Implement a QoS Algorithm for Real-Time Applications in the DiffServ-aware MPLS Network

Zuo-Po Huang, *Ji-Feng Chiu, Wen-Shyang Hwang and *Ce-Kuen Shieh
adrian@wshlab2.ee.kuas.edu.tw, gary@hpds.ee.ncku.edu.tw,
wshwang@mail.ee.kuas.edu.tw, shieh@eembox.ee.ncku.edu.tw
Department of Electrical Engineering,
National Kaohsiung University of Applied Sciences, Taiwan R.O.C.
*Department of Electrical Engineering,
National Cheng Kung University, Taiwan R.O.C.

Abstract

This paper presents an implementation of QoS algorithm (PPA, Preempted Probability Algorithm**)** for DiffServ-aware MPLS network under Linux platform. The algorithm, which comprises of optimal LSPs (Label Switching Paths) selection and the network resource allocation, is injected into the ingress router to verify the feasibility. The experimental results show that this approach can optimize network resources efficiently and distribute the traffic through the MPLS network.

Keywords: MPLS, Traffic Engineering, DiffServ-aware MPLS Network, and Real-Time Applications

1. Introduction

The network resources have to be managed efficiently due to the exponential growth of the bandwidth demand of new real-time Internet applications over the last years. The Internet technologies have to adapt new demands for increased bandwidth. These real-time Internet applications such as streaming, videoconference, interactive distance learning which impose throughput and delay constraints expected to get better delivery service through the Internet. The Internet architecture only offers the best-effort delivery service model, however, all customer packets are treated equally. These real-time applications are sensitive to the Quality of Service (QoS). The Internet Engineering Task Force (IETF) had proposed two fundamental techniques for supporting network QoS. These techniques are either Integrated Service (IntServ) [1] or Differentiated Service (DiffServ) [2]. IntServ is an architecture that associates and allocates resources to individual flow. It will lead to scalability problem when hundreds or thousands of flows are delivered through the backbone network. DiffServ is based on a simple model where traffic entering a network is classified at the boundaries of the network and assigned to different Behavior Aggregates (BAs) that are a collection of packets with the same Differentiated Service Code Point (DSCP) [3]. Per-flow state does not need to be maintained in the core routers, which leads to increase scalability.

The Multi-Protocol Label Switching (MPLS) integrates the label swapping of layer-2 technology with scalability. In MPLS network, the traffic is delivered through Label Switched Paths (LSPs). MPLS is also used to create LSPs for specific purposes, such as Traffic Engineering (TE). The objective of TE is to optimize network resources efficiency and improve network performance. Therefore, DiffServ and MPLS are viewed as complementary in the pursuit of end-to-end QoS provisioning at present. In the architecture, DiffServ provides the scalable end-to-end QoS, while MPLS performs TE to evenly distribute traffic load on available links and fast rerouting to route through nodes. Currently, the combination of DiffServ and MPLS is a promising technique to provide QoS, while efficiently exploiting network resources [10-11].

In traditional IP network, the shortest path is used to forward packets. This may cause congestion on a specific link.

Eventually, the link utilization is very low in the backbone network. Hence, traffic engineering is an important process to distribute traffic efficiently throughout all of the links from ingress router to egress router. In this paper, we consider how to promote the link utilization and resource management to achieve QoS requirements efficiently. First of all, we try to avoid all of the traffic congesting with the shortest path. It means that the next incoming traffic will be routed to another LSP unless the network is in the overload situation. Therefore, the link utilization will be promoted. Secondly, the higher-priority LSP will preempt the resources of lower-priority LSP when the bandwidth resources are restrained and then the lower-priority LSP has to release bandwidth, it has to be rerouted by selecting another LSP, but this LSP cannot ensure that the bandwidth resource will not be preempted again. If the situation occurs often, routers would have superfluous overhead and encounters an awful quality of service.

The PPA had proposed with simulation in the [14]. This proposed PPA can avoid preemption for every priority flow and load balancing in the MPLS networks. In order to implement the PPA under Linux platform, the PPA has to be injected into the ingress router of DiffServ-Aware MPLS network to distribute traffic efficiently.

The rest of this paper is organized as follows: Section 2 gives the operation of PPA briefly. Section 3 describes the RSVP-TE daemon for DiffServ over MPLS under Linux. Section 4 presents the experimental results. Finally, the conclusion is presented.

2. The Operation of PPA (Preempted Probability Algorithm)

The PPA was implemented in the MPLS network to support DiffServ-aware traffic engineering and all the LSPs are established by using RSVP-TE signaling from ingress router to egress router.

In the operation of PPA, the ingress router will calculate the preempted probability of each link of each feasible LSP according to the proposed PPA formula in [14] when a LSP setup request is arrival. After each link of preempted probability is calculated, the PPA selects the maximum preempted probability from each link of the feasible LSPs. And then the PPA selects the LSP to forward packets with the minimum preempted probability in all feasible LSPs. The flowchart of PPA is shown in Figure 1. More details about PPA are described in [14].

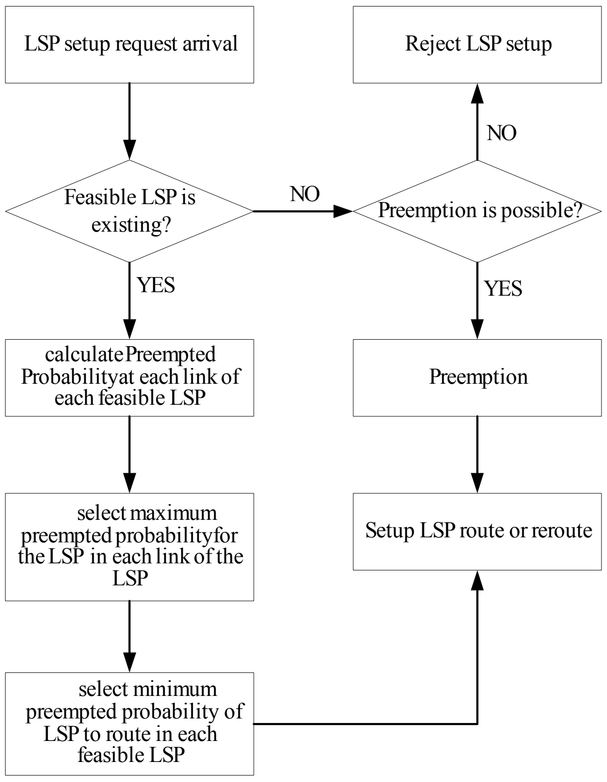

LSP setup request arrival
Reject LSP setup
NO
Feasible LSP is existing?
NO
Preemption is possible?
YES
YES
calculate Preempted Probability at each link of each feasible LSP
Preemption
select maximum preempted probability for the LSP in each link of the LSP
Setup LSP route or reroute
select minimum preempted probability of LSP to route in each feasible LSP

Figure 1. *The flowchart of PPA*

3. RSVP-TE daemon for DiffServ over MPLS under Linux

The RSVP-TE (RSVP Traffic Engineering) protocol is an extension to the Resource reSerVation Protocol (RSVP). Initially, RSVP is a signaling protocol for IntServ reservation. The RSVP-TE extends two practical functions for TE purpose in MPLS network. One is the possibility to set up LSPs and ER-LSPs, the other is the function for traffic engineering.

The overall structure works in the user space and kernel space. It is shown in Figure 2 [12]. The **netfilter** is the most important parts of the kernel space used to classify the packets, QoS and fair queuing. The chief component in the user space is the **RSVP daemon**. The daemon is bulit to response the RSVP signaling and to maintain the MPLS state. It is also responsible for the allocating and installation of the MPLS labels during LSP set up procedure. The component of **rtest** is a RAPI (RSVP Programming Interface) application that takes LSP requests and issues them to the daemon. The component of **rapirecv** is also a RAPI application that receives label requests at the egress router and sends a response back to the ingress router.

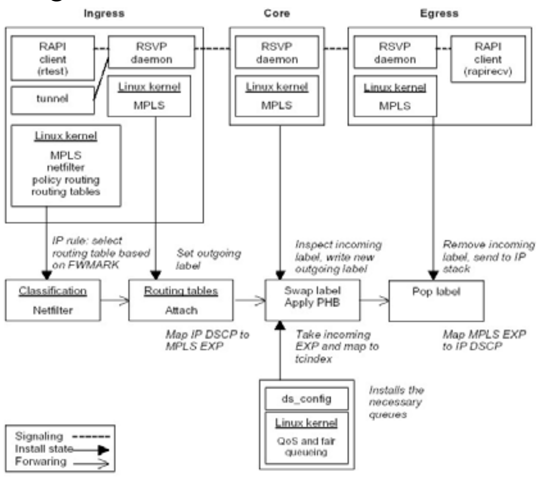


Figure 2. *DiffServ over MPLS using RSVP-TE under Linux overall architecture [12]*

4. Experiment Results

In order to verify the feasibility of PPA for the real-time applications, we constructed a DiffServ-aware MPLS domain as shown in Figure 3. The experiment platform contains three hosts (Host Adrian, Gary and Neo) and three diffserv-aware MPLS routers (LSR1 to LSR3). We injected the PPA into LSR1. Each of the links between two nodes is 10Mbps point-to-point Ethernet link unless the link from LSR3 to Host Neo is 100Mbps point-to-point Fast-Ethernet link. The host Gary generates the background traffic (marked as BE) as shown in Figure 4. The host Adrian generates the real-time traffic (marked as EF) as shown in Figure 5. The host Neo receives the traffic from host Adrian and Gary.

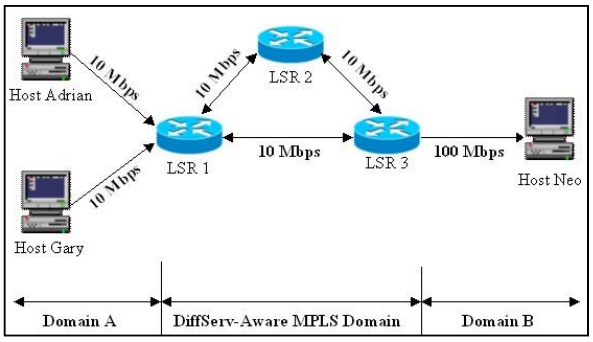


Figure 3. *Experiment platform*

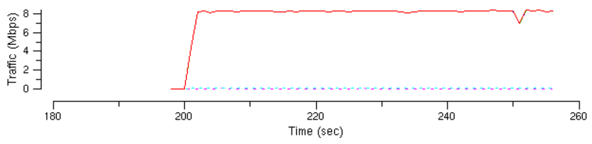


Figure 4. *Incoming best effort traffic (Gary)*

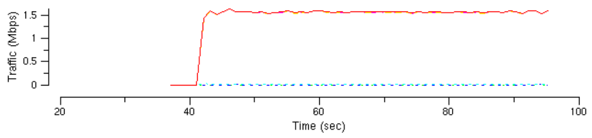


Figure 5. *Incoming real-time traffic*

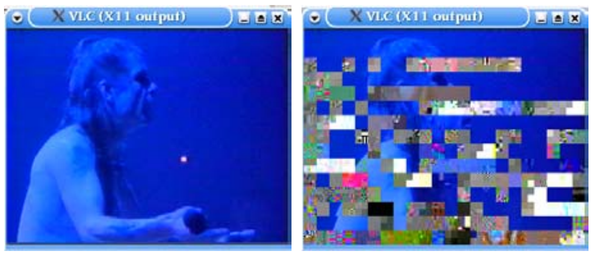
In this implementation, we treat all the packets as three different classes for DiffServ-aware MPLS network EF, AF21 and BE according to E-LSP (EXP-Inferred-PSC LSPs) [11]. Table 1 shows the PPA QoS requirement mappings in the DiffServ network over MPLS.

DiffServ class PHB	DSCP	MPLS EXP Field	MPLS Service class
EF	101110	000	Gold
AF21	010010	001	Silver
BE	000000	010	Best effort

Table 1. PPA QoS Requirement Mappings

A. Best effort delivery

We used VLC (VidelLan Client) [17] media player to play real-time traffic from host Adrian to host Neo. At the same time, the host Gary generates the overload traffic to host Neo. All the traffics are treated equally because of best effort only. Figure 6 shows that the best effort service model is used to deliver the traffic. Obviously, the packet loss occurs when the network resources of the link between LSR1 and LSR3 are unable to provide enough resources for real-time application. Thus, the receive node cannot get better QoS guarantee. The outgoing traffic is shown in Figure 7. The background traffic was decreased around 7Mbps after real-time traffic entered the network (started at eightieth second). Furthermore, the real-time traffic did not reach 1.5Mbps.



(a) Traffic on Host Adrian (b) Traffic on Host Neo

Figure 6. *Best effort traffic delivery with real-time application*

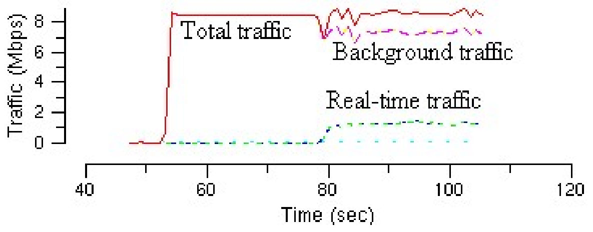


Figure 7. *Outgoing traffic of best effort*

B. DiffServ over MPLS with Preemption

Since the higher-priority LSP will preempt the resources of lower-priority LSP when the bandwidth resources are restrained, and the lower-priority LSP has to be rerouted by selecting another LSP. Figure 8 shows that the best-effort traffic was preempted (at 364 second) and rerouted to another LSP (from LSR1, LSR2 and LSR3). In this experiment, the real-time traffic could get better QoS, but the preemption times will be increased greatly when more and more traffics that are marked as different classes are delivered in the network. This problem will cause that the LSRs have to spend more time rerouting these preempted flows, and it is hard to achieve load balance purpose.

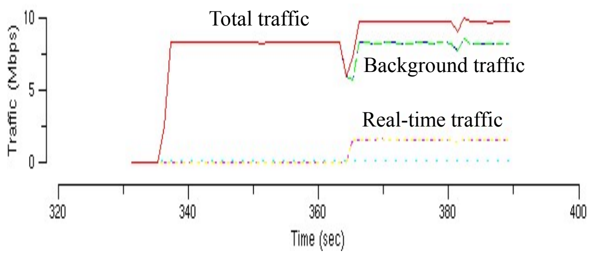


Figure 8. *Preemption and Rerouting in DiffServ-aware MPLS Network*

C. DiffServ over MPLS with PPA

We injected the PPA into LSR1. The LSR1 will select an optimal LSP to deliver traffic. With this scheme, the receive node can smoothly receive real-time traffic which is delivered from host Adrian smoothly (Figure 9). The real-time application traffic did not have any packet loss because the LSP was selected from LSR1, LSR2 and LSR3 by the LSR1, and the background traffic was not preempted by the higher-priority traffic. Hence, it can increase the throughput and link utilization. The outgoing traffic is shown in Figure 10. According to the figure, it does not only ensure the real-time traffic (keep 1.5Mbps) but also achieve an objective of preemption avoidance, and increase the throughput (from 8Mbps increasing to 10Mbps).

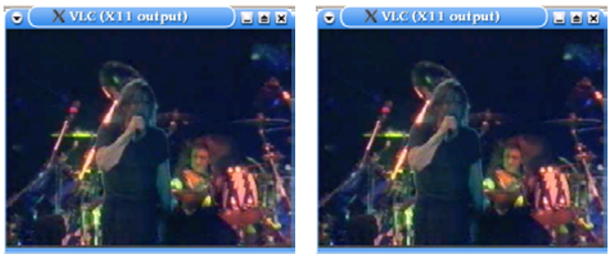


Figure 9. *DiffServ over MPLS with PPA scheme*

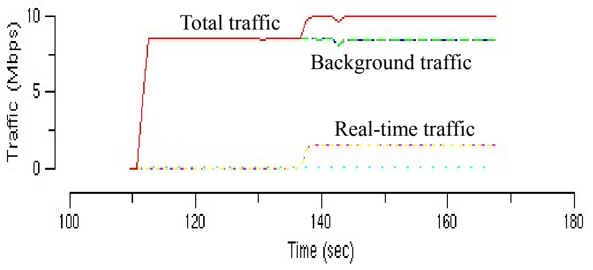


Figure 10. *Outgoing traffic for diffserv over MPLS with PPA scheme*

5. Conclusion

In this paper, we have verified the feasibility of the PPA in DiffServ-aware MPLS network for supporting the end-to-end QoS and the resource optimization by using the real-time applications. The PPA scheme could avoid preemption and load balancing in the DiffServa-aware MPLS network. The experiment results indicated that the PPA algorithm is better than traditional algorithm. Even though the higher-priority flow did not deliver the traffic by selecting the shortest path, it still achieved the expectable performance and load balancing.

Reference

[1] Braden, R., Clark, D. and Shenker, S., "Integrated Services in the Internet Architecture: an Overview", Internet RFC 1633, June 1994.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Services," RFC 2475, December 1998.

[3] K. Nichols, S. Blake, F. Baker and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," RFC 2474, December 1998.

[4] Heinanen, J., Baker, F., Weiss, W. and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.

[5] Jacobson, V., Nichols, K. and K. Poduri, "An Expedited Forwarding PHB", RFC 2598, June 1999.

[6] F. L. Faucheur, T. D. Nadeau, A. Chiu, W. Townsend, et al, "Requirements for support of DiffServ-aware MPLS traffic engineering", IETF Internet drafts, November 2000.

[7] F. L. Faucheur, T. D. Nadeau, A. Chiu, W. Townsend, et al, "Extensions to RSVP-TE and CR-LDP for support of DiffServ-aware MPLS traffic engineering", IETF Internet drafts, November 2000.

[8] E.Rosen, A.Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001

[9] D. Awduche, J. Malcolm, J. Agogbua, M.O'Dell and J.McManus, "Requirements for Traffic Engineering Over MPLS", RFC 2702, September 1999.

[10] Li t., Rekhter Y., "A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)", RFC 2430, October 1998.

[11] F. Le Faucheur, L. Wu, B Davie, S Davari, P. Vaananen, R. Krishnan, P. Cheval, J. Heinanen, "MPLS Support of Differentiated Services", RFC 3270, May, 2002.

[12] P. Van Heuven, S. Van Den Berghe, J. Coppens, P. Demeester, "RSVP-TE daemon for DiffServ over MPLS under Linux", http://dsmpls.atlantis.rug.ac.be.

[13] Lawrence. J, "Designing multiprotocol label switching networks**"**, Communications Magazine, IEEE, Volume: 39 Issue: 7, July 2001, Page(s): 134 –142.

[14] J.F.Chiu, Z.P.Huang, C.W.Lo, W.S.Hwang and C.K.Shieh, 2003, "Supporting End-to End Qos in DiffServ/MPLS Networks," 10th International Conference on Telecommunication (ICT 2003), Pages(s): 261-266.

[15] Tae-won Lee, Young-chul Kim, "Implementation of a MPLS router supporting DiffServ for QoS and high-speed switching", High Speed Networks and Multimedia Communications 5th IEEE International Conference, 2002, Page(s): 51 –55.

[16] White Paper, "Using MPLS for real-time services", Nortel Networks.

[17] VLC for Linux, http://www.videolan.org/vlc/

[18] Brain Adamson, "The MGEN Toolset", http://manimac.itd.nrl.navy.mil/MGEN

[19] Kenjiro Cho, "Software", http://www.csl.sony.co.jp/person/kic/software.html