An Efficient Bandwidth Management Scheme for Real-Time Internet Applications

Fugui Wang and Prasant Mohapatra
Dept. of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
{wangfugu, prasant}@cse.msu.edu

Abstract— **Differentiated services (DiffServ) has been proposed as a scalable solution for the Internet QoS. Within the DiffServ architecture, premium services is a service class which is proposed for interactive real-time applications such as real-time voice and video over the Internet. In order to ensure the service quality of premium services, each DiffServ domain need to appropriately negotiate a service level agreement(SLA) with its customers and neighboring domains. Because the resources for premium service is usually a small part of the total network bandwidth, dynamic SLA negotiation is preferred to maximize the resource utilization. However, a completely dynamic SLA negotiation scheme introduces scalability problem for the bandwidth broker(BB). In this paper, we introduce the concept of "pipe" as a viable solution that avoids the scalability problem while managing the Internet bandwidth efficiently. A threshold-based updating scheme for the pipe is used which minimizes the updating overhead for the BB while maintaining a high utilization for the pipe.**

Keywords— **Differentiated Services, End-to-End Guarantee, Expedited Forwarding, Pipe, Quality of Service**

I. Introduction

With the proliferation of multimedia and real-time applications, it is becoming more desirable to provide certain Quality of Service (QoS) guarantee for Internet applications. The Internet Engineering Task Force (IETF) has recently proposed the Differentiated Services (DiffServ) model [2] as a scalable solution to provide Internet QoS. Currently, IETF has defined one class for Expedited Forwarding (EF) [5] and four classes for Assured Forwarding (AF) [6]. EF was originally proposed by Jacobson in [7] as Premium Service, which is ideal for real-time applications such as voice and video transmission over the Internet.

In order to maintain the service quality, each ISP domain need to control the amount of incoming traffic, which is negotiated through a service level agreement (SLA). Each domain has a bandwidth broker (BB) which manages the bandwidth resources within the domain and negotiate SLA with neighboring domains. In the current proposals [10], SLAs for assured services are usually static while SLAs for premium services are usually dynamic because they are more expensive.

Most of the current research on DiffServ are focused on service specification, service architecture and components definition[8]. Few of them [3] discuss how resource management, especially dynamic SLA, could be implemented in the DiffServ environment. Without good resource management schemes, DiffServ itself would not provide any quality of service (QoS) assurances or guarantees. As proposed in [10], a dynamic signaling process could negotiate the exact amount of SLA for the premium service traffic. In this scheme, when a new flow need to enter the domain, the BB will receive a signaling message. It will then check the resource database to see whether it can update the corresponding SLA. This will cause even worse scalability problem compared to RSVP [1] because in RSVP the signaling and reservation is distributed among all routers within the domain. In order to solve the scalability problem for BB, the SLAs for premium services should not be updated upon the join/leave of each connection. A compromising approach can be used by aggregating the SLA updating requests and periodically signaling the BB for updating SLA.

In this paper, we introduce the concept of "pipe" as a solution for resource management within a DiffServ domain. A pipe is a destination-aware SLA from the ingress router to a specific egress router. It specifies the amount of premium traffic that could flow from the ingress router to the egress router, hence forms a virtual channel with a certain bandwidth between an ingress and an egress router. When a new connection joins/leaves the pipe, it only signals the ingress router. BB only gets involved when the pipe capacity needs to be updated. Thus, we offload and distribute parts of the resource management work to edge routers so that the BB can handle the scalability problem. The result shows that the use of pipe could greatly reduce the signaling overhead on BBs, while the utilization is comparable to the per-flow-based signaling schemes.

The rest of the paper is organized as follows. Section 2 discusses the concept of pipe. Section 3 studies different implementation schemes of pipe. Section 4 discusses how to improve pipe utilization through updating. The conclusions are reported in Section 5.

II. Concept of Pipe

A pipe is defined as a logical path between two end points on the network, having a pre-defined capacity. The choice of end points depends on many factors:

Service providers' point of presence: Technically, pipe could be within an ISP domain or extend through multiple domains as shown in Figure 1. However, it is preferable to have pipe within one ISP domain for convenience of management. Usually a pipe starts from an ingress router and end at an egress router.

Traffic between regions: The purpose of pipe is to reduce the signaling and computation overhead of BB. Usually, for the same type of traffic (eg. voice traffic), the aggregate traffic gets smoother as the total amount of traffic increases. Depending on the average traffic amount between an ingress router and an egress router pair, it may or may not be necessary to construct a pipe between this pair. If the total traffic between the two points exceed a certain threshold, we can construct a pipe. Since the aggregate traffic is relatively smooth, pipe capacity need not to be updated frequently, thus minimizing its load on the BB. If a connection need to cross an ingress-egress router pair without a pipe, it has to use the dynamic signaling process through the BBs.

Dynamism in the traffic pattern: The choice of end points may also depends on the dynamism of the traffic pattern. The smoother the aggregate traffic between the end points, the higher is the benefit obtained by constructing a pipe between the two points.

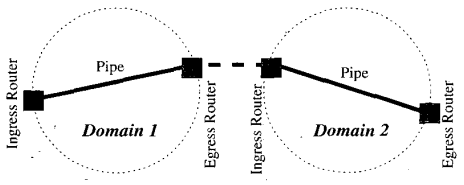


Fig. 1. Pipes in DiffServ domains.

Pipe is a destination-aware SLA. According to the definition in [2], the scope of SLA could be one of the three situation:

1. all traffic from ingress point A to any egress point
2. all traffic between ingress point A and egress point B
3. all traffic from ingress point A to a set of egress points

Pipe belongs to the second category. All of the pipes' configurations within the domain are stored in the domain's BB. BB could also store the topology of the domain and the link capacity between each pair of nodes. When a pipe needs to update its capacity, it talks to the BB and the BB decides whether the request should be granted or rejected. If the request is granted, the BB will notify the ingress router to reconfigure its traffic conditioner (TC), and thereby the SLA is updated.

When an end host need to make a connection with another end host using premium services, it contacts all the pipes it need to use in its path one by one. If the connection is admitted by all of the pipes, it will be set up successfully. Note that it only needs to signal those ingress routers. If one or more of the pipes do not have enough capacity to admit the new connection, the ingress router could reject the connection, or contact the corresponding BB to increase the pipe capacity to admit this connection, which in turn could be accepted or declined. By concatenating multiple pipes, an end-to-end QoS guaranteed connection can be established.

III. Implementation of Pipe

Unlike the virtual channel in ATM, a pipe in the DiffServ is just a destination-aware SLA. It is configured in the markers and policers in the edge routers only for admission control purpose. Once a packet enters the domain core, we cannot tell which pipe it comes from. Thus it does not need any modifications in the core routers architecture. The BB of the domain stores the pipe information. So BB knows whether the domain is able to increase a pipe capacity or construct a new pipe. There is no per-flow or per-pipe state information stored in the domain core routers. Depending on the topology of the domain, multiple pipe may share some common paths. The forwarding behavior in the core routers is only determined by the DiffServ Code Point (DSCP) [4] of the packet, which retains the per-hop behavior in the core. The service quality of the premium services is ensured through the following methods:

1. Premium service packets are queued separately and treated preferentially. The premium service queue could be implemented as a high priority queue over the RIO queue [8]. It could also be implemented as a WFQ with the RIO queue and the premium queue could be given higher weight compared to its actual traffic.
2. Pipe would ensure that the aggregate traffic through it would not exceed its capacity. Since each pipe is destination oriented, BB could make sure that for each link, the aggregate traffic of all the pipes through it would not exceed a certain percentage of the link capacity. (Usually, premium services would not occupy more than 20 percent of the total capacity of a link.) So even there is no per-pipe level service guarantee in the core routers, the QoS of each pipe could still be ensured.

We set up the following experiment to show the service quality of the aggregate scheme under different queuing disciplines. We use the *ns* [9] simulator to implement different queuing disciplines which include priority queuing (PQ) and different WFQs. Figure 2 shows the network topology that is simulated. Four nodes: n0, n1, n2, n3 are part of the network core. The link bandwidth of n0-n1, n1-n2, n2-n3 are 10Mbps, 5Mbps, and 10Mbps, respectively. A pipe with capacity of 10 voice streams exists between n0 and n3. A pipe with capacity of 15 voice streams also crosses link n0-n1. Similarly, a pipe with capacity of 3 voice streams and a pipe with capacity of 15 voice streams cross link n1-n2, n2-n3, respectively. These three pipes (n0-n1, n1-n2, and n2-n3) are used as cross traffic. All of the voice traffic use the premium services through pipes. The remaining capacity of the links are used by the best-effort TCP traffic. The premium service queue are implemented in all of the core routers n0, n1, n2, and n3.

We pick up one voice stream from the pipe between n0 and n3 to study the voice packet delay and jitter. In this experiment, we used several queuing disciplines:

1. *Priority Queue (PQ):* There are two queues in the router, one for the premium services and the other for best-effort services. The premium service queue has higher priority over the best-effort service queue. Thus, all the packets of the premium service queue are served ahead of

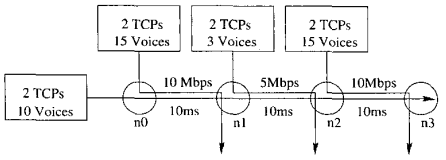


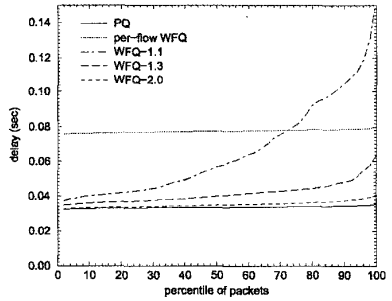
Fig. 2. Simulation topology.

the best-effort service queue.

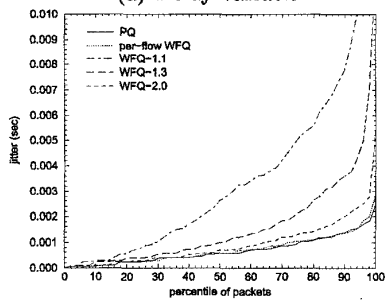
2. *Weighted Fair Queue (WFQ):* Same as PQ except that the two queues are weighted fair queues. In order to ensure the quality of premium services, the weight assigned to the premium service queue is more than the actual amount of traffic. For example, WFQ-2.0 means if the average premium service traffic is 1Mbps, we assign a weight equal to 2Mbps capacity to the premium service queue.

3. *Per-flow WFQ:* Each micro-flow has its own queue and is given the weight equal to the peak rate of the voice stream. This queuing scheme is not advisable for DiffServ. We use the result to compare with the first two schemes used in DiffServ, and show how well the aggregate schemes work.

In the simulation, we use the ITU G.711 PCM [11] VoIP traffic as our voice source. The bandwidth of each voice during talk spurt is 87.2 kbps, including the relative protocol headers. Each packet size is 218 bytes. The average burst time is 0.4 second. Average idle time is 0.6 second. It is an exponential ON/OFF model. The packet size of TCP flow is 1000 bytes.



(a) Delay variation.



(b) Jitter variation.

Fig. 3. Delay and Jitter in a pipe.

The simulation result in terms of delay and jitter for one of the voice stream from n0 to n3 is shown in Figure 3. Figure 3(a) is the plot of delay variation and Figure 3(b) is the plot for jitters. Observe from the graph that PQ has the best quality, and WFQ-2.0 has almost the same performance as PQ. The performance of WFQ-1.3 is also acceptable. In term of delay, PQ, WFQ-2.0, and WFQ-1.3 work even better than per-flow WFQ. The reason for this could be intuitively explained as follows. Per-flow WFQ could be deemed as a service in which each voice stream uses a thin link and the thin links are separated from each other. PQ, WFQ-2.0 and WFQ-1.3 use a fat link for all of the voice streams. The thin link will stretch individual packet, hence introduce longer delay. The fat link could transmit individual packet much faster, even though the link utilization are similar in both cases. However, per-flow WFQ has almost a fixed delay. So the jitter of per-flow WFQ is as good as PQ. The following conclusions could be drawn from these results:

1. Premium services could be implemented with a PQ or an aggregate level WFQ if the relative weight is given higher than 1.3. The performance is comparable to per-flow level WFQ.

2. With appropriate policing at the entrance of pipe, aggregating multiple pipes would not have much side effect on the quality of premium services.

IV. Improving Pipe Utilization Through Updating

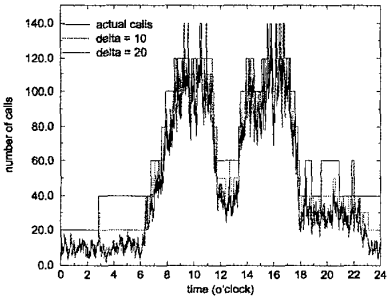
Depending on the burstiness of the aggregate traffic through the pipe, pipe capacity could be set as static or dynamic. If it is static, then all of the admission controls are done at the entrance of the pipe, BB will not receive any updating message from the pipe. However, in order to limit the rejection rate, we will have to reserve the peak of the traffic as the pipe capacity. This, of course, will make the utilization very low given the fact that the traffic could vary greatly during different time of a day and probably vary during different days. On the other hand, we can make the pipe capacity completely dynamic, that is, upon each new call arrival, we update the pipe capacity. In this case, the utilization could be up to 100 percent. However, BB will receive one updating message upon each call arrival/departure, which increases the load and thus defies the benefit of building pipes. So there is always a trade-off between the utilization and the updating overhead. One possible solution is, instead of using static or completely dynamic pipe, a hybrid scheme can be used by updating the pipe capacity periodically. The period is set to several seconds or minutes so that the updating overhead is negligible or acceptable. At the same time, the utilization could still be kept at a high level.

In order to show the effectivenes of the updating method, we built a simple simulator to mimic the arrival/departure process of telephone calls. The parameters were selected such that the arrival/departure process behaved similar to the intensity of telephone calls as published by the British Telecommunications. The length of each call is exponentially distributed with an average of 5 minutes. The 100 calls and 1000 calls referred in the results indicate the average number of calls in the pipe.

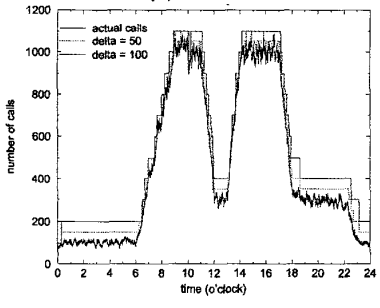
We propose a simple prediction scheme called *threshold-based prediction.* The motivation for the threshold-based prediction scheme is very simple: if we use the per-flow updating scheme, we need to increase the pipe capacity by one for each call arrival and decrease the pipe capacity by one for each call departure. This could make sure that the pipe has the highest resource utilization, but will incur very high updating overhead. However, if we increase the pipe capacity by δ ($\delta > 1$) when a new call arrives and the pipe is full, then we do not have to update the pipe capacity for every incoming arrival as long as the total number of calls does not exceed the new pipe capacity. By reserving more than we actually need right now, we loose a bit utilization, but we may be able to save a lot of updates, given the fact that the total number of calls in the pipe will not change drastically during a short period. The value for δ is selected based on the trade-off analysis between overhead and utilization. When the number of calls in the pipe drops below a threshold, we can decrease pipe capacity in order to increase the utilization. But we do not decrease the pipe capacity to the exact amount of calls. We can keep it higher than the actual number of calls so that the new coming calls will not trigger a new updating. The algorithm can be stated as follows:

1. Upon a call arrival, if the number of calls reaches the capacity of the pipe (*pipe_cap*), then *pipe_cap* is increased by δ.
2. Upon a call departure, if the number of calls is under $pipe\_cap - 2 \times \delta$, then *pipe_cap* is decreased by δ.

If $\delta = 1$, then this scheme is same as the per-flow updating scheme. Usually, δ is selected larger than 1. The larger δ is, the less frequently the pipe capacity is updated.



(a) 100 calls.



(b) 1000 calls.

Fig. 4. Threshold prediction (δ=delta).

The results of the threshold-based updating are shown in Figure 4. Figure 4(a) is the plot for the pipe with 100 calls. Here the gray line is the actual number of calls in the pipe. With the smaller δ, we have higher utilization, however, the updating is also more frequent. Figure 4(b) is the plot for the pipe with 1000 calls. We can observe that the pipe capacity predictions in Figure 4(b) are closer to the actual traffic compared to that in Figure 4(a) because the traffic is smoother. Instead of updating the pipe capacity periodically, we only update the pipe capacity when the actual number of calls falls out of the region between the upper and lower thresholds. So in Figure 4, the updates are less frequent during 0am to 6am, but are more frequent during 6am to 9am.

V. Concluding Remarks

In this paper we introduced the concept of "pipe" as a solution for resource management of premium services in the differentiated services environment. Pipe could be implemented in an aggregate level as a destination-aware SLA. We use VoIP traffic as an example and show the QoS of a voice steam under different queuing schemes of premium services. Since pipe is relative static compared to the per-flow completely dynamic resource management scheme, it greatly reduces the signaling overhead on bandwidth brokers. In order to improve the utilization of pipe, we proposed the threshold-based updating scheme. Through simulation, we have shown that this updating scheme incurs very little overhead while providing high utilization.

References

[1] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, *"Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification,"* RFC 2205, Sept. 1997.

[2] Y.Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Berma, *"A Framework for Differentiated Services,"* Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-diffserv-framework-02.txt, Feb. 1999.

[3] F. Reichmeyer, L. Ong, A. Terzis, L. Zhang, R. Yavatkar, *"A Two-Tier Resource Management Model for Differentiated Services Networks,"* Internet Draft, Nov. 1998.

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

[5] V. Jacobson, K. Nichols, K. Poduri, *"An Expedited Forwarding PHB,"* Internet Draft, <draft-ietf-diffserv-phb-ef-02.txt>, Feb. 1999.

[6] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, *"Assured Forwarding PHB Group,"* Internet Draft, <draft-ietf-diffserv-af-06.txt>, Feb. 1999.

[7] V. Jacobson, *"Differentiated Services Architecture,"* Talk in the Int-Serv WG at the Munich IETF, Aug. 1997.

[8] D. D. Clark and W. Fang, *"Explicit Allocation of Best Effort Packet Delivery Service,"* Tech. Report, MIT Laboratory of Computer Science.

[9] UCB/LBNL/VINT Network Simulator-ns(version 2), *"http://www-mash.cs.berkeley.edu/ns/".*

[10] K. Nichols, V. Jacobson and L. Zhang, *"A Two-bit Differentiated Services Architecture for the Internet,"* Internet Draft, Nov. 1997. Available at diffserv.lcs.mit.edu.

[11] International Telecommunication Union (ITU), *"http://www.itu.int/".*

[12] P. Goyal, A. Greenberg, C.R. Kalmanek, W.T. Marshall, P. Mishra, D. Nortz, K.K. Ramakrishnan, *"Integration of Call Signaling and Resource Management for IP Telephony,"*IEEE Network Magazine, VOl 13, No. 3, May/June 1999, pp-24-32.