File system Internals

3B: File system Internals: stat, fstat, ustat, link/unlink,dup

Subject:- Unix Operating

System Lab Class :- TYIT

**Name : Aditi Sudhir Ghate  
PRN : 2020BTEIT00044**

**3.2** Write the program to show file statistics using the fstat system call. Take the file name / directory name from user including path. Print only inode no, UID, GID, FAP and File type only.

**Objectives:**

1. To learn about File system Internals.

**Theory:**

Name:

stat, fstat, lstat - get file status

Syntax:

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

int stat(const char \**path*, struct stat \**buf*);

int fstat(int *fd*, struct stat \**buf*);

int lstat(const char \**path*, struct stat \**buf*);

Description:

These functions return information about a file. No permissions are required on the file itself, but-in the case of stat() and lstat() - execute (search) permission is required on all of the directories in *path* that lead to the file.

stat() stats the file pointed to by *path* and fills in *buf*.

lstat() is identical to stat(), except that if *path* is a symbolic link, then the link itself is stat-ed, not the file that it refers to.

fstat() is identical to stat(), except that the file to be stat-ed is specified by the file descriptor *fd*. All of these system calls return a *stat* structure, which contains the following fields: struct stat {

dev\_t st\_dev; /\* ID of device containing file \*/ino\_t st\_ino; /\* inode number \*/ mode\_t st\_mode; /\* protection \*/

nlink\_t st\_nlink; /\* number of hard links \*/

uid\_t st\_uid; /\* user ID of owner \*/

gid\_t st\_gid; /\* group ID of owner \*/

dev\_t st\_rdev; /\* device ID (if special file) \*/

off\_t st\_size; /\* total size, in bytes \*/

blksize\_t st\_blksize; /\* blocksize for file system I/O \*/

blkcnt\_t st\_blocks; /\* number of 512B blocks allocated \*/

time\_t st\_atime; /\* time of last access \*/

time\_t st\_mtime; /\* time of last modification \*/

time\_t st\_ctime; /\* time of last status change \*/

};

**Data Dictionary:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sr Number Variable/Function Datatype Use |  |  |
| 1 | s | char[] | Get file name. |
| 2 | fp | FILE\* | Pointer to file. |
| 3 | fn | int | File descriptor number. |
| 4 | sta |  | struct stat Store information about files. |

**Program:**

#include<stdio.h>

#include<stdlib.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <unistd.h>

int main()

{

char s[100];

gets (s);

//printf("%s",s);

FILE \*fp;

//link/unlink,dup;

if((fp=fopen(s,"r"))==NULL) return 1;

int fn=0; fn=fileno(fp);

struct stat sta;

if(fstat(fn,&sta) < 0) return 1;

printf("File size : %ld\n",(long)sta.st\_size);

printf("File inode Number : %ld\n",sta.st\_ino);

printf("File UID : %ld\n",(long)sta.st\_uid);

printf("File GID : %ld\n", (long)sta.st\_gid);

printf("File Permissions: \t");

printf( (S\_ISDIR(sta.st\_mode)) ? "d" : "-");

printf( (sta.st\_mode & S\_IRUSR) ? "r" : "-");

printf( (sta.st\_mode & S\_IWUSR) ? "w" : "-");

printf( (sta.st\_mode & S\_IXUSR) ? "x" : "-");

printf( (sta.st\_mode & S\_IRGRP) ? "r" : "-");

printf( (sta.st\_mode & S\_IWGRP) ? "w" : "-");

printf( (sta.st\_mode & S\_IXGRP) ? "x" : "-");

printf( (sta.st\_mode & S\_IROTH) ? "r" : "-");

printf( (sta.st\_mode & S\_IWOTH) ? "w" : "-");

printf( (sta.st\_mode & S\_IXOTH) ? "x" : "-");

printf("\n \n ");

printf("File type: ");

printf( (sta.st\_mode & S\_IRGRP) ? "r" : "-");

printf( (sta.st\_mode & S\_IWGRP) ? "w" : "-");

printf( (sta.st\_mode & S\_IXGRP) ? "x" : "-");

printf( (sta.st\_mode & S\_IROTH) ? "r" : "-");

printf( (sta.st\_mode & S\_IWOTH) ? "w" : "-");

printf( (sta.st\_mode & S\_IXOTH) ? "x" : "-");

printf("\n \n");

printf("File type: ");

switch (sta.st\_mode & S\_IFMT)

{

case S\_IFBLK: printf("block device\n"); break; case S\_IFCHR:

printf("character device\n");

break;

case S\_IFDIR:

printf("directory\n");

break;

case S\_IFIFO:

printf("FIFO/pipe\n"); break;

case S\_IFLNK:

printf("symlink\n");

break;

case S\_IFREG:

printf("regular file\n");

break;

case S\_IFSOCK:

printf("socket\n");

break;

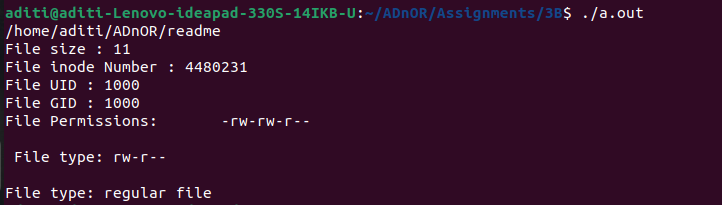
default: printf("unknown? \n");

break;

}

return 0;

}

**Output:**

**Conclusion:**

Stats of file like UID, GIDfile size,links, permissions, inode number and type of link can be retrieved using stat(),fstat() and link() and stored in a structure.

**References:**

https://www.lix.polytechnique.fr/~liberti/public/computing/prog/c/C/FUNCTIONS/stat.html