Internet Engineering Task Force
INTERNET DRAFT
Expires March 2000

Yonas Araya
Barry Beggs
Liang Li
Stephen Nadas
Govindaraj Sampathkumar
Gaurang Shah
IBM

Vinod Peris
Growth Networks

September 1999

IBM Diffserv Implementation Experiments
<draft-nadas-diffserv-experience-01.txt>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This informational memo contains a description of some measurements from early DiffServ (DS) experiments that used a preliminary version of IBM's DiffServ implementation on an IBM router over an egress T1/E1 PPP interface. The service level specification (SLS) may be configured on the router or optionally controlled by an LDAP-accessible policy repository. The measurements were intended to examine the behavior of a mixture of Expedited Forwarding (EF) traffic, Assured Forwarding (AF) traffic and Best Effort (BE) traffic on a congested E1 link. The router code includes an LDAP client that communicates with a policy server to obtain the policies that control the amount of resources configured for the DiffServ classes.

The diagrams referred to in this memo are available in the postscript version of this memo.

Changes from -00 version

Added the details of the buffering setup, a discussion of an anomaly in the AF delay results, added an additional measurement point related to this anomaly, updated author list, added acknowledgements, and miscellaneous editorial comments.

1. Introduction

This informational memo contains a description of some measurements from early DiffServ [2475] experiments that used a preliminary version of IBM's DiffServ implementation on an IBM router over an egress T1/E1 PPP interfaces. The service level specification (SLS) may be configured on the router or optionally controlled by an LDAP-accessible policy repository. The measurements were intended to examine the behavior of a mixture of Expedited Forwarding (EF) traffic [2598], Assured Forwarding (AF) traffic [2597] and Best Effort (BE) traffic on a congested E1 link. The router code includes an LDAP client that communicates with a policy server to obtain the policies that control the amount of resources configured for the DiffServ classes.

The main purpose of this memo is to assess the feasibility of providing DS function on low-end routers, e.g. those routers connecting branch offices and central sites. We are also looking at the effectiveness of the EF and AF PHB implementations.

The end-stations used in the experiments were personal computers running FreeBSD [FREEBSD] versions 2.2.7 and 2.2.8. The applications were the MGEN [MGEN], generating UDP traffic, and NETPERF [NETPERF], generating TCP traffic. This traffic entered the router on a 10 Mbps or 100 Mbps Ethernet interface and exited the router on a 2Mbps E1 link, completely congesting it. Policy in the router classified the UDP traffic into EF traffic and the TCP traffic in to AF and BE traffic. In order to provide differentiated services on the congested E1 link, IBM's DS implementation was used. The policy information used by the router client was stored in an LDAP [LDAP] accessible policy server.

This memo is organized as follows. The next section describes the implementation, the test bed, and measurements procedures that were used to make these measurements. Section 3 presents some results of these measurements and some conclusions.

2. Implementation, Measurement Environment and Setup

In this section we describe the IBM implementation, the measurement test bed, and the experiment setup.

2.1 IBM DiffServ Implementation

The IBM 2212 routers are low cost, edge routers with interfaces for LAN (Ethernet, Token Ring) and WAN (PPP and Frame Relay on T1/E1 or T3/E3). It consists of a central processing unit responsible for all packet forwarding and classification functions, to which a number of link adapters are connected. The DiffServ (DS) implementation can be divided into two major pieces; one being the classification of the incoming packets and the other being the scheduling and buffer management required to provide rate guarantees. The classification is in itself a complex function that deserves an extensive discussion. It is, however, beyond the scope of this draft whose primary focus is on the mechanisms used to implement service differentiation. The implementation is described in some detail in [INFOCOM99]; a high-level description follows.

The information required to classify the incoming packets as well as their service level is obtained from a policy database which can be store in an LDAP policy server. The router includes an LDAP client that is configured to contact the LDAP policy server at startup time, and periodically thereafter. At the time of box initialization the policy rules are downloaded from the LDAP server (optionally they can be locally configured on the router) and are used to initialize a Common Policy Engine (CPE) that is responsible for the MF classification. Once a packet is classified by the CPE its flow information is cached so that subsequent packets do not require a full classification. Policies in the CPE consist of different actions, in particular, a policy may have a DiffServ action which specifies the bandwidth amount and queue.

Our key design objective was a lightweight implementation that would provide delay differentiation between EF traffic and other traffic as well as the ability to provide rate guarantees for AF traffic. The delay differentiation was effected by maintaining two queues at each of the output interfaces which were served by a variant of the WFQ scheduler. These queues are called the premium queue (for EF traffic) and the assured/best-effort queue (for AF and BE traffic). The scheduler weights for each of the queues can be adjusted based on the level of delay differentiation that is desired between the EF and AF traffic. The number of queues was limited to two to minimize the sorting overhead when a packet has to be selected for transmission by the scheduler.

We introduce the notion of a stream, which is the unit of resource allocation. A stream is an aggregation of (micro)flows. All the (micro)flows whose policies refer to the same DiffServ action are aggregated into a single stream. Traffic conditioning actions are independently applied to streams. Each stream is associated with one of the two scheduler queues and has a certain amount of buffer assigned to it. If the traffic in a stream becomes idle for a timeout period, the buffer is reclaimed for use by other streams, and reallocated if and when the stream reactivates. Alternatively, streams can be viewed as logically separate queues on the two physical premium and assured/best-effort queues. In our experiments we define one stream in the

premium queue for EF and five streams in the assured/best effort queue to represent the four AF classes and one for the best-effort class.

When the first packet (of a flow) arrives it is MF classified by the CPE. Assuming the policy rule does not require this packet to be dropped, the CPE along with the DS component returns a stream identifier for the traffic. The stream identifier locates the description of the PHB and the traffic parameters used for traffic policing as well as buffer management. The stream identifier is then installed into the forwarding cache so subsequent packets from this flow will not require a CPE lookup, but will directly proceed to the buffer-management module using the cached stream identifier.

Traffic from an EF stream is policed with a simple token bucket using the rate obtained from policy. All traffic, either EF, AF or BE is then processed by the Rate Based Buffer Management module, again using rates obtained from the CPE. Each QoS (EF or AF) reserved stream has some private buffers that are dedicated for their use. In addition, there is a pool of buffers that is shared by both the non-QoS streams and the excess traffic in the QoS streams. Note that this allocation of the buffers is done purely through an accounting procedure and there are no physical buffers that are dedicated to the individual streams.

Rate guarantees are provided to a stream by using a simple buffer management scheme described in [SIGCOM98]. At the time of stream instantiation a certain amount of private buffers is allocated to each stream. The number of private buffers assigned to a stream is determined by the rate guaranteed to that stream and the rate of the egress link. When a packet arrives, if the stream's private buffer is available then the packet is enqueued for transmission. If not, the shared buffer pool is checked; if there is shared buffer available the packet is enqueued for transmission. To prevent a single stream from monopolizing all the shared buffer we implement the "holes" feature described in [SIGCOM98]. The basic idea is that each stream is restricted to using at most 1/2 of what is available in the shared buffer pool.

Packets that are enqueued are removed from their queue by a Self-clocked Fair Queuing (SCFQ) scheduler which is a variant of the WFQ [SCFQ]. Since the premium queue is assigned a high scheduling weight (90% by default) the EF packets see relatively low delay. This is preferable to having a strict priority between the two queues, as it provides some degree of control in the relative priorities assigned to the two queues and prevents complete starvation of the assured/best-effort queue.

2.2 Test Bed

Figure 1 shows the configuration of the test bed for these measurements. The sending end-system, taconic, uses MGEN to send EF traffic (UDP) and NETPERF to send AF traffic and BE traffic. The AF and BE traffic is TCP. The receiving end-systems are esopus, neversink, delaware, lab760el, h-one and phoencia. All

of these systems have their system clocks synchronized using NTP [NTP] to the system called olive, which is also the LDAP policy server. There is a separate data path for NTP traffic so that the clock synchronization is not affected by network congestion during tests. During the measurements the clocks were synchronized to an accuracy of several micro-seconds.

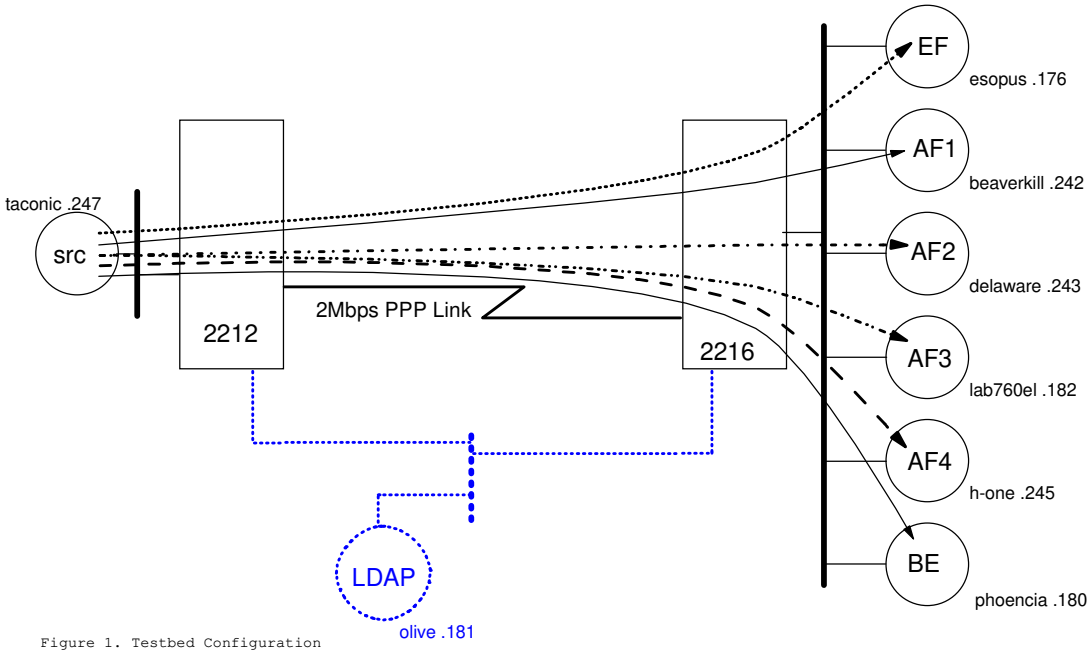

Figure 1. Testbed Configuration

Since a single NETPERF flow between the sender and a single receiver can completely congest the E1 link, this test arrangement can be used to determine some interesting items. Does EF traffic see improved delay compared with AF traffic? Do streams receive at least the amounts of resources that their policies specified?

At the first router, policies are in place to classify the MGEN traffic sent to esopus as EF, the NETPERF traffic sent to beaverkill as AF1, the NETPERF traffic sent to delaware as AF2, the NETPERF traffic sent to lab760el as AF3, the NETPERF traffic sent to h-one as AF4, and the NETPERF traffic sent to phoencia as BE. The sending system, taconic, is also sending low frequency MGEN packets to each of the AFx and BE receivers to obtain delay information.

As a sample scenario, the policies in place for this set of measurements are:

Class	Percentage of Egress E1 link
EF	19%
AF1	15%
AF2	10%
AF3	10%
AF4	5%

Table 1. Policy Configuration

In addition 10% of the link bandwidth was reserved for Best Effort traffic. This was done to ensure that traffic from legacy applications (regular unmarked traffic) is not completely denied transmission opportunities during periods of congestion. The amount of bandwidth that is reserved for Best Effort traffic is configurable. The level of delay differentiation between the EF and AF/BE traffic is configured through the setting of the scheduler weights. As mentioned earlier, the scheduler weight for the premium queue was set to 90%, and the scheduler weight for the assured queue was set to 10%. In this set of measurements, the EF traffic consisted of small UDP packets while the AF and BE traffic was large TCP packets. We tried three different TCP MTU sizes to see the effect on delay.

Measurements are made by clearing the statistics on the 2212, starting the MGEN receivers, and then starting the MGEN senders and NETPERF senders. The traffic is sent for two minutes and then the data is saved; measurement points are repeated to check for stability. Data is collected at the end-systems as well as at the router. This information is presented in the following section.

3. Preliminary Results

3.1 Preliminary Results and Discussion

Two charts suffice to show the results of these experiments. Figure 2 shows the throughput results and Figure 3 shows the delay results. In these sets of runs, the buffering was configured as follows. The ingress ethernet interface was configured with 100 receive buffers. The egress PPP interface, that is, the DS interface, was configured with 50 packets worth of buffering, or 27500 bytes.

In figure 2 (and table 2), the average throughput for each traffic class are presented. These results are taken from the 2212 router viewpoints as well as from the end-system viewpoint. The results labeled "Router measurements(%)" are the results that the router sees; these numbers include packet (IP and link) headers. The results labeled "Host payload throughput(%)" are the results that the end-systems report. These do not include the headers, and since the EF packets were small, this explains the large difference for the EF class. The results labeled "Policy configuration(%)" shows the percentage of the output bandwidth configured for that class. Finally the results labeled "Policy + expected share(%)" show the amount of bandwidth that each class should receive if the excess bandwidth were shared evenly. As mentioned in section 2.1, when there are N streams, this implementation will not allow any one stream to take more than 1/N of the remaining shared buffers [SIGCOMM98].

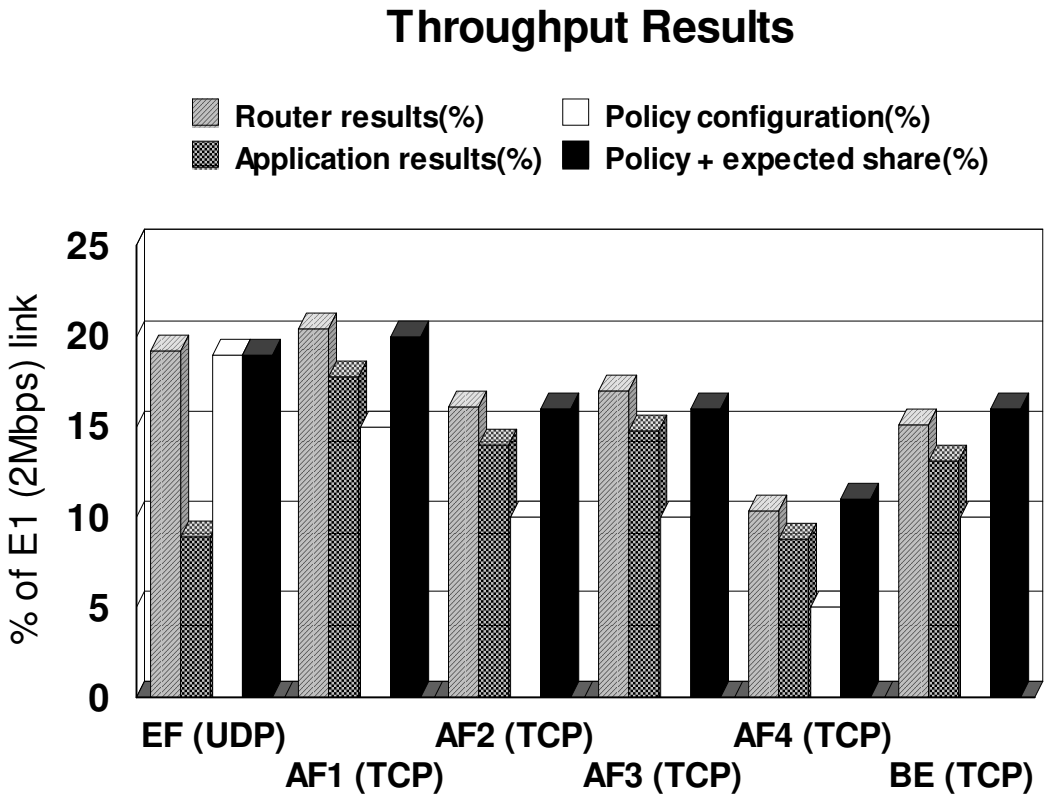

Figure 2

	Router Results %	Application Results %	Policy Configured %	Policy plus Expected %
EF (UDP)	19.2	9.0	19.0	19.0
AF1 (TCP)	20.4	17.7	15.0	20.0
AF2 (TCP)	16.1	14.0	10.0	16.0
AF3 (TCP)	16.9	14.7	10.0	16.0
AF4 (TCP)	10.3	8.8	5.0	11.0
BE (TCP)	15.1	13.1	10.0	16.0

Table 2. Throughput Results (Figure 2's data)

This graph and table shows several interesting points:

The packet header overhead can be clearly seen in the EF (UDP) results. It should be noted that there is no expected sharing for EF, thus the amount of traffic the router sees and the policed rate (and the policed rate plus the expected sharing) are all the same.

The AF results shows several phenomenon: (1) For this AF (TCP) traffic there is a only a very small difference due to the header

overhead. (2) Perhaps most importantly, streams are always getting at least their policy specified rate. (3) However, note that rate guarantees and the sharing of excess buffers are fairly coarse and that the streams do not always get exactly their policy plus expected share.

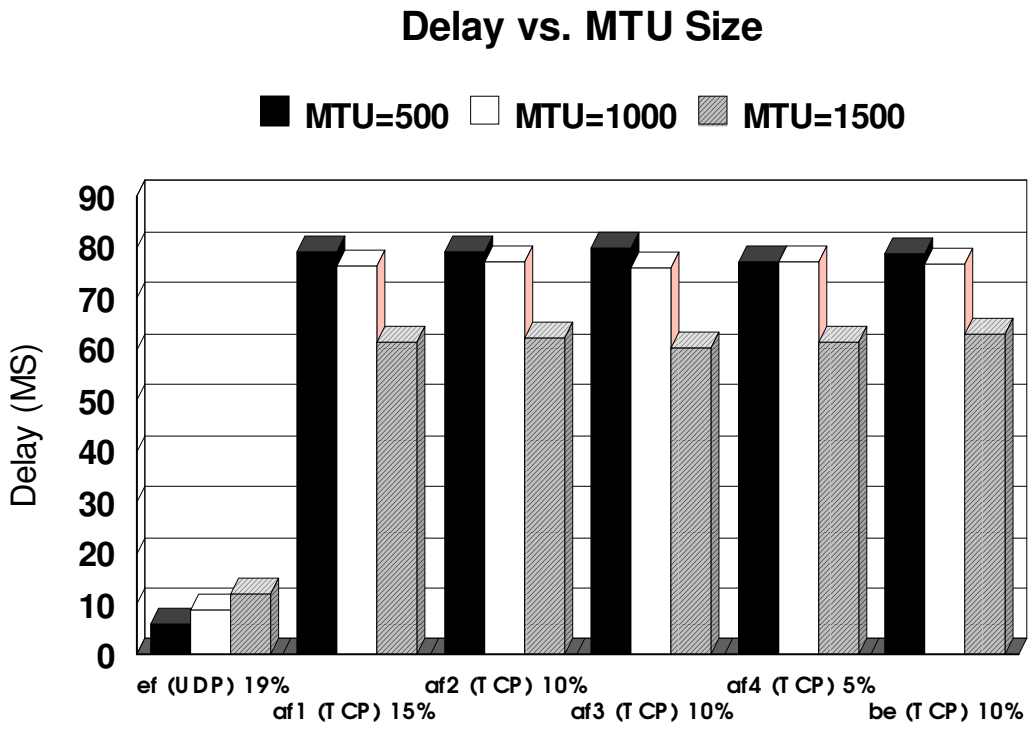

Figure 3

	MTU=500	MTU=1000	MTU=1500
EF	6.0	9.0	12.0
AF1	79.3	76.3	61.3
AF2	80.0	77.3	62.3
AF3	77.3	77.3	60.3
AF4	78.7	76.7	61.3
BE	78.6	76.7	63.0

Table 3. Delay vs. MTU Size (Figure 3's Data)

In figure 3 (and table 3), the results for delay are presented. These are always results from the MGEN tool running on the end-stations. These results show the average delay that packets in each of the classes experience. These results show that EF traffic is receiving preferential treatment over AF and BE traffic. The MTU sizes for the runs were as shown. In the test, EF packets can be delayed by up to two AF packets; for the 1500

byte MTU case this would mean a delay of nearly 12 milli-seconds to move those packets across the E1 link. This is quite close to the delay that the EF packets receive. EF traffic can see better absolute delays if the MTU size can be lowered for the TCP traffic.

Although the delay performance is not a part of the AF PHB, we observed that the AF delay appears to be inversely related to the MTU size. A question was raised as to why this is the case. The experiment was repeated with a larger amount of buffering: In this case, the ingress ethernet interface had 200 receive buffers and the egress PPP interface had 200 packets worth of buffering or 110000 bytes. Figure 4 and Table 4 show the results of this experiment.

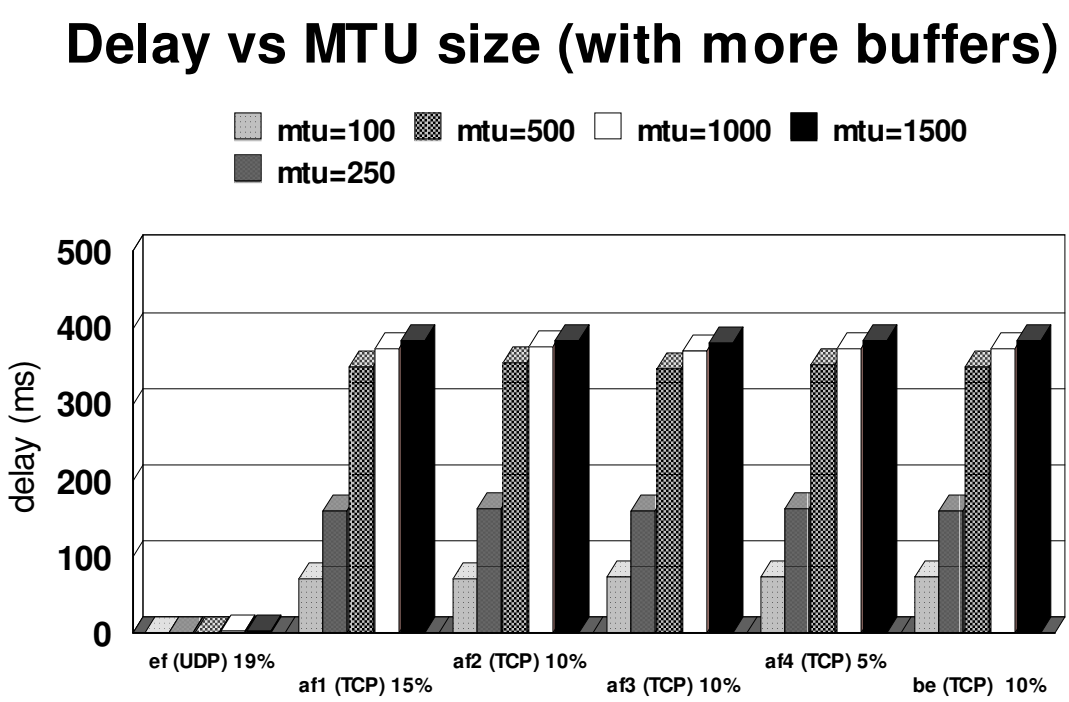

Figure 4

	MTU=100	MTU=250	MTU=500	MTU=1000	MTU=1500
EF	1.7	1	1	3	5.2
AF1	73.1	162.1	350.3	372.7	383.6
AF2	73.2	163.1	353.8	374.6	384.2
AF3	75.4	160.4	347.8	370.8	379.8
AF4	74.1	162.8	350.8	372.9	383.2
BE	73.3	160.9	348.5	372.5	384.3

Table 4. Delay vs. MTU With More Buffers (Figure 4's Data)

In this experiment, AF delay is directly related to the MTU one would expect. Work is ongoing to investigate this possible anomaly.

3.2 Conclusions

From the experimental result, we conclude that lightweight software techniques can be employed to implement differentiated services on low-cost edge routers. The measurements performed on our implementation (see [INFOCOM99] for details) showed that support for basic QoS guarantees can be achieved on edge devices with minimal impact on overall performance. In addition, the buffer management approach of [SIGCOM98] was indeed capable of providing reasonably accurate rate guarantees and a fair distribution of excess resources. We also demonstrated that a simple design based on 2 queues and a rudimentary WFQ scheduler can provide adequate delay differentiation to meet the requirements of most real-time applications.

Although we believe that eventually end-systems will properly mark their traffic, we think that in the interim low cost edge routers will play a role in the classification and conditioning of traffic. Also in network configurations where the end-system marking cannot be trusted, the edge-devices will be required, at a minimum, to provide marking and policing functionality. We are experimenting with three-color marking and further study is underway to test the effectiveness of the color-marking schemes.

4. Acknowledgements

The authors would like to acknowledge Roch Guerin and Ping Pan for their important contributions to this implementation.

5. References

[FREEBSD] Many Authors. See http://www.freebsd.org

[INFOCOM99] R. Guerin, L. Li, S. Nadas, P. Pan, and V. Peris, ``The Cost of QoS Support in Edge Devices - An experimental Study,'' in Proceedings of INFOCOMM '99, New York, NY, March 1999

[LDAP] Openldap Foundation. See http://www.openldap.org

[MGEN] B. Adamson, ``The Naval Research Laboratory (NRL) ''multi-generator'' (MGEN) toolset, ver. 3.0,'' Code is available from ftp://manimac.itd.nrl.navy.mil/Pub/MGEN/dist, 1998.

[NETPERF] R. Jones, (Netperf toolset, ver. 3.0,'' Code is available from http://www.netperf.org

[NTP] D. L. Mills, ``Network Time Protocol (version 3): Specification, implementation, and analysis,'' Request For Comments (Draft Standard) RFC 1305, Internet Engineering Task Force, March 1992.

[SCFQ] S. J. Golestani, ``Network delay analysis of a class of fair queueing algorithms,'' IEEE J. Select. Areas in Commun., vol. 13, no. 6, pp. 1057--1070, August 1995.

[SIGCOM98] R. Guerin, S. Kamat, V. Peris, and R. Rajan, ``Scalable QoS provision through buffer management,'' in Proceedings of SIGCOMM, Vancouver, British Columbia, CANADA, August 1998

[2212] IBM. See http://www.networking.ibm.com/2212/2212prof.html

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

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

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

6. Author's Addresses

Yonas Araya
Yonas@us.ibm.com
IBM NHD Laboratory
PO Box 12195
RTP, NC 27709

Vinod Peris
vperis@growthnetworks.com
Growth Networks
500 Ellis Street
Mountain View, CA 94043

Barry Beggs
beggs@us.ibm.com
IBM NSD Laboratory
PO Box 12195
RTP, NC 27709

Govindaraj Sampathkumar
gsampath@us.ibm.com
IBM NHD Laboratory
PO Box 12195
RTP, NC 27709

Stephen Nadas
nadas@raleigh.ibm.com
IBM NHD Laboratory
PO Box 12195
RTP, NC 27709

Gaurang Shah
gaurang@us.ibm.com
IBM NHD Laboratory
PO Box 12195
RTP, NC 27709

Liang Li
lli@us.ibm.com
IBM NHD Laboratory
PO Box 12195
RTP, NC 27709