1. Overview of Application

This is a VPN server. The client sends a message through the VPN server, which forwards the message to a destination server. The VPN server receives the response from the destination server and returns it to the client.

2. Client -> VPN Server Message Format

The client constructs a message with a specific format to communicate with the VPN server. The message format includes two key parts:

Message Format:<Destination Server IP>:<Destination Server Port>||<Actual Message>

Destination Server IP: The IP address of the server where the message will eventually be forwarded.

Destination Server Port: The port number on the destination server to which the message is directed.

Actual Message: The message content that the client wants to send to the destination server.

For example: 192.168.1.10:8080||Hello, this is a test message.

3. VPN Server -> Client Message Format

Once the VPN server receives the message from the client, it parses the message to extract the server's IP, port, and the actual message. The VPN server then establishes a connection with the target server and sends the actual message. After the VPN server receives the response from the destination server, it sends the response back to the client.

Format:
The VPN server simply forwards the raw response from the server back to the client.

Error Handling: If there is an error in processing the message (e.g., invalid destination server IP/port, connection failure), the VPN server sends an error message back to the client in the following format:

4. Example Output

Client -> VPN Server Message:
192.168.1.10:8080||converting from usd:4:25
VPN Server -> Server Message:
converting from usd:4:25
Server -> VPN Server Response:
100

VPN Server -> Client Response:
100

5. Interaction Between Network Layers

The client and VPN server use socket programming to handle network communication over TCP. Here's how the layers interact:

1. **Client Side**:
    - The client creates a TCP socket and connects to the VPN server using the VPN server's IP and port.
    - It then constructs a message with the server's IP, port, and actual message. This message is encoded in UTF-8 and sent to the VPN server.
    - After sending the message, the client waits for a response from the VPN server.
2. **VPN Server Side**:
    - The VPN server listens on a specific IP and port for incoming connections from the client.
    - Upon receiving a message from the client, the VPN server parses it to extract the server IP, port, and message content.
    - The VPN server then establishes a new TCP connection with the specified destination server, sends the message, and waits for a response.
    - Once a response is received, the VPN server forwards the response to the client and closes both connections.

6. Acknowledgments

I worked with Quinn He and a TA in drop-in hours.

Server Line tracing:

server starting - listening for connections at IP 127.0.0.1 and port 65432

Connected established with ('127.0.0.1', 50149)

Received client message: 'converting to usd:7.1:1000' [26 bytes]

sending result message '140.84507042253523' back to client

server is done!

VPN Line tracing:

VPN server starting - listening for connections at IP 127.0.0.1 and port 55554

Connection established with ('127.0.0.1', 50148)

Received client message: '127.0.0.1:65432||converting to usd:7.1:1000'

Parsed message to be forwarded to 127.0.0.1:65432

Received response from server and sent back to client

VPN server is done!

Client Line tracing:

(base) liuzhirou@liuzhiroudeMacBook-Pro csc-249-p2-simple-VPN-server % python client.py --message "converting to usd:7.1:1000"

client starting - connecting to VPN at IP 127.0.0.1 and port 55554

connection established, sending message '127.0.0.1:65432||converting to usd:7.1:1000'

message sent, waiting for reply

Received response: '140.84507042253523' [18 bytes]

client is done!