sendmmsg(2) — Linux manual page

NAME top

sendmmsg - send multiple messages on a socket

LIBRARY top

Standard C library (*libc*, *-lc*)

SYNOPSIS top

```
#define _GNU_SOURCE         /* See feature_test_macros(7) */
#include <sys/socket.h>

int sendmmsg(unsigned int n;
             int sockfd, struct mmsghdr msgvec[n], unsigned int n,
             int flags);
```

DESCRIPTION top

The **sendmmsg**() system call is an extension of sendmsg(2) that allows the caller to transmit multiple messages on a socket using a single system call. (This has performance benefits for some applications.)

The *sockfd* argument is the file descriptor of the socket on which data is to be transmitted.

The *msgvec* argument is a pointer to an array of *mmsghdr* structures. The size of this array is specified in *n*.

The *mmsghdr* structure is defined in *<sys/socket.h>* as:

```
struct mmsghdr {
    struct msghdr msg_hdr;  /* Message header */
    unsigned int  msg_len;  /* Number of bytes transmitted */
};
```

The *msg_hdr* field is a *msghdr* structure, as described in sendmsg(2). The *msg_len* field is used to return the number of bytes sent from the message in *msg_hdr* (i.e., the same as the return value from a single sendmsg(2) call).

The *flags* argument contains flags ORed together. The flags are the same as for sendmsg(2).

A blocking **sendmmsg**() call blocks until *n* messages have been sent. A nonblocking call sends as many messages as possible (up to the limit specified by *n*) and returns immediately.

On return from **sendmmsg**(), the *msg_len* fields of successive elements of *msgvec* are updated to contain the number of bytes transmitted from the corresponding *msg_hdr*. The return value of the call indicates the number of elements of *msgvec* that have been updated.

RETURN VALUE top

On success, **sendmmsg**() returns the number of messages sent from *msgvec*; if this is less than *n*, the caller can retry with a further **sendmmsg**() call to send the remaining messages.

On error, -1 is returned, and *errno* is set to indicate the error.

ERRORS top

Errors are as for sendmsg(2). An error is returned only if no datagrams could be sent. See also BUGS.

STANDARDS top

Linux.

HISTORY top

Linux 3.0, glibc 2.14.

NOTES top

The value specified in *n* is capped to **UIO_MAXIOV** (1024).

BUGS top

If an error occurs after at least one message has been sent, the call succeeds, and returns the number of messages sent. The error

code is lost. The caller can retry the transmission, starting at the first failed message, but there is no guarantee that, if an error is returned, it will be the same as the one that was lost on the previous call.

EXAMPLES top

The example below uses **sendmmsg**() to send *onetwo* and *three* in two distinct UDP datagrams using one system call. The contents of the first datagram originates from a pair of buffers.

```
#define _GNU_SOURCE
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>

int
main(void)
{
    int                 retval;
    int                 sockfd;
    struct iovec        msg1[2], msg2;
    struct mmsghdr      msg[2];
    struct sockaddr_in  addr;

    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sockfd == -1) {
        perror("socket()");
        exit(EXIT_FAILURE);
    }

    addr.sin_family = AF_INET;
    addr.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
    addr.sin_port = htons(1234);
    if (connect(sockfd, (struct sockaddr *) &addr, sizeof(addr)) == -1) {
        perror("connect()");
        exit(EXIT_FAILURE);
    }

    memset(msg1, 0, sizeof(msg1));
    msg1[0].iov_base = "one";
    msg1[0].iov_len = 3;
    msg1[1].iov_base = "two";
    msg1[1].iov_len = 3;

    memset(&msg2, 0, sizeof(msg2));
```

```
        msg2.iov_base = "three";
        msg2.iov_len = 5;

        memset(msg, 0, sizeof(msg));
        msg[0].msg_hdr.msg_iov = msg1;
        msg[0].msg_hdr.msg_iovlen = 2;

        msg[1].msg_hdr.msg_iov = &msg2;
        msg[1].msg_hdr.msg_iovlen = 1;

        retval = sendmmsg(sockfd, msg, 2, 0);
        if (retval == -1)
            perror("sendmmsg()");
        else
            printf("%d messages sent\n", retval);

        exit(0);
    }
```

SEE ALSO top

recvmmsg(2), sendmsg(2), socket(2), socket(7)

COLOPHON top

This page is part of the *man-pages* (Linux kernel and C library user-space interface documentation) project. Information about the project can be found at ⟨https://www.kernel.org/doc/man-pages/⟩. If you have a bug report for this manual page, see ⟨https://git.kernel.org/pub/scm/docs/man-pages/man-pages.git/tree/CONTRIBUTING⟩. This page was obtained from the tarball man-pages-6.16.tar.gz fetched from ⟨https://mirrors.edge.kernel.org/pub/linux/docs/man-pages/⟩ on 2026-01-16. If you discover any rendering problems in this HTML version of the page, or you believe there is a better or more up-to-date source for the page, or you have corrections or improvements to the information in this COLOPHON (which is *not* part of the original manual page), send a mail to man-pages@man7.org