NAME

sctp_recvmsg - receive a message from an SCTP socket

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/sctp.h>

ssize_t
sctp_recvmsg(*int s*, *void *msg*, *size_t len*, *struct sockaddr * restrict from*, *socklen_t * restrict fromlen*,
*struct sctp_sndrcvinfo *sinfo*, *int *flags*);

DESCRIPTION

The **sctp_recvmsg**() system call is used to receive a message from another SCTP endpoint. The **sctp_recvmsg**() call is used by one-to-one (SOCK_STREAM) type sockets after a successful **connect**() call or after the application has performed a **listen**() followed by a successful **accept**(). For a one-to-many (SOCK_SEQPACKET) type socket, an endpoint may call **sctp_recvmsg**() after having implicitly started an association via one of the send calls including **sctp_sendmsg**(), **sendto**() and **sendmsg**(). Or, an application may also receive a message after having called **listen**() with a positive backlog to enable the reception of new associations.

The address of the sender is held in the *from* argument with *fromlen* specifying its size. At the completion of a successful **sctp_recvmsg**() call *from* will hold the address of the peer and *fromlen* will hold the length of that address. Note that the address is bounded by the initial value of *fromlen* which is used as an in/out variable.

The length of the message *msg* to be received is bounded by *len*. If the message is too long to fit in the users receive buffer, then the *flags* argument will *not* have the MSG_EOR flag applied. If the message is a complete message then the *flags* argument will have MSG_EOR set. Locally detected errors are indicated by a return value of -1 with *errno* set accordingly. The *flags* argument may also hold the value MSG_NOTIFICATION. When this occurs it indicates that the message received is *not* from the peer endpoint, but instead is a notification from the SCTP stack (see sctp(4) for more details). Note that no notifications are ever given unless the user subscribes to such notifications using the SCTP_EVENTS socket option.

If no messages are available at the socket then **sctp_recvmsg**() normally blocks on the reception of a message or NOTIFICATION, unless the socket has been placed in non-blocking I/O mode. The select(2) system call may be used to determine when it is possible to receive a message.

The *sinfo* argument is defined as follows.

```
struct sctp_sndrcvinfo {
        uint16_t sinfo_stream;  /* Stream arriving on */
        uint16_t sinfo_ssn;     /* Stream Sequence Number */
        uint16_t sinfo_flags;   /* Flags on the incoming message */
        uint32_t sinfo_ppid;    /* The ppid field */
        uint32_t sinfo_context; /* context field */
        uint32_t sinfo_timetolive; /* not used by sctp_recvmsg */
        uint32_t sinfo_tsn;        /* The transport sequence number */
        uint32_t sinfo_cumtsn;     /* The cumulative acknowledgment point  */
        sctp_assoc_t sinfo_assoc_id; /* The association id of the peer */
};
```

The *sinfo*->*sinfo_ppid* field is an opaque 32 bit value that is passed transparently through the stack from the peer endpoint. Note that the stack passes this value without regard to byte order.

The *sinfo*->*sinfo_flags* field may include the following:

```
#define SCTP_UNORDERED          0x0400  /* Message is un-ordered */
```

The SCTP_UNORDERED flag is used to specify that the message arrived with no specific order and was delivered to the peer application as soon as possible. When this flag is absent the message was delivered in order within the stream it was received.

The *sinfo*->*sinfo_stream* field is the SCTP stream that the message was received on. Streams in SCTP are reliable (or partially reliable) flows of ordered messages.

The *sinfo*->*sinfo_context* field is used only if the local application set an association level context with the SCTP_CONTEXT socket option. Optionally a user process can use this value to index some application specific data structure for all data coming from a specific association.

The *sinfo*->*sinfo_ssn* field will hold the stream sequence number assigned by the peer endpoint if the message is *not* unordered. For unordered messages this field holds an undefined value.

The *sinfo*->*sinfo_tsn* field holds a transport sequence number (TSN) that was assigned to this message by the peer endpoint. For messages that fit in or less than the path MTU this will be the only TSN assigned. Note that for messages that span multiple TSNs this value will be one of the TSNs that was used on the message.

The *sinfo->sinfo_cumtsn* field holds the current cumulative acknowledgment point of the transport association. Note that this may be larger or smaller than the TSN assigned to the message itself.

The *sinfo->sinfo_assoc_id* is the unique association identification that was assigned to the association. For one-to-many (SOCK_SEQPACKET) type sockets this value can be used to send data to the peer without the use of an address field. It is also quite useful in setting various socket options on the specific association (see sctp(4)).

The *sinfo->info_timetolive* field is not used by **sctp_recvmsg**().

RETURN VALUES

The call returns the number of bytes received, or -1 if an error occurred.

ERRORS

The **sctp_recvmsg**() system call fails if:

[EBADF]	An invalid descriptor was specified.
[ENOTSOCK]	The argument *s* is not a socket.
[EFAULT]	An invalid user space address was specified for an argument.
[EMSGSIZE]	The socket requires that message be sent atomically, and the size of the message to be sent made this impossible.
[EAGAIN]	The socket is marked non-blocking and the requested operation would block.
[ENOBUFS]	The system was unable to allocate an internal buffer. The operation may succeed when buffers become available.
[ENOBUFS]	The output queue for a network interface was full. This generally indicates that the interface has stopped sending, but may be caused by transient congestion.
[EHOSTUNREACH]	The remote host was unreachable.
[ENOTCONN]	On a one-to-one style socket no association exists.

[ECONNRESET]	An abort was received by the stack while the user was attempting to send data to the peer.
[ENOENT]	On a one to many style socket no address is specified so that the association cannot be located or the SCTP_ABORT flag was specified on a non-existing association.
[EPIPE]	The socket is unable to send anymore data (SBS_CANTSENDMORE has been set on the socket). This typically means that the socket is not connected and is a one-to-one style socket.

SEE ALSO

getsockopt(2), recv(2), select(2), sendmsg(2), setsockopt(2), socket(2), write(2), sctp_send(3), sctp_sendmsg(3), sctp(4)