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 'bind.3p' command

$ man bind.3p

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

bind ? bind a name to a socket

SYNOPSIS

```
#include <sys/socket.h>
int bind(int socket, const struct sockaddr *address,
    socklen_t address_len);
```

DESCRIPTION

The bind() function shall assign a local socket address address to a socket identified by descriptor socket that has no local socket address assigned. Sockets created with the socket() function are initially un? named; they are identified only by their address family.

The bind() function takes the following arguments:

socket Specifies the file descriptor of the socket to be bound.

address Points to a sockaddr structure containing the address to be bound to the socket. The length and format of the address depend on the address family of the socket.

address_len Specifies the length of the sockaddr structure pointed to

by the address argument.

The socket specified by socket may require the process to have appro?
priate privileges to use the bind() function.

If the address family of the socket is AF_UNIX and the pathname in ad?
dress names a symbolic link, bind() shall fail and set errno to [EAD?
DRINUSE].

If the socket address cannot be assigned immediately and O_NONBLOCK is set for the file descriptor for the socket, bind() shall fail and set errno to [EINPROGRESS], but the assignment request shall not be aborted, and the assignment shall be completed asynchronously. Subse?
quent calls to bind() for the same socket, before the assignment is completed, shall fail and set errno to [EALREADY].

When the assignment has been performed asynchronously, pselect(), se?
lect(), and poll() shall indicate that the file descriptor for the socket is ready for reading and writing.

RETURN VALUE

Upon successful completion, bind() shall return 0; otherwise, -1 shall be returned and errno set to indicate the error.

ERRORS

The bind() function shall fail if:

EADDRINUSE

The specified address is already in use.

EADDRNOTAVAIL

The specified address is not available from the local machine.

EAFNOSUPPORT

The specified address is not a valid address for the address family of the specified socket.

EALREADY

An assignment request is already in progress for the specified socket.

EBADF The socket argument is not a valid file descriptor.

EINPROGRESS

O_NONBLOCK is set for the file descriptor for the socket and the

assignment cannot be immediately performed; the assignment shall be performed asynchronously.

EINVAL The socket is already bound to an address, and the protocol does not support binding to a new address; or the socket has been shut down.

ENOBUFS

Insufficient resources were available to complete the call.

ENOTSOCK

The socket argument does not refer to a socket.

EOPNOTSUPP

The socket type of the specified socket does not support binding to an address.

If the address family of the socket is AF_UNIX, then bind() shall fail if:

EACCES A component of the path prefix denies search permission, or the requested name requires writing in a directory with a mode that denies write permission.

EDESTADDRREQ or EISDIR

The address argument is a null pointer.

EIO An I/O error occurred.

ELOOP A loop exists in symbolic links encountered during resolution of the pathname in address.

ENAMETOOLONG

The length of a component of a pathname is longer than {NAME_MAX}.

ENOENT A component of the path prefix of the pathname in address does not name an existing file or the pathname is an empty string.

ENOENT or ENOTDIR

The pathname in address contains at least one non-<slash> char? acter and ends with one or more trailing <slash> characters. If the pathname without the trailing <slash> characters would name an existing file, an [ENOENT] error shall not occur.

ENOTDIR

A component of the path prefix of the pathname in address names an existing file that is neither a directory nor a symbolic link to a directory, or the pathname in address contains at least one non-<slash> character and ends with one or more trailing <slash> characters and the last pathname component names an existing file that is neither a directory nor a symbolic link to a direc? tory.

EROFS The name would reside on a read-only file system.

The bind() function may fail if:

EACCES The specified address is protected and the current user does not have permission to bind to it.

EINVAL The address_len argument is not a valid length for the address family.

EISCONN

The socket is already connected.

ELOOP More than {SYMLOOP_MAX} symbolic links were encountered during resolution of the pathname in address.

ENAMETOOLONG

The length of a pathname exceeds {PATH_MAX}, or pathname resolu? tion of a symbolic link produced an intermediate result with a length that exceeds {PATH_MAX}.

The following sections are informative.

EXAMPLES

The following code segment shows how to create a socket and bind it to a name in the AF_UNIX domain.

```
#define MY_SOCK_PATH "/somepath"
int sfd;
struct sockaddr_un my_addr;
sfd = socket(AF_UNIX, SOCK_STREAM, 0);
if (sfd == -1)
    /* Handle error */;
memset(&my_addr, '\0', sizeof(struct sockaddr_un));
                    /* Clear structure */
```

```
my_addr.sun_family = AF_UNIX;
strncpy(my_addr.sun_path, MY_SOCK_PATH, sizeof(my_addr.sun_path) -1);
if (bind(sfd, (struct sockaddr *) &my_addr,
    sizeof(struct sockaddr_un)) == -1)
    /* Handle error */;
```

APPLICATION USAGE

An application program can retrieve the assigned socket name with the getsockname() function.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

connect(), getsockname(), listen(), socket()

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

COPYRIGHT