Long Long Zero Sum

This one is not so much a reversing challenge because you are given the source code. You do have to exploit a "vulnerability" in the code to get the flag, though.

The challenge:

> *You'll need to connect to the port on the server below and follow the instructions you are given to get this flag. You can download the executable for local testing, and I'll even throw in the source code!*
>
> *(Keep in mind that once you have the solution you must run and submit it on the server to get the flag!)*
>
> *nc redacted-url 51742*

You're also given the binary file and the source code to download.

The source code is here:

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

long long addition(long long a, long long b) {
    unsigned long long result;
    result = a + b;
    return result;
}

void zero() {
    int c;
    FILE *file;
    file = fopen("flag.txt", "r");
    printf("\nCongrats, you've successfully broken the rules of mathematics!\n");
    printf("Here is the flag: ");
    if (file) {
        while ((c = getc(file)) != EOF)
            putchar(c);
        fclose(file);
    }
    else {
        printf("\nNo 'flags.txt' file found. If you are running this on the CTF
server\n");
        printf("please let the admin know!\n\n");
    }
}

int main() {
    setbuf(stdout, NULL);  // fixes buffer issue when using ncat
    long int first, second;
    int sum;
    printf("\nTo get the flag enter 2 positive integers that have a sum of 0.\n");
    printf("\nEnter the first number: ");
    if (scanf("%lu", &first) != 1) {
        printf("ERROR: NOT A NUMBER. GOODBYE\n");
        return 1;
    }
    if (first <= 0) {
        printf("ERROR: Not a positive integer!\n");
        return 1;
    }
    printf("Enter the second number: ");
    if (scanf("%lu", &second) != 1) {
        printf("ERROR: NOT A NUMBER. GOODBYE\n");
        return 1;
    }
    if (second <= 0) {
        printf("ERROR: Not a positive integer!\n");
        return 1;
    }
    sum = addition(first, second);
    printf("%lu + %lu = %d\n",first, second, sum);
```

```
    if (sum == 0)
        zero();
    else
        printf("That's no good!\n");
    return 0;
}
```

The 2 integers that are added together are ‘long integer’, the sum is a plain old integer. This means that the sum can be overflowed. Think of it like an old analog odometer in a car that goes from 999999 to 000000 because there aren’t enough digits to go any higher.

It is an integer overflow bug. The maximum integer size for this 64 bit binary is 2147483647. So it may take a little trial and error, but if we enter the following numbers we’ll get the flag:

```
To get the flag enter 2 positive integers that have a sum of 0.

Enter the first number: 2147483647
Enter the second number: 2147483649
2147483647 + 2147483649 = 0

Congrats, you've successfully broken the rules of mathematics!
Here is the flag: flag{two_plus_two_equals_five}
```