION:
➢ ifconfig is used to configure the kernel resident network interfaces.
➢ It is used at boot time to setup interfaces as necessary.
➢ After that, it is usually only needed when debugging or when system tuning is needed.
➢ If no arguments are given, ifconfig displays the status of the currently active interfaces.

9. man bind
SYNOPSIS:
bind[-m keymap] [-lpsvpsv]

10. man htons/ man htonl
NAME:
htonl, htons, ntohl, ntohs- convert values between host and network byte order.
SYNOPSIS:
#include<netinet/in.h>
uint32_t htonl(uint32_t hostlong);
uint16_t htons(uint32_t hostshort);
uint32_t ntohl(uint32_t netlong);
uint16_t ntohs(uint16_t netshort);

DESCRIPTION:

➢ The htonl() function converts the unsigned integer hostlong from host byte order to
network byte order.
➢ The htons() converts the unsigned short integer hostshort from host byte order to network
byte order.
➢ The ntohl() converts the unsigned integer netlong from network byte order to host byte
order.

11. man gethostname
NAME:
gethostname, sethostname- get/set host name.
SYNOPSIS:
#include<unistd.h>
Page | 7 int gethostname(char *name,size_tlen);
int sethostname(const char *name,size_tlen);

DESCRIPTION:
➢ These functions are used to access or to change the host name of the current processor.
➢ The gethostname() returns a NULL terminated hostname(set earlier by sethostname()) in
the array name that has a length of len bytes.
➢ In case the NULL terminated then hostname does not fit ,no error is returned, but the
hostname is truncated.
➢ It is unspecified whether the truncated hostname will be NULL terminated.

12. man gethostbyname
NAME:
gethostbyname, gethostbyaddr, sethostent, endhostent, herror, hstr – error – get network host entry.

SYNOPSIS:
#include<netdb.h>
extern int h_errno;
struct hostent *gethostbyname(const char *name);
#include<sys/socket.h>
struct hostent *gethostbyaddr(const char *addr)int len, int type);
struct hostent *gethostbyname2(const char *name,intaf);

DESCRIPTION:
➢ The gethostbyname() returns a structure of type hostent for the given hostname.
➢ Name->hostname or IPV4/IPV6 with dot notation.
➢ gethostbyaddr()- struct of type hostent / host address length
➢ Address types- AF_INET, AF_INET6.

➢ sethostent() – stay open is true(1).
➢ TCP socket connection should be open during queries.
➢ Server queries for UDP datagrams.
➢ endhostent()- ends the use of TCP connection.
➢ Members of hostent structure:
a) h_name
b) h_aliases
c) h_addrtype
d) h_length
e) h_addr-list
f) h_addr.

RESULT: Thus the basic functions used for Socket Programming was studied successfully.