《漏洞利用及渗透测试基础》第四次实验报告

varify检测中栈溢出的问题分析

1811463 赵梓杰 信息安全

- 代码样例

```
#include<stdio.h>
#include<iostream>
#define PASSWORD "1234567"
int verify_password(char* password)
{
    int authenticated;
    char buffer[12];
    authenticated = strcmp(password, PASSWORD);
    strcpy(buffer, password);
    return authenticated;
}
void main()
{
    int valid_flag = 0;
    char password[1024];
    while (1){
        printf("please input password: ");
        scanf("%s", password);
        valid_flag = verify_password(password);
        if (valid_flag) {
            printf("incorrect password!\n\n");
        }
        else {
            printf("Congratulation!You have passed the verification!");
            break;
        }
    }
    system("pause");
}
```

首先我们可以对代码进行简单的分析，即可发现这段代码是一段密码验证机制，主要的判断机制在于函数verify_password中，authenticated是一个int类型的变量，而我们同样可以发现，authenticated这个变量是根据strcmp函数的返回取值，随之传递给main函数，如果authenticated为0，则表示验证成功。

- IDA伪代码的分析

```
int __cdecl main()
{
  char v1; // [sp+Ch] [bp-444h]@1
  char Source; // [sp+4Ch] [bp-404h]@2
  int v3; // [sp+44Ch] [bp-4h]@1

  memset(&v1, -858993460, 0x444u);
  v3 = 0;
  while ( 1 )
```

```
  {
    printf("please input password: ");
    scanf("%s", &Source);
    v3 = verify_password(&Source);
    if ( !v3 )
      break;
    printf("incorrect password!\n\n");
  }
  printf("Congratulation!You have passed the verification!");
  return system("pause");
}


int __cdecl verify_password(const char *Source)
{
  char v2; // [sp+Ch] [bp-50h]@1
  char Dest; // [sp+4Ch] [bp-10h]@1
  int v4; // [sp+58h] [bp-4h]@1

  memset(&v2, -858993460, 0x50u);
  v4 = strcmp(Source, "1234567");
  strcpy(&Dest, Source);
  return v4;
}
```

IDA得到的伪代码如上，我们可以对伪代码进行简单的分析，不难发现我们的authenticated（v4）这个int类型的数值，我们的目的是让前面的char类型（其实应该是char数组，伪代码中只赋值初地址）完成越界操作，将四位字节写入变量v4中。

被调用的子函数中写入数据的长度，大于栈帧的基址到esp之间预留的保存局部变量的空间时，就会发生栈的溢出。要写入数据的填充方向是从低地址向高地址增长，多余的数据就会越过栈帧的基址，覆盖基址以上的地址空间。

所以如果我们输入的passwd超过了11个字符，字符串句末有结尾符号，那么我们越界的字符的ASCII码就会覆盖掉v4也就是authenticated的数值，如果我们的溢出能把authenticated改成0那么就完成了crack

- crack流程
  - 首先我们需要输入一个12个字符的字符串，因为算上结尾符一共13位，且结尾符为0，所以会覆盖掉v4的高字节使之为0
  - 输入的字符串应该大于1234567，因为strcmp执行后可以保证变量v5的高位为1，这样我们后续覆盖一个0也就可以将v4置零

- 心得体会

  这次实验简单了解了一下栈溢出漏洞，能掌握简单的栈溢出漏洞的payload构造。