International Journal of Engineering Innovation & Research
Volume 7, Issue 1, ISSN: 2277 – 5668

Modeling of SDN-based Network Behavior Based on Network Calculus Theory

Samer Salah Thabit* and Dr. Tariq Abdoalkarim Abdoalfathil
Corresponding author email id: olena.tkachova@nure.ua

Date of publication (dd/mm/yyyy): 08/02/2018

***Abstract* – A new approach to modeling behavior of SDN-based networks is proposed in the paper. The main feature of the proposed approach is the ability to modeling boundary meanings of quality of service characteristics of Software-Defined Networking, taking into account the dynamic change network infrastructure and particular services degradation and subsequent analysis of the results based on mathematical apparatus of the theory of network calculus. In accordance with the basic principles of network calculus theory formed mathematical model of Software-Defined Networking infrastructure functioning, the analysis of temporal parameters of the process of service delivery. Examples of modeling the behavior of commutation element of Software-Defined Networking, taking into account the intensity of incoming requests, as well as various methods of data transfer, with aggregation and without aggregation of data flows, are also proposed in the paper.**

***Keywords* – Network Behavior, Network Calculus Theory, Latency, Performance, Software-Defined Networking, Network Node.**

I. INTRODUCTION

Analysis of mechanisms of services provision in Software-Defined Networking (SDN) showed that the issues of guaranteed quality of service (QoS) of user data flows given insufficient consideration: in the existing specifications and recommendations described functional criteria interaction methods SDN elements and operations between them [1-3]. At that time, as the non-functional requirements for the network, methods for supporting quality of service and provide the required service performance levels are not precisely defined. Due to this situation, the task of analysis and evaluation of the correct formation of non-functional requirements, including dynamically changing the QoS parameters, in a SDN-based networks is important.

As a method of estimating the dynamic state change of SDN-based networks infrastructure that affect to the resulting quality indicators, the network theory proposed to use estimates (network calculus) [4-7]. The basis of this theory is the idea of forming deterministic estimates of quality indicators by analyzing the boundary scenarios for the functioning of individual network fragments and combining these scenarios with each other.

The network calculus theory allows formalizing the time dependencies of the processing of an aggregated data stream in the case of the composition of several elements, which is difficult to achieve by using other methods. The one of the main advantages of network calculus is that the theory allows to operate both exact values of network characteristics, and to determine their threshold values [4-6].

The network behavior is described as a system, the input is the data that the sender injects into the network, and the output is the data that arrives at the receiver. Such approach can be used to calculate the worst quality of service provision, or, conversely, to determine the minimum requirements for network infrastructure elements, which is an important practical task when developing and improving control systems for SDN-based networks.

Network calculus theory as a methodology for analyzing performance has found wide application in the design and implementation of network solutions supporting a guaranteed level of quality of service, in particular, with the support of the mechanisms IntServ or DiffServ [7, 8], wireless sensor networks [9], networks operating on the basis of Ethernet technology [10, 11], ATM technology, etc.

In [12], the application of network calculus theory for estimating the latency indices in SDN was proposed, but only a particular case of the presence of only one Open Flow switch is covered. In work [13] the elements of the theory of network calculus are used in modeling the process of functioning of the SDN controller and the subsequent analysis of its performance, however, the study is limited to a controller-switch type interaction.

II. AN APPLICATION OF NETWORK CALCULUS THEORY FOR MODELING SDN-BASED NETWORKS

The Software-Defined Networking is represented by a two-level architecture: the access level, which is characterized by the parameters of the switch and the level of control that is characterized by the parameters of the controller. Its main feature is the formation of aggregated data flows. Aggregation is carried out in accordance with the value of the ToS field of incoming traffic [17].

Aggregation of data flows is performed both on OpenFlow switches and on intermediate devices of the network - FlowVisor. Both the FlowVisor and the switch perform a list of possible actions only based on the control messages received from the controller. The block diagram of the SDN fragment (controller-commutator interaction) is shown in Figure 1.

The model of SDN-based network in accordance with the fundamentals of network calculus theory includes the next basic types of elements:

1. Handler nodes. Such type of nodes include OpenFlow switches and SDN controllers [14]. The specification of the processing node generates such characteristics as the maximum total processing speed of the data flow R, the average processing speed of the individual flow r_i ,

the volume of the input buffer *q*, and the service discipline *Q*.

The output of the handler node *P(t)* is estimated using the service curve *b(t)* and the send function *D(t)* [15].

A main feature of the model of SDN-based network is that the handler node can be represented by one physical network device (switch) or their combination (switch-controller-switch is an incremental function that determines the deviation of the current volume of processed data from the amount of data processed by the processing node for the previous time point the provision of services.

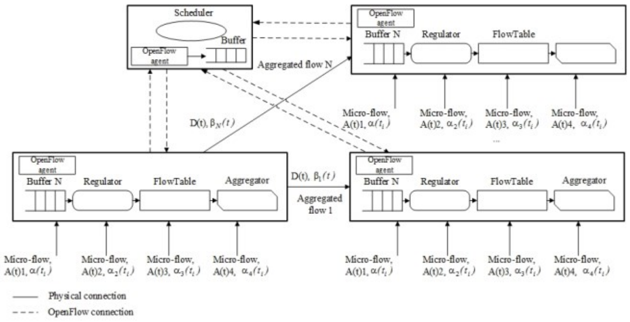


Fig. 1. Structural scheme of SDN-based network (aggregation of data flows)

2. Data flows. The data flows can be either a micro flow [4] formed during the interaction of the consumer node and the handler node or the aggregated flow [6], usually formed during the interaction between the handler nodes. For each type of flow, a specific load curve $\alpha(t)$ [4, 7]. When forming the load curve, the following characteristics are taken into account: the peak data arrival rate *p*, the change in the flux intensity *b*, the maximum data packet size *M*, and the average data flow rate averaged for each flow.

The set *(p, r, M, b)* forms the specification of the T-SPEC data stream [15]. Thus, the load curve is always defined and can be represented by an expression of the form $\alpha(t) = min(M + pt, rt + b)$.

When modeling handlers and data streams in the theory of network calculus, memory functions and curves are used that express the total amount of information transferred or processed by the simulated elements from the moment of origin [4, 8]. Below is a detailed presentation of the elements of the theory of network calculus that are used in the SDN-based network elements modeling process [16].

The arrival curve is an incremental function that characterizes the current value of the amount of incoming data at a specific time point to the handler node [4]. The load curve can be constructed "piecewise" in accordance with the allocated time intervals:

$$\alpha(\tau) = R(t_i) - R(t), \qquad (1)$$

where $R(t_i)$ is the amount of data arriving at the handler node at the time point t_i, $R(t)$ is the amount of data arriving at the handler node at the time point t.

Service curve $\beta(\tau)$ is an incremental function that determines the deviation of the current volume of processed data from the amount of data processed by the processing node for the previous time point [4]:

$$\beta(\tau) = R' \cdot \alpha(t_i) - R' \cdot \alpha(t), \qquad (2)$$

where $R'\alpha(t_i)$ is the amount of data processed by the handler node at the time t_i, $R' \cdot \alpha(t)$ is the amount of data processed by the handler node at the time t.

The arrival function of data flow is an accumulation function *A(t)*, which describes the dependence of the total amount of data received by *A(t)* on the processing node at certain points in time [15].

The function of sending a data flow is an accumulation function *D(t)*, which describes the dependence of the total amount of data processed by the handler at certain points in time.

The performance of each handler node can be modeled by the number of pairs of the form $< \alpha(\tau), \beta(\tau) >$ and $< A(t), D(t) >$. In this case, the value obtained from the pair $< \alpha(\tau), \beta(\tau) >$ is a boundary value. The value $< \alpha(\tau), \beta(\tau) >$ takes into account the worst or the best value of network performance, the value $< A(t), D(t) >$ is determines actual latency and average latency at a certain point in time.

Backlog *b(t)* is the dependence of the amount of data stored on the processing node N that belong to one thread from time to time. Backlog is modeled as a difference between the arrival and departure functions *A(t)* and the sending *D(t)* of the data stream served by it [15]:

$$b(t) = A(t) - D(t) \qquad (3)$$

The buffer size, both the controller and the switch, is characterized by the lag value. When analyzing the arrival and service curves, the curve characterizing the change in the lag value can be represented by the following formalism:

$$\forall t : b(t_i) \le \alpha(t_i) \otimes \beta(t_i) =: v(\alpha(t_i), \beta(t_i)). \qquad (4)$$

In the case of the analysis of the arrival curve and the service curve, the latency curve can be represented as follows [4]:

$$\forall t : d \le \sup_{\Delta > 0}\{\inf\{t \ge 0 : \alpha(t_i) \ge \beta(t_i)\}\} =: h(\alpha(t_i), \beta(t_i)). \qquad (5)$$

The performance curve of the handler node for each micro-flow *i*, depending on its priority, can be specified as follows:

$$r_i = \frac{\phi_i R_N}{\sum_{j=0}^{n} \phi_j} P_N(t), \qquad (6)$$

where ϕ_i , ϕ_j is the weight (priority) of each thread when serviced by the handler node, $P_N(t)$ is the current value of productivity (CPU) processor node, R_N is total processing speed.

Service curve of SDN-based network is modelling by the following equation:

$$\beta(\tau)=\sum\nolimits_{k=1}^{n}\beta_k(\tau),\forall t>0\,, \qquad (7)$$

where $\beta_k(\tau)$ is the service curve generated by each individual incoming data stream, n is the number of incoming streams. In accordance with (7) and (8), the maintenance curve for the micro stream can be generated by the following equation:

$$\beta_i(\tau)=\beta(\tau)-\sum\nolimits_{k=1,k\neq i}^{n}\alpha_k(\tau-\tau_k), \quad \forall\tau>0,\tau>\tau_k>0 \qquad (8)$$

where $\sum\nolimits_{k=1,k\neq i}^{n}\alpha_k(\tau-\tau_k)$ is the total load curve for incoming micro-flows, with the exception of the flow *i*, τ_k is The lag time introduced when the node processes the total flow $\alpha_k(\tau)$.

The load curve leaving the processing node, which will then be applied as an incoming load curve for further processing nodes, can be represented as [4]:

$$\alpha^*(\tau)=\begin{cases}\dfrac{b-\frac{M}{t}}{p-r}\le T_h,\ \alpha^*(\tau)=b+r(|D(t)|+t);\\[2ex] \dfrac{b-\frac{M}{t}}{p-r}\ge T_h,\ \alpha^*(\tau)=min\{R(|D(t)|+t)+\dfrac{b-\frac{M}{t}}{p-r}(p-R)^+,b+r(|D(t)|+t).\end{cases} \qquad (9)$$

Assumptions used in modeling: a regulator of the Leaky Bucket type is considered as a regulator [4, 17, 18]. OpenFlow switches use the First In First Out service discipline [4, 18].

III. MODELLING THE TIME CHARACTERISTICS OF A SDN-BASED NETWORKS THAT AFFECT TO THE QUALITY OF SERVICES

In accordance with the features of SDN OpenFlow, the switch can transmit data without accessing the controller. This situation is possible if there are coincidences between the fields of the packet that came to the switch and the fields of the redirection table (Flow Table). In this case, the value of the service curve is calculated in accordance with the formula (8), the value of the incoming load curve - (1) and the outgoing load curve - (9).

$$d=\alpha(t_i)-\beta(t)=(rt+b)-\frac{\sum\nolimits_{k=1}^{l}b_k}{R-\sum\nolimits_{k>l}^{n}r}\,. \qquad (10)$$

The upper value of the buffer length for certain characteristics of the switch can be calculated as follows:

$$b_i=\sum\nolimits_{k=1}^{n}(b_k+r_k t)-(R-\sum\nolimits_{k=1,k\neq i}^{n}r_k)t\,. \qquad (11)$$

If the OpenFlow switch redirects the incoming request from the consumer node to the controller, the value of the service curve changes. In this case, the following formula holds:

$$\beta(\tau_+)=[\beta_c(\tau_+)-\sum\nolimits_{k=1}^{l}\alpha^*(\tau)]^+\,, \qquad (12)$$

where τ_+ is time interval in view of receiving a response from the controller, as a rule $\tau_+\equiv\tau$, it is mean that processing time of the OpenFlow request by the controller is negligible, $\beta_c(\tau_+)$ is a service curve of controller, $\sum\nolimits_{k=1}^{l}\alpha^*(\tau)$ is load curve output from the handler node (OpenFlow switch).

Let the thread pass through several node-handlers. The total fragment of the network consists of N handler nodes. Each handler is defined by an individual set of characteristics that affect its performance. Suppose that as a regulator of the data flow on each node-handler, the Leaky Bucket controller is applied and the load curve emerging from the handler node is characterized by the equation (12), the service discipline - First in First Out [19].

In the process of modeling, an aggregated flow corresponding to a certain class of service is considered. In accordance with [4, 7], the total service curve is given by the next expression:

$$\beta(\tau)=\beta_1(\tau)\otimes\beta_1(\tau)\otimes\ldots\beta_N(\tau)\,. \qquad (13)$$

Thus, given a mathematical model based on a calculation theory network, allows to calculate and analyze the latency boundary values, the volume of the input buffer for each service device.

IV. EXPERIMENTAL RESULT

As an example, the network fragment with NFV technology support. Providing services with guaranteed quality of service in this fragment is achieved by means of load control using resource reservation mechanism (RSVP).

Suppose the flow T-SPEC determined and transmitted from the sender to the recipient through N nodes, where nodes 1 and N - finite elements of the system, the nodes 2, ..., N-1 - intermediate elements.

In general, the resource reservation process can be represented as follows: the connection initiator generates and sends a PATH message. In the body of the PATH message, data specifying the specification of the service flow (service class and priority φ_i. Information about the flow characteristics is viewed and analyzed by intermediate nodes, on the basis of the analysis, the device either generates a message about the impossibility of transmission or reserves the required amount of resources is shown on Figure 2.

To build a simulation model of a multi-service network segment, the OMNeT ++ 5.0 modeling system was chosen [20].

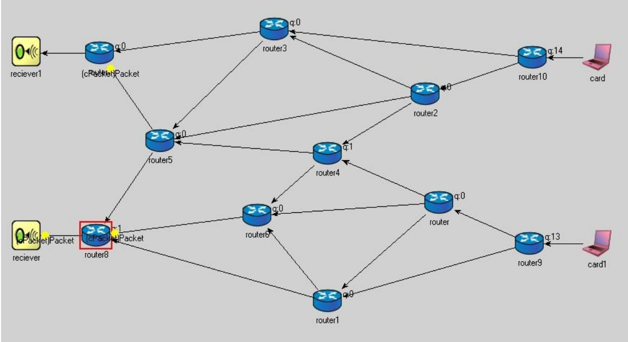


Fig. 2. Fragment of the experimental network infrastructure In the process of developing a segment of a multiservice network, the environment components for modeling have been created:

- Routing Block:

```
 simple Router
{ parameters:
        int      queueCapacity      =
default(25);
        volatile   int    queuePar    =
default(0);

@display("q=Rqueue;i=abstract/router");
     gates:
        input in[];
        output out[];}
```

- The processing queue of incoming packets with some probability introduces an error in the discharge of the packet:

```
 void    Router::handleMessage(cMessage
*msg)
 {    queuePar-
>setLongValue(queue.getLength());
      if(msg == endTransmittionEvent)
      {

      else if(msg->arrivedOn("in"))
      {EV << "Packet arrived ";
          if(transmittionActive     ==
true)
          {EV << "already
transmitting -- to queue ";
      if(queue.length()<queueCapacity)
                  {
      queue.insert(msg);
                  }
          else
          {
          delete msg;
          }}
          else
          {EV << "queue empty --
starting transmission. ";
          startTransmitting(msg);
          }}}
```

- Traffic Generation Module:

```
 void       Card::handleMessage(cMessage
*msg)
 {
       cPacket     *p       =      new
cPacket("Packet");
       p->setByteLength(202);
       p->setKind(4);
       p->setTimestamp();
       EV
 << "timeToSend"<<timeToSend<<"";
       if(timeToSend > 0)
             {
             scheduleAt(simTime()+
(timeToSend*=0.99), msg);      }
       else
       {scheduleAt(simTime()+ 0.5, msg);
       }     }
```

- A receiver module. The message receiver receives packets and records the reception fact, the delivery time, the number of intermediate nodes, the presence of errors:

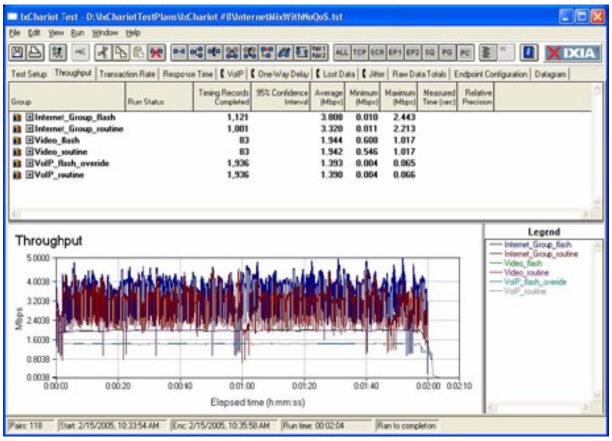
```
simple Reciever
{    parameters:

@display("i=block/wrx,yellow");
          volatile    int    queuePar    =
default(150);
      gates:
          input in[];}
 void   Reciever::handleMessage(cMessage
*msg)
 {    simtime_t eed = simTime() - msg-
>getCreationTime();
      endToEndDelay.record(eed);
      EV << "Recieved";
      bubble("Recieved");
      delete msg;}
```

- Backbone network module. Variable properties of the object: throughput, time of signal propagation, probability of error in the discharge. The backbone network transfers packets between sources, routers and receivers, limits bandwidth, and also introduces error and delay:

```
      channel    EtherChannel    extends
DatarateChannel
      {datarate = 1 Mbps;}
      Package generation module:
      cPacket     *p      =      new
cPacket("Packet");
      p->setByteLength(202);
      p->setKind(4);
      p->setTimestamp();
```

During the testing, a combination of all of the traffic types listed above was used. Then the received traffic was duplicated, reserving one for the implementation of the QoS policy, and the other for continuing the normal QoS policies. First, all pairs were started and the results were measured. After that, a QoS policy on duplicated pairs was introduced and appropriate QoS policies were configured on the routers. The duration of the test was chosen to be 2 minutes.

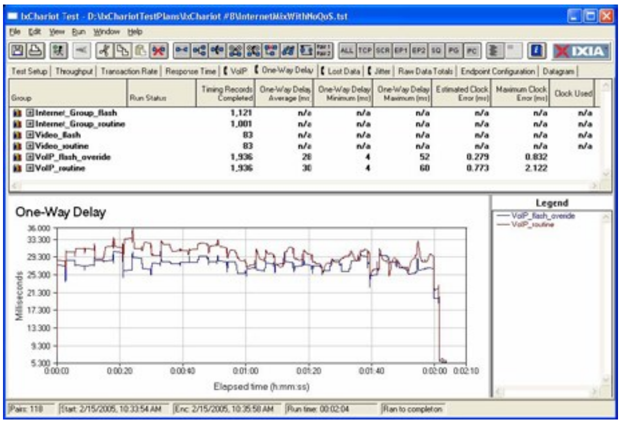
The plot of the change in throughput during the test without QoS is shown in Figure 3 a). Unidirectional latency of data flow on startup without QoS is shown in Figure 3 b).



Group	Run Status	Timing Records Completed	95% Confidence Interval	Average (Mbps)	Minimum (Mbps)	Maximum (Mbps)	Measured Time (sec)	Relative Precision
Internet_Group_flash		1,121		3.808	0.010	2.443		
Internet_Group_routine		1,001		3.320	0.011	2.213		
Video_flash		83		1.944	0.608	1.017		
Video_routine		83		1.942	0.546	1.017		
VoIP_flash_overide		1,936		1.393	0.004	0.065		
VoIP_routine		1,936		1.390	0.004	0.066		

```
Pairs: 118 | Start: 2/15/2005, 10:33:54 AM | End: 2/15/2005, 10:35:58 AM | Run time: 00:02:04 | Ran to completion
```

a)



Group	Run Status	Timing Records Completed	One-Way Delay Average (ms)	One-Way Delay Minimum (ms)	One-Way Delay Maximum (ms)	Estimated Clock Error (ms)	Maximum Clock Error (ms)	Clock Used
Internet_Group_flash		1,121	n/a	n/a	n/a	n/a	n/a	n/a
Internet_Group_routine		1,001	n/a	n/a	n/a	n/a	n/a	n/a
Video_flash		83	n/a	n/a	n/a	n/a	n/a	n/a
Video_routine		83	n/a	n/a	n/a	n/a	n/a	n/a
VoIP_flash_overide		1,936	28	4	52	0.279	0.832	
VoIP_routine		1,936	30	4	60	0.773	2.122	

```
Pairs: 118 | Start: 2/15/2005, 10:33:54 AM | End: 2/15/2005, 10:35:58 AM | Run time: 00:02:04 | Ran to completion
```

b)

Fig. 4. Results of the experiment. Throughput (a) and latency (b) characteristic changes

On the basis of the data obtained, it can be concluded that the results obtained with the help of the simulation model are adequate with a confidence probability of 0.9 with a reliability factor of the confidence interval equal to 0.1.

V. CONCLUSION

The article suggests a new approach to the modeling of software-defined networking behavior, which allows estimating the network characteristics that affect QoS indicators. A model of a network fragment is proposed, which takes into account the relationship "controller-commutator". The peculiarity of the proposed network is feedback, which is determined by the functioning of the OpenFlow protocol. Formulas for calculating such indicators as node performance, characterized by a service curve, latency and the amount of data stored in the buffer are given.

On the basis of a private model of the network, a model is proposed for calculating the latency and performance boundary indicators "from end to end" when there are several processing nodes. This model assumes the existence of one aggregated data stream, but on the basis of equations (10) - (12), its further development can be proposed, which allows modeling the behavior of processing nodes in the case of the formation of several aggregated data streams. Thus, the proposed approach can be used in the development and improvement of existing multiservice software-defined networking.

REFERENCES

[1]. Haleplidis E. *Software-Defined Networking: Experimenting with the Control to Forwarding Plane Interface*. European Workshop on Software Defined Networks (EWSDN), Darmstadt, Germany, 25–26 Oct. 2012, pp. 91–96.

[2]. Lawrence S. *Software Engineering: Theory and Practice (2nd ed.).* NJ: Prentice Hall, 2001. 630 p.

[3]. Infonetics Research Inc. (2014) SDN and NFV Strategies: Global Service Provider Survey [Online], 39 p. Available: http://alu.us.neolane.net/res/img/286758382c7e061c52883e873cee02e6.pdf.

[4]. Boudec J.-Y. Le, Thiran P. *Network calculus. A Theory of Deterministic Queuing Systems for the Internet*. Springer Verlag, 2012. 263 p.

[5]. Cruz R. L. "A calculus for network latency, Part II: Network analysis". *IEEE Transactions on Information Theory*, Vol 37(1), 1991. pp. 132–141.

[6]. Ciucu F., Burchard A., Liebeherr J. "A network service curve approach for the stochastic analysis of networks" *ACM SIGMETRICS*, 2005, P. 279–290.

[7]. Chang C.-S. "On deterministic traffic regulation and service guarantees: A systematic approach by filtering" *IEEE Transactions on Information Theory*, Vol 44(3), 1998, P. 1097–1110.

[8]. Chang C.-S. *Performance Guarantees in Communication Networks. Telecommunication Networks and Computer Systems.* Springer-Verlag, 2000. 365 p.

[9]. Schmitt J., Roedig U. "Sensor network calculus - a framework for worst case analysis" *Distributed Computing on Sensor Systems*, 2006. pp. 141–154.

[10]. 10. Skeie T., Johannessen S., Holmeide O. "Timeliness of real-time IP communication in switched industrial ethernet networks" *IEEE Transactions on Industrial Informatics*, Vol 2(1), 2006. P. 25–39.

[11]. Cruz R. L. "Quality of service guarantees in virtual circuit switched networks." *IEEE Journal on Selected Areas in Communications*, Vol 13(6), 2007, pp.1048–1056.

[12]. Shang Z., Wolter K. "Latency Evaluation of OpenFlow Network Based on Queueing Model" [Online]. Available: https://www.researchgate.net/publication/306397961_Latency_Evaluation_of_OpenFlow_Network_Based_on_Queueing_Model.

[13]. Bing Xiong, Kun Yangb, Jinyuan Zhaoc, Wei Lia, Keqin Lid. "Performance evaluation of OpenFlow-based software-defined networks based on queueing model" *Computer Networks*, Vol. 2(102), 2016, pp. 172–185.

[14]. Issam S. "Mathematical models for analysis Software-defined network". *International Journal "Information Technologies & Knowledge"*. Vol. 9 (2). 2015. pp.111 – 123.

[15]. Jiang Y. "Relationship between guaranteed rate server and latency rate server". *IEEE Global Telecommunications Conference (GLOBECOM '02).* Vol.3., 2002. pp. 2415–2419.

[16]. The ATM Forum. Technical Committee. “Traffic Management Specification”. Version 4.1 AF-TM-0121.000.

[17]. Bianco A., Birke R., Giraudo L. “Openflow switching: Data plane performance” *IEEE International Conference on. IEEE*, 2010, pp. 1–5.

[18]. Zhang H., Ferrari D. “Rate controlled service disciplines” *Journal of High Speed Networks*, Vol 4, 1994, P. 389–412.

[19]. Zhang Z.-L., Duan Z. “Fundamental trade-offs in aggregate packet scheduling”. *In SPIE* Vol. 4526, 2001.

[20]. Omnet tutotials [Online]. Available: https://omnetpp.org

AUTHOR’S PROFILE

Samer Salah Thabit
was born 12 November, 1978
2010-2012 study in Nimaz College of Osmania University, obtain master degree, Faculty of Informatics. Since 2013 postgraduate student in Al-Neelain University, Computer Sciences and Information Technology department.