Analyzing Methods and Opportunities in Software-Defined (SDN) Networks for Data Traffic Optimizations

Yasir Ali Matnee(Author)
Department of Computer Science
Basic Education College, Diyala University
Diyala,IRAQ
ee22a12@gmail.com

Chasib Hasan Abooddy*(Author)*
Department of Computer Science
Basic Education College, Diyala University
Bagdad, IRAQ
Abooddy@gmail.com

Zainab Qahtan Mohammed*(Author)*
Department of Computer Science
Basic Education College, Diyala University
Diyala, IRAQ
altmimi104@gmail.com

***Abstract*—**Computer networks are dynamic and require constant updating and monitoring of operations to meet the growing volume of data trafficked. This generates a number of cost issues as well as performance management and tuning to deliver granular quality of service (QoS), balancing data load, and controlling the occurrence of bottlenecks. As an alternative, a new programmable network paradigm has been used under the name of Software Defined Networks (SDN). The SDN consists of decoupling the data plane and controlling the network, where a programmable controller is responsible for managing rules for routing the data to various devices. Thus, the hardware that remains in the network data stream simply addresses the routing of the packets quickly according to these rules. In this context, this article conducts a study on different methods and approaches that are being used in the literature to solve problems in the optimization of data traffic in the network through the use of SDN. In particular, this study differs from other reviews of SDN because it focuses on issues such as QoS, load balancing, and congestion control. Finally, in addition to the review of the SDN's state-of-the-art in the areas mentioned, a survey of future challenges and research opportunities in the area is also presented. load balancing and congestion control. Finally, in addition to the review of the SDN's state-of-the-art in the areas mentioned, a survey of future challenges and research opportunities in the area is also presented. load balancing and congestion control. Finally, in addition to the review of the SDN's state-of-the-art in the areas mentioned, a survey of future challenges and research opportunities in the area is also presented.

Keywords*-Networks Defined by Software; Performance; Congestion; Challenges; Load Balancing.*

*****

1. INTRODUCTION

The ability of computer networks coupled with infrastructure problems and the increasing volume of data traffic are limiting today for performing scalable services that need high availability as well as adequate performance. This amount of data comes from a variety of sources, such as the large number of users who send and receive information, intelligent sensors and applications that perform large data transmissions [1]. Conventional networks use special algorithms implemented in dedicated components to control and monitor the data flow in the network, managing the routing of packets and reassessing, at regular intervals, the state of the connection between the network devices.

In the conventional network, when a packet is received by a network device, whether it is a router or switch, a rule set is built into its own firmware in order to find the destination device to optimize the packet forwarding path [2]. These network devices use a variety of time-consuming and complex protocols in terms of management and time, where network administrators are responsible for setting up policies to respond to a wide variety of network events and applications involved. In particular, administrators need to manually transform high-level policies into low-level configuration commands while adapting to changes in network conditions. Often, these tasks need to be done through proprietary tools that are limited in functionality.

Network management and fine-tuning of performance is quite challenging in the traditional scenario, becoming also prone to errors due to poor human interpretation of network events [3]. In this context, software program- mobility paradigm (SDN) ideas come back with the popularization of the OpenFlow protocol, eliminating the rigidity present in traditional networks. With this, a decoupling of the data and control layers occurs, where all the network management becomes programmable in a controller, while the network hardware (switches or routers, for example) is simply the task of routing packets according to the rules that are established in the software [4].

An SDN allows network behavior to be more flexible, as well as adaptable and manageable according to the needs of each business context. As discussed, OpenFlow is a standardized protocol for communication between switches and the SDN controller. It was proposed as a way for the scientific community to test new ideas and experiments. Initially OpenFlow was deployed in the network of an academic campus [5], but nowadays it is used in several US and European universities [6]. In addition, the industry has also adopted this protocol in its SDN implementations as a strategy to increase network functionality and reduce the costs and complexity of network hardware.

Several literature reviews were performed with a focus on SDN. For the most part, the focus of research is on (I) concepts, challenges and future opportunities for SDN [1, 2, 7]; (II) programmable networks [3]; (III) innovations in the Openflow protocol [4]; (IV) taxonomy for classification of literature reviews in SDN [8]; (V) solutions and problems caused when updating flows and rules in the SDN network [9, 10]. This article presents a literature review of methods and approaches that are being used by researchers to solve

problems of optimizing data traffic in networks through the use of SDN. The expression traffic optimization is observed in the analysis of quality of service (QoS), load balancing and congestion control. In this sense, the contribution of the article appears in the discussion of SDN's state-of-the-art in the view of these three areas and in an analysis of research challenges and SDN opportunities focused on optimization and network performance.

This article is organized in 7 sections. After the introduction, Section 2 sets out the criteria for choosing the articles used in this literature review. In Section 3, we present the SDN concept and architecture, also exploring features and the OpenFlow protocol architecture. Section 4 shows recent research related to methods and approaches for optimizing network traffic using SDN concepts. An analysis of research challenges and opportunities in the area is presented in Section 5. Finally, Section 6 re-presents the main conclusions of the research, also highlighting their contribution.

2. Criteria for choosing articles

In order to select the articles for the present literature review, an analysis was performed which was conducted by answering questions related to network traffic optimization through the use of SDN as follows: (I) As software defined networks are currently defined ? (II) How are software-defined networks being applied to network traffic optimization? (III) What techniques and approaches have been used to control QoS, load balancing and avoid congestion in the network through the use of SDN? (IV) What are the main open challenges and opportunities to be faced and overcome in the use of software-defined networks to optimize network operations?

Based on these analysis questions, the most relevant search terms were defined. They are: (I) software-defined networking or SDN; (II) quality of service or QoS; (III) load balancing; (IV) congestion control or congestion avoidance. The search terms were chosen following the following steps [11]:

• Define the main search terms;

• Identify alternative synonyms related to key terms;

• Use OR operators to incorporate alternate synonyms and ANDs to link main terms.

The databases used as the research source were: (I) Google Scholar; (II) IEEE Explore; (III) ACM Digital Library. The process of filtering articles adopted the following criteria: (I) year of publication: an interval of 3 years was adopted for the collection of material, that is, from 2014 to 2016; (II) type of publication: scientific journal or proceedings of conferences; (III) publication context: The context of the research was taken into account for the inclusion of items in this research, where it was chosen by academic research; (IV) language: English only articles. The initial research included articles from the past three years, and then references were used to review the subject and to determine the challenges and opportunities related to SDN and optimizing data traffic. At the beginning of the research, 694 papers were found and after the removal of impurities according to the above mentioned criteria, removal of duplicate articles, filter by title, filter by abstract, filter by introduction and filter by complete reading, we add to the 24 articles selected, as shown in Table 1.

Table 1: Works analyzed for the present literature review.

Article	**Modality**
An sdn-based multipath qos solution [12]	Service quality
Qamo-sdn: Qos aware multipath tcp for software defined optical networks [13]	Service quality
Experimental demonstration of latency-aware software defined networking for networking for openflow-based intra-datacenter optical interconnect networks [14]	Service quality
Minimizing latency of critical traffic through sdn [15]	Service quality
Application-oriented bandwidth and latency aware routing with openflownetwork [16]	Service quality
An adaptive routing scheme for heterogeneous data-flows using openflow [17]	Service quality
Congestion control using openflow in software defined data centernetworks [18]	Congestion Control
A joint approach to multipath routing and rate adaptation for congestion control in openflow software defined network [19]	Congestion Control
A congestion avoidance algorithm in sdn environment [20]	Congestion Control
Otcp: Sdn-managed congestion control for data center networks [21]	Congestion Control
A novel proposal to effectively combine multipath data forwarding fordata center networks with congestion control and load balancing using software-defined networking approach [22]	Congestion Control
Software-defined networking: Challenges and research opportunities for future internet [1]	Survey
A survey on software-defined network and openflow: From concept to implementation [2]	Survey
A survey on software-defined networking [7]	Survey
A survey of software-defined networking: Past, present, and future of programmable networks [3]	Survey
Network innovation using openflow: A survey [4]	Survey
A survey and a layered taxonomy of software-defined networking [8]	Survey
Rules placement problem in openflow networks: A survey [9]	Survey
The problems and solutions of network update in sdn: A survey [10]	Survey

 ISSN: 2321-8169
75 – 82

3. Software Defined Networks

Software-defined networks, also known as SDN, are a recent research topic that brings with it a new paradigm for networks when compared to the traditional network model [3]. SDN decouples the control and data layers. Thus, the software component is responsible for the whole control plan of the network, while the hardware is only responsible for the routing of packets according to the rules established by the software [4]. In this way, SDN offers a broad abstraction network that simplifies network management and reduces hardware costs and complexity [4] [3]. At this high level of abstraction, there is an environment in which all traffic engineering, data routing, and access control can be implemented quickly and efficiently without the need for low-level policy creation [28]. As a consequence, the complexity of the network devices is reduced, since only a simple interface is exposed for the administrator to perform the configurations, freeing him from acting manually on devices from different vendors [3]. Figure 1 presents a comparison between traditional network architecture against SDN network architecture.

The SDN standard architecture consists of three main layers: (I) Application Layer; (II) Control Layer; (III) Infrastructure Layer. The infrastructure layer consists of a set of network elements, such as switches and routers, which together represent the data plan with the responsibility of routing packets according to the instructions of the control plan [25]. The central layer, known as the control layer, is the network core defined by software, which is formed by a centralized controller that allows the network functions to be programmable in a fast and easy way. This controller provides a global view of the entire network infrastructure for the application layer and uses the southbound interface to configure the routing of data on network switches by installing rules on each of the switches [25] [2]. Currently, the predominant protocol for this communication channel between the controller and the network components is OpenFlow [29]. Section 4 will provide details about the OpenFlow protocol.

Finally, at the top of the layers is the application layer, which is formed by the applications and services of the network. It is considered applications and services of the network, modules that somehow implement a logic for functionalities of traffic engineering, data routing, load balancing, fault tolerance, etc. [25]. In Figure 2 we can see the organization of the three main layers of SDN. As a final result, SDN results in network improvements related to performance optimization, control and management of granular policies, and simplification of resource provisioning. In this model, administrators have the power to control the entire data stream by changing the way data is handled and routed in switches from a centralized location at run time through an application module that communicates with the controller, without the need to configure each of the devices manually [30].

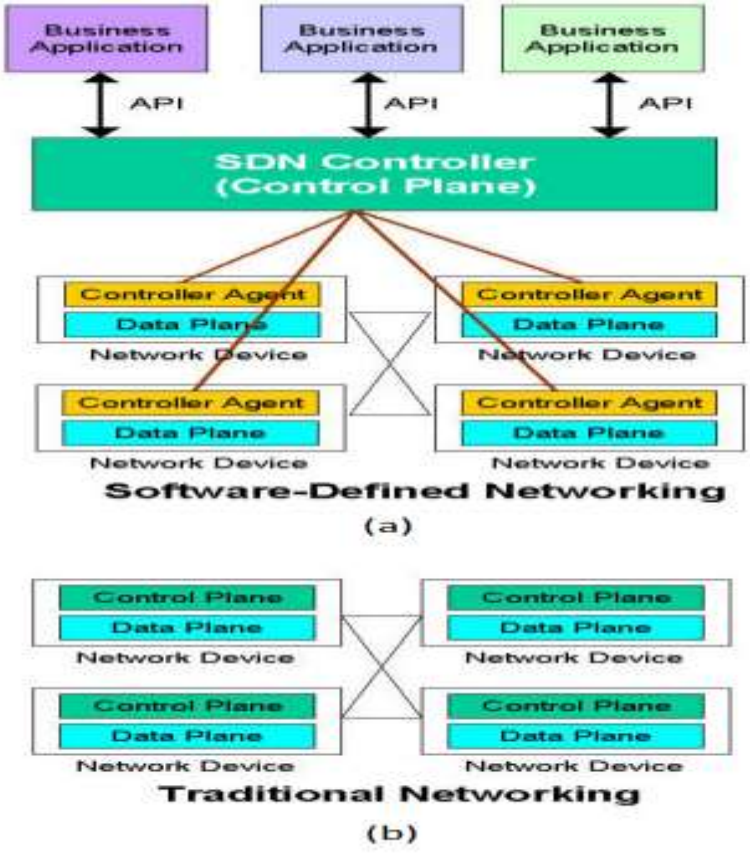

Figure 1: Comparison between the architectures of (a) traditional network; (b) SDN network

 ISSN: 2321-8169
75 – 82

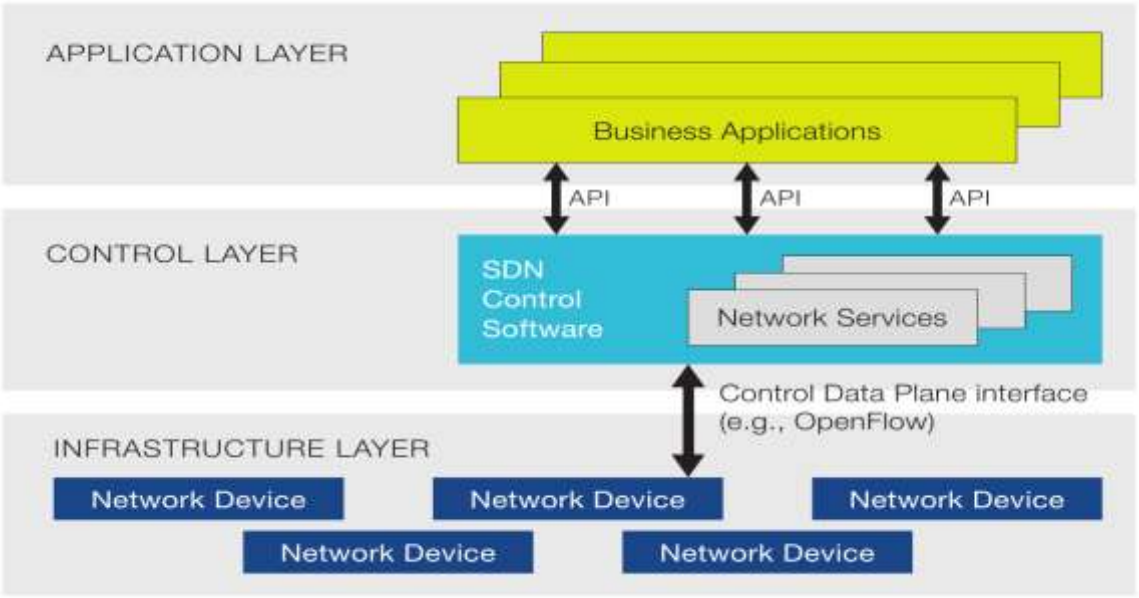

Figure 2: Main layers of SDN architecture

3.1 Design Models

SDN supports different models, which have different architectures, requirements and forms of control. We can cite as main the models: (I) centralized vs. distributed; and (ii) reactive vs. proactive policies, which will be detailed now [3].

3.1.1 Centralized vs. Distributed

SDN supports both the centralized model and the distributed model. The centralized model is based on a single central controller that has a global view of all elements of the network, performing all management and control over them [1]. With this, all intelligence and logic is centralized in a single point. This model is supported by the Open Network Foundation, and OpenFlow is the main protocol for management in this administration format [30].

Because it is centralized, it has as main advantages a lower implementation complexity, a single point of management and better control over the states of the network, due to the global knowledge of the network. However, this modality also has negative points. The first one is in relation to scalability, where all functionalities are in a single point that requires greater computational power, energy consumption and that consequently makes the storage of rules limited [1, 9]. Still, the constant search for network elements and their statistics produces an overload on the controller, which increases response times. In addition, another point of concern is related to fault tolerance, which becomes a problem when having a single controller as a single point of failure. In this respect, for fault tolerance, the OpenFlow protocol, in later versions, allows the connection of multiple controllers on the same switch, which offers the possibility of triggering a backup controller in case of failures [3].

In the distributed model, the single point of failure is eliminated, since the control and management happens through different controllers. This makes the distributed model more robust, scalable, and capable of responding quickly to network events. Despite these advantages, the complexity of deployment in this model becomes greater because it is difficult to obtain a global view of the network and keep the controllers synchronized. There are major challenges related to finding ways to synchronize local and distributed events to provide a global view of the network to controllers, mainly due to communications overhead [1]. Currently, among the initiatives that seek to find solutions to have a centralized logic with a distributed control plan, the research carried out by [31, 32] stands out.

3.1.2 Reactive vs. Proactive Policies

In the control of reactive policies, the network elements responsible for packet forwarding consult the controller for each decision that needs to be made. This happens for example in the arrival of a packet relative to a stream not yet mapped in the switch. This causes the delay to increase for each first packet of new streams, since upon arrival it is necessary that the packet be routed to the controller to make the necessary decision / action. After this process, the next packets of this stream can follow the same action without the need to query the controller. In this model, in some cases, a great impact does not occur, but in large geographically distributed networks this can generate a considerable negative impact [3].

In contrast, in the proactive model, the controller installs the necessary rules on the switches, avoiding that the equipment performs frequent queries on the controller during the discovery of new flows [3]. So, just do a search only on the local flow table of the switch. As a consequence, the proactive form eliminates any latency induced by querying a controller in each stream.

4. ONF AND THE OPENFLOW PROTOCOL

The Open Network Foundation (ONF) is an organization founded in 2011 that is dedicated to promoting the adoption of software-defined networks (SDN) through the development of open standards [30]. As a form of standardization for communication between the control plan and data plan, ONF proposed the OpenFlow protocol, which became the first protocol for this purpose to be used in SDN [19]. It is an open protocol that allows the implementation of SDN concepts for hardware and software [2]. It is currently among the most popular protocols for this purpose and generally describes how applications can program the flow table of different switches [4] [5].

As can be seen in Figure 3, the OpenFlow architecture consists of three main components [2, 29, 4]: (I) switches: Openflow manages and infers in the flow table of different switches and routers. This flows table is an input list of flows, which have a field of correspondence, counters and instructions / actions; (II) controller: it is a program responsible for manipulating the flow table of the network device through the OpenFlow protocol, where it performs update operations, inclusion or removal of rules flows. This process can be static, where the controller is a simple software drive running on a computer that establishes static rules during execution, or dynamic, where the establishment of rules for the path of the packets is changed during execution; (III) secure channel: it is an interface that connects the controller with all the switches of the network. It is through this channel that the controller performs management, which includes receiving and sending control packets between the switches involved.

When a packet arrives on the switch, it is computed on-the-fly and compared to the match field of each of the rules present in the switch's flow table [3]. If any of the rules match, the packet is processed according to the action specified for the rule in question. Otherwise, the packet is encapsulated and sent to the controller to decide which action to take [4]. The vast majority of OpenFlow switches support at least the following rules: (I) send the packet to a particular port; (II) encapsulate and send the packet to the controller; (III) discard the package.

Different versions of the OpenFlow specification are currently available. The first version was the 0.2.0 released in March 2008. Later versions were released until arriving at 1.0.0 [33] in December 2009, which became the most deployed. After this version, versions 1.1.0 [34], 1.2 [35], 1.3 [36], 1.4 [37] were still released until the current 1.5 version [38] was released.

5. OPTIMIZING NETWORK TRAFFIC

It is well known that data traffic on the Internet is growing rapidly, and this volume of information comes from a variety of sources, such as Internet users, smart sensors and mobile devices [39]. Communication networks end up suffering from their inflexibility and inability to adapt to the different demands of transmitted traffic [40]. It is necessary to have efficient strategies capable of managing and handling this volume of information in an optimized and efficient way. In this context, SDN is a promising and powerful concept that introduces new dimensions of flexibility and adaptability in communication networks [40]. It makes the network become programmable by providing control and centralized view of the entire network, creating opportunities for optimizing network traffic that can be extracted through congestion control, quality of service management, and load balancing with adaptive and intelligent data routing for traffic engineering.

5.1 SDN applied in QoS

Quality of Service (QoS) provides performance guarantees for applications through metrics such as bandwidth, delay and packet loss [12]. Generally, the idea is to have the ability to provide different priorities for the applications, ensuring a certain level of performance to a data flow. Nowadays, Internet architectures offer only the best effort service, so it is not possible to guarantee QoS for applications [12]. Initiatives such as the Internet Engineering Task Force (IETF) have exploited QoS architectures such as IntServ [1] and DiffServ [2], but have not been successful in their adoption. One of the main reasons for the lack of success of these architectures is the absence of knowledge regarding the global state of the network [2]. In contrast, with the OpenFlow protocol, it is possible to offer a granular support in QoS, allowing to specify the way to handle individual flows, considering the different metrics of the network.

In [13] the QAMO-SDN architecture for fiber optic networks is presented using the new MPTCP transport protocol with a QoS provisioning algorithm for SDN in cloud datacenters. QAMOSDN has the ability to be adaptive and self-configurable dynamically based on the current network situation. In [12], the authors present the HiQoS project, a proposal that offers assurances of bandwidth utilization through the use of SDN. HiQoS uses different routes between a source and destination, as well as a queuing mechanism to ensure bandwidth for different types of traffic. When data traffic of a particular application type is greater than the bandwidth allocated to the queue, a new least-used path is added to the data traffic. Results show that HiQoS reduces data transmission delays and server response times, thus increasing system throughput.

In [14] the LaSDN architecture is proposed. An architecture aware of the latency that OpenFlow uses in intra-datacenter networks. The architecture is able to organize services with sensitivity in the current latency by introducing a latency-aware service scheduling strategy (LaSS). In this way, LaSDN organizes and accommodates applications with the necessary QoS considering the latency factor and offering greater availability and responsiveness to intra-datacenter demands. In the proposal of [15], latency is also used as a metric to manage the delay of important data traffic. A solution that performs periodic analysis to find a path with the least delay and to carry out data routing through this path is proposed. The authors of [16] propose aware routing latency and bandwidth for different services. The idea is to allocate distinct routes for two types of applications: bandwidth dependent applications and latency dependent applications. They provide an API where the application communicates with the controller to register the need for which type of routes. Based on this preference, the controller inserts the appropriate streams into the respective switches.

Later, in [17] a controller was proposed that uses OpenFlow and takes into account different needs beyond the latency and bandwidth, such as reliability and jitter. In this proposal, the applications send the profile they want in an OpenFlow protocol header field. This profile refers to a combination of different weights to consider for bandwidth, latency, and packet loss. Based on these parameters, the Dijkstra algorithm is used to generate the lowest cost path and then update the flows on the required switches. In both [16] and [17] the path analysis is calculated and changed completely, that is, there is no analysis and individualized change of the links for each path.

5.2 SDN applied in congestion control

Controlling network congestion is a very important task. Network congestion occurs when many packets are traveling on the same part of the subnet, exceeding their capacity. As a consequence there is delay and loss of packets, leading to a marked drop in performance. A congestion control should reduce the transmission delay and bandwidth consumption, causing an optimization to the transfer rate and consequently the network performance [18].

[18] Present a method to control the congestion in data centers and, in this way, to optimize the performance of the network. In the proposed method, the recognition of congestion in the links is discovered through the constant statistical check of the ports of the switches. When checking a flow with more than 70% of use, based on the current statistics of the network, the recalculation of a new path with more free resources starts. Similarly, [19] propose a congestion control using Openflow with a combination of different routes allied with a resource to adapt the delivery rate. It can be concluded that this combination produces a better performance than the traditional single path model and the different route model without adaptation of the delivery rate. The authors of [20] propose an algorithm to control congestion flows in SDN networks. It is intended to predict and infer in the network before the occurrence of packet loss. For this purpose, when it reaches 70% of network usage, the proposal recalculates a new route for data traffic based on the use of switch ports. It is observed that the solution is able to reduce the congestion in environments of high congestion. Thus, it would be possible to replace the slow start of the TCP congestion control algorithm with the proposed algorithm.

The Omniscient TCP (OTCP), presented by [21], is an approach that uses SDN to compute environment-specific congestion control parameters based on the available network properties found in the controller. Targeting the internal data center network, the goal is to address TCP deficiencies under aggregate workloads, achieving high throughput and low and stable latency. For this, TCP congestion parameter configurations are performed for each point-to-point route considering the topology, latency, throughput, and buffer. In [22] we present a method that aims to integrate dynamic load balancing and multiple thingswith congestion control through the use of SDN's purely applied approach to data center networks. Although some scenarios have not been tested, it may be noted that the approach has certain advantages and improvements over existing approaches. With the introduction of SDN, the proposed dynamic algorithm can update and react readily to load unbalance and traffic congestion.

5.3 SDN applied for load balancing

In traditional networks, the best way of balancing the workload is a very important issue. Today, load balancing servers are used to meet the high volume data demand; however this technique requires a lot of investment capital and offers low scalability and flexibility. This makes it difficult to support highly dynamic workload demands [41].

In [23] it is suggested an elastic approach to manage a video transmission service on different servers that run in the cloud. The strategy employs load balancing based on OpenFlow coupled with an elastic management mechanism responsible for increasing and decreasing the number of active transmission servers through a horizontal elasticity strategy. According to current user demand, resource utilization (memory usage, CPU and throughput) and elasticity policy thresholds, the mechanism either increases or decreases the transmission servers in the cloud and recalculates the routing of data traffic through OpenFlow features.

The authors of [24] propose a load-balancing algorithm for the OpenDaylight controller [42], focusing on load management on switch ports. The algorithm collects traffic load statistics from the switch ports, and by the time the number of bytes exceeds a given threshold, it generates a new stream to be installed in the switch's flow table. In this way load balancing occurs between the ports that have access to the backhaul. In [25], the authors present approaches to bring SDN to an end device (computer, smartphone, etc.) considering the scenario of load balancing across multiple network interfaces. It has as a differential the idea of taking load balancing to the client side by installing and configuring the OpenFlow protocol and an SDN controller on the client device.

[26] propose a proactive algorithm for routing with different paths called MRPL. The method aims to achieve load balancing through a linear program that uses heuristics to implement it in a distributed way. As a result, you can balance power consumption, maximize network life, and minimize the total cost of transmission for low-power networks. Also related to energy efficiency, [27] presents the OCAEE-LB scheme for load balancing and congestion reduction, aware of energy efficiency for data routing in wireless sensor networks. The schema uses the information acquired during the route discovery process and considers a composite routing metric to determine the congestion status of a node to apply traffic load balancing or not.

6. CHALLENGES AND RESEARCH OPPORTUNITIES

The use of SDN networks has been growing not only in research applications, but also in industry. There are several opportunities still open for optimizing network traffic through the use of SDN. Some of these challenges include: scalability, availability, adaptability and reliability. As the SDN network expands, network drivers become a performance bottleneck because of the large number of signaling messages received and routing requests. To solve

this and other problems related to scalability, decentralized control architecture with fully distributed control plans can be used. Architectures that consider the forms of communication between controllers and the use of programming strategies that employ algorithms to avoid collisions in messages are fundamental. Another possibility is to use a hierarchy model to coordinate controllers, in which the closer to the root, the more global decisions are made [2].

SDN offers a global understanding of current data in the network topology. In this context, it is possible to apply machine learning techniques to make the network smarter and more adaptive. Decisions can be made based on the type and characteristic of the traffic that is received. In this way, the development of machine learning algorithms optimizes resources, increases the performance and security of the network, joining the world of networks with artificial intelligence. With the growing heterogeneity of applications and requirements in networks, it is important to find ways to simultaneously satisfy different QoS metrics. SDN offers intelligent routing and along with virtualization capabilities, you can separate the network into slices or zones to isolate the different needs. Every need with QoS policy most appropriate for your nature. For example, for real-time video transmission, packet loss can be tolerated, but high bandwidth for data transmission is important, while a network of data capture sensors in an industry can reverse. Thus, transparent mechanisms that combine network metrics such as baud rate, bandwidth, jitter, delay, and packet loss are expected.

It is observed that the attention and application of the vast majority of the research is focused on networks of data centers, cloud computing environments, sensor networks and video transmission. Different applications can be exploited through the use of SDN. Efforts focused on other applications involving large volumes of data and different types of needs, such as smart cities and mobile networks, will be important. Finally, the transition from traditional network architectures to SDN-based architectures is also an open question. Despite the emergence of some network devices with native OpenFlow support, it is difficult to replace all the hardware that makes up the global network fabric. In this way, mechanisms, protocols and interfaces are required that allow the coexistence between both architectures to enable and accelerate the adoption, deployment and globalization of software-defined networks.

CONCLUSIONS

SDN is a technology that is growing and gaining strength, both in academia and industry. By decoupling the control plane and data, it becomes possible to program the network centrally and efficiently, without having to deal with proprietary low-level languages. This simplifies the network and allows data traffic to be controlled intelligently and with fast action considering current network conditions. As a result, data traffic optimization occurs as granular service quality policies, dynamic load balancing, and congestion control exist.

In this article, we examine different methods and approaches being used by researchers to solve problems in optimizing data traffic in networks through the use of SDN. We explore jobs that propose different ways to deal with quality of service, congestion control and load balancing. Finally, we point out future research challenges and opportunities within this context, which are currently mainly related to scalability, availability, adaptability, reliability and security

REFERENCES

[1] HAKIRIA, A. et al. Software-defined networking: Challenges and research opportunities for future internet. Computer Networks, v. 75, p. 453-471, Oct 2014.

[2] HU, F .; HAO, Q .; BAO, K. A survey on software-defined network and openflow: From concept to implementation. IEEE Communications Surveys & Tutorials, v. 16, p. 2181-2206, may 2014.

[3] NUNES, B. A. A. et al. A survey of software-defined networking: Past, present, and future of programmable networks. IEEE Communications Surveys & Tutorials, v. 16, p. 1617-1634, Feb. 2014.

[4] LARA, A .; KOLASANI, A .; RAMAMURTHY, B. Network innovation using openflow: A survey. IEEE Communications Surveys & Tutorials, p. 493-512, Feb. 2014.

[5] MCKEOWN, N. et al. Openflow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, v. 38, p. 69-74, Mar 2008.

[6] OPENFLOW deployments. 2016. Available in: <http://archive.openflow.org/wp/current-deployments/>. Accessed on Sep 28. 2017.

[7] XIA, W .; NIYATO, D. A survey on software-defined networking. IEEE Communications Surveys & Tutorials, v. 17, p. 27-51, Mar 2015.

[8] JARRAYA, Y .; MADI, T .; DEBBABI, M. A survey and a layered taxonomy of software-defined networking. IEEE Communications Surveys & Tutorials, v. 16, p. 1955-1980, apr 2014.

[9] NGUYEN, X.-N. et al. Rules placement problem in openflow networks: A survey. IEEE Communications Surveys & Tutorials, v. 18, p. 1273-1286, dec 2015.

[10] WANG, S .; LI, D .; XIA, S. The problems and solutions of network update in sdn: A survey. Computer Communications Workshops (INFOCOM WKSHPS), p. 474-479, Aug. 2015.

[11] BRERETONA, P. et al. Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software, v. 80, p. 571-583, apr 2007.

[12] JINYAO, Y. et al. Hiqos: An sdn-based multipath qos solution. China Communications Magazine Co. Ltd., v. 12, p. 123-133, Jun 2015.

[13] TARIQ, S .; BASSIOUNI, M. Qamo-sdn: Qos aware multipath tcp for software defined optical networks. Consumer Communications and Networking Conference (CCNC), p. 485-491, jul 2015.

[14] YANG, H. et al. Experimental demonstration of latency-aware software defined networking for openflowbased intra-datacenter optical interconnect networks. Information and Communications Technology, Mar 2013.

[15] LLOPIS, J. M .; PIECZERAK, J .; JANASZKA, T. Minimizing latency of critical traffic through sdn. Networking, Architecture and Storage (NAS), p. 420-423, Aug 2016.

[16] U-Chupala, P. et al. Application-oriented bandwidth and latency aware routing with openflow network. Cloud Computing Technology and Science (CloudCom), Feb 2015.

[17] GOEL, P .; SRINIVASAN, L .; VARMA, V. An adaptive routing scheme for heterogeneous data-flows using openflow. Cloud Computing in Emerging Markets (CCEM), Nov 2015.

[18] GHOLAMI, M .; AKBARI, B. Congestion control using openflow in software defined data center networks. 19th International ICIN Conference - Innovations in Clouds, Internet and Networks, Paris, p. 1-5, Mar 2016.

[19] HERTIANA, S. N .; KURNIAWAN, A. A joint approach to multipath routing and rate adaptation for congestion control in openflow software defined network. Wireless and Telematics (ICWT), Nov 2015.

[20] SONG, S. et al. A congestion avoidance algorithm in sdn environment. Information Networking (ICOIN), p. 420-423, Jan 2016.

[21] JOUET, S .; PERKINS, C .; PEZAROS, D. Otcp: Sdn-managed congestion control for data center networks. Network Operations and Management Symposium (NOMS), p. 209-225, apr 2016.

[22] MALLIK, A .; HEGDE, S. A novel proposal to effectively combine multipath data forwarding for data center networks with congestion control and load balancing using software-defined networking approach. Recent Trends in Information Technology (ICRTIT), apr 2014.

[23] REGO, P.A. L. et al. An openflow-based elastic solution for cloud-cdn video streaming service. Global Communications Conference (GLOBECOM), dec 2015.

[24] SELVI, H. et al. Load balancing in openflow-enabled switches for wireless access traffic aggregation. Network Operations and Management Symposium (NOMS), p. 1013-1014, apr 2016.

[25] AL-NAJJAR, A .; LAYEGHY, S .; PORTMANN, M. Pushing sdn to the end-host, network load balancing using openflow. International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), Sydney, p. 1-6, Mar 2016.

[26] MOGHADAM, M. N .; TAHERI, H .; KARRARI, M. Minimum cost load balanced multipath routing protocol for low power and lossy networks. Wireless Networks, v. 20, nov 2014.

[27] SUNITHA, G. P .; KUMAR, B. P. V .; KUMAR, S. M. D. Optimized congestion aware energy efficient traffic load balancing scheme for routing in wireless sensor networks. Information Processing (ICIP), p. 696-701, dec 2015.

[28] ELDEFRAWY, K .; KACZMAREK, T. Byzantine fault tolerant software-defined networking (sdn) controllers. Computer Software and Applications Conference (COMPSAC), p. 208-213, Aug. 2016.

[29] FOUNDATION, O. N. OpenFlow. 2016. Available in: <http://archive.openflow.org/>. Access on 15 Sep. 2017.

[30] FOUNDATION, O. N. ONF. 2016. Available at: <https://www.opennetworking.org/>. Access on 15 Sep. 2017.

[31] TOOTOONCHIAN, A .; GANJALI, Y. Hyperflow: a distributed control plane for openflow. Proceedings of the 2010 internet network management conference on Research on enterprise networking, p. 3-6, apr 2010.

[32] Koponen, T. et al. Onix: A distributed control platform for large-scale production networks. OSDI, v. 10, p. 1-14, oct 2010.

[33] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.0.0 (Wire Protocol 0x01). 2016. Available at: <http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf>. Access on 21 Sep. 2017.

[34] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.1.0 (Wire Protocol 0x02). 2016. Available at: <http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf>. Access on 20 Sep. 2017.

[35] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.2 (Wire Protocol 0x03). 2016. Available at: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow / openflow-spec-v1.2.pdf>. Access on 15 Sep. 2017.

[36] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.3 (Wire Protocol 0x04). 2016. Available at: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/ openflow / openflow-spec-v1.3.0.pdf>. Access on 14 Sep. 2017.

[37] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.4.0 (Wire Protocol 0x05). 2016. Available at: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/ openflow / openflow-spec-v1.4.0.pdf>. Access on 15 Sep. 2017.

[38] FOUNDATION, O. N. OpenFlow Switch Specification Version 1.5.0 (Protocol version 0x06). 2016. Available at: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/ openflow / openflow-switch-v1.5.0.noipr.pdf>. Access on 15 Sep. 2017.

[39] SASAKI, S. et al. A proposal of qos-aware power saving scheme for sdn-based networks. International Conference on Network-Based Information Systems, p. 71-77, dec 2015.

[40] DURNER, R .; BLENK, A .; KELLERER, W. Performance study of dynamic management for openflowenabled sdn switches. International Symposium on Quality of Service (IWQoS), p. 177-182, Jun 2015.

[41] ZHONG, H. et al. An efficient sdn load balancing scheme based on variance analysis for massive mobile users. Mobile Information Systems, Oct 2015.

[42] FOUNDATION, O. OpenDaylight Controller. 2016. Available at: <https://www.opendaylight.org/>. Accessed on 17 Sep . 2017.