CSci 4061: Introduction to Operating Systems

Recitation - 4

I/O System Calls

October 2, 2017

TA : Rahul R. Sharma

Agenda

- Some Useful System Calls
- C Programs Using System Calls
  - lstatdemo.c - Print file type (dir, reg, link)
  - makdir.c - Create a directory
  - filecopy.c – Create a copy of a file
- Our own ls program: myls.c

System Calls

Some Essential System Calls

- File system calls:
  - open() – opens a file
  - close() – closes a file
  - read () – read from an open file descriptor
  - write() – write to an open file descriptor
  - lseek() – moves file pointer to desired position
  - stat() – get metadata of a file (from file inode)

Some Essential System Calls

- Directory system calls:
  - mkdir() – creates a directory
  - readdir() – reads a directory
  - opendir() – opens a directory
  - getcwd() – returns current working directory
  - scandir() – scans a directory

C programs using System Calls

makedir.c

mkdir() system call is used to create a directory

Check man page: ***man -s 2 mkdir***

makedir.c - Creates a directory in C using mkdir() system call

cc makedir.c -o makedir

makedir.c (contd.)

```
int main (int argc, char* argv[]) {
mode_t  perms =  0740;
if ( argc != 2) {
     printf( "Usage: %s Directory-Path-Name\n ", argv[0] );
     exit(1);
 }
 if ( (mkdir(argv[1], perms)) == 0 ) {
     printf( "New directory created successfully.\n");
 }
  else  {
     perror("Error in directory creation");
     exit(1);
  }
}
```

lstatdemo.c

- Prints file type (dir, regular, link) for each file in a directory
- *Check man page:* ***man lstat***
- *cc lstatdemo.c -o lstatdemo*
- Run as ./lstatdemo <input-directory>

lstatdemo.c (contd.)

```
#define NAMESIZE 256

int main(int argc, char *argv[])
{
    char *dirname;
    struct stat statbuf;
    DIR *dp;
    struct dirent *direntry;
    int totalsum = 0;
    dirname = (char*)malloc(NAMESIZE*sizeof(char));

     if(argc < 2)       /*If the user does not enter any directory name*/
     {
    printf("No directory name specified. Executing the function in the current directory.\n");
    dirname = getcwd(dirname,NAMESIZE);
     }
     else               /* If the user enters a directory name */
     {
    dirname = getcwd(dirname,NAMESIZE);
            strcat(dirname,"/");
            strcat(dirname,argv[1]);

     }
```

lstatdemo.c (contd.)

```
    stat(dirname,&statbuf);

    if(!(S_ISDIR(statbuf.st_mode))){
        printf("The directory name is not valid. Directory does not exist\n");
        exit(1);
    }

    if((dp=opendir(dirname))==NULL){
        perror("Error while opening the directory");
        exit(1);
    }
    /* Loop through the directory structure */
    chdir(dirname); //previously missing
    while( (direntry = readdir(dp)) != NULL )
    {
      lstat(direntry->d_name,&statbuf);

      if (S_ISDIR(statbuf.st_mode)) {
   printf("Dir: %s\n",direntry->d_name);
      }
      if (S_ISREG(statbuf.st_mode)) {
   printf("Reg: %s\n",direntry->d_name);
      }
      if (S_ISLNK(statbuf.st_mode)) {
   printf("Lnk: %s\n",direntry->d_name);
      }
    }
}
```

filecopy.c

- It creates a copy of an existing file

- Check man page: man -s 2 open
  man -s 2 read

cc filecopy.c -o filecopy

- ./filecopy <src-file> <dest-file>

filecopy.c (contd.)

```
#define BUFSIZE  256
void  main (int argc, char * argv[])  {
char  buffer[BUFSIZE];
ssize_t  count;
mode_t   perms;
int fdin,  fdout;
perms    = 0740;
if  ( argc !=3 ) {
   printf( "Incorrect use: Usage: %s  source-file-name  target-copy-name\n",  argv[0] );
   exit( 1 );
}
if  ( (fdin = open ( argv[1],  O_RDONLY))  == -1) {
   perror ( "Error in opening the input file:");
   exit (2);
}
if  ( (fdout = open (argv[2], (O_WRONLY | O_CREAT),  perms)) == -1 ) {
   perror ( "Error in creating the output file:");
   exit (3);
}
```

filecopy.c (contd.)

```
while ( (count=read(fdin, buffer, BUFSIZE)) > 0 ) {
       if ( write (fdout, buffer, count) != count ){
          perror ("Error in writing" );
            exit(4);
       }
}

if ( count == -1 ) {
     perror ( "Error while reading the input file: ");
     exit(5);
}

close(fdin);
close(fdout);
}
```

filecopy.c

Exercise Problem:

1. Modify the filecopy.c program to count the total number of bytes written to the copy file.

2. Print this count when copying is complete.

3. Verify this count using the filesize value obtained using the “ls” command.

Our own ls program

myls.c

Problem:

Write a program that takes a director name as argument, and prints the name of files in that directory. If no directory name is given then it looks into the current directory.

Program:

```
int main (int argc,  char* argv[] ) {
DIR   *dpntr;
struct  dirent   *dentry;
   if ( argc > 2 ) {
      printf ("Usage:  %s  [directory-name]\n", argv[0] );
      exit( 1 );
   }
   if ( argc ==  2 )
      dpntr =  opendir ( argv[1] );
   else
      dpntr =  opendir ( "." );
```

myls.c

```
if ( dpntr ==  0 ) {
      fprintf (stderr, "Error in opening directory:  %s\n",
argv[1] );
      perror( "Could not opne directory");
      exit(2);
   }

   dentry =   readdir ( dpntr );

   while ( dentry != 0 )  {
      printf( " %s\n",  dentry->d_name );
      dentry =   readdir ( dpntr );
   }
}
```

Other useful programs

You should study the codes of the following program.

- getcwd.c
- myfind.c
- statdemo.c
- access.c