# CSED415: Lab 02 Report

20230499 / 김재환 / Kim Jaehwan

## 1. Overview

과제 환경에 접속하면 target 실행파일이 있다. target 실행파일에서 취약점을 찾아 print_flag() 함수를 실행시키도록 공격해야 한다.

## 2. Analyze target

실행파일만 제공되므로 GDB를 통해 바이너리를 분석해야 한다. 먼저 main 함수를 보면, main 함수의 주소를 출력한 후 vulnerable 함수를 실행시키고 있음을 알 수 있다.

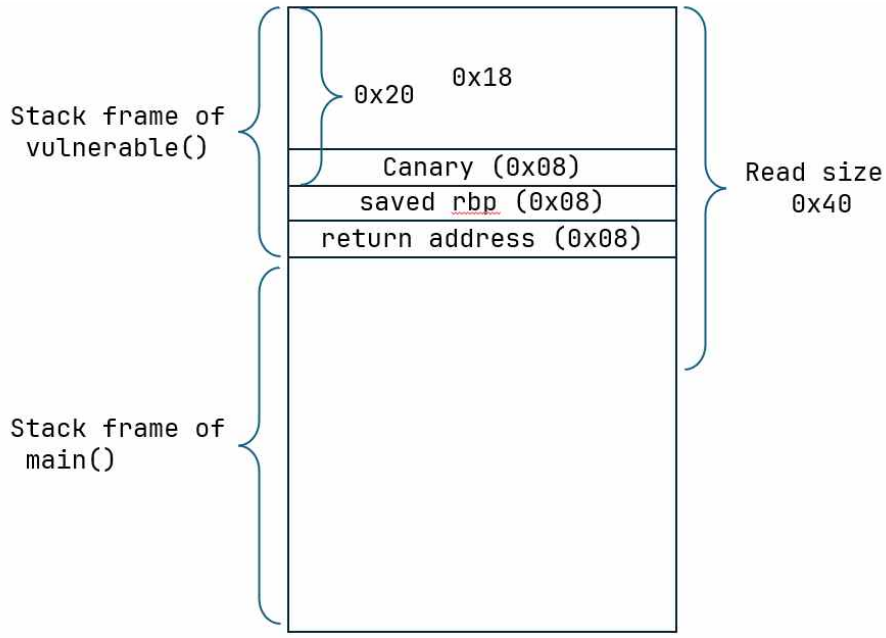
vulnerable 함수 안에서는 문자열을 최대 0x40개만큼 입력받은 뒤, 0x40 길이의 문자열의 모든 문자(바이트)에 대해서 바이트별로 검사를 수행한다. 0x40개의 모든 바이트가 0x20 이하이거나, 0x54 초과여야 한다. 모든 바이트에 대해 해당 검사가 모두 통과했다면 비정상 종료(exit 1) 없이 vulnerable 함수가 종료되게 된다.

그러나 입력 범위가 적절하지 않아, 입력 범위가 스택 프레임을 초과하게 된다. 이를 막기 위해서 바이너리 파일은 커스텀 카나리를 만들어 사용하고 있다.

실행 시 동작하는 main 함수, vulnerable 함수에서는 print_flag 함수 호출 부분을 찾을 수 없다. 따라서 스택 프레임의 리턴 주소를 덮어씌워 print_flag 함수를 호출하는 것이 적절할 것이다. 이를 위해서는 카나리의 값과 print_flag 함수의 주소가 필요하다.

## 3. Stack layout

바이너리를 분석해 그린 스택 레이아웃은 다음과 같다.




Read size인 0x40 바이트 영역을 바이트별로 검사한다.

## 4. Exploit the vulnerability

이 과제에서는 다음 취약점을 사용한다.

1. 실행파일에서는 버퍼 오버플로우 공격을 방어하기 위해 카나리를 직접 구현하였으나, 공격자가 카나리의 생성 로직을 따라 할 수 있다. gcc 컴파일러 등에서 기본적으로 제공하는 stack-protector가 아니라 rand() 함수를 이용해 직접 카나리를 구현하였다. 그러나 rand() 함수는 완전한 랜덤이 아니라 주어진 시드 값을 시작으로 어떤 계산을

통해 얻는 유사 랜덤 함수이므로, 시드 값을 알 수 있다면 카나리의 값을 알 수 있다.

취약한 바이너리는 srand(time(0)); rand();를 이용하여 카나리를 계산하고 있다. srand(time(0))은 현재 시간을 시드로 하여 유사 랜덤을 계산하라는 의미이기 때문에 프로그램의 실행 시간을 시드로 하여 rand() 함수를 이용하면 취약한 바이너리의 카나리와 동일한 카나리를 얻을 수 있다.

2. ASLR 기법을 이용해 여러 영역의 주소에 랜덤 오프셋을 더해 print_flag() 함수의 주소를 알 수 없게 해두었지만 main() 함수의 주소가 유출되고 있으므로 이를 추적할 수 있다. main 함수와 vulnerable 함수는 동일한 코드 영역에 있으므로 두 함수의 위치 차이 오프셋은 항상 동일하다. 따라서 취약한 바이너리가 출력하는 main 함수에 일정한 오프셋을 더해서 print_flag 함수의 주소를 알 수 있다.

카나리와 리턴 주소를 알아냈다 하더라도 리턴 주소로 이동하기 위해서는 0x40바이트가 검사를 통과하도록 해야 한다. 카나리와 리턴 주소 부분이 아닌 부분은 임의로 0x54보다 큰 바이트, 예를 들어 영어 소문자 등을 입력하면 되지만, 카나리와 리턴 주소는 입력해야 할 값이 정해져 있다. 따라서 카나리와 리턴 주소에 0x20 초과, 0x54 이하 바이트가 있다면 익스플로잇이 실패하게 된다. 그러나 이 두 값은 실행할 때마다 값이 달라지므로 검사를 통과할 때까지 공격을 시도하면 된다.

## 5. Result

exploit 파일을 통해 얻은 플래그를 첨부하였다. 성공적으로 플래그를 얻어내었다.

```
csed415-lab02@csed415:/tmp/KJH$ python3 loop_payload.py
(생략..)
[+] Starting local process '/home/csed415-lab02/target': pid 2015
[+] Receiving all data: Done (1.05KB)
[*] Process '/home/csed415-lab02/target' stopped with exit code 0 (pid 2015)
<class 'bytes'>
Enter input: This is your flag:

944583A6CFFB89C892AEABE82B57E278E3C7E11D863FC79B091BF71E9AE70425
B8A8E18F0E9EDD5531DDE3684F9032320574E9B1F7FBD626C5A0C74AD1AC4490
7C169759BFDF7307A996F6C23DB9D8B2D407F7A30E82696C743C2C840E5B4966
32039E3E15E07ACD63794785BB3E573296370E6AC9315E277D5013A8A5B72EC7
96CC6C829E221C7F939EC8B6BCC2B2DA5728E03F2E87685A4359F2546C74713D
0798ED30AD551051594F4ABAECA5522800FF547C2B8E4F83C20E427409B61FB2
7E6A1523801B16B90E28E794A8BD43C7E6AEF6E40E9EE2F444BE28228B2FE446
0422F78E6A196D37ED6537B8366700390419613FED089168B3CC1089AEAAAF1B
4A746F460C13E5200A299CBC905F439F123E22F070DC6B1E3023BB1E70FDECBC
C33B63FC058222A7A73C2E41BE34B66B17ABAF583D8D1774F8C4945EBCC4C134
44683E0195DEE2E398201C02B9804DE2693828528E0468BCAC6AE2CD4F4EA008
75D078A119303FB924C74AF976EDE8B58BEACB34ED1170879ADEEFB61B6A86A6
E246AC349E4D538166B5A1E38E26B51670504DAF84E7BDEA02FC6C59A8A1052B
2CF00C3CB538B4A83C8E1181ACF729F2C57832973046A69FB0CDDCDF7173F8D9
841E08469087308499F70A08D3BC17867608606039290 67D11149D80305AA477
1EA03136520658D6A41C4ED22D64184070A46B96AC30CCBAFD25D3637315DAFA
```