Programming Exercise 1

Mehraz Abedin Raz (2022293)

Introduction

This project involves building a simple UDP-based ping client-server application. The client sends pings to a server and calculates the round-trip time (RTT) for each ping. We also simulate packet loss and calculate statistics like the packet loss rate. In the second part of the project, a UDP heartbeat mechanism is introduced to check server availability.

System Design

UDP Protocol: We chose UDP because it is connectionless and lightweight, making it ideal for scenarios like pinging or heartbeat checks where reliability isn't the primary concern.

Packet Loss Simulation: The server simulates packet loss using a random integer generation. 30% of the packets are intentionally "dropped" to simulate real-world network issues.

Timeout Handling: On the client side, we implement a 1-second timeout for receiving responses. If no response is received, the client assumes the packet was lost.

Part 1: The client sends 10 pings to the server and calculates the RTT for each.

Part 2 (UDP Heartbeat): The client continuously pings the server and shuts down if it detects three consecutive packet losses.

Assumptions

Unreliable Network: The simulated 30% packet loss reflects an assumption that the network is unreliable, leading to dropped packets.

Response Time Calculation: We assume that the client can calculate the RTT by recording the time a ping is sent and comparing it with the time the response is received.

Client-Server Consistency: We assume the server processes pings in the order they are received, and the client processes responses sequentially.

Implementation Details

Server (server.py):

- The server listens for incoming UDP packets on a specified port (10404). When a packet is received, the server processes it by first determining whether to drop it. If the packet is not dropped, the server echoes the ping message back to the client with a timestamp. Packet loss is simulated using a random number generator.
- The server runs continuously, handling multiple pings without requiring restarts. The use of UDP eliminates the need for a connection handshake, allowing the server to remain lightweight and efficient.

Client (client.py):

- The client sends 10 pings to the server. Each ping consists of a sequence number and a timestamp indicating when the ping was sent. After sending a ping, the client waits for the server's response, measuring the time it takes for the response to return (RTT). If no response is received within 1 second, the client reports a "Request timed out."
- The client calculates the minimum, maximum, and average RTT after all 10 pings are sent. It also calculates the packet loss rate by comparing the number of successfully received pings to the total number of pings sent.

  RTT: The time taken for a ping to travel from the client to the server and back.

  Packet Loss Rate: The percentage of packets that did not receive a response.

UDP Heartbeat (udpheartbeat.py):

- The UDP heartbeat is designed to monitor the server's availability. The heartbeat client sends pings continuously, monitoring whether the server responds within a set time. If three consecutive pings do not receive a response, the client assumes that the server is down and stops sending further pings.
  The UDP heartbeat mechanism is useful in scenarios where regular monitoring of a server's uptime is required, such as in health checks for network services.

Modifications for Heartbeat:

- Both the client and server code from Part 1 are adapted to continuously send and respond to pings, respectively. The client counts the number of consecutive missed responses and stops when the server is presumed to be unavailable (after 3 consecutive timeouts).

Results and Performance

The client reports the minimum, maximum, and average RTT values after sending all pings. Additionally, the packet loss rate is calculated based on how many pings were successfully received.

The UDP Heartbeat client stops after detecting 3 consecutive packet losses.
The average 3-consecutive missed responses from the server is 55 packets from the heartbeat.