Experiment No .8
Write a program to demonstrate File handling and dup system calls in linux.

By

Harshali Bhuwad

I/O System calls

- **What is the File Descriptor?**
  File descriptor is integer that uniquely identifies an open file of the process.

1. Create: Used to Create a new empty file.

Syntax in C language:

```
    int create(char *filename, mode_t mode)
```

2. open: Used to Open the file for reading, writing or both.

Syntax in C language

```
#include<sys/types.h>
#include<sys/stat.h>
#include <fcntl.h>
int open (const char* Path, int flags [, int mode ]);
```

- **flags :** How you like to use
  - **O_RDONLY**: read only, **O_WRONLY**: write only, **O_RDWR**: read and write, **O_CREAT**: create file if it doesn’t exist, **O_EXCL**: prevent creation if it already exists

3. close: Tells the operating system you are done with a file descriptor and Close the file which pointed by fd.

Syntax in C language

```
          #include <fcntl.h>
          int close(int fd);
```

4. read: From the file indicated by the file descriptor fd, the read() function reads cnt bytes of input into the memory area indicated by buf. A successful read() updates the access time for the file.

Syntax in C language

```
          size_t read (int fd, void* buf, size_t cnt);
```

Parameters:

- **fd:** file descriptor
- **buf:** buffer to read data from
- **cnt:** length of buffer

5. write: Writes cnt bytes from buf to the file or socket associated with fd. cnt should not be greater than INT_MAX (defined in the limits.h header file). If cnt is zero, write() simply returns 0 without attempting any other action.

```
#include <fcntl.h>
size_t write (int fd, void* buf, size_t cnt);
```

dup() Linux system call

- The dup() system call creates a copy of a file descriptor.
- It uses the lowest-numbered unused descriptor for the new descriptor.
- If the copy is successfully created, then the original and copy file descriptors may be used interchangeably.
- They both refer to the same open file description and thus share file offset and file status flags.

Syntax:

```
int dup(int oldfd);
oldfd: old file descriptor whose copy is to be created.
```

```
//create a file test.txt

#include<unistd.h>
#include<stdio.h>
#include<fcntl.h>
int main()
{
        int old_fd, new_fd;
        char buff[10];
        old_fd=open("test.txt",O_RDWR);
        printf("File descriptor is %d\n",old_fd);
        read(old_fd,buff,10);//read first 10 characters using old file descriptor
         write(1,buff,10);//prints them on screen
        new_fd=dup(old_fd);//duplicates file descriptor
        printf("New file descriptor is %d\n",new_fd);
        read(new_fd,buff,10);//this read will read the next 10 characters even if new file
descriptor is used
        write(1,buff,10);
}
```