COMP 8505 ASSIGNMENT 3 TESTING DOC

Yiaoping Shu + Anthony Figueroa

A00930347, A00927466 October 20 2018

Introduction

The purpose of this assignment is to become familiar with packet-sniffing backdoors implemented in Linux. Backdoor is a technique where the system security is bypassed undetectably to access the computer or information on the computer itself. This backdoor that we've created functions so that when running on the server, if a user were to look at the process table and try to determine if there were any malicious software running, we can hide the process table by naming it to something else that runs normally on Linux to make it look non-malicious. The backdoor commands in from the client and processes them to be ran, sending the results back to the client, encrypted both ways. We also spoof the IP so that the user does not know where the IP is coming from.

Usage

Before running the program, ensure you are the root user and have the zip folder stored somewhere on your computer. Navigate to the zip/tar file and extract it to a location of your choice. Go to the extracted files location.

Ensure each file can be ran by performing a chmod on the files for reading the file.
Inside the folder are 2 files, a client and a server file. Choose one computer to be the client side and another computer to be the server side.

To run the *client* program, type in the following command with the following switches.

#python3 client.py -d [destination Ip] -t [process title] -s [spoof Ip]

Destination Ip: The IP target that you want to connect to, which is where the server program is located

Process title: The name of the process that you want appeared on the server to hide it from the victim

Spoof Ip: Typing in an IP will show that the datagram being sent is from this spoofed IP.

Ensure the extension is typed for each file image and name.

To run the server, type in the following command

#python3 server.py

After running both the server and the client, type in the command that you want. The client will ask you to enter in a command which will be sent to the server, where it processes the command and sends the results back. For example, typing the command "ps aux" would result in the server sending the current processes being ran on the server computer.

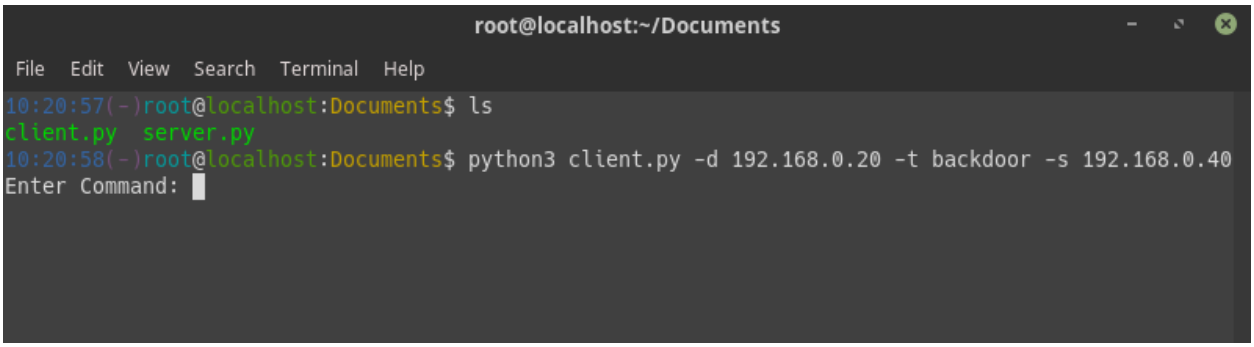
Testing

Test #	Description	Expected Result	Result (Pass/Fail)
1	Navigate to source folder. Execute the client program with correct switches	User can successfully execute the client program	Pass
2	Execute server-side program	User can successfully execute the server-side program	Pass
3	Enter any command to send in client side	A result is displayed to client user	Pass
4	Enter a proper command such as “pwd” on client side	The correct current directory the server is residing in is shown to client	Pass
5	Enter the command “ls” on client side	Client is displayed correct files that server program resides in	Pass
6	Add another file in the same directory as server file and type “ls” on client side	Client user is displayed the new file in terminal along with old files	Pass
7	Test for process title. Type in “ps ax | grep [process title]”	Client is displayed the correct process title that resides on server side from entering arguments	Pass
8	Send a command from client to server. Check Wireshark to ensure data is encrypted	Wireshark displays something that user cannot read without decrypting	Pass
9	Client side: Create a file by going “touch test file” on the server. Type in “ls” to check for new file	New file is created and shown to the client.	Pass
10	Client side: Manipulate the ip tables by dropping all INPUT packets	All input packets are dropped when IP tables are checked on server side	Pass
11	IP spoofing is working correctly. Type in an IP to spoof in the cmd terminal	IP is spoofed and displayed in Wireshark the spoofed IP	Pass

Screenshots

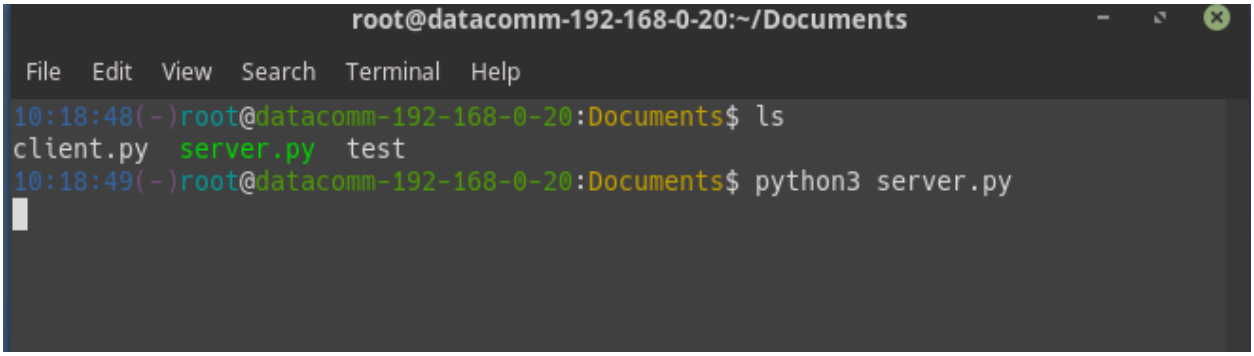
Test #1

As you can see below, the client is running properly and asking the user for the command.



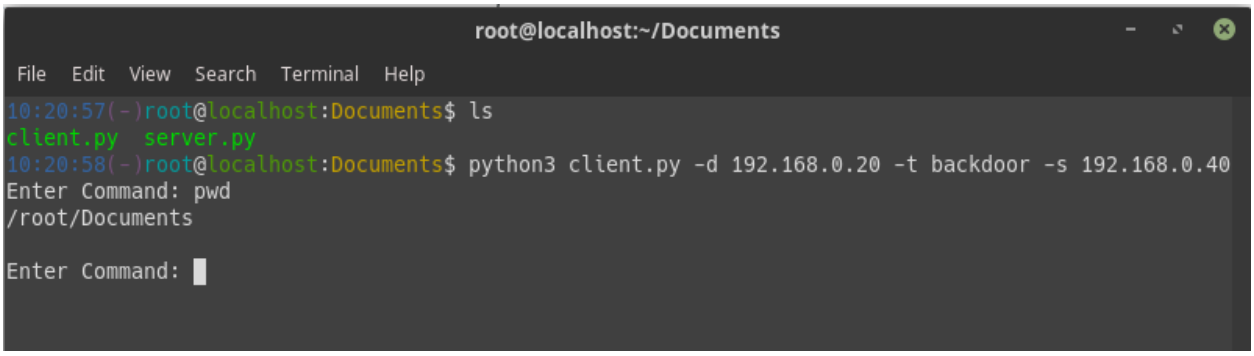
Test #2

Below shows the server being ran properly, waiting for the command from the client



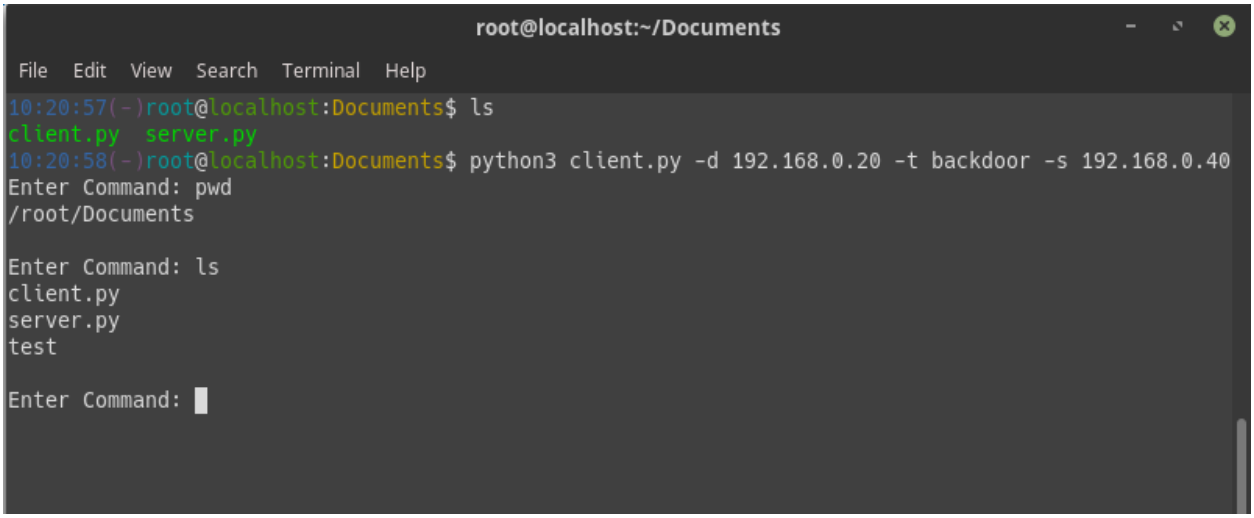
Test #3

We type in the command “pwd” which gets sent to the server, then returns the path that the server file is currently in.



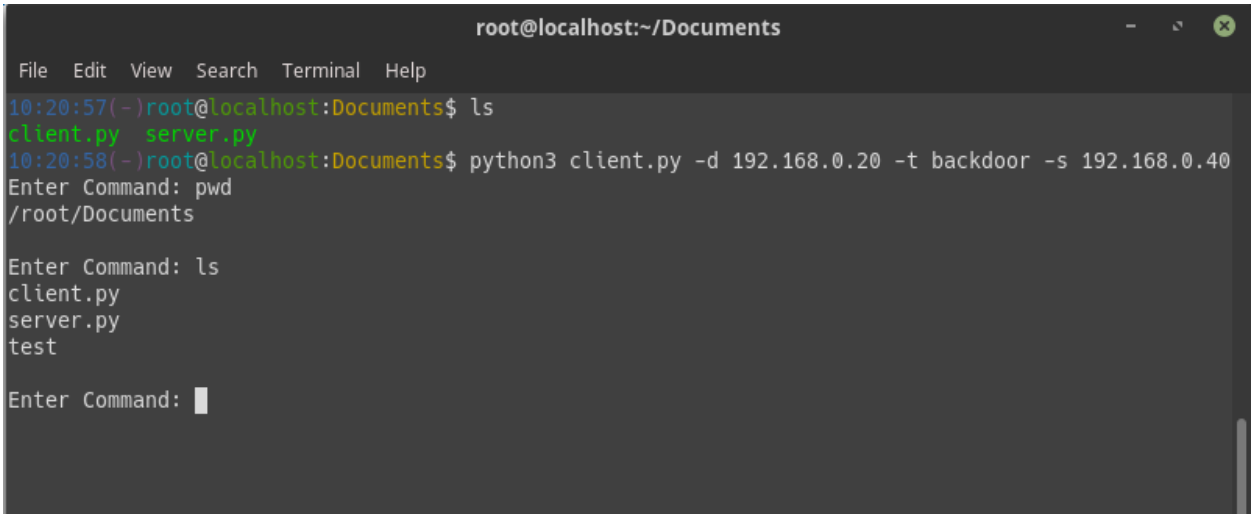
Test #4

We type in the command “ls” to show the current files that are in the same folder as the server.



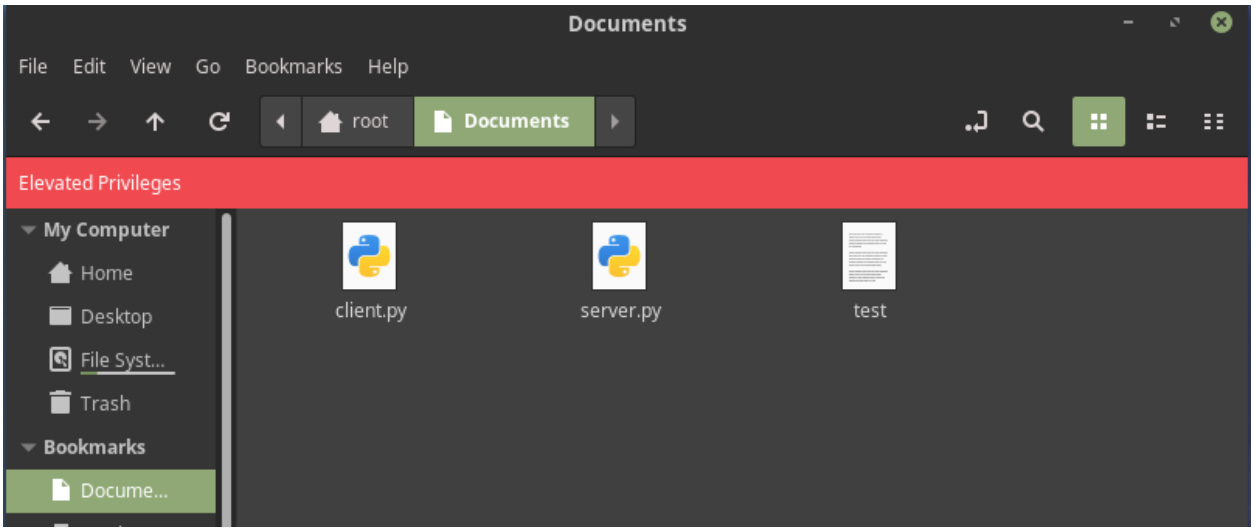
Test #5

We type in the command “pwd” which gets sent to the server, then returns the path that the server file is currently in.q

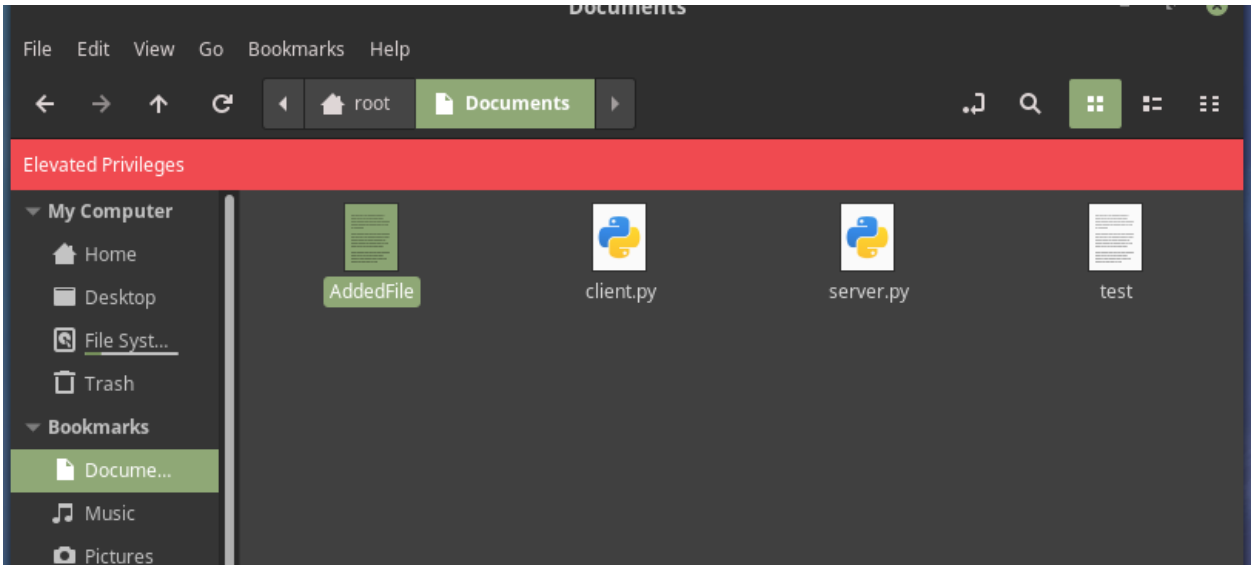


Test #6

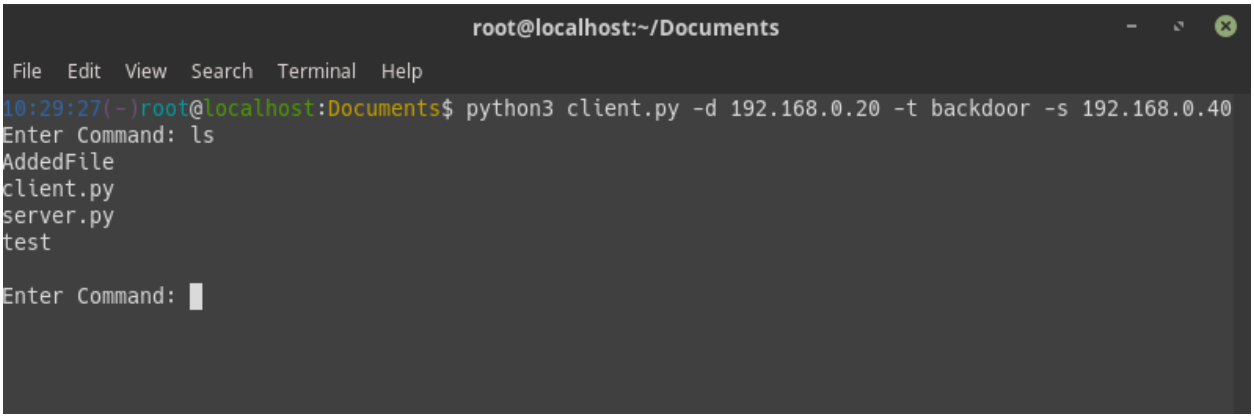
Displayed below is the server showing the current files in the folder.



We create a file called “AddedFile” in the server.

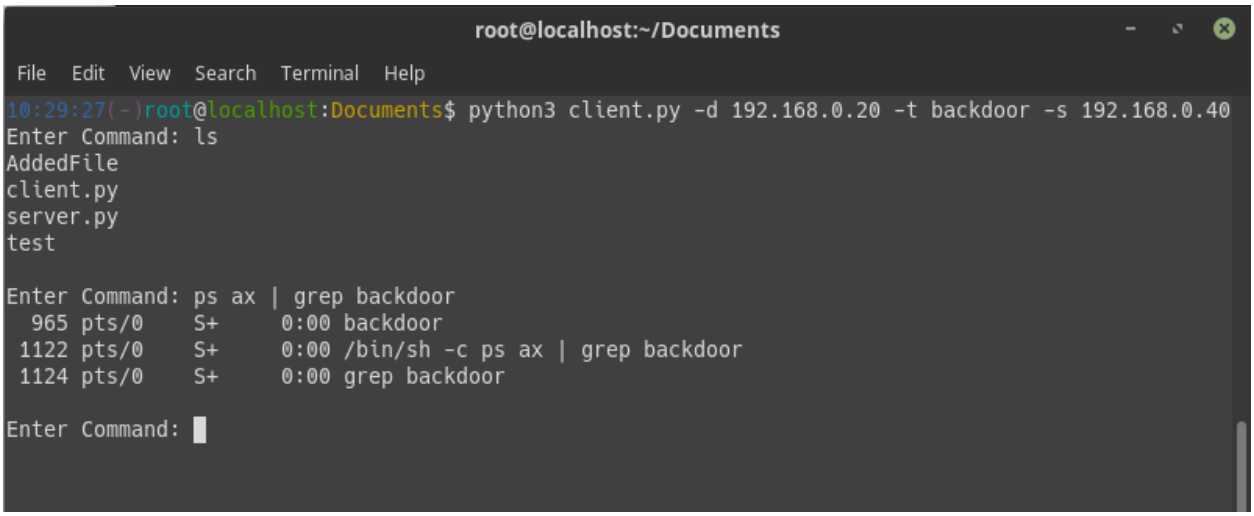


Next, the client types in “ls” to show that the files in the server directory has now been updated to display the new file.



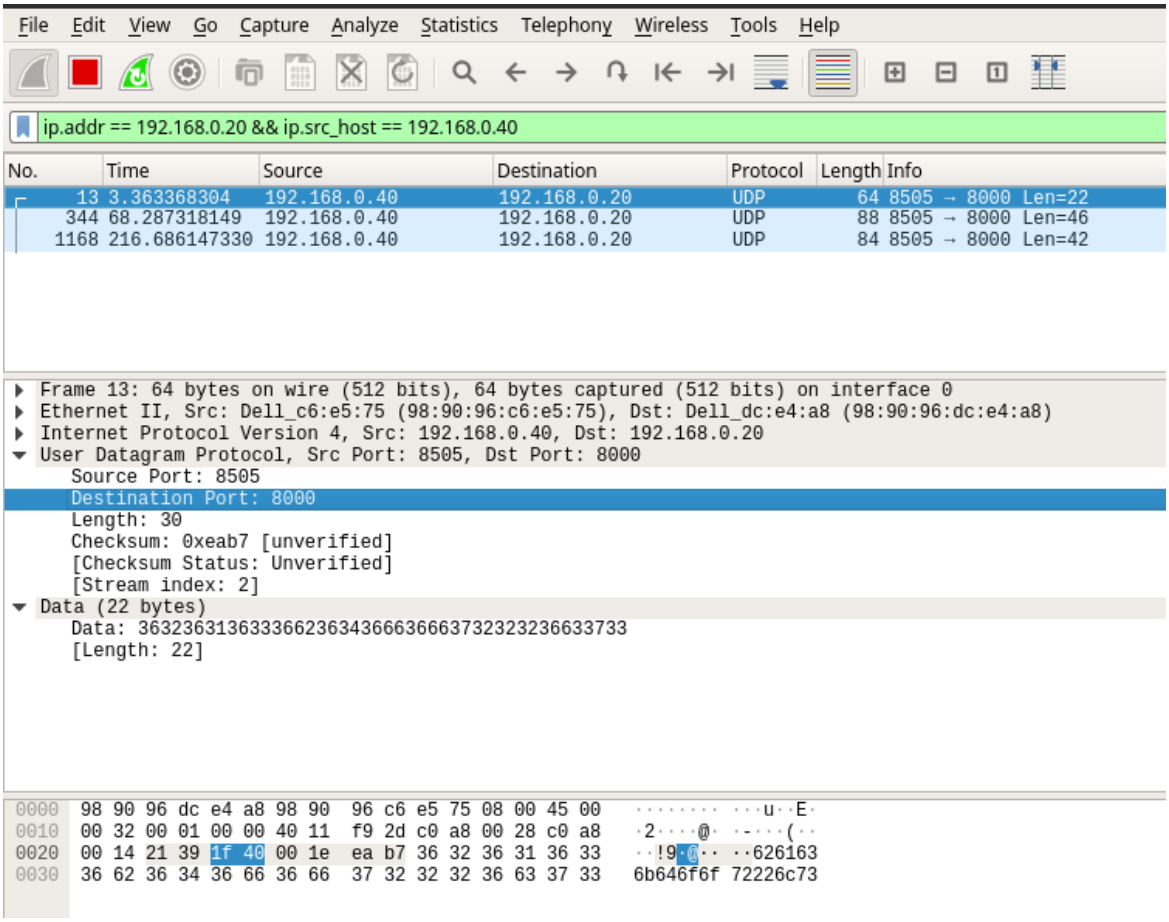
Test #7

We test to see the name of the process that is displayed by using the “ps ax | grep [process name]. Below shows the processes being ran with our name that we used in our arguments.



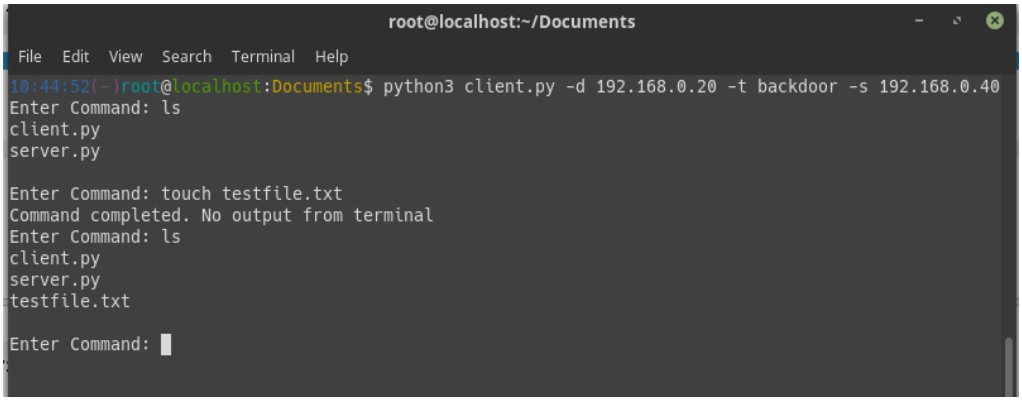
Test #8

Wireshark displays the data that was sent, encoded in Ascii.



Test #9

Below shows the command to create a testfile. Server completes the command and we display the current files in the folder.



Test # 10

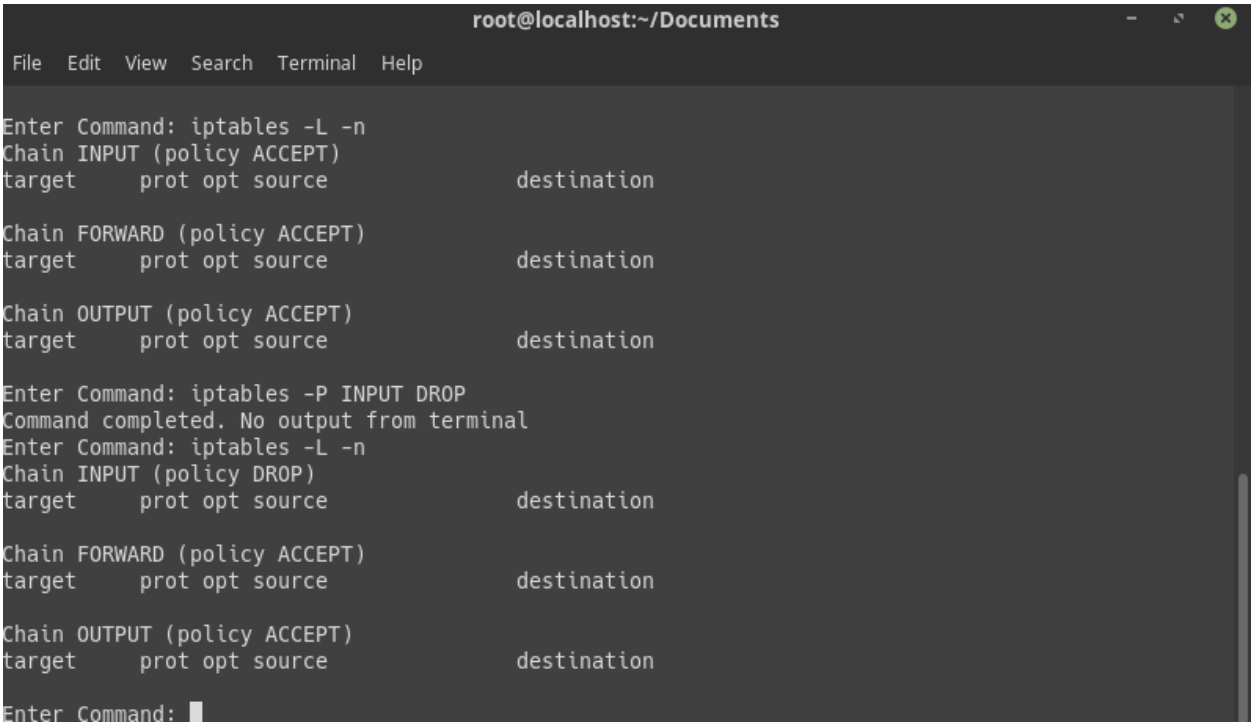
We display the current IP tables on the server. It shows that it's accepting all packets.

```
10:47:16(-)root@datacomm-192-168-0-20:~$ iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Next, on the client we type in to display the IP tables. The following command drops all packets from INPUT. We display the new IP tables and see that the IP tables are now dropping any from any input.



On the server, we now display the IP tables and see that it is indeed dropping all packets.

```
root@datacomm-192-168-0-20:~
File  Edit  View  Search  Terminal  Help
10:46:44(-)root@datacomm-192-168-0-20:~$ iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
10:46:47(-)root@datacomm-192-168-0-20:~$
```

Test #11

The below Wireshark shows the IP being spoofed. We spoofed the IP to become 192.168.0.40 from the computer 192.168.0.112.