**Name: Rohan Nivas Chavan**

**PRN:2019BTEIT00084**

**UOS LAB**

**3B a. Write a program to lock the file using lockf system call. Check for mandatory locks, the file must be a regular file with the set-group-ID bit on and the group executes permission off. If either condition fails, all record locks are advisory.**

**Objectives:**

1. To learn about File locking-mandatory and advisory locking.

**Theory:**

Mandatory locking suspends a process until the requested file segments are free. Advisory locking returns a result indicating whether the lock was obtained or not. A process can ignore the result of advisory locking. You cannot use both mandatory and advisory file locking on the same file at the same time. The mode of a file at the time the file is opened determines whether locks on a file are treated as mandatory or advisory.

## Cautions About Mandatory Locking

* Mandatory locking works only for local files. Mandatory locking is not supported when accessing files through NFS.
* Mandatory locking protects only the segments of a file that are locked. The remainder of the file can be accessed according to normal file permissions.
* If multiple reads or writes are needed for an atomic transaction, the process should explicitly lock all such segments before any I/O begins. Advisory locks are sufficient for all programs that perform in this way.
* Arbitrary programs should not have unrestricted access permission to files on which record locks are used.
* Advisory locking is more efficient because a record lock check does not have to be performed for every I/O request.

Only advisory file locking is supported on NFS. File locking is not supported for the proc and fd file systems.

Only advisory file locking is supported on NFS. File locking is not supported for the proc and fd file systems.

### Opening a File for Locking

You can only request a lock for a file with a valid open descriptor. For read locks, the file must be open with at least read access. For write locks, the file must also be open with write access. In the following example, a file is opened for both read and write access.

filename = argv[1];

fd = open (filename, O\_RDWR);

if (fd < 0) {

perror(filename);

exit(2);

}

...

### Setting a File Lock

To lock an entire file, set the offset to zero and set the size to zero.

You can set a lock on a file in several ways. The choice of method depends on how the lock interacts with the rest of the program, performance, and portability. This example uses the POSIX standard-compatible [fcntl(2)](https://docs.oracle.com/docs/cd/E19683-01/816-0212/6m6nd4n9c/index.html) interface. The interface tries to lock a file until one of the following happens:

* The file lock is set successfully.
* An error occurs.
* MAX\_TRY is exceeded, and the program stops trying to lock the file

**Program:**

#include <fcntl.h>

#include <stdio.h>

struct flock lck;

lck.l\_type = F\_WRLCK; /\* setting a write lock \*/

lck.l\_whence = 0; /\* offset l\_start from beginning of file \*/

lck.l\_start = (off\_t)0;

lck.l\_len = (off\_t)0; /\* until the end of the file \*/

if (fcntl(fd, F\_SETLK, &lck) <0) {

if (errno == EAGAIN || errno == EACCES) {

(void) fprintf(stderr, "File busy try again later!\n");

return;

}

perror("fcntl");

exit (2);

}

**Conclusion:**

1. File locking-mandatory and advisory locking are studied.

**Refrences :**

https://docs.oracle.com/cd/E19683-01/806-4125/6jd7pe6bk/index.html