Load Balancing Using SDN

Anand Wani*, Bhavik Dhandhalya†, Prof. Virendra Shekhawat‡
Computer Science & Information Systems, Birla Insitute of Technology & Science, Pilani, IND.
Email: *h20180143@pilani.bits-pilani.ac.in, †h20180118@pilani.bits-pilani.ac.in, ‡vsshekhawat@pilani.bits-pilani.ac.in

***Abstract*—Traditional networks find it difficult to cope up to the demands of the current applications. SDN addresses this issue by decoupling the control and data planes. Software Defined Network is new network technology which is increasing functionality of networks by making it programmable and vendor neutral. Our Network should be flexible to handle large amount of traffic. In this project we aim to implement an efficient load balancing technique. Here we have analysed and compared the performance of random, round-robin, least connection based strategy and weighted round-robin for balancing the load on the servers using SDN framework want to use it if say you.**

***Index Terms*—Software Defined Networks, Load Balancing, Round Robin.**

I. Introduction

Software Defined Networking (SDN) is the network architecture approach where network control is separated from forwarding plane (data plane) and is directly programmable [1]. SDN model consists of infrastructure layer which consists of switching devices. This layer is responsible for forwarding and data processing capabilities of the network. Next layer of SDN architecture is control layer. It acts as a interface between infrastructure layer and application layer. Control plane is the logical entity that receives instructions or requirements from the SDN Application layer and relays them to the networking components. The control layer also extracts information about the network from the hardware devices and communicates back to the SDN Applications with an abstract view of the network, including statistics and events about what is happening. The application layer consists of SDN application, that is designed to fulfil requirement of user. This layer is residing above the control layer. OpenFlow protocol used in SDN helps in communication of control plane with data plane. The brain of network is implemented in the software called controller. The intelligence or the logic of where and how to make forwarding is residing in control plane. An OpenFlow switch consists of one or more flow tables consisting of rules stating actions to be taken for the incoming packets. Incoming packet is matched against value of rule that are stored in flow tables necessary actions are taken. [4] Actions can be either to drop that packet, forward to controller or forward by normal pipeline processing. On the nature of rules installed, an OpenFlow switch can behave like a switch, firewall, Load balancer, router etc. Fig. 1. highlights the SDN architecture. This project aims at performing load balancing on the servers using various strategies in the SDN network infrastructure [2] Various strategies used for load balancing are Random, Round Robin, Weighted Round Robin and Least connection based strategy.

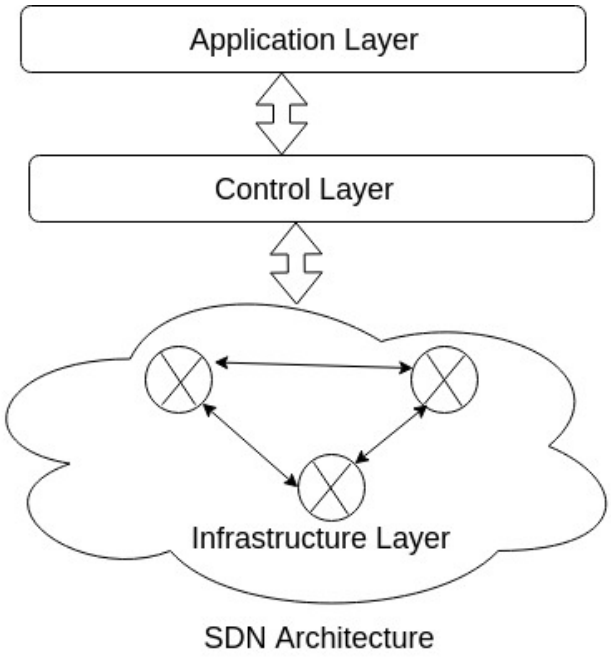

Fig. 1. SDN Architecture

II. System Model

Consider a system in which there are M servers and a single controller for load balancing and distribution on these servers. There are N clients wanting server CPU time and other resources.

III. PROPOSED STRATEGY

In this section, we have presented various strategies used for solving the load balancing problem.

A. Random Strategy

As its name suggests, this algorithm selects a server on random basis, i.e. using an underlying random number generator. In cases wherein the load balancer receives a large number of requests, a Random algorithm will be able to distribute the requests evenly to the nodes. So like Round Robin, the Random algorithm is sufficient for clusters consisting of nodes with similar configurations (CPU, RAM, etc).

B. Round Robin Strategy

Round Robin is the most widely used algorithm. It is easy to implement and easy to understand. In Random strategy, while user requests come to controller it randomly selects one of the available servers [3]. But in round-robin fashion, it

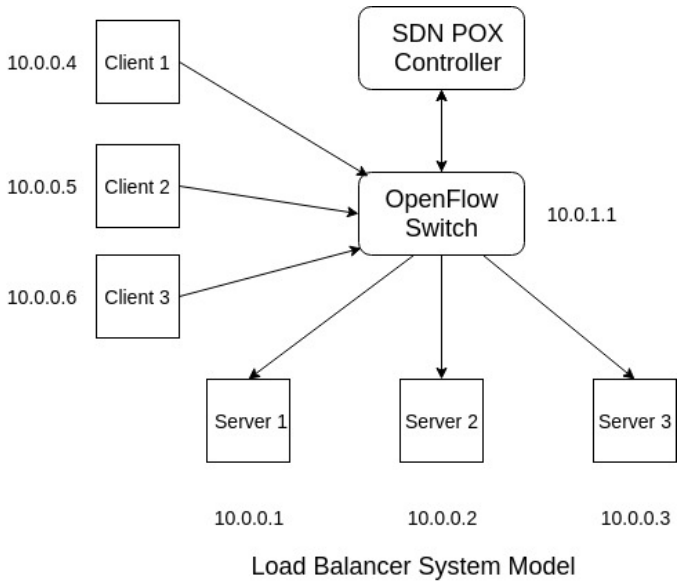

Fig. 2. System Model

tries to distribute load uniformly to all servers. Results show that Round-robin is better than Random strategy in terms of response time, latency and throughput. Let's say we have 2 servers waiting for requests behind the load balancer. Once the first request arrives, the load balancer will forward that request to the 1st server. When the 2nd request arrives, that request will then be forwarded to the 2nd server. Because the 2nd server is the last in this cluster, the next request (i.e., the 3rd) will be forwarded back to the 1st server, the 4th request back to the 2nd server, and so on, in a cyclical fashion.

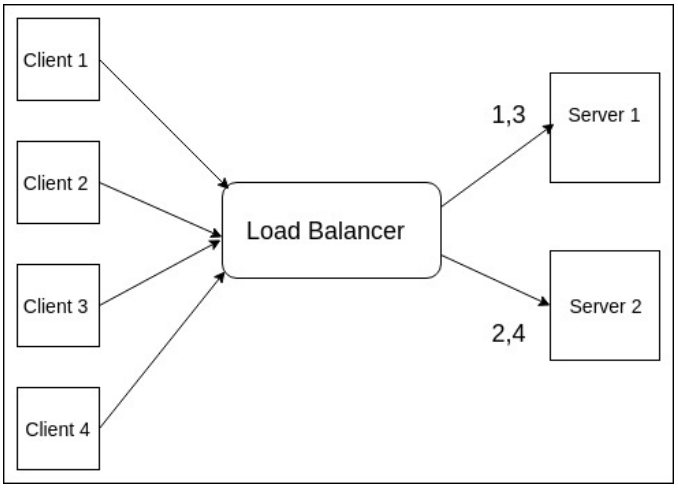

Fig. 3. Round Robin Load Balancer

Even though this method is simple to implement, it has some limitations, For example, if Server 1 has more CPU, RAM, and other specs compared to Server 2, then Server 1 will be able to handle a higher workload than Server 2. Unfortunately, a load balancer running on a round robin algorithm won't be able to treat the two servers accordingly. In spite of the two servers' disproportionate capacities, the load balancer will still distribute requests equally. As a result, Server 2 can get overloaded faster and probably even go down. The Round Robin algorithm is best for clusters consisting of servers with identical specs.

C. Weighted Round Robin Strategy

In this approach we can differentiate between heterogeneous servers. Example, if Server 1 is having higher specs than Server 2, the algorithm will assign more requests to the server with a higher capability of handling load. This algorithm is called as Weighted Round Robin [2]. The Weighted Round Robin is similar to the Round Robin in a sense that the requests are assigned to the servers in cyclical manner only, but with a twist. The server with the higher specs will be assigned a greater number of requests [5]. Here when we set up the load balancer, we assign "weights" to each node. The node with the higher specs will be given the higher weight. Weights are in proportion to actual capacities. For example, if Server 1's capacity is 2x more than Server 2's, then server 1 is assigned a weight of 2 and Server 2 is assigned a weight of 1.

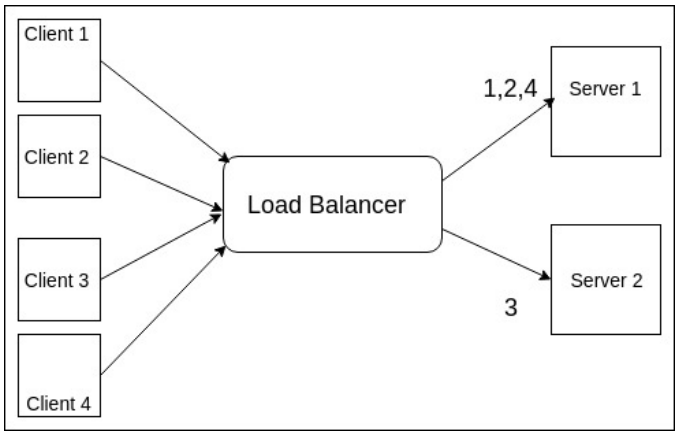

Fig. 4. Weighted Round Robin

So when clients requests arrive, the first 2 client request will be assigned to server 1 and the 3rd to server 2. If more clients come in, the same sequence will be followed. That is, the 4th, 5th, 7th, 8th will all go to Server1, and the 6th to Server 2, and so on. Capacity isn't the only basis for choosing the Weighted Round Robin (WRR) algorithm. Sometimes, if you want server one to get a substantially lower number of connections than an equally capable server for the reason that the first server is running business critical applications and you don't want it to be easily overloaded.

D. Least Connections

Consider an instance where, even if two servers in a cluster have exactly the same specs, one server can still get overloaded considerably faster than the other. The possible reason for this to happen is because clients connecting to Server 2 stay connected much longer than those connecting to Server 1. This can cause the total current connections in Server 2 to

pile up, while those of Server 1 with clients connecting and disconnecting over shorter times, would virtually remain the same. As a result, Server 2's resources can run out faster. In such a situations, the Least Connections algorithm is a better fit. This algorithm takes into consideration the number of current connections each server has. When a client attempts to connect, the load balancer will try to determine which server has the least number of connections and then assign the new connection to that server.

IV. EXPERIMENTS

We have used mininet for simulating the network. We created the topology as shown in the system model, it has 6 hosts, one SDN switch and one SDN controller. Controller used is the POX Controller. Protocol used is openflow. 3 hosts are servers and remaining are clients. Servers are running HTTP server application and clients are requesting files of various sizes from the servers. Client program also logs the response time for each GET request. This response time is used for plotting graphs and comparing the response time for various approaches.

V. OBSERVATIONS

In this sections, we present our observations with the help of various graphs.

A. Round Robin And Random

In this case all servers are of equal configuration, so weights are same, here we have compared the response time, for 100 requests from 3 hosts for different file sizes. The graph shows that the average response time for each file size is less that is better in case of Round Robin that random strategy. The Table. 1. shows, the average response time of different file sizes for Round Robin and Random strategy.

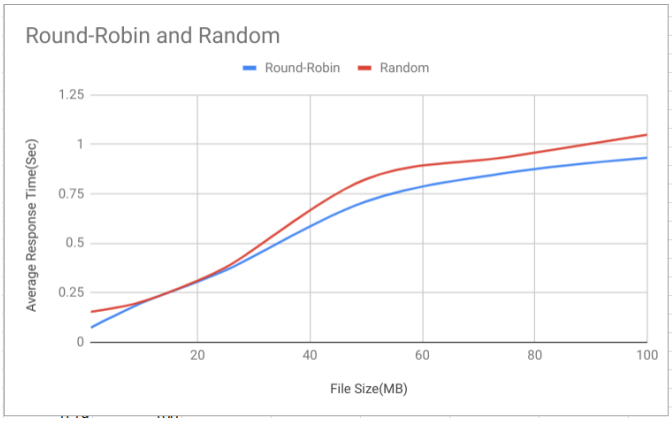

Fig. 5. Round Robin and Random

B. Weighted Round Robin, Round Robin and Random

In real world networks, all servers are not identical in terms of specifications, so we have assigned weights to each server based on its capacity to process requests. Based on these weights the controller will insert flow rules in the switch, so that server with higher weight will get more client requests. We have used link delays for simulating servers of different configurations. We test this case for various iterations and in each iteration for various file sizes, then plotted graphs of response time vs file size.

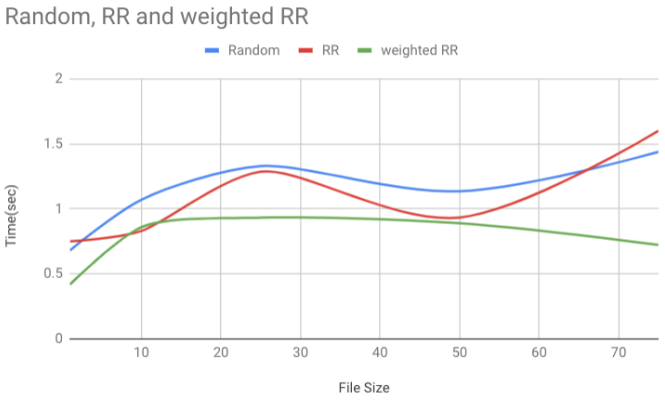

Fig. 6. Round Robin, Random and Weighted Round Robin, 50 File Requests per client

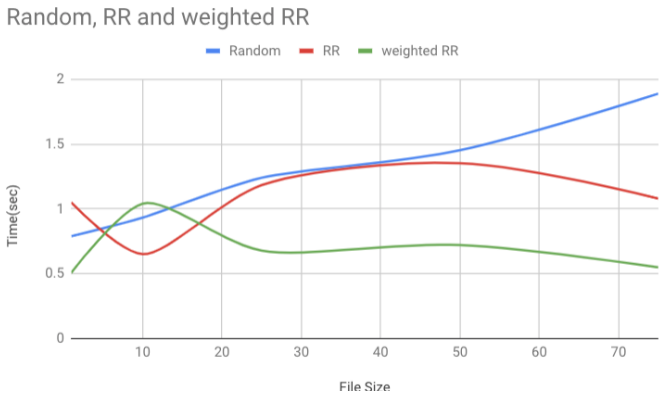

Fig. 7. Round Robin, Random and Weighted Round Robin, 100 File Requests per client

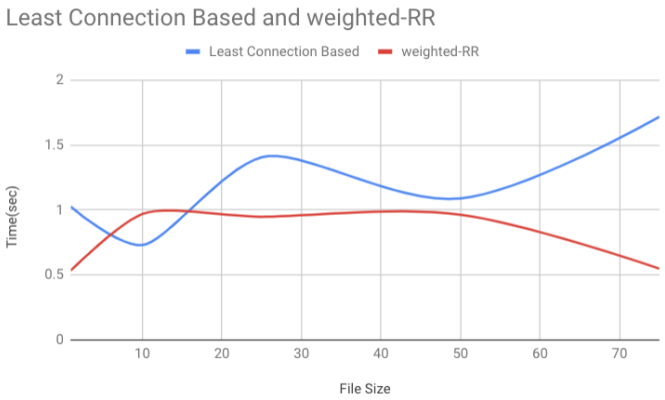

Fig. 8. Least Connection Based and Round Robin

C. Comparison based on Turn around time

According to fig. 9 weighted-RR is performing better than random and traditional RR even when packets/files were having different size. During these implementation we have assumed that we know either link delay or server capacity and based on that we have fixed threshold. The relation between server capacity and threshold is still an open question. If one can find the threshold manually and fix it, then we can have enormous performance advantage over any other algorithms as shown in fig. 10. In fig. 10 we have considered only similar sized packets and we have fixed the threshold manually. That manual threshold lead us to enormous performance.

Fig. 9. Turn Around Time with different packet size

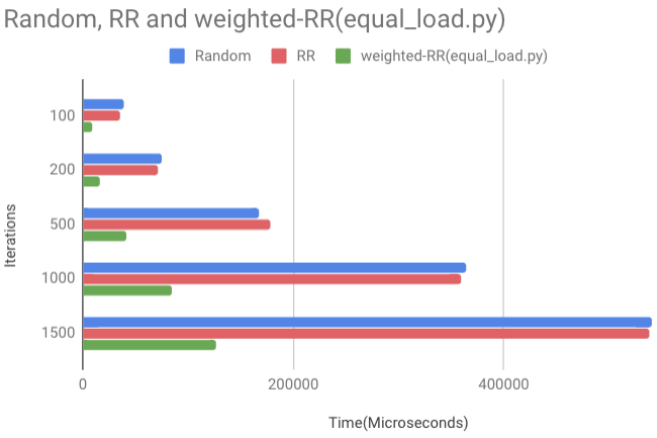

Fig. 10. Turn Around Time with same packet size

VI. CONCLUSIONS

Here we have implemented strategy of load balancing using POX controller that works on the principle of SDN. Network is simulated using mininet. We simulated different strategies like random, round robin, weighted round robin and least connections algorithm. The results of our simulation show that weighted round robin out performs all other mentioned algorithms. Random algorithm was giving quite close performance to round-robin because of least number of servers(i.e. 3 in our case). The limitation of our weighted round robin is that we have assumed that initially either we know the link delay or server capacity. This parameter helps to calculate threshold for weighted round robin. We have also assumed the channel/bandwidth of same capacity and error free.

VII. FUTURE SCOPE

In this paper, while implementing Random algorithm we assumed all bandwidth have same capacity and they are error free. In future it can be considered as challenge to improve load balancing algorithm. Average response time & Turn around time was considered as performance metric, but there can be other parameters like bandwidth/channel utilization, Server utilization, ..etc. In weighted-RR, we were not able to come up with relation between link-delay to threshold, so in future it can be improved based on a machine learning statistics result.

REFERENCES

[1] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, "Software Defined Networking : A Comprehensive Survey," Proceedings of the IEEE, Vol. 103, No. 1, January 2015.

[2] Wen-Hwa Liao, Ssu-Chi Kuai and Cheng-Hsiu Lu, "Dynamic Load-Balancing Mechanism for Software-Defined Networking," *International Conference on Networking and Network Applications*, 2016.

[3] Sukhveer Kaur, Japinder Singh, Krishan Kumar and Navtej Singh Ghumman, "Round-Robin Based Load Balancing in Software Defined Networking," Institute of Electrical and Electronics Engineers IEEE, 2015.

[4] Saket Bhelekar, Mrdvika Iyer, Gargee Mehta and Sheetal Chaudhari, "Dynamic Load Balancing Strategy in Software-Defined Networking," International Conference on Trends in Electronics and Informatics ICEI, 2017.

[5] Walber Jose Adriano Silva, Kelvin Lopes Dias, Djamel Fawzi Hadj Sadok, "A Performance Evaluation of Software Defined Networking Load Balancers Implementations,"Institute of Electrical and Electronics Engineers IEEE, 2017.