Ivy Truong

3/11/2022

CSCI 3800

Homework 3

1. In C file named mydup2.c

2. Text

Description automatically generated

Diagram

Description automatically generated

fd1 and fd2 point to the same file, since fd2 is a copy of fd1. fd3 points to a file of the same name, but it is a separate instance of the file since it calls open on that file name.

The descriptors that are affected by a fcntl on fd1 with a command of F\_SETFD are descriptors fd1 and fd2, since they both point to the same file, meaning that the flag changes for the file descriptor will be shown in both file descriptors. The descriptors that are affected by a fcntl on fd1 with a command of F\_SETFL are fd1, fd2, and fd3 because they all are instances of the same file, and this command sets the flags for the actual file.

3. The main difference between a soft link and a hard link is that a soft link is a pointer to a file with that name, while a hard link is another instance and exact/hard copy of the file in the file table with an alias or unique name from the file it was copied from.

Diagram

Description automatically generated

Diagram

Description automatically generated

4. When testing out the sample commands in the c program, mkdir(“~/tmp/foo”) automatically fails because it says no such file or directory. In the c program, I adjusted the file path to “/tmp/foo” and mkdir worked with making this directory.

The C file code used to simulate this is in num4.c

The directory was removed successfully when the rmdir function was executed. Below are the results for when the current directory trying to be read using the readdir function. To read the directory using readdir, the directory has to be opened first using opendir to obtain the pointer to the directory stream.

1. ‘.’ : It reads the directory successfully.
2. ‘../foo’ : It cannot open the directory, therefore reading the directory would give an error.
3. ‘/tmp/foo’ : It cannot open the directory, therefore reading the directory would give an error.

5. The main difference between fflush and fsync is that fflush is a C library function that will take any unwritten data from the output buffer and write it to the file pointed if pointer is not NULL otherwise it is flushed(to the kernel buffer), while fsync is a kernel/system call function that will take the data from the (kernel) buffer and write to the physical disk. When fflush(aFilePointer); and fsync(fileno(aFilePointer)); are called in succession, it will force to write out everything stored in the buffer to the file specified by the stdio file pointer with fflush, then fsync calls the kernel to write everything from the specified file to the disk until it has finished writing before returning the data and file attributes.

<https://stackoverflow.com/questions/2340610/difference-between-fflush-and-fsync>