UDP Pinger

Mahan Madani

Abstract

The goal of this project is to build a local ping server and allow clients to ping the server. UDP is used as the transport-layer protocol (instead of the standard ICMP protocol) and packet loss may occur. The client will also display some statistics about the connection including average RTT and packet loss percentage.

Setting Up the Server

All of the server code can be found in 'ServerMain.py' located alongside this report. The server is implemented using the code provided in the project document. The goal of the ping server is to echo the messages it receives back to the client.

A server socket is created and bound to the host address and port. In order to use UDP, the server socket must be of the type SOCK_DGRAM. It then enters Listening mode, awaiting messages from clients. Every message received has a 70% chance to be echoed back to the client and a 30% chance not to be responded to. This is a random process and therefore, unpredictable.

Setting Up the Client

All of the client code can be found in 'ClientMain.py' located alongside this report. The client side of the code must have the server's IP address and its port to create a socket. Similar to the server socket, the client socket must be of the type SOCK_DGRAM to send UDP messages.

UDP is a connectionless protocol, meaning it doesn't require a handshake in order to establish a connection with the server. 10 UDP messages are sent to the server via a socket, but since the server may not respond to every message, a timeout parameter is set to prevent the client from waiting indefinitely. The Timeout is set to 1 second, after which the client assumes the sent packet has been lost. This causes the application to throw an exception which is easily handled.

```
UDP connection to 127.0.0.1:12000
Request timed out
PING 2 - SENT= 1717577600.986 - RTT= 0.0
PING 3 - SENT= 1717577600.986 - RTT= 0.0008563995361328125
PING 4 - SENT= 1717577600.987 - RTT= 0.0
PING 5 - SENT= 1717577600.987 - RTT= 0.0
PING 6 - SENT= 1717577600.987 - RTT= 0.0
PING 7 - SENT= 1717577600.987 - RTT= 0.0
Request timed out
Request timed out
PING 10 - SENT= 1717577603.008 - RTT= 0.0

Minimum RTT= 0.0
Maximum RTT= 0.0008563995361328125
Average RTT= 0.00012234279087611606
Packet Loss= 30.0%
```

Figure 1: The result of running the UDP Pinger

Figure 1 displays one result of running the client code. Due to the randomness of the server code, rerunning the code might yield different results with different packets getting lost each time. The packet loss rate might change as well.

Each message sent from the client contains the number of the message (1 to 10), and the exact time it was sent. The Round Trip Time (RTT) is calculated after receiving a response from the server. The client prints the information contained in a response alongside the RTT of the message. The minimum, maximum, and average RTT are displayed at the end.

Conclusion

This exercise demonstrates the UDP protocol and its downsides while also highlighting the speed at which messages are delivered using this protocol. Ultimately, UDP is unreliable and cannot guarantee data delivery unless measures are implemented in the application layer.