Scientific Research Publishing

Communications and Network, 2021, 13, 25-35
https://www.scirp.org/journal/cn
ISSN Online: 1947-3826
ISSN Print: 1949-2421

Analysis and Evaluation of MPLS Network Performance

Fontaine Rafamantanantsoa*, Razafindramonja Clément Aubert, Rabetafika Louis Haja

University of Fianarantsoa, Fianarantsoa, Madagascar
Email: *fontainerafamant@yahoo.fr

How to cite this paper: Rafamantanantsoa, F., Aubert, R.C. and Haja, R.L. (2021) Analysis and Evaluation of MPLS Network Performance. *Communications and Network*, **13**, 25-35.
https://doi.org/10.4236/cn.2021.131003

Received: January 9, 2021
Accepted: February 23, 2021
Published: February 26, 2021

Copyright © 2021 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

Abstract

Currently, the increasing network connectivity is becoming more and more complex and the integration of several services continues to appear. These approaches need a guarantee in terms of throughput, the performance of which they potentially impact. So, the basic aim of this research is to get the best MPLS network performance towards linux and FreeBSD operating systems. This is why in this report, we have worked on MPLS network performance, which will help to identify performance metrics. To evaluate the performance, we used those operating systems implementing the MPLS architecture in order to define the best performing between the two on this domain. For this, we used scapy to measure the response times by varying the size of the packets sent and validate the measurements with the MATLAB Simulink. After all experiments, we realized that FreeBSD operating system is more reliable than linux in MPLS network base.

Keywords

MPLS, Simulink, Scapy, Performance

1. Introduction

Nowadays, we have got an important number of internet users. So as result, the existing networks in the internet network are unstoppably increasing. It meaningfully requires use of powerful equipment such as a router in order to ensure networks interconnection. A router works hardly if networks are multiple because it includes the routing tables that provide packet routing owing to discover the next receiver router of the packet (NEXT-HOP). The routing tables entries enhance (increase) with the networks raise and could be accessed sequentially by destination IP address that leads sometimes to an access time delay. That is why MPLS networks are implemented by researcher, especially for backbone interconnection networks which based on label routing.

Lots of studies and researches are done on MPLS networks performances. In [1], the authors conversed the comparative performance analysis between conventional and MPLS network. In [2], they compared VOIP communication traffic performance between the IP network infrastructure model and the MPLS one. In [3], they implemented architecture used by ISPS to confirm the existence of network performance degradation based on LSP re-signaling process. In [4], they harden the basic functionality of sending and receiving packets in an efficient manner along with label swapping used in internet backbone networks. In [5], resources optimization of heterogeneous networks is hard-bitten with the MPLS network kernel. In [6], BGP and MPLS failover functionality performance was analyzed as well. In [7], the research was focused on the Optimizing Application Traffic on MPLS-Enabled Network Links. And In [8], the authors had concentrated on the use of Multiprotocol Label Switching (MPLS) to improve IP Network Traffic Engineering. In studies from [9]-[21], they have focused on the performance analysis of voice, wan optimization (bandwith), optimal LSP selection, comparison performance, security evaluation, traffic engineering, QOS routing over the MPLS networks.

In this work, the main objective is to analyze and evaluate the MPLS network performance in open Source by implementing on Linux and FreeBSD kernel owing to find in which system the performance of this network gets better. So, this paper is organized as follows: the first section describes the MPLS network performance, the second section talks about the experimentation and the third one shows the MPLS experimental results. The fourth section gives us the traffic measures and models between the two systems and finally the conclusion is presented at the last one.

2. MPLS Network Performance

In this part, different evaluation methods of performance with all necessary parameters will be shown as it suits.

2.1. Performance Metrics

The performance evaluation requires the definition of a few parameters that represent the system to be studied. These parameters can be divided into two categories: system parameters and network load parameters. The system parameters include the hardware and software parameters that do not vary under different configurations of the system under test and the load parameters represent the characteristics of the requests. They vary from one configuration to another.

2.2. Performance Factors

The performance factor is a subset of the performance parameters that contains the parameters that vary during the performance evaluation process. Choosing these parameters is a very important step in any performance evaluation project. The values of these parameters are characterized by levels representing their degree of variance. The following factors are identified: The number of packets sent: which measures the impact of its passages in the network. Packet size: which measures the load in the network on the system under test. The type of service: measures the impact of the service requested from the system. We see that these factors relate to certain performance metrics.

3. Experimentation

The configuration of the experiments is given in **Figure 1**. In order to avoid the problem of clock synchronization, we used a single machine capturing the time between its two interfaces (capturing the start time on the router interface. input and arrival time in the output router interface in the MPLS (Multi-Protocol Label Switching) domain). In all the experiments in this work, three machines are used respectively a LER (Label Edge Router), an LSR (Label Switch Router) and another LER. These machines are in "dual boot" of two operating systems: "Linux" and "FreeBSD (Free Berkeley Software Distribution)". These systems are implemented with MPLS technology. Another machine aimed at capturing the residence time of packets passing through MPLS.

The configuration of these devices is illustrated in **Table 1**.

In different cases, we present the trial results obtained by calculating the average of the values found (the time when each packet only transits on the MPLS network) by rising the length of packets from 10 to 1500 bytes and varying the traffic type to be analyzed. For each packet, this operation is periodically repeated (especially five minutes).

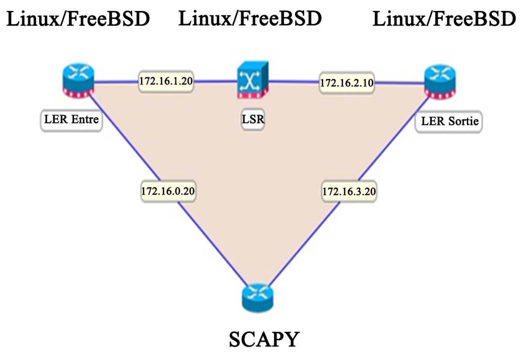


Figure 1. Topology of the experimentation.

Table 1. Devices configuration.

	Router LER	Router LSR	Router LER	Host
CPU	1.5 GHz	1.5 GHz	1.5 GHz	3.00 GHz
RAM	256 Mo	256 Mo	256 Mo	2 Go
Operating System	Linux FreeBSD	Linux FreeBSD	Linux FreeBSD	Linux
Tools				Scapy

4. MPLS Experimental Results

Several experiments were made to compare the MPLS network performance between the Linux system and FreeBSD one.

The "scapy" tool which is installed in the "Host" machine, which collects the different performance metrics results of this network such as the number and size of the packet sent, the type of service requested.

Experiment 1: relation between the size of the packet sent and the average time by collecting only UDP traffic in the network by considerably increasing the size of the packet.

Experiment 2: relation between the size of the packet sent and the average time by collecting only TCP traffic.

Experiment 3: relation between the size of the packet sent and the average time by collecting only ICMP traffic.

Experiment 4: average label stacking time depending on the number of labels.

Experiment 1: Relation between packet size and average time (UDP Traffic)

Table 2 shows the result on the linux and FreeBSD UDP traffic by MATLAB Simulink (Figure 2).

Table 2. Relation between packet size and average time (UDP traffic).

Packet size sent in bytes	Average Service Time E [s] in second	
	Linux	FreeBSD
10	0.33	0.22
100	0.43	0.31
200	0.69	0.51
500	1.03	0.76
800	1.39	1.32
1000	1.62	1.6
1200	2.15	2.0
1500	2.51	2.42

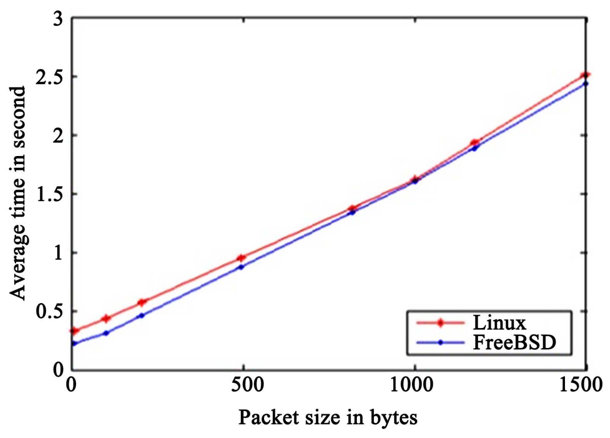


Figure 2. Relation between packet size and average time (UDP Traffic).

In the time interval [1.6; 1.62], the delay is almost the same for Linux and FreeBSD. By increasing the packet size to 1500 bytes, Linux significantly loses its performance which means that UDP has no throttling mechanism if packets can be sent without any prior contact to be determined if the host is ready for data reception.

In UDP traffic, packets have reliability problems, sent packets are not sure in the same order on the remote host. While UDP is disconnected, the connectionless makes it suitable for broadcasting many-to-many types of messages.

Experiment 2: Relation between packet size and average time (TCP Traffic)

Table 3 shows the result on the linux and FreeBSD TCP traffic by MATLAB Simulink (Figure 3).

Since the same interval as we saw earlier, the delay is also identical for Linux and FreeBSD but remains stable. This stability can be explained that in the event of congestion, a TCP sender will reduce its send rate in order to recover on the network. This adaptation function attempts to reach the highest possible data transfer rate without triggering the loss of consistent data.

Table 3. Relation between packet size and average time (TCP traffic).

Packet size sent in bytes	Average Service Time E [s] in second	
	Linux	FreeBSD
10	0.38	0.24
100	0.49	0.35
200	0.65	0.47
500	1.10	0.91
800	1.40	1.29
1000	1.62	1.61
1200	1.97	1.96
1500	2.47	2.47

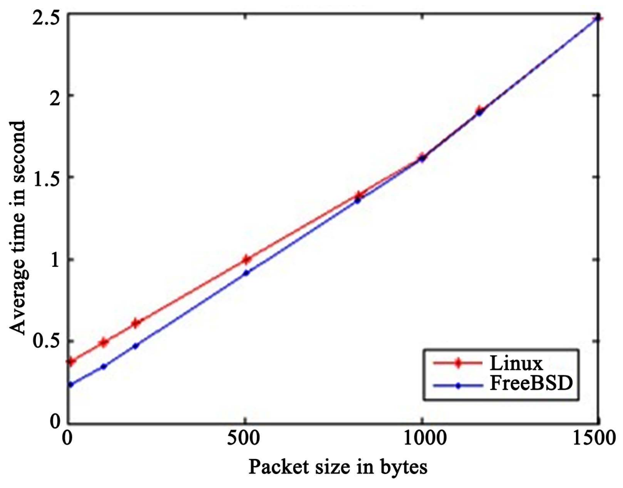


Figure 3. Relation between packet size and average time (TCP Traffic).

Experiment 3: Relation between packet size and average time (ICMP Traffic)

Table 4 shows the result on the linux and FreeBSD ICMP traffic by MATLAB Simulink (Figure 4).

Contrary to TCP and UDP, ICMP is located in layer 3 of the OSI model, so it is encapsulated in IP. ICMP error messages are transported over the network in the form of a datagram, like any data. However, in the event of an error on an ICMP message, an error frame is not sent. We can see in this figure that FreeBSD is much more efficient than Linux.

Experiment 4: Average time for stacking labels depending on the labels number function.

One of the most powerful features of MPLS is stacking labels. A marked package can carry a large number of labels, organized as a last in, first out of the stack (LIFO). Processing is always based on the top label. A label can be added to the stack (push operation) or removed from the stack (pop operation). The labels stacking allow agreeing in a single LSP (path through the MPLS network) for part of the route through a network, it is the creation of a tunnel. The LSP

Table 4. Relation between packet size and average time (ICMP traffic).

Packet size sent in bytes	Average Service Time E [s] in second	
	Linux	FreeBSD
10	0.22	0.12
100	0.26	0.14
200	0.31	0.17
500	0.41	0.23
800	0.51	0.28
1000	0.59	0.33
1200	0.68	0.39
1500	0.8	0.48

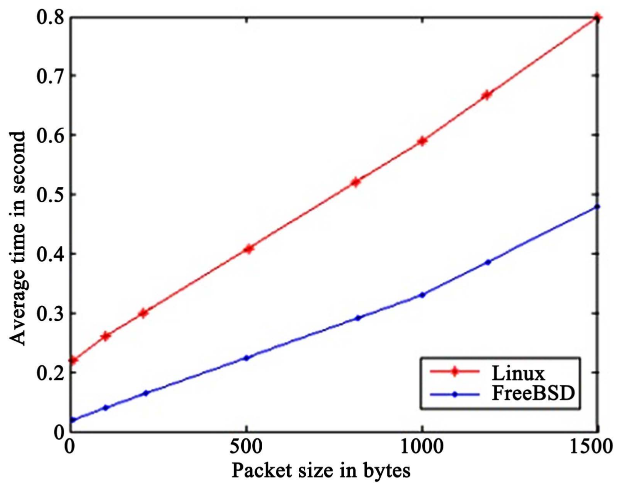


Figure 4. Relation between packet size and average time (ICMP Traffic).

starts at the Label Edge Router (LER). The LER chooses which label to attach to the package depending on the FEC. Then it forwards this packet to the next router in the way. This router forwards the packet to the LSP. He makes no decisions; it just forwards the package by swapping the label at the top of the stack. The last router in the LSP removes the label and forwards the packet based on header information such as the IP address. Since packet switching in an LSP is opaque to the upper layers, the LSP is sometimes called an MPLS tunnel.

At the start of the tunnel, a LSR assigns the same label to packets from a number of LSPs by pushing the stack label on each packet. At the end of the tunnel, another LSR appears at the top of the label stack, picking up the inner label. So stacking labels offers tremendous flexibility. A company can establish its network on different MPLS sites, establish numerous LSPs at each site and aggregate several flows of its own traffic.

In this work, we experimented with several encapsulations to analyze the progression of time when MPLS performs a stack of labels (Figure 5).

In both FreeBSD and Linux cases, the tag stacking feature has its own performance of all tested methods (depending on the number of tags). However, the variation between the transmission of labels for the two different operating systems is minimal in either case. This is because the MPLS stacking functions are shortened as much as possible so that the speeds of their executions are optimized.

5. Measures—Models

5.1. UDP Traffic (Figure 6)

The UDP traffic curve for FreeBSD of average service time E [S] as a function of packet size approximates a quadratic curve which is given by the MATLAB Simulink of formula:

$$Y = 1.8\mathrm{e}-07x^2 + 0.0012x + 0.2 \tag{1}$$

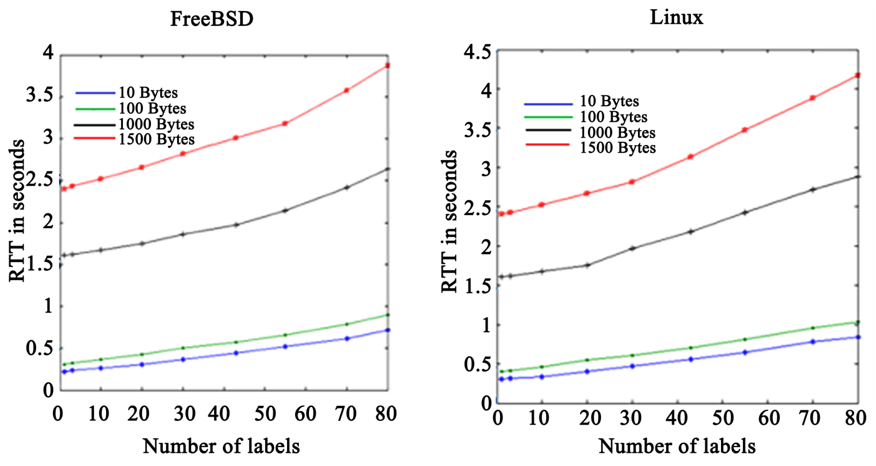
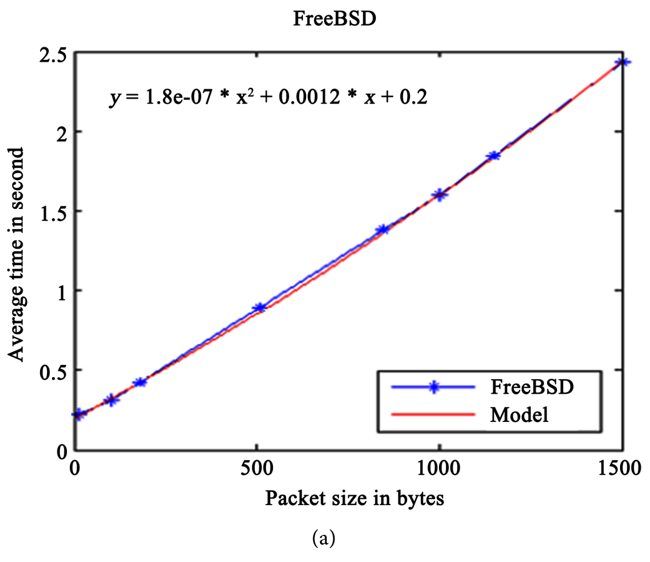


Figure 5. Average time for stacking labels depending on the labels number function between FreeBSD and Linux.



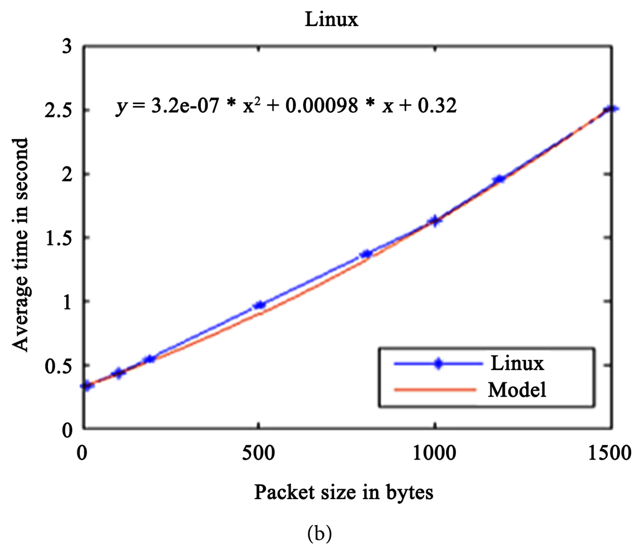


Figure 6. (a): MPLS network performance for FreeBSD: measurement and model (UDP traffic); (b): MPLS network performance: measure and model (UDP traffic).

The UDP traffic for Linux curve of average service time E [S] as a function of packet size approximates a quadratic curve which is given by the Simulink MATLAB of formula:

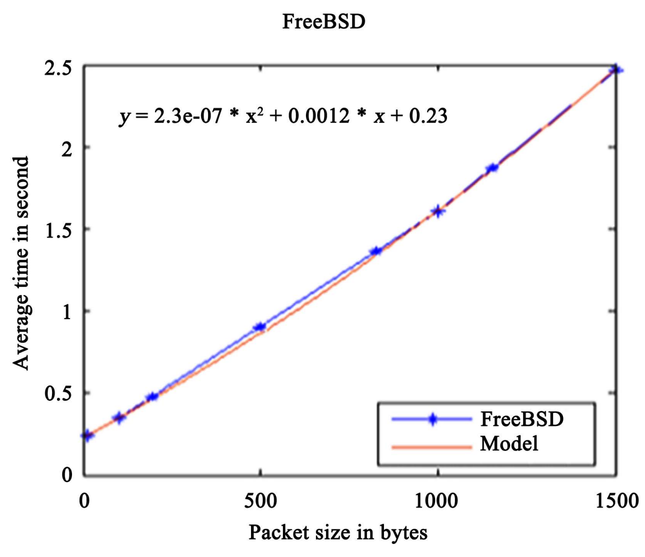
$$Y = 3.2\mathrm{e}-07x^2 + 0.00098x + 0.32 \tag{2}$$

It can be concluded that the average E [S] uptime with FreeBSD using UDP traffic is faster than the I [S] uptime under Linux.

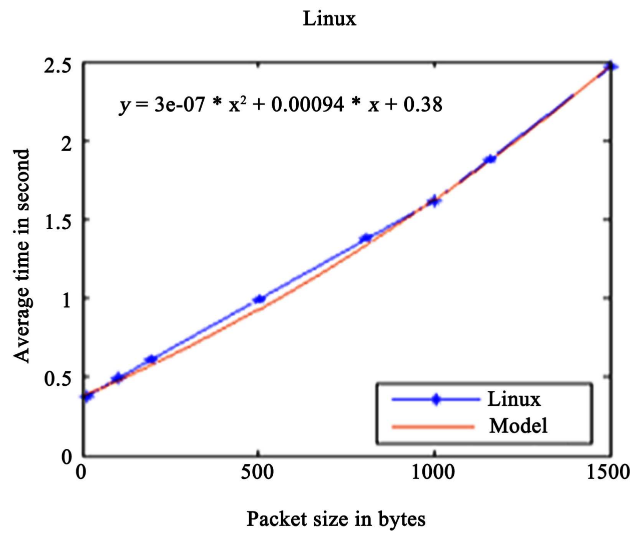
5.2. TCP Traffic (Figure 7)

The TCP traffic curve for FreeBSD of average service time E [S] as a function of packet size approximates a quadratic curve which is given by the MATLAB Simulink of formula:

$$Y = 2.3\mathrm{e}-07x^2 + 0.0012x + 0.23 \tag{3}$$



(a)



(b)

Figure 7. (a): Performance of the MPLS network for FreeBSD: measurement and model (Traffic TCP); (b): Performance of the MPLS network for Linux: measurement and model (traffic TCP).

The TCP for Linux traffic curve of average E [S] service time as a function of packet size approximates a quadratic curve which is given by the MATLAB Simulink of formula:

$$Y = 3\mathrm{e} - 07x^2 + 0.00094x + 0.38 \tag{4}$$

6. Conclusions

This work is based on the analysis of different traffics and the overload introduced in the encapsulation of the MPLS network in two different operating systems where the network topology and the different simulation parameters are chosen as common parameters in all experiments to assess the performance of the MPLS network. First, we made some real measurements in two operating systems based on an MPLS architecture. The various simulations made in this work made it possible to highlight a few observations: conventional IP routing is no longer suitable to convey multiservice. Next, MPLS is a tactic that has proven to be very effective in optimizing the use of network resources, especially since it allows the transmission of multiservice traffic while respecting the requirements of different applications. Some parameters were considered to examine the performance of this network by increasing the size of the packet sent and by changing the type of traffic to be evaluated. And finally, in an MPLS network, we can realize that Linux has its drawback in its scalability which is one of the reasons for choosing the FreeBSD-based MPLS network.

All these results are obtained after several experiments and the calculation of the average time E [S] where each packet only transits the MPLS cloud.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Rahman, A., Kabir, A.H., Lutfullah, K.A.M., Hassan, M.Z. and Amin, M.R. (2008) Performance Analysis and the Study of the Behavior of MPLS Protocols. *International Conference on Computer and Communication Engineering*, Kuala Lumpur, 13-15 May 2008. https://doi.org/10.1109/ICCCE.2008.4580601

[2] Naoum, R.S. and Maswady, M. (2012) Performance Evaluation for VOIP over IP and MPLS. *World of Computer Science and Information Technology Journal*, **2**, 110-114.

[3] Narula, D., Rojasmartinez, M. and Rayipati, V. (2010) Evaluating Performance on an ISP MPLS Network. Masters in Interdisciplinary Telecommunications, The University of Colorado, Boulder.

[4] IJERT (2012) Implémentation MPLS sous FreeBSD. IJERT, Vol. 1.

[5] Rachdi, M.A. (2007) Optimisation des ressources de réseaux hétérogènes avec cœur de réseau MPLS. Ecole doctorale: Edsys.

[6] Tan, G.C. (2006) A Performance Analysis of BGP/MPLS VPN Failover Functionality.

[7] Kingsley, O.O. (2008) Optimizing Application Traffic on MPLS-Enabled Network Links. Master of Science in Electrical Engineering, Blekinge Institute of Technology, Karlskrona.

[8] Gonzales, F., Chang, C.-H., Chen, L.-W. and Lin, C.-K. (2000) Using Multiprotocol Label Switching (MPLS) to Improve IP Network Traffic Engineering.

[9] Sourabh Jain IES, IPSA, Indore, RGPV Bhopal, Bhopal (2012) Performance Analysis of Voice over Multiprotocol Label Switching Communication Networks with Traffic Engineering. *International Journal of Advanced Research in Computer Science and Software Engineering*, **7**.

[10] Rahman, R.A., Kassim, M. and Ariffin, N. (2011) Performance Analysis on Wan Optimizations: Bandwidth Management in Multi-Protocol Level Switching (MPLS) Virtual Private Network (VPN). Faculty of Electrical Engineering, Universiti Tek-

nologi MARA, Selangor.

[11] Kuribayashi, S. (2011) Proposed Optimal LSP Selection Method in MPLS Networks. Department of Computer and Information Science, Seikei University, Tokyo.

[12] Kocak, C., Erturk, I. and Ekiz, H. (2003) Comparative Performance Analysis of Mpls over ATM and IP over ATM Methods for Multimedia Transfer Applications. Sakarya University, Technical Education Faculty, Esentepe.

[13] Abboud, K. (2011) Conception et évaluation d'un modèle adaptatif pour la qualité de service dans les réseaux MPLS. Ecole Centrale De Lille.

[14] Charbonnier, L. (2007) Evaluation de la Securite des Reseaux Prives Virtuels sur MPLS. Ecole De Technologie Supérieure Université Du Québec, Decembre.

[15] Olivier Ferveur (2009) Optimisation des architectures IP/MPLS de transport mutualisé. École Doctorale IAEM Lorraine, November.

[16] Al-Quzwini, M.M. and Ibrahim, S.K. (2012) Performance Evaluation of Traffic Engineering Signal Protocols in IPV6 MPLS Networks. *Communications and Network*, **4**, 298-305. https://doi.org/10.4236/cn.2012.44035

[17] Zeng, X.M., Lung, C.-H. and Huang, C.C. (2004) A Bandwidth-Efficient Scheduler for MPLS DiffServ Networks. Department of System and Computer Engineering, Carleton University, Ottawa.

[18] Li, Y.H. and Panwar, S. (2004) Performance Analysis of MPLS TE Queues for QoS Routing. Electrical and Computer Engineering Department, Polytechnic University, Brooklyn.

[19] Tan, C.C. (2002) Performance Analysis of Voice Traffic in MPLS Communication Networks. San Jose State University, San Jose.

[20] Zhang, D.L. and Ionescu, D. (2007) QoS Performance Analysis in Deployment of DiffServ-Aware MPLS Traffic Engineering. 8*th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing* (*SNPD* 2007), Qingdao, 30 July-1 August 2007. https://doi.org/10.1109/SNPD.2007.541

[21] Saidi, M.Y. (2008) Méthodes de contrôle distribué du placement de LSP de secours pour la protection des communications unicast et multicast dans un réseau MPLS. Université de Rennes 1, novembre.