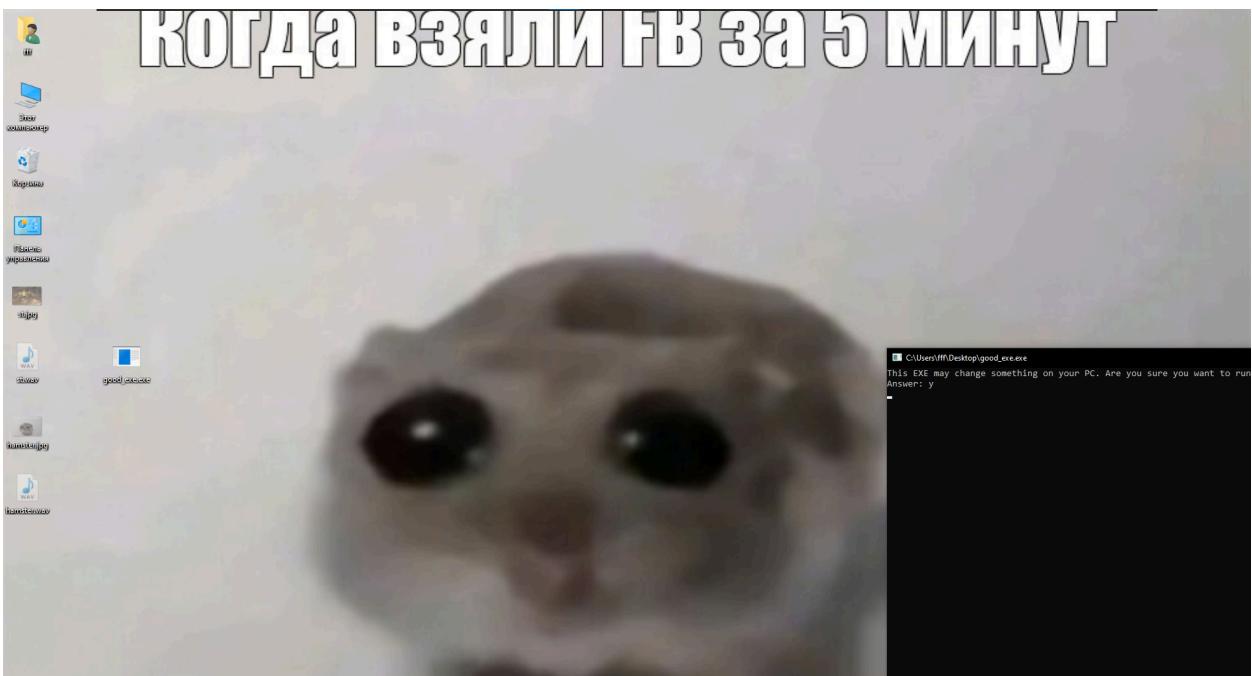
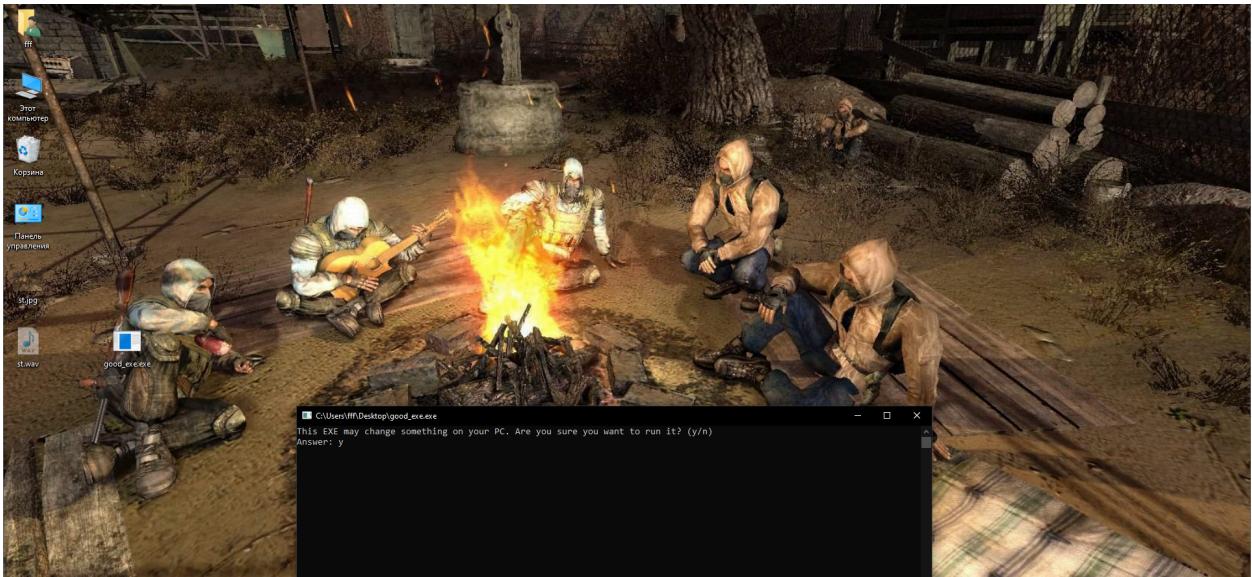


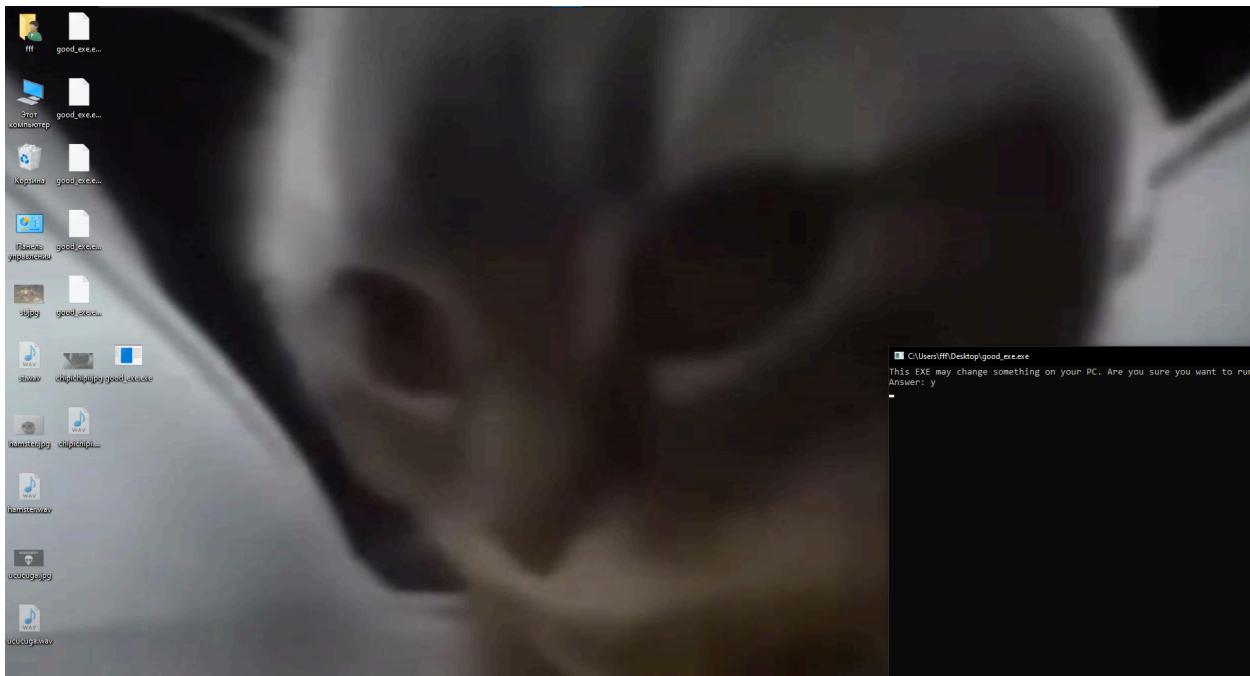
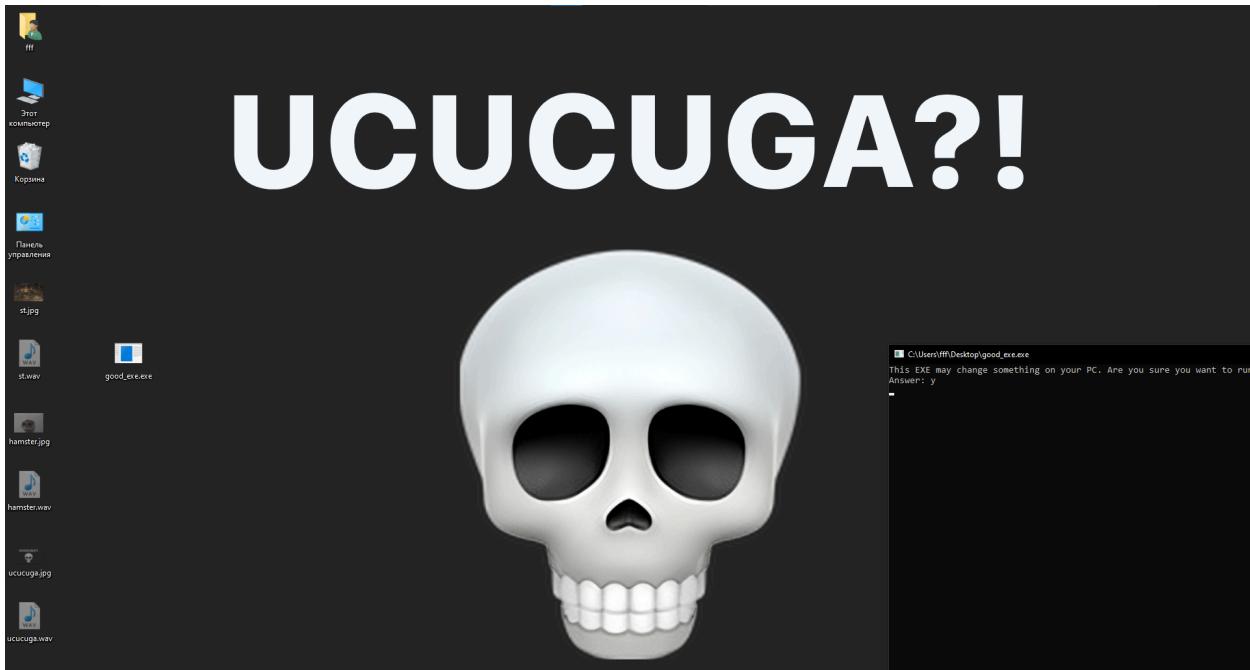


Название:	Весёлый EXE
Категория:	Реверс-инжиниринг
Уровень:	Лёгкая
Очки:	150
Описание:	Тут уже немного сложнее, но цель всё ещё проста - ввести верный флаг
Теги:	Побайтовое шифрование
Автор:	ROP

Прохождение:

Запускаем данный таск.







Он меняет обои и воспроизводит музыку .

Через IDA изучаем [main](#) .

```

int __fastcall main(int argc, const char **argv, const char **envp)
{
    FILE *v3; // rax
    FILE *v4; // rax
    char v6; // [rsp+2Bh] [rbp-5h]
    int i; // [rsp+2Ch] [rbp-4h]

    sub_4023F0(argc, argv, envp);
    byte_404020 = 32;
    puts("This EXE may change something on your PC. Are you sure you want to run it? (y/n)");
    printf("Answer: ");
    v3 = off_4052A0();
    v6 = fgetc(v3);
    v4 = off_4052A0();
    fgetc(v4);
    if ( v6 != 121 )
        return 0;
    for ( i = 0; i <= 99; ++i )
    {
        sub_401A11();
        sub_401B5E(30LL);
    }
    return 0;
}

```

```

int sub_401A11()
{
    int result; // eax
    __int64 v1; // [rsp+0h] [rbp-80h] BYREF
    struct sockaddr name; // [rsp+20h] [rbp-60h] BYREF
    SOCKET s; // [rsp+1C8h] [rbp+148h]

    result = WSAStartup(0x202u, (&v1 + 6));
    if ( !result )
    {
        s = socket(2, 1, 0);
        if ( s == -1LL )
        {
            return WSACleanup();
        }
        else
        {
            name.sa_family = 2;
            *name.sa_data = htons(0x3714u);
            if ( inet_pton(2, "62.173.140.174", &name.sa_data[2]) > 0 && connect(s, &name, 16) != -1 )
            {
                if ( sub_4015F0(s, &unk_404A00) )
                    sub_4016A7(s);
            }
            closesocket(s);
            return WSACleanup();
        }
    }
    return result;
}

00000E11|sub_401A11:1 (401A11)|_

```

```
|__int64 __fastcall sub_4015F0(int a1, const char *a2)
{
    int v2; // eax
    char buf[8]; // [rsp+25h] [rbp-Bh] BYREF
    __int16 v5; // [rsp+2Dh] [rbp-3h]
    unsigned __int8 v6; // [rsp+2Fh] [rbp-1h]

    v6 = 1;
    v2 = strlen(a2);
    if ( send(a1, a2, v2, 0) == -1 )
    {
        perror(&ErrMsg);
        v6 = 0;
    }
    *buf = 0LL;
    v5 = 0;
    if ( recv(a1, buf, 10, 0) == -1 )
        v6 = 0;
    if ( strncmp(buf, "OK", 2uLL) )
        return 0;
    return v6;
}
```

```

int __fastcall sub_4016A7(int a1)
{
    FILE *v1; // rax
    __int64 v2; // rdx
    __int64 v3; // rdx
    size_t ElementCount; // [rsp+20h] [rbp-60h]
    size_t Size; // [rsp+28h] [rbp-58h]
    CHAR v7[8]; // [rsp+30h] [rbp-50h] BYREF
    __int64 v8; // [rsp+38h] [rbp-48h]
    char FileName[8]; // [rsp+40h] [rbp-40h] BYREF
    __int64 v10; // [rsp+48h] [rbp-38h]
    char Src[1036]; // [rsp+50h] [rbp-30h] BYREF
    char buf[4]; // [rsp+45Ch] [rbp+3DCh] BYREF
    FILE *v13; // [rsp+460h] [rbp+3E0h]
    FILE *Stream; // [rsp+468h] [rbp+3E8h]
    FILE *v15; // [rsp+470h] [rbp+3F0h]
    FILE *v16; // [rsp+478h] [rbp+3F8h]
    void *v17; // [rsp+480h] [rbp+400h]
    void *Buffer; // [rsp+488h] [rbp+408h]
    int v19; // [rsp+494h] [rbp+414h]
    _QWORD *v20; // [rsp+498h] [rbp+418h]
    int v21; // [rsp+4A4h] [rbp+424h]
    void *v22; // [rsp+4A8h] [rbp+428h]
}

```

```

*buf = 0;
LODWORD(v1) = recv(a1, buf, 4, 0);
if ( v1 != -1 )
{
    v20 = malloc(*buf);
    v22 = v20;
    v21 = 0.

```

00000AA7|sub_4016A7:1 (4016A7)|

Таск загружает данные с сервера. Пока нет ничего для работы с флагом. В самих файлах флага тоже нет.

Проверим, что за пределами `main`. Кликаем на `main`, жмём X и выбираем вариант с `call`.

IDA View-A Pseudocode-A Hex View-1 Local Types

; Attributes: bp-based frame

; int __fastcall main(int argc, const char **argv, const char **envp)

main proc near

var

var

xrefs to main

Direction	Type	Address	Text
Up	p	sub_401180+242	call main
Do...	o	.pdata:00000000004070B4	RUNTIME_FUNCTION <rva sub_401FD7, \
Do...	o	.pdata:00000000004070C0	RUNTIME_FUNCTION <rva main, \

Line 1 of 3

OK Cancel Search Help

pus

mov

sub

cal

mov

lea

cal

lea rcx, Format ; "Answer: "

call printf

mov ecx, 0

mov rax, cs:off_4052A0

call rax ; sub_4038F0

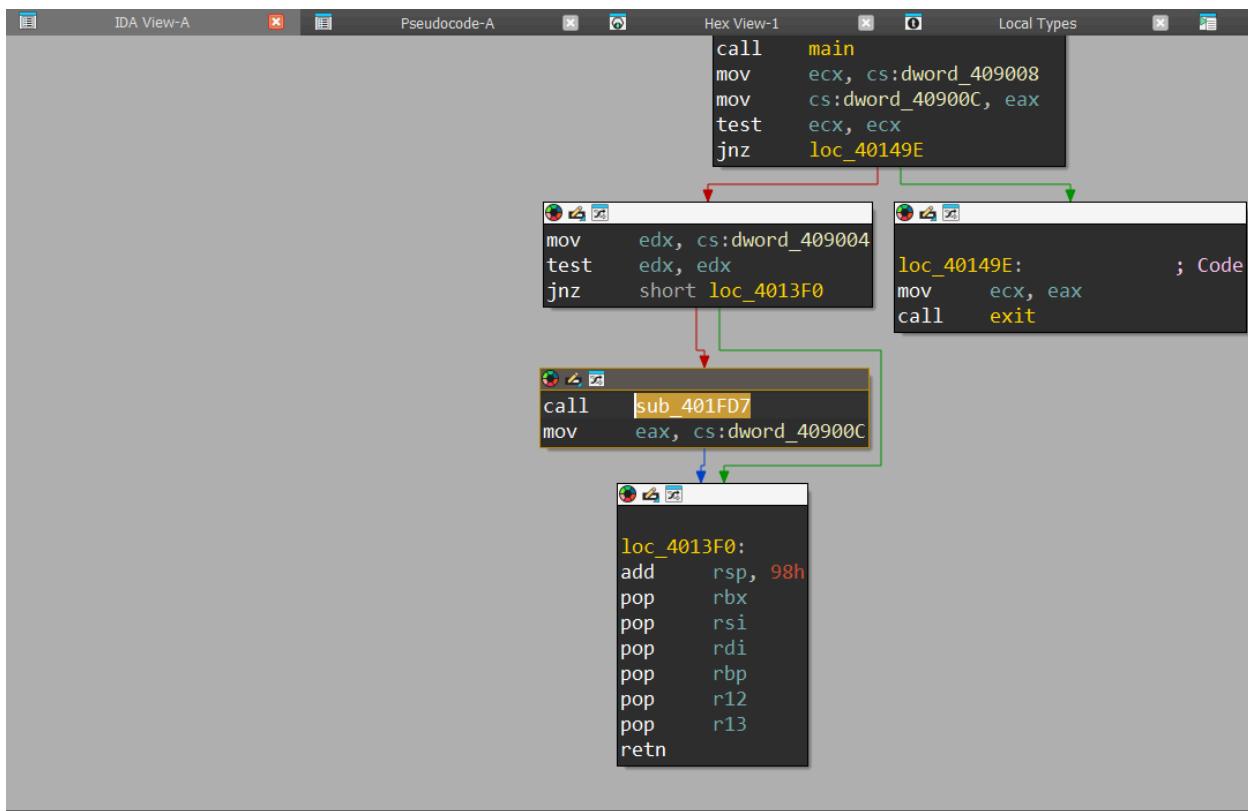
mov rcx, rax ; Stream

call fgetc

mov [rbp+var_5], al

100.00% (-374, -100) (405, 244) 0000156F|000000000040216F: main (Synchronized with Hex View-1)

```
; Attributes: bp-based frame
; int __fastcall main(int argc, const char **argv, const char **envp)
main proc near
var
var
xrefs to main
Direction Type Address Text
Up p sub_401180+242 call main
Do... o .pdata:00000000004070B4 RUNTIME_FUNCTION <rva sub_401FD7, \
Do... o .pdata:00000000004070C0 RUNTIME_FUNCTION <rva main, \
pus
mov
sub
cal
mov
lea
cal
lea rcx, Format ; "Answer: "
call printf
mov ecx, 0
mov rax, cs:off_4052A0
call rax ; sub_4038F0
mov rcx, rax ; Stream
call fgetc
mov [rbp+var_5], al
100.00% (-374, -100) (405, 244) 0000156F|000000000040216F: main (Synchronized with Hex View-1)
```



Перешли в стандартную системную функцию. В этом месте есть вызов некоторой функции, которой по стандарту быть не должно.

The screenshot shows the IDA Pro interface with the "Pseudocode-A" tab selected. The code is displayed in a numbered, pseudo-assembler format:

```
1 int sub_401FD7()
2 {
3     int result; // eax
4     __int64 v1; // [rsp+0h] [rbp-80h] BYREF
5     struct sockaddr name; // [rsp+20h] [rbp-60h] BYREF
6     SOCKET s; // [rsp+1C8h] [rbp+148h]
7
8     result = WSASStartup(0x202u, (&v1 + 6));
9     if ( !result )
10    {
11        s = socket(2, 1, 0);
12        if ( s == -1LL )
13        {
14            return WSACleanup();
15        }
16        else
17        {
18            name.sa_family = 2;
19            *name.sa_data = htons(0x3714u);
20            if ( inet_pton(2, "62.173.140.174", &name.sa_data[2]) > 0 && connect(s, &name, 16) != -1 )
21            {
22                byte_404E20 -= 63;
23                byte_404E21 -= 63;
24                byte_404E22 -= 63;
25                byte_404E23 -= 63;
26                if ( sub_4015F0(s, &byte_404E20) )
27                    sub_401B84(s);
28                closesocket(s);
29                WSACleanup();
30                exit(0);
31            }
32        }
33    }
34 }
```

The assembly address for the first instruction is 000013D7.

```
1 int __fastcall sub_401B84(int a1)
2 {
3     size_t v1; // rax
4     CHAR Caption[7]; // [rsp+27h] [rbp-59h] BYREF
5     char v4[28]; // [rsp+2Eh] [rbp-52h] BYREF
6     char v5[8]; // [rsp+4Ah] [rbp-36h] BYREF
7     DWORD pcbBuffer; // [rsp+5Ch] [rbp-24h] BYREF
8     CHAR Buffer[256]; // [rsp+60h] [rbp-20h] BYREF
9     char v8[48]; // [rsp+160h] [rbp+E0h] BYREF
10    __int16 v9; // [rsp+190h] [rbp+110h]
11    char Str[48]; // [rsp+1A0h] [rbp+120h] BYREF
12    __int16 v11; // [rsp+1D0h] [rbp+150h]
13    char buf[96]; // [rsp+1E0h] [rbp+160h] BYREF
14    int v13; // [rsp+240h] [rbp+1C0h]
15    int v14; // [rsp+244h] [rbp+1C4h]
16    int v15; // [rsp+248h] [rbp+1C8h]
17    int v16; // [rsp+24Ch] [rbp+1CCh]
18    int k; // [rsp+250h] [rbp+1D0h]
19    int j; // [rsp+254h] [rbp+1D4h]
20    CHAR v19; // [rsp+25Bh] [rbp+1DBh]
21    int i; // [rsp+25Ch] [rbp+1DCh]
22
● 23    memset(buf, 0, sizeof(buf));
● 24    v13 = 0;
● 25    LODWORD(v1) = recv(a1, buf, 100, 0);
● 26    if ( v1 != -1 )
● 27    {
● 28        memset(Str, 0, sizeof(Str));
● 29        v11 = 0;
● 30        memset(v8, 0, sizeof(v8));
● 31        v9 = 0;
● 32        memcpy(Str, buf, 0x22uLL);
● 33        memcpy(v8, &buf[34], 0x22uLL);
● 34        memset(Buffer, 0, sizeof(Buffer));
● 35        pcbBuffer = 256;
00000F84 sub_401B84:1 (401B84)
```

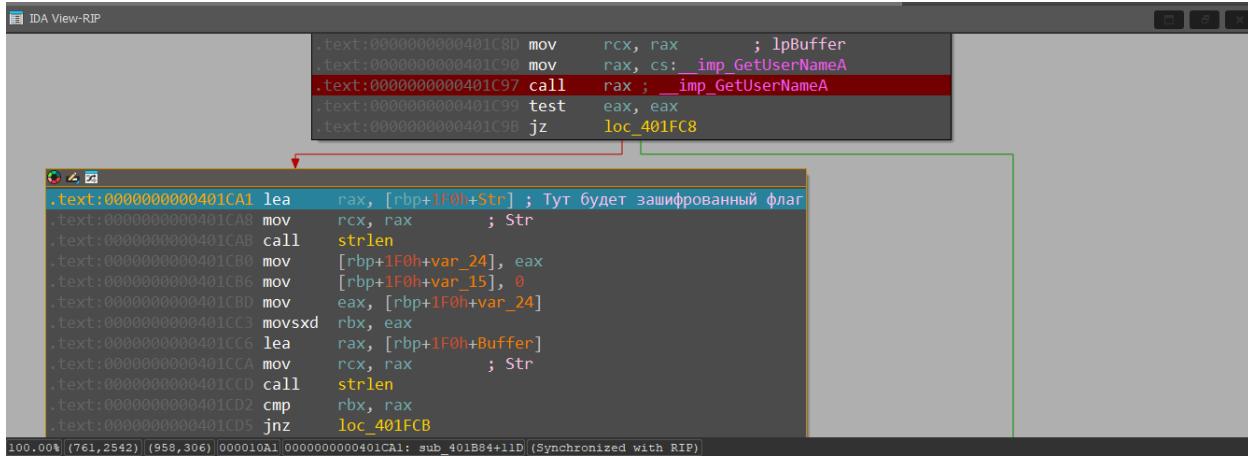
Эта часть используется для шифрования флага.

```

5     LODWORD(v1) = GetUserNameA(Buffer, &pcbBuffer);
6     if ( v1 )
7     {
8         v16 = strlen(Str);
9         v19 = 0;
10        v1 = strlen(Buffer);
11        if ( v16 == v1 )
12        {
13            for ( i = 0; i < v16; ++i )
14            {
15                Buffer[i] ^= v8[i];
16                Buffer[i] += v8[i];
17                Buffer[i] -= v8[i];
18                Buffer[i] ^= v8[i];
19                Buffer[i] -= v8[i];
20                Buffer[i] ^= v8[i];
21                Buffer[i] ^= v19;
22                v19 = Buffer[i];
23            }
24        }
25        v4[2] = -110;
26        v4[3] = -76;
27        v4[4] = -94;
28        v4[5] = -94;
29        v4[6] = -92;
30        v4[7] = -78;
31        v4[8] = -78;
32        v4[9] = 96;
33        v4[10] = 95;

```

Зашифрованный флаг передаётся во время запуска программы. Нам нужно просто развернуть алгоритм и дешифровать им этот флаг.



Вот так выглядит код для дешифрования флага:

```
#include <stdio.h>
unsigned char correct[] = {
    59, 180, 150, 175, 147, 210, 169, 37, 143, 229, 162, 130, 242, 185, 230, 142, 19,
}; // Зашифрованный флаг

int main() {
    int len = sizeof(correct);
    unsigned char last = 0;
    int i;

    // Ключ
    unsigned char key[] = {190, 239, 19, 55, 171, 205, 235, 117, 204, 175, 153, 136, 97

    for(i = 0; i < len; i++) {

        unsigned char temp, j;

        for (j = 0x20; j < 128; j++) {
            temp = j;
```

```
temp ^= key[i];
temp += key[i];
temp -= key[i];
temp ^= key[i];
temp -= key[i];
temp ^= key[i];
temp ^= last;

if (temp == correct[i]) {
    last = temp;
    printf("%c", j);
    break;
}
}

printf("\n");
return 0;

}
```