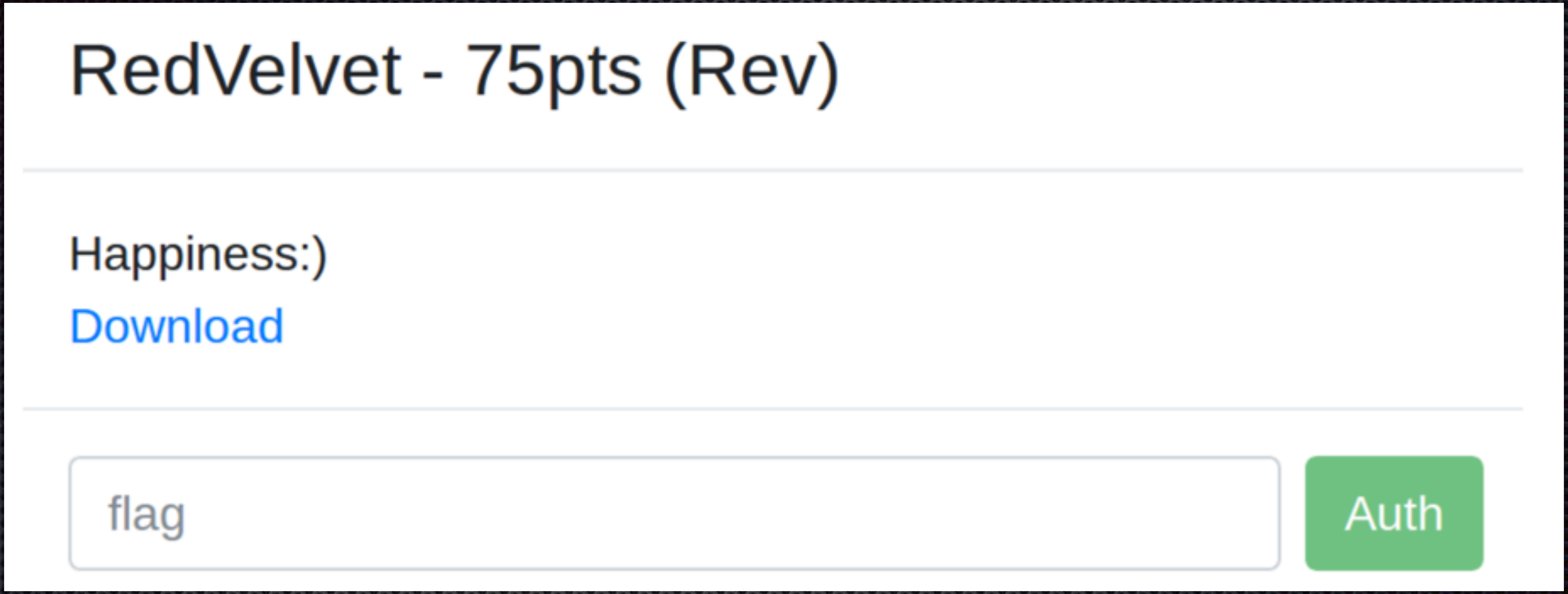
CODEGATE - RedVelvet



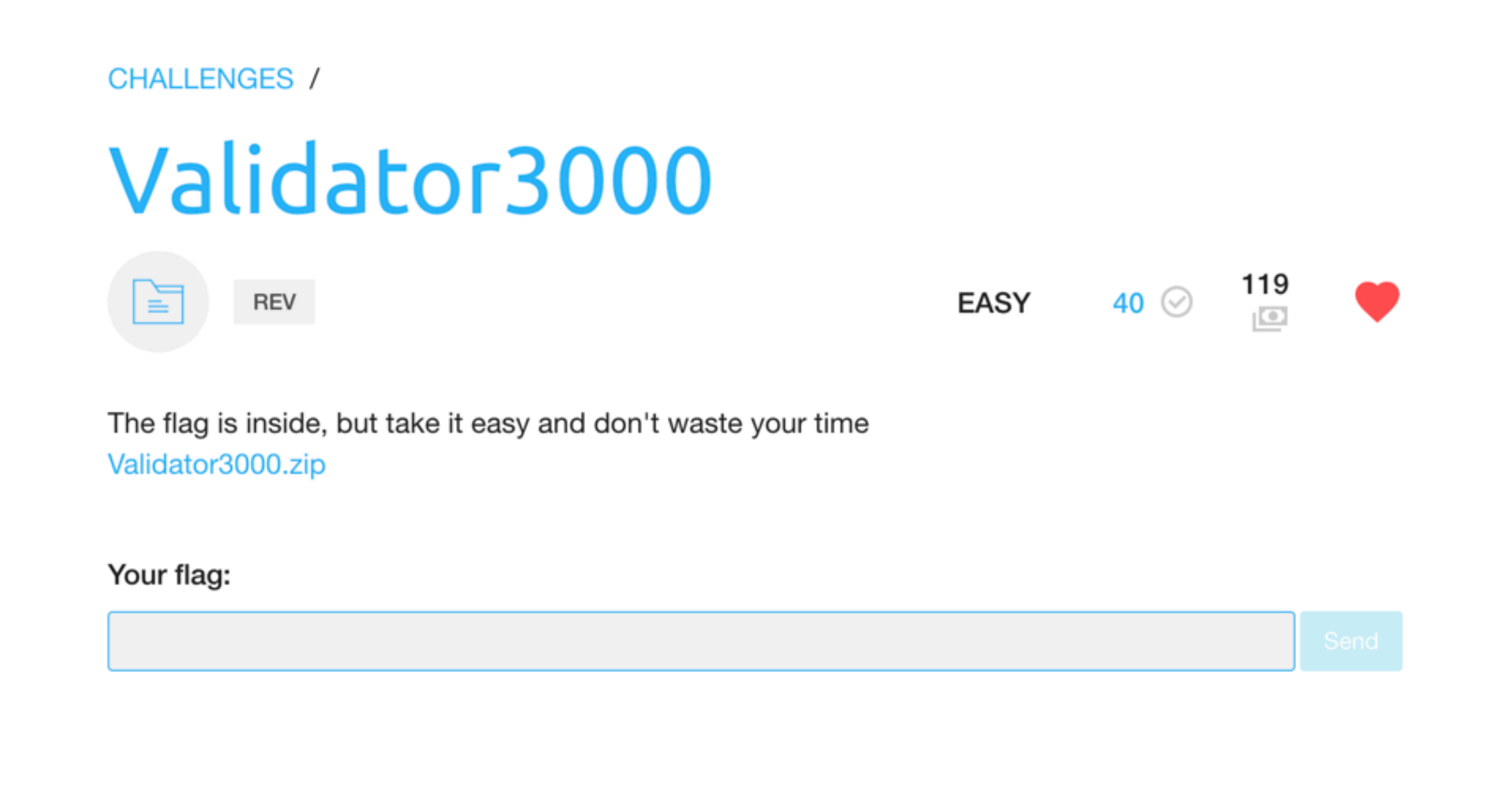
RedVelvet - 75pts (Rev)

Happiness:)

Download

SCP 김성준
Pvgodd

CTF ZONE Vlidator30000



CHALLENGES /

Validator3000

REV

EASY

40 ✓

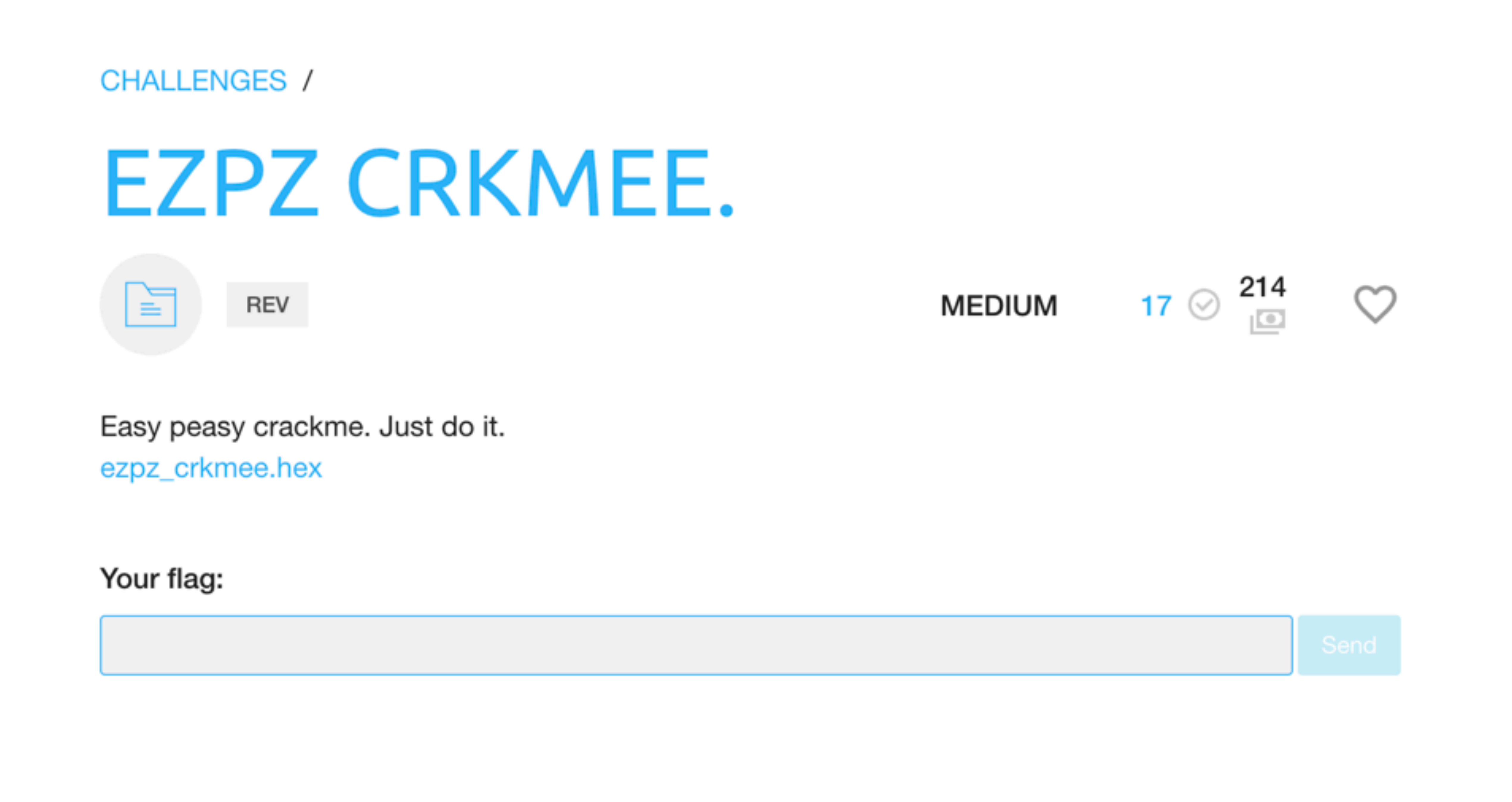
119

The flag is inside, but take it easy and don't waste your time

Validator3000.zip

Your flag:

CTF ZONE ezpz_crmee



CHALLENGES /

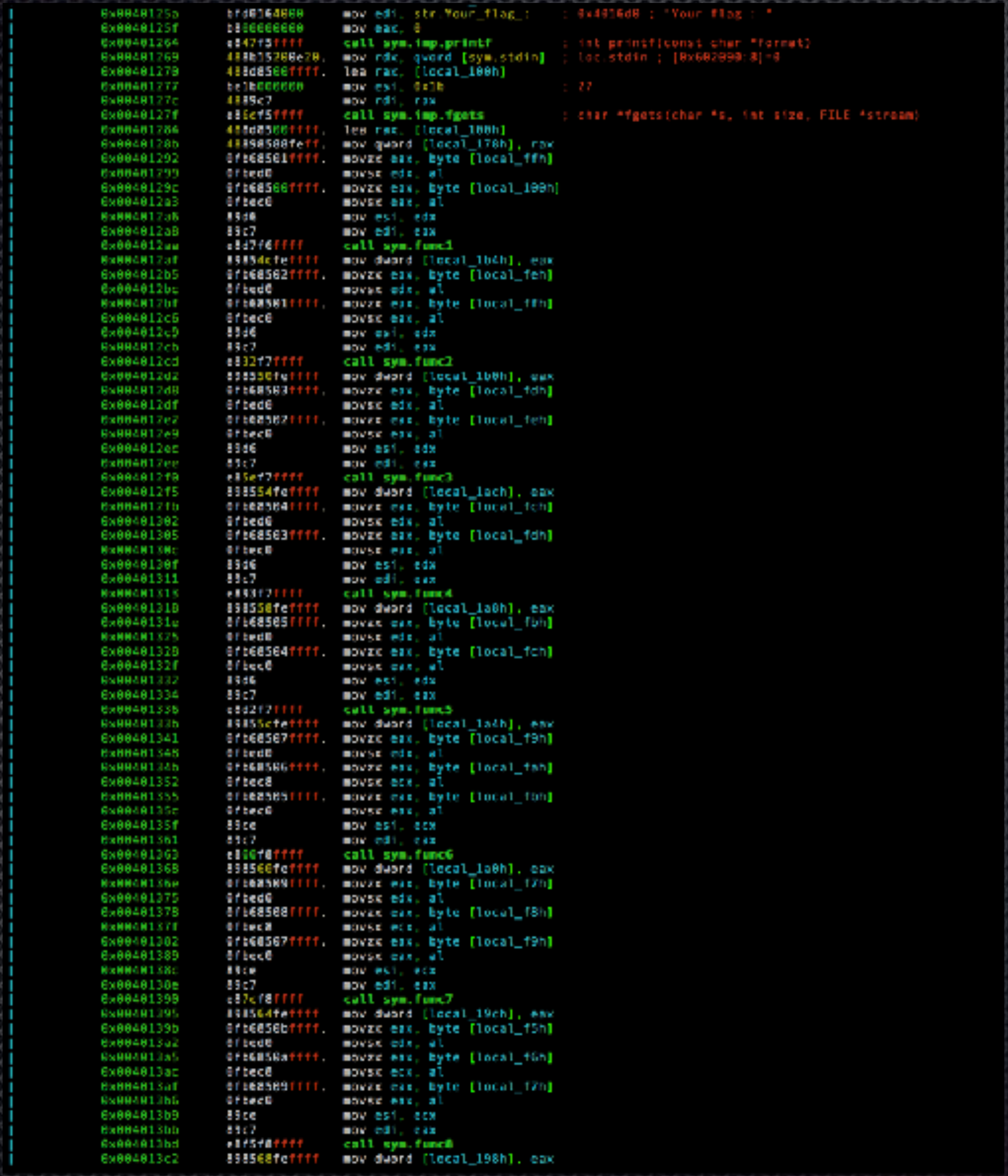
EZPZ CRKMEE.

Easy peasy crackme. Just do it.

ezpz_crkmee.hex

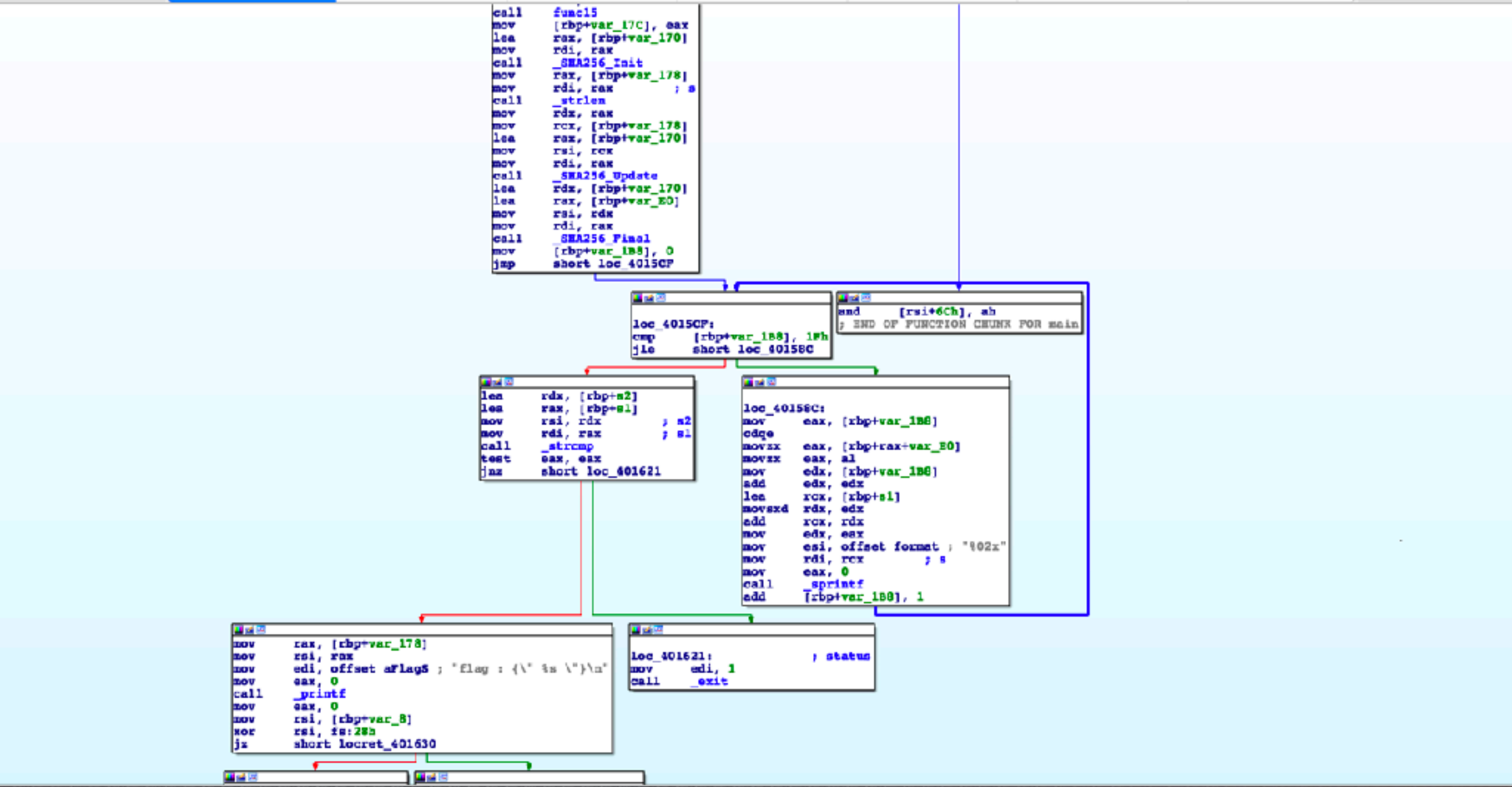
Your flag:

CODE GATE - Redvelvet



```
|           0x0040125a      bfd8164000     mov edi, str.Your_flag_:    ; 0x4016d8 ; "Your flag : "
|           0x0040125f      b800000000     mov eax, 0
|           0x00401264      e847f5ffff     call sym.imp.printf         ; int printf(const char *format)
|           0x00401269      488b15200e20.  mov rdx, qword [sym.stdin]  ; loc.stdin ; [0x602090:8]=0
|           0x00401270      488d8500ffff.  lea rax, [local_100h]
|           0x00401277      be1b000000     mov esi, 0x1b               ; 27
|           0x0040127c      4889c7         mov rdi, rax
|           0x0040127f      e85cf5ffff     call sym.imp.fgets          ; char *fgets(char *s, int size, FILE *stream)
|           0x00401284      488d8500ffff.  lea rax, [local_100h]
|           0x0040128b      48898588feff.  mov qword [local_178h], rax
|           0x00401292      0fb68501ffff.  movzx eax, byte [local_ffh]
|           0x00401299      0fbed0         movsx edx, al
|           0x0040129c      0fb68500ffff.  movzx eax, byte [local_100h]
|           0x004012a3      0fbec0         movsx eax, al
|           0x004012a6      89d6           mov esi, edx
|           0x004012a8      89c7           mov edi, eax
|           0x004012aa      e8d7f6ffff     call sym.func1
|           0x004012af      89854cfeffff   mov dword [local_1b4h], eax
|           0x004012b5      0fb68502ffff.  movzx eax, byte [local_feh]
|           0x004012bc      0fbed0         movsx edx, al
|           0x004012bf      0fb68501ffff.  movzx eax, byte [local_ffh]
|           0x004012c6      0fbec0         movsx eax, al
|           0x004012c9      89d6           mov esi, edx
|           0x004012cb      89c7           mov edi, eax
|           0x004012cd      e832f7ffff     call sym.func2
|           0x004012d2      898550feffff   mov dword [local_1b0h], eax
|           0x004012d8      0fb68503ffff.  movzx eax, byte [local_fdh]
|           0x004012df      0fbed0         movsx edx, al
|           0x004012e2      0fb68502ffff.  movzx eax, byte [local_feh]
|           0x004012e9      0fbec0         movsx eax, al
|           0x004012ec      89d6           mov esi, edx
|           0x004012ee      89c7           mov edi, eax
|           0x004012f0      e85ef7ffff     call sym.func3
|           0x004012f5      898554feffff   mov dword [local_1ach], eax
|           0x004012fb      0fb68504ffff.  movzx eax, byte [local_fch]
|           0x00401302      0fbed0         movsx edx, al
|           0x00401305      0fb68503ffff.  movzx eax, byte [local_fdh]
|           0x0040130c      0fbec0         movsx eax, al
|           0x0040130f      89d6           mov esi, edx
|           0x00401311      89c7           mov edi, eax
|           0x00401313      e893f7ffff     call sym.func4
|           0x00401318      898558feffff   mov dword [local_1a8h], eax
|           0x0040131e      0fb68505ffff.  movzx eax, byte [local_fbh]
|           0x00401325      0fbed0         movsx edx, al
|           0x00401328      0fb68504ffff.  movzx eax, byte [local_fch]
|           0x0040132f      0fbec0         movsx eax, al
|           0x00401332      89d6           mov esi, edx
|           0x00401334      89c7           mov edi, eax
|           0x00401336      e8d2f7ffff     call sym.func5
|           0x0040133b      89855cfeffff   mov dword [local_1a4h], eax
|           0x00401341      0fb68507ffff.  movzx eax, byte [local_f9h]
|           0x00401348      0fbed0         movsx edx, al
|           0x0040134b      0fb68506ffff.  movzx eax, byte [local_fah]
|           0x00401352      0fbec8         movsx ecx, al
|           0x00401355      0fb68505ffff.  movzx eax, byte [local_fbh]
|           0x0040135c      0fbec0         movsx eax, al
|           0x0040135f      89ce           mov esi, ecx
|           0x00401361      89c7           mov edi, eax
|           0x00401363      e860f8ffff     call sym.func6
|           0x00401368      898560feffff   mov dword [local_1a0h], eax
|           0x0040136e      0fb68509ffff.  movzx eax, byte [local_f7h]
|           0x00401375      0fbed0         movsx edx, al
|           0x00401378      0fb68508ffff.  movzx eax, byte [local_f8h]
|           0x0040137f      0fbec8         movsx ecx, al
|           0x00401382      0fb68507ffff.  movzx eax, byte [local_f9h]
|           0x00401389      0fbec0         movsx eax, al
|           0x0040138c      89ce           mov esi, ecx
|           0x0040138e      89c7           mov edi, eax
|           0x00401390      e87cf8ffff     call sym.func7
|           0x00401395      898564feffff   mov dword [local_19ch], eax
|           0x0040139b      0fb6850bffff.  movzx eax, byte [local_f5h]
|           0x004013a2      0fbed0         movsx edx, al
|           0x004013a5      0fb6850affff.  movzx eax, byte [local_f6h]
|           0x004013ac      0fbec8         movsx ecx, al
|           0x004013af      0fb68509ffff.  movzx eax, byte [local_f7h]
|           0x004013b6      0fbec0         movsx eax, al
|           0x004013b9      89ce           mov esi, ecx
|           0x004013bb      89c7           mov edi, eax
|           0x004013bd      e8f5f8ffff     call sym.func8
|           0x004013c2      898568feffff   mov dword [local_198h], eax
```

CODE GATE - Redvelvet



```
call    func15
mov     [rbp+var_17C], eax
lea     rax, [rbp+var_170]
mov     rdi, rax
call    _SHA256_Init
mov     rax, [rbp+var_178]
mov     rdi, rax        ; s
call    _strlen
mov     rdx, rax
mov     rcx, [rbp+var_178]
lea     rax, [rbp+var_170]
mov     rsi, rcx
mov     rdi, rax
call    _SHA256_Update
lea     rdx, [rbp+var_170]
lea     rax, [rbp+var_E0]
mov     rsi, rdx
mov     rdi, rax
call    _SHA256_Final
mov     [rbp+var_1B8], 0
jmp     short loc_4015CF
```

```
loc_4015CF:
cmp     [rbp+var_1B8], 1Fh
jle     short loc_40158C
```

```
and     [rsi+6Ch], ah
; END OF FUNCTION CHUNK FOR main
```

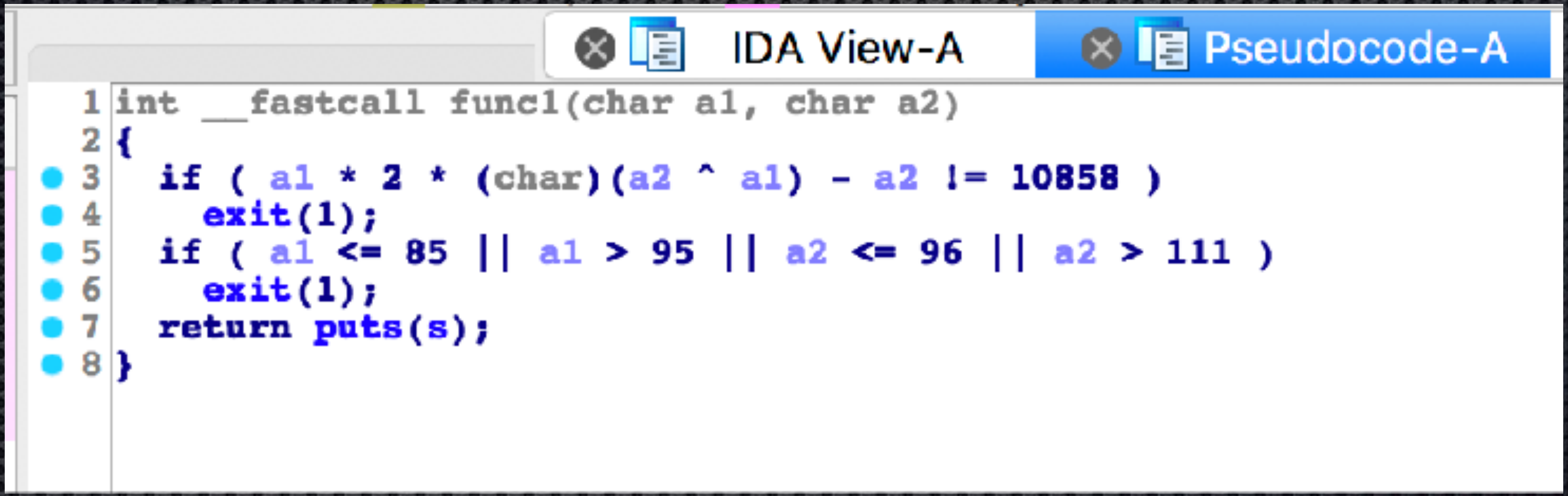
```
lea     rdx, [rbp+s2]
lea     rax, [rbp+s1]
mov     rsi, rdx        ; s2
mov     rdi, rax        ; s1
call    _strcmp
test    eax, eax
jnz     short loc_401621
```

```
loc_40158C:
mov     eax, [rbp+var_1B8]
cdqe
movzx   eax, [rbp+rax+var_E0]
movzx   eax, al
mov     edx, [rbp+var_1B8]
add     edx, edx
lea     rcx, [rbp+s1]
movsxd  rdx, edx
add     rcx, rdx
mov     edx, eax
mov     esi, offset format ; "%02x"
mov     rdi, rcx        ; s
mov     eax, 0
call    _sprintf
add     [rbp+var_1B8], 1
```

```
mov     rax, [rbp+var_178]
mov     rsi, rax
mov     edi, offset aFlagS ; "flag : {\" %s \"}\n"
mov     eax, 0
call    _printf
mov     eax, 0
mov     rsi, [rbp+var_8]
xor     rsi, fs:28h
jz      short locret_401630
```

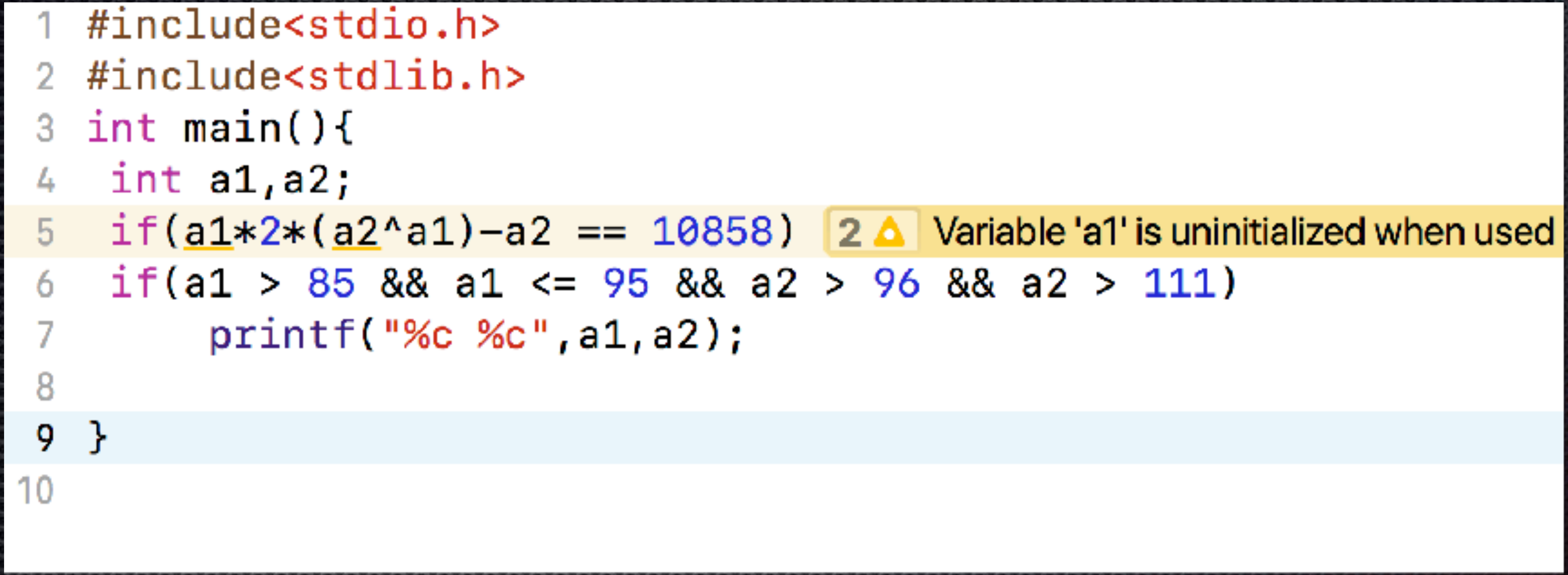
```
loc_401621:             ; status
mov     edi, 1
call    _exit
```

CODE GATE - Redvelvet



```
int __fastcall func1(char a1, char a2)
{
  if ( a1 * 2 * (char)(a2 ^ a1) - a2 != 10858 )
    exit(1);
  if ( a1 <= 85 || a1 > 95 || a2 <= 96 || a2 > 111 )
    exit(1);
  return puts(s);
}
```

CODE GATE - Redvelvet



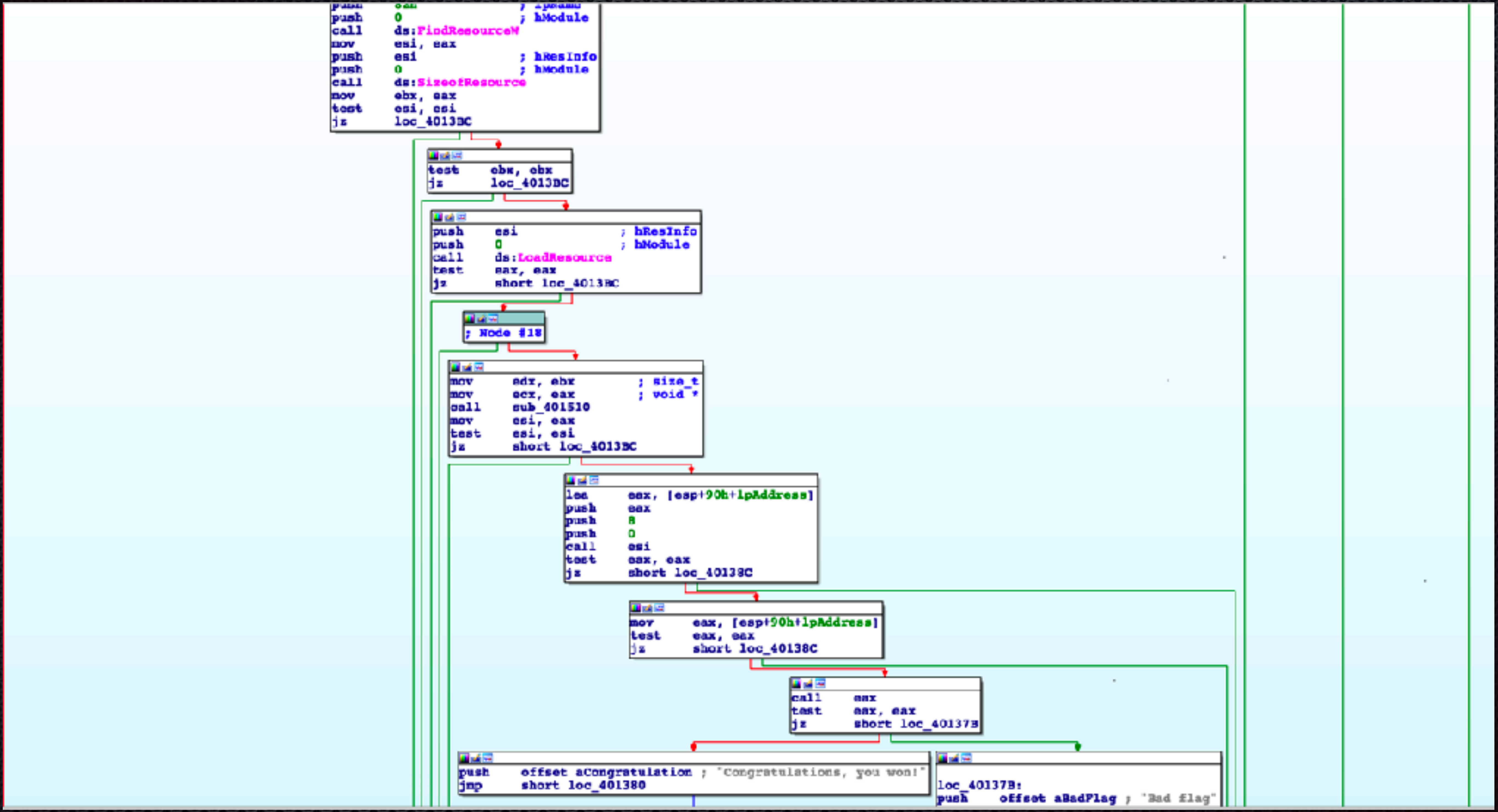
```c
#include<stdio.h>
#include<stdlib.h>
int main(){
 int a1,a2;
 if(a1*2*(a2^a1)-a2 == 10858)
 if(a1 > 85 && a1 <= 95 && a2 > 96 && a2 > 111)
     printf("%c %c",a1,a2);

}

```

2 ⚠ Variable 'a1' is uninitialized when used

CTF ZONE - Validator 3000



```
push    eax                 ; lpName
push    0                   ; hModule
call    ds:FindResourceW
mov     esi, eax
push    esi                 ; hResInfo
push    0                   ; hModule
call    ds:SizeofResource
mov     ebx, eax
test    esi, esi
jz      loc_4013BC
```

```
test    ebx, ebx
jz      loc_4013BC
```

```
push    esi                 ; hResInfo
push    0                   ; hModule
call    ds:LoadResource
test    eax, eax
jz      short loc_4013BC
```

```
; Node #18
```

```
mov     edx, ebx            ; size_t
mov     ecx, eax            ; void *
call    sub_401510
mov     esi, eax
test    esi, esi
jz      short loc_4013BC
```

```
lea     eax, [esp+90h+lpAddress]
push    eax
push    8
push    0
call    esi
test    eax, eax
jz      short loc_40138C
```

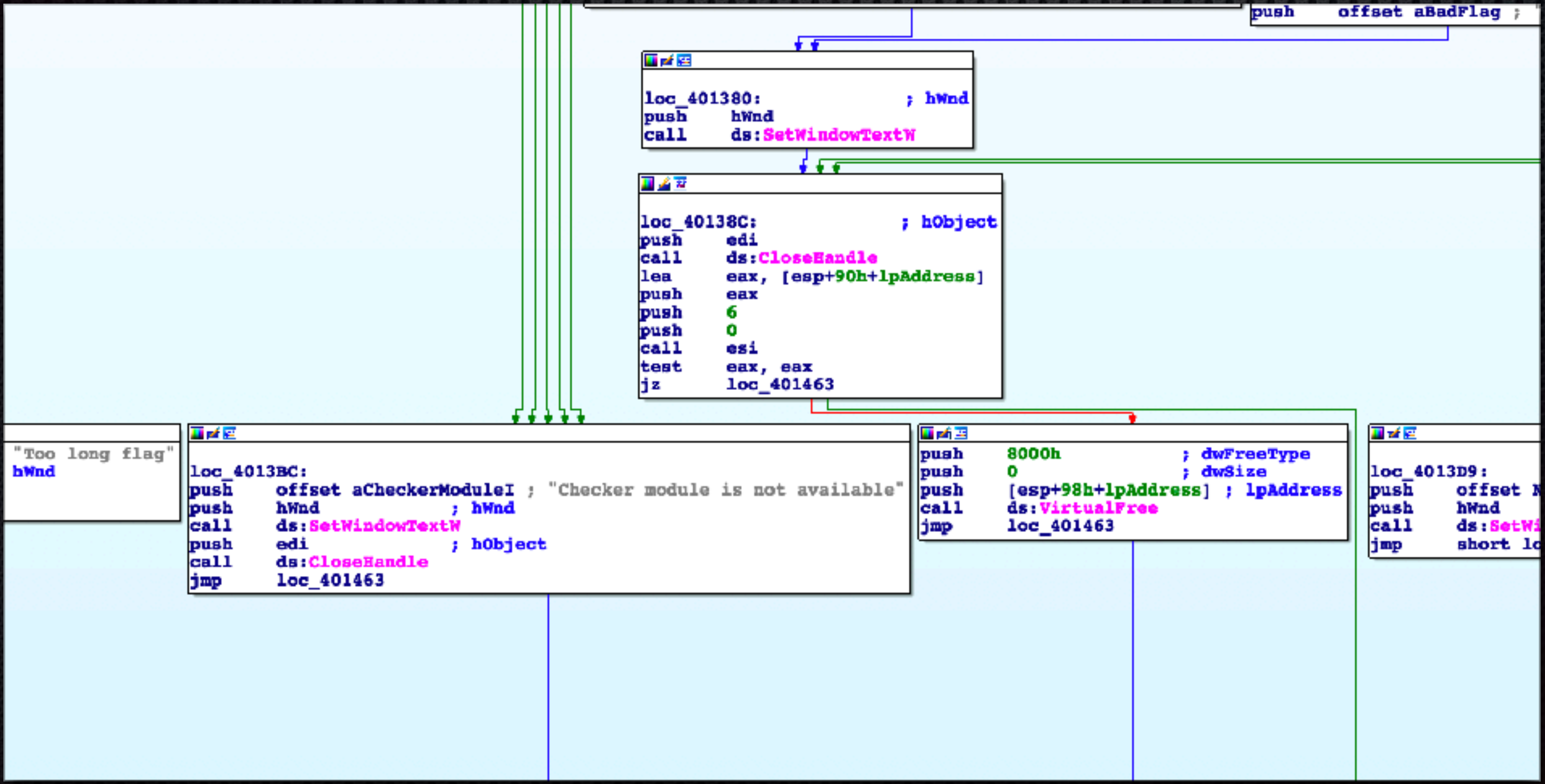
```
mov     eax, [esp+90h+lpAddress]
test    eax, eax
jz      short loc_40138C
```

```
call    eax
test    eax, eax
jz      short loc_40137B
```

```
push    offset aCongratulation ; "Congratulations, you won!"
jmp     short loc_401380
```

```
loc_40137B:
push    offset aBadFlag ; "Bad flag"
```

CTF ZONE - Validator 3000



```
push    offset aBadFlag ;
```

```
loc_401380:                ; hWnd
push    hWnd
call    ds:SetWindowTextW
```

```
loc_40138C:               ; hObject
push    edi
call    ds:CloseHandle
lea     eax, [esp+90h+lpAddress]
push    eax
push    6
push    0
call    esi
test    eax, eax
jz      loc_401463
```

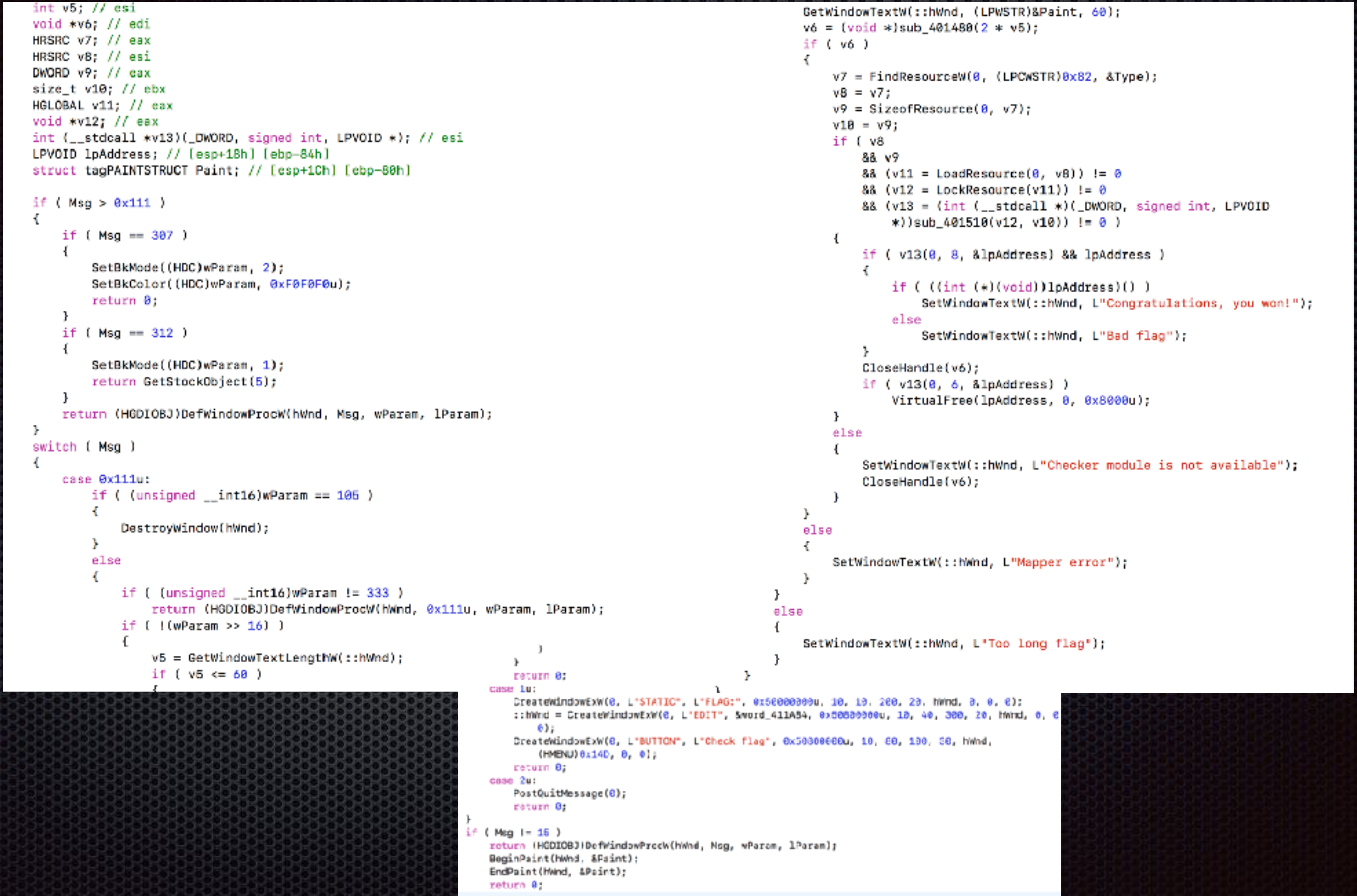
```
"Too long flag"
hWnd
```

```
loc_4013BC:
push    offset aCheckerModuleI ; "Checker module is not available"
push    hWnd            ; hWnd
call    ds:SetWindowTextW
push    edi             ; hObject
call    ds:CloseHandle
jmp     loc_401463
```

```
push    8000h              ; dwFreeType
push    0                  ; dwSize
push    [esp+98h+lpAddress] ; lpAddress
call    ds:VirtualFree
jmp     loc_401463
```

```
loc_4013D9:
push    offset N
push    hWnd
call    ds:SetWi
jmp     short lo
```

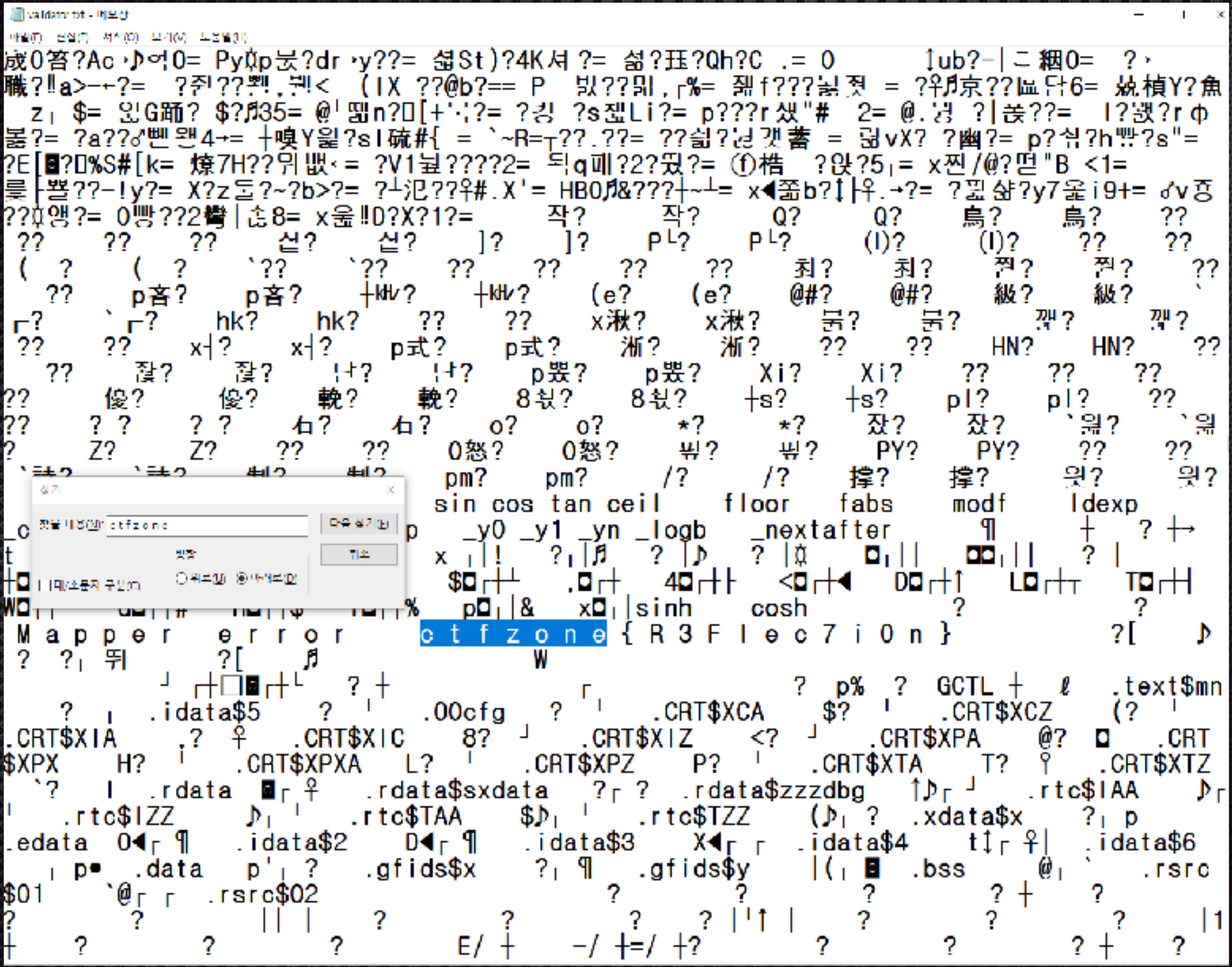
CTF ZONE - Validator 3000



```
int v5; // esi
void *v6; // edi
HRSRC v7; // eax
HRSRC v8; // esi
DWORD v9; // eax
size_t v10; // ebx
HGLOBAL v11; // eax
void *v12; // eax
int (__stdcall *v13)(_DWORD, signed int, LPVOID *); // esi
LPVOID lpAddress; // [esp+18h] [ebp-84h]
struct tagPAINTSTRUCT Paint; // [esp+1Ch] [ebp-80h]

if ( Msg > 0x111 )
{
    if ( Msg == 307 )
    {
        SetBkMode((HDC)wParam, 2);
        SetBkColor((HDC)wParam, 0xF0F0F0u);
        return 0;
    }
    if ( Msg == 312 )
    {
        SetBkMode((HDC)wParam, 1);
        return GetStockObject(5);
    }
    return (HGDIOBJ)DefWindowProcW(hWnd, Msg, wParam, lParam);
}
switch ( Msg )
{
    case 0x111u:
        if ( (unsigned __int16)wParam == 105 )
        {
            DestroyWindow(hWnd);
        }
        else
        {
            if ( (unsigned __int16)wParam != 333 )
                return (HGDIOBJ)DefWindowProcW(hWnd, 0x111u, wParam, lParam);
            if ( !(wParam >> 16) )
            {
                v5 = GetWindowTextLengthW(::hWnd);
                if ( v5 <= 60 )
                {
                    GetWindowTextW(::hWnd, (LPWSTR)&Paint, 60);
                    v6 = (void *)sub_401480(2 * v5);
                    if ( v6 )
                    {
                        v7 = FindResourceW(0, (LPCWSTR)0x82, &Type);
                        v8 = v7;
                        v9 = SizeofResource(0, v7);
                        v10 = v9;
                        if ( v8
                          && v9
                          && (v11 = LoadResource(0, v8)) != 0
                          && (v12 = LockResource(v11)) != 0
                          && (v13 = (int (__stdcall *)(_DWORD, signed int, LPVOID
                                *))sub_401510(v12, v10)) != 0 )
                        {
                            if ( v13(0, 8, &lpAddress) && lpAddress )
                            {
                                if ( ((int (*)(void))lpAddress)() )
                                    SetWindowTextW(::hWnd, L"Congratulations, you won!");
                                else
                                    SetWindowTextW(::hWnd, L"Bad flag");
                            }
                            CloseHandle(v6);
                            if ( v13(0, 6, &lpAddress) )
                                VirtualFree(lpAddress, 0, 0x8000u);
                        }
                        else
                        {
                            SetWindowTextW(::hWnd, L"Checker module is not available");
                            CloseHandle(v6);
                        }
                    }
                    else
                    {
                        SetWindowTextW(::hWnd, L"Mapper error");
                    }
                }
                else
                {
                    SetWindowTextW(::hWnd, L"Too long flag");
                }
            }
            }
        }
        return 0;
    case 1u:
        CreateWindowExW(0, L"STATIC", L"FLAG:", 0x50000000u, 10, 10, 200, 20, hWnd, 0, 0, 0);
        ::hWnd = CreateWindowExW(0, L"EDIT", &word_411A54, 0x50800000u, 10, 40, 300, 20, hWnd, 0, 0
            0);
        CreateWindowExW(0, L"BUTTON", L"Check flag", 0x50000000u, 10, 60, 100, 30, hWnd,
            (HMENU)0x14D, 0, 0);
        return 0;
    case 2u:
        PostQuitMessage(0);
        return 0;
}
if ( Msg != 15 )
    return (HGDIOBJ)DefWindowProcW(hWnd, Msg, wParam, lParam);
    BeginPaint(hWnd, &Paint);
    EndPaint(hWnd, &Paint);
    return 0;
```

CTF ZONE - Validator 3000



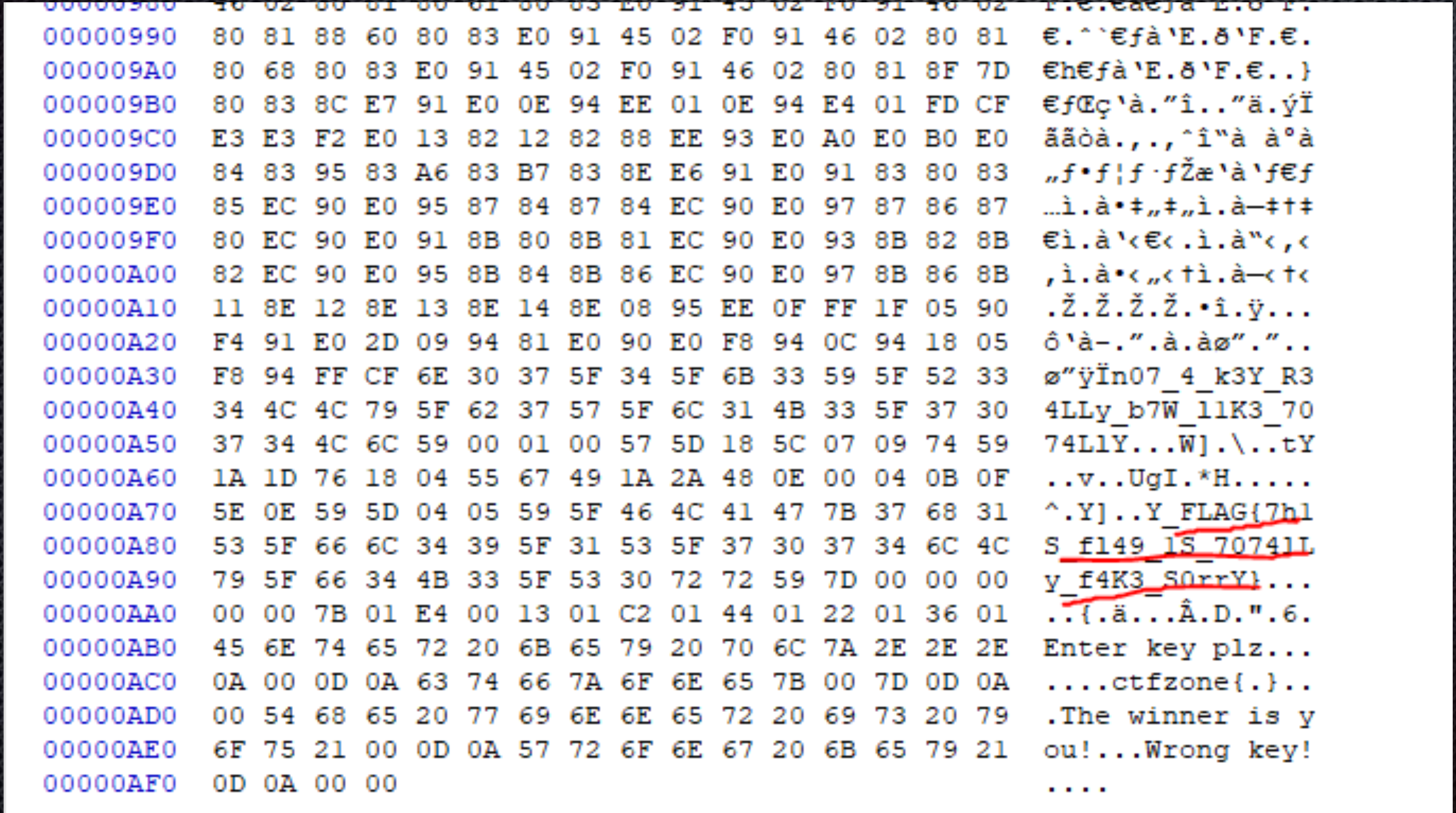
찾을 내용(N): ctfzone

방향: ○위로(U) ◉아래로(D)

□ 대/소문자 구분(C)

```
sin cos tan ceil   floor   fabs   modf   ldexp
_y0 _y1 _yn _logb   _nextafter
sinh   cosh
Mapper error    ctfzone{R3Flec7i0n}
GCTL   .text$mn
.idata$5   .00cfg   .CRT$XCA   .CRT$XCZ
.CRT$XIA   .CRT$XIC   .CRT$XIZ   .CRT$XPA   .CRT
$XPX   .CRT$XPXA   .CRT$XPZ   .CRT$XTA   .CRT$XTZ
.rdata   .rdata$sxdata   .rdata$zzzdbg   .rtc$IAA
.rtc$IZZ   .rtc$TAA   .rtc$TZZ   .xdata$x
.edata   .idata$2   .idata$3   .idata$4   .idata$6
.data   .gfids$x   .gfids$y   .bss   .rsrc
$01   .rsrc$02
```

CTF ZONE - ezpz_crmee



```
00000990  80 81 88 60 80 83 E0 91 45 02 F0 91 46 02 80 81  €.ˆ`€ƒà'E.ð'F.€.
000009A0  80 68 80 83 E0 91 45 02 F0 91 46 02 80 81 8F 7D  €h€ƒà'E.ð'F.€..}
000009B0  80 83 8C E7 91 E0 0E 94 EE 01 0E 94 E4 01 FD CF  €ƒŒç'à.”î..”ä.ýÏ
000009C0  E3 E3 F2 E0 13 82 12 82 88 EE 93 E0 A0 E0 B0 E0  ããòà.,.,ˆî“à à°à
000009D0  84 83 95 83 A6 83 B7 83 8E E6 91 E0 91 83 80 83  „ƒ•ƒ¦ƒ·ƒŽæ'à'ƒ€ƒ
000009E0  85 EC 90 E0 95 87 84 87 84 EC 90 E0 97 87 86 87  …ì.à•‡„‡„ì.à—‡†‡
000009F0  80 EC 90 E0 91 8B 80 8B 81 EC 90 E0 93 8B 82 8B  €ì.à'‹€‹.ì.à“‹,‹
00000A00  82 EC 90 E0 95 8B 84 8B 86 EC 90 E0 97 8B 86 8B  ,ì.à•‹„‹†ì.à—‹†‹
00000A10  11 8E 12 8E 13 8E 14 8E 08 95 EE 0F FF 1F 05 90  .Ž.Ž.Ž.Ž.•î.ÿ...
00000A20  F4 91 E0 2D 09 94 81 E0 90 E0 F8 94 0C 94 18 05  ô'à-.”.à.àø”.”..
00000A30  F8 94 FF CF 6E 30 37 5F 34 5F 6B 33 59 5F 52 33  ø”ÿÏn07_4_k3Y_R3
00000A40  34 4C 4C 79 5F 62 37 57 5F 6C 31 4B 33 5F 37 30  4LLy_b7W_l1K3_70
00000A50  37 34 4C 6C 59 00 01 00 57 5D 18 5C 07 09 74 59  74LlY...W].\..tY
00000A60  1A 1D 76 18 04 55 67 49 1A 2A 48 0E 00 04 0B 0F  ..v..UgI.*H.....
00000A70  5E 0E 59 5D 04 05 59 5F 46 4C 41 47 7B 37 68 31  ^.Y]..Y_FLAG{7h1
00000A80  53 5F 66 6C 34 39 5F 31 53 5F 37 30 37 34 6C 4C  S_fl49_1S_7074lL
00000A90  79 5F 66 34 4B 33 5F 53 30 72 72 59 7D 00 00 00  y_f4K3_S0rrY}...
00000AA0  00 00 7B 01 E4 00 13 01 C2 01 44 01 22 01 36 01  ..{.ä...Â.D.".6.
00000AB0  45 6E 74 65 72 20 6B 65 79 20 70 6C 7A 2E 2E 2E  Enter key plz...
00000AC0  0A 00 0D 0A 63 74 66 7A 6F 6E 65 7B 00 7D 0D 0A  ....ctfzone{.}..
00000AD0  00 54 68 65 20 77 69 6E 6E 65 72 20 69 73 20 79  .The winner is y
00000AE0  6F 75 21 00 0D 0A 57 72 6F 6E 67 20 6B 65 79 21  ou!...Wrong key!
00000AF0  0D 0A 00 00                                      ....
```