Đầu tiên ta sẽ thực hiện mở file lib bằng ida, ta sẽ xem hàm flg

A screenshot of a computer

Description automatically generated

Tiếp tục kiểm tra hàm main

A screenshot of a computer

Description automatically generated

Ở đây ta thấy được rằng với mỗi lần thực hiện ta sẽ có 1 flag khác nhau, nên ta sẽ code lại và tìm

// Example program

#include <iostream>

#include <string>

#include <stdio.h>

int j\_j\_\_\_modsi3(int a, int b)

{

    return a % b;

}

int j\_j\_\_\_divsi3(int a, int b)

{

    return a / b;

}

char flg(int a1, char \*out)

{

    char \*v2; // r6@1

    int v3;   // ST0C\_4@1

    int v4;   // r4@1

    int v5;   // r0@1

    int v6;   // ST08\_4@1

    int v7;   // r5@1

    int v8;   // r0@1

    int v9;   // r0@1

    char v10; // ST10\_1@1

    int v11;  // r0@1

    int v12;  // r5@1

    int v13;  // r0@1

    int v14;  // ST18\_4@1

    int v15;  // r0@1

    int v16;  // r0@1

    char v17; // r0@1

    char v18; // ST04\_1@1

    int v19;  // r0@1

    char v20; // r0@1

    int v21;  // r1@1

    int v22;  // r5@1

    int v23;  // r0@1

    char v24; // r0@1

    v2 = out;

    v3 = a1;

    v4 = a1;

    v5 = j\_j\_\_\_modsi3(a1, 10);

    v6 = v5;

    v7 = 20 \* v5;

    \*v2 = 20 \* v5;

    v8 = j\_j\_\_\_divsi3(v4, 100);

    v9 = j\_j\_\_\_modsi3(v8, 10);

    v10 = v9;

    v11 = 19 \* v9 + v7;

    v2[1] = v11;

    v2[2] = v11 - 4;

    v12 = v4;

    v13 = j\_j\_\_\_divsi3(v4, 10);

    v14 = j\_j\_\_\_modsi3(v13, 10);

    v15 = j\_j\_\_\_divsi3(v4, 1000000);

    v2[3] = j\_j\_\_\_modsi3(v15, 10) + 11 \* v14;

    v16 = j\_j\_\_\_divsi3(v4, 1000);

    v17 = j\_j\_\_\_modsi3(v16, 10);

    // LOBYTE(v4) = v17;

    v4 = v17;

    v18 = v17;

    v19 = j\_j\_\_\_divsi3(v12, 10000);

    v20 = j\_j\_\_\_modsi3(v19, 10);

    v2[4] = 20 \* v4 + 60 - v20 - 60;

    v21 = -v6 - v14;

    v22 = -v21;

    v2[5] = -(char)v21 \* v4;

    v2[6] = v14 \* v4 \* v20;

    v23 = j\_j\_\_\_divsi3(v3, 100000);

    v24 = j\_j\_\_\_modsi3(v23, 10);

    v2[7] = 20 \* v24 - v10;

    v2[8] = 10 \* v18 | 1;

    v2[9] = v22 \* v24 - 1;

    v2[10] = v6 \* v14 \* v10 \* v10 - 4;

    v2[11] = (v10 + v14) \* v24 - 5;

    v2[12] = 0;

    return \*v2;

}

int main()

{

    char out[256], flag = 0;

    for (unsigned int i = 0; i <= 4294967295 - 1; ++i)

    {

        flag = 0;

        memset(out, 0, 256);

        flg(i, out);

        if (strlen(out) >= 10)

        {

            for (int j = 0; j < 12; ++j)

            {

                if ((out[j] >= 'a' && out[j] <= 'z') || (out[j] >= 'A' && out[j] <= 'Z') ||

                    (out[j] >= '0' && out[j] <= '9') || out[j] == '\_')

                    continue;

                else

                {

                    flag = 1;

                    break;

                }

            }

            if (flag == 0)

                printf("%s\n", out);

        }

    }

    std::cout << "END";

    return 0;

}

Sau đó ta thấy kết quả có sự lặp lại ở 20 chuỗi sau và không ngắt flag

PvrMb7Fv3Al1

PvrMa7iv3Al1

PvrNb7Fv3Al1

PvrNa7iv3Al1

PvrOb7Fv3Al1

PvrOa7iv3Al1

PvrPb7Fv3Al1

PvrPa7iv3Al1

PvrQb7Fv3Al1

PvrQa7iv3Al1

PvrRb7Fv3Al1

PvrRa7iv3Al1

PvrSb7Fv3Al1

PvrSa7iv3Al1

PvrTb7Fv3Al1

PvrTa7iv3Al1

PvrUb7Fv3Al1

PvrUa7iv3Al1

PvrVb7Fv3Al1

PvrVa7iv3Al1

Như vậy ta dự đoán rằng chương trình tự động generate flag liên tục và tuần tự ở 20 flag này