

## LKS SMK 2022 XXX Cyber Security

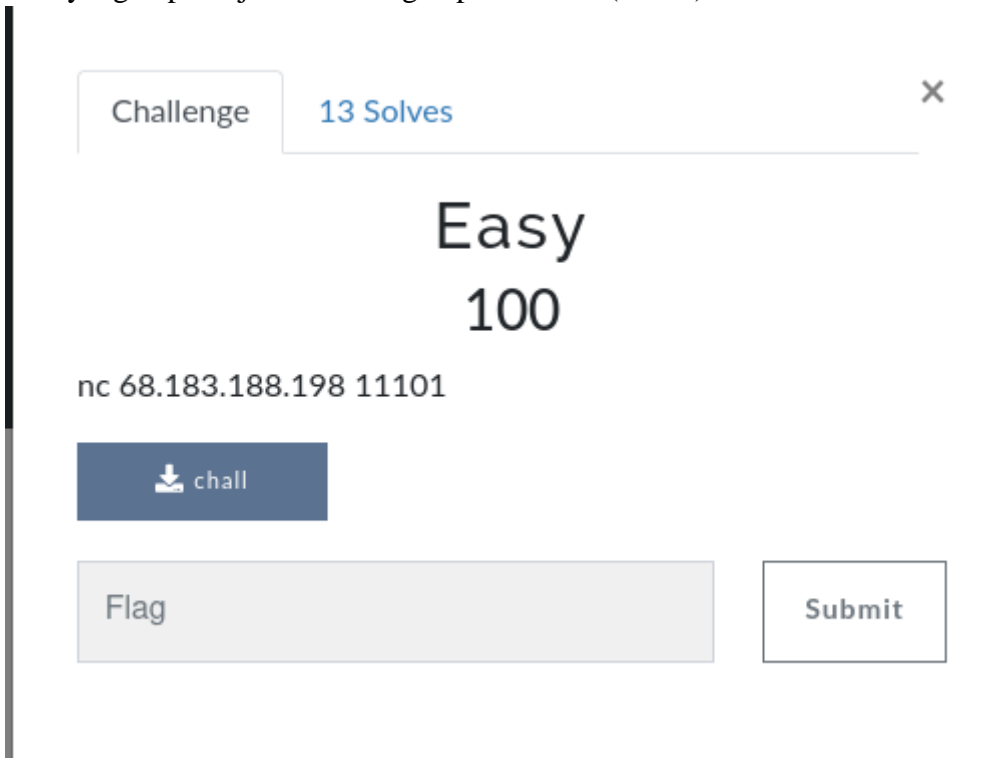
### Writeup Day 2 (CTF Jeopardy Competition)

Di Day 2 ini, kami diberikan tantangan berupa soal-soal CTF (Capture The Flag). Berikut adalah beberapa challenge yang berhasil kami selesaikan.

#### A. Binary Exploitation

##### 1. Easy

Di challenge ini, terdapat sebuah file executable dan juga deskripsi berupa IP dan Port yang dapat dijalankan dengan perintah nc (netcat).



Hal pertama yang kita lakukan adalah menjalankan netcat IP di terminal atau menjalankan file chall yang sudah di download.

```
(kali㉿kali)-[~/LKS-Nasional/Day 2/easy]  
$ nc 68.183.188.198 11101  
You have old laptop, then you want to sell it to buy flag  
How much do you want to sell your laptop ?
```

Ketika dijalankan, akan muncul pertanyaan seperti di atas. Dari sini, kita bisa

mendapatkan flag nya dengan cara memasukkan sejumlah angka negatif, yang dimana saya memasukkan angka -1000 untuk mendapatkan flag-nya.

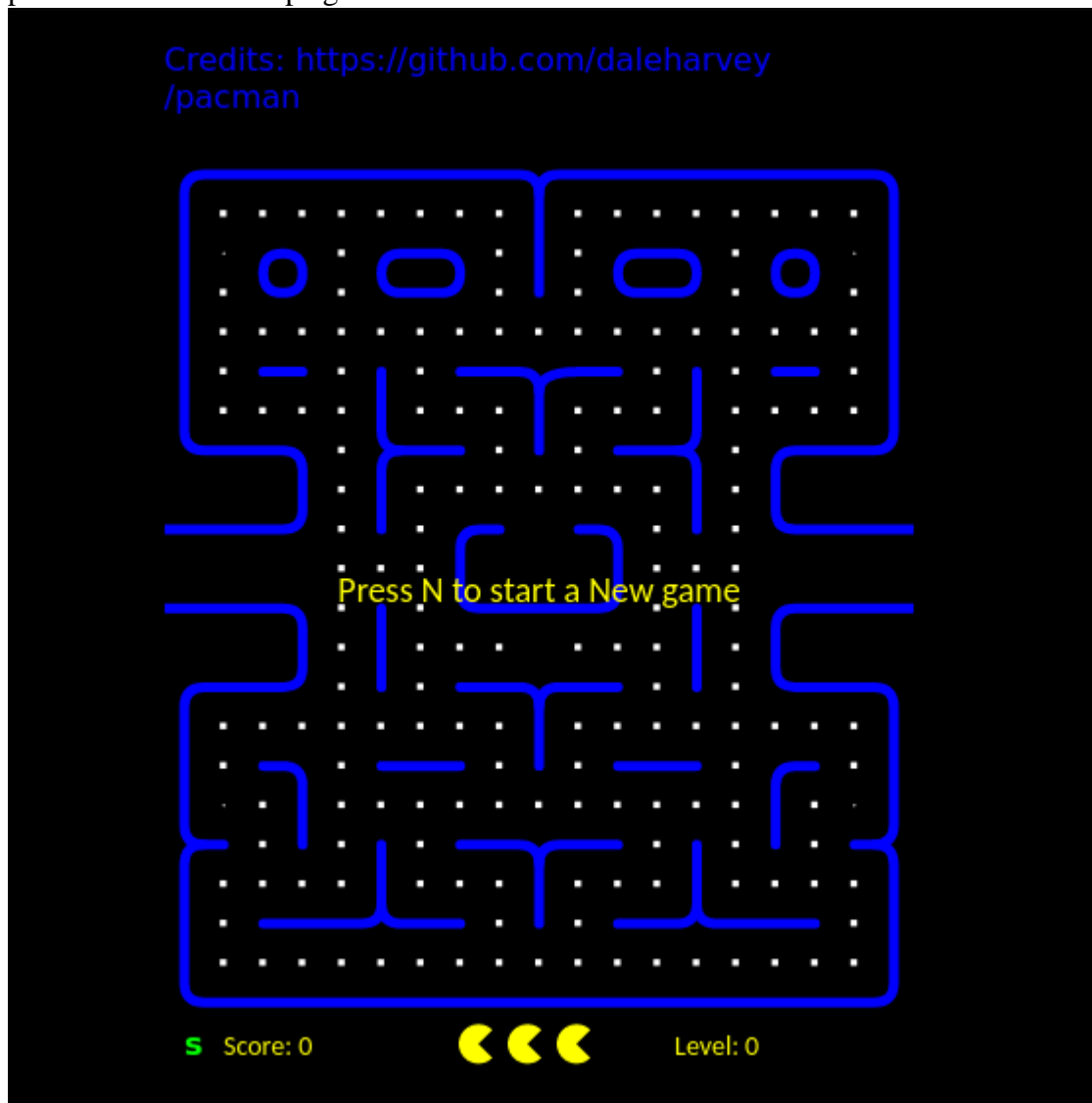
```
(kali@kali)-[~/LKS-Nasional/Day 2/easy]
$ nc 68.183.188.198 11101
You have old laptop, then you want to sell it to buy flag
How much do you want to sell your laptop ? -1000
OK you have -1000$
Ok you will get bonus : LKSN{basic_integer_overflow}
```

Flag : LKSN{basic\_integer\_overflow}

## B. Web Exploitation

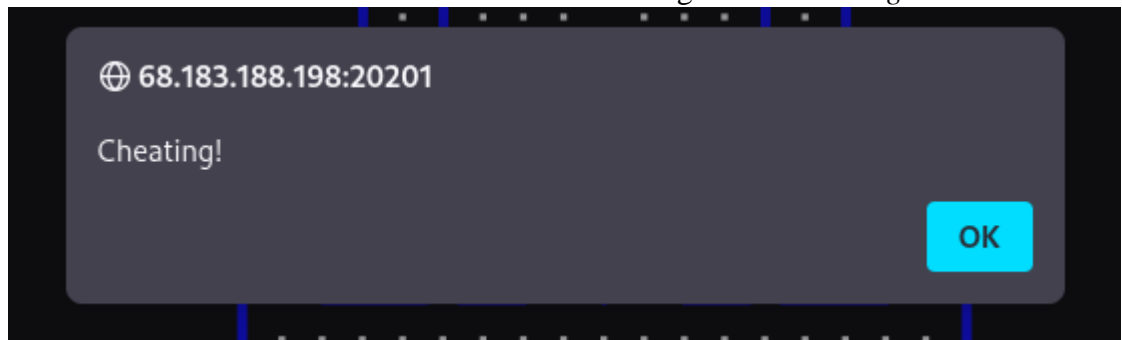
### 1. Arcade Sidescreen

Di challenge ini, terdapat sebuah link yang jika dibuka, akan mengarahkan kita pada sebuah web berupa game *PacMan*.

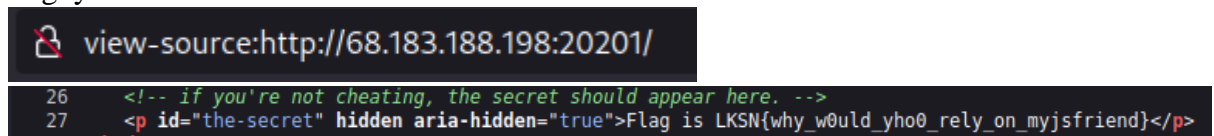


Untuk mendapatkan flag-nya, kita perlu mengakses Page Source dari web ini. Biasanya untuk mengakses web page source itu bisa dilakukan dengan klik kanan

lalu klik View Page Source. Akan tetapi, jika kita melakukan itu, web akan memunculkan notifikasi bahwa kita melakukan kecurangan atau *cheating*.



Untuk mengakali ini, kita akan melakukan ketik manual pada search browser yang kita gunakan. Ketik `view-source:http://68.183.188.198:20201/`. Dan akan muncul flagnya.

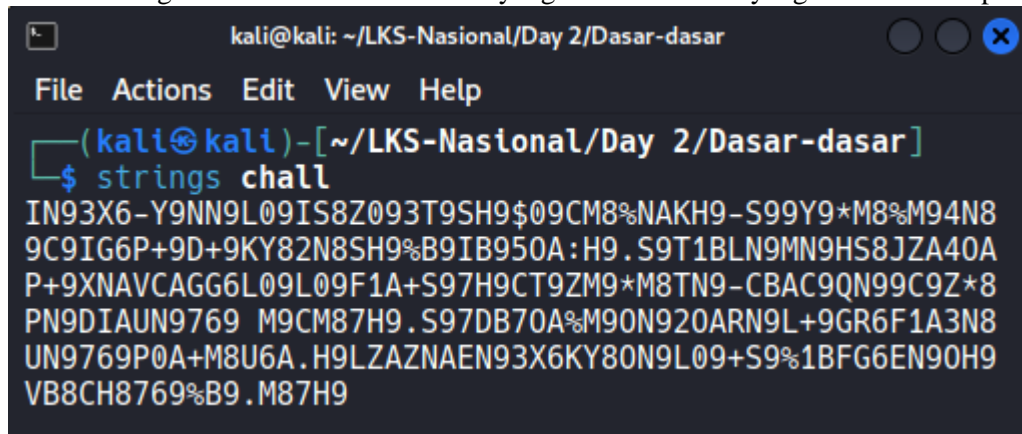


Flag : LKSN {why\_would\_yho0\_rely\_on\_myjsfriend}

### C. Cryptography

#### 1. Dasar-dasar

Pada challenge ini disediakan sebuah file yang berisi kata-kata yang sudah di enkripsi



Awalnya kami tidak mengetahui jenis enkripsi apa kata-kata di atas, akhirnya kami memakai online tool yang bernama *Cipher Identifier* pada link berikut

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

### Search for a tool

★ SEARCH A TOOL ON dCODE BY KEYWORDS:  
e.g. type 'boolean'

★ BROWSE THE [FULL dCODE TOOLS' LIST](#)

### Results

dCode's analyzer suggests to investigate:

↑↓	↑↓
<a href="#">Base45 Encoding</a>	■■■■■■■■■
<a href="#">Substitution Cipher</a>	■
<a href="#">Shift Cipher</a>	□
<a href="#">Homophonic Cipher</a>	□
<a href="#">Hexadecimal (Base 16)</a>	□
<a href="#">ASCII Code</a>	□
<a href="#">ASCII85 Encoding</a>	□

#7

### Cryptography > Cipher Identifier

#### ENCRYPTED MESSAGE IDENTIFIER

★ CIPHERTEXT TO RECOGNIZE (?)

IN93X6-Y9NN9L09IS8Z093T9SH9\$09CM8%NAKH9-S99Y9\*M8%M94N89C9IG6P+9D+9KY82N8SH9%89IB950A:H9.S9T18LN9MN9H58JZA40AP+9XNAVCA6G6L09L09F1A+S97H9CT9ZM9\*M8TN9-CBAC9QN99C9Z\*8PN9DIAUN9769M9CM87H9.S97DB70A%M90N920AR9L+9GR6F1A3N8UN9769P0A+M8U6A.H9LZAZNAEN93X6KY80N9L09+S9%18FG6EN90H9VB8CH8769%B9.M87H9

★ CLUES/KEYWORDS (IF ANY)

[▶ ANALYZE](#)

See also: [Frequency Analysis](#) – [Index of Coincidence](#)

#### SYMBOLS IDENTIFIER

▶ Go to: [Symbols Cipher List](#)

#### Answers to Questions (FAQ)

##### How to decrypt a cipher text?

To decrypt / decipher an encoded message, it is necessary to know the

Dengan tool online tersebut, kita dapat mengetahui jenis enkripsi apa yang digunakan, setelah mengetahui bahwa enkripsi yang digunakan adalah Base45 maka kita langsung saja mendecrypt kata-kata nya.

### Search for a tool

★ SEARCH A TOOL ON dCODE BY KEYWORDS:  
e.g. type 'sudoku'

★ BROWSE THE [FULL dCODE TOOLS' LIST](#)

### Results

IN93X6-Y9NN9...7H9

KN5FMTKSGDFGTLJJGVC2SKJBLFM4CMK5CWIV2XNRNFEVCUJJIFI2SWJZLGWMMKQRDEUTSVNR5FQU2VGF6FVLEJ5LVK2CMKYHYHVKVIVFFKURQKZHF3C2J5LGYSSYK5KTSTKWNN4EOVCVKZHF03CNPFJXUVSHKJ5FEVKTGFLEWV2UKJJFAVBQHFIFCPJ5

### BASE45 ENCODING

Informatics > Character Encoding > Base45 Encoding

#### BASE45 DECODER

★ BASE45 CIPHERTEXT (?)

IN93X6-Y9NN9L09IS8Z093T9SH9\$09CM8%NAKH9-S99Y9\*M8%M94N89C9IG6P+9D+9KY82N8SH9%89IB950A:H9.S9T18LN9MN9H58JZA40AP+9XNAVCA6G6L09L09F1A+S97H9CT9ZM9\*M8TN9-CBAC9QN99C9Z\*8PN9DIAUN9769M9CM87H9.S97DB70A%M90N920AR9L+9GR6F1A3N8UN9769P0A+M8U6A.H9LZAZNAEN93X6KY80N9L09+S9%18FG6EN90H9VB8CH8769%B9.M87H9

★ RESULTS FORMAT ☒ ASCII (PRINTABLE) CHARACTERS

☐ HEXADECIMAL 00-7F-FF

☐ DECIMAL 0-127-255

☐ OCTAL 000-177-377

☐ BINARY 00000000-11111111

☐ INTEGER NUMBER

Terlihat terdapat hasil yang masih berupa kata-kata enkripsi, maka kita identifikasi lagi jenis enkripsi tersebut dengan tool *Cipher Identifier* sebelumnya.

### Search for a tool

★ SEARCH A TOOL ON dCODE BY KEYWORDS:  
e.g. type 'boolean'

★ BROWSE THE [FULL dCODE TOOLS' LIST](#)

### Results

dCode's analyzer suggests to investigate:

↑↓	↑↓
<a href="#">Base32</a>	■■■■■■■■■
<a href="#">Base64 Coding</a>	■
<a href="#">Base62 Encoding</a>	■
<a href="#">Substitution Cipher</a>	■
<a href="#">Shift Cipher</a>	□
<a href="#">Base 58</a>	□
<a href="#">Homophonic Cipher</a>	□
<a href="#">Affine Cipher</a>	□
<a href="#">Mono-alphabetic Substitution</a>	□
<a href="#">Cipher Disk/Wheel</a>	□

### Cryptography > Cipher Identifier

#### ENCRYPTED MESSAGE IDENTIFIER

★ CIPHERTEXT TO RECOGNIZE (?)

KN5FMTKSGDFGTLJJGVC2SKJBLFM4CMK5CWIV2XNRNFEVCUJJIFI2SWJZLGWMMKQRDEUTSVNR5FQU2VGF6FVLEJ5LVK2CMKYHYHVKVIVFFKURQKZHF3C2J5LGYSSYK5KTSTKWNN4EOVCVKZHF03CNPFJXUVSHKJ5FEVKTGFLEWV2UKJJFAVBQHFIFCPJ5

★ CLUES/KEYWORDS (IF ANY)

[▶ ANALYZE](#)

See also: [Frequency Analysis](#) – [Index of Coincidence](#)

#### SYMBOLS IDENTIFIER

▶ Go to: [Symbols Cipher List](#)

#### Answers to Questions (FAQ)

##### How to decrypt a cipher text?

To decrypt / decipher an encoded message, it is necessary to know the encryption used (or the encoding method, or the implemented cryptographic principle). Without knowing the technique chosen by the sender of the message, it is impossible to decrypt it (or decode it). Knowing

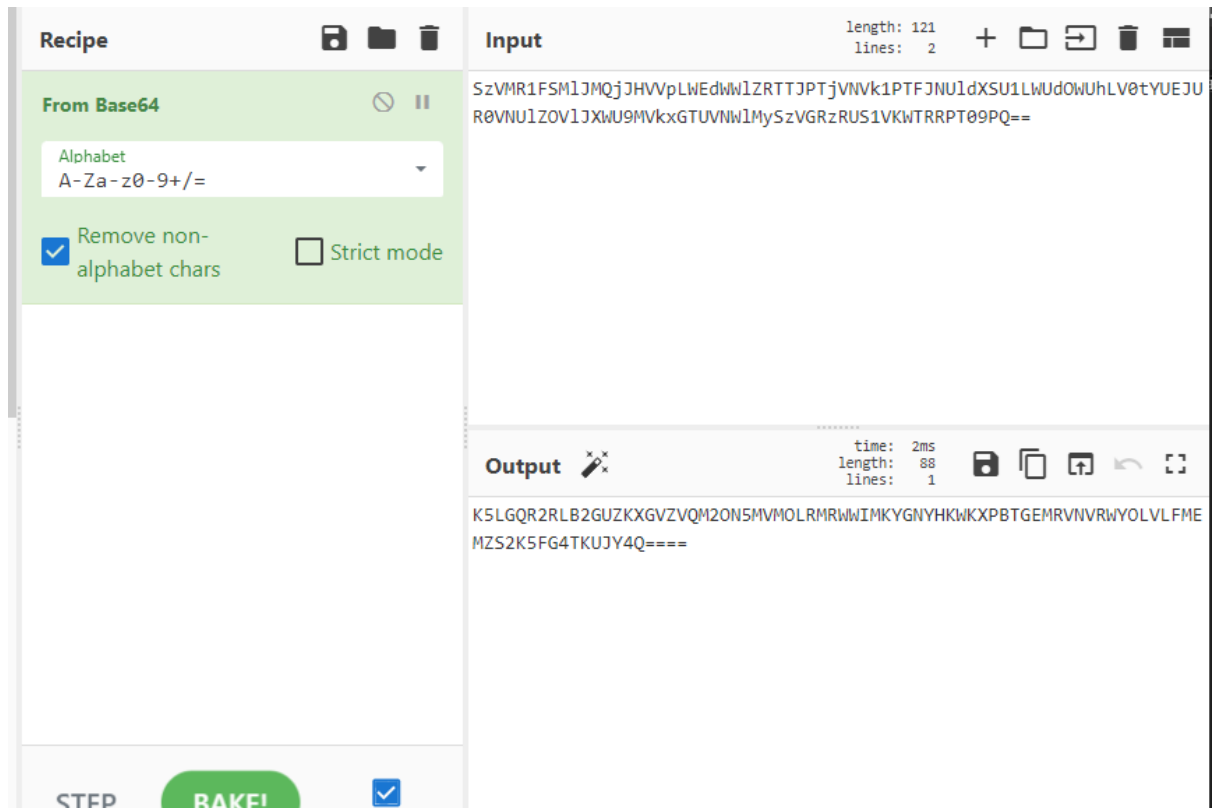
Dan didapatkan hasil bahwa kata-kata tersebut merupakan Base32, maka kita decrypt lagi kata-katanya.

The screenshot shows the dCode website interface. On the left, the 'Search for a tool' section has a search bar with 'e.g. type 'random'' and a 'BROWSE THE FULL dCODE TOOLS' LIST' link. Below, the 'Results' section displays the Base32 string 'SzVMR1FSM1JMQjJHVvpLWEdWw1ZRRTTjPTjVNVk1PTFJNU1dXSU1LWUdOWUhLV0tYUEJUR0VNU1ZOV1JXWU9MVkxGTUVNW1MySzVGRzRUS1VKWTRRPT09PQ=='. It identifies the tool as 'Base32 - dCode' and lists the tag '(s) : Character Encoding'. On the right, the 'BASE32 DECODER' section includes a warning not to confuse it with mathematical base 32 conversion, a 'Go to: Base N Convert' link, and a 'BASE 32 CIPHERTEXT (?)' section containing the same Base32 string. Below this, the 'RESULTS FORMAT' section shows radio buttons for 'ASCII (PRINTABLE) CHARACTERS' (selected), 'HEXADECIMAL 00-7F-FF', 'DECIMAL 0-127-255', 'OCTAL 000-177-377', and 'BINARY 00000000-11111111'.

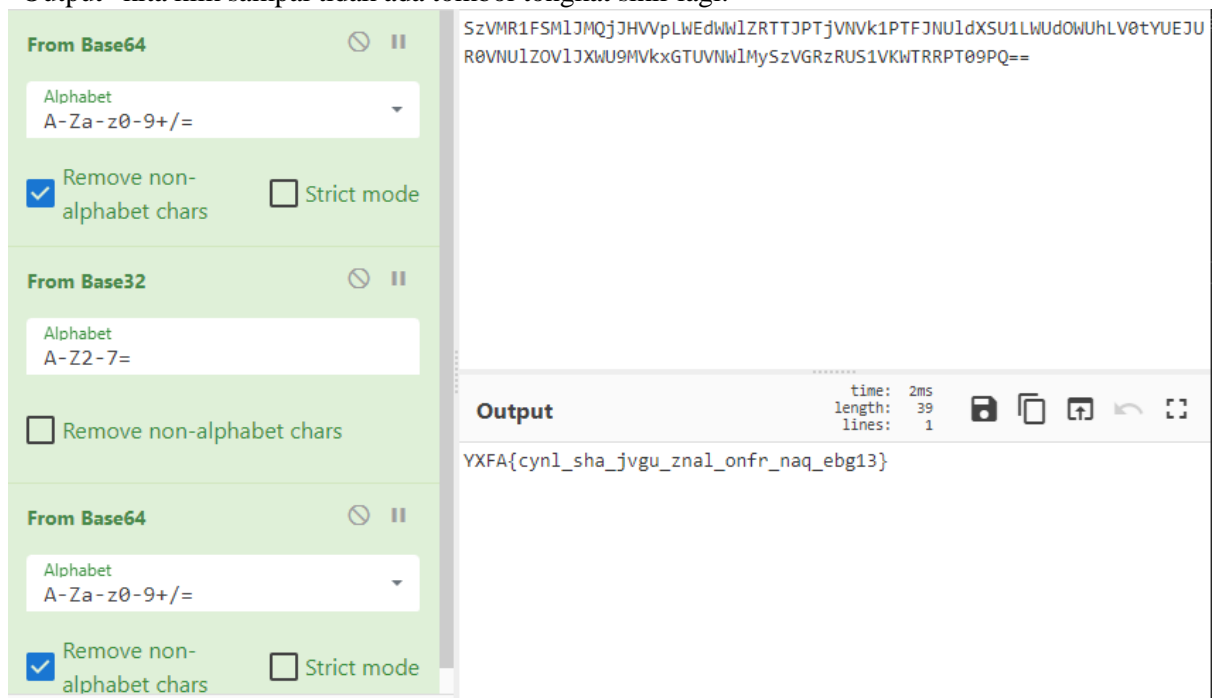
Dan lagi!!!!, kita mendapatkan hasil yang masih terenkripsi, maka kita ulangi lagi langkah-langkah sebelumnya sampai kita mendapatkan hasil yang benar-benar sudah terdecrypt dan bisa dibaca oleh manusia.

The screenshot shows the dCode website interface. On the left, the 'Search for a tool' section has a search bar with 'e.g. type 'boolean'' and a 'BROWSE THE FULL dCODE TOOLS' LIST' link. Below, the 'Results' section displays a list of tools suggested by dCode's analyzer: 'Base64 Coding', 'Base62 Encoding', 'Trithemius Cipher', 'Vigenere Cipher', 'Substitution Cipher', 'Autoclave Cipher', 'Beaufort Cipher', and 'Rozier Cipher'. On the right, the 'CIPHER IDENTIFIER' section includes a 'Cryptography · Cipher Identifier' header, an 'ENCRYPTED MESSAGE IDENTIFIER' section with a 'CIPHERTEXT TO RECOGNIZE (?)' section containing the same Base64 string, and a 'CLUES/KEYWORDS (IF ANY)' section with an 'ANALYZE' button. Below this, there are links to 'Frequency Analysis - Index of Coincidence', 'SYMBOLS IDENTIFIER', and 'Go to: Symbols Cipher List'. At the bottom, the 'Answers to Questions (FAQ)' section includes a link to 'How to decrypt a cipher text?' and a note that to decrypt a message, it is necessary to know the key.

Setelah mengetahui bahwa enkripsi tersebut Base64 maka kita akan menggunakan online tools yang lain yaitu *Cyber Chef*.

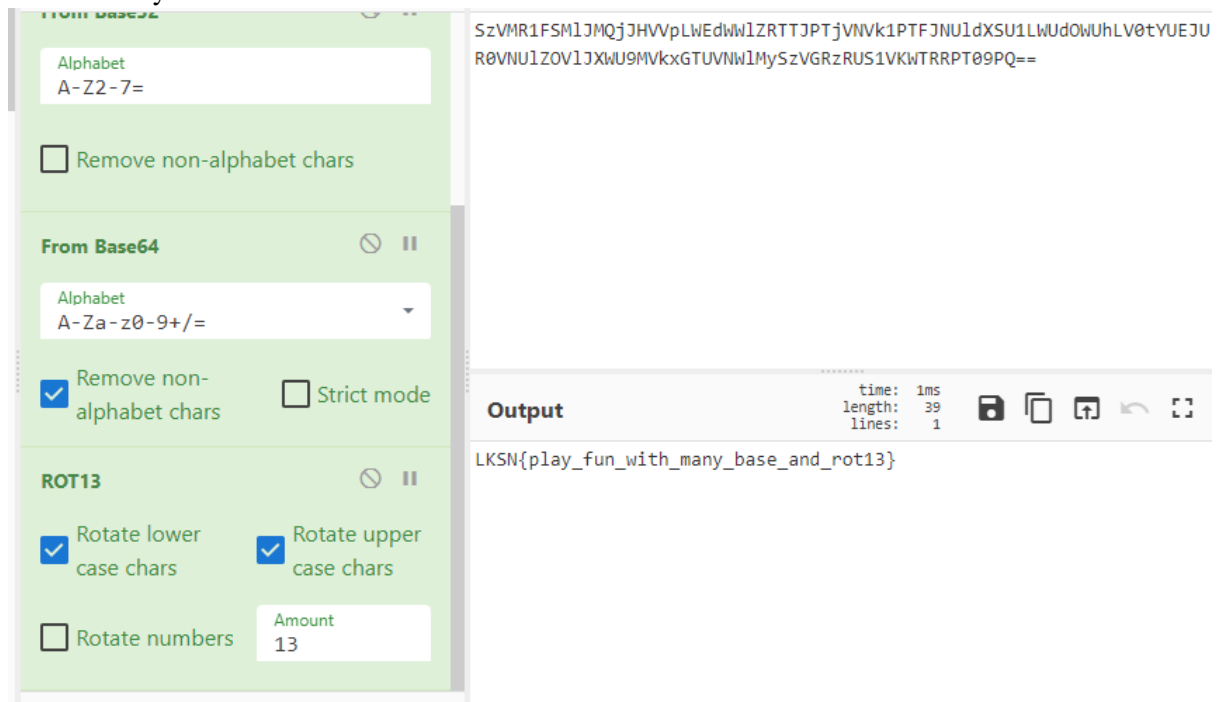


Ternyata hasilnya masih berupa kata-kata terenkripsi, maka kita lakukan decrypt lagi dengan cara mengeklik tombol seperti tongkat sihir yang berada di samping kata “Output” kita klik sampai tidak ada tombol tongkat sihir lagi.



Nah, sampai di sini, kami mengalami kesusahan karena ketika kami menggunakan tool *Cipher Identifier* tidak menemukan jenis enkripsi yang cocok sama sekali, kemudian kami menyadari bahwa kata akhir yang berupa angka 13 tersebut bisa jadi jenis enkripsi

ROT13, maka tanpa pikir panjang lagi, kami mendekripsi nya menggunakan ROT13 dan inilah hasilnya.



Flag: LKSN{play\_fun\_with\_many\_base\_and\_rot13}

#### D. Reverse Engineering

##### 1. Bahasa Ular

Pada challenge ini, kami diberikan challenge dengan deskripsi yang disertai file berupa file bahasaular.pyc dan flag.txt.malware yang terenkripsi.

Challenge

8 Solves



# Bahasa Ular

## 151

Selamat datang di challenge reversing pertama! Yuk kita pahami bersama bahasa "ular" ini ~ Bahasa ini sangat umum digunakan untuk melakukan **scripting** lho! Dapatkah kamu me-recover hasil orisinil dari flag.txt yang sudah terenkripsi? :(



bahasaular.pyc



flag.txt.malw...

Flag

Submit

Kami pikir, file .pyc itu sama saja dengan format file python biasa yaitu .py, akan tetapi saat kami berusaha untuk menjalankannya, terjadi error. Akhirnya kami mencari cara bagaimana mengeksekusi file .pyc. Dan ternyata untuk menjalankan



file .pyc, kita harus mendecompile atau mengubah file .pyc menjadi file .py. Kita menggunakan tool *pycdc* yang didapat dari internet untuk melakukannya.

```
(kali@kali)-[~/Downloads/LKS-Nasional/pycdc]
$ ./pycdc /home/kali/Downloads/LKS-Nasional/reveng/bahasaular.pyc
# Source Generated with Decompyle++
# File: basasaular.pyc (Python 3.8)

Unsupported opcode: BEGIN_FINALLY
import this
input_peserta = input('Yuk belajar bahasa ular yang sudah di-"goreng"!\\nMasukkan nama file kamu dan kami mengubahnya menjadi format yang k
eren! >>')
f_handler = open(input_peserta, 'rb').read()
f_transform = []
for karakter in f_handler:
    f_transform.append(karakter ^ 2 ^ 3 ^ 7 ^ 9 ^ 11 ^ 13)
f_final = ''
for karakterlagi in range(len(f_transform)):
    f_final += chr(f_transform[karakterlagi] ^ ord(this.s[karakterlagi % len(f_transform)]))
# WARNING: Decompyle incomplete
```

Command bisa berbeda-beda tergantung dimana kita menaruh file basasaular.pyc kita.

Setelah kita decompile menggunakan pycdc, kita copy hasil decompile nya dan kita masukkan ke text editor.

```
1 # Source Generated with Decompyle++
2 # File: basasaular.pyc (Python 3.8)
3
4 Unsupported opcode: BEGIN_FINALLY
5 import this
6 input_peserta = input('Yuk belajar bahasa ular yang sudah di-"goreng"!\\nMasukkan nama file kamu dan kami mengubahnya menjadi format yang keren! >>')
7 f_handler = open(input_peserta, 'rb').read()
8 f_transform = []
9 for karakter in f_handler:
10     f_transform.append(karakter ^ 2 ^ 3 ^ 7 ^ 9 ^ 11 ^ 13)
11 f_final = ''
12 for karakterlagi in range(len(f_transform)):
13     f_final += chr(f_transform[karakterlagi] ^ ord(this.s[karakterlagi % len(f_transform)]))
14 # WARNING: Decompyle incomplete
```

Setelah kami baca code di atas, kami menyadari bahwa variable `f_final` tidak pernah di print, maka dari itu kami tambahkan `print(f_final)` seperti pada gambar di bawah ini.

```
1 # Source Generated with Decompyle++
2 # File: basasaular.pyc (Python 3.8)
3
4 import this
5 input_peserta = input('Yuk belajar bahasa ular yang sudah di-"goreng"!\\nMasukkan nama file kamu dan kami mengubahnya menjadi format yang keren! >>')
6 f_handler = open(input_peserta, 'rb').read()
7 f_transform = []
8 for karakter in f_handler:
9     f_transform.append(karakter ^ 2 ^ 3 ^ 7 ^ 9 ^ 11 ^ 13)
10 f_final = ''
11 for karakterlagi in range(len(f_transform)):
12     f_final += chr(f_transform[karakterlagi] ^ ord(this.s[karakterlagi % len(f_transform)]))
13     print(f_final)
14 # WARNING: Decompyle incomplete
```

Setelah itu kami coba jalankan script nya dan kami masukkan flag.txt.malware dan inilah hasilnya.

```

(kali㉿kali)-[~/Downloads/LKS-Nasional/reveng]
└─$ python bahasaular.py
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
Yuk belajar bahasa ular yang sudah di-"goreng"!
Masukkan nama file kamu dan kami mengubahnya menjadi format yang keren! >>flag.txt.malware
L
LK
LKS
LKSN
LKSN{
LKSN{w
LKSN{w4
LKSN{w4h
LKSN{w4h_
LKSN{w4h_k
LKSN{w4h_ka
LKSN{w4h_kam
LKSN{w4h_kamU
LKSN{w4h_kamU_
LKSN{w4h_kamU_b

```

```

LKSN{w4h_kamU_b!sa_84hasa_uLar_yang_diG0reng_a.k.a_PYC}

```

Flag: LKSN(w4h\_kamU\_b!sa\_84hasa\_uLar\_yang\_diG0reng\_a.k.a\_PYC)

Sekian writeup CTF dari kami, kurang lebihnya mohon maaf. Sekian dan terimakasih.