

Echo Protocol

Myles G-B

I wrote my echo protocol in python, and in order to implement it I only had to edit the `baseTCPProtocolC()` function and all the other boilerplate code is pretty much untouched. This is my protocol function:

```
def baseTCPProtocolC(csoc):
    print("Starting Echo Protocol...\n")

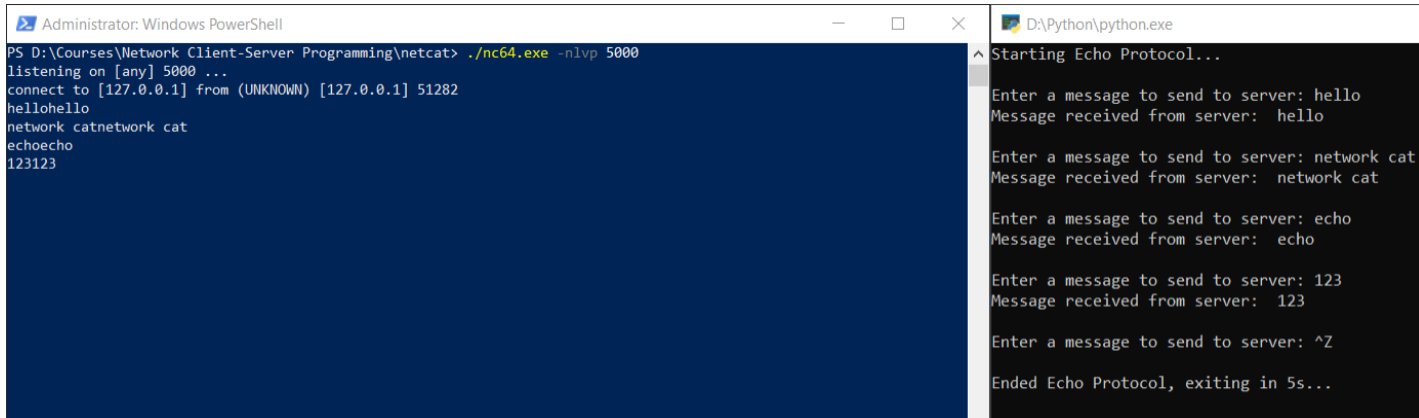
    while True:
        try:
            msg = input("Enter a message to send to server: ")
            csoc.sendall(msg.encode("utf-8"))
            data = loopRecv(csoc, len(msg) + 1)
            print("Message received from server: ", data.decode())
        except EOFError:
            break

    print("\nEnded Echo Protocol, exiting in 5s...")
    time.sleep(5)
```

When the function is called it goes into a while loop with a try-except in it, where the exception being checked is `EOFError`. Since I am on windows, `ctrl+z` is EOF, so if the user wants to stop sending messages to the server they can simply hit `ctrl+z`. At this point, the while loop will break and the user is informed that the program is exiting, and the socket is closed.

The function uses the length of the message the user typed plus one in order to know how much to `loopRecv` from the server, and once it gets the message from the server it decodes it with `data.decode()` so that it can be printed as a normal string and not a byte array.

Example Run



The screenshot shows two side-by-side terminal windows. The left window is an Administrator Windows PowerShell running netcat as a listener on port 5000. It receives a connection from 127.0.0.1 and echoes back the messages 'hellohello', 'network cat', and '123123'. The right window is a Python terminal running a script that starts an 'Echo Protocol'. It prompts the user to enter messages, which are then received from the server and echoed back. The messages 'hello', 'network cat', 'echo', and '123' are shown being received from the server. The protocol ends with a 5-second timeout.

```
Administrator: Windows PowerShell
PS D:\Courses\Network Client-Server Programming\netcat> ./nc64.exe -nlvp 5000
listening on [any] 5000 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 51282
hellohello
network cat
network cat
echoecho
123123
```

```
D:\Python\python.exe
Starting Echo Protocol...
Enter a message to send to server: hello
Message received from server:  hello
Enter a message to send to server: network cat
Message received from server:  network cat
Enter a message to send to server: echo
Message received from server:  echo
Enter a message to send to server: 123
Message received from server:  123
Enter a message to send to server: ^Z
Ended Echo Protocol, exiting in 5s...
```

Source Code

```
import socket
import time

def loopRecv(csoc, size):
    data = bytearray(b" "*size)
    mv = memoryview(data)

    while size:
        rsize = csoc.recv_into(mv, size)
        mv = mv[rsize:]
        size -= rsize

    return data

def baseTCPProtocolC(csoc):
    print("Starting Echo Protocol...\n")

    while True:
        try:
            msg = input("Enter a message to send to server: ")
            csoc.sendall(msg.encode("utf-8"))
```

```
        data = loopRecv(csoc, len(msg) + 1)
        print("Message received from server: ", data.decode())
    except EOFError:
        break

    print("\nEnded Echo Protocol, exiting in 5s...")
    time.sleep(5)

if __name__ == "__main__":
    # create the socket
    # defaults family=AF_INET, type=SOCK_STREAM, proto=0, filno=None
    commsoc = socket.socket()

    # connect to localhost:5000
    commsoc.connect(("localhost", 5000))

    # run the application protocol
    baseTCPProtocolC(commsoc)

    # close the comm socket
    commsoc.close()
```