Volume 6, Issue 6, June 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Weighted Round-Robin Load Balancing Using Software Defined Networking

Sabiya, Japinder Singh
Department of Computer Science and Engineering, SBSSTC, Ferozpur,
Punjab, India

Abstract— In today's environment our networks have to handle enormous amount of traffic, serve thousands of clients. For a single server it is very troublesome to handle such huge load. The solution is to use multiple servers with load balancer acting as a front end. The load balancer will receive the request from multiple clients. Depending upon various load balancing strategy the purpose of load balancer is to forward these incoming requests of the client to different servers. Load balancer use customized hardware. These traditional load balancers are exorbitant and not flexible to change. As traditional load balancer are vendor locked and due to its non programmable nature network administrators does not have capability to build their algorithms by own. But in case of SDN load balancers, it provide the facility to the programmers to build and design our own load balancing strategy thus makes the SDN load balancer programmable and flexible. Another strong point of using Software Defined Networking load balancer is that it does not require any separate hardware that behaves as a load balancer. The simple dumb silicon device can be converted to a powerful load balancer by the application that is built on top of controller without the need of network administration. In this paper we highlighted the implementation detail as well as experimentation results of weighted Round-Robin load balancing strategy using an OpenFlow switch connected to a POX controller that are based on python.

Keywords— Software Defined Networking, Mininet, Load Balancer, POX, OpenFlow

I. INTRODUCTION

Traditional network's technology is very complicated and difficult to administer. The devices that are helpful in running these devices are locked and specific to particular vendor. It is responsibility of network administrators to configure each individual network devices. Even more distressing is, if same vendor write different products it require almost different configuration. Conventional Networks has manual configuration, very inflexible to change, and require high operational expenditure for running the network. With the advent of new technology particularly in the areas of mobile, social, and big data centres computer networks require high bandwidth and dynamic management. This growing network requires changes in exciting computer network so that complexity can be managed.

Hence new emerging field in the area of information technology is the Software-Defined Networking (SDN) where network control is separated from forwarding plane (data plane) and is programmable directly, hence eliminate the need of network administer [1]. SDN model consists of infrastructure layer which consists of switching devices. Main aim of this layer is to collect network status, temporary stored them in local devices and pass them to controller. Next layer of SDN architecture is control layer. It acts as a interface between infrastructure layer and application layer [3]. The application layer consists of SDN application that is designed to fulfil requirement of user. This layer is residing above the control layer [7]. OpenFlow protocol used in SDN helps in communication of control plane with data plane. The logic of control plane is implemented in the software called controller. The intelligence of where and how to make forwarding is residing in control plane. An OpenFlow switch consists of one or more flow tables. Rules of packet handling are stored in flow tables. Incoming packet is matched against value of rule that are stored in flow tables and traffic that match a rule against entries stored in flow table corresponding actions are performed on the them. 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 [2]. Figure 1 highlights the SDN architecture.

The main Contribution of this research paper is:

- Performing Weighted Round-Robin Load balancing code.
- To test our load balancing application Mininet emulator was used.
- Contrasting on different load of Weighted Round Robin load balancing using SDN with Round Robin load balancing based on parameter Response Time and Transactions executed per Seconds using Openload testing tool.

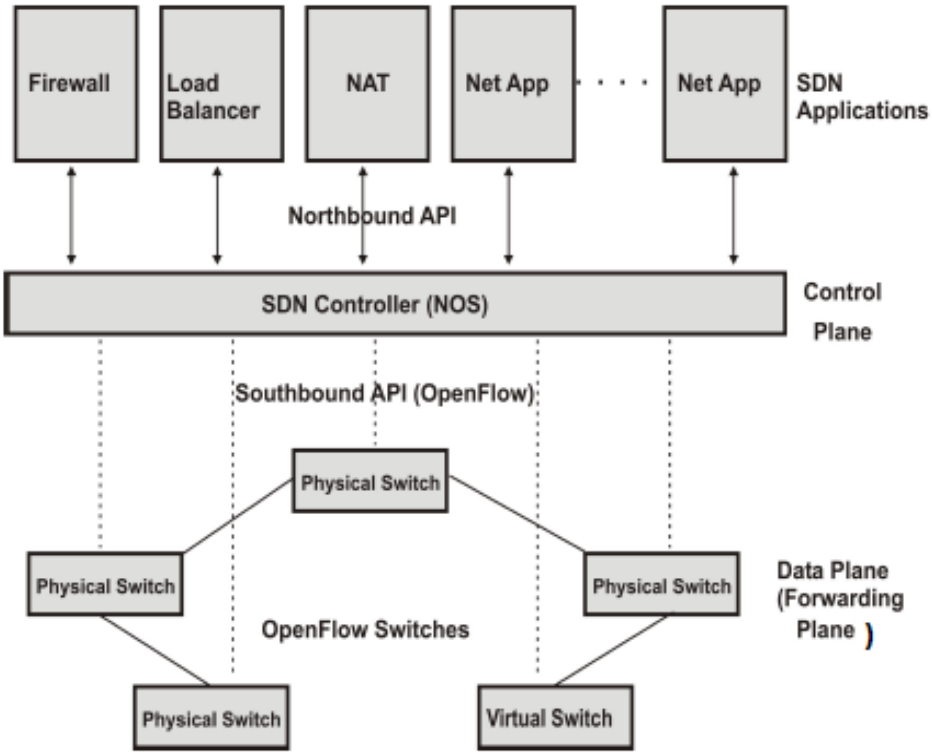

Fig 1: Architecture of Software defined Networking

The blueprint of our paper is describes as. Section II describe paradigm of load balancing. Section III contains architecture of load balancer. Section IV describes various load balancing strategies. Section V describes implementation details. Section VI covers Experimental Evaluation setup and Section VII covers conclusion and proposed future work.

II. MOTIVATION

With the advent of new internet technology the load on the servers are increasing day by day. Therefore there is a need to balance these load on servers in order to provide efficient services to end users without any delay. Main application of using load balancer in the data centers where each client's requests are replicated among various servers for the purpose of minimizing response time and maximizing transaction rate. Technique of Load balancer helps to forward the requests send by client across various servers. This Traditional load balancer architecture is implemented on dedicated hardware, very expensive, non programmable and vendor-specific devices. In order to remove these complexity SDN load balancer was implemented. SDN Load balancer helps in increases network performance by well utilization of available resources and results in minimizing Response Time and maximizing number of Transaction that are executed per second. Technique used for our SDN load balancer execution does not require any isolated load balancer device. It allows result for the flexibility of network topology. Our solution can be scaled well while increasing number of switches and servers, and handling large number of client request.

Our task includes implementing weighted round-robin load balancing strategy using POX controller and results are contrasted with round-robin based load balancing strategy.

III. ARCHITECTURE OF LOAD BALANCER

The architecture of load balancing believes here consists of a load balancer that are connected to several target servers. The load balancer receives requests from the clients, the requests received from the client are then redirect to the target servers following a policy that are fixed by administer. Fig 2 illustrates the design of load balancer. The architecture of load balancing consists of OpenFlow switch connected to one POX controller and multiple servers that are linked through OpenFlow switch's ports [4]. Static IP address is assigned to each server and the dictionary of live servers is maintained in POX controllers that are connected to the OpenFlow switch. Controller contains a virtual address. All requests coming from the clients are redirect to the virtual IP address .When the client dispatch a request packet to the virtual IP, information that are contained in the packet header, OpenFlow switch uses this information and contrast these information with the information stored in flow entries of switch, if the entries in flow table matches with client's packet header information then based on strategy implement on load balancer switch modifies the destination virtual IP address to the address of one of the servers and forward the packet to that particular server, If entries in flow table does not matches with any header information contained in packet flow entry, then the OpenFlow redirect packet to the controller.

With the help of OpenFlow table the Controller inserts new flow entries to the switch's flow table [9]. To implement load-balancing application, we wrote python modules that are executed by the POX controller.

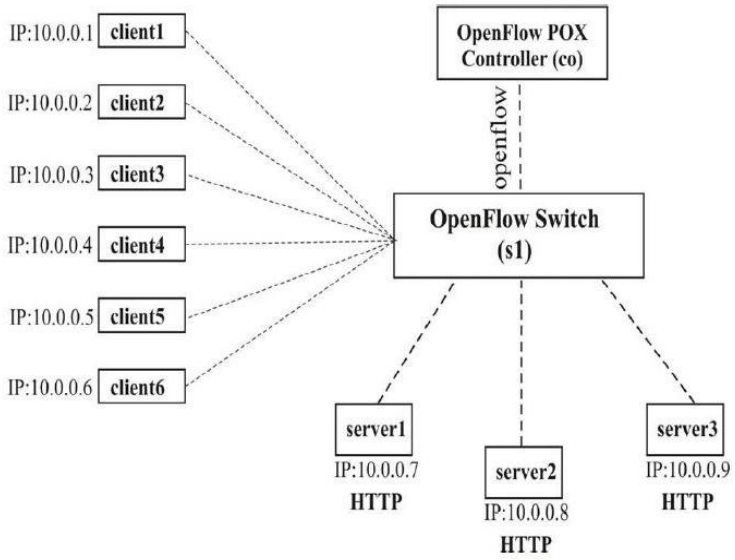

Fig 2: Load balancer architecture

IV. LOAD BALANICNG STRATEGIES

To effectively schedule the routing of requests from a client to the respective servers in an optimized way, several load balancing strategy are used that are as follows:

- **RANDOM STRATEGY**: From a list of live servers, the Load Balancer will randomly choose a server for sending request. This policy has large overheads.
- **ROUND-ROBIN**: In this, each servers receive the request from clients in circular manner. The requests are allocated to various live servers on round robin base [6].
- **WEIGHTED ROUND-ROBIN**: In this, each server receive the request from the client based on criteria that are fixed by the site administrator. In other world A Static weight is assigned to each server in Weighted Round Robin (WRR) policy. We usually specify weights in proportion to actual capacities. So, for example, if Server 1's capacity is 5 times more than Server 2's, then we can assign a weight of 5 to server 1 and weight of 1 to server 2.

This strategy is generally used when we want one server 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 we don't want it to be easily overloaded. Another importance of using weighted round robin strategy is in Case of heterogeneous servers that is one server having i3 processor another having i5 processors instead of sending similar types of load to each server, static weights is assigned to each server so that server having least capacity will handle only small amount of requests irrespective of server having higher capacity.

V. EXECUTION

The following softwares were helpful in order to execute the functionality of the load balancer:

- **VMPlayer** - In order to run several virtual machines VMPlayer is used.
- **Emulation Tool** - Mininet emulator exports a Python API to create custom experiments and topologies. Mininet is a freely available open source network emulator for prototyping software defined networks. With Python language, Mininet is simple and easy to use. Testing the code on mininet takes just a few seconds. In mininet running, editing, debugging loop takes very less time [5]. As compared to hardware test beds like Emulab, GENI it is inexpensive, always available, and quickly re-configurable. Another advantage is that the code that runs on mininet emulator is the same as the one to be deployed into a real network.
- **Controllers** - Controller decides where and how to make forwarding decision based on application that is written on top of POX Controller. POX is an open source SDN Controller whose modules are implemented in python language. Southbound interface in controller provides information to the switch and northbound interface allow POX controller to interact with various applications. It provides a framework for communicating with SDN switches using OpenFlow protocol.
- **Testing Tool** - In our experiment Openload testing tool is used to test out weighted round robin code. OpenLoad is an open source tool that helps in testing the code based on parameter like Response Time and Number of Transactions that are executed per sec. In this paper, Results of weighted round robin load balancing in SDN are compared with round robin load balancing in SDN.

VI. EXPERIMENTAL SETUP AND RESULTS

Our topology consists of 1 OpenFlow POX controller (c0), 1 Open flow switch (s1), 6 clients (client 1, client 2.....client 6) and server 1, server 2, server 3 are used as 3 web servers. In our topology, we are making dumb OpenFlow switch s1 that operate like a load balancer. Our Load Balancer consists of service IP and service IP's mainly two variety of addresses. Client will dispatch the requests to the service IP. The address that is specified to load balancer was service IP

[8]. It is responsibility of Load Balancer that the incoming request of the client is redirect to the server depending upon the strategy that are implement in load balancer [6]. Our paper implements weighted round robin strategy in which 60 percentage of load is dispatch to server 1, 30 percentages to server 2 and 10 percentage of requests is handling by server 3. This topology is explained in Fig 3. Fig 4 shows the output after implementing weighted round robin strategy on load balancer in software defined networking.

We evaluate results of two load balancing strategies that is contrasting weighted round robin load balancing in SDN with round-robin load balancing in SDN and make comparison on the basis of average response time and transaction per sec. Tool used for testing purpose are OpenLoad. Client 1 to client 6 was used as testing purpose. In our experiment we capture the reading by sending independent load from client 1 to client 6. Results were clearly shown in Fig 5 and Fig 6.

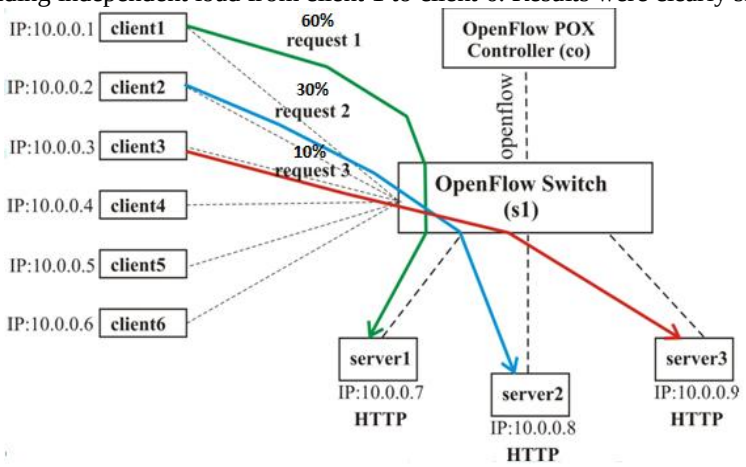

Fig: 3 Weighted Round-robin load balancing

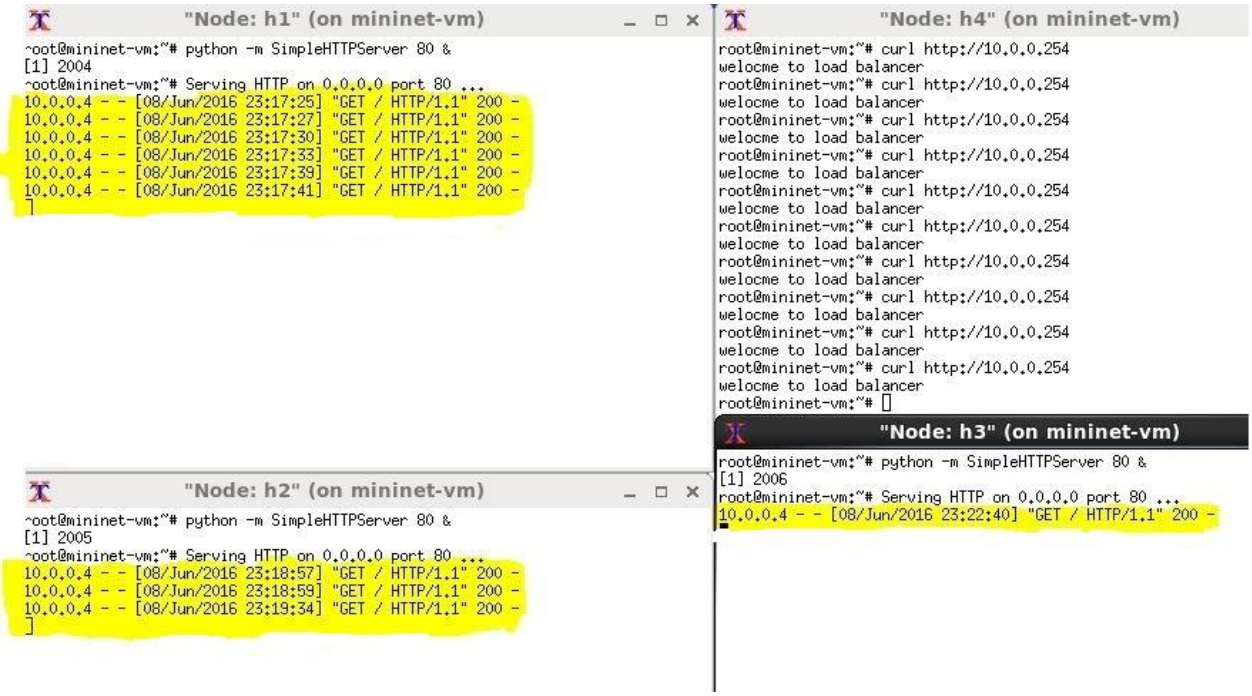

Fig 4: Output of Weighted Round-Robin load balancing

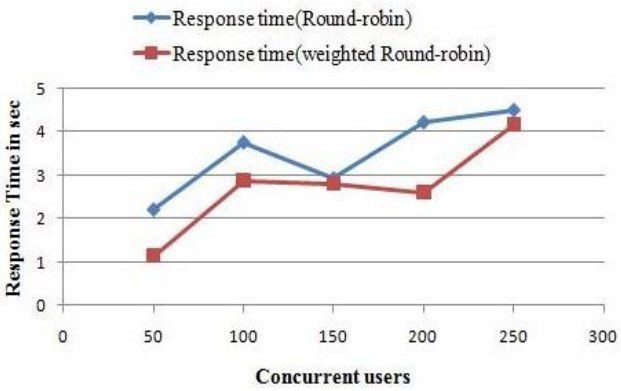

Fig 5: Average Response Time

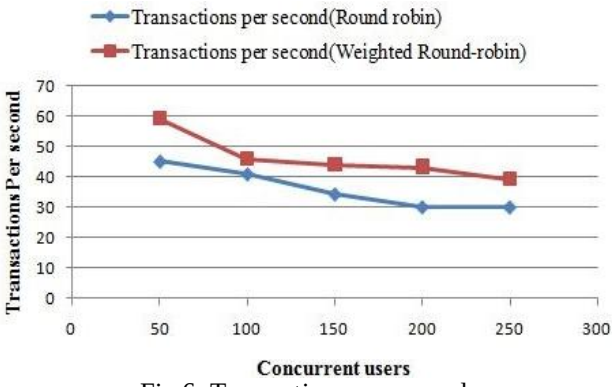

Fig 6: Transactions per second

VII. CONCLUSIONS AND FUTURE SCOPE

Here, we implement strategy of load balancing in POX controller that works on the principle of SDN. In comparison to traditional load balancers SDN load balancers solves many problems. Weighted round robin load balancing using OpenFlow switch in SDN is implemented in this paper and results are co related with round robin load balancer in SDN. Results reveal that weighted round robin strategy is more prominent than round robin strategy. Plus point of using weighted round-robin is that in round robin load balancer it assumes that all the servers are homogeneous in nature that is all servers carry uniform load but in some cases servers are heterogeneous in nature in that case weighted round robin is use. Another advantage about weighted round robin is that sometimes if we want one server 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 we don't want it to be easily overloaded.

Short coming of our task is that only POX controller was used to test our code. We did not take into account any other controllers. Performance of SDN applications also determine by the capability of controller. It could be range of future dissertations .

ACKNOWLEDGMENT

I would like to thanks to my senior friends sukhveer kaur and karmjeet kaur and reveal gratitude to Mr.Vipin Gupta for his useful help.

REFERENCES

[1] Doria, A., Salim, J. H., Haas, R., Khosravi, H., Wang, W., Dong, L., Gopal (2010). "Forwarding and control element separation (forces) protocol specification". Internet requests for comments, RFC editor.

[2] Feamster, N., Rexford, J., and Zegura, E. (2013). "The road to sdn. Queue", 11(12):20.

[3] Erickson, "The beacon openflow controller". In proceeding of second ACM SIGCOMM workshop on hot topics in software defined networking, pages 13-18. ACM.

[4] Sukveer kaur, Japinder Singh." Mininet as a software defined networking testing platform".

[5] Koerner (2012). "Multiple service load-balancing with openflow"., 2012 IEEE 13th international conference on In high performance switching and routing, pages 210-214,IEEE.

[6] Sukveer kaur, Japinder Singh. "Round-robin load balancing in software defined networking". 2nd international conference on computing for sustainable development, 2015.

[7] M.M nunes. "A survey of software defined networking past, present and future of programmable networks".IEEE communication surveys & tutorials, 2014.

[8] Shang, chen. "Design and implementation of server cluster dynamic load balancing based on open flow". In awareness science and technology and ubi-media computing, 2013 international joint conference , pages 691-697, IEEE .

[9] Uppal, H.and Brandon (2010). "Openflow based load blancing" , Proceedings ofCSE561:networking,project report. University of Washington, Spring.

