고등학생부 2위 차현수(cha5126568) Write up

Too Easy(Rev,50) : IDA 로 열면 바로 플래그 보입니다.

```
.text:00401080
.text:00401080                push    ebp
.text:00401081                mov     ebp, esp
.text:00401083                sub     esp, 104h
.text:00401089                mov     eax, ___security_cookie
.text:0040108E                xor     eax, ebp
.text:00401090                mov     [ebp+var_4], eax
.text:00401093                push    0FFh            ; Size
.text:00401098                lea     eax, [ebp+Dst]
.text:0040109E                push    0               ; Val
.text:004010A0                push    eax             ; Dst
.text:004010A1                call    memset
.text:004010A6                push    offset aPassword ; "Password: "
.text:004010AB                call    sub_401020
.text:004010B0                lea     eax, [ebp+Dst]
.text:004010B6                push    eax
.text:004010B7                push    offset a36s     ; "%36s"
.text:004010BC                call    sub_401050
.text:004010C1                push    24h             ; MaxCount
.text:004010C3                lea     eax, [ebp+Dst]
.text:004010C9                push    offset Str2     ; "dimigo{do_you_think_rev100_is_hard?}"
.text:004010CE                push    eax             ; Str1
.text:004010CF                call    ds:strncmp
.text:004010D5                add     esp, 24h
.text:004010D8                test    eax, eax
.text:004010DA                jnz     short loc_4010E3
.text:004010DC                push    offset aCorrect ; "\nCorrect\n"
.text:004010E1                jmp     short loc_4010E8
```

Warm REV(Rev,100) : dnSpy 로 열어서 버튼 클릭 쪽에 가면 바로 플래그 보입니다.

```
private void button1_Click(object sender, EventArgs e)
{
    if (this.Input.Text == "dimigo{CS_reversing_is_easy}")
    {
        this.Input.Text = "Correct!";
        return;
    }
    this.Input.Text = "Try Again!";
}
```

What is the End(Rev,200) : 이거 출제 의도는 문자열의 맨 마지막이 널이기 때문에, 뒤에서부터 따라가라는 의미 이지만, 앞에서부터 따라가도 무리가 없는 문제입니다.[1][2]

```
#include<stdio.h>
#include<windows.h>
int main(void)
{
        unsigned int rand_table[] = { 0x6B8B4567, 0x327B23C6, 0x643C9869, 0x66334873,
0x74B0DC51, 0x19495CFF, 0x2AE8944A, 0x625558EC, 0x238E1F29, 0x46E87CCD, 0x3D1B58BA,
0x507ED7AB, 0x2EB141F2, 0x41B71EFB, 0x79E2A9E3, 0x7545E146, 0x515F007C, 0x5BD062C2,
0x12200854, 0x4DB127F8, 0x216231B, 0x1F16E9E8, 0x1190CDE7, 0x66EF438D, 0x140E0F76,
0x3352255A, 0x109CF92E, 0x0DED7263, 0x7FDCC233, 0x1BEFD79F, 0x41A7C4C9, 0x6B68079A,
0x4E6AFB66 };
        unsigned char v41[34];
        unsigned char table[] = {
0xAC,0xAB,0x1e,0x2c,0xa6,0xa1,0x9c,0xe8,0xff,0x61,0x9,0x53,0x25,0x14,0x82,0x3c,0xa5,0x91,
0xa5,0xdb,0xe9,0x4,0x60,0xe0,0x1a,0x6e,0x61,0x41,0xb7,0x4f,0x53,0xcd };
        char now = 'd';
        for (int dst = 0;dst < 33;dst++)
        {
        printf("%c", now);
        for (int i = 32;i < 128;i++)
        {
        v41[dst] = now;
        v41[dst + 1] = (i & 0xff);
        v41[dst] ^= (0xff) ^ (rand_table[dst] & 0xff);
        v41[dst + 1] ^= (0xff) ^ (rand_table[dst + 1] & 0xff);
        if ((v41[dst] ^ v41[dst + 1]) == table[dst])
        {
        now = i & 0xff;
        break;
        }
        }
        }
}
```

1) Reversing.kr 의 MetroApp 도 앞에서부터 따라가도 무리가 없는 것처럼 말이죠.
2) 이 경우에는 플래그가 “DIMIGO{” 혹은 “dimigo{” 로 시작하는 것을 알기 때문에 더욱이나 그럴 필요가 없었습니다.

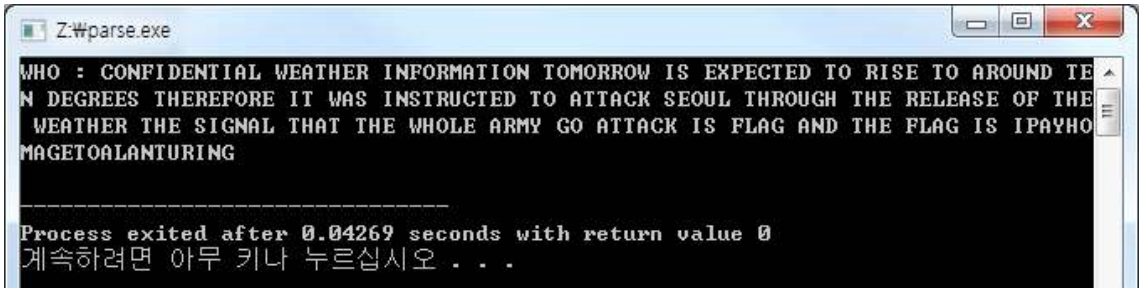
Riddle Machine(Rev,300) : 이 문제의 바이너리는 pyInstaller 로 감싸져 있는데, pydata 부분을 추출하고, pyinstxtractor.py를 이용하여 압축을 풀어주면.. RIDDLE 이라는 파일이 나옵니다. pyc 매직을 넣어주고, Easy Python Decompiler을 이용하여 코드를 보고, 조금 정리해보면.. 키가 될 수 있는 경우의 수가 26^3 으로 매우 작고[3], 암호화와 복호화 함수가 동일하여, Brute-Force 프로그램을 손쉽게 작성할 수 있었습니다.[4]

```
plain = "WKXVJIXWPQJX YVPRDIV BCDBEJXUQEX GFXVHLSL NH CQKPDNUZ KZ NQCC ND LTSZST
sTemp = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
for a1 in sTemp:
    for a2 in sTemp:
        for a3 in sTemp:
            ID = a1+a2+a3
            encode = ""
            Init()
            for i in plain:
                if i == ' ':
                    encode += ' '
                    continue
                IncIndicatorDrums()
                encode += chr(Scrambler(ID, DR, ord(i) - 65) + 65)
            print ID + " : " + encode
```

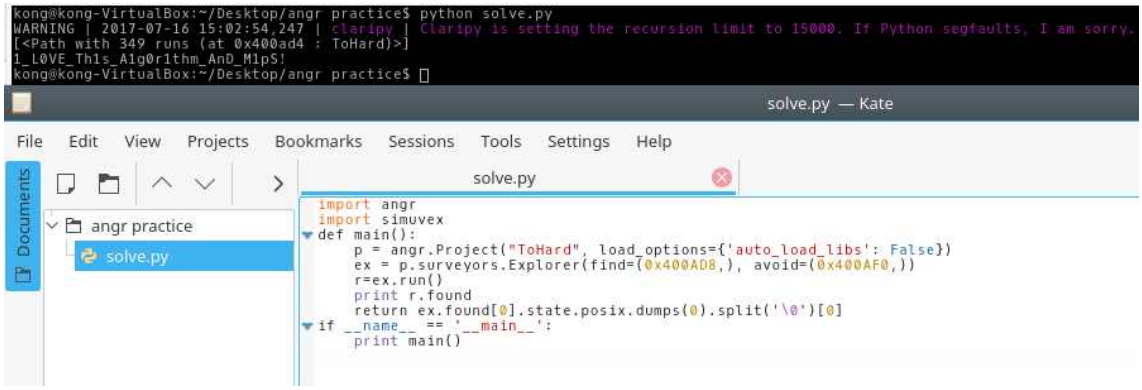
파이썬에서 메인함수를 이런식으로 변경 해주고, 명령프롬프트에서 파일로 저장 해준 다음 C 에서 불러와 작업을 하였습니다.

```
#include<stdio.h>
#include<string.h>
int main(void)
{
        FILE *fp = fopen("real_data.txt", "r");
        char *p;
        char buf[1000], buf2[1000];
        char vowel[7] = "AIEOUY";
        while (!feof(fp))
        {
        fgets(buf, 1000, fp);
        memcpy(buf2, buf, 1000);
        strtok(buf, " : ");
        strtok(NULL, " ");
        while (p = strtok(NULL, " "))
        {
        int cnt = 0;
        for (int i = 0;i < 6;i++)
        if (strchr(p, vowel[i]) != NULL) cnt++;
        if (cnt == 0)
        break;
        }
        if (p == NULL)
        printf("%s", buf2);
        }
}
```

3) 요즘 사용되는 블록 암호들은 2^{128} 정도는 기본이죠.
4) 영어 문장에서 나타나는 특성(각 단어에 반모음을 포함한 모음이 반드시 하나 이상은 들어간다)를 사용하면 키를 특정 지을 수 있습니다.



Too Hard(Rev,400): 평소 자주 분석하던 것과 다른 아키텍처 인데, IDA 마저 디컴파일을 지원하지 않을 때는 보통 Retargetable Decompiler[5]을 사용하거나 angr을 이용하게 되는데, 처음에는 angr로 안 풀려서 포기하려고 했으나 최신버전으로 업데이트 후 다시 해보니 돼서 편하게 풀었던 문제입니다.

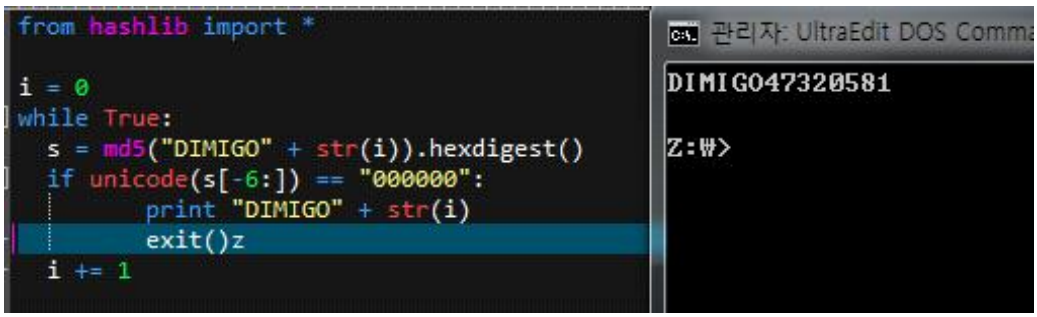


Too Easy(MISC,1) : dimigo{Welcome_to_DIMICON}

DIMI-CRYPT(MISC,100) :
HxD 에 주어진 값을 그대로 붙여 넣고 앞에서부터 인덱스 번호로 xor 한다음 다시 base64 로 디코딩 하면 되는 문제입니다.

5) 32비트 바이너리 한정

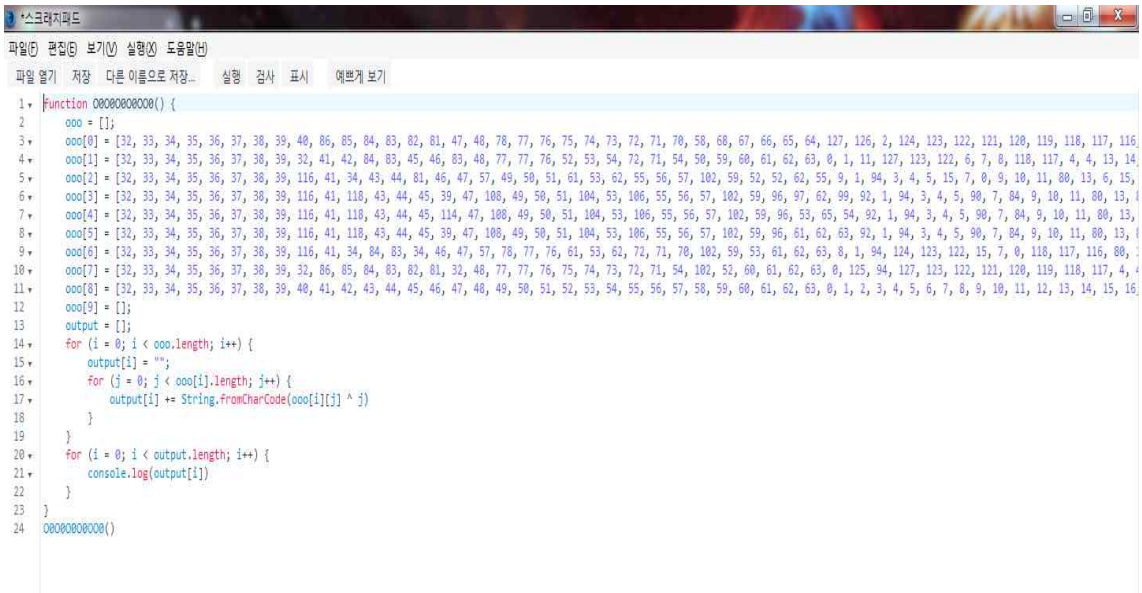
DIMI-Coin(MISC,100) :



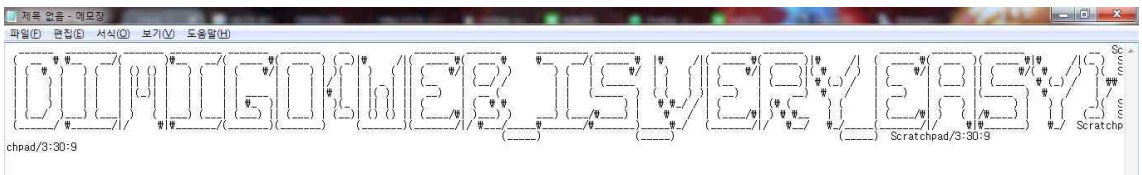
DIMI-114(MISC,100):

EXIF 정보에 있는 GPS 값을 이용하여 구글 맵에서 검색하는게 정석 풀이이지만, 사진의 제목과 설명에 광화문이라고 대놓고 나와 있어서, 그냥 바로 인증했습니다.

Find The Flag(Web,100):



script.js에서 뽑은 내용을 실행시키고, 결과를 적당한 에디터에 붙여 넣어서 보면...



Be a GoD(Web,200):
2048 게임이 끝나면 score.php 에 POST로 점수를 넘기는데, 이걸 9를 적당히 많이 끼워 넣어서, 전송해준 다음, rank.php 에 보면 플래그를 줍니다. Fiddler 의 Composer 기능을 사용하여 풀면 간단하게 풀 수 있습니다.[6)]

What is SQL? - Be a GoD 2(Web,300) :
이것도 기본적으로 Be a GoD 와 풀이가 같으나, loginUpdate.php 에 POST 로 DIMIGO'#[7)] 같은걸 넣어주고 나머지 과정을 모두 Be a GoD 와 똑같이 하면 풀 수 있습니다.

6) 이거, 다른사람들도 풀고 있어서 그런지 상당히 빠르게 해야합니다. 그래서 적당히 Log Requests 에 체크 해두고 미리 rank.php 에 빠르게 갈수 있도록 만든 다음, 빠르게 클릭 클릭 해주면 됩니다.
7) 싱글쿼터 + 뒤에 전부 주석