Weighted Fair Queueing (WFQ) Server: Implementation and Performance Report

P. Kavinesh Kumar (EE21B068)

May 26, 2025

Implementation Overview

Implemented a Weighted Fair Queueing (WFQ) server in Python using per-packet virtual finish time (VFT) scheduling. Each incoming packet is assigned a VFT based on its flow's weight and the server's virtual time. Packets are stored in a min-heap (priority queue) sorted by VFT. When the buffer is full, the packet with the highest VFT may be replaced if a new packet has a lower VFT, ensuring fairness among flows. The server supports configurable capacity, buffer size, and per-flow weights.

Configuration and Running Instructions

- Edit `SERVER_PORT`, `CLIENT_PORTS`, `FLOW_WEIGHTS`, `CAPACITY`, and `BUFFER_SIZE` in the script as needed.
- Start the server: `server.py`
- Start clients with appropriate port and sending rate.
- Throughput is measured per flow and reported at the server.

Performance Table

Table 1 reports the performance of the WFQ server under various traffic scenarios.

Table 1: WFQ Server Configuration

Capacity C (Pkts/sec)	**Flow Weights** $\{W\}$	**Buffer Size** B (Pkts)	**Client Rate** (Pkts/sec)	**Flow Throughput(s)** (Pkts/sec)
10	1:1:1	10	10:10:10	3.33:3.33:3.33
10	1:1:1	10	10:20:40	3.33:3.33:3.33
10	1:2:4	10	10:10:10	1.43:2.86:5.70
20	1:2:4	100	10:10:10	3.40:6.80:9.80

*Note: There are very small deviations in last 2 cases from the expected throughput. This is because of `time.sleep()` used in the code. Improvements have been done to increase the accuracy.