medium.com /@joshuaudayagiri/linux-system-calls-open-e194e6fe2395

Linux System calls: open

Joshua U ⋮⋮ 8/9/2023

Joshua U

This system call is used to open a file and obtain a file descriptor. `open` function is a key file-related system call in Linux that is used to open a file and obtain a file descriptor, which is a unique identifier for the opened file. This descriptor is then used in subsequent file operations. The `open` function provides a wide range of options to specify file access mode, permissions, and flags.

Syntax:

```
;
```

Parameters:

`pathname` : Path to the file you want to open.

`flags` : Flags that specify the file access mode and behavior. Flags can be combined using the Bit-wise OR operator.

`mode` : Permissions to set if file is created.

Return value:

- On success the `open` function returns a non-negative integer, which is the file descriptor.
- On failure, it returns `-1` , and you can use the `errno` variable to determine the specific error.

Here are some common flags that can be used with the `open` function:

- `O_RDONLY` : Open for read-only
- `O_WRONLY` : Open for write-only
- `O_RDWR` : Open for read and write.
- `O_CREAT` : Create the file if it doesn’t exist.
- `O_TRUNC` : Truncate the file to zero length if it exists.
- `O_APPEND` : Append data to the end of the file.
- `O_EXCL` : Used with `O_CREAT` , returns an error if the file already exists.

Example:

```
#
#
#
```

```
intmain() {
constchar *filename = "example.txt";


// Open the file for writing, create if it doesn't exist, and set permissions
to 0644
int fd = open(filename, O_WRONLY | O_CREAT | O_TRUNC, 0644);


if (fd == -1) {
     perror("Error opening file");
return1;
 }


// Write data to the file
constchar *data = "Hello, world!\n";
ssize_t bytes_written = write(fd, data, strlen(data));


if (bytes_written == -1) {
     perror("Error writing to file");
return1;
 }


// Close the file
if (close(fd) == -1) {
     perror("Error closing file");
return1;
 }


printf("File opened, written, and closed successfully.\n");

  ;}
```

In this example, the program opens a file named `example.txt`, writes data to it, and then closes the file. The `O_WRONLY` flag indicates that the file is opened for write-only access, `O_CREAT` creates the file if it doesn’t exist, and `O_TRUNC` truncates the file if it exists. The permissions are set to `0644`, allowing read and write access for the owner and read-only access for others.