International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

# Performance Analysis of different Load BalancingAlgorithms in SDN Based Data Center Networks

Sujayanth K Vishwakarma
Dept of CSE
RV College of Engineering Bengaluru,
India

Pavithra H
Dept of CSE
RV College of EngineeringBengaluru,
India

***Abstract*—In order to fulfil the demands of modern IT, it is necessary to address the many limitations of current networking infrastructures. Software Defined Networking (SDN) is emerging as the new networking strategy to get around these constraints. Existing Network Infrastructures use static switches, which results in a substandard utilization of the network resources, which is one of the main problems. The main issues with the present networking trend are the slow response and delays. In this research different load balancing algorithms for SDN based Data Center network is implemented to overcome these issues. The POX controller as SDN controller is used for implementation and Mininet software to emulate the network. The load balancing algorithms program is written in the Python programming language, which is also used to create the network topology. Finally, Siege HTTP load testing tool is used to test network performance. The testing focused on Quality of Service (QoS) parameters throughput, average response time and transaction rate. The results show that the Weighted algorithm outperforms all other considered algorithms.**

***Keywords—Load balancing algorithms, Software Defined Net-working, Data Center Network, POX Controller***

## I. INTRODUCTION

In the past 20 years, there have been significant changes in the network requirements, exponential growth in traffic, and the need for more challenging end-to-end objectives. The primary hosting infrastructures for Internet applications and services are data centers (i.e., social networks, Internet bank- ing, and multimedia content). In such data center networks, traditional load balancing algorithms use customized hard- ware devices to divide network traffic among various server copies. Although this method generally produces excellent performance, it is expensive and has a rigid configuration that cannot be dynamically changed in response to the status of the network or other information. Nowadays, software defined networking is used to move a lot of data. There is a lot of traffic, which has significantly grown. It is vital to address this kind of congestion issue so that networks can be made more effective. Decoupling the data and control planes is made possible by software defined networking. As a result, the network administrators use an external SDN controller to operate their network. It makes use of a unique kind of controller with the ability to manipulate the network's forwarding behavior. The SDN-depicted network architecture is very versatile, and the network administrator can benefit from the programmability of SDN-enabled switches.

### *A. Data Centre Network*

The Data Center Network (DCN), which connects all the data center resources, is crucial to the operation of a data cen- ter. To meet the expanding expectations of Software Defined Networks, DCNs must be scalable and effective enough to connect tens of thousands or even hundreds of thousands of servers (SDN). One of the main problems with data centers, regardless of their many configurations—whether they are physical data centers or virtual data centers—is load balancing. For improved performance metrics and traffic load balancing, an Openflow Data Center network architecture based on Soft- ware Defined Networks is deployed. Network load balancing must be utilized to expand the amount of available bandwidth, maximize throughput, and add redundancy. The capacity to balance traffic across several Internet connections is known as network load balancing. By spreading out the amount of bandwidth consumed by each LAN user over numerous connections, this capability balances network sessions like Web, email, etc., so improving the total amount of available bandwidth.

### *B. SDN Architecture*

A SDN architecture is made up of networking equipment such as routers and switches in the data plane. These un- derlying objects are nothing more than conventional forward- ing components devoid of any autonomous decision-making software. Using flow rules, the forwarding components are given instructions on what to do with the packets that have been received by the southbound interfaces (eg, OpenFlow). A control plane composed of control logic found in controllers and applications is responsible for controlling the items of the data plane. Application developers can use the northbound interface using an API provided by SDN controller. A north- bound interface conceptualizes the low-level instructions worn by a southbound interface while planning forwarding devices. From Fig. 1, it is clear that SDN is a bottom-up, three- layer design that includes an infrastructure layer, a control layer, and an application layer. An unambiguous and transpar- ent programming interface between network devices (routers,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

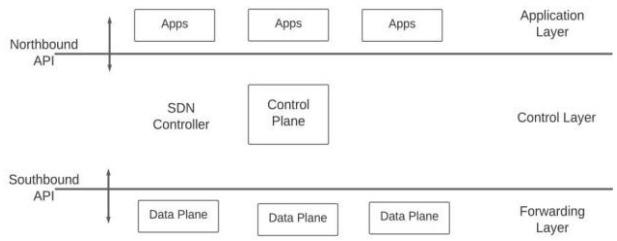


Fig. 1. SDN Architecture

switches), and the SDN controller, enables the separation between control plane and data plane to be seen. By breaking down the network control problem into digestible parts, this decoupling between the two planes is crucial for creating abstractions in networking and fostering network expansion and revolution. Control plane makes decisions about where and how to direct traffic as well as managing network topology. It has total control over the components of the forwarding layer. Applications in the application layer use the capabilities offered by the northbound interface to carry out network authority. Applications for load balancing and routing are both feasible. The most crucial component in this situation is a centralized logical controller (SDN controller), which will build the network configuration based on the rules and regulations of the network operator.

## II. Literature Review

In [1], the authors present a novel technique by implementing the load balancer in SDN, to improve throughput utilization. Mininet software is used to build the network environment. Moreover, the floodlight controller and Python programming language were used to implement. Two sce- narios were tested and evaluated, before and after the load balancer using TCP and UDP protocols. The results show a huge increment in network performance when using the load balancer with SDN. In [2] authors study a weighted response time algorithm is proposed. In the weighted response time algorithm, client requests are directed to the server with the lowest ratio of weight and response time values. The results of this study showed that a network that implemented a weighted response time algorithm requires less response time and pro- duces better throughput. In [3] authors implements Round Robin load balancing strategy and Random load balancing strategy using an OpenFlow switch. Mininet simulation results using round robin algorithm have led to higher throughput and decreased packet loss when compared to default random scheduling algorithm. In [4] authors focus on exploring a load balancing algorithm for a cloud platform where Virtual Network Functions are offered as a service. As the initial stage, they have used Weighted Round-Robin and Least Connection algorithms and conducted an experimental study to compare the performances of the two approaches. The results show that the weighted round robin algorithm performs better in terms of the workload distribution and average response time. In [5] authors implement and analyze the performance of Round- Robin load balancing strategy using a POX controller in a virtual network emulator called Mininet. Unlike the traditional hardware-based networking approach, this project implements load balancing with the help of software. In [6] authors discuss the various kinds of load balancing algorithms which can help in better utilization of resources and linear service delivery across multiple clients in an SDN environment and provides a brief analysis of some load balancing algorithms for services in a software defined network with the main concentration to balance the loads. The various load balancing algorithms are compared on the basis of different types of parameters. In [7] authors compare the performance of two load balancing algorithms such as flow-based load balancing algorithm and traffic pattern-based load balancing algorithm with distributed controllers' architecture. The Siege HTTP Load Balancing test tool is used to test the performance of the load balancing algorithms. The result shows that the flow- based load balancing algorithm minimizes response time and enhances transaction rate.

## III. Algorithms Applied to SDN

In this section, the different algorithm used in this researchare presented.

*1) Random Load Balancing Algorithm::*

Algorithm 1:
Input: Request, Set of available servers.Output: Request allocation to servers.Begin
Get list of servers
Whenever a new client request comes in
Pick a random server from the list of servers usingchosen server = random.choice(Available Servers) Assign the request to a random chosen server
End

Algorithm 1 shows the implementation for scheduling resource requests from client nodes and allocating servers to the client nodes using Random Load Balancing Algorithm. In a given set of available servers, the Load Balancer arbitrarily chooses one server node for forwarding the request from client nodes.

*2) Round Robin Load Balancing Algorithm:*

Algorithm 2:
Input: Request, Set of available servers.
Output: Efficient request allocation to servers.Begin
Get list of servers
Whenever a new client request comes in Pick a server from the list of servers using
lastServerindex=(self. lastServerindex +1) / len(AvailableServers)
chosen server = Available Servers [lastServerindex] Request is forwarded to each server in a cyclic fashionEnd

Algorithm 2 shows the implementation for scheduling resource requests from client nodes and allocating servers to the client nodes using Round Robin Load Balancing Algorithm. Round

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

Robin is the most widely used algorithm as it is easy to implement and understand. In here, each server node receives the requests from client nodes in a circular pattern. Load Balancer allocates the request to several servers in a round robin manner.

*3) Weighted Round Robin Load Balancing Algorithm:*

Algorithm 3:
Input: Request, Set of available servers.
Output: Efficient request allocation to servers.Begin
Get list of servers
Whenever a new client request comes in Pick a server from the list of servers using
lastServerindex =( lastServerindex +1)/ len(Available Servers)lastServerindex = lastServerindex + weight
chosen server = Available Servers [lastServerindex] Request is forwarded to each server in a cyclic fashionEnd

Algorithm 3 shows the implementation for scheduling resource requests from client nodes and allocating servers to the client nodes using Weighted Round Robin Load Balancing Algorithm. In this approach we can differentiate between heterogeneous servers. For Example, if Server 1 is having higher weight than Server 2, the algorithm will assign more requests to the server with a higher capability of handling load. Here the specific weight is assigned to each server. 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 the server with the higher specs will beassigned a greater number of requests.

*4) Least Connection Load Balancing Algorithm:*

Algorithm 4:
Input: Request, Set of available servers.Output: Request allocation to servers.Begin
Get list of servers
Whenever a new client request comes in
Pick a server with minimum connection from the list of available servers
Assign the request to chosen serverEnd

Algorithm 4 shows the implementation for scheduling resource requests from client nodes and allocating servers to the client nodes using Least Connection Load Balancing Algorithm. Consider a situation where, despite the fact that two servers in a cluster have identical specifications, one server can yet become overloaded far more quickly than the other. This can be the result of clients connecting to Server 2 staying connected for a lot longer than clients connecting to Server
1. As a result, Server 2's overall number of connections may increase, but Server 1's connections, which are affected by clients connecting and disconnecting more quickly, would essentially stay the same. As a result, Server 2's resources can deplete more quickly. The Least Connections algorithm fits these circumstances better. The number of active connections that each server has is taken into account by this algorithm.

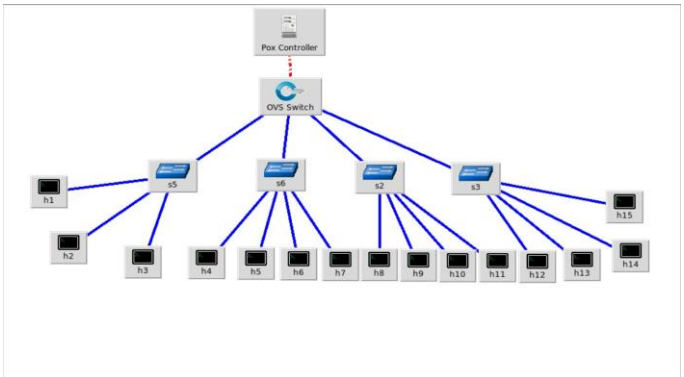


Fig. 2. The simulated 3-level tree Topology

The load balancer will seek to identify the server that has the fewest connections when a client tries to connect, and it will then assign the new connection to that server.

*5) Equal Load Balancing Algorithm:*

Algorithm 5:
Input: Request, Set of available servers.
Output: Efficient request allocation to servers.Begin
Get list of servers
Whenever a new client request comes in Pick a server from the list of servers using
lastServerindex = (lastServerindex +1)/ len(Available Servers)lastServerindex = lastServerindex + equal load
chosen server = Available Servers [lastServerindex] Request is forwarded to each server in a cyclic fashionEnd

Algorithm 5 shows the implementation for scheduling resource requests from client nodes and allocating servers to the client nodes using Equal Load Balancing Algorithm. In this approach we can differentiate between heterogeneous servers. This algorithm is similar to the Weighted Round Robin in a sense that the equal weight is assigned to each servers. The controller assigns the equal number of requests to all the servers in cyclic manner.

## IV. IMPLEMENTATION

The objective of the research is to implement different load balancing algorithms in the data center network and analyze the network in term of throughput, transaction rate and average response time. The load balancing experiment is carried out using the Ubuntu operating system. The network topology is generated using a Mininet and the SDN POX controller. The Network topology and the algorithm were developed using the Python programming language. The simulated 3-level tree topology is shown is figure 2 consists of 2 types of switches, Legacy switches at distribution level and OVS Switch which is connected to the Pox Controller. In order to create the data center topology, Mininet is linked to the Pox controller via the port 6633. The "Ping all" command is then used to determine whether all hosts in the configured network can be reached.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

Xming server is used to access the command prompt of each host of the topology. The hosts running the Simple HTTP Server application are depicted as servers and other hosts are depicted as clients. The clients send their request to the SDN Controller's virtual switch IP and Controller distributes the received requests to the servers based on the Load Balancing strategy. Curl command is used to generate the GET request from the client. Siege an HTTP Load Testing tool which sends number of requests from concurrent hosts is used to test network performance. The QoS parameters Average Response time, transaction rate and throughput are recorded for the analysis.

## V. RESULTS AND ANALYSIS

This section discusses the results obtained by simulating the load balancing techniques in SDN using Mininet emulator. The performance of the load balancing algorithms is compared using the QoS parameters average response time, throughput and transaction rate which are defined as follows.

Average Response Time: It is defined as Total time taken to respond during the selected period divided by the numberof responses in the selected period.

Throughput: It is defined as the amount of data packets traveling through the communication channel successfully in each period.

Transaction rate: The transaction rate is defined as the total number of flows processed by the controllers per second. The transaction rate is inversely proportional to the response time.

The Siege HTTP Load testing tool is used to test the performance of load balancing algorithm. The Average Re- sponse Time, Throughput and Transaction rate parameters values obtained from the tool are recorded and tabulated as shown in the Table 1.

The Comparison of average response time for different load balancing algorithms is shown in figure 2. It is observed that the weighted round robin algorithm has lowest average response time when compared to other algorithms. The Equal load algorithm and Round Robin algorithm has lesser average response time value than Least Connection algorithm. And the Random algorithm has the highest load balancing algorithm than other considered algorithm.

The fig 3 shows the Comparison of Transaction Rate for different load balancing algorithms. It is observed that the weighted round robin algorithm has the highest transaction rate when compared to other algorithms. The Equal load algorithm and Round Robin algorithm has more transaction rate value than Least Connection algorithm. And the Random algorithm has the lowest load balancing algorithm than other considered algorithm.

The Comparison of Throughput for different load balancing algorithms is shown in figure 4. It is observed that weighted round robin and equal load balancing algorithms have the

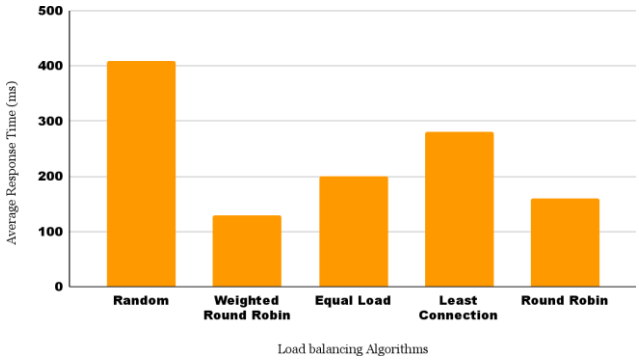


Fig. 3. Comparison of Average Response Time for different load balancingalgorithms

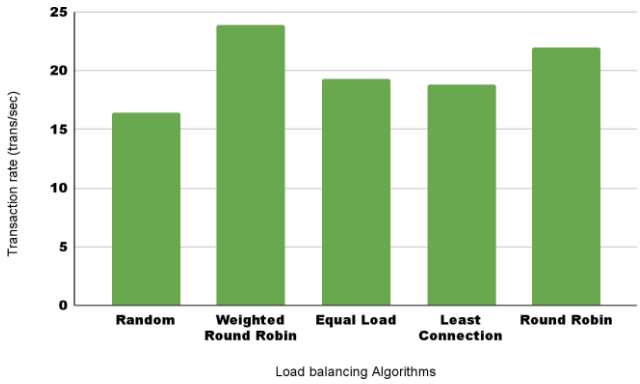


Fig. 4. Comparison of Transaction Rate for different load balancing algo-rithms

highest throughput values. The round robin algorithm has the throughput value lesser than weighted round robin and equal load balancing algorithms. The random algorithm and least connection algorithm have the lowest throughput values.

The figure 5 shows the graph of Average response time against different file size. The python script was written to generate

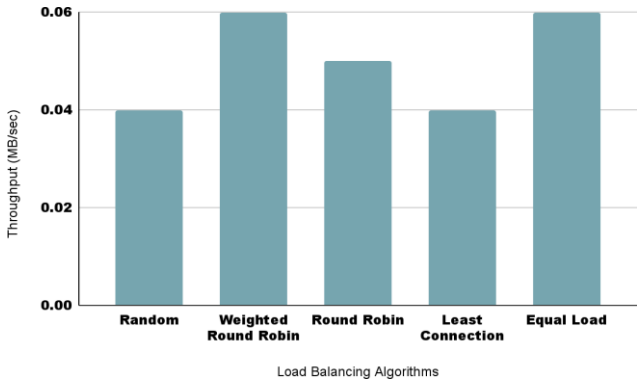


Fig. 5. Comparison of Throughput for different load balancing algorithms

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

TABLE I
COMPARISON OF AVERAGE RESPONSE TIME, TRANSACTION RATE AND CONCURRENCY FOR DIFFERENT LOAD BALANCING ALGORITHMS

| Load Balancing Algorithm | Average Response Time (sec) | Transaction rate (trans/sec) | Throughput (MB/sec) | Concurrency |
|---|---|---|---|---|
| **Weighted Round Robin** | 0.13 | 23.91 | 0.06 | 3.00 |
| **Round Robin** | 0.16 | 21.94 | 0.05 | 3.48 |
| **Equal Load** | 0.2 | 19.30 | 0.06 | 3.73 |
| **Least connection** | 0.28 | 18.82 | 0.04 | 5.29 |
| **Random** | 0.41 | 16.46 | 0.04 | 6.71 |

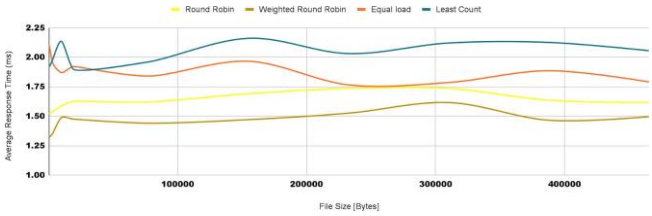


Fig. 6. Average Response Time vs File Size of different load balancing

200 GET requests from the client hosts requesting various sizes of data files. The 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 load balancing algorithms.

## VI. CONCLUSION

Traditional conservative network management presents more challenges, therefore SDN is an emergent architecture approach that addresses these problems. A very large-scale network switches can be regulated with the help of the open and adaptable SDN paradigm. The forwarding information base's centralization enables deterministic end-to-end routing calculations for each flow across the topology. SDN has gained a lot of traction, and as it moves closer to a time of modern data networks, it needs a highly flexible and clever load balancing system. In this project, the SDN-based POX controller is used to implement the load balancing approach. The Mininet emulator is used to replicate the SDN-based Data Center Network. The Pox controller simulates several load balancing techniques, such as random, round robin, weighted round robin, and least connections method. The performance of load balancing algorithms is assessed using the Siege HTTP load testing tool. The outcomes demonstrate that weighted round robin outperforms every other method presented. The assumption is that the channel bandwidth will remain constantand be error-free.

## VII. FUTURE WORK

In this paper, while implementing load balancing algorithms it is assumed that 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, Transaction rate and throughput was considered as performance metric, but there can be other parameters like bandwidth/channel utilization, Server utilization, ..etc. Here, the algorithms are implemented in Mininet emulator, in future it can be deployed in real time scenario and analyze the behavior.

## REFERENCES

[1] Y. A. H. Omer, A. B. A. Mustafa and A. G. Abdalla, "Performance Analysis of Round Robin Load Balancing in SDN," 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), 2021, pp. 1-5, doi: 10.1109/ICCCEEE49695.2021.9429662.

[2] H. Nurwasito and R. Rahmawati, "Weighted Response Time Algorithm for Web Server Load Balancing in Software Defined Network," 2021 3rd International Conference on Electronics Representation and Algorithm (ICERA), 2021, pp. 143- 148, doi: 10.1109/ICERA53111.2021.9538792.

[3] K. A. Jadhav, M. M. Mulla and D. G. Narayan, "An Effi- cient Load Balancing Mechanism in Software Defined Networks," 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN), 2020, pp. 116- 122, doi: 10.1109/CICN49253.2020.9242601.

[4] M. D. B. P. Senevirathne, W. H. Rankothge, N. D. U. Gamage, M.
M. T. R. Ariyawansa, H. G. H. Dewwiman and S. A. A. Suhail, "An Expermental Study on Load Balancing for a Software Defined Network based Virtualized Network Functions Platform," 2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communica- tion Conference (IEMCON), 2021, pp. 0092-0097, doi: 10.1109/IEM- CON53756.2021.9623144.

[5] K. Rishabh, K. Angadi, K. Chegu, D. S. Harikrishna and S. Ramya, "Analysis of Load Balancing Algorithm in Software Defined Network- ing," 2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS), 2017, pp. 1-4, doi: 10.1109/CSITSS.2017.8447852.

[6] Anish Ghosh, Mrs. T. Manoranjitham "A study on load balancing techniques in SDN" International Journal of Engineering and Technology(IJET),vol 7. pp. 174-177 2018

[7] Prabakaran, Senthil Ramar, Ramalakshmi. (2021). Software Defined Net- work: Load Balancing Algorithm Design and Analysis. The International Arab Journal of Information Technology. 18. 10.34028/iajit/18/3/7.

[8] Smriti Bhandarkar, Kotla Amjath Khan "Load Balancing in Software- defined Network (SDN) Based on Traffic Volume" Advances in Computer Science and Information Technology (ACSIT) 2015 Volume 2, Number7 pp 72-75

[9] L. Liu et al., "An SDN-based Hybrid Strategy for Load Balancing in Data Center Networks," 2019 IEEE Symposium on Computers and Communi- cations (ISCC), 2019, pp. 1-6, doi: 10.1109/ISCC47284.2019.8969673.

[10] C. Hu, Z. Huang and Z. Lin, "An information transport dynamic load balancing policy based on software defined mobile networks," 2018 International Conference on Information and Computer Technologies (ICICT), 2018, pp. 83-86, doi: 10.1109/INFOCT.2018.8356845

[11] A. K. Arahunashi, G. G. Vaidya, N. S and K. V. Reddy, "Implementation of Server Load Balancing Techniques Using Software-Defined Network- ing," 2018 3rd International

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 07, July-2022

Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), 2018, pp. 87-90, doi: 10.1109/CSITSS.2018.8768754.

[12] D. Perepelkin and V. Byshov, "Visual design environment of dynamic load balancing in software defined networks," 2017 27th International Conference Radioelektronika (RADIOELEKTRONIKA), 2017, pp. 1-4, doi: 10.1109/RADIOELEK.2017.7936643.

[13] Senthil Prabakaran, Ramalakshmi Ramar "Software Defined Network: Load Balancing Algorithm Design and Analysis" The International Arab Journal of Information Technology, Vol. 18, No. 3,pp. 312-318 May 20

[14] Mohammed Moin Mulla, M. M. Raikar, M. K. Meghana, Nagashree S. Shetti and R. K. Madhu, "Load Balancing for Software-Defined Networks", Emerging Research in Electronics Computer Science and Technology, pp. 235-244, 2019.

[15] Ma Jiefei, W. Rankothge et al., "An overview of a load balancer architecture for vnf chains horizontal scaling", IEEE CNSM, 2019.

[16] M. Ramona Trestian, OFLoad: An OpenFlow-based Dynamic Load Balancing Strategy for Datacenter Networks, November 2017.

[17] W. Rankothge, Helena Ramalhinho and Jorge Lobo, "On the Scaling ofVirtualized Network Functions", IEEE IM, 2019