Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'socket.3p' command

$ man socket.3p

SOCKET(3P) POSIX Programmer's Manual SOCKET(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

socket ? create an endpoint for communication

SYNOPSIS

#include <sys/socket.h>

int socket(int domain, int type, int protocol);

DESCRIPTION

The socket() function shall create an unbound socket in a communica? tions domain, and return a file descriptor that can be used in later function calls that operate on sockets. The file descriptor shall be allocated as described in Section 2.14, File Descriptor Allocation.

The socket() function takes the following arguments:

domain Specifies the communications domain in which a socket is to be created.

type Specifies the type of socket to be created.

protocol Specifies a particular protocol to be used with the socket. Specifying a protocol of 0 causes socket() to use an un? specified default protocol appropriate for the requested

socket type.

The domain argument specifies the address family used in the communica?
tions domain. The address families supported by the system are imple?
mentation-defined.

Symbolic constants that can be used for the domain argument are defined in the <sys/socket.h> header.

The type argument specifies the socket type, which determines the se?
mantics of communication over the socket. The following socket types are defined; implementations may specify additional socket types:

SOCK_STREAM Provides sequenced, reliable, bidirectional, connection-mode byte streams, and may provide a transmission mechanism for out-of-band data.

SOCK_DGRAM Provides datagrams, which are connectionless-mode, unreli?able messages of fixed maximum length.

SOCK_SEQPACKET

Provides sequenced, reliable, bidirectional, connection-mode transmission paths for records. A record can be sent using one or more output operations and received using one or more input operations, but a single operation never transfers part of more than one record. Record boundaries are visible to the receiver via the MSG_EOR flag.

If the protocol argument is non-zero, it shall specify a protocol that is supported by the address family. If the protocol argument is zero, the default protocol for this address family and type shall be used.

The protocols supported by the system are implementation-defined.

The process may need to have appropriate privileges to use the socket() function or to create some sockets.

RETURN VALUE

Upon successful completion, socket() shall return a non-negative inte?ger, the socket file descriptor. Otherwise, a value of -1 shall be re?turned and errno set to indicate the error.

ERRORS

The socket() function shall fail if:

EAFNOSUPPORT

The implementation does not support the specified address fam? ily.

EMFILE All file descriptors available to the process are currently open.

ENFILE No more file descriptors are available for the system.

EPROTONOSUPPORT

The protocol is not supported by the address family, or the pro? tocol is not supported by the implementation.

EPROTOTYPE

The socket type is not supported by the protocol.

The socket() function may fail if:

EACCES The process does not have appropriate privileges.

ENOBUFS

Insufficient resources were available in the system to perform the operation.

ENOMEM Insufficient memory was available to fulfill the request.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

The documentation for specific address families specifies which proto? cols each address family supports. The documentation for specific pro? tocols specifies which socket types each protocol supports.

The application can determine whether an address family is supported by trying to create a socket with domain set to the protocol in question.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.14, File Descriptor Allocation, accept(), bind(), connect(), getsockname(), getsockopt(), listen(), recv(), recvfrom(), recvmsg(),

send(), sendmsg(), setsockopt(), shutdown(), socketpair()

The Base Definitions volume of POSIX.1?2017, <netinet_in.h>, <sys_socket.h>

COPYRIGHT