NAME

stat, lstat, fstat − get file status

SYNOPSIS

#include <sys/types.h>
#include <sys/stat.h>

int stat(const char ****path*, struct stat ****buf***)**
int lstat(const char ****path*, struct stat ****buf***)** **(Minix-vmd)**
int fstat(int *fd***, struct stat ****buf***)**

DESCRIPTION

Stat obtains information about the file *path*. Read, write or execute permission of the named file is not required, but all directories listed in the path name leading to the file must be reachable.

Lstat is like **stat** except in the case where the named file is a symbolic link, in which case **lstat** returns information about the link, while **stat** returns information about the file the link references. (Minix-vmd)

Fstat obtains the same information about an open file referenced by the argument descriptor, such as would be obtained by an **open** call. Pipe descriptors look like named pipes with a link count of zero. The st_size field of pipes or named pipes shows the amount of bytes currently buffered in the pipe.

Buf is a pointer to a **stat** structure into which information is placed concerning the file. The contents of the structure pointed to by *buf* is as follows:

```
struct stat {
        dev_t       st_dev;       /* device inode resides on */
        ino_t       st_ino;       /* this inode's number */
        mode_t      st_mode;      /* file mode, protection bits, etc. */
        nlink_t     st_nlink;     /* number or hard links to the file */
        uid_t       st_uid;       /* user-id of the file's owner */
        gid_t       st_gid;       /* group-id of the file's owner */
        dev_t       st_rdev;      /* the device type, for inode that is device */
        off_t       st_size;      /* total size of file */
        time_t      st_atime;     /* time of last access */
        time_t      st_mtime;     /* time of last data modification */
        time_t      st_ctime;     /* time of last file status change */
};
```

st_atime   Time when file data was last read or modified. Changed by the following system calls: **mknod**(2), **utime**(2), **read**(2), and **write**(2). For reasons of efficiency, st_atime is not set when a directory is searched, although this would be more logical.

st_mtime   Time when data was last modified. It is not set by changes of owner, group, link count, or mode. Changed by the following system calls: **mknod**(2), **utime**(2), **write**(2).

st_ctime   Time when file status was last changed. It is set both both by writing and changing the i-node. Changed by the following system calls: **chmod**(2) **chown**(2), **link**(2), **mknod**(2), **rename**(2), **unlink**(2), **utime**(2), **write**(2).

The file type information in **st_mode** has bits:

```
#define S_IFMT        0170000     /* type of file */
#define   S_IFIFO     0010000     /* named pipe */
#define   S_IFCHR     0020000     /* character special */
#define   S_IFDIR     0040000     /* directory */
#define   S_IFBLK     0060000     /* block special */
#define   S_IFREG     0100000     /* regular */
#define   S_IFLNK     0120000     /* symbolic link (Minix-vmd) */
```

The mode bits 0007777 encode set-uid/gid bits and permission bits (see **chmod**(2)).

RETURN VALUE

Upon successful completion a value of 0 is returned. Otherwise, a value of −1 is returned and **errno** is set to indicate the error.

ERRORS

Stat and **lstat** will fail if one or more of the following are true:

[ENOTDIR] A component of the path prefix is not a directory.

[ENAMETOOLONG] The path name exceeds PATH_MAX characters.

[ENOENT] The named file does not exist.

[EACCES] Search permission is denied for a component of the path prefix.

[ELOOP] Too many symbolic links were encountered in translating the pathname. (Minix-vmd)

[EFAULT] *Buf* or *name* points to an invalid address.

[EIO] An I/O error occurred while reading from or writing to the file system.

Fstat will fail if one or both of the following are true:

[EBADF] *Fildes* is not a valid open file descriptor.

[EFAULT] *Buf* points to an invalid address.

[EIO] An I/O error occurred while reading from or writing to the file system.

SEE ALSO

chmod(2), **chown**(2), **utime**(2).