Performance Analysis of Quality of Service in Software-Defined Networking

Amir Hossein Moravejosharieh

Information Technology Programme, Auckland Institute of Studies, New Zealand

Tel: +64 98455606 Ext. 237 E-mail: amirm@ais.ac.nz

Jaime Lloret

Instituto de Investigacion para la Gestion Integrada de Zonas Costeras, Universitat Politecnica de Valencia, Spain

Tel: +34 609549043 E-mail: jlloret@dcom.upv.es

Received: September 13, 2018 Accepted: December 28, 2018 Published: December 31, 2018

DOI:10.5296/npa.v10i4.14054 URL: https://doi.org/10.5296/npa.v10i4.14054

Abstract

Software-Defined Networking (SDN) is a new networking strategy designed to overcome issues experienced in traditional IP network e.g. high level of complexity and inability to adaptively respond to newly arisen Quality of Service (QoS) requirements in a timely fashion. In SDN, control plane and data plane are decoupled which justifies the need to have a central controller to receive the application requirements (e.g. Quality of Service requirements) and implements a set of network policies on the data plane to eventually satisfy the requirements of the application. Implementing a proper set of policies on data plane can be quite a challenging task. In many cases implementing a set of policies in order to satisfy the requirements of an application negates requirements of other applications. In this paper, a simulation study is conducted to evaluate the performance of a QoS policy (i.e. reserving bandwidth) on a specific type of multimedia traffic (e.g. video, audio and data) and its influences on other types of multimedia traffic. The outcome of the simulation study has motivated the authors to conduct a mathematical analysis on the sensitivity of network applications over all possible combination of network policies to eventually implement a proper set of policies that imposes minimum destructive impact on other network applications or services.

Keywords: Bandwidth Reservation, Performance Evaluation, Quality of Service, Software Defined Networking

1 Introduction

This paper introduces an extension to our previous study [1]. Providing acceptable level of Quality of Service (QoS) for different network elements such as end-users, network applications and types of traffic has again drawn research communities attention. Internet Service Providers (ISPs) and network equipment vendors have been competing to attract users by offering higher QoS for different types of media e.g. audio, video and data. However, due to large number of limitations imposed by the nature of traditional IP networks, offering high level of QoS has become a difficult task. For example, traditional IP networks are very complex and very hard to manage and maintain. More importantly, traditional IP network is unable to deal with changes and newly-arisen QoS requirements in a timely fashion. Such failures are mainly caused by the nature of traditional IP networks in which control plane and data plane are vertically coupled together and implemented in network devices [2].

The limitations and constraints experienced in traditional IP networks have motivated research communities to design a new networking paradigm in order to resolve IP network's shortcomings. Software-Defined Networking (SDN) is a new emerging network strategy in which control plane and data plane are separated from each other and their interaction is controlled by a central controller called SDN controller [3]. This separation has made the data plane to become forwarding devices that are instructed by the SDN controller on how to deal with different network traffic. More specifically, the SDN controller receives the network application requirements, translates them into low-level command lines and implements them on data plane. This networking strategy turns out to an effective solution for implementing new network policies in order to satisfy the needs of network applications, hence achieving higher end-user satisfaction [4]. The communication between network application, controller and data plane is accomplished by the use of well-defined programming interfaces. For example, OpenFlow [5] is a widely accepted interface that acts as a medium between the SDN controller and data plane (forwarding devices).

In this paper, the main focus is on providing acceptable level of QoS for those applications or media services (different types of traffic e.g. audio, video and data) that are quite sensitive to variation of QoS policies. Clearly, implementing a set of QoS policies to meet the requirements of a specific type of traffic may conflict with the requirements of other QoS-related network applications or media services. For example, a QoS policy that is meant to increase the network throughput for a certain traffic may increase the experienced packet loss rate for other types of traffic. In some other cases, decreasing the end-to-end latency for a certain type of traffic may strongly impact the successful transmission of same and other types of traffic. Last but not least, increasing the bandwidth utilisation for a certain traffic may cause experiencing prolonged delay for other traffic e.g. business applications that are in need of higher bandwidth for critical transactions. Clearly, implementing a proper set of QoS policies that meet the QoS requirements of *all* QoS-sensitive application can be quite challenging. In an ideal SDN-enabled network, whenever the SDN controller enforces a QoS policy, it has to make sure that the outcome of the implemented policy has met the QoS requirements of the application as well as incurring minimum negative impact on the other QoS-sensitive applications such as other types of media traffic. In the case of conflict between the outcome of applying a number of QoS policies, the SDN controller has to choose a new subset of policies to minimise the impact of the conflict. Implementing a subset of QoS policies on data plane and analysing the outcome may repeatedly take place until the best subset of QoS policies is determined. Clearly, performing this repeated course of actions contradicts with the nature of QoS-sensitive applications, particularly real-time applications.

This paper consists of two major sections: The first section conducts a simulation study to provide an example on how the SDN controller enforces a QoS policy (i.e. bandwidth reservation) to satisfy the QoS requirement of a certain type of media traffic followed by a discussion that highlights the imposed issues on the QoS requirements of other types of media. The obtained results from the conducted simulation study has motivated the author to provide a mathematical analysis on the sensitivity of QoS-related network applications over all possible combinations of QoS policies and also the probability of implementing the best combination of QoS policies on data plane in the first try. Therefore in the second section, this paper proposes to employ "2^k factorial design to evaluate the sensitivity of QoS-sensitive network applications which enables the SDN controller to choose the best available set of QoS policies to implement on data plane in the first try.

The remainder of this article is as follow: Section 2 provides a brief overview of SDN structure and the relationship between control and data planes while Section 3 discusses several related works in regards with improving QoS in the SDN enabled networks. Section 4 explains the system model used for the simulation study followed by comprehensive discussion on the performance of bandwidth reservation – as a QoS policy – and its influence on the target and other QoS-related media service traffic. 2^k factorial design and its incorporation with QoS policies in SDN is discussed in Section 5. Finally, the conclusion is presented in Section 6.

2 SDN Structure and Background

Generally, SDN is designed to provide network and system engineers with an opportunity of programming the logical behaviour of a computer network. Such an opportunity allows us to perform better network traffic measurement, thus enhancing network traffic management. Considering the strong relationship between the network traffic management strategies and QoS requirements, it is safe to state that enhanced network traffic management results in better satisfying QoS requirements of different types of media services and applications. In SDN, the network's control logic is completely separated from the forwarding devices. More specifically, a controller (central element) receives/collects specific application's or media service's requirements (e.g. QoS requirements such as throughput) in the form of high-level language programs, converts them into low-level command lines and instructs them on forwarding devices, thereafter [5], [6], [7] and [8].

OpenFlow is one of the most popular frameworks that closely implements SDN and it is available as an open standard. The SDN structure defined in OpenFlow framework contains forwarding devices which are called OpenFlow switches and the central controller is then called OpenFlow controller. The main responsibilities of an SDN controller are making routing decisions, receiving provisioning QoS, security and load-balancing requirements as well as instructing new set of configurations on forwarding devices (OpenFlow switches).

Fig. 1 illustrates three layers that are mainly considered in SDN structure, namely: Application layer, Control layer and Infrastructure layer. The application layer is mainly responsible to maintain the application requirements e.g. special routing requirements, bandwidth allocation, security and QoS enforcement policies. The control layer, however, consists of a central controller (also known as network operating system) that receives the high-level application re-

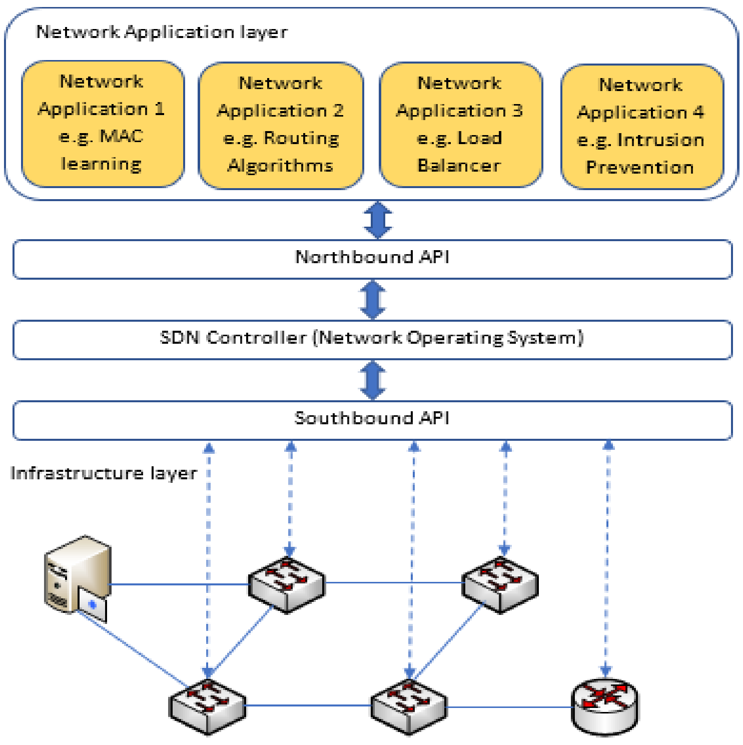
Network Application layer
Network Application 1 e.g. MAC learning
Network Application 2 e.g. Routing Algorithms
Network Application 3 e.g. Load Balancer
Network Application 4 e.g. Intrusion Prevention
Northbound API
SDN Controller (Network Operating System)
Southbound API
Infrastructure layer

Figure 1: Layered structure of software-defined networking

quirements from the application layer, translates them to low-level command lines and installs a new set of configurations on the forwarding devices. The infrastructure layer consists of forwarding devices whose actions are dictated by the SDN controller. Moreover, there are two main interfaces that enable communication between specific layers, namely: Northbound and Southbound interfaces. Northbound and Southbound interfaces are Application Programming Interfaces (APIs) that allow the controller to communicate with application and infrastructure layers, respectively. For example, when a controller receives the abstract high-level application requirements (e.g. QoS requirements) via northbound interface, the controller employs/defines necessary communication policies and protocols to be implemented on forwarding devices [9] and [10]. Once the relevant policies are converted to low-level command lines, the controller uses the southbound interface to communicate with forwarding devices and instructs them accordingly [11]. In the above-mentioned SDN structure, a traffic flow is defined as a sequence of packets between a source and a destination and the forwarding decisions are made using flow-based approach rather than destination-based [12] and [13]. In an SDN-enabled network, traffic flows are abstract which unifies the behaviour of forwarding devices. This means that all packets of a flow are provisioned to receive similar treatment by the forwarding devices [14].

To further elaborate the QoS requirements and the relevant policies, we first need to discuss Traffic Engineering (TE) in SDN-enabled networks. Generally, there are two main terms discussed in TE, namely: traffic management and traffic measurement. There are specific activities and processes involved in either of the above-mentioned terms. Activities and processes such as traffic monitoring, measuring and obtaining network status information (e.g. end-to-end latency, network resource utilisation, packet loss ratio and other traffic metrics) and network topology structure mainly fall under network traffic measurement category. While network traffic management includes the actions taken on the information provided by the network traffic measurement. More specifically, network traffic management includes activities such as managing network traffic via implementing relevant network policies and schedule network resources (e.g. bandwidth reservation) aiming to satisfy application requirements e.g. QoS. For example, the SDN controller periodically or on-demand collects QoS information (e.g. end-to-end delay) from data plane and determines the paths or segments of network that offer shorter delay. Then, as soon as a business critical application initialises to establish a communication with a destination, the SDN controller uses the acquired QoS information (subject to the validity of information) to implement the relevant QoS policies on data plane and instruct them to comply with the new QoS requirements considered for the business application traffic. To sum up, the SDN controller uses TE to effectively govern the entire SDN-enabled network. The collaboration between network traffic measurement and network traffic management is shown in Fig. 2.

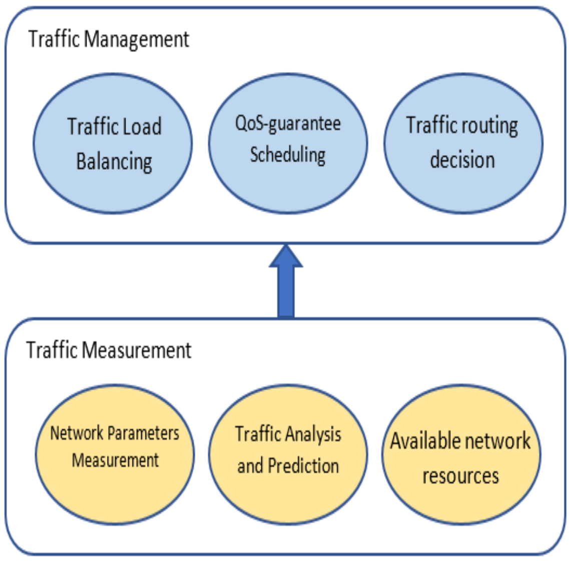

Figure 2: Simplified TE in SDN

3 Related Work

This section reviews studies and related works conducted on QoS for multimedia traffic (video and audio) in SDN as these types of traffic are quite sensitive and require thorough scrutiny.

Authors in [15] have proposed an OpenFlow architecture that guarantees the QoS for the scalable video encoding. This architecture introduces two concurrent routing schemes performed by the SDN controller. In the first routing scheme, the SDN controller transmits the scalable encoding traffic to analyse the QoS of different routs. In the second routing (also known as best effort routing), the SDN controller simply determines the shortest path to the destination peer. Finally, the SDN controller, based on acquired network status information, determines the best path for a pair of source-destination. Their obtained results showed that other traffic including best effort routing were strongly influenced by transmission of the scalable video encoding in first routing scheme which caused significant amount of packet losses. In another experiment [16], the scalable video encoding and best effort routing approaches have been separately used in two different scenarios to analyse the performance of commonly-used routing algorithms for multimedia in SDN when dealing with QoS-related traffic. The results, however, shows relatively some QoS improvement for multimedia traffic in either scenarios.

Another approach to guarantee the QoS for multimedia traffic is to reserve/allocate bandwidth and allocate it when needed. In [17] a SDN controller analyses the collected network status information to prioritise traffic along with performing smart traffic management. As a result of this analysis, the SDN controller implements relevant QoS policies on data plane in order to satisfy the QoS requirements of the multimedia traffic. Multimedia Gateway proposed in [18] is another QoS-guaranteed approach in which network traffic is identified and classified according to its type of service (e.g. audio, video and data). The SDN controller then forwards each class of service towards the destination according to corresponding bandwidth reservation and priority policies. Constantly monitoring network traffic allows the SDN controller to allocate certain network resources for multimedia traffic such as re-directing video streams to servers with lower loads [19] and re-route the video streams when a server is overloaded. Classifying the communication paths based on their speed into fast-lane and slow-lane could be another approach to improve the QoS for different multimedia traffic. In [20] authors argued that allocating fast-lane communication paths to the sensitive multimedia traffic and slow-lane communication paths to non-sensitive traffic (e.g. web page loads) significantly improves the QoS for video streams. Another example of classifying multimedia traffic is “FlowQoS” proposed in [21] in which multiple classifiers (i.e. SDN-based limiter, DNS-based limiter and flow classifier) collaborate with each other to identify different types of service type and allocate network resources accordingly.

Enhancing the Dijkstra algorithm (finding the shortest path between a source and destination) and changing its service model also improved the QoS of multimedia traffic [22] and [23]. Other approaches such as QoS routing [24], [25] and [26] and enhanced IntServ Model [27] have also studied the QoS of multimedia traffic in SDN and proposed a number of approaches to improve the end-to-end delivery of QoS for multimedia traffic. Artificial Intelligence (AI) incorporated in SDN is another approach in which reinforcement learning is employed to use network status and determine the best path between sender-receiver pairs [28]. In another case authors in [29] proposed to use an artificial intelligence system for detecting and correcting errors in multimedia transmission in a surveillance IoT environment connected through an SDN-enabled network.

While the discussed related works have improved the QoS of different types of media in SDN-enabled networks, there is none or very small amount of research on the adaptability of SDN controller when it is dealing with varying QoS requirements for different network applications or types of media. In this paper, a simulation study is conducted to evalate the adaptibility of a SDN controller in terms of considering the QoS reqirements of different types of media, simultaneously. Moreover, this paper proposes a mathematical approach to determine the sesitivity of different types of services or applications over all possible combinations of QoS policies and choose the best combination if required.

4 Performance Evaluation of Bandwidth Reservation

This section presents a simulation study to analyses the performance of the bandwidth reservation – as a popular QoS policy – and evaluates its influence on the QoS requirement (i.e. throughput) of other traffic.

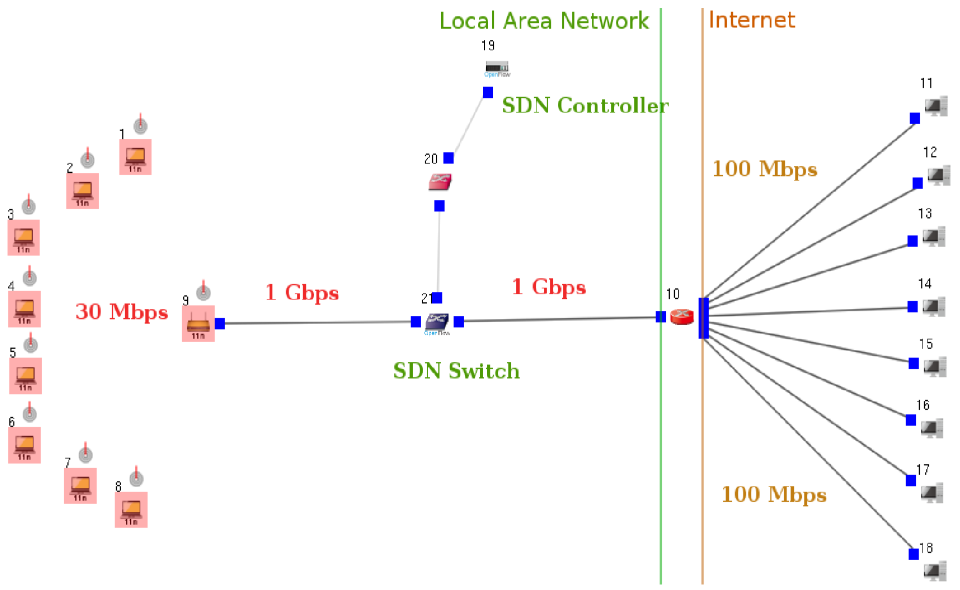

Figure 3: Generic network structure

4.1 System Model

Prioritising different types of media services and reserving bandwidth for them accordingly seems to be an effective approach to improve the QoS for multimedia traffic. This section evaluates the performance of bandwidth reservation for a certain type of service and analyses its influence on other types of services. More specifically, two types of prioritisation schemes are used to evaluate the performance of bandwidth reservation, namely: flow-based and user-based. In former prioritisation scheme, the different types of traffic flow (e.g. UDP and TCP) are prioritised with different priority values, while in the latter scheme, certain users experience different priority values.

A generic network structure that contains both wired and wireless sub-networks is designed. In this paper we mainly focus on the wireless sub-network. Fig. 3 presents the considered generic network structure that includes both SDN-enabled network (wireless sub-network) and traditional IP network (wired sub-network). To simulate such network structure, Estinet 10 network simulator is used [30] and [31]. While authors have employed Estinet 10 as their SDN simulator, other alternatives are discussed in [32].

The wireless sub-network contains an OpenFlow controller (Ryu SDN controller), an OpenFlow switch controller that acts as a medium between SDN controller and an OpenFlow switch, an Access Point (AP) and eight end-users who are connected to the AP and are configured to act as receiver nodes. While a router and eight other end-users (configured as senders) have formed the wired sub-network. To keep the simulation study as generic as possible, packet sizes for UDP and TCP traffic as well as packet generating intervals are drawn randomly using uniform distribution. In this simulation study, each sender communicates to only one receiver and only the incoming throughput has been monitored for both flow-based and user-based prioritisation schemes. Note that the incoming throughput is defined as the number of bits per second that is successfully reached a receiver node and the acknowledgement is also received by the corresponding sender. Table 1 shows the network specifications as well as their considered values.

In the wireless sub-network, the bandwidth is shared among all receiver nodes. Therefore, prioritising different service flows and/or users would influence the level of QoS experienced by them. The following explains four different scenarios in which the SDN controller prioritises either service flows or different users and allocates the bandwidth to them accordingly:

Scenario one: this scenario focuses on the flow-based prioritisation scheme where TCP traffic is the only communication traffic between senders and receivers. The priority of the TCP traffic for all receiver nodes is configured to priority 1 which is the lowest priority. This scenario is considered to act as a benchmark where prioritisation schemes are not practiced. Scenario two: in this scenario TCP is the only communication traffic between senders and receivers. However, the priority of receiver nodes differs. More specifically, the priorities of receiver nodes 1 and 2 are set to the highest priority value 7, nodes 3 to 6 are configured to priority 3 (middle priority value) and nodes 7 and 8 are configured to 1 which is the lowest priority in this simulation study. This scenario allows us to study the behaviour of bandwidth reservation when user-based prioritisation scheme is employed. Scenario three: in this scenario all receiver nodes are prioritised to 7 (highest priority value). Here also TCP is the only communication traffic in the entire network. This scenario indicates the behaviour of bandwidth reservation approach when the number of end-users who are demanding the highest priority becomes larger. Scenario four: In this scenario both UDP and TCP communication traffic have been used as communication traffic between different pairs of sender-receiver. More specifically, nodes 1 and 2 are configured to use UDP traffic for the communication purposes with their peers while enjoying the highest priority. The configuration for the rest of the nodes remains similar to scenario 2. This scenario presents the behaviour of the bandwidth reservation approach when flow-based prioritisation scheme is employed.

In this simulation study, all sender nodes are only responsible to generate traffic and transmit it to their peers. For each scenario, the minimum of 10 simulation runs have been carried out. Extra simulation runs were added to reach the 95% confidence interval level with the half-width no larger than 5% of the confidence interval. Each simulation run is configured to 300 seconds and the average of incoming throughput is calculated to evaluate the performance of the bandwidth reservation.

Table 1. Network parameters specifications

Parameter	***Value***
TCP packet size	66 bytes - 1514 bytes
UDP packet size	132 bytes - 1066 bytes
TCP packet generating interval	0.109 ms - 989 ms
UDP packet generating interval	2.16 ms - 90 ms
Wireless net. Total bandwidth	30 Mbps
Wired net. total bandwidth	100 Mbps
AP-OF switch-router link delay	0.5 ms
Access Point transmit power	-16 dbm
Max data rate supported by Access Point	150 Mbps
Frequency bands used by Access Point	2.4 GHz and 5 GHz
Client connection link delay	5 ms
Flow priority range	1-7 (lowest-highest)

4.2 Results and Discussion

Four scenarios have been considered in which the incoming throughput – as one of the most important QoS parameters – is calculated to evaluate the performance of bandwidth reservation approach in SDN. The obtained result of scenario one is presented in Fig. 4. It shows that when all nodes are configured to the lowest priority value (no priority for all nodes) the average of incoming throughput for all the receiver nodes is approximately 400 kbps. Fig. 5, however, shows the obtained result of scenario two where different receiver nodes experienced different priority values. This figure clearly shows that those receiver nodes that are prioritised to the highest priority are treated differently by the AP in terms of the amount of allocated bandwidth and ended up experiencing almost 2000 kbps throughput compared to middle and the lowest priorities for which the average experienced throughput is approximately 350 kbps. The experienced throughput in scenario three is presented in Fig. 6. As a reminder, in this scenario, all receiver nodes are configured to the highest priority value 7. This scenario indicates a situation where a large number of nodes (in our case all of receiver nodes) is in need of high level of QoS. The experienced throughput for all receiver nodes in this scenario, is on average around 800 kbps as AP allocated equal amount of bandwidth to all receiver nodes. The experienced throughput values obtained from scenario one and three shows 400 kbps difference when all receiver nodes are configured to the highest priorities in scenario three. However, 800 kbps throughput would not be considered guaranteed high QoS. So far, TCP protocol was the only communication traffic between senders and their corresponding receivers and user-based prioritisation was the only focus in these three scenarios. Scenario four, however, mainly focuses on the performance of flow-based prioritisation scheme in bandwidth reservation approach in which the UDP traffic flow destined to receiver nodes 1 and 2 is prioritised to the highest, while the TCP traffic destined to nodes 3 to 6, 7 and 8 are prioritised to the middle and the lowest, respectively. Fig. 7 depicts the average experienced throughput for all receiver nodes in scenario four. The experienced throughput in scenario four presents significant difference between prioritising UDP traffic compared to TCP traffic which is mainly due to connection-less nature of UDP traffic and its relatively higher packet generation ratio. Moreover, prioritising UDP traffic has significantly influenced the throughput gain of TCP traffic end-users in SDN-enabled networks. This is mainly due to reserving a large portion of the bandwidth for UDP traffic (e.g. video-conferencing) when seamless connectivity is a must. This reservation has increased the throughput of UDP traffic users to approximately 4000 kbps while it has strongly influenced the achievable throughput of other types of traffic which is almost 3 kbps for TCP traffic users in our case.

This simulation study is one simple example to show how implementing a network policy to satisfy the QoS requirement of a multimedia traffic influences other type of multimedia traffic. Therefore, there is a tangible need to verify the impact of various network policies on network applications and use this information to implement the best available combination of network policies with the minimum possible negative impact on other network applications or services.

5 Proposed Approach

The results obtained from the simulation study clearly shows that fulfilling the QoS requirements of a target media type or an application may destructively impact the QoS requirements of other services and/or applications. More specifically, the SDN controller, by default,

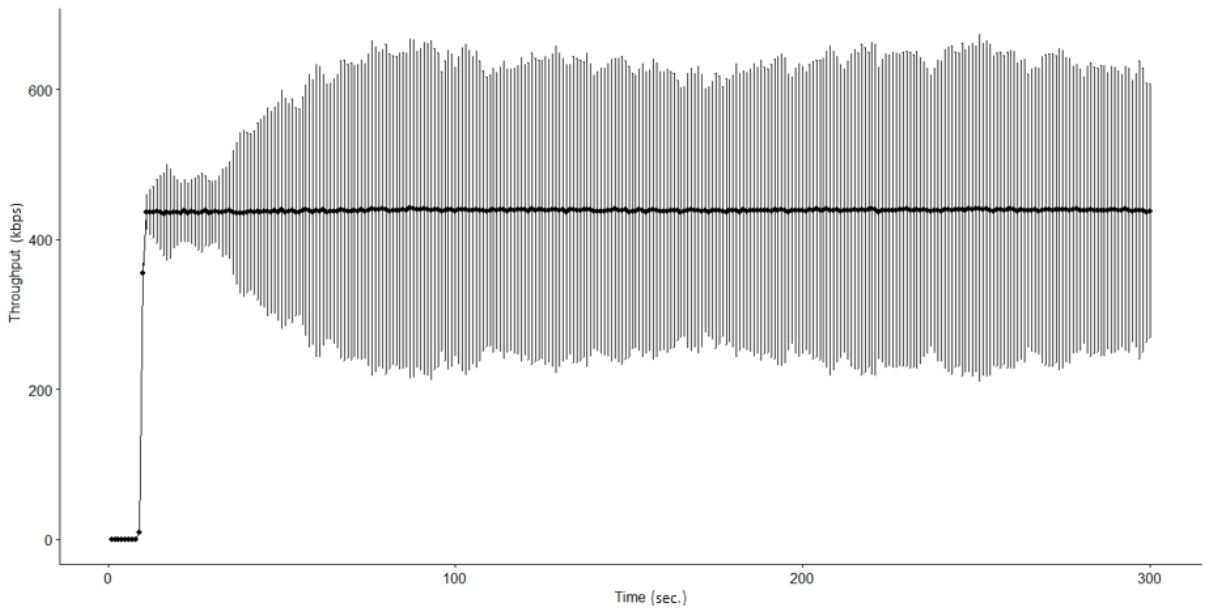

Figure 4: Incoming throughput experienced in the scenario one:all nodes are prioritised to the lowest priority value 1. The communication traffic is TCP

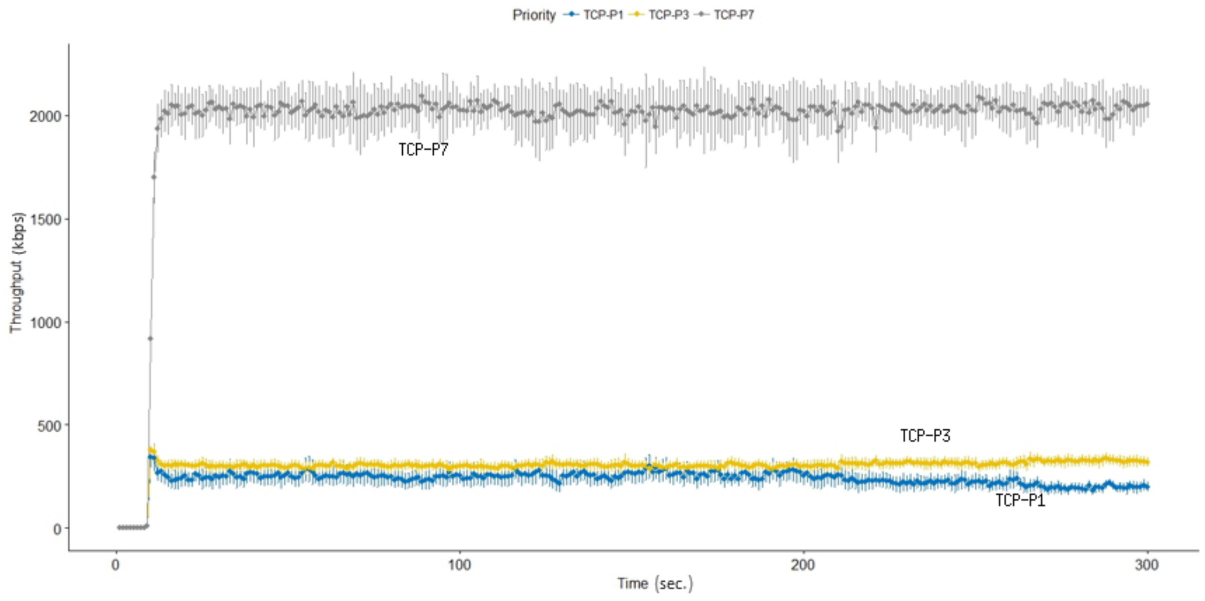

Figure 5: Incoming throughput experienced in the scenario two: nodes are prioritised to the lowest, medium and highest priority values 1, 3 and 7, respectively. The communication traffic is TCP

is not capable of intelligently choosing the best set of QoS policies to satisfy the QoS requirements of a target type of media service and at the same time incurring minimum negative impact on other services. Note that an SDN controller equipped with complex programming codes may result in implementing policies with lesser impact on other types of services or applications. However, the complexity of the program installed in SDN controller must be able to handle newly-arisen QoS requirements, policies, types of services and/or applications. To implement the best available QoS policy with minimum destructive impact on other QoS-related applications with different QoS requirements, the SDN controller is required to implement all

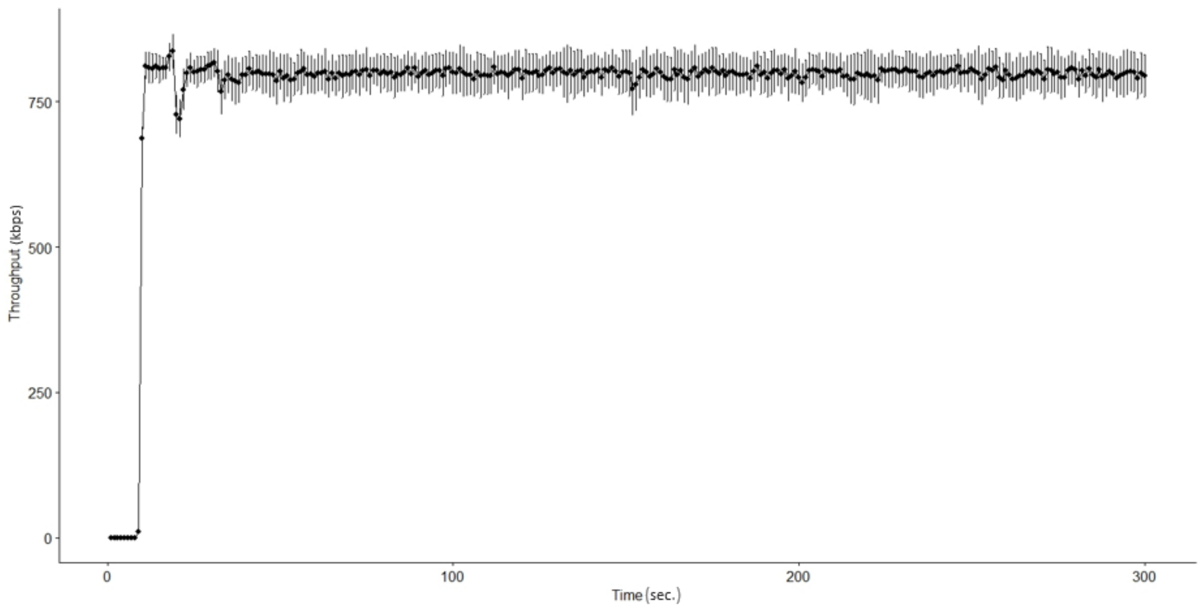

Figure 6: Incoming throughput experienced in the scenario three: all nodes are prioritised to the highest priority value 7. The communication traffic is TCP

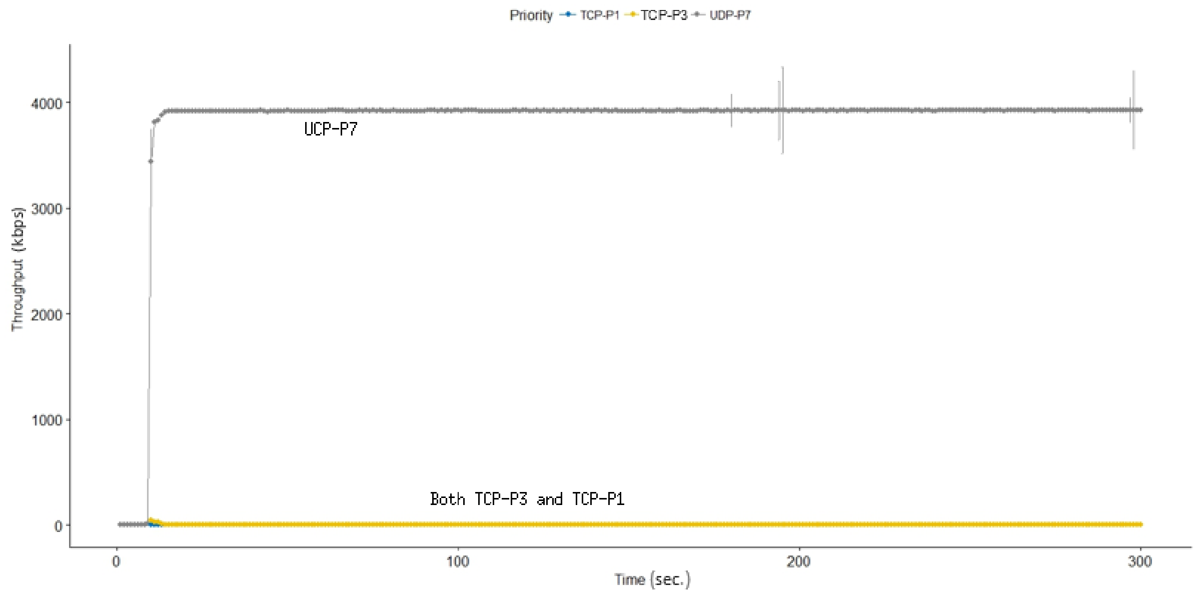

Figure 7: Incoming throughput experienced in the scenario four: some nodes are configured to the lowest and medium priority values 1 and 3 respectively, with TCP as their communication protocol while other nodes are configured to the highest priority value 7 with UDP as their communication protocol

possible combinations of QoS policies until it finds the best combination of QoS-policies that works out for all QoS-related applications or at least incurs the minimum destructive impact on other QoS-related applications or media type services. This procedure is not only very time consuming but also it is highly likely that it causes network instability and eventually users dissatisfaction.

A way to find out the best set of QoS policies and their influences on different types of media services and/or QoS-related applications is to determine the *sensitivity* of QoS-related applications over all possible combinations of QoS policies for different types of media services. 2^k factorial design is a mathematical method to determine the sensitivity of a set response variables over all possible combinations of system parameters [33, 34].

5.1 2^k Factorial Design

In this section, the 2^k factorial design – incorporated with QoS policies in SDN-enabled network – is introduced as an approach to determine the impact of all possible combinations of QoS policies on QoS requirements of a target type of media service. Additionally, 2^k factorial design attains the most influential QoS policies on a set of target QoS-related applications which allows system engineers to better design and adjust QoS policies, accordingly. An analytical study in conducted to determine the efficiency of an SDN-enabled controller using 2^k factorial design compared to a typical SDN controller.

2^k factorial design presents a mathematical relationship between a set of independent variables (QoS policies, e.g. bandwidth reservation with prioritisation scheme) and a set of dependent variables (QoS-related applications, e.g. throughput). More specifically, a dependent variable (such as throughput or end-to-end latency) $y \in \mathcal{R}$ is presumably under influence of variations of independent variables $p \in \mathcal{R}$ and the impact of this influence has been determined over n number of observations while k simply presents the number of QoS policies participating in this equation. Therefore, the relationship between QoS-related application and a set of QoS policies would be presented as (1):

$$y = g(p_1, p_2, ..., p_k) + \epsilon \tag{1}$$

where $g()$ represents the function of the system that relates QoS-related application or service y to a set of QoS policies $p_1, p_2, ..., p_k$ and ϵ is considered the error of the model. Note that the QoS policies may have measures in their physical units must be converted to their dimensionless quantities (coded format) with same characteristics i.e. standard deviation and zero mean values. Therefore, we have a regression model as shown below ((2)):

$$\hat{y} = \alpha_0 + \sum_{i=1}^{k} \alpha_i p_i + \sum_{i=1}^{k} \alpha_{ii} {p_i}^2 + \sum \sum_{i<j} \alpha_{ij} p_i p_j \tag{2}$$

where $\hat{y}$ is a candidate QoS-related application, p_i and p_j are the QoS policies (after conversion to their coded format) and $\alpha_0, \alpha_i, \alpha_{ii}, \alpha_{ij}$ are the coefficient of intercept, linear, quadratic and interaction, respectively which are calculated after n simulation or experimental runs. Depending on the number of QoS policies k participating in this regression model, 2^k combinations of QoS policies are constructed that each of which combination $(p_1, \ldots, p_k) \in \{-1, 1\}^k$ may influence in a different way on a set of QoS-related applications or services $y_{p_1,\ldots,p_k}$.

The coefficients α mentioned in (2) present a linear relationship. Therefore, they can all be presented as a vector:

$$\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_k, \alpha_{2,1}, \alpha_{3,1}, \alpha_{3,2}, \alpha_{4,1}, \ldots, \alpha_{k,k-1}) \tag{3}$$

and the QoS-related applications (the dependent variable) as a vector

$$\hat{y} = (y_{(-1,\dots,-1)}, y_{(-1,\dots,-1,1)}, \dots, y_{(1,\dots,1)}) \tag{4}$$

Therefore, the linear system can be presented as (5):

$$\hat{y} = S \cdot \beta \tag{5}$$

where S is called "Sign Matrix". In the sign matrix, each row represents a combination of QoS policies that eventually influences the QoS-related application $y_{p_1,\dots,p_k}$ where $p_i \in \{-1, 1\}$. The sign matrix, after its formation, would look like $(1, p_1, \dots, p_k, p_2{\cdot}p_1, p_3{\cdot}p_2, p_3{\cdot}p_1, \dots, p_k{\cdot}p_{k-1})$. The value for each p_i would be either "1" or "-1". The orthogonality of the sign matrix must be determined either by including terms of the higher degree of interactions i.e. three, four, …, k in (2) or by computing the least square solution for it. In cases where the there are 2^k rows and only $1 + \frac{k(k+1)}{2}$ columns, the linear system of equations is over-determined, thus the least square solution must be computed.

The coefficient of intercept α_0 is the mean value of all observed QoS-related applications as presented in (6):

$$\alpha_0 = \bar{y} = \frac{1}{2^k} \sum_{p \in \{-1,1\}^k} y_p \tag{6}$$

this can be used to calculate "Sum-of-Squares-Total" (SST) which indicates the total amount of variation experienced presented in (7):

$$SST = \sum_{p \in \{-1,1\}^k} (y_p - \bar{y})^2 \tag{7}$$

Considering the orthogonality of the columns in the sign matrix S we have:

$$SST = 2^k(\alpha_1^2 + \ldots + \alpha_k^2 + \alpha_{2,1}^2 + \ldots + \alpha_{k,k-1}^2) \tag{8}$$

hence, the relative impact of a QoS policy i or any interaction of QoS policies i, j can be shown as (9):

$$\frac{2^k \alpha_i^2}{SST} \quad , \quad \frac{2^k \alpha_{i,j}^2}{SST} \tag{9}$$

which is the contribution of each QoS policy in the total observed variation.

To measure the precision of the proposed model, more specifically, how harmful it is to ignore higher degree of interactions, two more measurements must be calculated, namely: "Sum-of-Square-Errors" (SSE) and the coefficient of determination (also known as R^2 value). SSE is a measure to present the total error introduced in the regression process as presented in (10), where x_i is a QoS policy:

$$SSE = \sum_{p \in \{-1,1\}^k} \left(y_p - \left(\alpha_0 + \sum_{i=1}^{k} \alpha_i \cdot x_i + \sum_{i=1}^{k} \sum_{j<i} \alpha_{i,j} \cdot x_i \cdot x_j \right) \right)^2 \tag{10}$$

The coefficient of determination (as shown in (11)), however, is a measure to present the quality of the regression model where larger values present higher quality of the regression model compared to smaller values:

$$R^2 = \frac{SST - SSE}{SST} \tag{11}$$

2^k factorial design is a method to determine the influence of involved QoS policy on QoS-related applications or services. Once the SDN controller determines the impact of each and every QoS policy and their combinations on QoS-related applications, it can easily choose the best combination of QoS policies to satisfy the QoS requirements of a target application while considering the QoS requirements of other running QoS-related applications. Moreover, 2^k factorial design identifies the most influential QoS policies on a target set of QoS-related applications. This is beneficial for the SDN controller for future interactions with the most influential QoS policies.

5.2 Performance Analysis and Discussion

According to full factorial design, with total number of m QoS policies there will be $2^m - 1$ all possible combinations of QoS policies. Each combination may result in experiencing different outcome for the response variables (QoS-related applications or services). Therefore, in a typical SDN-enabled network, the SDN controller needs to employ "sampling without replacement" to eventually find the best combination of QoS policies to be implemented on data plane. More specifically, sampling without replacement determines all possible combinations of w number of QoS policies out of a set of m number of QoS policies which is shown in Equation (12):

$${}^{m}C_w = \binom{m}{w} = \frac{m!}{w! \times (m-w)!}, 0 < w < m \tag{12}$$

The main concern is to determine the only one combination of QoS policies that is beneficial for all QoS-related applications or at least imposes minimum negative impact on Qos-related applications other than the target one. The probability of choosing the only combination of QoS policies out of a pool of all possible combinations is presented in (13):

$$P(w) = \frac{1}{{}^{m}C_w} \tag{13}$$

Now the probability of selecting a combination of $w (0 < w < m)$ and $w \in \{1, 2, 3, 4, ..., m-1\}$ number of QoS policies in the *first try* is shown in (14):

$$P(w) = \frac{1}{\prod_{w=1}^{m} \frac{m!}{w!(m-w)!}} \tag{14}$$

The following example better presents the process of choosing the best combination of QoS policies out of $m = 5$ total number of available QoS policies:

$$\begin{aligned} P(w) &= P(1) \times P(2) \times P(3) \times P(4) \times P(5) \\ &= 1/\left((\frac{5!}{1! \times 4!}) \times (\frac{5!}{2! \times 3!}) \times (\frac{5!}{3! \times 2!}) \times (\frac{5!}{4! \times 1!}) \times (\frac{5!}{5! \times 0!}) \right) = 0.0004 \end{aligned} \quad (15)$$

The probability of choosing the most appropriate set of QoS policies would be the multiplications of the probability of choosing one and the probability of a combination of two and the probability of a combination of three and so on. It can be observed that for a relatively small number of QoS policies, chances are very slim to choose the best set of QoS policies in the first try to satisfy the requirements of a target QoS-related application while imposing minimum impact on other QoS-related applications. More specifically, the SDN controller has to repeatedly implement different combinations of QoS policies to eventually find the best available set of QoS policies. Note that optimisation methods or meta-heuristic approaches (e.g. evolutionary algorithms) have not been considered in this paper due to bringing higher degree of complexity to our proposed approach.

As discussed earlier, 2^k factorial design allows the SDN controller to determine the influence on all possible combinations of QoS policies on a target set of QoS-related applications. Moreover, it enables the SDN controller to store the outcome of each and every combination in a different category. Whenever the SDN controller plans to instruct data plane with a set of QoS policies, it simply picks a category containing a set of QoS policies that – with probability close to one – result in close-to-anticipated outcome.

We define response time as the time taken by the SDN controller to choose the most suitable set of QoS policies, implements them on data plane and receive the corresponding feedback. Relation (16) compares the total response time RT_{total} experienced in a typical SDN-enabled network with the one that employed our proposed approach :

$$RT_{total} \times \prod_{w=1}^{m} {}^{m}C_{w} > RT_{total} \times \prod_{m}^{m} {}^{m}C_{m} \quad (16)$$

Furthermore, the probability of choosing the best set of QoS policies in a typical SDN-enabled network ($P_{Typical}$) compared to an SDN-enabled network that employed our proposed approach ($P_{Proposed}$) is shown in relation (17):

$$\frac{RT_{total}}{P_{Typical}} > \frac{RT_{total}}{P_{Proposed}}, P_{Typical} \ll 1, P_{Proposed} \approx 1 \quad (17)$$

Considering $P_{Typical}, P_{Proposed} and RT_{total}$ in both relations (16) and (17), it can be observed that the total response time experienced in a typical SDN-enabled network is significantly longer compared to an SDN-enabled network that uses our proposed approach. In our proposed approach, the SDN controller chooses the best available set of Qos policies in the first try, converts them to low-level command lines and installs them on data plane. Probably the only main concern about the proposed approach is when a new QoS policy is added to the system. Whenever a new policy is added to the system, its impact alone along with the impact caused by its combination with other QoS policies on QoS-related applications must be determined. This could result in new set of QoS policy categories. However, this limitation can be eliminated if network operators use network simulators to simulate the entire network and perform the sensitivity analysis to obtain new categories and equip the SDN controller with obtained outcome, thereafter.

6 Conclusion

In this paper, a simulation study is conducted to determine the impact of a QoS policy (bandwidth reservation with prioritisation schemes) on two types of network traffic. The obtained results shows that satisfying a QoS requirements of a specific type of network traffic strongly impacts the QoS requirements of other types of traffic. This means that there is a tangible need to choose a set of QoS policies that not only satisfies the QoS requirements of a target set of QoS-related applications but also imposes minimum destructive impact on other QoS-related network applications. In a typical SDN-enabled network, the SDN controller has to take the trial and error approach (at least once) to determine the influence of all possible combinations of QoS policies on considered QoS-related applications. This is not only a time consuming and challenging task but also it is highly likely that this approach causes agitation and network instability. To eliminate this challenge, this paper presented a mathematical analysis that enables the SDN controller to choose the best set of QoS policies, convert them to low-level command lines and instructs them on forwarding devices. More specifically, this paper proposes to exploit 2^k factorial design method to determine the impact of all possible combinations of QoS policies on QoS-related applications and categorise them based on their attained results, accordingly. An analytical performance evaluation is presented to evaluate the probability of choosing the best set of QoS policies in the first try for both typical SD-enabled network and the one using 2^k factorial design. Satisfying the requirements of multiple QoS-related network applications at the same time can be quite challenging. This paper has evaluated the response time of a typical SDN-enabled network environment in which a controller is required to take the trial and error approach to eventually install an appropriate subset of QoS-related network policies on data plane that satisfies the requirements of a target group of QoS-related network applications. This paper has also proposed an approach in which QoS-related network policies will be categorised based on their influences on a target group of QoS-related network applications. An analytical comparison has been provided between a typical SDN-enabled network environment without using our approach and an SDN-enabled network environment that benefits from using our approach in terms of the probability of selecting an appropriate combination of QoS-related network policies in the first try. We showed that our proposed approach, by far, increases the probability of selecting an effective combination of QoS-related network policies, and thus, reducing the overall response time required to fulfill the requirements of QoS-related network applications. As a future work, authors suggest to conduct a simulation or an experimental study to evaluate the performance of an SDN controller when employing 2^k factorial design to determine the sensitivity of time-critical network applications over a set of QoS policies.

Acknowledgements

This work has been partially supported by the ”Ministerio de Economia y Competitividad” in the ”Programa Estatal de Fomento de la Investigacon Cientifica y Tecnica de Excelencia, Subprograma Estatal de Generacion de Conocimiento” within the project under Grant TIN2017-84802-C2-1-P.

References

[1] A. H. Moravejosharieh, M. J. Watts, and Y. Song, “Bandwidth reservation approach to improve quality of service in software-defined networking: A performance analysis,” in *2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, July 2018, pp. 1–6. DOI: 10.1109/JCSSE.2018.8457339

[2] Z. Shu, J. Wan, J. Lin, S. Wang, D. Li, S. Rho, and C. Yang, “Traffic engineering in software-defined networking: Measurement and management,” *IEEE Access*, vol. 4, pp. 3246–3256, 2016. DOI: 10.1109/ACCESS.2016.2582748

[3] F. M. Ramos, D. Kreutz, and P. Verissimo, “Software-defined networks: On the road to the softwarization of networking,” *Cutter IT journal*, 2015.

[4] T. Koponen, M. Casado, N. Gude, and J. Stribling, “Distributed control platform for large-scale production networks,” Sep. 2014, uS Patent 8,830,823. DOI: http://doi.acm.org/10.1145/1355734.1355746

[5] A. Lara, A. Kolasani, and B. Ramamurthy, “Network innovation using openflow: A survey,” *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 493–512, 2014. DOI: 10.1109/SURV.2013.081313.00105

[6] A. Moravejosharieh, M. j. Watts, and K. Ahmadi, “An overview of multimedia QoS in sdn-enabled ip networks,” in *31st Computing and Information Technology Eduacation New Zealand (CITRENZ)*. ITX/CITRENZ, July 2018.

[7] M. R. Parsaei, R. Javidan, A. Fatemifar, and S. Einavipour, “Providing multimedia QoS methods over software defined networks: A comprehensive review,” *International Journal of Computer Applications*, vol. 168, no. 9, pp. 1–4, 2017.

[8] Y. Jarraya, T. Madi, and M. Debbabi, “A survey and a layered taxonomy of software-defined networking,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014. DOI: 10.1109/COMST.2014.2320094

[9] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, “Frenetic: A network programming language,” *ACM Sigplan Notices*, vol. 46, no. 9, pp. 279–291, 2011. DOI: 10.1145/2034574.2034812

[10] A. Voellmy and P. Hudak, “Nettle: Taking the sting out of programming network routers,” in *International Symposium on Practical Aspects of Declarative Languages*. Springer, 2011, pp. 235–249.

[11] P. Saint-Andre, “Extensible messaging and presence protocol (xmpp): Core,” ” 2011.

[12] P. Newman, G. Minshall, and T. L. Lyon, “IP switching ATM under IP,” *IEEE/ACM Transactions on Networking (TON)*, vol. 6, no. 2, pp. 117–129, 1998. DOI: 10.1109/90.664261

[13] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 3, pp. 105–110, 2008. DOI: 10.1145/1384609.1384625

[14] H. Jamjoom, D. Williams, and U. Sharma, "Don't call them middleboxes, call them middlepipes," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 19–24. DOI: 10.1145/2620728.2620760

[15] S. Civanlar, M. Parlakisik, A. M. Tekalp, B. Gorkemli, B. Kaytaz, and E. Onem, "A qos-enabled openflow environment for scalable video streaming," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*. IEEE, Dec 2010, pp. 351–356. DOI: 10.1109/GLOCOMW.2010.5700340

[16] H. E. Egilmez, B. Gorkemli, A. M. Tekalp, and S. Civanlar, "Scalable video streaming over OpenFlow networks: An optimization framework for QoS routing," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011, pp. 2241–2244.

[17] S. Tomovic, N. Prasad, and I. Radusinovic, "SDN control framework for QoS provisioning," in *Telecommunications Forum Telfor (TELFOR), 2014 22nd*. IEEE, 2014, pp. 111–114. DOI: 10.1109/TELFOR.2014.7034369

[18] R. F. Diorio and V. S. Timóteo, "Multimedia content delivery in openflow SDN: An approach based on a multimedia gateway," in *Computational Science and Computational Intelligence (CSCI), 2016 International Conference on*. IEEE, 2016, pp. 612–617. DOI: 10.1109/CSCI.2016.0121

[19] S. Yilmaz, A. M. Tekalp, and B. D. Unluturk, "Video streaming over software defined networks with server load balancing," in *Computing, Networking and Communications (ICNC), 2015 International Conference on*. IEEE, 2015, pp. 722–726. DOI: doi.ieeecomputersociety.org/10.1109/ICCNC.2015.7069435

[20] H. Habibi Gharakheili, V. Sivaraman, T. Moors, A. Vishwanath, J. Matthews, and C. Russell, "Enabling fast and slow lanes for content providers using software defined networking," *IEEE/ACM Transactions on Networking (TON)*, vol. 25, no. 3, pp. 1373–1385, 2017. DOI: 10.1109/TNET.2016.2627005

[21] M. S. Seddiki, M. Shahbaz, S. Donovan, S. Grover, M. Park, N. Feamster, and Y.-Q. Song, "FlowQoS: per-flow quality of service for broadband access networks," Georgia Institute of Technology, Tech. Rep., 2015.

[22] S. Laga, T. Van Cleemput, F. Van Raemdonck, F. Vanhoutte, N. Bouten, M. Claeys, and F. De Turck, "Optimizing scalable video delivery through openflow layer-based routing," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–4. DOI: 10.1109/NOMS.2014.6838378

[23] M. Karl, J. Gruen, and T. Herfet, "Multimedia optimized routing in openflow networks," in *Networks (ICON), 2013 19th IEEE International Conference on*. IEEE, 2013, pp. 1–6. DOI: 10.1109/ICON.2013.6781969

[24] H. E. Egilmez, S. T. Dane, K. T. Bagci, and A. M. Tekalp, “Openqos: An openflow controller design for multimedia delivery with end-to-end quality of service over software-defined networks,” in *Signal & Information processing association annual summit and conference (APSIPA ASC), 2012 Asia-Pacific*. IEEE, 2012, pp. 1–8.

[25] H. E. Egilmez and A. M. Tekalp, “Distributed qos architectures for multimedia streaming over software defined networks,” *IEEE Transactions on Multimedia*, vol. 16, no. 6, pp. 1597–1609, 2014. DOI: 10.1109/TMM.2014.2325791

[26] T.-F. Yu, K. Wang, and Y.-H. Hsu, “Adaptive routing for video streaming with qos support over sdn networks,” in *Information Networking (ICOIN), 2015 International Conference on*. IEEE, 2015, pp. 318–323.

[27] H. Owens II and A. Durresi, “Video over software-defined networking (VSDN),” *Computer Networks*, vol. 92, pp. 341–356, 2015. DOI: https://doi.org/10.1016/j.comnet.2015.09.009

[28] S. Sendra, A. Rego, J. Lloret, J. M. Jimenez, and O. Romero, “Including artificial intelligence in a routing protocol using software defined networks,” in *Communications Workshops (ICC Workshops), 2017 IEEE International Conference on*. IEEE, 2017, pp. 670–674. DOI: 10.1109/ICCW.2017.7962735

[29] A. Rego, A. Canovas, J. M. Jiménez, and J. Lloret, “An intelligent system for video surveillance in IoT environments,” *IEEE Access*, vol. 6, pp. 31 580–31 598, 2018. DOI: 10.1109/ACCESS.2018.2842034

[30] “A professional company in software-defined netowrking (sdn),” Estinet Technologies Inc., Tech. Rep.

[31] S.-Y. Wang, C.-L. Chou, and C.-M. Yang, “Estinet openflow network simulator and emulator,” *IEEE Communications Magazine*, vol. 51, no. 9, pp. 110–117, 2013. DOI: 10.1109/MCOM.2013.6588659

[32] J. M. Jimenez, J. O. Romero Martínez, A. Rego, A. Dilendra, and J. Lloret, “Study of multimedia delivery over software defined networks,” in *Network Protocols and Algorithms*, vol. 7, no. 4. Macrothink Institute, 2015, pp. 37–62.

[33] K. W. Bauer, G. S. Parnell, and D. A. Meyers, “Response surface methodology as a sensitivity analysis tool in decision analysis,” *Journal of Multi-Criteria Decision Analysis*, vol. 8, no. 3, pp. 162–180, 1999. DOI: http://dx.doi.org/10.1002/(SICI)1099-1360(199905)8:3⟨162::AID-MCDA241⟩3.0.CO;2-X

[34] R. L. Iman and J. C. Helton, “An investigation of uncertainty and sensitivity analysis techniques for computer models,” *Risk Analysis*, vol. 8, no. 1, pp. 71–90, 1988. DOI: http://dx.doi.org/10.1111/j.1539-6924.1988.tb01155.x