Comparative analysis of the possibilities of building a decentralized control plane of a Software Defined Networks

Jordan Raychev, Diyana Kinaneva, Georgi Hristov, Plamen Zahariev

The objective of the current paper is to be conducted a comparative analysis of the available possibilities for building a decentralized controller plane of a Software Defined Network (SDN). The SDN networks are an emerging paradigm, which aims to change telecommunications networks as we known them today. The constantly growing social media, mobile communications and technologies like server virtualization that can be found in the modern data centers are pushing the limits of traditional computer networks. That led to a lot of researching in the field of network virtualization and more specifically the field of software defined networking. One of the most important challenges in front of the new concept for software defined networking is the single point of failure that is created due to the centralization of the control. Unfortunately, as every new technology and this one have its flaws. One of the biggest research fields is investigating the possibilities of building a decentralized control plane to avoid issues related to the single point of failure, scalability, high availability and security.

Сравнителен анализ на възможностите за изграждане на децентрализирана контролна равнина при софтуерно дефинираните мрежи (Йордан Райчев, Дияна Кинанева, Георги Христов, Пламен Захариев). *Целта на настоящият доклад е да бъде извършен сравнителен анализ на познатите до момента решения за създаване на децентрализирана контролна равнина при софтуерно дефинираните (СД) мрежи. Софтуерно дефинираните мрежи представляват съвременна парадигма, чиято основна цел е да промени телекомуникационните мрежи, които познаваме днес. Непрекъснато разрастващите се социални мрежи, мобилните комуникации и технологии като сървърната виртуализация, намерила широко приложение в съвременните центрове за съхранение на данни и публичните облачни услуги достигат лимита на възможностите на традиционните телекомуникационни мрежи. Това налага изследването и имплементирането на технологии като мрежовата виртуализация и по-конкретно софтуерно дефинираните мрежи. Един от основните недостатъци на първоначалната концепция на софтуерно дефинираните мрежи е създаването на единична точка за отказ на мрежата поради наличието на един единствен управляващ контролер. Съществуват редица изследвания, които са съсредоточени върху този проблем и неговото отстраняване. Всички тези изследвания се свеждат до създаване на децентрализирана контролна равнина, която има за цел, както да отстрани изцяло проблем с отпадането на управляващия контролер и загуба на пълна свързаност в софтуерно дефинираната мрежа, така и увеличаване мащабируемостта и сигурността на дадената мрежа.*

Keywords – software defined networks, decentralization, single point of failure

Introduction

The requirements of modern application and their users are pushing the conventional communication networks to their limits. Dynamic development of the modern information and communication technologies during the last decade is the major factor, which forces current network architecture to be revised. Providing high quality of services, dynamic, centralized and programmable management of telecommunication networks are part of the issues that arise when it comes down to building and standardizing new network architectures like software defined networks [1].

Software defined networks is a new technology which provide programmable management of computer networks. They aim to minimize the problems that accompany traditional telecommunication networks. Such problems are management difficulty, the lack of flexibility and scalability, heterogeneity of network devices and others. The idea of creating programmable computer networks was born more than a decade ago [2] with a proposal to increase the level of programmability of the communication networks that can be achieved by separating functionality of control plane from the data plane. This in turn would allow rapid and effective deployment of new network services, coexistence of network devices from different manufacturers without creating conflicts, virtualization of network equipment and others. The major and most important stage from the development process of the new architecture is the separation of control plane from the data plane which sits at the physical layer. Physical and logical separation of the two planes provides the possibility for dynamic deployment of new functions to networks, something that would be practically impossible with conventional networks. In software defined networks this is achieved by physical separation of the control plane and centralizes its functions to a separate network device called network controller. Using a centralized control plane has its benefits, for example, the controller has a single point of view that facilitate network management which is expressed in traffic optimization, easy and rapid functional deployment, reducing management cost and so on.

Software-defined network architecture

As it can be noticed in fig. 1, which shows network architecture with centralized controller, all control functions from the physical layer are moved to the control plane as a separate device known as controller. Such separation of the control logic leaves the devices at the physical layer with no control functionalities, just a simple flow table, which entries are populated via the SDN controller.

Flow tables that are illustrated in fig. 1 represent pseudo flow tables which reflect packet switching from high level of abstraction. In this line of thought the real SDN switches and in particular their flow tables are composed of multiple elements (not only these shown in fig.1) such as match field, priority, packet counter, instruction and so on, that are not discussed in this paper.

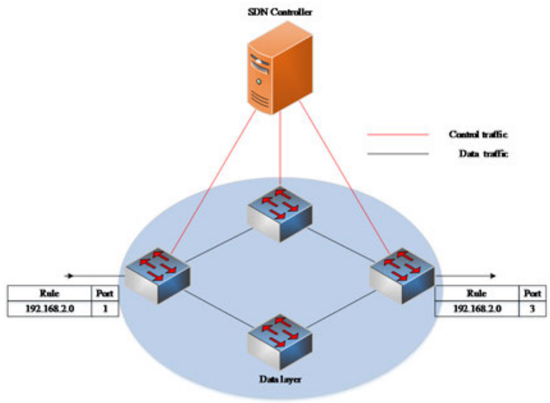

Fig. 1. Simple Architecture of SDN network.

For the purpose of our explanation it is only important to be noticed that when a packet arrive at the ingress port, the SDN switch looks-up the flow table in order to decide how to process the packet. There are two outcomes in such situation. First, if the switch has an entry in its flow table for the corresponding packet (there is a match) will be switched to egress port regarding the rule found in the entry. In the second case the switch does not have any entry for the destination of the incoming packet. In that case the switch will query the controller and as a response the controller will install an entry with a corresponding rule to the flow table of the switch and then the switch will have the capability to forward all subsequent packets. As it can be seen, the SDN switch is just a forwarding device that can switch or route packets regarding the rule installed from the controller. Routing and switching are interchangeable terms in the context of software defined networks and the only difference from conventional networks is in the way how packets and frames are processed. In conventional networks each device has its own control plane and makes decisions by itself, while in SDN networks the controller hold the control and decides how the network will operate generally.

The architecture of software defined networks presented in [1] and its newer version [3] is just a reference and it is used as base for most modern implementation of SDN controllers. However this architecture does not give a description how the control place could be implemented – distributed or centralized. The answer to this question might seem easy to determine based on the analysis provided above, but it is actually a quite difficult task. Using single management controller raises questions related to scalability of the network and control plane performance issues. Researches show for example that the NOX controller managed to process 30,000 flows per second [4]. With the constantly increasing requirements of the modern data centers and even the needs of most Internet service providers (ISP) this flow processing speed is insufficient. Scalability of the control plane is a major issue for successful implementation of SDN in modern communication networks. The control plane of SDN networks is scalable only if its architecture is able to dynamically adapt to the constant changes occurring in the network as well as satisfying the quality of services. In order to achieve this level of scalability two approaches are proposed. The first approach is related to transferring some of the function of control plane to the switches in forwarding plane. This will transfer some of the tasks from the SDN controller to the switches, which in turn will reduce the queries sent to the control plane and at the same time will increase the overall network performance. However, the technological implementation of that approach is quite difficult. The feasibility of such solution is accompanied by creating integrated motherboards that would enhance the functionalities of the forwarding elements. However, moving part of the control function back to forwarding devices breaks the ideology behind the software defined networks, turning them back to conventional communication networks.

The second approach is to create a distributed control plane architecture where tasks are distributed among multiple SDN controllers. This approach allows queries to be process in parallel by different SDN controllers while at the same time the controllers communicate between each other in order to synchronize the information and avoid conflicts. The advantage of this approach is that technological simplicity of network elements from forwarding plane is kept. Another advantage of this approach is that the idea behind SDN networks and the centralized management are also kept and in addition the single-point of failure is not an issue any more since there are more than one controllers that can continue working if one controller fails.

Models of managing the control plane of software defined networks

The management of the control plane can be classified into three main categories – centralized, decentralized and hierarchical. Decentralized and hierarchical management can be classified into a single category – distributed management structures.

Model of centralized control management

The centralized control management is a model that conceptually describes the relationship between individual nodes of a particular structure. In the context of computer networks centralized control management means the management of a particular group of resources through a single master control unit. The first architecture of software defined networks is represented by a centralized control management model, as shown in fig.2. In this management model all queries are sent to the controller, which is responsible for their processing and giving a response back to network elements of forwarding plane. The controller is able to manage the network since it is the only element of the network that has a global view of the topology. Having a global view of the network gives the controller the ability to perform correct routing decisions.

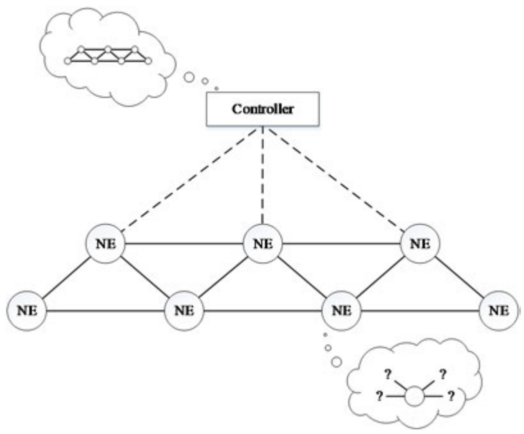

Fig. 2. Centralized control management of SDN networks.

Centralized control management has the following features and advantages:

- The ability for easy management due to the fact that in most cases the controller is implemented into a single physical server;
- Improved network security due to the minimum of logical and physical connection from and to the controller;
- Low cost of the elements forming the data plane;
- Ability to make better use of available physical resources.

There are also some disadvantages of centralized control management:

- Using a one centralized controller opens a vulnerability place in the network even in software and hardware. In case of software vulnerably, for example found in the operating system of the controller would compromise the entire network. Hardware vulnerability could be seen in case of controller failure where the network becomes unmanageable;

- Lack of scalability. Increasing the number of forwarding devices in the data plane will increase the number of queries sent to the controller. In this case the controller would not have the ability to process the query on time due to network congestion. The entire network performance decrease as a result.

Model of decentralized control management

Decentralized computer system is a group of autonomous network elements, which communicate with each other in order to implement particular task. A distinctive feature of decentralized control management is that control elements can be located in a physical room but most often they are located in geographically different locations. An example of decentralized system is telephone system. The telephone system can be perceived as a heterogeneous system due to the fact that it is made up of many different devices (computers, terminals, etc.) [5].

In the context of software defined networks decentralized system has a different meaning. Due to the fact that in SDN networks the control plane is physically separated for data plane it is necessary to find an approach that will distribute the management among network elements from the control plane. In case of a large amount of queries the overall load can be distributed among the controllers, which would remove the problem with overloading found in the centralized approach for management. In case of controller failure there will always be another controller which will take over the control. In that case the operation of the network will not be interrupted and it will continue working. All of the aforementioned are only part of the advantages of decentralized control management. There are other advantages such as:

- Scalability;
- Security;
- Ability to fast recovery;
- High availability.

There are several disadvantages of this category management:

- Higher cost of network maintenance;
- Lower efficiency compared to centralized management due to the fact it is almost impossible to use all of the available resource;
- The network becomes more complex.

Decentralized control management in software defined network is conditionally divided in two subgroups – management with global and local view of the network topology. In case of management with global view of the network topology the control plane is implemented by N SDN controllers, each of which has global view of the network topology. One of the main advantages of this type of network architecture is the small delay of responses to the queries coming from the physical plane and going to the controllers. The architecture of decentralized control plane is given in fig. 3.

Decentralized management with local view of network topology has the following characteristics: the control plane is again implemented by multiple controllers in order to provide scalability but this time the controllers do not have a global view of the network topology. Instead of having global view of the network topology the controllers only manage the devices that are in their administrative domain.

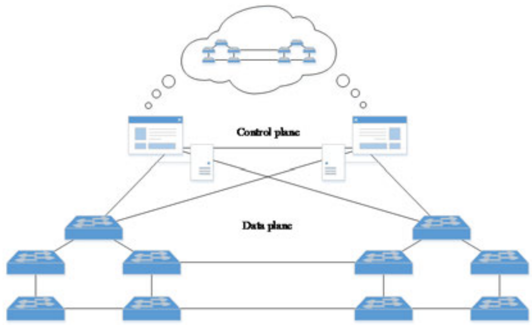

Fig. 3. Architecture of decentralized control management with global view of network topology.

In opposed to decentralized control management with global view of the network topology, where the delay for replay the queries is significantly low, the delay in decentralized management with local view of the topology would be higher due to the fact that in some cases the controller in one domain have to communicate with controllers in other domains. In this case the time for synchronizing the information between the controllers in different domain would add additional delay.

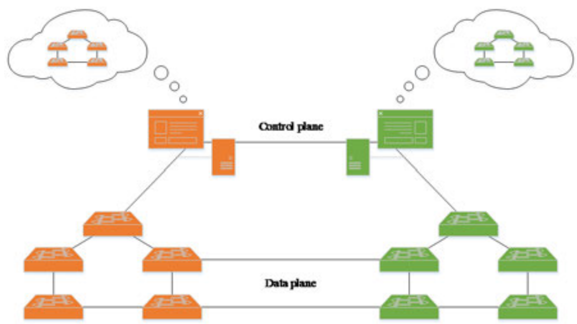

Fig. 4. Architecture of decentralized control plane with local view of the network topology.

The architecture of control plane implementation with decentralization and local view of the network topology is given in fig. 4. In order to communicate the controllers from different domain use the so called east-west interfaces. East-west communication is used for synchronizing control information between controllers in different domains.

Model of hierarchical control management

The model of hierarchical implementation of control plane can be referred to the model with decentralized control plane implementation with local topology view. The difference between the two is that in hierarchical implementation there is one additional master controller that will control all of the other controllers. The architecture of the hierarchical model is given in fig. 5. As it can be noticed in the figure, the network topology again is separated to several administrative domains and each domain is managed by the corresponding for the domain controller. In addition to the domain controllers the architecture adds additional controller which main purpose is to monitor and manage the domain controllers.

Fig.5. Architecture of decentralized control plane with hierarchical view of network topology.

A major disadvantage of this type of control plane implementation is the existence of additional controller which adds complexity to the architecture of SDN networks and creates a prerequisite for overlapping the controllers' functionalities.

The hierarchical structure of the control plane separates the controllers in two main categories – global and local. The local SDN controller manages only one administrative domain. They generate rules and install them in local flow tables. The global SDN controller on the other hand manages multiple administrative domains and participates in the routing decision if and only if there is a need for global view of network topology.

Types of architectures for building a decentralized control plane in SDN

Different models for control plane management in Software defined networks have been examined. In this section a comparative analysis will be carried out between existing solutions for building a decentralized control plane.

Kandoo

The architecture of Kandoo control plane [6] is based on the model with hierarchical management. This type of network architecture differs greatly from the abovementioned solutions for control plane management. The control plane in Kandoo architecture is realized based on two levels of hierarchy, which is achieved by using additional controller also called local controller. This network architecture type is shown in fig.6. The local controller is responsible for queries that occur more frequently while the main controller response to all other queries.

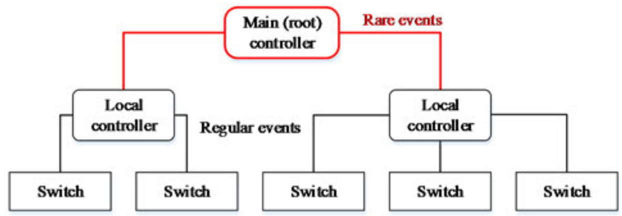

Fig.6. Two-layered hierarchy of control plane architecture.

With the structure shown in fig.6 the local controllers manage one or more switches in their domains, while the main (root) controller is responsible for managing all local controllers or in particular the main controller manage more rarely happened events. A major characteristic of main controller is that it is the only device in the network than has a global view of network topology. Another feature of the architecture is the ability to implement local controller functions in device from the access layer (the switches) which will significantly simplify the structure of the network. The approach of Kandoo architecture is a diversion from the ideology of software defined networking where the main approach relays on physical separation of control logic from data plane.

ONIX

The decentralized architecture of ONIX controller [7] is composed of four main elements – physical and communicational infrastructure; application programming interface of ONIX, and control logic.

The **physical infrastructure** consists of network devices (switches and routers) that work at datalink and network layers of the OSI (Open System Interconnection) other devices that are used as traffic balancers. There are requirements for ONIX devices – they have to support ONIX application layer protocol with read and write permissions.

The **Communication infrastructure** is an element from the architecture which is responsible for network connectivity between physical infrastructure and the application of ONIX. The control channel that is built between the physical infrastructure and the applications can be implemented in two different ways – in-band and out-of-band. In in-band implementation the control channel is shared between the control traffic and the data traffic, while out-of-band implementation separates the control traffic from data traffic via a separate communication channel.

The distributed control system is implemented via a cluster of multiple physically connected servers. On each server several application programming interfaces (APIs) can be installed. The APIs give read and write permissions to network resources.

The control logic is the last element of the system and it is implemented above the application programming interface of ONIX. The major function of control logic is to determine the state of each object from the physical infrastructure. A copy of each determined state for each physical object is stored in a database which is called NIB or Network information base. The NIB database is similar to the routing tables which are essential for network devices that work in layer 3 of the OSI model.

HyperFlow

The architecture of HyperFlow [8] consists of OpenFlow switches (which built-up the data plane) and NOX SDN controllers (which built-up the control plane). HyperFlow is running on each controller and there are also an application and a system which are responsible for the communication between the controllers. A characteristic of this architecture is the fact that each controller has a global view of the network topology and each controller work as an independent unit such as it is the only controller in the network. Fig. 7 illustrates an example network which uses the HyperFlow network architecture. As it can be noticed in fig. 7 each controller manages a group of switches usually the nearest ones. In case of controller failure the switches under its management are reconnected to other controller again the nearest one.

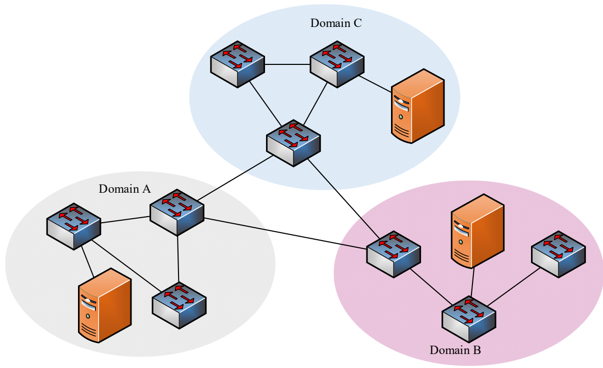

Fig.7. HyperFlow architecture.

DIFANE

The architecture called DIFANE (Distributed Flow Architecture for Network Enterprises) [9] is a different approach for control plane management and implementation. DIFANE architecture increases network performance and scalability by moving part of the responsibilities of the controller back to the access layer – in particular to a specialized switch called Authority switch. The role of Authority switch is to manage a group of switches from data layer. In a standard SDN architecture when a packet arrives in the switch ingress port, the switch looks for a match in its flow table. If there is a match the switch forwards the packet to its destination, otherwise the switch sends a query to the SDN controller and the controller response with a rule how the switch should process the packet. In DIFANE architecture when a switch receives a first packet from a flow it automatically sends the packet to Authority switch. The Authority switch is responsible for packet delivery to its destination, as shown in fig. 8. This is possible due to the controller pre-installs rules in the flow table of Authority switch. The second packet from the flow is forwarded by the switch from the data plane since it already has a cashed entry in its flow table.

The architecture illustrated in fig. 8 has two main ideas:

- The main (root) SDN controller distributes and pre-installs rules in the Authority switches with the goal to improve the scalability of the network. The rules are equally distributed among all authority switches;
- Switches in the data plane process the traffic based on cashed rules. In case of no cashed rules the switch redirect packets to authority switches.

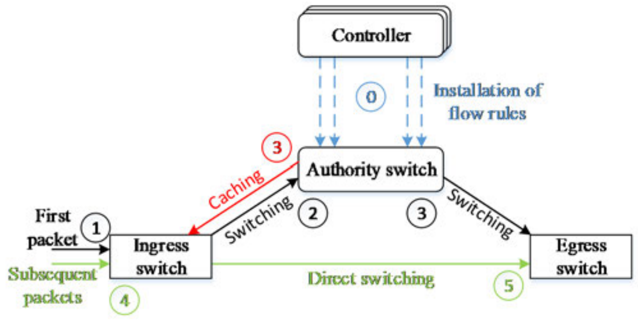

Fig. 8. DIFANE architecture.

DevoFlow

The idea of DevoFlow architecture presented in [10] is similar to DIFANE architecture – moving part of the responsibilities of the controllers back to the switches or in particular applying additional control logic to the switched which the goal to keep most of the traffic flows in the data plane. The authors represent several new mechanisms for distinguishing the traffic between control plane and data plane. The idea behind DevoFlow breaks the concept of SDN networks for separating control logic from data plane.

ONOS

ONOS (Open Network Operating System) [11] is distributed system for building decentralized control plane with high performance, availability, scalability, and security. ONOS is extremely flexible system, which allows implementing centralized or decentralized control plane with global view of network topology. Researches [12] show that the architecture of the control plane with global view of the network topology does not provide the highest level of scalability, however the authors choose to implement such architecture and on top of that divide the topology into logical administrative domains. The ONOS approach transforms the control plane architecture to architecture with local view of the network topology, which has a better scalability.

Conclusion

Software defined networks rely on the idea for logical separation of the control plane from the data plane while at the same time the control plane is physically move to a centralized server also known as SDN controller. Despite the numerous advantages and applications of such approach it is important to remember that centralized architectures do not scale well and have scalability and availability issues.

A lot of researches have been carried out during the past several years exploring the possibilities of building distributed management for the control plane that would increase the availability, security and scalability of SDN networks.

The main characteristics of the different architectures – centralized, decentralized with local or global view of the network topology and hierarchical, have been revealed in this paper. In addition different control plane solutions that are available were also reviewed. From the analysis it can be concluded that building decentralized control plane (no matter of its type) is obvious and a must in order to meet the requirements for scalability and availability of networks of large sizes (data centers, ISPs and others) which also have demand for high quality of services, availability and low latency.

Acknowledgement

The work presented in this paper is completed as partial fulfilment of a PROJECT 2018 - FEEA - 02 “Development and evaluation of a methodology for sensing, analysis and visualization of open data through LoRaWAN based communication infrastructure”, financed under the Scientific Fund of the University of Ruse.

REFERENCES

[1] Open Networking Foundation, SDN architecture, Issue 1, June 2014.

[2] Campbell, A., De Meer, H., Kounavis, M., Miki, K., Vicente, J., and D. Villela, "A Survey of Programmable Networks", ACM SIGCOMM Computer Communication Review, Volume 29, Issue 2, pp. 7-23, September 1992.

[3] Open Networking Foundation, SDN Architecture, Issue 1.1, 2016.

[4] A. Tavakkoli et al., “Applying NOX to the Datacenter,” Proc. ACM HotNets-VIII Wksp., 2009.

[5] Gray J., An Approach to Decentralized Computer Systems, Technical Report 85.4, June 1985.

[6] S. Yeganeh , Y. Ganjali, Kandoo: a framework for efficient and scalable offloading of control applications, Proceedings of the first workshop on Hot topics in software defined networks, August 13-13, 2012, Helsinki, Finland.

[7] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu , R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, S. Shenker, Onix: a distributed control platform for large-scale production networks, Proceedings of the 9th USENIX conference on Operating systems design and implementation, p.1-6, October 04-06, 2010, Vancouver, BC, Canada.

[8] A. Tootoonchian and Y. Ganjali, “HyperFlow: a distributed control plane for OpenFlow,” in Proceedings of the 2010 internet network management conference on Research on enterprise networking, ser. INM/WREN’10. Berkeley, CA, USA: USENIX Association, 2010, pp. 3–3.

[9] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, “Scalable flow-based networking with difane,” SIGCOMM Comput. Commun. Rev., vol. 41, no. 4, pp. –, Aug. 2010.

[10] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, S. Banerjee, “DevoFlow: scaling flow management for high- performance networks,” Comput. Commun. Rev., vol. 41, no. 4, pp. 254–265, Aug. 2011.

[11] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O’Connor, P. Radoslavov, W. Snow, and G. Parulkar, “ONOS: Towards an open, distributed SDN OS,” in Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, ser. HotSDN ’14. New York, NY, USA: ACM, 2014, pp. 1–6.

[12] Hu, Jie et al. “Scalability of control planes for Software defined networks: Modeling and evaluation.” 2014 IEEE 22nd International Symposium of Quality of Service (IWQoS) (2014): 147-152.

Received on: 05.03.2019