JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A Research on load balancing on software defined networks

Buhyavarapu Manasa, A.Ramesh Babu

Research Scholar, Professor

Chaitanya Deemed to be University.

Abstract: The traditional networks are facing difficulties in dealing with services provided through cloud computing, big data and the Internet of Things (IoT), as users increasingly rely on their services. Software Defined Networking (SDN) has sparked enthusiasm in the system for integrating technology and features in line with consumer requirements for both academia and business, and its adoption has begun using real frameworks. Due to the increase in demand and the scarcity of resources, the problem of load balancing must be properly addressed in order to manage visitors and internal resources and improve the overall performance of the network. One of the most important issues is the placement of the SDN controller to stabilize the weight for better Quality of Service (QoS). Although there have been a few survey articles written about load balancing, there is no detailed and systematic evaluation done on load balancing in SDN. Therefore, this paper expands and clarifies the dialogue with a classification of the current emerging load balancing technologies in SDN in a systematic way by classifying the technologies as traditional and purely AI-based strategies for better provider optimization. Several fully SDN-based load balancing strategies are discussed to improve the overall performance of SDNs. Therefore, this paper discussed load balancing regulators in SDNs and discussed their advantages and disadvantages.

Keywords: Load Balancing, software defined network, Quality of service, internet of Things.

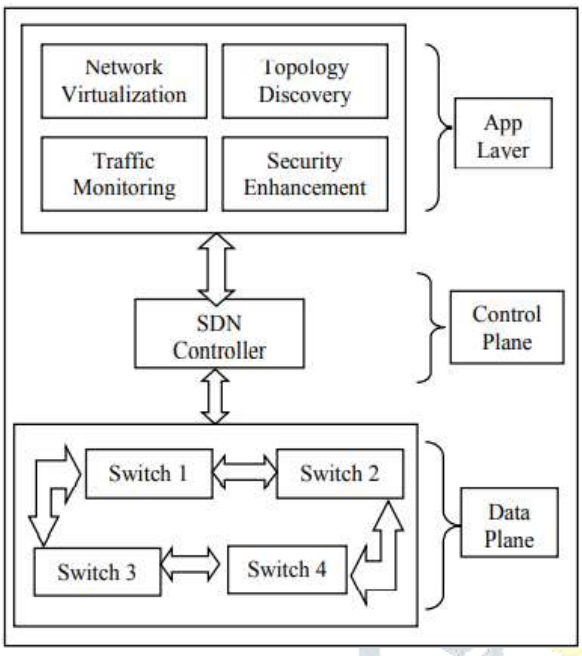
I. INTRODUCTION

Software is a common term within the vocabulary of most technologies, and its integration with communications and networking works ends up in emerging technologies such as Software Defined Networking (SDN) [1] and Network Functional Virtualization (NFV). Besides the benefits, it also brings demanding situations as they can be two sides of the same coin. This technology requires network extensions to include software within the devices to control them. Although the idea of having centralized control of the network is not new, the holistic view of separating the control plane from the reality plane has made tech geeks benefit from the knowledge. The SDN function to effectively manage the most added capacity of the community has earned the trust of transport service providers. OpenFlow and Path Computation Element are

two technologies that support SDN. OpenFlow is a generic protocol promoted by the Open Networking Foundation (ONF), which separates the control plane from the delivery and provides an interface between the control planes inside the console and the statistics plane on the connection switch. The PCE is Internet Engineering Task Force (IETF) compliant [2] and is desirable for closed environments such as data centers where address computation is migrated from the network to the console.

SDN Architecture

SDN (software defined network) has three layered architectures comprised of Control layer and Data forwarding layer, the application layer shown in fig .1



II. LOAD BALANCING

Load balancing acts as a conscious routing protocol in Software Defined Networks (SDN), and is an essential entity that aids in availability and scalability, resulting in minimal software latency. Millions of people are on the Internet, which leads to increased site visitors which leads to network congestion and packet loss. Therefore, to solve this problem, the method of load balancing strategies increases the efficiency of the community [3]. The current era of the load balancing framework is going in vain towards the Software Defined Network (SDN) framework. The master station or community needs manager is terminated using packages. Handling the server load is easy because it is tied to the console in the framework. But special server control standards are needed to balance the load of servers within the SDN [4]. Workload is divided directly into a central processing unit (CPU) or represents a group of computers, a network hyperlink, or a computer. In the form of load balancing, incoming IP web page traffic can be split across a pair of servers, improving server operability. Redirect client request to backend servers and get responses from servers back to user. It also makes it less difficult for clients to connect to backend servers at this time, softening the look of the internal network and preventing community abuse.

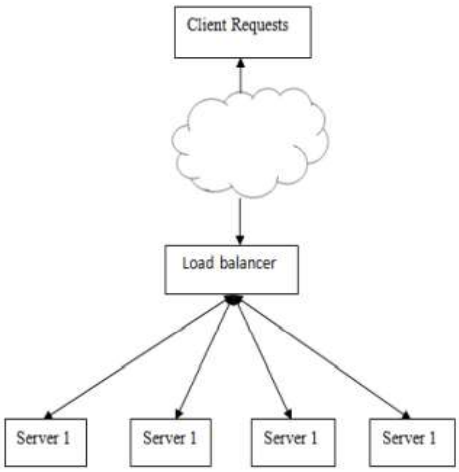


Fig.1 Load balancing architecture

III. REVIEW OF LITERATURE

In this section, several articles examine the network identified by load balancing programs. In this work, it was determined that various problems in the software characterized the network to overload information on the network. To win over weight in the software network, I studied several optimization methods used.

Usman Ahmed et al. [2021] proposed a load balancing algorithm that improved the usability of the sensor by using the computing power of the sensor and the desired of the source for intrusion detection, which can enrich the attack on the community. This paper mixed the entropy-based active study version to effectively identify parasitism patterns. The effects showed that the designed model could help and improve the selection limit with the help of increasing the didactic example through the aggregation strategy and the degree of entropy uncertainty.

Kiran A Jadhav et al. [2020] Implemented Round Robin load balancing method and random load balancing approach using OpenFlow transport to deal with the difficulties in traditional networks due to network increase in users consisting of computing in the cloud and a lot of information. Mininet simulation results: The use of round robin ruleset resulted in better performance and reduced packet loss compared to the default random scheduling algorithm. Load balancing on a traditional network relies on local community statistics and becomes unprogrammable. But SDN controllers have an international view of the network and are also programmable, allowing them to design improved load balancing mechanisms.

Mohammed Moin Mulla et al. [2019] The OpenFlow protocol was proposed and became a basic building block for SDNs. SDN runs on the OpenFlow protocol on which the idea of Programmable Networks was created. SDN transforms the traditional community structure by separating the network logic into different levels: the management level and the statistics level. Load balancing techniques are implemented in the constructed structure by two methods, i.e. designated separation procedures and site visitors. The overall performance evaluation of load balancing techniques was compared with network quality of service parameters such as throughput, delay, and packet loss.

Shang et al. [2018] A service-oriented load balancing mechanism for software-defined networks was proposed to solve the problems of societal load imbalance and scalability at the top of the level. In the model, the statistics of slip requests between switches was used as the base unit of the controller, and all controllers periodically publish the total number of drift requests and go with the mean skew of flow requests changed for their load added to the conscious reputation of the drivers. On this basis, the load balancing algorithm is mainly based on.

Zhao et al. [2017] this paper attempted to answer the following question: How to perform per-controller load balancing and hyperlink load balancing in SDN? They formulated the Load Balancing Directive for Hyperlinks and Controllers (LBR-LC) on SDN and demonstrated its NP robustness. A set of rules entirely based on approximation is proposed to solve the problem of scalability and driver loading, and the overall performance approximation is analyzed. In addition, they discussed the effective mechanism for renewing the network's reputation among exchange controllers.

C. Yu et al., [2017] mentioned that the software describes a hub-based network architecture and a new method for load balancing. Dijkstas and Q-learning are simpler than DNQ, and more complex than two . To optimize the c language at the time to re-select the direction, the median-based architecture will search for suboptimal paths.

Kaur et al. [2016] An SDN application has been developed that performs server load balancing. The main problem with the traditional load balancer was that they used dedicated hardware. Those devices became expensive and hard. Network administrators cannot write their own personal algorithms because traditional load balancers are blocked by the sender. So, to address these issues, they created SDN software that turned a simple OpenFlow tool into an efficient load balancer. There are already some load balancing algorithms in SDN, but the main problem with this type of algorithm is that every request and return message must pass through the load balancer. Offers useless latency. To address these issues, they have implemented load balancing algorithms that are mainly based on direct routing. In the direct routing, the weight balancer was not concerned with the return message from the internet server to the buyer. The proxy server immediately responds to the user bypassing the load balancer for this reason, which leads to improved performance.

Davoli et al. [2015] They proposed an architecture that integrated the SDN model with TE primarily based on SR, providing an open source reference implementation. They designed and implemented a simple guide for TE/SR for flow mapping. Traffic engineering (TE) in IP carrier networks was one of the functions that the software-defined networking model could have. With the logical centralization of community management, it will be possible to "schedule" floating routing based on TE dreams. Traditional layout routing requires real-time interaction between the SDN controller and each node involved in the traffic paths. Depending on the granularity and temporary homes of the streams, this can result in scalability being released for the amount of routing state you want to keep on the core community nodes and for visitors to the desired configuration. On the contrary, segment routing (SR) was an emerging method of routing that could simplify

route implementation by delegating all the configuration and according to the flow domain at the community boundary.

Zhang et al. [2014] A new type of controller country synchronization scheme, called load variance synchronization (LVS) based synchronization, has been proposed to improve weight balancing performance within multi-zone controller SDN internet paints. Compared to PS-based schemes, LVS-based schemes primarily perform better robust domain synchronizations between controllers while the weight of a particular server or region exceeds a positive threshold, which significantly reduces the synchronization burden of controllers. . Simulation results show that LVS performs loop-free redirection and excellent load balancing performance with much lower synchronization burdens, compared to current schemes.

Hui, et al. [2013] In intermediate information networks, the method of balancing workloads is a major difficulty with rapidly increasing network schedules. The Open Flow protocol, a powerful nuisance solution candidate, offers each consumer programmatic manipulation of specific flows, in an effort to determine their paths through the community. However, the most effective current responses based on open flow try to find a static routing path throughout the initialization step, while the static routing path often suffers from poor overall performance because community settings can alternate more in the course of information transmission. To solve the inconvenience, this document proposes LABERIO, a unique routing algorithm, to achieve dynamic stabilization of traffic at some point in transmission. Community engineering method experiments show that LABERIO outperforms various regular load balancing algorithms such as Round Robin and LOBUS by reducing transmission time by up to 13%.

Kwangtae Jeong et al. [2012] The QoS Aware Network Operating System (QNOX) has been proposed for SDN with the widespread OpenFlows. Practical modules and QNOX processes to provide SDN services with awareness of service quality with key components (e.g., Provider Element (SE), Control Details (CE), Management Element (ME) and Knowledge Element (CKE)) explained in details. The current reputation for prototyping implementation and performance is explained. QNOX scalability is also analyzed to verify that the proposed framework can be implemented online for large scale carrier-class companies.

Richard Wang et al. [2011] provided algorithms that computed summary wildcard patterns that completed target traffic distribution and routinely adjust adjustments to load balancing policies without disrupting existing connections. They applied these algorithms to the NOX OpenFlow console, evaluated their effectiveness, and suggested several methods for conducting similar studies. In fact, custom load scales have become highly priced and are quickly becoming a single factor for failure and congestion. The OpenFlow standard provides an opportunity approach where switches on a commodity network segment traffic across server replicas, based on compliance with rules set by an independent controller. However, the simple method of mounting a separate base for each user connection (or "microflow") results in a wide variety of policies on the switches and a significant load on the controller.

Wenyu Zhou et al. [2010] designed and implemented a load balancing scheme entirely based on a policy of mapping dynamic aids to groups of digital devices, which were placed on a set of physical Machines (PM)

in a common garage architecture. Monitors real-time asset usage from VMs and PMs, along with CPU, memory, and network, then uses out-of-the-box reset of VMs that were running in the same MP to get load balancing of adjacent VMs, even when using stay In VM migration between PM to perform international VM load balancing. Optimize resource allocation to virtual machines for an international load balancing of the pool of digital devices.

Pushpendra et al. [2009] Suggested a set of rules for a wide variety of workload situations, including I/O intensive blocks and deep memory. However, from this challenge, the device's CPU needs were minimal, as the tasks usually presented were video recovery commitments that required little interaction with the system, yet consumed a lot of I/O. The proposed algorithm was intended to pin requests across the entire server stack based on their memories, CPU needs, and input/output so that the reaction time and end time for each activity are minimal. Preventive job migrations are not considered here. The normal transaction of its model should be defined as the period between the reputation of the assignment on the device and the fulfillment of its needs through the system. The job needs are video files that the device has to load from a secondary storage device and stream the video continuously to the leaving user who initiated the request.

Nick McKeown et al. [2008] Suggested OpenFlow: A way for researchers to run experimental protocols on the networks they use on a daily basis. OpenFlow relied primarily on Ethernet pass-through, with an internal bypass desk and a standard interface for rendering and removing flow inputs. Their goal was to inspire networking vendors to add OpenFlow to their ICs to deploy to campus backbones and wiring closets. We are confident that OpenFlow is a realistic compromise: on the one hand, it enables researchers to conduct experiments on heterogeneous switches consistently with high line loading and port densities; while instead, companies do not need to expose the inner workings of their keys. In addition to allowing researchers to evaluate their ideas in real-world visitor environments, OpenFlow can serve as a useful on-campus component in proposed large-scale testing modules such as GENI. Two Stanford University homes will soon be running OpenFlow networks, using industrial Ethernet switches and routers. We will draw to inspire diffusion in different universities; and we inspire you to remember to implement OpenFlow in your university network too.

IV. LIMITATIONS IN EXISTING LOAD BALANCING ALGORITHMS

In computing, the load can also vary like CPU load, memory load, network load, etc. Load balancing is a technique for dealing with website visitors in an efficient manner in which server resources are lightly applied to some of the available servers. Many mainstream boards have implemented efficient resource use, but role response time has been reduced. Cloud computing wants to handle a large number of requests, such as search requests and relay requests. The project here is to maintain the same overall performance while ordering an explosion of information within the center of the facts. Therefore, we proposed dynamic load balancing algorithms that use resources efficiently, reduce response time, and handle fast traffic. Other problems are explained below.

With the analysis of the implementation of the previous researcher, here we summarize some limitations.

Automated Service Provisioning

Flexibility is the main criteria in cloud computing, and offerings are set mechanically. The main challenge is how to successfully allocate or offload server assets.

Spatial Distribution of the Cloud Nodes

Several algorithms have been proposed to be useful to the intranet community, reducing ad waiting time. However, it is very difficult to support a set of load balancing rules for geographically divided nodes located in exclusive locations. For this, business element such as network link rate, distance between nodes, distance between consumer and challenge filter nodes must be considered.

Virtual Machine Migration

With virtualization, a single device can be seen as a file, a set of files, or a pair of virtual machines. The primary objective is to distribute the workload in the data center or in the organization of statistics facilities for the ratio of the workload between the machine overloaded and the machine under load.

Algorithm Complexity

Load balancing algorithms are less difficult when it comes to implementation and implementation. Some issues need more dedicated data and better communication to investigate and address. But setup time causes additional problems and also affects network performance.

V. CONCLUSION

In this paper, the survey presents the possibility of different methodologies, unusual techniques for load balancing and related works studied for predicting load balance mechanisms using several distinctive techniques. Like mainstream society, SDN's path-repetition era succeeded in bringing strength and balance to the community. However, the slight distribution of traffic across multiple routes has been shown to be a pressing problem for SDN researchers. In the future, we can always look at load balancing schemes within distributed SDN architectures and look for higher load balancing approaches to better reflect the usefulness of the SDN fabric and promote further optimization of the SDN.

REFERENCES

1. H. Xue, and H. Youn, 2019, ‘‘Dynamic load balancing of softwaredefined networking based on genetic-ant colony optimization,’’, pp. 311.
2. R. Ramar and H. Gasmelseed, 2019, ‘‘Traffic pattern-based load-balancing algorithm in software-defined network using distributed controllers,’’, page no. e3841.
3. M. N. Jahantigh and A. Rezaee, 2020, ‘‘Integration of Internet of Things and cloud computing: A systematic survey,’’, pp. 165–176.

4. Y. E. Oktian and J. Lam, 2017, ‘‘Distributed SDN controller system: A survey on design choice,’’ Comput. Netw., vol. 121, pp. 100–111.
5. Usman Ahmed, Gautam Srivastava, 2021, "Network-Aware SDN Load Balancer with Deep Active Learning based Intrusion Detection Model", Neural Networks (IJCNN) 2021 International Joint Conference on, pp. 1-6.
6. Kiran A Jadhav, Mohammed Moin Mulla,Narayan D. G, 2020, “An Efficient Load Balancing Mechanism in Software Defined Networks”, IEEE, pp, 116-122.
7. Mohammed Moin Mulla, M. K. Meghana, 2019, “Load Balancing for Software-Defined Networks”, pp. 235-244.
8. F. Shang, L. Mao and We. Gong, 2018, “Service-aware adaptive link load balancing mechanism for Software-Defined Networking”, pp. 452-464.
9. G. Zhao, L. Huang, Z. Li and H. Xu, “Load-Balancing Software- Defined Networking Through Hybrid Routing” Published in: International Conference on Wireless Algorithms, Systems, and Applications WASA, pp. 96-108, 2017.
10. C. Yu and H. Zhang, 2017, “Intelligent Optimizing Scheme for Load Balancing in Software Defined Networks,”, pp. 1-5.
11. S. Kaur and J. Singh, “Implementation of Server Load Balancing in Software Defined Networking”, In Information Systems Design and Intelligent Applications, Springer, New Delhi, 2016, pp. 147-157.
12. Davoli, Luca, et al, 2015, "Traffic engineering with segment routing: SDNbased architectural design and open source implementation" Software Defined Networks(EWSDN).
13. Zhang, Hailong, and Xiao Guo. "SDN-based load balancing strategy for the server cluster." Cloud Computing and Intelligence Systems (CCIS), 3rd International Conference IEEE 2014.
14. Long, Hui, et al, 2013, "LABERIO: Dynamic load-balanced routing in OpenFlow-enabled networks.” IEEE 27th International Conference on.
15. Kwangtae Jeong, Jinwook Kim and Young-Tak Kim, “QoS-aware Network Operating System for Software Defined Networking with Generalized OpenFlows” 2012 IEEE/IFIP 4th Workshop on Management of the Future Internet (ManFI).
16. Richard Wang, Dana Butnariu, Jennifer Rexford. OpenFlow-based server load balancing gone wild. Hot-ICE, 2011. Proceedings. 2011.
17. Wenyu Zhou, Shoubao Yang, Jun Fang, et al. Vmctune: A load balancing scheme for virtual machine cluster using dynamic resource allocation. International Conference on Grid and Cloud Computing, 2010: 81-86.
18. Pushpendra Kumar Chandra, Bibhudatta Sahoo, 2009, “Performance analysis of load balancing algorithms for cluster of videos on demand servers”. IEEE International Advance Conputing Conference, page no. 408-412.
19. Nick McKeown, Hari Balakrishnan, et al, 2008, “OpenFlow: enabling innovation in campus networks”.