Writeup For Wednesday Thursday Friday

Name : Wednesday Thursday Friday

Domain : Reversing

Difficulty : Medium

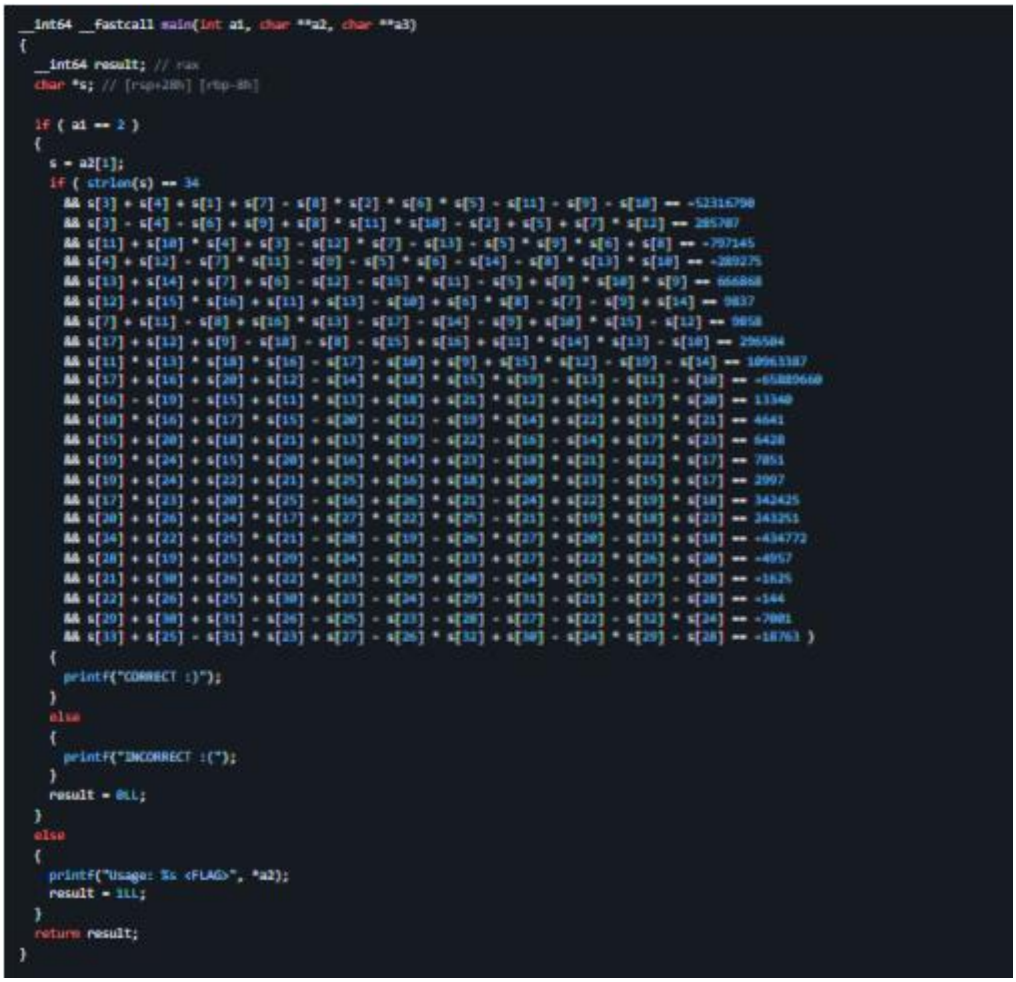
Description : Enter the Flag !!

Points : 283

Author : Yogesh Rathod, Hrishikesh Mahajan

File given : solveme

when we disassemble the elf file with IDA we will see the below code.



```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  __int64 result; // rax
  char *s; // [rsp+28h] [rbp-8h]

  if ( a1 == 2 )
  {
    s = a2[1];
    if ( strlen(s) == 34
      && s[3] + s[4] + s[1] + s[7] - s[8] * s[2] * s[6] * s[5] - s[11] - s[9] - s[10] == -52316790
      && s[3] - s[4] - s[6] + s[9] + s[8] * s[11] * s[10] - s[2] + s[5] + s[7] * s[12] == 285707
      && s[11] + s[10] * s[4] + s[3] - s[12] * s[7] - s[13] - s[5] * s[9] * s[6] + s[8] == -797145
      && s[4] + s[12] - s[7] * s[11] - s[9] - s[5] * s[6] - s[14] - s[8] * s[13] * s[10] == -289275
      && s[13] + s[14] + s[7] + s[6] - s[12] - s[15] * s[11] - s[5] + s[8] * s[10] * s[9] == 666868
      && s[12] + s[15] * s[10] + s[11] + s[13] - s[10] + s[6] * s[8] - s[7] - s[9] + s[14] == 9837
      && s[7] + s[11] - s[8] + s[16] * s[13] - s[17] - s[14] - s[9] + s[10] * s[15] - s[12] == 9858
      && s[17] + s[12] + s[9] - s[18] - s[8] - s[15] + s[16] + s[11] * s[14] * s[13] - s[10] == 296504
      && s[11] * s[13] * s[18] * s[16] - s[17] - s[10] + s[9] + s[15] * s[12] - s[19] - s[14] == 10963387
      && s[17] + s[18] + s[20] + s[12] - s[14] * s[18] * s[15] * s[19] - s[13] - s[11] - s[10] == -65889660
      && s[16] - s[19] - s[15] + s[11] * s[13] + s[18] + s[21] * s[12] + s[14] + s[17] * s[20] == 13340
      && s[18] * s[16] + s[17] * s[15] - s[20] - s[12] - s[19] * s[14] + s[22] + s[13] * s[21] == 4641
      && s[15] + s[20] + s[18] + s[21] + s[13] * s[19] - s[22] - s[16] - s[14] + s[17] * s[23] == 6428
      && s[19] * s[26] + s[15] * s[20] + s[16] * s[14] + s[23] - s[18] * s[21] - s[22] * s[17] == 7851
      && s[19] + s[24] + s[22] + s[21] + s[25] + s[16] + s[18] + s[20] * s[23] - s[15] + s[17] == 2997
      && s[17] * s[23] + s[20] * s[25] - s[16] + s[26] * s[21] - s[24] + s[22] * s[19] * s[18] == 342425
      && s[20] + s[26] + s[24] * s[17] + s[27] * s[22] * s[25] - s[21] - s[19] * s[18] + s[23] == 343251
      && s[14] + s[22] + s[25] * s[21] - s[28] - s[19] - s[26] * s[27] * s[20] - s[23] + s[18] == -434772
      && s[28] + s[19] + s[25] + s[20] - s[24] - s[21] - s[23] + s[27] - s[22] * s[26] + s[28] == -4957
      && s[21] + s[90] + s[26] + s[22] * s[23] - s[29] + s[28] - s[24] * s[25] - s[27] - s[28] == -1625
      && s[22] + s[26] + s[25] + s[30] + s[23] - s[24] - s[29] - s[31] - s[25] - s[27] + s[28] == -144
      && s[20] + s[30] + s[31] - s[26] - s[25] - s[23] - s[28] - s[27] - s[22] - s[32] * s[24] == -7001
      && s[33] + s[25] - s[31] * s[23] + s[27] - s[26] * s[32] + s[30] - s[24] * s[29] - s[28] == -18763 )
    {
      printf("CORRECT :)");
    }
    else
    {
      printf("INCORRECT :(");
    }
    result = 0LL;
  }
  else
  {
    printf("Usage: %s <FLAG>", *a2);
    result = 1LL;
  }
  return result;
}
```

ok we can see that the elf file is just checking some conditions and if all of them are true we will get CORRECT else we will get INCORRECT.

we can solve it easily using Z3. Below is a script for that

```
from z3 import *

s = []

for i in range(34):

    byte = BitVec("%s" % i, 8)

    s.append(byte)

z = Solver()

z.add(s[3] + s[4] + s[1] + s[7] - s[8] * s[2] * s[6] * s[5] - s[11] - s[9] -
s[10] == 4242650506)

z.add(s[3] - s[4] - s[6] + s[9] + s[8] * s[11] * s[10] - s[2] + s[5] + s[7] *
s[12] == 285707)

z.add(s[11] + s[10] * s[4] + s[3] - s[12] * s[7] - s[13] - s[5] * s[9] * s[6] +
s[8] == -797145)

z.add(s[4] + s[12] - s[7] * s[11] - s[9] - s[5] * s[6] - s[14] - s[8] * s[13] *
s[10] == -289275)

z.add(s[13] + s[14] + s[7] + s[6] - s[12] - s[15] * s[11] - s[5] + s[8] * s[10]
* s[9] == 666868)

z.add(s[12] + s[15] * s[16] + s[11] + s[13] - s[10] + s[6] * s[8] - s[7] - s[9]
+ s[14] == 9837)

z.add(s[7] + s[11] - s[8] + s[16] * s[13] - s[17] - s[14] - s[9] + s[10] *
s[15] - s[12] == 9858)

z.add(s[17] + s[12] + s[9] - s[18] - s[8] - s[15] + s[16] + s[11] * s[14] *
s[13] - s[10] == 296504)

z.add(s[11] * s[13] * s[18] * s[16] - s[17] - s[10] + s[9] + s[15] * s[12] -
s[19] - s[14] == 10963387)

z.add(s[17] + s[16] + s[20] + s[12] - s[14] * s[18] * s[15] * s[19] - s[13] -
s[11] - s[10] == -65889660)

z.add(s[16] - s[19] - s[15] + s[11] * s[13] + s[18] + s[21] * s[12] + s[14] +
s[17] * s[20] == 13340)

z.add(s[18] * s[16] + s[17] * s[15] - s[20] - s[12] - s[19] * s[14] + s[22] +
s[13] * s[21] == 4641)

z.add(s[15] + s[20] + s[18] + s[21] + s[13] * s[19] - s[22] - s[16] - s[14] +
s[17] * s[23] == 6428)

z.add(s[19] * s[24] + s[15] * s[20] + s[16] * s[14] + s[23] - s[18] * s[21] -
s[22] * s[17] == 7851)
```

```
z.add(s[19] + s[24] + s[22] + s[21] + s[25] + s[16] + s[18] + s[20] * s[23] - s[15] + s[17] == 2997)

z.add(s[17] * s[23] + s[20] * s[25] - s[16] + s[26] * s[21] - s[24] + s[22] * s[19] * s[18] == 342425)

z.add(s[20] + s[26] + s[24] * s[17] + s[27] * s[22] * s[25] - s[21] - s[19] * s[18] + s[23] == 243251)

z.add(s[24] + s[22] + s[25] * s[21] - s[28] - s[19] - s[26] * s[27] * s[20] - s[23] + s[18] == -434772)

z.add(s[28] + s[19] + s[25] + s[29] - s[24] - s[21] - s[23] + s[27] - s[22] * s[26] + s[20] == -4957)

z.add(s[21] + s[30] + s[26] + s[22] * s[23] - s[29] + s[20] - s[24] * s[25] - s[27] - s[28] == -1625)

z.add(s[22] + s[26] + s[25] + s[30] + s[23] - s[24] - s[29] - s[31] - s[21] - s[27] - s[28] == -144)

z.add(s[29] + s[30] + s[31] - s[26] - s[25] - s[23] - s[28] - s[27] - s[22] - s[32] * s[24] == -7001)

z.add(s[33] + s[25] - s[31] * s[23] + s[27] - s[26] * s[32] + s[30] - s[24] * s[29] - s[28] == -18763)

flag_format = "VishwaCTF{"

# check if first 10 chars will be like flag_format

for i in range(10):

    z.add(s[i] == ord(flag_format[i]))

# check if all chars will be ascii printable

for i in range(10,34):

     z.add(s[i] >= ord('!'))

     z.add(s[i] <= ord('~'))

# check if the last char will be "}"

z.add(s[-1] == ord('}'))

# check if z3 can solve it

if z.check() == sat:

    solution = z.model()

    flag = ""
```

```
    for i in range(0, 34):
        flag += chr(int(str(solution[s[i]])))
    print(flag)
#Check if z3 can't solve it
elif z.check() == unsat:
    print("Condition is not satisfied, would recommend crying: " +
str(z.check()))
```

We use the Z3 library to solve the problem.

We significantly reduce the time required to achieve the result by providing the known flag format and limiting the possibilities to printable characters.

Flag: VishwaCTF{N3V3r_60NN4_61V3_Y0U_UP}