UDP Server

#include <arpa/inet.h>

#include <netinet/in.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <unistd.h>

int main(void)

{

struct sockaddr\_in si\_me, si\_other;

int s, i, slen=sizeof(si\_other);

char buf[500];

s=socket(AF\_INET, SOCK\_DGRAM, IPPROTO\_UDP))==-1)

si\_me.sin\_family = AF\_INET;

si\_me.sin\_port = htons(9930);

si\_me.sin\_addr.s\_addr = htonl(“127.0.0.1”);

bind(s, &si\_me, sizeof(si\_me));

for (i=0; i<10; i++) {

recvfrom(s, buf, BUFLEN, 0, &si\_other, &slen);

printf("Received packet from %s:%d\nData: %s\n\n",

inet\_ntoa(si\_other.sin\_addr), ntohs(si\_other.sin\_port), buf);

}

close(s);

return 0;

}

UDP Client

#include <arpa/inet.h>

#include <netinet/in.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <unistd.h>

int main(void)

{

struct sockaddr\_in si\_other;

int s, i, slen=sizeof(si\_other);

char buf[500];

s=socket(AF\_INET, SOCK\_DGRAM, IPPROTO\_UDP));

si\_other.sin\_family = AF\_INET;

si\_other.sin\_port = htons(9930);

for (i=0; i<10; i++) {

printf("Sending packet %d\n", i);

sprintf(buf, "This is packet %d\n", i);

sendto(s, buf, 500, 0, &si\_other, slen);

}

close(s);

return 0;

}

Functions

The *sendto*() function shall send a message through a connection-mode or connectionless-mode socket. If the socket is connectionless-mode, the message shall be sent to the address specified by *dest\_addr*. If the socket is connection-mode, *dest\_addr* shall be ignored.

The *sendto*() function takes the following arguments:

*socket*

Specifies the socket file descriptor.

*message*

Points to a buffer containing the message to be sent.

*length*

Specifies the size of the message in bytes.

*flags*

Specifies the type of message transmission. Values of this argument are formed by logically OR'ing zero or more of the following flags:

MSG\_EOR

Terminates a record (if supported by the protocol).

MSG\_OOB

Sends out-of-band data on sockets that support out-of-band data. The significance and semantics of out-of-band data are protocol-specific.

*dest\_addr*

Points to a **sockaddr** structure containing the destination address. The length and format of the address depend on the address family of the socket.

*dest\_len*

Specifies the length of the **sockaddr** structure pointed to by the *dest\_addr* argument.

The *recvfrom()* function receives a message from a connection-mode or connectionless-mode socket. It is normally used with connectionless-mode sockets because it permits the application to retrieve the source address of received data.

The function takes the following arguments:

*socket*

Specifies the socket file descriptor.

*buffer*

Points to the buffer where the message should be stored.

*length*

Specifies the length in bytes of the buffer pointed to by the *buffer* argument.

*flags*

Specifies the type of message reception. Values of this argument are formed by logically OR'ing zero or more of the following values:

MSG\_PEEK

Peeks at an incoming message. The data is treated as unread and the next *recvfrom()* or similar function will still return this data.

MSG\_OOB

Requests out-of-band data. The significance and semantics of out-of-band data are protocol-specific.

MSG\_WAITALL

Requests that the function block until the full amount of data requested can be returned. The function may return a smaller amount of data if a signal is caught, if the connection is terminated, if MSG\_PEEK was specified, or if an error is pending for the socket.

*address*

A null pointer, or points to a **sockaddr** structure in which the sending address is to be stored. The length and format of the address depend on the address family of the socket.

*address\_len*

Specifies the length of the **sockaddr** structure pointed to by the *address* argument.