

# PETIR CYBER SECURITY

## QUALIFICATION 2019



Nama Lengkap	:	<u>Stephanley Herman</u>
Jurusan	:	<u>Cyber Security</u>
NIM	:	<u>2301866113</u>
Username	:	<u>pandyhermann3</u>

## [Judul Soal]

### Langkah Penyelesaian:

(tuliskan langkah penyelesaian disini, sertakan screenshot bila perlu).

**Flag:** (flag yang didapatkan ditulis disini)

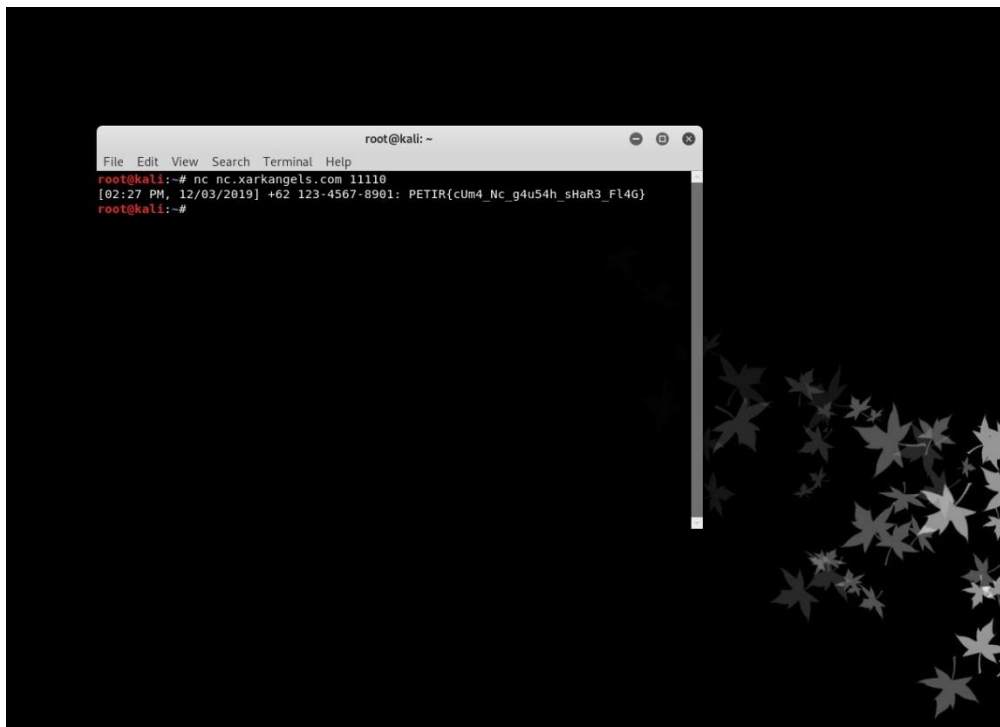
---

[ence]

### Langkah Penyelesaian:

**netcat** (often abbreviated to **nc**) is a computer networking utility for reading from and writing to network connections using **TCP** or **UDP**.

Dengan informasi ini saya memasukan "nc nc.xarkangels.com 11110" Ke terminal Linux yang sudah terinstall NetCat.



**Flag:** `Petir{cUm4_Nc_g4u54h_sHaR3_Fl4G}`

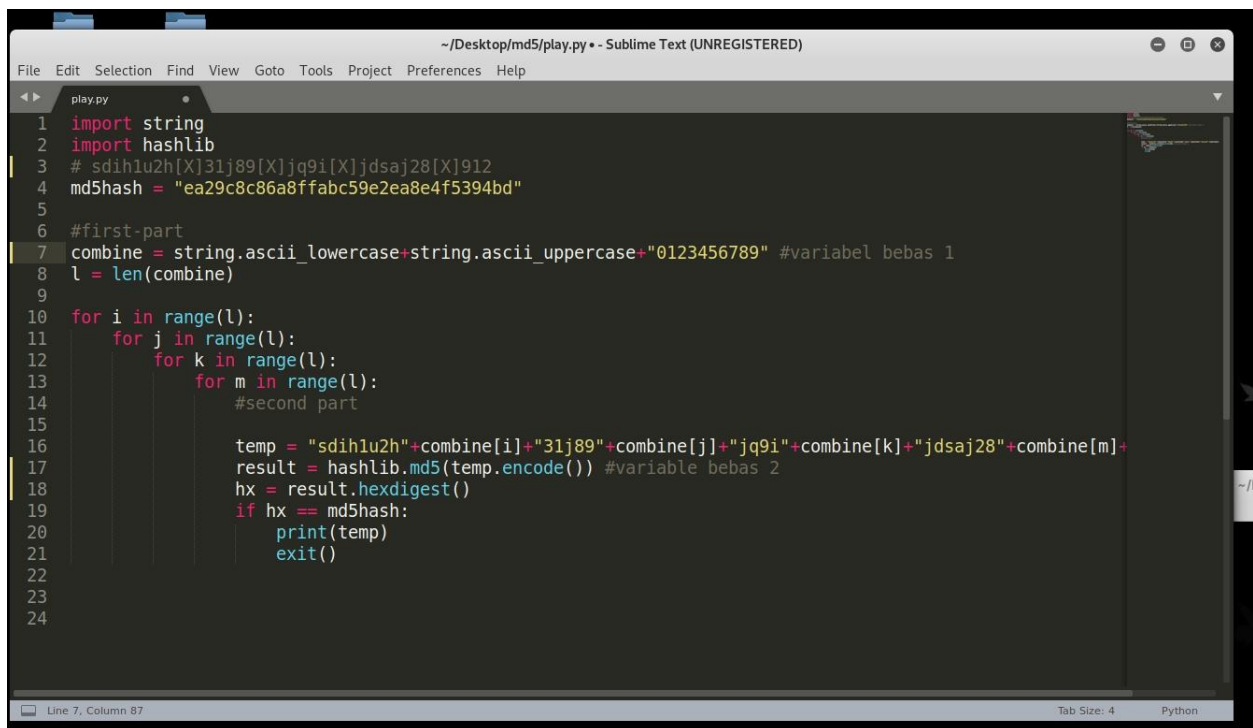
## [hbrute]

### Langkah Penyelesaian:

Seperti yang dinyatakan soal, kita memiliki hasil hash, saya sudah mengidentifikasi ini semua md5-hash, dengan melihat Panjang hash-nya, hash sebuah md-5 panjangnya 32 karakter.

Ada hash lain yang panjangnya 32 karakter juga (128 bit), tapi ini tebakan pertama saya.

Dari diberitahu soal bahwa bagian 'X' itu adalah kata hilang yang musti kita ganti.



```
~/Desktop/md5/play.py • Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

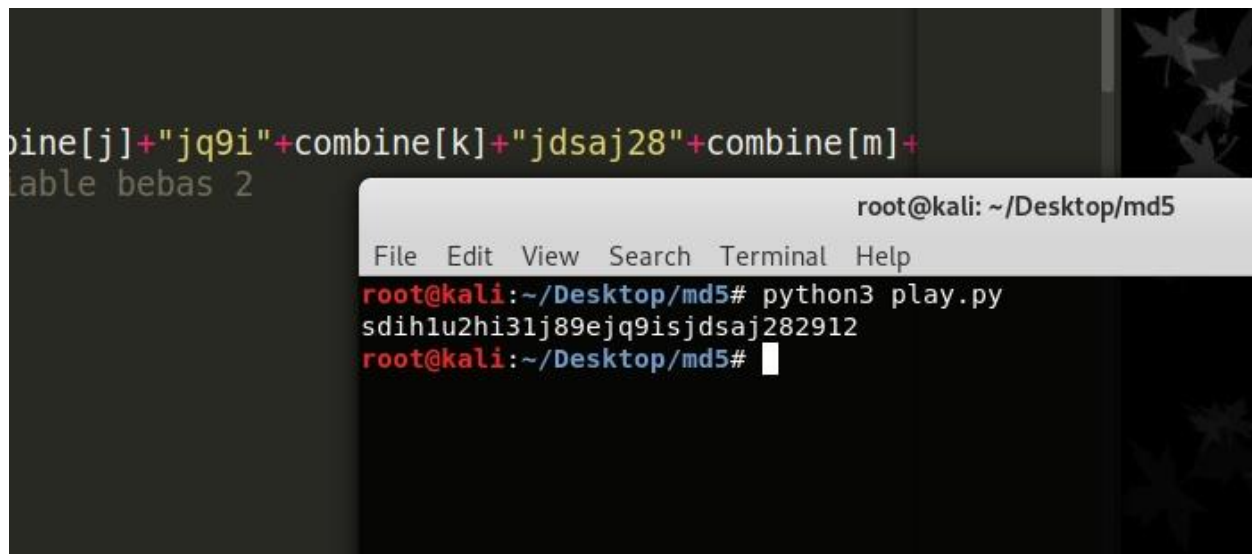
play.py
1 import string
2 import hashlib
3 # sdi1u2h[X]31j89[X]jq9i[X]jdsaj28[X]912
4 md5hash = "ea29c8c86a8ffabc59e2ea8e4f5394bd"
5
6 #first-part
7 combine = string.ascii_lowercase+string.ascii_uppercase+"0123456789" #variabel bebas 1
8 l = len(combine)
9
10 for i in range(l):
11     for j in range(l):
12         for k in range(l):
13             for m in range(l):
14                 #second part
15
16                 temp = "sdi1u2h"+combine[i]+"31j89"+combine[j]+"jq9i"+combine[k]+"jdsaj28"+combine[m]+
17                 result = hashlib.md5(temp.encode()) #variable bebas 2
18                 hx = result.hexdigest()
19                 if hx == md5hash:
20                     print(temp)
21                     exit()
22
23
24
```

Jadi pertama saya buat script simple di python. Yang memiliki dua Variable bebas.

Variabel bebas 1, (variable-nya bernama combine di line 7) variable dari kombinasi yang saya pilih untuk substitusi X.

Variabel bebas 2, md5 hashing yang apa bila tidak ditemukan saya akan coba tipe hasing yang lain. (line 17)

Saya jalankan script-nya,



```
combine[j]+"jq9i"+combine[k]+"jdsaj28"+combine[m]+  
#variabel bebas 2  
root@kali: ~/Desktop/md5  
File Edit View Search Terminal Help  
root@kali:~/Desktop/md5# python3 play.py  
sdihl2hi31j89ejq9isjdsaj282912  
root@kali:~/Desktop/md5#
```

Dibutuhkan 13 detik untuk script saya mendapatkan plaintext yang asli.



```
combine = string.ascii_lowercase+string.ascii_uppercase+"0123456789" #variabel bebas 1  
l = len(combine)  
for i in range(l):  
    for j in range(l):  
        for k in range(l):  
            for m in range(l):  
                #second part  
                temp = "sdihl2h"+combine[i]+"31j89"+combine[j]+"jq9i"+combine[k]+"jdsaj28"+combine[m]+"912"  
                result = hashlib.md5(temp.encode()) #variable bebas 2  
                hx = result.hexdigest()  
                if hx == md5hash:  
                    print(temp)  
                    exit()
```

Jadi pertama dengan menggunakan Variabel bebas 1 yang panjangnya:

26 alphabet kecil + 26 alphabet besar + 10 base10 digit = 62

$62 \times 62 \times 62 \times 62 = 14776336$  kombinasi [x][x][x][x]

Apa yang saya lakukan ada mencycle semua kemungkinan dari keempat variable saya [i] , [j] , [k] , [m].

Dan setiap kemungkinan saya ubah jadi md5 Hash yang masih dalam bentuk Hex yang kemudian saya ubah jadi string.

Menggunakan `.hexdigest()` , kemudian saya bandingkan dengan md5hash yang sudah diberikan, apabila hasil-nya sama akan dimunculkan ke layer seperti diatas.

Ketika saya coba rupanya beruntung bahwa kombinasi dari [X][X][X][X] yang tidak diketahui ada di Variable bebas 1 dan juga ini merupakan md5 hash.

**Flag:** Petir{sdiH1u2hi31j89eq9isjdsaj282912}

## [Basic]

### Langkah Penyelesaian:

Cipher text yang diberikan adalah ini:

\*\*

```
VXpza3BnVHh0dHZmU3Z3dnZ4dCBUYXl4OlpgSTeZLCBDbHVlOlZpZ2VuZXJlIEtF
WTpQRVRJUg==
```

\*\*

Saya pertama menebak ini adalah sebuah Base64 yang sudah di encode, dikarenakan ada padding "==" dibelakang.

Padding ada di base64 karena Panjang dari plaintext(N) yang diberikan tidak ( $N \% 3 == 0$ ) dan apabila ada dua padding berarti Panjang plaintext awal adalah  $x3$  (perkalian dari 3) + 1.

Ini pertama Cuma hipotesis.

Jadi saya coba memasukan ini kedalam base64 decoder encoder online, yang saya dapat adalah berikut.

\*\*

```
UzskpgTtxtvfSvwwvxt Tayx:ZFI13, Clue:Vigenere KEY:PETIR
```

\*\*

Dari hint yang diberikan adalah, and I quote:

"

*If you see any clues in your cipher, use it and throw the clues away after.*

"

Oke jadi sekarang ada Clue yaitu Vigenere.

Jadi saya masukan lagi ke Vigenere cipher yang online menggunakan key yang diberikan, tanpa meng-include Clue dan KEY.

Saya menginput "UzskpgTtxtvfSvwwvxt Tayx:ZFI13" saja.

Dan saya mendapatkan.

\*\*

FvzcyrPelcgbZnfgrel Clue:ROT13

\*\*

Oke clue selanjutnya ROT13, jadi saya menggunakan ROT13 online dan mendapatkan

\*\*

SimpleCryptoMastery

\*\*

Website yang digunakan:

<https://www.base64decode.org/>

<https://www.dcode.fr/vigenere-cipher>

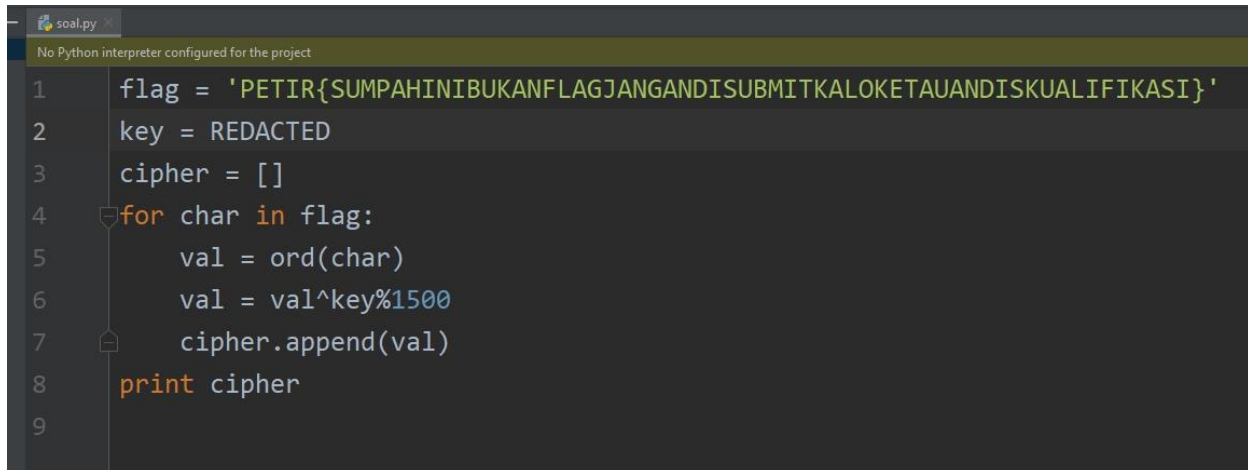
<https://rot13.com/>

**Flag:** Petir{ SimpleCryptoMastery}

## [Key]

### Langkah Penyelesaian:

Diberikan dua file, soal.py and cipher



```
1 flag = 'PETIR{SUMPAHINIBUKANFLAGJANGANDISUBMITKALOKETAUANDISKUALIFIKASI}'
2 key = REDACTED
3 cipher = []
4 for char in flag:
5     val = ord(char)
6     val = val^key%1500
7     cipher.append(val)
8 print cipher
9
```

Dari apa yang saya lihat bahwa sudah diberi peringatan yang sangat jelas bahwa variable flag isinya adalah bukan flagnya.

Dari yang saya lihat program ini berjalan seperti ini;

Isi dari flag satu persatu di ambil dan di copy ke variable char(line 4) dalam bentuk karakter

Kemudian variable char tersebut akan diubah menjadi order-nya dalam ASCII dalam bentuk base10 dan disimpan di val (line 5)

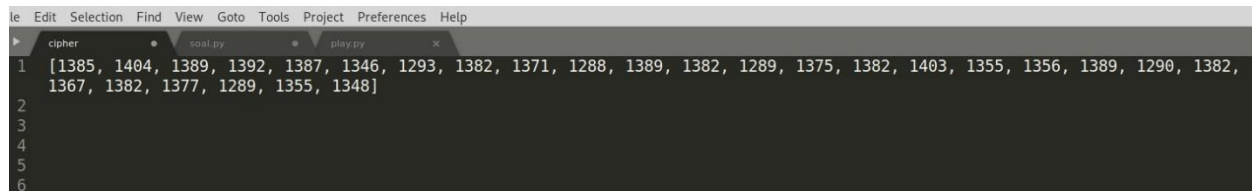
Kemudian kita akan melakukan XOR, untuk val XOR key MOD 1500.

Saya tidak begitu mempedulikan MOD 1500 , karena ini Cuma berarti apa pun yang didapat akan dikecilkan  $< 1500$ .

Kemudian hasilnya akan kita tambahkan ke cipher.



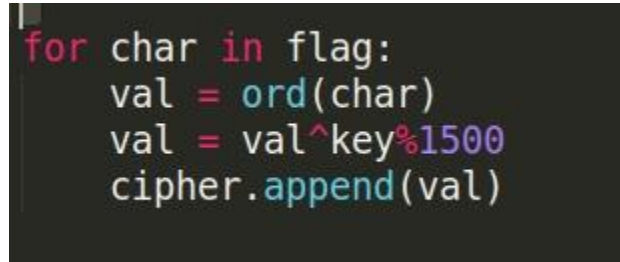
Kita dapat file yang bernama cipher yang berisi



```
le Edit Selection Find View Goto Tools Project Preferences Help
cipher
1 [1385, 1404, 1389, 1392, 1387, 1346, 1293, 1382, 1371, 1288, 1389, 1382, 1289, 1375, 1382, 1403, 1355, 1356, 1389, 1290, 1382,
2 1367, 1382, 1377, 1289, 1355, 1348]
3
4
5
6
```

Jadi hipotesis saya dalam ini adalah,

File Cipher ini adalah hasil dari



```
for char in flag:
    val = ord(char)
    val = val^key%1500
    cipher.append(val)
```

Pertama untuk saya buktikan ini saya mencoba ini.

Bila saya benar proses-nya seperti ini

CHAR → VAL^KEY%1500 → CIPHER

Saya tau susunan dari flag adalah PETIR{.....}

Jadi kita bisa hipotesis kan bahwa ini benar

'P' → VAL^KEY%1500 → [1385]

'P' XOR KEY → CIPHER(yaitu 1385, index ke 0 dari cipher text yang diberikan)

'P' bila diubah menggunakan ord() adalah 80

80 XOR KEY → 1385

Dan kita juga bisa menggunakan salah satu peraturan XOR

Untuk melakukan reverse programming.

Salah satu peraturan XOR itu adalah

a XOR b = c is equal with b xor c = a

oke pertama saya coba untuk

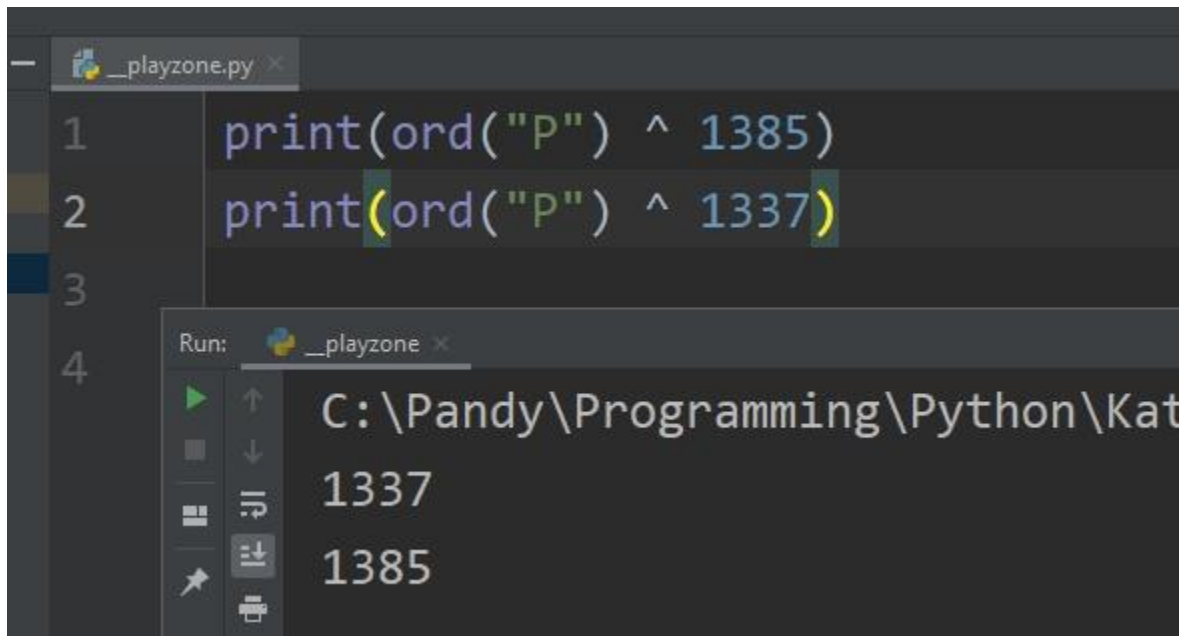
80 XOR 1385 → KEY



```
1 print(ord("P") ^ 1385)
```

Run: C:\Pandy\Programming\Python  
1337

Dan untuk memastikan saya tidak salah saya coba mengembalikannya



```
1 print(ord("P") ^ 1385)
2 print(ord("P") ^ 1337)
```

Run: C:\Pandy\Programming\Python\Kat  
1337  
1385

Oke benar,

Sekarang saya menghipotesiskan bahwa key-nya semua akan memiliki nilai yang sama yaitu 1337

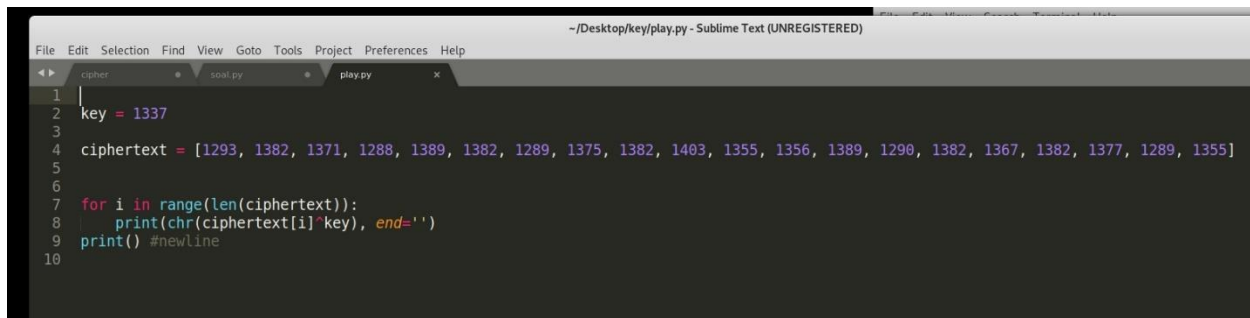
Tapi saya tetap mencoba dan mendapatkan



```
1 print(ord("P") ^ 1385)
2 print(ord("E") ^ 1404)
3 print(ord("T") ^ 1389)
4 print(ord("I") ^ 1392)
5 print(ord("R") ^ 1387)
```

Run: C:\Pandy\Pro  
1337  
1337  
1337  
1337  
1337

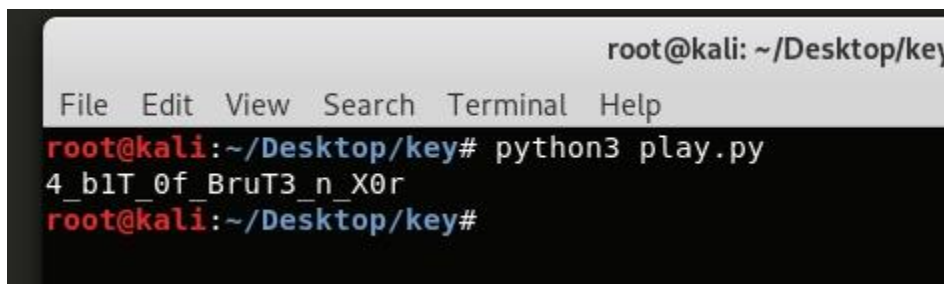
Oke, sekarang berarti saya bisa membuat script untuk sisa dari cipher dengan menggunakan key yang udah saya dapatkan.



```
~/Desktop/key/play.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
1
2 key = 1337
3
4 ciphertext = [1293, 1382, 1371, 1288, 1389, 1382, 1289, 1375, 1382, 1403, 1355, 1356, 1389, 1290, 1382, 1367, 1382, 1377, 1289, 1355]
5
6
7 for i in range(len(ciphertext)):
8     print(chr(ciphertext[i]^key), end='')
9 print() #newline
10
```

Variable ciphertext(line 4) saya dapat dari cipher.txt yang diberikan oleh soal.

`chr()` fungsinya adalah mengubah integer menjadi ASCII Characters saya menghilangkan bagian depan yaitu `PETIR{` dan bagian belakang `}`



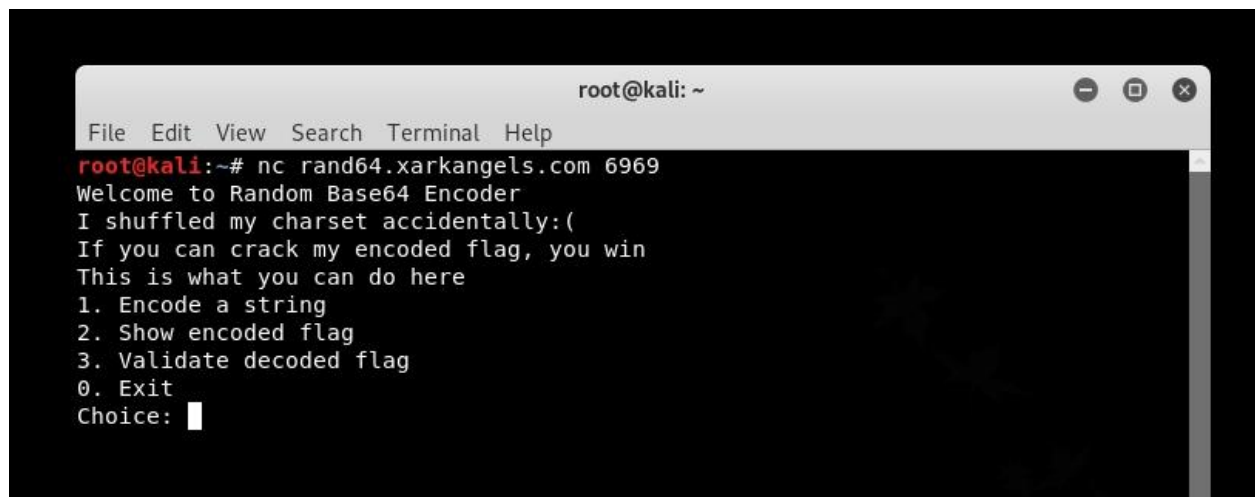
```
root@kali: ~/Desktop/key
File Edit View Search Terminal Help
root@kali:~/Desktop/key# python3 play.py
4_b1T_0f_BruT3_n_X0r
root@kali:~/Desktop/key#
```

**Flag:** PETIR{4\_b1T\_0f\_BruT3\_n\_X0r}

## [Rand64]

### Langkah Penyelesaian:

nc rand64.xarkangels.com 6969



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nc rand64.xarkangels.com 6969  
Welcome to Random Base64 Encoder  
I shuffled my charset accidentally:(  
If you can crack my encoded flag, you win  
This is what you can do here  
1. Encode a string  
2. Show encoded flag  
3. Validate decoded flag  
0. Exit  
Choice: █
```

Oke jadi cara kerja base64, adalah dia mengambil 3 8bit karakter dan dibuat menjadi 4 6bit,

01000100 01000100 01000100

01000100 01000110 01000101

010001 000100 011001 000101

Dan dari 4 6bit tersebut akan diubah ke base10

010001 = 17

000100 = 4

011001 = 25

000101 = 5

Dan masing-masing akan menjadi Index dari charset yang berisi 64;

Charset[17]+Charset[4]+Charset[25]+Charset[5].

Terus akan dimasukan charset index itu kedalam string yang sudah di encoded.

Perbedaannya adalah sekarang kita tidak tahu apa yang terletak di index tersebut, seperti kita tidak tahu karena ini random apa yang bakal terletak di Charset[4], karena random.

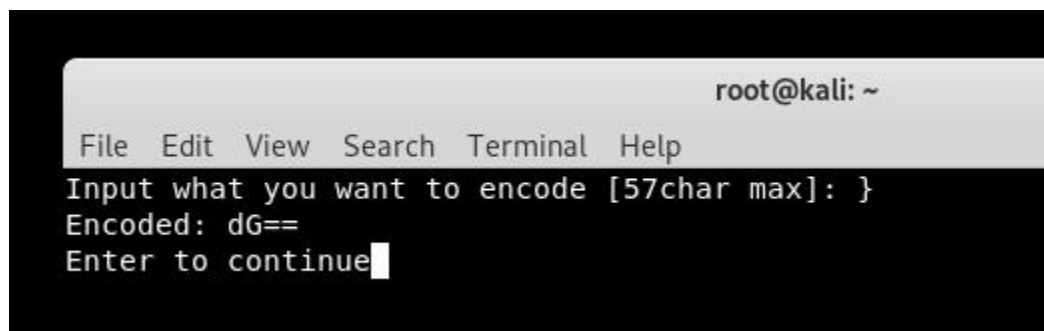
Tapi kita dibantu soal karena soal bisa menunjukan kita encoded flag dan kita bisa memvalidasi membenaran tersebut.

Saya tahu 6 karakter pertama adalah **PETIR{**

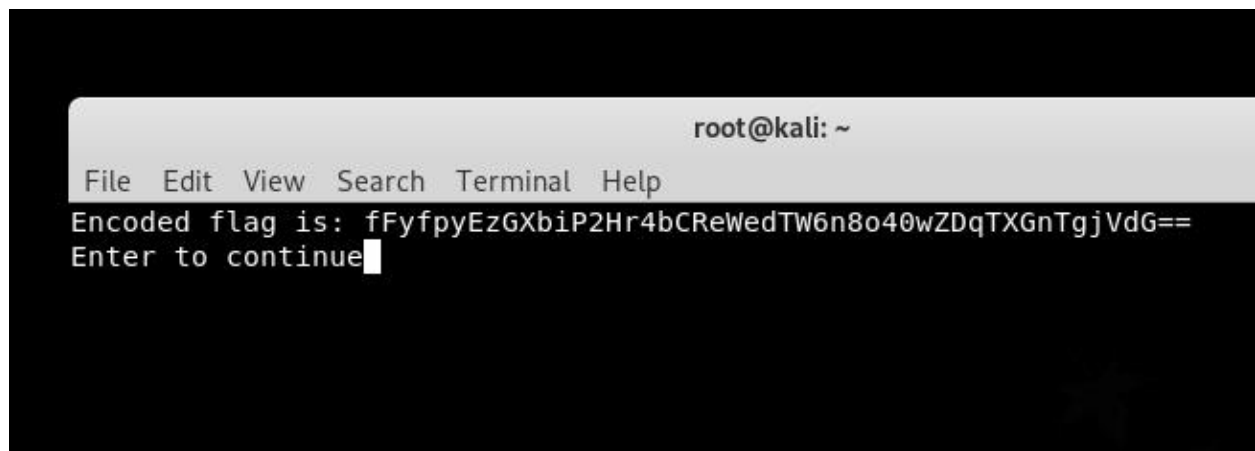
Berarti sudah terpakai **PET** menjadi 4 karakter 6bit

**IR{** akan menjadi 4 karakter 6bit selanjutnya

Dan rupanya juga yang paling belakang adalah satu karakter **}**.



```
root@kali: ~  
File Edit View Search Terminal Help  
Input what you want to encode [57char max]: }  
Encoded: dG==  
Enter to continue
```



```
root@kali: ~  
File Edit View Search Terminal Help  
Encoded flag is: fFyfpYEzGXbIP2Hr4bCRWedTW6n8o40wZDqTXGnTgjVdG==  
Enter to continue
```

Lihat karakter paling belakang adalah dG==.

Saya sudah coba reset dan yang paling belakang selalu sama

Dan yang depan **PETIR{** juga selalu sama.

Dari ini saya tahu bahwa isinya plaintext flag tanpa **PETIR{** dan **}**

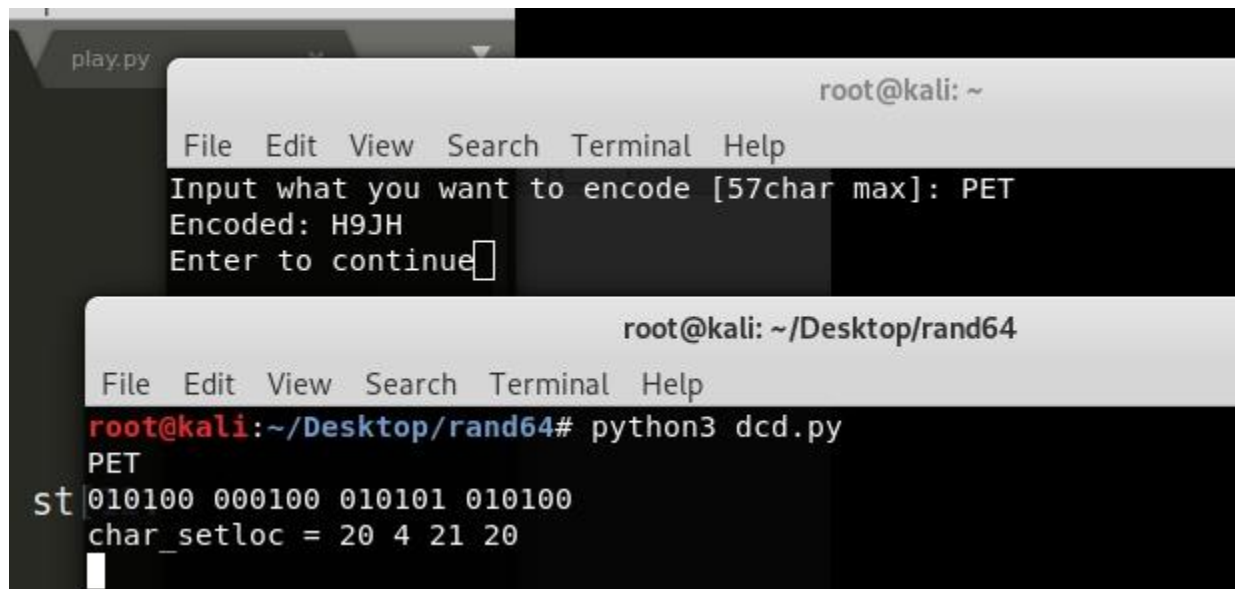
Adalah perkalian dari 3.

Jadi apa yang saya musti lakukan adalah saya musti ambil dari 4 6bit yang sudah dihasilkan oleh setiap perkalian tiga.

Saya hitung bahwa ditengah ada 9 baris 8bit yang diubah menjadi 36 6bit.

Charsetnya mungkin memang random, tapi index lokasi mereka selalu sama.

Apa yang saya lakukan adalah mencoba input 3 karakter asal ke encoder, kemudian dari hasil 4 6bit saya pinpoint lokasi mereka berada di index berapa di charset tersebut.



```
root@kali: ~  
File Edit View Search Terminal Help  
Input what you want to encode [57char max]: PET  
Encoded: H9JH  
Enter to continue  
  
root@kali: ~/Desktop/rand64  
File Edit View Search Terminal Help  
root@kali:~/Desktop/rand64# python3 dcd.py  
PET  
st 010100 000100 010101 010100  
char_setloc = 20 4 21 20
```

01010000 0100000101 01010100 = P E T

010100 010110 011101 100010

Charset[20] + Charset[4] + Charset[21] + Charset[20]

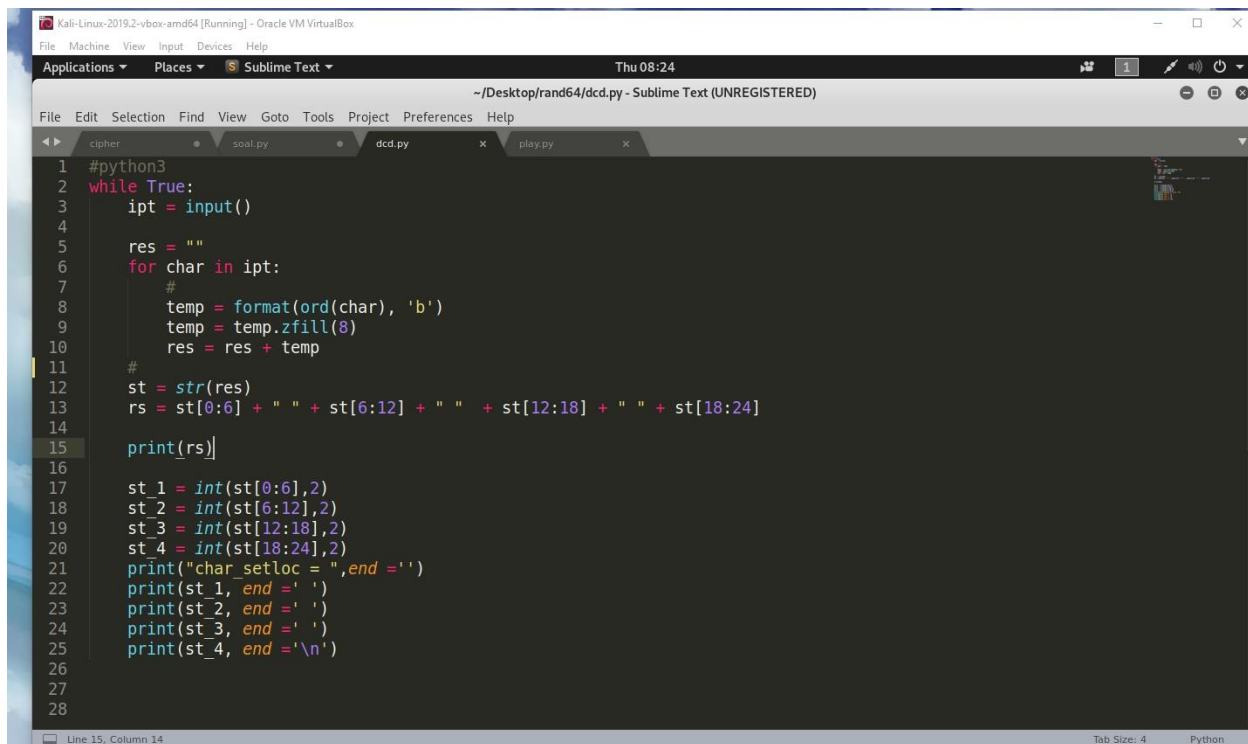
Itu berisi

Charset[20] = H

Charset[4] = 9

Charset[21] = J

Script yang saya gunakan adalah ini.



```
1 #python3
2 while True:
3     ipt = input()
4
5     res = ""
6     for char in ipt:
7         #
8         temp = format(ord(char), 'b')
9         temp = temp.zfill(8)
10        res = res + temp
11    #
12    st = str(res)
13    rs = st[0:6] + " " + st[6:12] + " " + st[12:18] + " " + st[18:24]
14
15    print(rs)
16
17    st_1 = int(st[0:6],2)
18    st_2 = int(st[6:12],2)
19    st_3 = int(st[12:18],2)
20    st_4 = int(st[18:24],2)
21    print("char_setloc = ",end='')
22    print(st_1, end=' ')
23    print(st_2, end=' ')
24    print(st_3, end=' ')
25    print(st_4, end='\n')
26
27
28
```

(line 6)

semua input karakter saya ubah kedalam bentuk binary.tetapi input binary panjangnya tidak 8bit karena ada yang paling depannya kosong.

(line 9)

Sebelum saya append saya membuat panjang binary-nya menjadi 8bit.

(line 12&13)

Saya ubah menjadi string supaya bisa dipisahkan dengan spasi di line 13

(line 17&25)

Saya gunakan untuk mengubah binary tersebut menjadi decimal yang saya tampilkan satu"

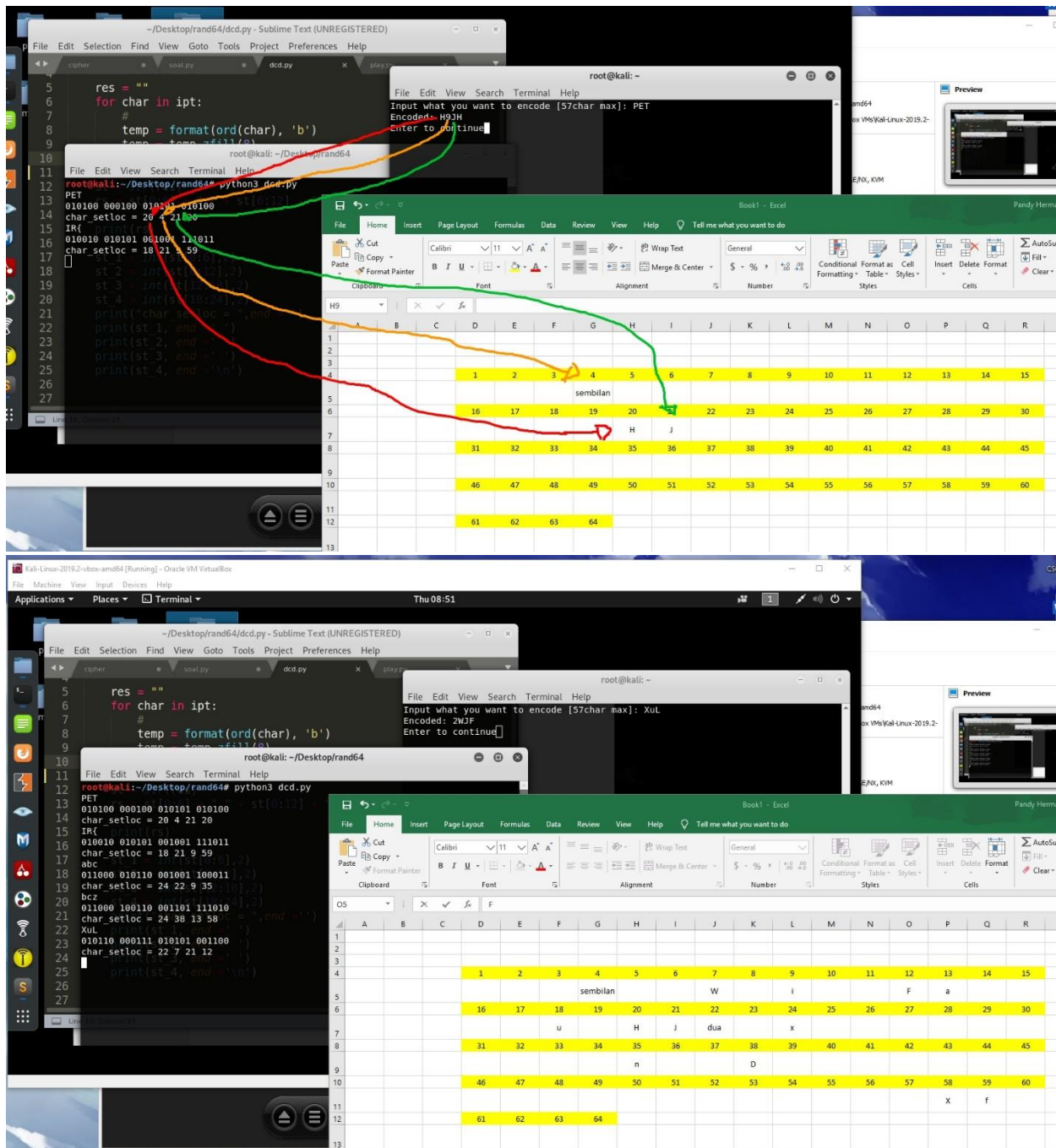
Saya terus melakukan ini sampai charsetnya setidaknya lumayan terisi.

Dan isinya saya catat di Excel Seperti ini

Saya melakukan ini terus sampai beberapa kali.

Sampai sudah lumayan terisi sampai tinggal sedikit yang belum terisi.

Ketika ini sudah lumayan terisi saya bisa piece kembali dari 4 6bit ini menjadi 3 8bit.



Lama-Lama akan terisi dan bisa di substitusi di encoded Flagnya.



```
root@kali: ~
File Edit View Search Terminal Help
Encoded flag is: H9JHuJifkDgdFNQLagS4Z0Zzt0TqebaEmpyYtDkqt2cGzk==
Enter to continue
```

H9JH = 20 4 21 20

uJif = 18 21 9 59

kDgd = ... ..

FNQL = ... ..

agS4 = ... ..

Z0Zz = ... ..

t0Tq = ... ..

ebaE = ... ..

mpyY = ... ..

tDkq = ... ..

t2cG = ... ..

zk== = 31 16

dari tabel yang kita punya sekarang kita bisa substitusi beberapa

H9JH = 20 4 21 20

uJif = 18 21 9 59

kDgd = ... 38 ...

FNQL = ... ..

agS4 = ... ..

Z0Zz = ... ..

t0Tq = ... ..

ebaE = ... 13 ...

mpyY = ... ..

tDkq = ... ..

t2cG = ... ..

zk== = 31 16

apabila satu baris sudah terisi saya bisa melakukan reversenya, sebagai contoh:

baris pertama 20 4 21 20

010100 010110 011101 100010

010100 010110 011101 100010

01010001 01100111 01100010 Dalam ascii text ini adalah PET

saya melakukan substitusi ini terus sekitar 10 kali, dan setelah 10 kali, tabelnya akan menjadi penuh tapi masih akan ada yang kosong.

Sebagai contoh.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64											

Tidak memiliki isi

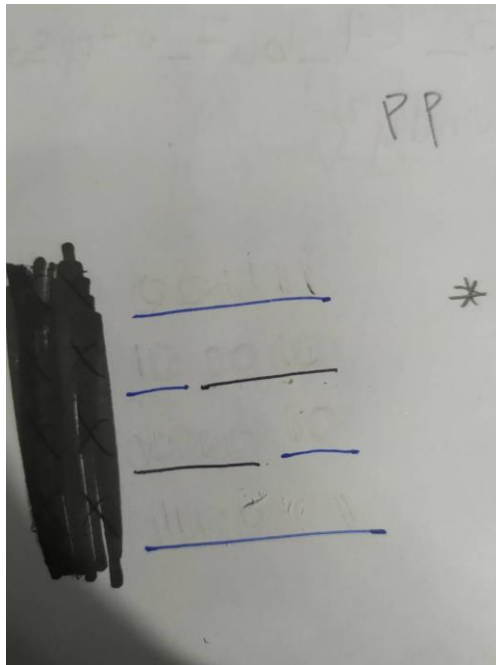
Sudah memiliki isi

①	20	4	21	20
②	18	21	9	59
③	16	38	5	51
④	12	53		54
⑤	13	5	61	34
⑥	29	19	29	
⑦	27		17	48
⑧	28	3	13	36
⑨	23	55		52
⑩		38	16	48
⑪	27	22	49	57
⑫	31	16		

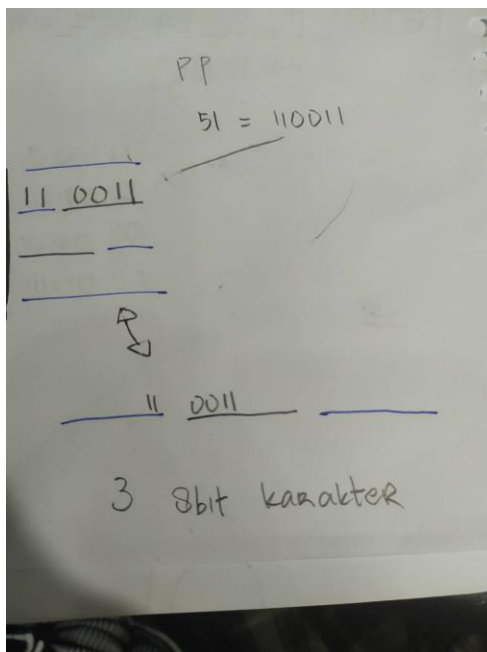
Sebagai contoh keadaan substitusi nya seperti ini kemaren, masih ada yang kosong.

Dan menurut saya apabila saya tebak terus akan sangat lama untuk mengisi yang tidak memiliki saja. Karena pada titik ini saya Cuma mendapatkan index yang saya sudah miliki

jadi saya buat seperti ini.

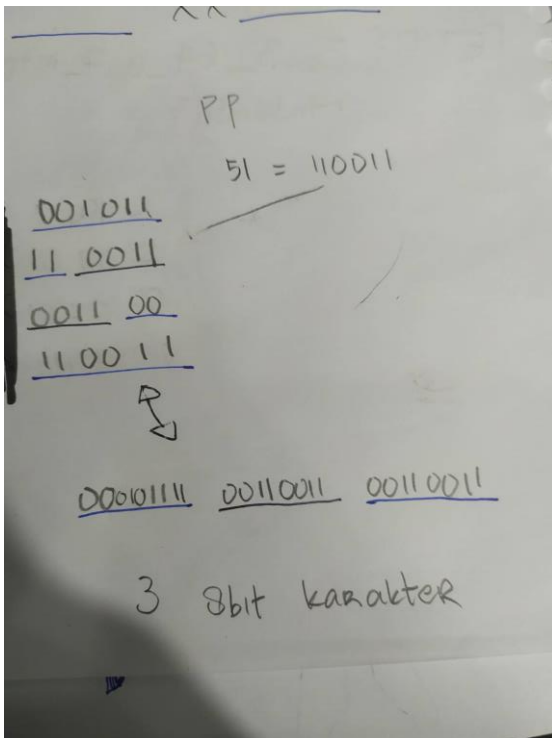


Apabila ingin mencari 51.



Kemudian sisanya musti di isi dengan bit yang bisa menghasilkan 3 8bit karakter yang memiliki simbol agar bisa kita masukan kedalam soal program.

Lokasi pemasukan yang ingin dicari bisa dibaris 1 sampai 4, tergantung yang mana yang sesuai.



00101111 00110011 00110011

Akan menghasilkan  
/33

Apabila saya susunnya

00001111 00110011 00110011

Ini tidak valid karena dalam tabel  
ascii 15 adalah (Shift In) yang tidak  
memiliki simbol

33

Saya tidak bisa menginput ini ke soal  
lewat keyboard saya.

```

in ipt:
    = format(ord(char), 'b')
    = temp + fill(0)
    res +

res)
:6] + 001011 110011 001100 110011
char_setloc = 11 51 12 51

t(st[0:6],2)
t(st[6:12],2)
t(st[12:18],2)
t(st[18:24],2)
  
```

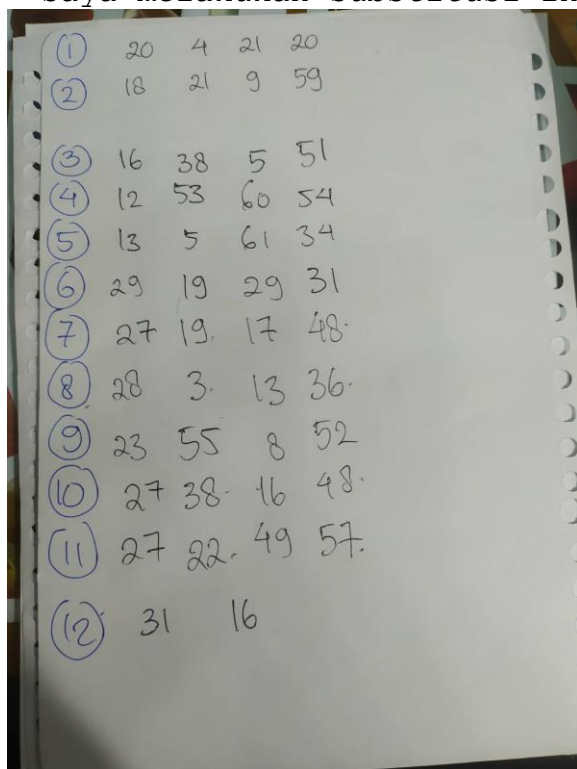
```

root@kali: ~/Desktop/rand64
File Edit View Search Terminal Help
Input what you want to encode [57char max]: /33
Encoded: 0LwL
Enter to continue

root@kali: ~/Desktop/rand64# python3 dcd.py
/33
001011 110011 001100 110011
char_setloc = 11 51 12 51
  
```

51 ada kawan"

Saya melakukan substitusi ini terus sampai tabel saya penuh.



①	20	4	21	20
②	18	21	9	59
③	16	38	5	51
④	12	53	60	54
⑤	13	5	61	34
⑥	29	19	29	31
⑦	27	19	17	48
⑧	28	3	13	36
⑨	23	55	8	52
⑩	27	38	16	48
⑪	27	22	49	57
⑫	31	16		

Terus ini masing masing baris dapat di translet

Dengan melakukan ini

Contoh baris ketiga

16 38 5 51

Dalam binary

00010000 00100110 00000101 00110011

Kedua baris paling depan dihilangkan

00010000 00100110 00000101 00110011

010000 100110 000101 110011

Kemudian kita pisahkan menjadi 3 8bit

010000 100110 000101 110011

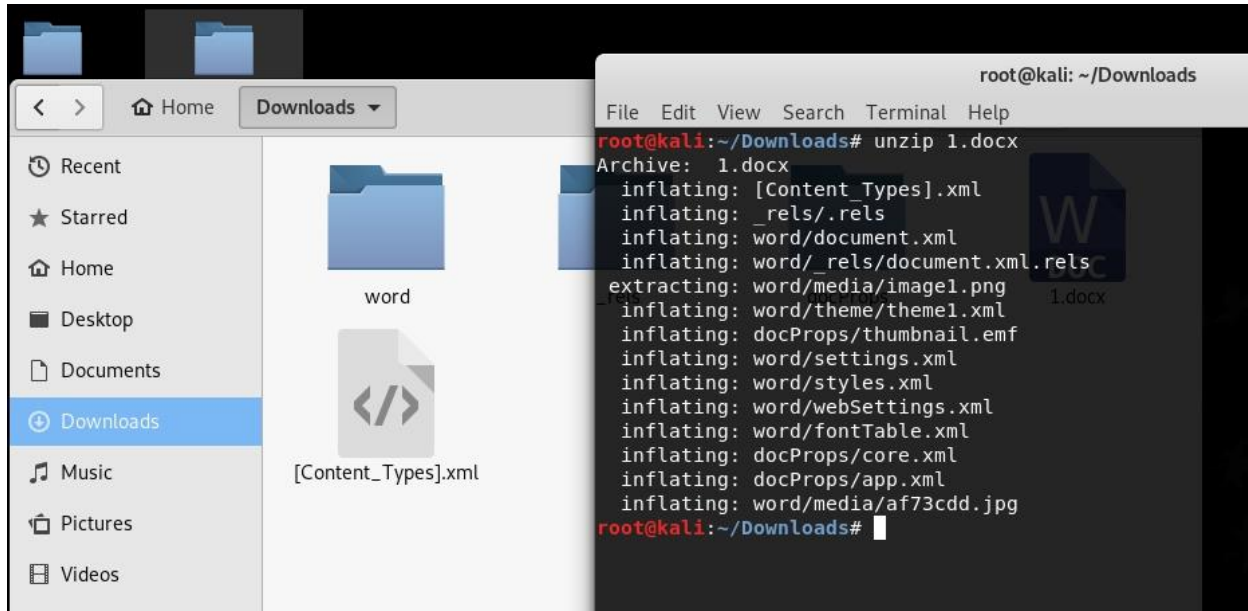
01000010 01100001 01110011 = B a s

Saya melakukan ini untuk sisanya dan mendapatkan flagnya.

**Flag:** PETIR{Bas3\_64\_bu7\_m4pp3d\_r4nd0mly}

[Welcome]

### Langkah Penyelesaian:



Saya pertama coba xxd, dan di grep tidak ketemu flag.

Kedua saya coba extract karena .docx menyimpan data" dari xml sampai media(gambar dan suara) Cuma di pasang tempel dan bisa dibongkar lagi.

<https://www.howtogeek.com/50628/easily-extract-images-text-and-embedded-files-from-an-office-2007-document/>

word>media>af73ccd.jpg

**Flag:** PETIR{welc0me\_t0\_pet1r\_qual\_2019}

[dots\_signal]

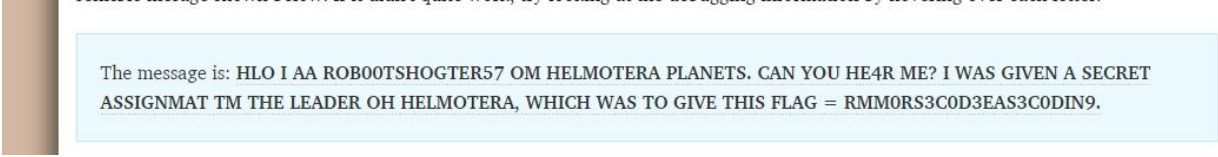
**Langkah Penyelesaian:**

Saya dengar ini ada dua suara yang berbeda.

Jadi tebakkan saya morse code

Saya menggunakan online morse code audio decoder

<https://morsecode.scphillips.com/labs/decoder/>

A screenshot of a web-based Morse code decoder. The interface has a light blue background. At the top, there is a text input field containing a series of dots and dashes representing a Morse code signal. Below the input field, the decoded message is displayed in a monospaced font. The message is: "HLO I AA ROB00TSHOGTER57 OM HELMOTERA PLANETS. CAN YOU HE4R ME? I WAS GIVEN A SECRET ASSIGNMAT TM THE LEADER OH HELMOTERA, WHICH WAS TO GIVE THIS FLAG = RMM0RS3C0D3EAS3C0DIN9." The text is enclosed in a light blue box with a thin border.

The message is: HLO I AA ROB00TSHOGTER57 OM HELMOTERA PLANETS. CAN YOU HE4R ME? I WAS GIVEN A SECRET ASSIGNMAT TM THE LEADER OH HELMOTERA, WHICH WAS TO GIVE THIS FLAG = RMM0RS3C0D3EAS3C0DIN9.

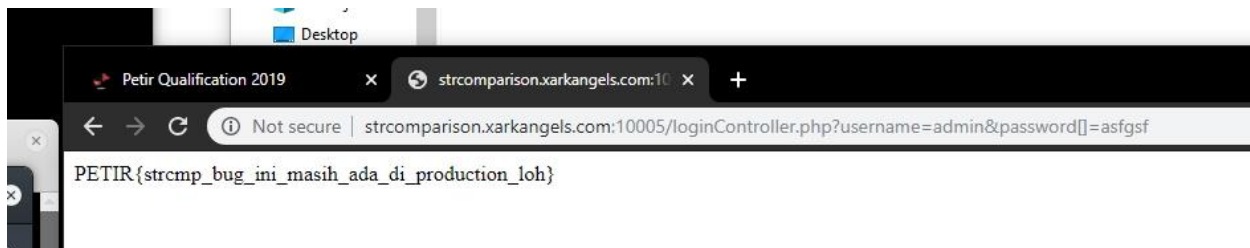
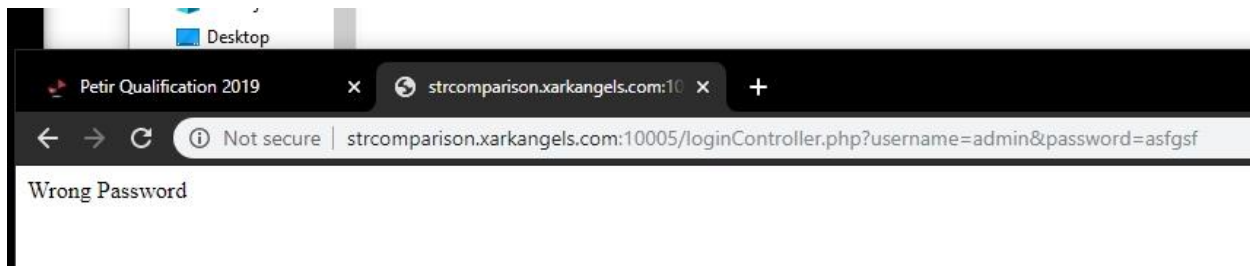
**Flag:** PETIR{RMM0RS3C0D3EAS3C0DIN9}

## [Login StrComparison]

### Langkah Penyelesaian:

Pertama coba sqli, gaada yang bisa.

Terus saya liat di web, ada yang kasih liat bahwa apabila perbandingannya dengan tipe data yang berbeda akan terjadi error dan menghasilkan True.



#### 2. BYPASS PHP STRCMP() FUNCTION

strcmp is a function created to compare strings.

int strcmp(string \$str1, string \$str2);

- Return <0 if \$str1 < \$str2
- Return 0 if \$str1 == \$str2
- Return >0 if \$str1 > \$str2

For example: Consider following PHP code handling to check for valid user's token.

```
<?php
$token = "0e37bd1f669d8bb5eae47ef80013e4d3d8287c11";
if(isset($_COOKIE['token']) && strcmp($_COOKIE['token'], $token) === 0) {
    // access to privilege area
}
else {
    // login require
}
?>
```

If we request with cookie token is an array to pass an array instead of a string to strcmp(), it will give a warning ("WARNING strcmp() expects parameter 2 to be string, array given on line number...") but the compare result return 0.

This request look like:

```
GET / HTTP/1.1
Host: example.com
Cookie: token[]= ""
.....
```

```
=> $_COOKIE['token'] = array(0 => "");
```

strcmp(array(0 => ""), "0e37bd1f669d8bb5eae47ef80013e4d3d8287c11") will return 0.

Apabila perbandingnya dengan sebuah array.

password saya ubah menjadi password[]

yang mengubah datatype si password jadi waktu di compare akan menghasilkan True.

<https://hydrasky.com/network-security/php-string-comparison-vulnerabilities/>



**Flag:** PETIR{strcmp\_bug\_ini\_masih\_ada\_di\_production\_loh}

## [Secure Password 1]

### Langkah Penyelesaian:

<http://securepassword1.xarkangels.com:10001/>

diberi link.

Saya pertama cek F12.

Disana diberi tau bahwa source berada di loginController?debug

```
<!-- securePassword source -> ?debug -->
<?php
    include_once 'flag.php';
    function randomPassword() {
        $alphabet = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890';
        $pass = array();
        $alphaLength = strlen($alphabet) - 1;
        for ($i = 0; $i < 15; $i++) {
            $n = rand(0, $alphaLength);
            $pass[] = $alphabet[$n];
        }
        return implode($pass);
    }
    if(isset($_GET['debug']))
    {
        highlight_file(__FILE__);
        die();
    }
    if(isset($_POST['password']) && isset($_POST['bibit']) && !empty($_POST['password']) && !empty($_POST['bibit'])) ←
    {
        $password = $_POST['password'];
        $bibit = $_POST['bibit'];
        srand($bibit);
        $checkpassword = randomPassword();
        if(isset($_GET['check'])) ←
        {
            echo $checkpassword;
        }
        if($password === $checkpassword)
            echo $flag;
        else
            echo "Wrong Password!";
    }
?>
```

Baris hijau menyatakan bahwa bibit dan password musti ada isinya tidak penting yang penting ada.



```
<!doctype html>
<html lang="en">
  <head>...</head>
  <body>
    <form action="loginController.php" method="POST">
      <p>Enter the Password: </p>
      <input type="text" name="password">
      <input type="submit" value="Login">
      ... <input type="hidden" name="bibit" value="1425" == $0
    </form>
    <!-- Source at loginController.php?debug -->
  </body>
</html>
```

Bibit sudah ada isi, dan password tinggal di-isi.

Sekarang satu lagi yang musti ada itu baris merah di gambar 1.

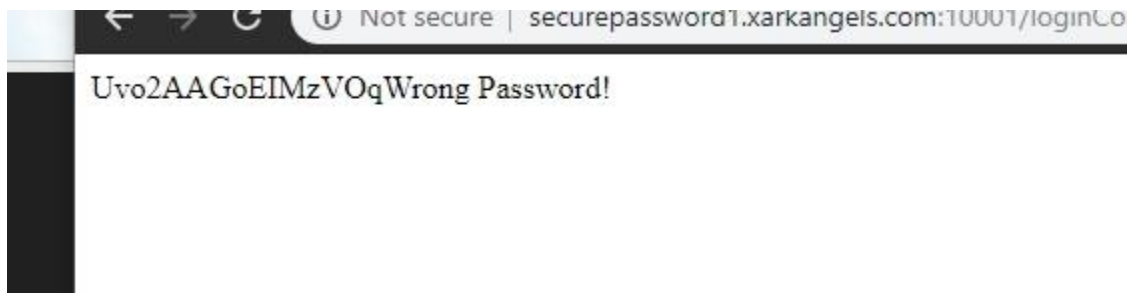
Yaitu check.



Jadi dengan ini akan dapat check juga dengan Get.

Jadi bisa melewati yang If bibit dan password ada isi

Dan If check juga ada isinya



Ini nilai checkpassword yang di echo

Enter the Password:

FLAG: PETIR{not\_secure\_anymore}

**Flag:** PETIR{not\_secure\_anymore}

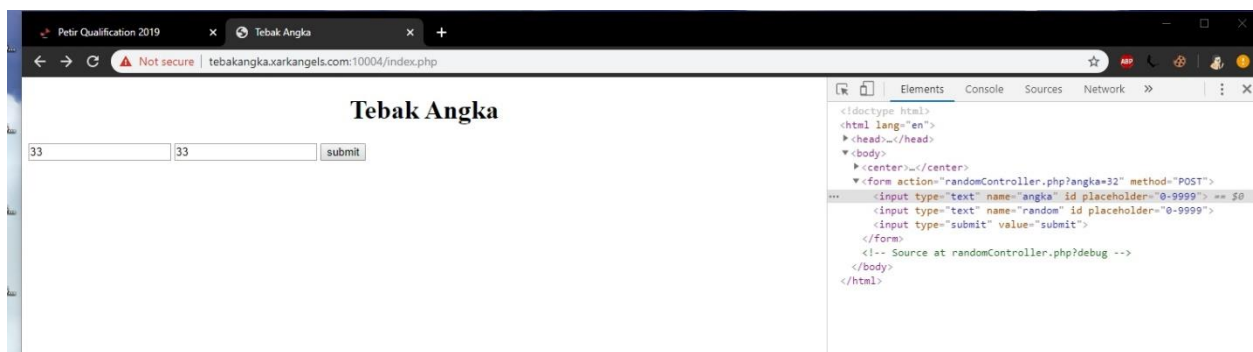
## [Tebak Angka]

### Langkah Penyelesaian:

Debug juga sama.

If Pertama Apabila dia ada isi dan merupakan angka dan ini dalam bentuk GET.

Saya memahami, ini akan meng-streamkan semua input POST kedalam variable yang sesuai dengan Key-Value yang diberikan dalam POST-streamnya



Saya ubah form ini menjadi POST dan saya kirim angka dalam bentuk GET agar If pertama dapat dilewati, musti terisi dan merupakan terisi angka.

Saya ubah nama form menjadi angka dan menambahkan satu lagi bernama random yang akan langsung di streamkan oleh phpnya kedalam variable yang bernama 'KEY' yang saya berikan yaitu `name` nya.

Berarti sekarang input saya musti sama. Seperti diatas



**Flag:** PETIR{overwrite\_value\_1337}

Sekian Write-Up dari saya, Terima Kasih Banyak sudah membaca Write-up Saya.