Quality of Service using Traffic Engineering over MPLS: An Analysis*

Wei Sun Praveen Bhaniramka
Department of Computer and Information Science
The Ohio State University
Columbus, OH 43210-1277
Email: {wsun,bhaniram}@cis.ohio-state.edu

Raj Jain
Nayna Networks, Inc.
157 Topaz St., Milpitas, CA 95035
Email: raj@nayna.com

Abstract

We compare the service received by TCP and UDP flows when they share either a link or a Multiprotocol Label Switching (MPLS) traffic trunk. Since MPLS traffic trunks allow non-shortest path links also to be used, the total network throughput goes up with proper traffic engineering. In this paper, we found that if UDP and TCP flows are mixed in a trunk, TCP flows receive reduced service as the UDP flows increase their rates. Also, we found that in order to benefit from traffic engineering, MPLS trunks should be implemented end-to-end (first router to last router). If some part of the network is MPLS trunk-unaware, the benefits are reduced or eliminated.

Key topics: MPLS, Traffic Engineering, Traffic Trunk, Label Switched Path

1. Introduction

This paper presents an analysis of performance of TCP and UDP flows using MPLS traffic trunks.

MPLS stands for "Multiprotocol Label Switching"[5]. It's a layer 3 switching technology aimed at greatly improving the packet forwarding performance of the backbone routers in the Internet or other large networks. The basic idea is to forward the packets based on a short, fixed length identifier termed as a 'label', instead of the network-layer address with variable length match. The labels are assigned to the packets at the ingress node of an MPLS domain. Inside the MPLS domain, the labels attached to packets are used to make forwarding decisions. Thus, MPLS uses indexing instead of a longest address match as in conventional IP routing. The labels are finally popped out from the packets when they leave the MPLS domain at the egress nodes. By doing this, the efficiency of packet forwarding is greatly improved. Routers which support MPLS are known as "Label Switching Routers", or "LSRs". A path from the ingress node to the egress node of an MPLS domain followed by packets with the same label is referred to as a Label Switched Path(LSP).

Although the original idea behind the development of MPLS was to facilitate fast packet switching, currently its main goal is to support traffic engineering and provide quality of service(QoS). The goal of traffic engineering is to facilitate efficient and reliable network operations, and at the same time optimize the utilization of network resources. Most current network routing protocols are based on the shortest path algorithm, which implies that there is only one path between a given source and destination end system. In contrast, MPLS supports explicit routing, which can be used to optimize the utilization of network resources and enhance traffic oriented performance characteristics. For example, non-shortest paths can be chosen to forward traffic. Multiple paths can also be used simultaneously to improve performance from a given source to a destination. Based on the idea of label switching, MPLS provides explicit routing without requiring each IP packet to carry the explicit route, which makes traffic engineering easier. Another advantage is that using label switching, packets of different flows can be labeled differently and thus received different forwarding (and hence different quality of service).

In the context of MPLS, A traffic trunk is an aggregation of traffic flows of the same class, which are placed inside an LSP[1, 2](Similar trunk concepts are also defined by other authors, such as TCP Trunk in [3]). Therefore, all packets on a traffic trunk have the same label and the same 3-bit class of service (currently experimental) field in the MPLS header. Traffic trunks are routable objects like virtual circuits in ATM and Frame Relay networks. These trunks can be established either statically or dynamically (on demand) between any two nodes in an MPLS domain.

A trunk can carry any aggregate of micro-flows, where each micro-flow consists of packets belonging to a single TCP or UDP flow. In general, trunks are expected to carry

[0]This research was sponsored in part by NASA Glenn Research Center, Cleveland, OH.

several such micro-flows of different transport types. However, as shown in this analysis, mixing different transport types can cause performance problems such as starvation and unfairness for certain traffic.

Figure 1 illustrates the relationships between the various abstractions we have described above.

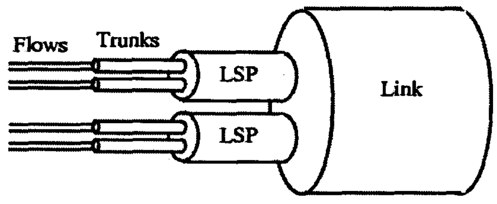

Figure 1. Relationships among Flows, Trunks, LSPs and Links

In this paper, we analyze the performance impact of mixing TCP and UDP traffic. The two transport protocols have very different congestion responses. Specifically, TCP has congestion control and reduces its traffic in response to packet loss whereas, UDP has no congestion control and does not respond to losses. While it is possible to design UDP applications that are sensitive to congestion losses, very few such applications exist. The effect of mixing TCP and UDP traffic has been studied before (such as in [10, 11, 12]), with the main conclusion that when congestion happens, TCP traffic will backoff and suffer, while UDP traffic will take advantage. The contribution of this paper is to study this effect in the context of MPLS, using traffic trunk.

This paper is organized as follows: In the following section, we describe the simulation model we have used. In section 3, we compare the results of different simulations conducted for 4 different scenarios. And finally, we give some conclusions based on our simulations in section 4.

2. Network Topology

We used Network Simulator version 2 (ns2)[8] for our analysis with new modules for label switching and CBQ (Class-based Queuing)[9] for trunk service.

In the simulations, the network topology shown in Figure 2 was used. It consists of six routers and six end-systems. The routers are MPLS capable. There are 3 flows. Source S1 sends UDP traffic to destination D1. Sources S2 and S3 send TCP traffic to destination D2 and D3, respectively(here n=3). The UDP source sends traffic at a given rate. The TCP sources are "infinite ftp" sources and send packets whenever its congestion window allows. The actual throughputs are monitored at the destination nodes. In the simulations, we use TCP Tahoe, the default version of TCP in ns2.

The two TCP flows are different only in their packet sizes. The first TCP flow (TCP1) between S2 and D2 uses a MSS of 512 bytes. The second flow (TCP2) between S3 and D3 uses an MSS of 1024 bytes. Thus, the two flows are almost identical except for the packet sizes.

From Figure 2 we can see, there exist two parallel paths between routers R2 and R5, one (R2-R3-R5) of a high bandwidth (45 Mbps) and the other (R2-R4-R5) of comparatively low bandwidth (15Mpbs). The link delay of all the links in the network is 5ms.

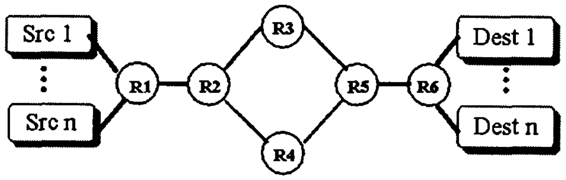

Figure 2. Network Topology

3. Simulation Results

To see the effect of MPLS traffic trunks, we analyzed four different scenarios with different intermixing of TCP and UDP flows. In each scenario, we vary the UDP rate and measure the throughput of the three flows. These scenarios and the analysis results are as follows.

* *Case 1: No Trunks, No MPLS*

The first case we analyzed is current IP with best effort shortest path routing without any trunks. All traffic in this case follows the high-speed path R2-R3-R5 and the alternative path R2-R4-R5 is unused. The results are shown in Figure 3.

Notice that as the UDP flow increases its rate, the throughputs of both TCP flows decrease. When the UDP rate reaches 45Mbps, both TCP flows can send almost nothing because all the bandwidth of link R2-R3 is used by UDP. That is, the congestion- sensitive traffic suffers at the hands of congestion-insensitive traffic. This result is the same as those of the previous studies[10, 11, 12].

Notice also that the two TCP flows that are different only in their packet sizes get very different throughput. The flow with smaller packet size (TCP1) gets very small throughput. This is a known behavior of TCP congestion mechanism[7, 8].

* *Case 2: Two Trunks using Label Switched Paths*

In this case we configured two trunks over the network in the following way. The first trunk carries UDP and

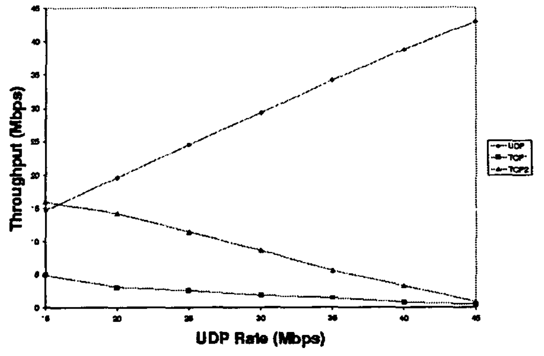

Figure 3. Results with Normal IP routing

TCP1 and follows the label switched path (LSP) R1-R2-R3-R5-R6 (high bandwidth path). The second trunk carries TCP2 and follows the path R1-R2-R4- R5-R6 (low bandwidth path). We vary the UDP source rate and observe the throughput for the various flows. The results are shown in Figure 4.

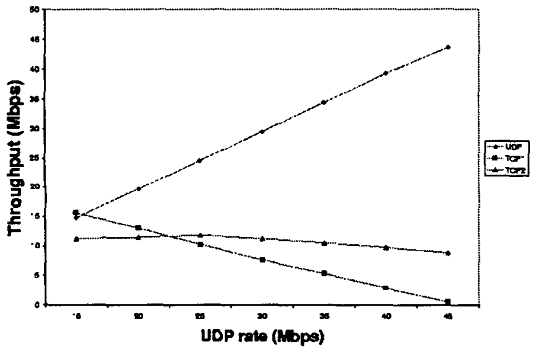

Figure 4. Results using two trunks, with TCP and UDP mixed on one Trunk

Notice that there is a significant improvement in the throughput of the flows. This is because the low bandwidth link which was not being utilized till now is also being used. Hence, TCP2 which is directed on a separate trunk and follows the low bandwidth link is unaffected by the increase in the UDP source rate and hence shows an almost constant throughput. But, we also observe that as the UDP source rate increases, the throughput of the TCP1 flow mixed with it on the same trunk suffers. This is because the TCP flow cuts down its rate in response to network congestion. Thus, we can say that use of separate trunks definitely improves the network performance but, is not sufficient to provide the quality of service which might be desirable for some applications. Thus, in order to guarantee the quality of service to a given flow, we need to isolate UDP flows and TCP flows in different trunks so that even if a given UDP flow increases its source rate, TCP flows are unaffected by this.

* *Case 3: Three Trunks with isolated TCP and UDP flows*

In this case, we use three isolated trunks, one each for the UDP and TCP flows, respectively. This is done using Class Based Queuing (CBQ) to guarantee bandwidth allocations for all the different trunks at the routers. By doing this, we essentially separate the UDP flow from the TCP flows and thus get the results shown in Figure 5.

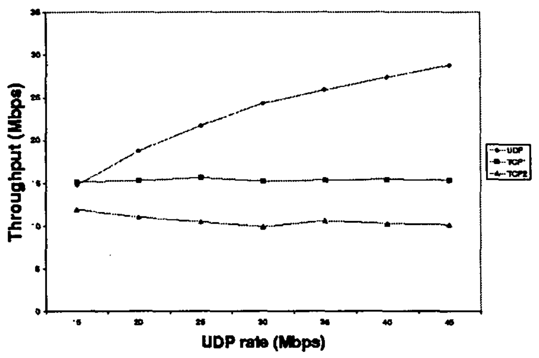

Figure 5. Results using a separate trunk for different flows

Here we see that the increase in the UDP source rate does not affect either of the TCP sources and both are able to get a fairly constant throughput. Thus, by isolating UDP and TCP flows, we can guarantee a given quality of service to sources which are responsive to congestion control also. Although this is achieved at the overhead of maintaining separate queues for each of the trunks at each of the routers. According to Li and Rekhter[1], the number of trunks within a given topology is within the limit of (N*(N-1)*C), where N is the number of routers in the topology and C is the number of traffic classes. This means the overhead is within a reasonable limit.

It is worth noting that although we use one flow in each trunk, this is not the general case. There could be multiple flows in a trunk. The simulations suggest that we should separate UDP flows from TCP flows using different trunks.

* *Case 4: Non end-to-end Trunks*

In this case we analyze a scenario where the trunks are not end-to-end. Specifically, the trunks are being initialized only from R2. In this case, the flows interfere with each other for part of the path since R1 does not distinguish between the various flows. The results are shown in Figure 6.

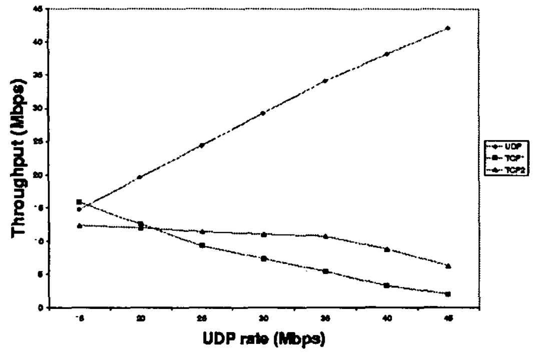

Figure 6. Results using Non end-to-end trunks

The above results show that the network really cannot guarantee anything in such a case. This happens because at R1 the flows are treated identically and during periods of congestion the TCP sources reduce their source rates. This leads to very poor throughput for the TCP sources even though they are treated distinctly at R2. Thus, in order that this kind of a scheme be successful, the flows should be separated as soon as they share a common link in the network.

4. Summary

In this paper we study the effect of using traffic trunks to separate TCP and UDP flows, which have different responses when congestion happens. Simulation results show that the total network throughput improves significantly with proper traffic engineering. Congestion-unresponsive (UDP) flows affect the throughput of congestion-responsive (TCP) flows. Therefore, different types of flows should be isolated in different trunks in order to guarantee quality of service. For the above scheme to be effective, the trunks need to be end-to-end, otherwise the advantages of isolation in other parts of the network are eliminated or reduced significantly.

References

[1] T. Li, Y. Rekhter, "A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)", RFC 2430, Oct. 1998

[2] D. Awduche, J. Malcolm, M. O'Dell, J. McManus, "Requirements for Traffic Engineering Over MPLS", RFC 2702, Sep. 1999

[3] H. T. Kung, S. Y. Wang, "TCP Trunking: Design, Implementation and Performance", ICNP'99, Nov. 1999

[4] X. Xiao, et al., "Traffic Engineering with MPLS in the Internet", 1999

[5] Eric C. Rosen, A. Vishwanathan, R. Callon, "Multiprotocol Label Switching Architecture", works in progress, Jul. 2000

[6] Eric C. Rosen, Y. Rekhter, "MPLS Label Stack Encoding", works in progress, Jul. 2000

[7] M. Allman, V. Paxson, W. Stevens, "TCP Congestion Control", RFC 2581, Apr. 1999

[8] UCB, LBNL, VINnT Network Simulator - ns2. http://www-mash.cs.berkeley.edu/ns/ns.html

[9] Sally Floyd, Van Jacobson, "Link-sharing and Resource Management Models for Packet Networks", IEEE/ACM Transactions on Networking, Vol. 3, No. 4, Aug. 1995

[10] I. Stoica, S. Shenker, H. Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks", ACM SIGCOMM'98, 1998

[11] W. Feng, et al., "Blue: A New Class of Active Queue Management Algorithms", U. Michigan CSE-TR-387-99, Apr. 1999

[12] Jim Wu, M. Hassan, "On Fairness of ATM UBR Service in Carrying Heterogeneous Internet Traffic", ICT'00, 2000