**Bluffer Overflow**

***PWN***

**Challenge:**

Maybe it's your first time pwning? Can you overwrite the variable?

nc ctf.tcp1p.com 17027

Attached - dist.zip

**Approach:**

The zip file gave me a C file named chall.c, so we’re gonna go analyze it first

The main lines to focus at are these:

char buff[20];

int buff2;

.

.

.

void buffer(){

buff2 = 0;

printf("Can you get the exact value to print the flag?\n");

printf("Input: ");

fflush(stdout);

gets(buff);

if (buff2 > 5134160) {

printf("Too high!\n\n");

} else if (buff2 == 5134160){

printf("Congrats, You got the right value!\n");

system("cat flag.txt");

} else {

printf("Sad, too low! :(, maybe you can add \*more\* value 0\_0\n\n");

}

printf("\nOutput : %s, Value : %d \n", buff, buff2);

}

Let me explain in short what’s happening

At first the C program is defining a char array named buff of size 20 bytes and an integer variable named buff2

In the buffer function, it declares the value of buff2 = 0 and asks the user for an input.

We notice it takes the input using the get() function in the buff which is definitely not safe.

After the user input, it compares the value of buff2 with 5134160 and gives a correct response along with the flag if its value matches.

So our objective:

Change the value of buff2 to 5134160 by using a buffer overflow.

First, to achieve buffer overflow, as the name suggests we need to fill the buffer. The input is taken into the buff variable which is of 20 bytes so we’ll prepend 20 bytes of random text to our payload (eg: “AAAAAAAAAAAAAAAAAAAA’). Then after this text we’re going to append the characters to the payload such that its value ends up being equal to 5134160.

First we’ll convert the required decimal value into hex

5134160 = 0x4E5750 = 0x4E 0x57 0x50

In little endian it would 0x50 0x57 0x4E

We then convert it into ASCII text and get the string:

0x50 - P ; 0x57 - W ; 0x4E - N

Hence:

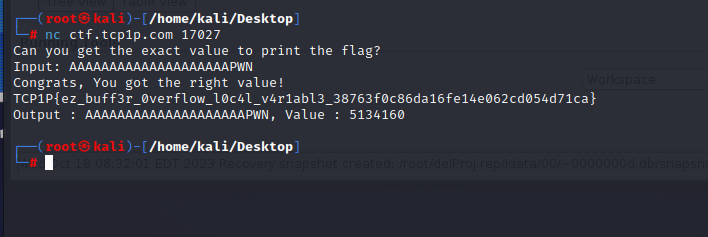
/0x50/0x57/0x4E = PWN

We’re going to append this to our payload:

“A”\*20 + PWN

Thus, final payload: AAAAAAAAAAAAAAAAAAAAPWN

We enter this payload into the program and boom:



**Flag: TCP1P{ez\_buff3r\_0verflow\_l0c4l\_v4r1abl3\_38763f0c86da16fe14e062cd054d71ca}**

Congrats!!

Happy Hacking!