GR-343 An Experimental Study: Performance of Load Balancing Algorithms in SDN-Enabled Cloud Computing Networks

Introduction/Abstract

In today's world, it's crucial to address the limitations of current cloud computing networking infrastructures. Software Defined Networking (SDN) is an innovative approach that utilizes software-based controllers or APIs to manage network traffic on top of hardware infrastructure. SDN plays a crucial role in Cloud Computing as it enables quick adaptation to changes, streamlines network configuration, and boosts network monitoring and performance. This study aims to analyze network performance by implementing load-balancing algorithms for SDN using the POX controller and Mininet software to emulate the network. Programmed in Python, the algorithms create network topology. The findings demonstrate that the Weighted Round Robin algorithm outperforms all the other algorithms that were evaluated.

Objective

To experiment and evaluate the performance of different load balancing algorithms such as Random choice, Round Robin and Weighted Round Robin using a POX controller and emulating the network with Mininet.

Materials and Methods

- Linux Ubuntu 22.04
- Mininet
- POX
- Python 3
- Siege HTTP tool

Experimental Setup

In this study, we implemented three load balancing algorithms - Round Robin, Weighted Round Robin, and Random with Two Choices. The load balancing experiment was conducted on the Ubuntu operating system, utilizing the SDN POX controller and a network topology generated by Mininet. The topology and algorithm were both developed using the Python programming language [3].

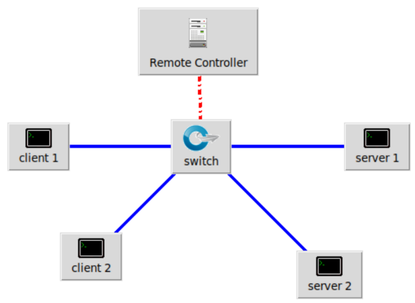

Fig.1 shows a single topology, consisting of a switch that connected the 2 clients (h1, h2), 2 servers (server 1, server 2), and a controller.

To generate this topology, Mininet was connected remotely to the POX controller via port 6633. The clients sent their requests to the virtual switch IP of the SDN controller, which then distributed the requests to the servers based on the chosen load-balancing strategy.

The Curl command was utilized to generate GET requests from the clients. This command is for a single hit to the server. The Siege HTTP tool was used for load testing all the algorithms. Here we use each client node to simulate a N number of concurrent users. This way it is easy for us to represent multiple concurrent users graphically. We plan to measure the average response time and throughput to suggest the best performing algorithm.

Results

To analyze the performance of different load balancing algorithms, we have considered a set of 10, 50 and 100 concurrent users hitting the server. Through the Siege HTTP testing tool, we continuously send requests to the virtual switch, and we examine how each algorithm responds to the heavy load. The results are tabulated as shown below.

Overall Performance of Load Balancing Algorithms		
	Average Response Time (s)	Average Throughput
Random Choice	0.53	0.14
Round-Robin	0.46	0.17
Weighted Round-Robin	0.45	0.18

Table1 shows the average response time in seconds (s) and the average throughput that was obtained by load testing with 10, 50 and 100 concurrent users.

The individual results of Response Time and Throughput are presented in the graphs below

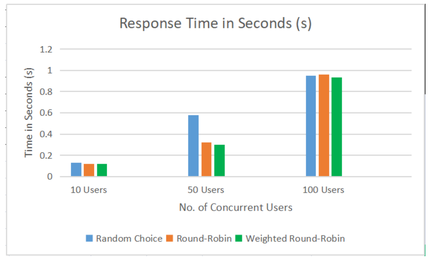

Fig.2 shows the Response Times Achieved when the network is exploited by 10, 50 and 100 concurrent users. We observe that in all the cases, the Weighted Round-Robin Algorithm has the lowest response time when compared with the rest.

Fig.3 shows the Throughput when the network is exploited by 10, 50 and 100 concurrent users. We observe that in all cases, the Weighted Round-Robin Algorithm ha the highest Throughput when compared with the rest.

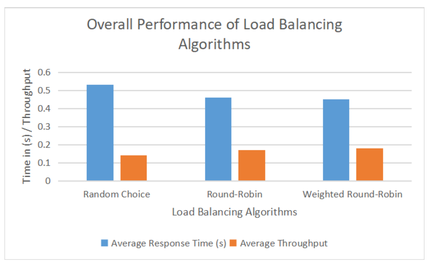

Fig.4 shows the overall performance of each load balancing Algorithm by taking the average of all cases. From this graphical representation we can infer that, the Weighted Round-Robin algorithm has the best performance.

Analysis

After the experiment, we see that the Weighted Round-Robin has the best performance. It is important to analyze as to why the Weighted Round-Robin(WRR) performed better than the Round-Robin(RR) and Random Choices(RC) Algorithms. Let's understand what each algorithm does when hit by a client request.

Random choices Algorithm distributed the requests among servers in a random manner. In such a case, there's a possibility that majority requests are routed to one server though the other server is not busy.

In the case of Round-Robin, the algorithm distributes requests to servers in a sequential manner, starting from the first server to the last one and then cycling back to the first server. Here, we can observe continuous switch between servers, this involves connecting and disconnecting to servers frequently.

Weighted Round-Robin, on the other hand, distributes requests based on each server's weight setting, with servers with higher weight settings receiving more requests. Since the continuous switch between servers is comparatively less than the round-robin method, Weighted Round-Robin outperforms the other two listed algorithms.

Conclusion & Future Scope

As SDN gains more attention and becomes increasingly prevalent in cloud computing networks, the need for a flexible and intelligent load-balancing system becomes paramount. Typically, a load balancer is placed between the firewalls and the server pool. This load balancer implements SDN.

In this project, a load-balancing algorithm is implemented using the SDN-based POX controller, and an SDN-based network is simulated using the Mininet network emulator. The POX controller replicates a variety of load-balancing methods, including the round-robin, random, and weighted round-robin algorithms. According to the experiment's findings, the weighted round-robin algorithm outperforms the other two algorithms in terms of network performance.

Through this experimentation, we have understood how SDN is implemented in load balancing and would want to implement this as a testbed to connect to the real cloud parameters and modify the algorithm to achieve better results.

Contact Information

Email ID of Author: schalla4@students.kennesaw.edu
Email ID of Supervisor: alee146@kennesaw.edu

References

[1] https://medium.com/@danish_raza/software-defined-networks-sdn-7b5e3c25ba97

[2] A. Al-Hasnawi, Sally, and Mahmood K. Ibrahem. 2021. "EMULATION OF THE POX CONTROLLER AS A LOAD BALANCER". Iraqi Journal of Information and Communication Technology 4 (2):9-22.

[3] Vishwakarma, Sujayanth K., and Pavithra H. "Performance Analysis of Different Load Balancing Algorithms in SDN Based Data Center Networks." INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) 11, no. 7 (2022).

[4] Son, Jungmin, and Rajkumar Buyya. "A taxonomy of software-defined networking (SDN)-enabled cloud computing." ACM computing surveys (CSUR) 51, no. 3 (2018): 1-36.

[5] Al-Mashhadi, Saif & Anbar, Mohammed & Jalal, Rana & Al-Ani, Ayman. (2020). Design of Cloud Computing Load Balance System Based on SDN Technology. 10.1007/978-981-15-0058-9_13.

KENNESAW STATE UNIVERSITY
COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING

Author: Samhitha Challagundla schalla4@students.kennesaw.edu
Advisor: Dr. Ahyoung Lee alee146@kennesaw.edu