NAME

udp, UDP – Internet User Datagram Protocol

SYNOPSIS

#include <sys/socket.h>

#include <netinet/in.h>

s = socket(AF_INET, SOCK_DGRAM, 0);

t = t_open("/dev/udp", O_RDWR);

DESCRIPTION

UDP is a simple datagram protocol which is layered directly above the Internet Protocol (IP). Programs may access UDP using the socket interface, where it supports the **SOCK_DGRAM** socket type, or using the Transport Level Interface (TLI), where it supports the connectionless (**T_CLTS**) service type.

Within the socket interface, UDP is normally used with the **sendto( )**, **sendmsg( )**, **recvfrom( )**, and **recvmsg( )** calls (see **send**(3N) and **recv**(3N)). If the **connect**(3N) call is used to fix the destination for future packets, then the **recv**(3N) or **read**(2) and **send**(3N) or **write**(2) calls may be used.

UDP address formats are identical to those used by the Transmission Control Protocol (TCP). Like TCP, UDP uses a port number along with an IP address to identify the endpoint of communication. The UDP port number space is separate from the TCP port number space (that is, a UDP port may not be “connected” to a TCP port). The **bind**(3N) call can be used to set the local address and port number of a UDP socket. The local IP address may be left unspecified in the **bind( )** call by using the special value **INADDR_ANY**. If the **bind( )** call is not done, a local IP address and port number will be assigned to the endpoint when the first packet is sent. Broadcast packets may be sent (assuming the underlying network supports this) by using a reserved “broadcast address”; This address is network interface dependent. Broadcasts may only be sent by the privileged user.

Options at the IP level may be used with UDP; see **ip**(7P).

There are a variety of ways that a UDP packet can be lost or corrupted, including a failure of the underlying communication mechanism. UDP implements a checksum over the data portion of the packet. If the checksum of a received packet is in error, the packet will be dropped with no indication given to the user. A queue of received packets is provided for each UDP socket. This queue has a limited capacity. Arriving datagrams which will not fit within its *high-water* capacity are silently discarded.

UDP processes Internet Control Message Protocol (ICMP) error messages received in response to UDP packets it has sent. See **icmp**(7P). ICMP “source quench” messages are ignored. ICMP “destination unreachable,” “time exceeded” and “parameter problem” messages disconnect the socket from its peer so that subsequent attempts to send packets using that socket will return an error. UDP will not guarantee that packets are delivered in the order they were sent. As well, duplicate packets may be generated in the communication process.

SEE ALSO

read(2), **write**(2), **bind**(3N), **connect**(3N), **recv**(3N), **send**(3N), **icmp**(7P), **inet**(7P), **ip**(7P), **tcp**(7P)

Postel, Jon, *User Datagram Protocol*, RFC 768, Network Information Center, SRI International, Menlo Park, Calif., August 1980

DIAGNOSTICS

A socket operation may fail if:

EISCONN — A **connect( )** operation was attempted on a socket on which a **connect( )** operation had already been performed, and the socket could not be successfully disconnected before making the new connection.

EISCONN — A **sendto( )** or **sendmsg( )** operation specifying an address to which the message should be sent was attempted on a socket on which a **connect( )** operation had already been performed.

ENOTCONN — A **send( )** or **write( )** operation, or a **sendto( )** or **sendmsg( )** operation not specifying an address to which the message should be sent, was attempted on a socket on which a **connect( )** operation had not already been performed.

EADDRINUSE — A **bind( )** operation was attempted on a socket with a network address/port pair that has already been bound to another socket.

EADDRNOTAVAIL — A **bind( )** operation was attempted on a socket with a network address for which no network interface exists.

EINVAL — A **sendmsg( )** operation with a non-NULL **msg_accrights** was attempted.

EACCES — A **bind( )** operation was attempted with a “reserved” port number and the effective user ID of the process was not the privileged user.

ENOBUFS — The system ran out of memory for internal data structures.