SDN-Based Load Balancing to Achieve Energy Efficiency in Enterprise Networks

Shabana Muhammad, Karrar Muhammad and Muhammad Imran Majid

October 23, 2024

SDN-based load balancing to achieve energy efficiency in enterprise networks

Shabana Muhammad
Department of Electrical Engineering, IoBM
Karachi, Pakistan
shabana.sm23@gmail.com

Karrar Muhammad Bhutto
Department of Electrical Engineering, IoBM
Karachi, Pakistan
karrarmuhammadbhutto@gmail.com

Dr. M. Imran Majid
Department of Electrical Engineering, IoBM
Karachi, Pakistan
imran.majid@iobm.edu.pk

Abstract— Due to a lack of proper traffic management framework traditional networks suffer from underutilization of devices and network congestion. High power consumption and the carbon footprint of traditional networks is another major problem. Here equivalent distribution of resources using load balancing and network power management are essential to optimizing network resource usage and performance. The software-defined network (SDN) is a promising solution for network administrators to address these issues. Software-defined network-based load balancing and energy awareness for enterprise networks and explore how these both correlate is discussed in this paper. The main objective is to minimize energy consumption and optimize resource usage through proper load distribution. Round-robin and hashing static load balancing techniques are used for the analysis. TCP and UDP protocols are used to generate traffic and evaluate network performance. Using the Iperf tool, transfer (KB), jitter (ms), and packet loss (%) parameters of traffic flow parameters are observed. Average RTT decreased by 53.2% with round robin and 12.8% with hash-based and overall jitter by 0.9% CPU utilization (%CPU) is evaluated for energy consumption before and after load balancing. Results show better performance of round-robin load balancing. Even distribution of load via round-robin reduces %CPU and also idle time (<90%).

Keywords— Software-defined network (SDN), Energy awareness, Load balancing, SDN control plane, Controller policies

I. Introduction

Networking and communication demands of current networks, including the Internet of Things (IoT), enterprises, education backbone, and data centers, have varying software and hardware requirements. A growing number of heterogeneous devices, connections, and applications with varying configuration requirements increase the complexity of such networks, as shown in "Fig. 1" [1]. New devices and connections also change existing infrastructure, network functionality, bandwidth requirements, network traffic patterns, and end-user experience, increasing network complexity and data processing. One solution to these challenges is enhancing network adaptability to meet dynamic demands. Furthermore, IoT networks are constrained by security, scalability, and energy efficiency challenges. Currently, software-defined network (SDN) technology addresses these technical problems of traditional networks. The decoupling of data and control layers enables programmability and enhances network functionality and resource management using protocols. With the virtualization feature in software-defined networks, multiple network layers are possible without changing the physical structure, adjusting network performance based on real-time statistics. Traffic management techniques further help satisfy the quality of service [2].

SDN networking framework also promises to address issues related to high energy requirements. Current networks are environmentally and economically inefficient due to higher CO_2 emissions and operating costs. One study estimates that the information, communication, and technology sector is responsible for 2-4% of global greenhouse gas emissions [3]. Hence, researchers and network administrators strive for a trade-off between energy-efficient network operations and performance. Dynamic traffic management and energy-aware routing algorithms enable SDN to save energy [2].

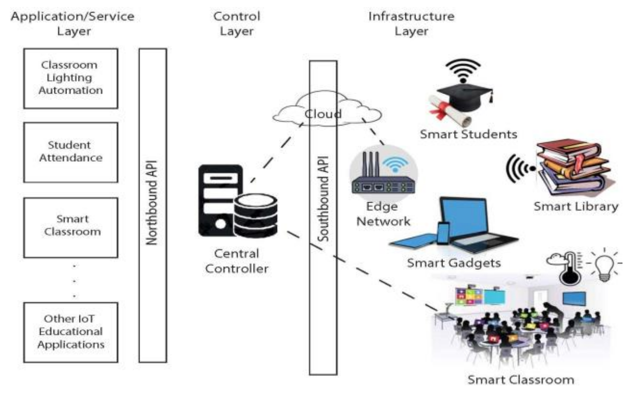

Fig. 1. SDN-based IoT Architecture, and connectivity layout for application in Education Sector [1].

II. Literature review

A. Software Defined Network (SDN)

SDN infrastructure provides a centralized control of the network, where control and data lie on separate layers. The control layer is responsible for path flow decisions and the data plane decides transmission of packets. The separation of the control plane and data forwarding plane and logical centralized control of network devices have attracted a huge research interest in SDN. SDN improves resource management by separating data and control layers. Software-based programmability and flexible control of network devices have promoted SDN deployment in diverse platforms ranging from radio networks to data centers and educational institutes [4], [5].

1) *SDN Control plane*

SDN architecture transformation from the traditional network rests on four key features, including decoupling of the control and the data plane, centralized controller, open interfaces between network elements in control and data planes, and network programmability [6]. The control plane consists of one or multiple controllers and many distributed components called switches.

The controller is software with logically centralized control of the entire network and is independent of the physical hardware. It is responsible for managing all network events, such as deciding, installing, removing, and modifying packet forwarding and routing rules and instructing switches [7]. Controller programs these distributed devices using northbound and southbound OpenFlow interfaces, as shown in “Fig. 2”. Switches lie in the data plane and only perform packet forwarding functions, which are decided by the centralized controller. With programmable interfaces, the controller can enhance its functionalities, authorize some applications to manage network tasks, and set control constraints on network devices [8].

The controller sets rules for traffic segregation, distribution, routing, scheduling, and forwarding. With these features, the controller can forward all traffic to specific links and devices and put others in off mode using load-balancing techniques, which minimizes energy consumption. The controller can also decide routing path optimization based on the energy levels of each node [9].

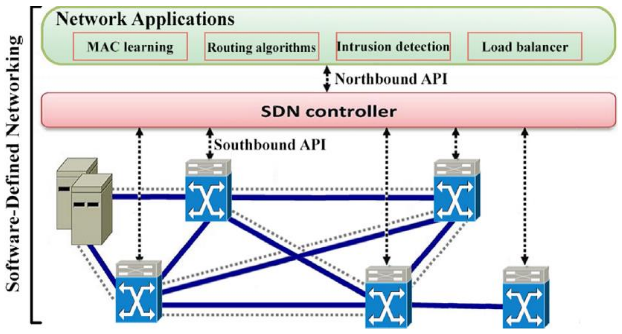

Fig. 2. A simplified view of SDN architecture [6].

2) *SDN Control Plane Classification*

SDN control plane is classified as physically centralized, physically distributed, logically centralized, and logically distributed, as shown in “Fig. 3”. These classifications have specific drawbacks and advantages. Physically centralized architectures have a single controller that manages the whole network. It suffers from scalability and reliability issues and struggles to ensure performance as network size grows. Physically distributed architectures address scalability, reliability, single point of failure, and performance. Logically centralized architectures consist of a main controller and backup controllers. The main controller has the decision-making authority in the network. Backup controllers take over the network charge and manage resources in case of failure of the main controller. Logically centralized infrastructure works best for intra-domain management of networks. However, logically centralized controllers fail to manage heterogeneous networks. Logically distributed control extends compatibility for heterogeneous architectures, such as IoT and backbone networks [10].

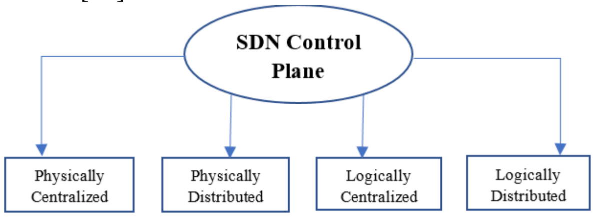

Fig. 3. SDN Control Plane Classification

3) *Controller Customization Policies in SDN*

Customization policies in an SDN controller aim to improve network quality of service performance, enhance security, and reduce network complexity, data processing, and overall operating costs. Customization policies for the SDN controllers that improve performance include application awareness, identity awareness, security awareness, and energy awareness [11], [12], [13], [14], as shown in “Fig.4”. These are discussed in detail as follows.

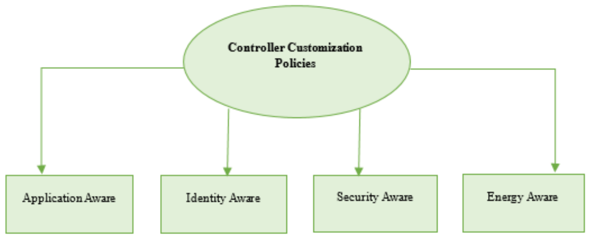

Fig. 4. Controller customization policies in SDN.

a) Application Awareness

Since different applications have different bandwidth, latency, and other quality of service requirements, network operators map and manage traffic and performance specific to these requirements. Traffic is differentiated according to traffic identifiers, such as application name, application category, device class, and media type. The controller manages traffic flow based on bandwidth, latency, and specific traffic identifiers [11], [12].

b) Identity Awareness

The controller manages traffic based on a person (or a machine) that takes part in a specific communication. Network infrastructures with identity awareness accommodate network users based on pre-defined privileges. The chief purpose of assigning privileges is to ensure security or quality of service for each user category. Traffic identifiers may include personal ID, name, or groups [13].

c) Security Awareness

Security awareness establishes access constraints on communications between end users or devices to prevent network intrusion and security attacks. Security awareness-based architectures can establish an alternative path, route traffic to specific routes to prevent traffic congestion and detect DDoS attacks. Traffic management and security policies are specified in the network controller [14].

d) Energy Awareness

An SDN controller with an energy awareness policy adjusts network energy consumption to be proportionate to the traffic load. For instance, the controller can switch off some low-load network devices and preserve energy resources for other devices and applications. It is a sustainable solution for rapidly advancing technologies like cloud computing where energy consumption increases with more users and data processing. The energy-saving approach optimizes network performance and reduces operating costs [15].

B. Controller Policies for Energy Awareness

High energy consumption and load imbalance are among the major problems as the network grows in complexity. Network complexity increases energy consumption and overall network operating costs, and both these constraints lead to significant design and network optimization challenges.

SDN presents an opportunity to optimize network energy and balance load simultaneously. Shortest-path criteria for routing network traffic achieve energy efficiency by minimizing the engagement time of nodes and end-to-end delay [16].

Assefa and Özkasap [17] divide energy awareness policies into software-based and hardware-based categories. They further classify these broad categories into sub-categories. Software-based energy-saving approaches include traffic awareness, end system awareness, and rule placement. Traffic-aware policies tweak network elasticity, topology awareness, active/inactive time, and queuing engineering. End system awareness is achieved by utilizing system resources efficiently via relieving underutilized servers and engaging only some limited servers for the tasks. Rule placement refers to setting up rules to minimize memory utilization and the number of active links. Hardware-based policies optimize the power consumption of network devices (especially forwarding switches) through specific configurations and minimize information storage requirements.

Some research attempts find that changing device configuration parameters like the number, type, and location of forwarding entries and the number of tables used for these entries can alter data storage requirements and consequently total power consumption [18].

Neama et al. [19] propose an Energy Efficient Integral Routing (EEIR) algorithm to optimize energy consumption. EEIR algorithm optimizes the number of active links and switches of unnecessary links while maintaining route feasibility for network demands. Their results highlight the potential of EEIR in saving power up to 44.42%. Meeting traffic demands with a minimum number of active links reduces energy consumption and saves energy by up to 60% in another study [20]. They use traffic engineering to minimize overall power consumption.

He et al. [21] present a joint approach to save energy and balance load in data centers. They manage network traffic and balance load with the maximum available bandwidth multipath routing technique. They save energy by scheduling traffic flows and minimizing active switches and links. Galán-Jiménez et al. [22] address energy efficiency and load balancing in IP/SDN hybrid networks using Hybrid Spreading Load Algorithm (HSLA). Their algorithm balances network traffic and power consumption by minimizing link usage.

C. Load Balancing in SDN

SDN paradigm introduces network scalability to accommodate a growing number of users and devices with programmability in controllers. However, large-scale networks also generate huge data, which leads to controller overloading. Network overloading or load imbalance is a major problem in networking. Network overloading and improper load distribution can exhaust the network, compromise network performance, and even introduce delay.

To address these problems, load balancers or load balancing policies are fused into the network management plane. Load balancing enhances adjustability in the SDN network to distribute workload efficiently. With load balancing policies, the network adapts to changing traffic patterns and flows, prevents network overloading, and optimizes resource allocation. The main purpose of a load balancer is to boost network performance, maintain stability, build fault tolerance, and incorporate adjustability for future modifications [23].

a) Types of Load Balancing Algorithms

There are primarily two load-balancing algorithms in software-defined networks: static and dynamic. Some research studies and experiments have attempted to combine the best features of static and dynamic and devised a hybrid load-balancing algorithm [24].

- *Static Load Balancing*

The static algorithm divides traffic load within the servers equally. It is suitable for networks with low load variation and the algorithm is aware of available resources.

- *Dynamic Load Balancing*

The dynamic algorithm searches for servers with low loads and engages them accordingly. It is based on the system's real-time statistics and network traffic to adjust the load. The network load is distributed based on the current traffic pattern.

- *Hybrid Load Balancing*

Hybrid load balancing addresses shortcomings of static and dynamic algorithms and combines their best features to manage and distribute traffic flow efficiently and improve network performance.

D. Performance Evaluation Metrics

The literature review highlights that traffic engineering substantially affects overall network energy consumption [20], [21], [22]. For instance, underutilized or overburdened network resources have varying energy requirements. Uneven load distribution can affect network performance. Thus, we evaluate the correlation of proper load balancing on energy efficiency. Our objective is to identify whether optimal resource utilization can minimize energy consumption. The effect of load distribution is analyzed using round-trip time (RTT), transfer, bitrate, jitter, and packet loss parameters. CPU utilization statistics are used to analyze and gauge overall energy consumption.

III. METHODOLOGY

An SDN-based enterprise network is created in Mininet emulator 2.3.1b4, installed in Ubuntu 20.04 LTE VM Box. Mininet's CLI is used to develop manual topologies and analyze two network architectures. The first architecture consists of two hosts (h1, h2) with IP addresses '10.0.0.1' and '10.0.0.2' connected to two distributed switches (S1, S2) and a controller with IP address '127.0.0.1' at port 6633 (see Fig. 5). Later, the network is scaled up by connecting a third switch (S3), which helps evaluate the impact on load balancing and energy consumption (see Fig. 6). The load balancer class is inherited from ryu-manager 4.34 to configure the RYU as a load balancing server. RYU controls the entire network framework and manages packet flow based on pre-defined policies or dynamic traffic patterns. Switches are configured to communicate with the controller for packet flow using the default features of OpenFlow 1.3. RTT is observed to evaluate the communication connection status between hosts. The "Iperf" tool is used to create TCP/UDP traffic flows and monitor network performance in a client/server model using performance metrics, i.e., transfer, bitrate, jitter (ms), and packet loss (%). Lastly, CPU utilization is analyzed to observe energy consumption using round-robin and hash-based load-balancing techniques.

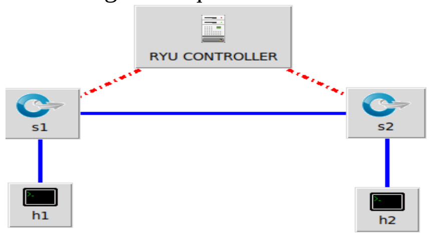

Fig. 5. Architecture with two hosts, switches, and a controller.

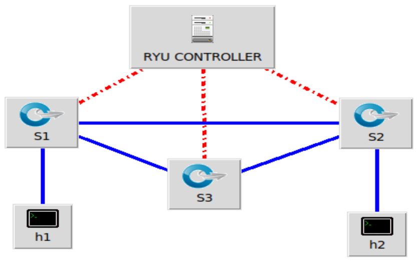

Fig. 6. Architecture with two hosts, three switches, and a controller.

IV. RESULTS AND DISCUSSION

A. Round trip time evaluation before and after load balancing

Load balancing improves network response time and reduce latency. RTT and ping time are evaluated to check network response time using the 'ping' command. RTT is the total time taken by a packet from sender to receiver and vice versa. Ping displays four values of RTT: minimum, average, maximum, and median deviation. Average RTT values for 4 random packet samples is analyzed, as given in Table 1 and Table 2. "Equation 1" and "Equation 2" are used to measure efficiency in RTT response before and after load balancing. Results show that the average RTT for 60 packets decreases by 53.2% with round robin and 12.8% with the hash-based approach for the first architecture model. However, average RTT increases in both load-balancing techniques by 2.8% and 16.8% for the second model. Zero packets are lost in all cases. Given a confidence level of 95%, the confidence interval for all cases is less than 1.

TABLE 1 AVERAGE RTT BEFORE LB

Before LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model				
Avg. RTT (ms)	0.091	0.102	0.068	0.156
Average standard deviation	0.2ms			
Second model				
Avg. RTT (ms)	0.065	0.084	0.070	0.069
Average standard deviation	0.04 ms			

TABLE 2 AVERAGE RTT AFTER LB

After LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model (round robin)				
Avg. RTT (ms)	0.143	0.096	0.074	0.073
Average standard deviation	0.04ms			
Second model (round robin)				
Avg. RTT (ms)	0.082	0.111	0.088	0.067
Average standard deviation	0.0645ms			
First model (hash-based)				
Avg. RTT (ms)	0.065	0.267	0.106	0.136
Average standard deviation	0.306ms			
Second model (hash-based)				
Avg. RTT (ms)	0.083	0.106	0.090	0.083
Average standard deviation	0.0665ms			

%decrease = {(Avg. RTT before LB-Avg. RTT after LB)/ Avg. RTT before LB} *100 (1)

%increase = {(Avg. RTT after LB-Avg. RTT before LB)/ Avg. RTT after LB} *100 (2)

B. Analysing TCP and UDP outputs before and after load balancing

The impact of load balancing in an SDN-based enterprise network is evaluated using TCP and UDP protocols. The throughput was slightly lower due to the overhead from load-balancing logic and flow rule installation, including round-robin and hash-based approaches. Throughput was higher without load balancing but at the cost of uneven traffic distribution, as some paths are underutilized while others handle more load. The ultimate choice of using a load-balancing technique is a tradeoff between various performance parameters. Table 3 and Table 4 show TCP traffic flow before and after load balancing. Table 5 and Table 6 show UDP traffic flows. UDP traffic statistics show load balancing slightly reduced jitter (0.9%). Overall, TCP bitrate is higher in round-robin compared to hash-based load balancing, whereas UDP performance remains unchanged. Furthermore, TCP has better performance in terms of lower packet delay and packet loss than UDP in both load-balancing approaches.

TABLE 3 TCP OUTPUT BEFORE LOAD BALANCING

TCP before LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model				
Transfer (GBytes)	12.1	21.8	38.9	57.0
Bitrate (Gbits/s)	10.4	9.35	8.36	8.16
Second model				
Transfer (GBytes)	13.4	23.4	47.7	68.8
Bitrate (Gbits/s)	11.5	10.1	10.2	9.86

TABLE 4 TCP OUTPUT AFTER LOAD BALANCING

TCP after LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model (round robin)				
Transfer (GBytes)	11.8	22.9	47.0	53.0
Bitrate (Gbits/s)	10.1	9.82	10.1	7.59
Second model (round robin)				
Transfer (GBytes)	11.8	23.1	36.1	54.7
Bitrate (Gbits/s)	10.2	9.92	7.74	7.83
First model (hash-based)				
Transfer (GBytes)	7.55	18.1	36.3	47.4
Bitrate (Gbits/s)	6.48	7.79	7.80	6.78
Second model (hash-based)				
Transfer (GBytes)	8.82	12.0	25.4	40.1
Bitrate (Gbits/s)	7.58	5.16	5.46	5.73

TABLE 5 UDP OUTPUT BEFORE LOAD BALANCING

UDP before LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.04	1.05	1.05	1.05
Second model				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.05	1.05	1.05	1.05

TABLE 6 UDP OUTPUT AFTER LOAD BALANCING

UDP after LB				
Interval	0.0-10	0.0-20	0.0-40	0.0-60
No. of packets	10	20	40	60
First model (round robin)				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.04	1.05	1.05	1.05
Second model (round robin)				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.05	1.05	1.05	1.05
First model (hash-based)				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.04	1.05	1.05	1.05
Second model (hash-based)				
Transfer (MBytes)	1.25	2.50	5.00	7.50
Bitrate (Mbits/s)	1.05	1.05	1.05	1.05

C. *CPU energy consumption*

Network devices and servers consume power while operating, computing, and storing information. The chief performance goal is to utilize these resources resourcefully in their active operating state, otherwise, idle state power consumption is a waste. Energy consumption is evaluated based on the information that the CPU is the core component to estimate the overall power consumption of a server, CPU utilization and energy consumption are linearly linked, and servers consume 60-70% of their peak power in an idle state [25] [26]. The main objective is to minimize CPU utilization (%CPU) and idle time (id). If id>90%, CPU capacity is underutilized. Load balancing helps make a tradeoff between CPU utilization and id to minimize overall power consumption. %CPU is evaluated using Mininet CLI using the command '`top -p $(pgrep ovs-vswitchd)`'. Table 7 and Table 8 show CPU statistics before and after load balancing. CPU utilization is lower and stable after load balancing. Similarly, the id statistic in the round-robin is lower than the hash-based technique.

TABLE 7 CPU UTILIZATION BEFORE LB

Before LB		
	First model	Second model
Idle time (id)	90.4	53.3
%CPU	1.3	0.7

TABLE 8 CPU UTILIZATION AFTER LB

After LB				
	First model (round robin)	Second model (round robin)	First model (hash-based)	Second model (hash-based)
Idle time (id)	58.4	84.5	92.0	90.9
%CPU	0.3	0.3	0.3	0.3

V. CONCLUSION

In this paper, traffic management issues and the high energy consumption of traditional networks are addressed using the SDN framework. Round-robin and hash-based techniques were used to manage traffic among network devices and evaluate the impact of load balancing on energy efficiency. Simulation results using TCP and UDP traffic flow highlight that efficient load balancing can minimize energy consumption and network congestion without packet loss. Overall, results show round robin has better performance than the hash-based load balancing technique.

REFERENCES

[1] M.I. Majid et al., "NFV and Secure Cognitive SDN for Educational Backbone Network Deployment: Cognitive Enabled Routing in SDN", Published in *Spectrum and Power Allocation in Cognitive Radio Systems*, K. Suriyan, R. Dhaya, R. Nagarajan, and A. Karthick, IGI Global, 2024, ch.10, pp. 300.

[2] B. G. Assefa and Ö. Özkasap, "A Survey of Energy Efficiency in SDN: Software-based Methods and Optimization Models", *Journal of Network and Computer Applications*, vol. 137, pp.127–143, 2019, https://doi.org/10.1016/j.jnca.2019.04.001

[3] I. Godlovitch, A. Louguet, D. Baischew, M. Wissner, and A. Pirlot. "Environmental impact of electronic communications". Study for BEREC. Accessed: June 29, 2024. [Online]. Available: https://www.berec.europa.eu/sites/default/files/files/document_register_store/2022/3/BoR%20%2822%29%2034_External%20Sustainability%20Study%20on%20Environmental%20impact%20of%20EC.pdf

[4] A. Hodaei and S. Babaie, "A Survey on Traffic Management in Software-Defined Networks: Challenges, Effective Approaches, and Potential Measures", *Wireless Personal Communications*, vol.118, pp.1507–1534, 2021.

[5] F. Bannour, S. Souihi, and A. Mellouk, "Software-defined networking". Published in *Extending Sdn Control to large-scale networks*, vol. 2. London, Hoboken, NJ: ISTE Ltd; John Wiley & Sons, Inc, 2022.

[6] O. David, P. Thornley, and M. Bagheri, "Software Defined Networking (SDN) for Campus Networks, WAN, and Datacenter," *International Conference on Smart Applications, Communications and Networking (SmartNets)*, Istanbul, Turkiye, 2023, pp. 1-8, doi:10.1109/SmartNets58706.2023.10215722.

[7] O. Padon, N. Immerman, A. Karbyshev, O. Lahav, M. Sagiv, and S. Shoham, "Decentralizing SDN Policies", in *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2015, pp. 663-676.

[8] J. Xie, D. Guoa, Z. Hua, T. Qua, and P. Lv, "Control plane of software defined networks: A survey", *Computer Communications*, pp. 1-10, 2015.

[9] M.F. Tuysu, Z. K. Ankarali, and D. Gözüpek, "A survey on energy efficiency in software defined networks", *Computer Networks*, vol. 113, pp.188-204, 2017.

[10] F. Bannour, S. Souihi, and A. Mellouk, *Software-defined networking. volume 2: Extending Sdn Control to large-scale networks Vol 2*. London, Hoboken, NJ: ISTE Ltd; John Wiley & Sons, Inc, 2022.

[11] M. H. Al-Bowarab, N. A. Zakaria, and Z.Zainal-Abidin, "Load Balancing Algorithms in Software Defined

Network", *International Journal of Recent Technology and Engineering (IJRTE)*, vol.7, 2019.

[12] F.N. Nife and Z. Kotulski, "Application-Aware Firewall Mechanism for Software Defined Networks", *Journal of Networking System Management*, vol. 28, pp.605-626, 2020.

[13] L. Polčák, L. Caldarola, A. Choukir, D. Cuda, M. Dondero, D. Ficara, B. Frankova, M. Holkovic, R. Muccifora, and A. Trifilo, "High-Level Policies in SDN", in *E-Business and Telecommunications: 12th International Conference, Colmar, France,* 2016, doi:10.1007/978-3-319-30222-5_2

[14] V. Varadharajan, K. Karmakar, U. Tupakula and M. Hitchens, "A Policy-Based Security Architecture for Software-Defined Networks," *IEEE Transactions on Information Forensics and Security*, vol. 14 (4), pp. 897-912, April 2019, doi: 10.1109/TIFS.2018.2868220

[15] V. Verma and M. Jain, "Energy-efficient Techniques in SDN: Software, Hardware, and Hybrid Approaches", *Philippine Journal of Science,* vol. 153(1), pp.1-22, February 2024.

[16] Rego, A., Sendra, S., Jimenez, J. M., & Lloret, J. "OSPF routing protocol performance in Software Defined Networks", in *Fourth International Conference on Software Defined Systems (SDS),* 2017, doi:10.1109/sds.2017.7939153

[17] B. G. Assefa and Ö. Özkasap, "A survey of energy efficiency in SDN: Software-based methods and optimization models", *Journal of Network and Computer* Applications, vol. 137, pp. 127–143, 2019.

[18] R. Gandotra and Levi Perigo, "Comparing Energy Efficiencies of SDN Hardware Based on Forwarding Configurations", in *29th International Conference on Computer Communications and Networks (ICCCN),* 2020, doi: 10.1109/ICCCN49398.2020.9209678

[19] G. N. Neama and M. K. Awad, "An Energy Efficient Integral Routing Algorithm for Software-Defined Networks", in *IEEE 86th Vehicular Technology Conference (VTC-Fall),* 2017 doi:10.1109/vtcfall.2017.8288351

[20] A. Fernández-Fernández, C. Cervell´o-Pastor, and L. Ochoa-Aday, "Achiveing Energy Efficiency: An Enrgy-Aware Approach in SDN", in *IEEE 59th Global Communications Conference,* 2016, doi:10.1109/GLOCOM.2016.7841561

[21] Y. He, Z. Lu, J. Lei, S. Deng, and X. Gao, "Joint optimization of energy saving and load balancing for data center networks based on software defined networks", *Concurrency and Computation: Practice and Experience,* vol.33(9), 2020, doi:10.1002/cpe.6134

[22] J. Galán-Jiménez, M. Polverini, F.G. Lavacca, J. L. Herrera, and J. Berrocal, "Joint energy efficiency and load balancing optimization in hybrid IP/SDN networks", *Annals of Telecommunications*, vol. 78, pp. 13-31, 2023.

[23] L. Peterson, C. Cascone, B. O'Connor, T. Vachuska, and B. Davie, "Software-Defined Networks: A Systems Approach", Systems Approach LLC, 2021, pp. 194.

[24] A. Ghosh and T. Manoranjitham, "A study on load balancing techniques in SDN", *Journal of Engineering and Technology*, vol. 7, pp.174-177, 2018.

[25] F. Armenta-Cano, A. Tchernykh, J. M. Cortés-Mendoza, R. Yahyapour, A. Y. Drozdov, P. Bouvry, D. Kliazovich, and A. Avetisyan, "Heterogeneous Job Consolidation for Power Aware Scheduling with Quality of Service", in *RuSCDays'15 - The Russian Supercomputing Days*, 2015.

[26] T. V. Duy, Y. Sato, and Y. Inoguchi, "Performance evaluation of a Green Scheduling Algorithm for energy savings in Cloud computing", *IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW),* 2010, doi:10.1109/IPDPSW.2010.5470908.