SV SCP

230712

CTF + what

김건희

230705 ~ 230712

CTF

- wargame write-up
  - Dreamhack
  - CodeEngn
- 첫 CTF 참여

- CTF 문제 제작
  - serial입력 문제
  - Win API 공부

WHAT

- KUCIS
  - Kali Linux 악성코드 제작 (Metasploit)

why serial

기본..?

들어가는 코드와 수식에 따라
난이도 조절+ 다양한 변형 가능

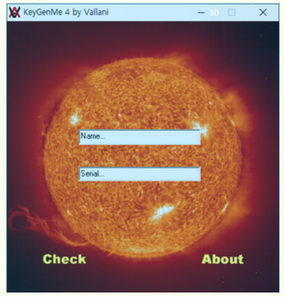
기본틀 코드

```
sigma@DESKTOP-ILHADQ2: ~
#include<stdio.h>
#include<string.h>
#define SIZE 13

int main(int argc, char* argv[]){
        int TF;
        char a1[SIZE];
        char arr[SIZE] = {0x74, 0x65, 0x73, 0x74, 0x5f, 0x74, 0x65, 0x73, 0x74, 0x31, 0x32, 0x74, 0x0};

        for (int j = 0; j < SIZE; j++) {
                printf("%c", arr[j]);
        }
        printf("\n");
        printf("Input : ");
        fgets(a1, SIZE, stdin);
        printf("%s\n", a1);
        for (int t = 0; t < SIZE; t++) {
                if (a1[t] == arr[t])
                        TF = 0;
                else {
                        TF = 1;
                        break;
                }
        }
        if (TF == 0)
                printf("collect\n");
        else
                printf("wrong\n");
        return 0;
```

실행시

```
sigma@DESKTOP-ILHADQ2:~$ ./ctftest_1
Input : test_test12t
collect
sigma@DESKTOP-ILHADQ2:~$ ./ctftest_1
Input : test_test123
wrong
```

what do this



- Name에 넣은 값을 반복문을 통해 serial 값 생성
- Key 값을 하나 주고 키값으로 반복문을 돌려 serial 생성

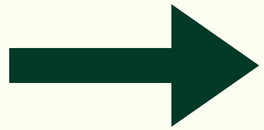
+ 패킹 후 키값을 언패킹 전에 PUSH 하게 코드이동(stolen byte)

기본틀 문제점

```
0x5555554007fb <main+49>     mov    byte ptr [rbp - 0x15], 0x74
0x5555554007ff <main+53>     mov    byte ptr [rbp - 0x14], 0x65
0x555555400803 <main+57>     mov    byte ptr [rbp - 0x13], 0x73
0x555555400807 <main+61>     mov    byte ptr [rbp - 0x12], 0x74
0x55555540080b <main+65>     mov    byte ptr [rbp - 0x11], 0x5f
0x55555540080f <main+69>     mov    byte ptr [rbp - 0x10], 0x74
0x555555400813 <main+73>     mov    byte ptr [rbp - 0xf], 0x65
0x555555400817 <main+77>     mov    byte ptr [rbp - 0xe], 0x73
0x55555540081b <main+81>     mov    byte ptr [rbp - 0xd], 0x74
0x55555540081f <main+85>     mov    byte ptr [rbp - 0xc], 0x31
0x555555400823 <main+89>     mov    byte ptr [rbp - 0xb], 0x32
```

배열의 flag값을
리버싱 툴 에서 바로 확인가능

응용 + 보완



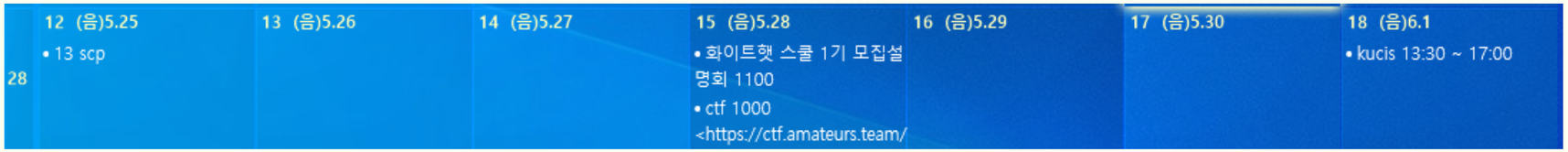
```
pwndbg> ni
0x00005555554007c8 in main ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
[ REGISTERS / show-flags off / show-compact-regs off ]
 RAX  0x0
 RBX  0x0
 RCX  0x0
 RDX  0x7fffffffe2f4 ◂— 0xdaa11a0000007fff
 RDI  0x7fffffffe2f0 ◂— 0x7fff00000000
 RSI  0x7fffffffe3e8 —▸ 0x7fffffffe603 ◂— '/home/sigma/ctf1_test'
 R8   0x7ffff7dced80 (initial) ◂— 0x0
 R9   0x7ffff7dced80 (initial) ◂— 0x0
 R10  0x1
 R11  0x0
 R12  0x5555554006b0 (_start) ◂— xor ebp, ebp
 R13  0x7fffffffe3e0 ◂— 0x1
 R14  0x0
 R15  0x0
 RBP  0x7fffffffe300 —▸ 0x555555400940 (__libc_csu_init) ◂— push r15
 RSP  0x7fffffffe250 —▸ 0x7fffffffe3e8 —▸ 0x7fffffffe603 ◂— '/home/sigma/ctf1_test'
*RIP  0x5555554007c8 (main+142) ◂— mov byte ptr [rbp - 0x6b], 0x73
[ DISASM / x86-64 / set emulate on ]
   0x5555554007b4 <main+122>    mov    byte ptr [rbp - 0x70], 0x70
   0x5555554007b8 <main+126>    mov    byte ptr [rbp - 0x6f], 0x74
   0x5555554007bc <main+130>    mov    byte ptr [rbp - 0x6e], 0x70
   0x5555554007c0 <main+134>    mov    byte ptr [rbp - 0x6d], 0x65
   0x5555554007c4 <main+138>    mov    byte ptr [rbp - 0x6c], 0x70
 ▶ 0x5555554007c8 <main+142>    mov    byte ptr [rbp - 0x6b], 0x73
   0x5555554007cc <main+146>    mov    byte ptr [rbp - 0x6a], 0x70
   0x5555554007d0 <main+150>    mov    byte ptr [rbp - 0x69], 0x74
   0x5555554007d4 <main+154>    mov    byte ptr [rbp - 0x68], 0x70
   0x5555554007d8 <main+158>    mov    byte ptr [rbp - 0x67], 0x5f
   0x5555554007dc <main+162>    mov    byte ptr [rbp - 0x66], 0x70
[ STACK ]
00:0000 | rsp 0x7fffffffe250 —▸ 0x7fffffffe3e8 —▸ 0x7fffffffe603 ◂— '/home/sigma/ctf1_test'
01:0008 |     0x7fffffffe258 ◂— 0x100000000
02:0010 |     0x7fffffffe260 ◂— 0xffffffff
03:0018 |     0x7fffffffe268 ◂— 0x0
04:0020 |     0x7fffffffe270 —▸ 0x7ffff7ffb2a8 ◂— add byte ptr [rax], al /* 'J' */
05:0028 |     0x7fffffffe278 ◂— 0x0
... ↓        2 skipped
[ BACKTRACE ]
 ▶ 0   0x5555554007c8 main+142
   1   0x7ffff7a03c87 __libc_start_main+231
```

```
[ STACK ]
00:0000 | rsp 0x7fffffffe250 —▸ 0x7fffffffe3e8 —▸ 0x7fffffffe603 ◂— '/
01:0008 |     0x7fffffffe258 ◂— 0x100000000
02:0010 |     0x7fffffffe260 ◂— 0x1fffffff
03:0018 |     0x7fffffffe268 ◂— 0x190000000c /* '₩x0c' */
04:0020 |     0x7fffffffe270 ◂— 0x65747fff00000000
05:0028 |     0x7fffffffe278 ◂— 'st_test12t'
06:0030 |     0x7fffffffe280 ◂— 0xa696e007432 /* '2t' */
07:0038 |     0x7fffffffe288 ◂— 0x0
```

Win API

win API 를 사용하여 문제만들기
Win API 공부 중....

do next week



CTF

Win API 공부 + CTF문제 만들기
CTFtime | CTF 참여
wargame

WHAT

KUCIS 강의 - 랜섬웨어
화이트햇 스쿨 1기 모집설명회

End