GeeksforGeeks
A computer science portal for geeks

Custom Search

COURSES Login

HIRE WITH US

dup() and dup2() Linux system call

dup() and dup2() Linux system call

Check if given Preorder, Inorder and Postorder traversals are of same tree | Set 2

Difference between pointer to an array and array of pointers

Count substrings that contain all vowels | SET 2

▲

C program to sort an array using pointers

Basic Code Optimizations in C

dot (.) operator in C/C++

Features and Use of Pointers in C/C++

How can we use Comma operator in place of curly braces?

OpenMP | Hello World program

Difference between while and do-while loop in C, C++, Java

Sum of an array using MPI

__builtin_inf() functions of

▲

GCC compiler

C Program to count the Number of Characters in a File

time.h header file in C with Examples

scanf("%[^\n]s", str) Vs gets(str) in C with Examples

AKTU (UPTU) Previous Year Solved Papers | C Programming

Constants vs Variables in C language

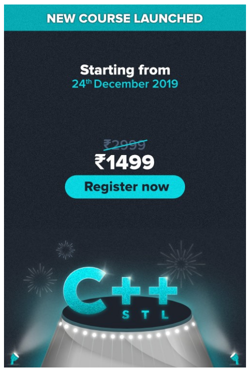
Analyzing BufferOverflow with GDB

C program to Insert an element in an Array

Types of

▲

▲



dup() and dup2() Linux system call

dup()

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:

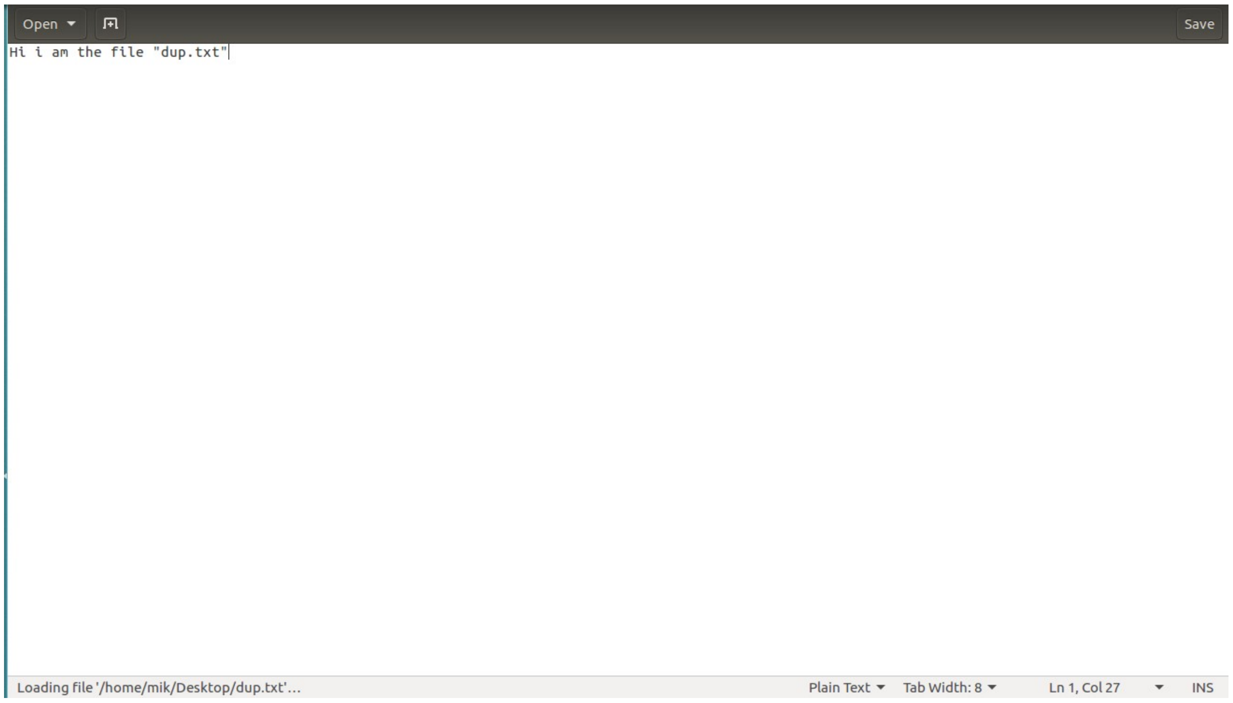
```
int dup(int oldfd);
oldfd: old file descriptor whose copy is to be created.
```

```
// CPP program to illustrate dup()
#include<stdio.h>
#include <unistd.h>
#include <fcntl.h>

int main()
{
    // open() returns a file descriptor file_desc to a
    // the file "dup.txt" here"

    int file_desc = open("dup.txt", O_WRONLY | O_APPEND);

    if(file_desc < 0)
        printf("Error opening the file\n");

    // dup() will create the copy of file_desc as the copy_desc
    // then both can be used interchangeably.

    int copy_desc = dup(file_desc);

    // write() will write the given string into the file
    // referred by the file descriptors

    write(copy_desc,"This will be output to the file named dup.txt\n",

    write(file_desc,"This will also be output to the file named dup.tx

    return 0;
}
```

Note that this program will not run in the online compiler as it includes opening a file and writing on it.

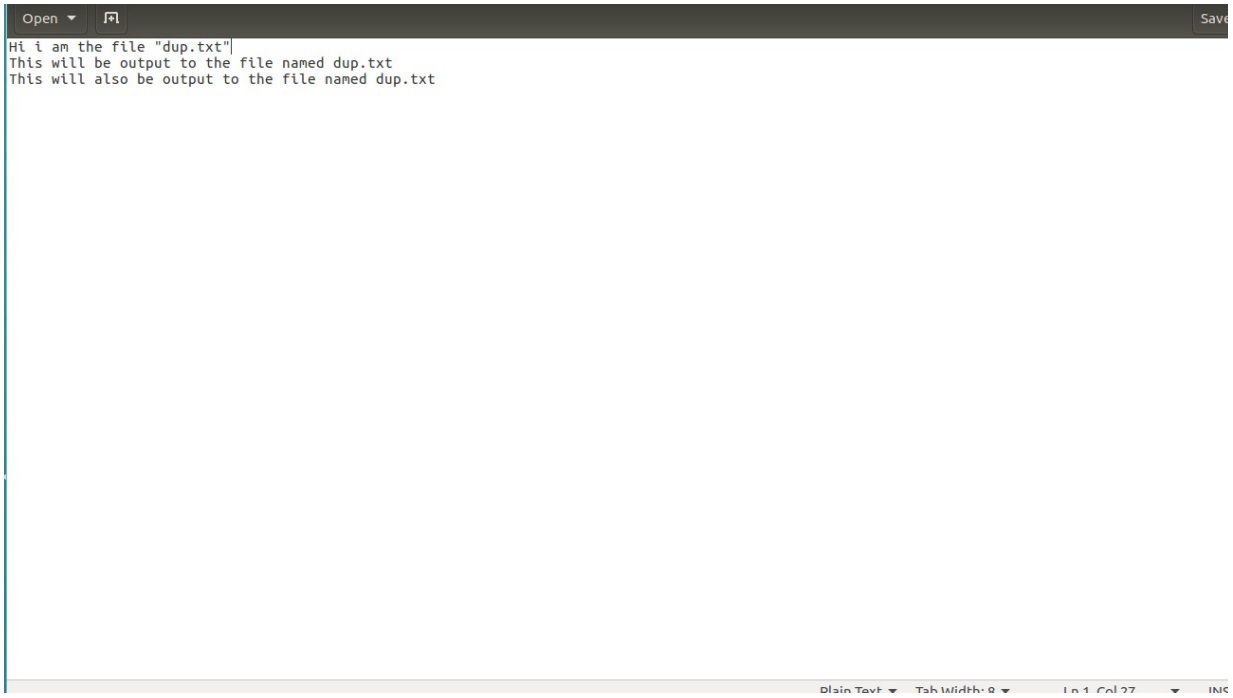
Explanation: The open() returns a file descriptor file_desc to the file named “dup.txt”. file_desc can be used to do some file operation with file “dup.txt”. After using the dup() system call, a copy of file_desc is created copy_desc. This copy can also be used to do some file operation with the same file “dup.txt”. After two write operations one with file_desc and another with copy_desc, same file is edited i.e. “dup.txt”.

Before running the code, Let The file “dup.txt” before the write operation be as shown below:



```
Hi i am the file "dup.txt"
```

After running the C program shown above, the file “dup.txt” is as shown below:



```
Hi i am the file "dup.txt"
This will be output to the file named dup.txt
This will also be output to the file named dup.txt
```

dup2()

The dup2() system call is similar to dup() but the basic difference between them is that instead of using the lowest-numbered unused file descriptor, it uses the descriptor number specified by the user.

Syntax:

```
int dup2(int oldfd, int newfd);
oldfd: old file descriptor
newfd new file descriptor which is used by dup2() to create a copy.
```

Important points:

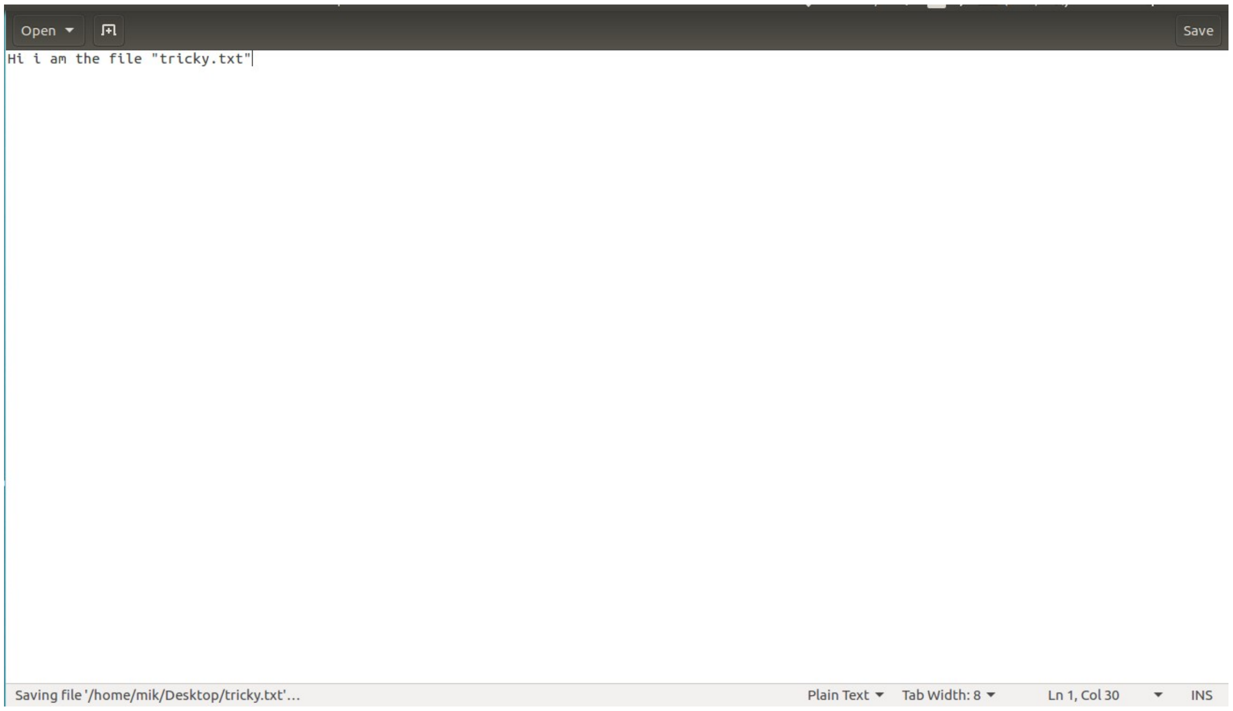
- Include the header file unistd.h for using dup() and dup2() system call.
- If the descriptor newfd was previously open, it is silently closed before being reused.
- If oldfd is not a valid file descriptor, then the call fails, and newfd is not closed.
- If oldfd is a valid file descriptor, and newfd has the same value as oldfd, then dup2() does
  nothing, and returns newfd.

A tricky use of dup2() system call: As in dup2(), in place of newfd any file descriptor can be put. Below is a C implementation in which the file descriptor of Standard output (stdout) is used. This will lead all the printf() statements to be written in the file referred by the old file descriptor.

```
// CPP program to illustrate dup2()
#include<stdlib.h>
#include<unistd.h>
#include<stdio.h>
#include<fcntl.h>

int main()
{
    int file_desc = open("tricky.txt",O_WRONLY | O_APPEND);

    // here the newfd is the file descriptor of stdout (i.e. 1)
    dup2(file_desc, 1) ;

    // All the printf statements will be written in the file
    // "tricky.txt"
    printf("I will be printed in the file tricky.txt\n");

return 0;
}
```

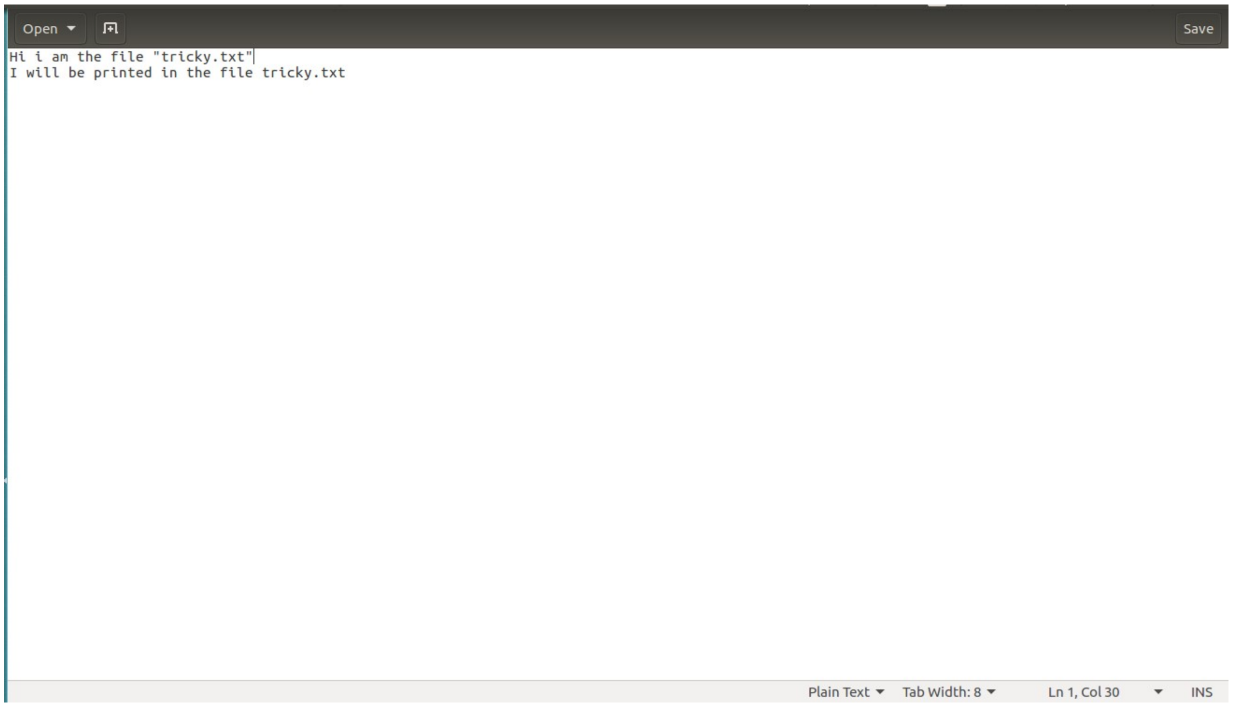
This can be seen in the figure shown below:

Let The file “tricky.txt” before the dup2() operation be as shown below:



```
Hi i am the file "tricky.txt"
```

After running the C program shown above, the file "tricky.txt" is as shown below:



```
Hi i am the file "tricky.txt"
I will be printed in the file tricky.txt
```

Reference: dup(2) – Linux man page

This article is contributed by **MAZHAR IMAM KHAN**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeks-forgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article a pearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

pipe() System call

Accept system call

Wait System Call in C

Difference between Call by Value and Call by Reference

What is Linux System Administration?

Groups in Linux System Administration

proc file system in Linux

Users in Linux System Administration

SAR command in Linux to monitor system performance

Linux Operating System | CLI (Command Line Interface) and GUI (Graphic User Interface)

A C/C++ Function Call Puzzle

Can we call an undeclared function in C++?

Is it possible to call constructor and destructor explicitly?

Difference between CALL and JUMP instructions

Linux Virtualization : Linux Containers (lxc)

Article Tags : C Linux-Unix system-programming

Practice Tags : C

1

3.6

▲

☐ To-do ☐ Done

Based on **11** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

▲