Report (Evaluation Study)

1. Introduction:-

Load Balancing is an import mechanism in distributed systems, ensuring efficient distribution of clients requests among multiple servers. It enhances application throughput and reduces latency. The report analyses 3 load balancing strategies under the scenario where one server is slow in response.

1. Experiment setup:-

* Topology:- Mininet topology with 3 nodes and 2 servers.
* Load Balancer:- Supports RR, Random and Weighted Response Time Mechanisms.

Server Simulation:-

* Server 1 :- Normal response time
* Server 2 :- Simulated delay

Python scripts to measure average response time, setup mininet topology, and xterms for server simulation.

1. Metrics:-

Latency:- Average time taken for a request to be processed and to be responded.

Throughput:- Number of successful requests processed per second.

1. Results:-

Strategy Average Latency (in ms)

Response time 150

Round Robin 300

Random 400

Strategy Throughput (requests/second)

Response time 60

Round Robin 40

Random 35

1. Analysis:-

Response time:- Assigning more requests to faster server, minimizes latency and optimizes throughput. This is a adaptive request allocation strategy.

Round Robin:- Requests are evenly distributed among servers irrespective of the speed of the servers. So, the slower server increases the average latency and reduced throughput.

Random:- Leads to unpredictable results due to random distribution of requests. It is an inconsistent strategy with higher latency and lower throughput when the slower server receives more requests.

1. Conclusion:-

The response time strategy outperforms the other two strategies and minimizes latency and maximizes throughput. Dynamically adjusts request allocation based on response times, making it ideal for latency-sensitive cases. Continuously monitor server response times to dynamically adjust the weights.