NAME

getsockopt, setsockopt − get and set options on sockets

SYNOPSIS

```
#include <sys/types.h>        /* See NOTES */
#include <sys/socket.h>

int getsockopt(int sockfd, int level, int optname,
               void *optval, socklen_t *optlen);
int setsockopt(int sockfd, int level, int optname,
               const void *optval, socklen_t optlen);
```

DESCRIPTION

getsockopt() and **setsockopt**() manipulate options for the socket referred to by the file descriptor *sockfd*. Options may exist at multiple protocol levels; they are always present at the uppermost socket level.

When manipulating socket options, the level at which the option resides and the name of the option must be specified. To manipulate options at the sockets API level, *level* is specified as **SOL_SOCKET**. To manipulate options at any other level the protocol number of the appropriate protocol controlling the option is supplied. For example, to indicate that an option is to be interpreted by the **TCP** protocol, *level* should be set to the protocol number of **TCP**; see **getprotoent**(3).

The arguments *optval* and *optlen* are used to access option values for **setsockopt**(). For **getsockopt**() they identify a buffer in which the value for the requested option(s) are to be returned. For **getsockopt**(), *optlen* is a value-result argument, initially containing the size of the buffer pointed to by *optval*, and modified on return to indicate the actual size of the value returned. If no option value is to be supplied or returned, *optval* may be NULL.

Optname and any specified options are passed uninterpreted to the appropriate protocol module for interpretation. The include file *<sys/socket.h>* contains definitions for socket level options, described below. Options at other protocol levels vary in format and name; consult the appropriate entries in section 4 of the manual.

Most socket-level options utilize an *int* argument for *optval*. For **setsockopt**(), the argument should be nonzero to enable a boolean option, or zero if the option is to be disabled.

For a description of the available socket options see **socket**(7) and the appropriate protocol man pages.

RETURN VALUE

On success, zero is returned for the standard options. On error, −1 is returned, and *errno* is set appropriately.

Netfilter allows the programmer to define custom socket options with associated handlers; for such options, the return value on success is the value returned by the handler.

ERRORS

EBADF The argument *sockfd* is not a valid file descriptor.

EFAULT The address pointed to by *optval* is not in a valid part of the process address space. For **getsockopt**(), this error may also be returned if *optlen* is not in a valid part of the process address space.

EINVAL *optlen* invalid in **setsockopt**(). In some cases this error can also occur for an invalid value in *optval* (e.g., for the **IP_ADD_MEMBERSHIP** option described in **ip**(7)).

ENOPROTOOPT
The option is unknown at the level indicated.

ENOTSOCK
The file descriptor *sockfd* does not refer to a socket.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, SVr4, 4.4BSD (these system calls first appeared in 4.2BSD).

NOTES

POSIX.1 does not require the inclusion of *<sys/types.h>*, and this header file is not required on Linux. However, some historical (BSD) implementations required this header file, and portable applications are probably wise to include it.

For background on the *socklen_t* type, see **accept**(2).

BUGS

Several of the socket options should be handled at lower levels of the system.

SEE ALSO

ioctl(2), **socket**(2), **getprotoent**(3), **protocols**(5), **ip**(7), **packet**(7), **socket**(7), **tcp**(7), **udp**(7), **unix**(7)

COLOPHON

This page is part of release 5.02 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.