Proceedings of the International Conference on Sustainable Computing and Smart Systems (ICSCSS 2023)
IEEE Xplore Part Number: CFP23DJ3-ART; ISBN: 979-8-3503-3360-2

An Investigation into Round Robin and Random Algorithms for the Purpose of Load Balancing on Web Servers in SDN Environments

Mr. Sanjay Kadam,
Information Technology,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India
sanjaykadam23@gmail.com

Dr. D. R. Ingle,
Computer Science,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India
dringleus@gmail.com

Tushar Tanaji Jadhav,
Information Technology,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India
jadhavt724@gmail.com

Hitesh Lakshmidhar Damal,
Information Technology,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India
hiteshdamal97@gmail.com

Ashitosh Bramhadeo Kadam,
Information Technology,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India.
kadamashitosh2001@gmail.com

Aditya Nagesh Kale,
Information Technology,
Bharati Vidyapeeth College of Engineering,
Navi Mumbai,India.
kaleaditya0@gmail.com

2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS) | 979-8-3503-3360-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICSCSS57650.2023.10169404

Abstract: **The Future of Computer Networks will be Software Defined Networks (SDN) as it is efficient and has the capabilities to replace the Traditional Networks due to the increasing number of nodes connected to the internet, it is becoming more difficult to adapt to the latest technology in order to maintain its scalability. This is why it is important that the research community is able to test the various aspects of Software-Defined Networking (SDN) to ensure that it can meet the requirements of the industry. Besides being able to provide a solution that can overcome the traditional network limitations, it also allows the researchers to develop a more robust and resilient network. One of the most common controllers used for implementing SDN features is the Ryu Controller. This research aims to provide a comprehensive analysis of the performance of this component in terms of load balancing, through a simulation tool known as Mininet, the researchers were able to perform various tests on the system. The framework known as the Ryu is a software-defined networking framework that is built on components. It allows developers to create easy-to-use control and management applications for network devices. It supports various protocols such as OpenFlow and OVSDB, Ryu fully supports OpenFlow versions 1.0, 1.2, 1.3, 1.4, 1.5, and Nicira Extensions.**

Keywords: **Software Defined Networks (SDN), Mininet, OpenFlow, Ryu , Load Balancing**.

I. INTRODUCTION

The rise of the Internet has made it possible to connect all parts of the world. Unfortunately, implementing and managing traditional IP networks can be very time-consuming and challenging. One of the most common methods of networking is by using a combination of software and hardware. This method involves routing traffic through a network of switches and routers. Unfortunately, due to their vendor-specific nature, traditional load balancers can't be programmed.[1]

The impression of software-defined networking (SDN) allows network managers to generate a centralized global control center for their networks. This method involves separating the various control planes from one another. The control operations are then moved to a logical central controller using the OpenFlow protocol[2]. The controller is usually responsible for the configuration of devices that are compliant with the latest software defined networking (SDN) standards. It can also provide fault management, performance, and network topology analysis.

It can manage the connections between multiple devices. A well-designed and centralized algorithm can handle various performance issues. SDN operates with centralized decision making, meaning each network device takes instructions from one central SDN controller. The controller monitors and analyses network activity in real time for enhanced management and control over its operations.

A load balancing algorithm is applied to maintain the performance of a cluster of computers or a multitude of devices, such as CPUs, network connections, and disks. It can prevent crashes, optimize throughput, and reduce reaction time. It can be used to get rid of physical servers during a cycle so that the rest of the servers can continue to work. By running many instances of an application or service on a server and basing those operations on the analyzed data, it additionally assists with sharing resources [3].

Load balancers, which are also known as application delivery Controllers, have progressed to the point that they are now capable of providing features like as security, acceleration, and authentication in a form that is both small and malleable. Traditional networks rely on inaccurate network information to carry out load balancing. With SDN, load balancing can be performed centrally, which eliminates the need for dispersed decision-making. It also allows for various flow characteristics and link usage rates to be thoroughly planned. Through the use of SDN, network operators can implement network services more easily. The flexibility and cost-effectiveness in managing their traffic streams that organisations seek are made possible by load balancing, which allows these organisations to fulfil their goals. For instance, load balancing will automatically distribute the incoming traffic over all of the current nodes once a new server is introduced. This ensures that the data traffic is efficiently distributed across the network. Consideration should be given when selecting an SDN Ryu controller for use as a load balancer, such as its open-source nature, its support of various protocols and networking technologies, libraries APIs or custom app development capabilities as criteria to guide selection of an appropriate controller.

The proposed method takes into account the use of random and round-robin methods for load balancing in software defined networks. The performance of these two algorithms is analyzed through a variety of scenarios in Mininet. In addition, packet loss and throughput are also examined to compare their performance.[4]

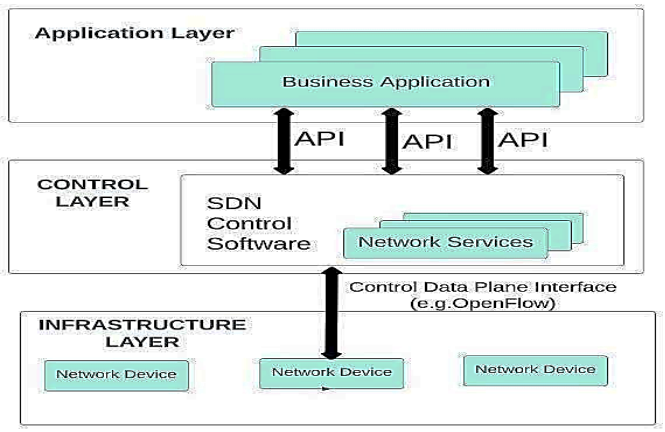

Fig 1:-SDN ARCHITECTURE

Figure 1 shows the SDN architectures consists of three components:

SDN Networking Devices / Network Infrastructure: The SDN networking devices has the control of forwarding data and data processing capability for the network which consists of the forwarding as well as the processing of the data flow. For example, turn the switch on (open flow protocol)
South Bound Interface: It is the link between the controller and the physically networking gear, and it is referred to as the South Bound Interface.

SDN Controller: An SDN controller is an SDN application that delivers a flow control framework for improving the performance and administration of networks. Its protocol-based platform runs on a server and instructs switches where to send and receive packets. RYU, OpenDayLight, ONOS, FloodLight, and so on are a few examples.

SDN Applications:
SDN Applications are computer applications that interact with the SDN Controller by means of an APIs. Applications such as networking administration, statistics, and enterprise applications are all examples of this category, and they are used to handle enormous data centres. This is where research, inventions, fresh concepts, etc. are done.. The connection between the controller and applications is called Northbound Interface.

II. RELATED WORK

In [1] The author has developed a load balancing algorithm that uses a round robin algorithm and a random algorithm strategy, which can be used in combination with OpenFlow Switch. The results of the study show that the algorithm can improve the performance of the network by reducing packet loss and increasing throughput.

An AI-powered routing strategy that provides benefits for software-defined networking was created and tested by the author, according to [2]. The most important purpose of this routing strategy is to reduce the amount of work that must be done by the delivery servers, improve the level of network usage, and keep the network from becoming congested. A comprehensive comprehension of the simulated network was achieved via the development of the prototype.

In[3] In order to analyze the performance of a cloud-based software-defined networking solution, the author used a load balance mechanism to manage the traffic between the client and the server. According to the outcomes of the research conducted, the HA proxy had satisfactory response times and did not turn down any requests even after processing 6 million requests. The Round Robin method was used, which is appropriate for use with pools that include resources that are all the same. However, using the weighted round-robin approach is recommended when dealing with non-identical pools since it ensures a fair distribution of resources.

In [4] the author has calculated the round-trip time, also known as RTT, for two different scenarios: one with a connection list application that uses the Dijkstra method, and another that does not use the algorithm. During the evaluation of the system using the mininet emulator, it was found that the RTT reduced following the implementation of the algorithm. This was a positive finding. This ultimately prompted the author of the study to reach her conclusion.

In[5]the author focused on the effects of load balancing on software defined networking (SDNs). In two cases, RTT was calculated with the use of Dijkstra and least connection algorithms. The result showed that the load balancing algorithm can reduce the time it takes to transmit a packet.

In [6] the author analyzed the various factors that affect the performance of a data center's traffic load balancing system. He focused on the network architecture and the flow table's flexibility. He also analyzed the control plane's concentration and the number of algorithms that can be used to manage the traffic.

In [7] research work aims to analyze the discussion related to the modern load balancing techniques in software-defined networking. They are divided into two categories: AI-based techniques and conventional techniques. The study also takes into account the various parameters of the system to check its efficiency.

In[8] research work both random and round robin balancing techniques are implemented and then weighted and round robin load balancer is used to increase the throughput ratio and address the little utilization of server.

In [9] this research the traffic is distributed in two flows i.e elephant flow and mice flow also there are routed by different method and to determine whether the reroute mechanism is required to be activated the moment vary merit of the variance of link utilized is calculated.

In [10] this research presents an evaluation of the efficiency of a standard approach to implementing a network software defined controller (SDC) framework. The method is designed to improve the efficiency of the system and reduce the overhead.

In[11] the primary purpose of this study is to explore the traffic efficiency of load balancing algorithms such as round-robin (RR), least connected, and least loaded in the distribution of traffic. This will be accomplished by comparing the three methods. The research project consisted of analysing the CPU usage of eight HTTP servers that were distributed via a load balancer. The Opnet network was used so that the actual network could be simulated.

In[12] the research presented in this article demonstrates that methods for closest load balancing display substantial efficiency even in asymptotic terms when an initial load distribution is random rather than worst-case. The investigation was carried out on a d-dimensional mesh network that included computers, and the results suggest that employing diffusion to balance time needs (log n)2/d.

In [13], an original load-balancing algorithm is proposed by taking each server's state into consideration to optimize performance - offering an innovative solution to an important challenge in this field. Fast Switch-Based Approach, this algorithm uses a switch-based fast approach that dynamically balances load efficiently and effectively, providing efficient load balancing solutions in dynamic environments. With its ability to adapt rapidly to changing server conditions quickly, making this the optimal choice in these instances of environment flexibility. The proposed approach does not allow comparison with more modern approaches; as a result, generalization could be limited and only cover load balancing for distributed systems - and not other topics like fault tolerance or scalability. Historically speaking, load balancing solutions remain relevant when managing large distributed systems with multiple nodes that span multiple locations and time zones. This paper lacks detailed implementation details on its algorithm implementation that could impede its usability and reproducibility in practice

III. PROPOSED METHEDOLOGY

SDN (Software Defined Networking) is an emerging networking architecture that separates data and control planes for seamless operation of data paths. SDN stands out by this characteristic compared to traditional networking architectures which tightly couple both planes.

The ability to cluster multiple controllers for high availability and fault tolerance is a feature of the SDN framework. It allows network administrators to easily modify the packet headers in their networks. One of the most important advantages of the software defined switch is its ability to provide load balancing. This allows them to manage the data load on their servers. It can also run multiple applications at the same time and analyze the data to save time and effort. Each server has its hardware/software limitation.

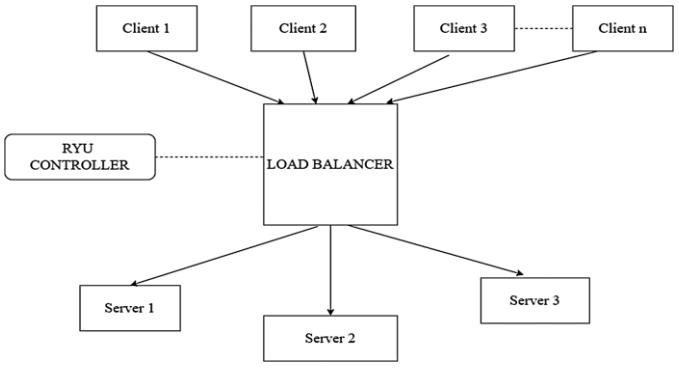

Fig 2. Load balancing Architecture.

Due to the complexity of the task of managing the data load on a single server, a load balancer can be created to distribute the load across the different servers.[7]

This is a load balancing system that uses an OpenFlow switch and a RYU controller. Figure 2 shows the load balancing architecture. It allows the host to communicate with the server and distribute traffic across the network. The virtual IP of the load balancer is also used to allow the client to send and receive information. When the client sends a request, the information is compared with the information in the flow entry of the OpenFlow switch. The switch then adjusts the destination address of the packet to the one of the servers that it is connected to. If the packet does not correspond with an entry in a flow, RYU's switch will forward it directly to their controller; they will decide how best to treat this information packet.

This study will introduce two algorithms that will help us to perform this process: Round Robin and Random.[5]

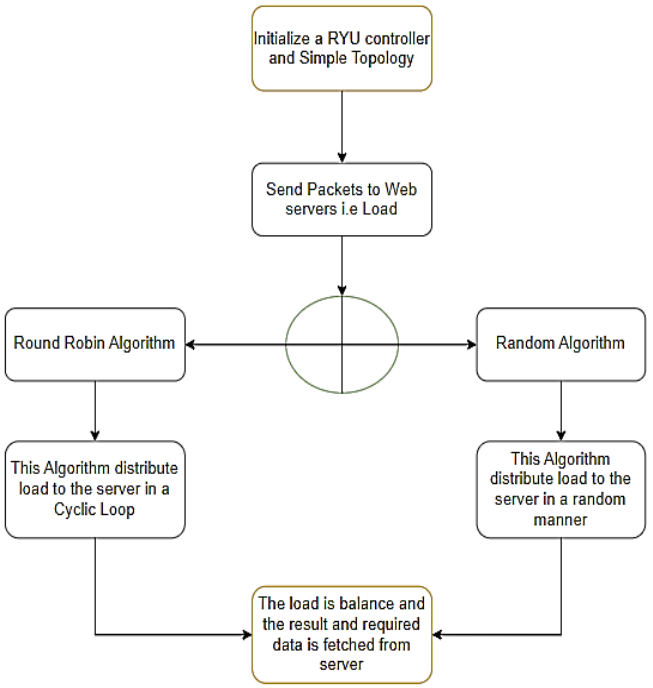

Fig 3. Flowchart

Using a random selection algorithm, Algorithm 1 implements the scheduling of resource requests from client nodes and allocation of servers to client nodes. The load balancer arbitrarily selects one server node from a set of available servers to forward requests from client nodes. Figure 3 shows the detailed flow chart of the proposed methodology.

Algorithm 1: Random algorithm

BEGIN

Randomly assign Flowtime a value between 25 and 75 seconds.

Obtain an inventory of servers that are available (SV_HOSTS).

When a new request from a client is received:

chosen_server = Using random.choices(SV_HOSTS), arbitrarily select a server from the list of available servers.

Assign the request to the selected server.

END

If the load balancing algorithm is not able to distribute requests efficiently due to the disproportionate amount of traffic being sent and received by the server, it can still distribute the requests without considering that this can affect the network. With the help of the Round Robin algorithm, Algorithm 2 uses a random selection method to distribute requests to its clients. The Round Robin algorithm is very easy to implement and comprehend. It is used in the load balancing system to distribute requests to various servers in a circular manner. It can also be used in clusters of servers with the same hardware specifications.

Algorithm 2: Round Robin algorithm
BEGIN
Set Flowtime to a random value between 20-80 seconds
Obtain a list of available servers (sv_hosts)
Set self.last_server_idx = 0
When a new client request is received:
chosen_server = sv_hosts[self.last_server_idx]
Increment self.last_server_idx by 1 (self.last_server_idx = (self.last_server_idx + 1) % len(sv_hosts))
Assign the request to the chosen_server
END

IV. IMPLEMENTATION

A simple and lightweight python-based controller for implementing random selection and round robin algorithms is used with Mininet[14]. It is an Open Source SDN emulation software that can be used to create and test virtual networks. Mininet's command line interface makes it easy to customize and test the networks The ability to improve the performance of paths through OpenFlow protocol is a great help in overcoming the traffic issue caused by complex network topology. The standard enables the control plane and data plane of SDN to communicate with one another. This permits direct communication between the control plane and the forwarding plane. The forwarding plane is comprised of various networking devices, including routers and switches.
The framework "Ryu" is a software-defined networking platform that is built on components [15]. It provides a well-defined API that allows developers to create control applications and network management systems. It supports various protocols such as OpenFlow and OVSDB.

This research used the Round Robin algorithm and the Random selection algorithm to conduct load balancing.
(1) Random Algorithm - This method uses a random number generator to distribute requests to the nodes in the load balancer. It will do so evenly between the servers and clients

(2) Round-Robin Algorithm: A round-robin approach to load balancing involves sending client requests to each server at the same time. The algorithm then guides the load balancer to return to the top position in the list.

This research established a connection between switch (s1) to 17 hosts (h1,h2,h3,h4,h5,h6…) ,where h1 h2 h3 acts as a client and h4 h5 h6 acts as server. Load is distributed based on algorithm used i.e. (Round Robin or Random).[6] We connect all the servers created to a web server through Http request all the server are located in same port i.e port case virtual IP is 10.1.1.100 .Virtual IP then sends request to Ryu Controller. Ryu controller consists of a code which perform the decided algorithm to distribute the requests and S1 as a switch which is responsible for transmitting and receiving the request between devices (Clients/Server).

V. RESULTS AND ANALYSIS:

This research work has performed load balancing using two algorithms i.e Round Robin Algorithm and Random Algorithm. Working of the load balancing was examined and comparative results of the same are displayed. As a controller we used Ryu controller and we have chosen Mininet emulator as an app environment.
Comparing the performance of two algorithms using Round Trip Time (RTT), has been utilized. Round Robin-based load balancing result of this operation using round trip for Host 1 and Host 15. The result of Round robin algorithm was displayed by the means of RTT between host 1 and host 15. The duration values are given in Figure 4 and Figure 5.

Round Robin Implementation

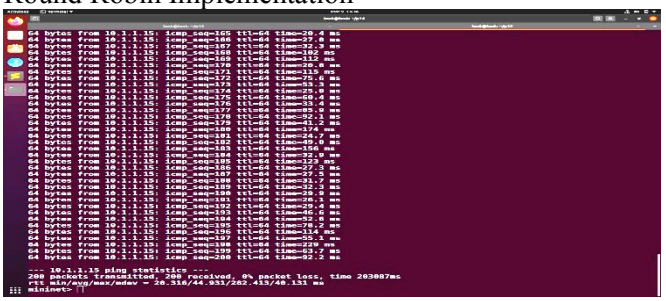

Fig.4 Load is more

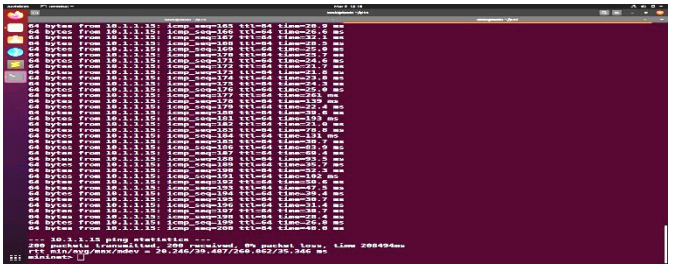

Fig.5 Load is less

Figure 4 and 5 shows how load balancing works under the round robin algorithm. Traffic and Load Balancing policy has also been closely monitored.

Random based Load balancing using Round-trip time results from pinging. The duration values are given in Figure 6 and figure 7.

Random Implementation

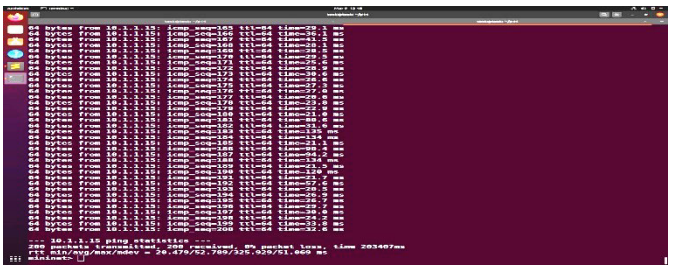

Fig.6 Load is more(Random Implementation)

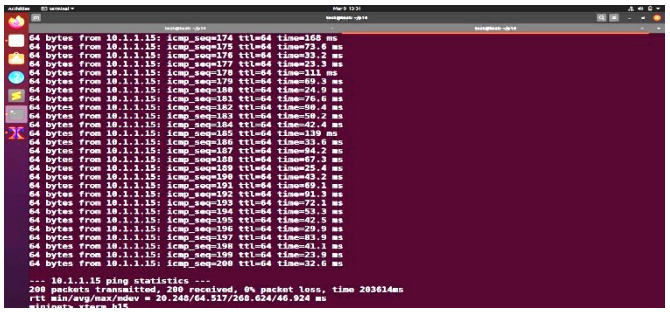

Fig.7 Load is less(Random Implementation)

Now, the efficiency of algorithms is analyzed by sending different number of packets

Table I shows the values(min/avg/max/mdev) at 100 packets: -

Table I 100 Packets Values

Algorithm	RTT Values (ms) (min/avg/max/mdev)			
	Min	Avg.	Max.	mdev
Random	20.31	32.93	256.72	33.13
Round Robin	22.24	32.48	270.86	35.4

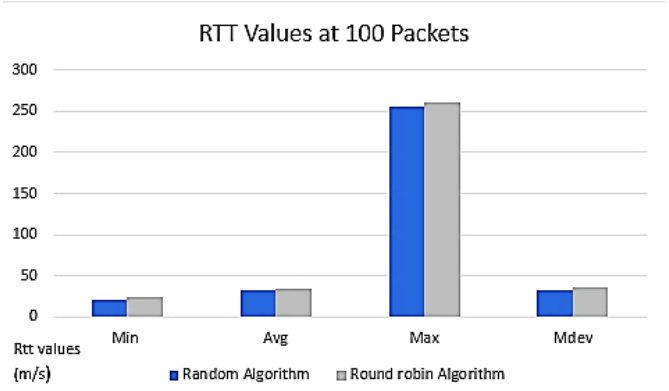

Fig. 8 Round Robin and Random Comparative RTT values of both algorithms

Figure 8 shows the Round Robin and Random Comparative RTT values of both algorithms are listed above. When the values in Table I are examined, Random algorithm is more efficient and has lower values Visible

Table 2 shows the values (min/avg/max/mdev) at 200 packets:-

Table 2 200 Packets Values

Algorithm	RTT Values (ms) (min/avg/max/mdev)			
	Min	Avg.	Max.	mdev
Random	20.31	44.93	282.41	40.13
Round Robin	20.24	29.48	260.86	35.4

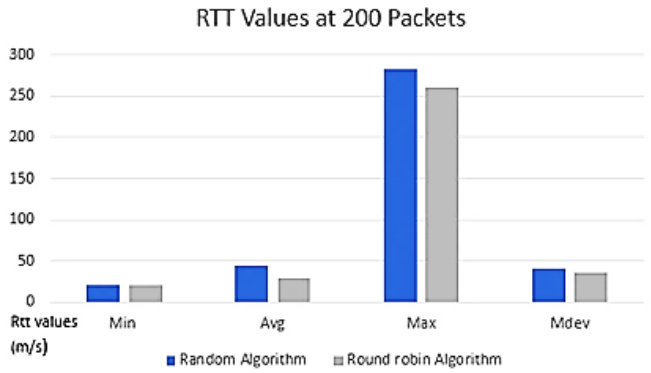

Fig. 9 Round Robin and Random Comparative RTT values

Figure 9 shows the Round Robin and Random Comparative RTT values of both algorithms are listed above. When the values in Table II are examined, Round Robin algorithm is more efficient and has lower values Visible. [8]

Table III shows values (min/avg/max/mdev) at 300 packets: -

Table III 300 Packets Values

Algorithm	RTT Values (ms) (min/avg/max/mdev)			
	Min	Avg.	Max.	mdev
Random	22.31	23.93	28.41	0.965
Round Robin	20.24	21.76	26.66	0.912

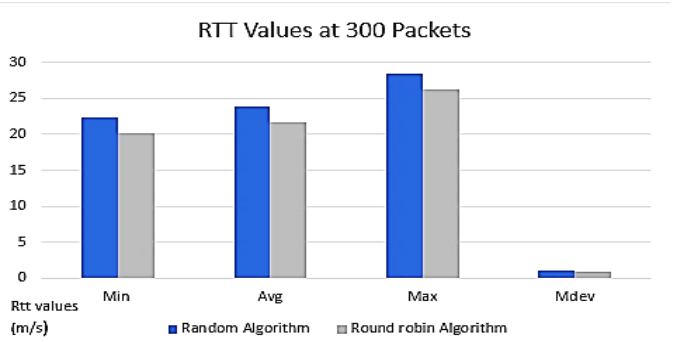

Fig. 10

Figure 10 shows the Round Robin and Random Comparative RTT values of both algorithms are listed below. When the values in Table III are examined, Round Robin is more efficient and has lower values Visible.

From the above analysis we conclude that when the load of packets was less Random Algorithm is efficient and when the load of packets were observed to be more Round Robin Algorithm was found efficient.

VI. RESULT IMAGES:

Figure 11 shows the initialization Of client and servers.

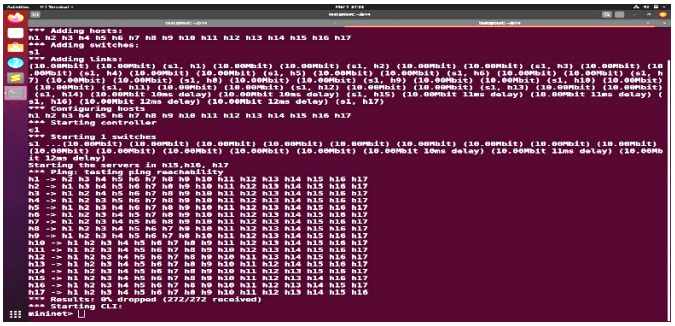

Fig. 11 Initialization of client and servers.

Figure 12 shows the image when Load Distributed Among Servers using ROUND ROBIN ALGORITHM

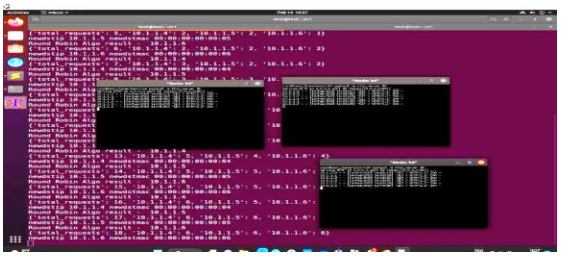

Fig. 12 Load Distributed Among Servers using ROUND ROBIN ALGORITHM

Figure 13 shows the image when Load Distributed Among Servers using RANDOM ALGORITHM

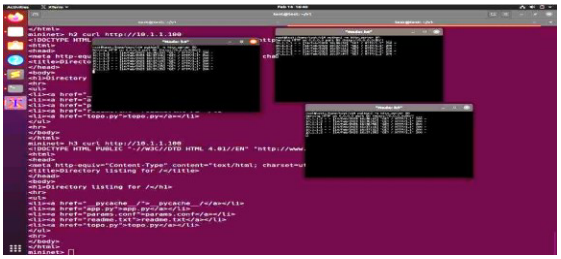

Fig. 13 Load Distributed Among Servers using RANDOM ALGORITHM

VII. CONCLUSION AND FUTURE WORK:

As observed from the data obtained, the RTT values of Round Robin algorithm is less as compared to Random Algorithm when the load was heavy. So from this it results that Round Robin Algorithm is more effective than Random Algorithm under the condition when the load was more. On the other hand, the RTT values of Round Robin Algorithm is more as compared to Random Algorithm when the load was less. So from this it results that the Random Algorithm was more efficient than Round Robin Algorithm under the condition when the load was less. Due to the complexity of network management, it is often difficult to implement a suitable architecture for managing networks. With the emergence of Software Defined Networking (SDN), it is now possible to solve this issue by providing an open and flexible framework for network management. This is very important for network administrators as it allows them to easily implement and manage their networks.

The paper discusses the design of the Software Defined Networking (SDN) and how it can be used to improve the performance of the network. The implementation of the round robin selection method can reduce the packet loss and provide better performance. It is also beneficial for reducing the overall network congestion and optimizing the utilization of the available resources After analyzing the collected data during the load balancing process, the researchers discovered that the Round Robin algorithm is more efficient than the Random algorithm when it comes to distributing the load.

VIII. ACKNOWLEDGEMENT

We would like to express our appreciation to Mrs. Sandhya Jadhav, the college's principal, for her support, and to the college administration for providing us with such a significant opportunity.

IX. REFERENCES:

[1] K. A. Jadhav, M. M. Mulla, and D. G. Narayan, “An Efficient Load Balancing Mechanism in Software Defined Networks,” in Proceedings - 2020 12th International Conference on Computational Intelligence and Communication Networks, CICN 2020, 2020. doi: 10.1109/CICN49253.2020.9242601.

[2] P. Bakonyi, T. Boros, and I. Kotuliak, “Classification Based Load Balancing in Content Delivery Networks,” in 2020 43rd International Conference on Telecommunications and Signal Processing, TSP 2020, 2020. doi: 10.1109/TSP49548.2020.9163470.

[3] Y. A. H. Omer, M. A. Mohammedel-Amin, and A. B. A. Mustafa, “Load Balance in Cloud Computing using Software Defined Networking,” in Proceedings of: 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering, ICCCEEE 2020, 2021. doi: 10.1109/ICCCEEE49695.2021.9429607.

[4] H. Sufiev and Y. Haddad, “A dynamic load balancing architecture for SDN,” in 2016 IEEE International Conference on the Science of Electrical Engineering, ICSEE 2016, 2017. doi: 10.1109/ICSEE.2016.7806104.

[5] B. Rukmini Bhat, N. S. Sneha, K. Bhat, C. C. Kamath, and C. Naik, “Improving the efficiency of software defined network through load balancing algorithms,” in Proceedings of the 3rd International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2021, 2021. doi: 10.1109/ICICV50876.2021.9388512.

[6] R. H. Hwang and H. P. Tseng, “Load Balancing and Routing Mechanism Based on Software Defined Network in Data Centers,” in Proceedings - 2016 International Computer Symposium, ICS 2016, 2017. doi: 10.1109/ICS.2016.0041.

[7] M. R. Belgaum, S. Musa, M. M. Alam, and M. M. Su’Ud, “A Systematic Review of Load Balancing Techniques in Software-Defined Networking,” IEEE Access, vol. 8, 2020, doi: 10.1109/ACCESS.2020.2995849.

[8] M. N. Abdulla and Z. A. Baqer, “Performance evaluation for load balancing in SDN controller,” Journal of Advanced Research in Dynamical and Control Systems, vol. 11, no. 4 Special Issue, 2019.

[9] X. Cui, Q. Meng, and W. Wang, “A Load Balancing Mechanism for 5G Data Centers,” in 2020 International Wireless Communications and Mobile Computing, IWCMC 2020, 2020. doi: 10.1109/IWCMC48107.2020.9148062.

[10] F. Pakzad, M. Portmann, W. L. Tan, and J. Indulska, “Efficient topology discovery in OpenFlow-based Software Defined Networks,” Comput Commun, vol. 77, 2016, doi: 10.1016/j.comcom.2015.09.013.

[11] M. ElGili Mustafa, “Load Balancing Algorithms Round-Robin (RR), Least-Connection and Least Loaded Efficiency,” International Journal of Computer and Information Technology, vol. 1, no. 1, 2017.

[12] P. Sanders, “Analysis of nearest neighbor load balancing algorithms for random loads,” Parallel Comput, vol. 25, no. 8, 1999, doi: 10.1016/S0167-8191(99)00040-X.

[13] J. Zhang et al., "Fast Switch-Based Load Balancer Considering Application Server States," in IEEE/ACM Transactions on Networking, vol. 28, no. 3, pp. 1391-1404, June 2020, doi: 10.1109/TNET.2020.2981977.

[14] F. Keti and S. Askar, "Emulation of Software Defined Networks Using Mininet in Different Simulation Environments," 2015 6th International Conference on Intelligent Systems, Modelling and Simulation, Kuala Lumpur, Malaysia, 2015, pp. 205-210, doi: 10.1109/ISMS.2015.46.

[15] Ryu SDN framework¶ (no date) RYU SDN Framework - Ryubook 1.0 documentation. Available at: https://book.ryu-sdn.org/en/html/ (Accessed: May 5, 2023).