**ERROR HANDLING**

When a system function encounters an error, it notifies the caller about the error using a negative value -1. There is no additional information about the error. But additional information can be viewed through the global integer variable errno, and is usually set by the system call to a constant value that gives additional information. Note: We don't set this value, it is set by the system.

On Linux, the error constants are listed in the errno manual page. You can also view the values in the header file <errno.h>. Valid error numbers are all nonzero; errno is never set to zero by any system call or library function.

errno is defined by the ISO C standard to be a modifiable lvalue of type int, and must not be explicitly declared like this;

extern int errno;

The value returned is valid only immediately after an errno-setting function indicates an error (usually by returning -1). Developers should quickly process this error and make suitable action. Because the variable can be modified during the successful execution of another or this function.

The errno variable may be read or written directly; it is a modifiable lvalue. The value of errno maps to the textual description of a specific error. A preprocessor #define also maps to the numeric errno value. For example, the preprocessor define EACCESS equals 1

Some examples of the description mapped to errno are

EACCES Permission denied (POSIX.1)

EISDIR Is a directory (POSIX.1)

ENAMETOOLONG Filename too long (POSIX.1)

The C library provides a handful of functions for translating an errno value to the corresponding textual representation. This is needed only for error reporting, and the like; checking and handling errors can be done using the preprocessor defines and errno directly.

The two functions we need to be aware are

#include <string.h>

1. char \*strerror(int errnum);

Returns: pointer to message string

This function takes the errnum argument, which is typically the errno value and returns a pointer to the string.

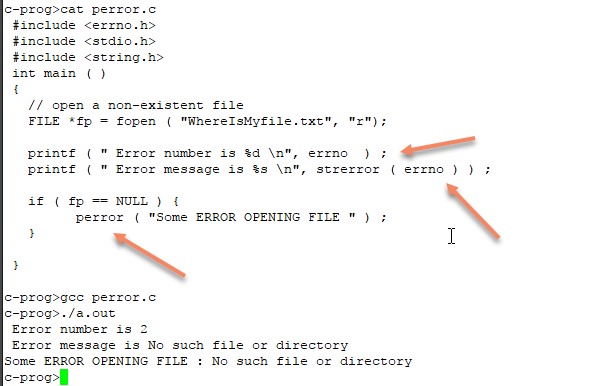
The other function , perror function produces an error message on the standard error, based on the current value of errno, and returns.

#include <stdio.h>

2. void perror(const char \*msg) ;

This outputs the string (usually passed by the application) pointed to by msg, followed by a colon and a space, followed by the error message corresponding to the value of errno, followed by a newline.

Here is the sample output



Another example is :

