A stateless approach for improving TCP performance using Diffserv

Eitan Altman[a]

[a]INRIA, BP93, 2004 Route des Lucioles, 06902 Sophia Antipolis Cedex, France

In TCP connections, the loss of some segments has more impact than the loss of others on the performance of the connection. These segments are (i) the connection establishment segments, (ii) the segments sent when the connection has a small window, and (iii) the segments sent after a timeout or a fast retransmit. We call these "vulnerable" segments, or packets. In the two recent papers [9,10], the authors show that by marking these segments with a higher priority and implementing the priority using a diffserv architecture, the performance of the TCP connection considerably improves. The marking proposed in these references requires, however, that network layer elements be aware of transport layer information, i.e. of the state of the TCP connection. The goal of our paper is to show that one can achieve prioritization of sensitive segments without any use of transport layer information, thus simplifying the implementation of diffserv marking of TCP packets.

1. Introduction

Recently, a new window based marking mechanism has been proposed in [9,10] in order to improve the performance of TCP connections using Assured Forwarding service of diffserv [2]. A three level marking has been proposed; the highest priority marking is given to segments sent while the congestion window size is below some threshold $HIGH_{thresh}$ as well as to the SYN packets (that starts a TCP connection) and to fast retransmit of fast recovery packets. If segments correspond to congestion window in-between the thresholds $HIGH_{thresh}$ and MED_{thresh} then they are marked in a probabilistic way with either the high or the medium priority, and if they correspond to a congestion window of more than MED_{thresh} then the priority is chosen probabilistically among all three levels[1]. Once in the network, differentiation between priority levels is obtained using several virtual queues with three multi RED (Random Early Detection) buffer management, with priority-dependent parameters. This prioritization is called in [10] "TCP-state based approach".

In order to implement this approach, the policers at the edge routers need to use the information of the TCP state, which is unfortunately not contained in the TCP packet. Thus in order to be implemented, either the structure of TCP packets have to change, or the extra TCP state has to be encapsulated within the data part of the TCP packet, or some additional signaling mechanism has to be designed. It is thus quite desirable to explore alternative prioritization approaches that may lead to performance improvements

[1]The notation $HIGH_{thresh}$ and MED_{thresh} are taken from [10].

of TCP connection and yet would not require TCP state information. In particular, we wish to check whether standard policers are able to mark most of the required TCP segments mentioned above without TCP-state information.

To that end we make the observation that the first packets of a TCP connection, which are especially sensitive to losses (their losses is likely to produce a TimeOut) and in particular the SYN packets, are all sent while the TCP connection has a low congestion window, and therefore also a low throughput. We can thus take advantage of policers that mark with high priority packets transmitted at a low rate. This is also true for the first packets transmitted after a silence period or after a timeout; in both cases a slow start is called upon and the initial transmission rate is low. Our first objective is thus to check the improvement of TCP connections using rate-based policers.

We explore in this paper the above approach for improving TCP performance through a simulation study using ns2 [1] (we used the version 2.1b9a). We study first the effect of protecting vulnerable packets on HTTP connections (we consider the HTTP/1.1 version in which a source does not open simultaneously more than one TCP connection; we assume that a new connection starts after some silence period that follows the end of the previous connection) and then the effect on ftp type traffic (in which a new ftp connection can start before the previous one ends. This will allow us to test cases in which each source has more than one simultaneously ongoing TCP connections). We show that for HTTP type traffic, our approach increases considerably the system's throughput. This is due to the fact that a transfer starts after some thinking time, which begins when the previous transfer ends, and decreasing losses allows to start new transfers earlier. In ftp file transfers, on the other hand, the arrival time of a file is independent of the time at which previous files end transmission, so as long as the system is stable, the average throughput does not depend on whether vulnerable packets are prioritized. However we show that the transfer times of files considerably decreases for ftp transfers with our marking approach.

The structure of the paper is as follows. We begin by describing in the next section the simulation scenario and the traffic models. We then present and analyze our simulation results in Section 3 for HTTP type traffic and in Section 4 for ftp type traffic. We end the paper with a concluding Section in which we present some guidelines for the range of parameters for which our marking approach can be expected to improve the performance.

2. The simulated scenario

2.1. Preliminaries on the service differentiation

Two priority levels are defined: the higher "In packets" or "green packets" and the lower "Out packets" or "red packets". We focus on the simplest policer available in ns: the time-sliding window [6], called TSW2CM in ns [11]. A CIR (committed information rate) is defined for each edge router. As long as the connection's rate is below CIR, all packets are marked as high priority. When the rate exceeds CIR, packets are marked probabilistically such that at the average, the rate of packets marked with high priority corresponds to the CIR. The transmitted rate is computed as the rate averaged over a period called the "TSW window". In our simulations its duration is 20 msec.

In our experimentations we vary the CIR level at the source edge nodes and study its impact on performance.

2.2. The topology

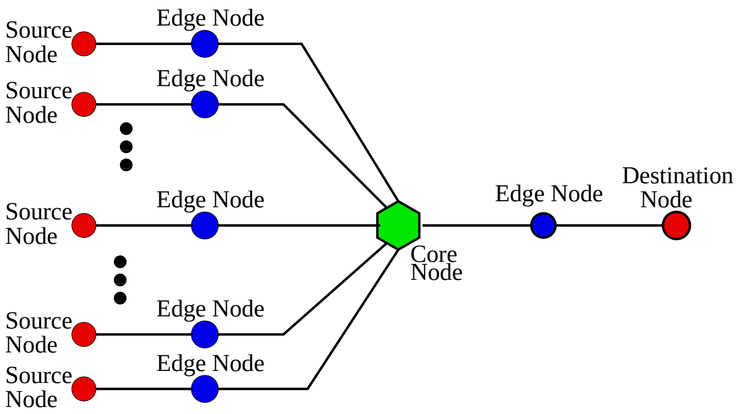

Figure 1. Network topology

We consider the simple network topology with a single bottleneck, depicted in Fig. 1.[2] Each source node is connected to a corresponding edge node where the traffic is marked according to parameters that will be specified. The edge routers are connected to a bottleneck core router, and then through another edge router, to a destination node. There are 20 source nodes, and each one of them generates TCP connections. We experiment with a high-speed local area type network (short propagation delays) with completely symmetric links:

(i) Links between the edge nodes and the corresponding source nodes have delays of 10 μsec and 6 Mbps bandwidth.

(ii) Links between the edge node and the corresponding destination node has 10 μsec delay and 10 Mbps bandwidth.

(iii) Links between the core node and edge nodes that are attached to the sources have 0.1 msec delay and 6 Mbps bandwidth.

(iv) The link between the core node and edge node attached to the destination has 1 msec delay and 10 Mbps bandwidth.

2.3. The traffic model

A transfered file has a Pareto distribution with shape parameter 1.25 and an average size of 10 kbytes (see [4,13] for similar parameters). Recall that for Pareto distribution,

[2]The choice of a single bottleneck network is motivated by growing evidence that performance measures (such as throughputs, delays and loss probabilities) are determined essentially by a bottleneck node. Indeed, both theoretical [5] as well as experimental [3] evidence are available. These references show that indeed when a connection crosses several routers, all routers other than the slowest one can typically be modeled by adding a fixed component to the propagation delay.

$E[size] = \beta k/(\beta - 1)$ where β is the shape parameter, and that $Pr(size > s) = (k/s)^{\beta}$. We consider two scenarios for traffic originating from a node:

(i) HTTP type traffic: The time between the end of transmission of a file till the beginning of the next transmission is exponentially distributed with a mean of 0.1 sec. This is called a "thinking time". Thus from each source node there can be only one file transmitted at the same time (at most one active session). The total number of active sessions is bounded by the number of source nodes, see e.g. Fig. 2. (We consider here only HTTP versions that open a single TCP connection at a time; this is the case with the HTTP/1.1 version.)

(ii) ftp type traffic: Files to be transmitted arrive at each source node according to a Poisson process with an average rate of 5 files per second. Several sessions from the same source node can be active simultaneously.

2.4. Queueing management parameters

Queue can build up only at the bottleneck router, i.e. at the link between the core node and the edge node that connects to the destination. We chose its size to be of 100 packets. Thus the queue management parameters at other nodes did not have an influence on the results. In the bottleneck queue at the core node, a multi-RED queue management is used with a RIO-D version[3]; we chose the same parameters for both priority levels (more details will be given below). Our aim in choosing the parameters was not to obtain necessarily an optimal performance but rather to create conditions that allow us to study the effect of diffserv on diminishing the loss probabilities of vulnerable segments, and the impact of this action on TCP performance (delay, throughput). For that reason, we chose the same parameters for the two priority levels (this will be explained below).

For each color of packets (red, green), the averaged queue sized is monitored (using the standard exponential averaging with weighting parameter $w_q = 0.01$, see e.g. [7]). Packets of a given color start to be dropped when the averaged number of queued packets of this color exceeds min_w; we chose $min_w = 15$; this drop probability increases linearly with the averaged queue size until it reaches the value $max_w = 45$, where the drop probability is taken to be $max_p = 0.5$. When this value is exceeded, the drop probability is 1.

Note that often the differentiation between the priorities is done using different sets of parameters: drops are performed at a larger queue size for green packets (e.g. [8,12]). We preferred not to use this approach, since with rejection at a larger window size we also get larger delays, which in some experimented parameters resulted in a lower throughput. By giving the same parameters to both priorities, we can learn about the direct effect of protecting vulnerable packets on the TCP performance. The differentiation is then done by using the RIO-D approach, in which the rejection probability of each type of color depends on the average number of packets of that type. Thus to have green packets dropped less than red ones, we simply choose their throughput to be lower (and consequently this will

[3]RIO stands for RED with In/Out; it has been proposed by Clark and Fang as an extension of RED to the case where packets are differentiated to In and to Out packets according to their marking; the "D" in RIO-D stands for the "decoupled" version in which the rejection probability of each type of packets depends on the average number of packets of that type. This is in contrast with the "coupled" version RIO-C where rejection probabilities depend on the average number of total queued packets of all types

imply that the corresponding average queue size is smaller too); this is done by the proper choice of the CIR value which determines the fraction of packets that will be marked green.

3. Simulation of HTTP type traffic

Simulations lasted 20 secs. All simulations use the same random sequence of thinking times as well as that of the file sizes.

Losses: We first analyze the influence of the CIR marking rate on the loss probabilities of two types of vulnerable packets: (i) the SYN packets (that start the connection), (ii) the first data packet in a connection. Recall that we perform a "blind" marking, we do not actually use the transport layer information of which is the packet type or its sequence number of the packet. Our only way to "protect" vulnerable packets is by a proper choice of the CIR parameter. The number of losses of SYN packets and of first packets in a session are given in Table 1.

Note that if we mark at extremely low rate or if in the contrary, we mark all packets, then no differentiation between vulnerable packets and other packets is achieved, so we obtain the maximum number of losses of vulnerable packets. This is confirmed in Table 1. We see that the total number of losses of the vulnerable packets considered decreased by more than a factor of 3 for CIR in the wide range of 200 kbps till 1 Mbps. At the optimal value of 300 kbps the gain is of a factor of 11.

Throughput: Next we have a closer look at the influence of CIR on the total number of packets transmitted (including retransmissions), the total number of "good" packets that arrived at the destination (we do not count retransmissions: they are denoted TxPckts), the number of losses due to buffer overflow (denoted as ldrops) and losses due to early drops (denoted as edrops; those are due to the buffer management dropping). All this is given in Table 2. Here is what we learn from it:

(i) The decrease in losses of vulnerable packets is accompanied with an improvement of performance. In the region of CIR of 200 kbps to 1 Mbps we have an improvement of throughput of between 11% to 21% (i.e. in the number of "good" transmitted packets).

(ii) With the best performance, i.e. the one with CIR of 300 kbps, 27.9% of the packets have been marked green. In the entire region of CIR of 200 kbps to 1 Mbps, the fraction of marked green packets varied between 20% to 60%. Thus, a considerable improvement in reducing losses of vulnerable packets and in throughput can be obtained even if traffic parameters are not known exactly; the improvement in performance is quite robust to the choice of the CIR parameter.

(iii) The total amount of losses increased along with the increase in the throughput: from

Table 1
Protection of vulnerable packets as a function of CIR

CIR in bps	1	10	10^4	100^5	2×10^5	3×10^5	10^6	10^7
lost SYN packets	18	22	15	4	4	2	5	17
First packets lost	16	14	19	13	4	1	6	14

Table 2
Packets statistics as a function of CIR

CIR in bps	1	10	10^4	100^5	2×10^5	3×10^5	10^6	10^7
All TotPkts	19363	19064	20072	21259	21884	23553	22721	19367
All TxPkts	18999	18745	19740	20773	21379	23054	22407	19022
All ldrops	24	0	0	0	0	0	0	0
All edrops	340	319	332	486	505	499	314	345
green TotPkts	0	7	285	2418	4481	6575	13715	19367
green TxPkts	0	7	285	2418	4481	6575	13643	19022
green ldrops	0	0	0	0	0	0	0	0
green edrops	0	0	0	0	0	0	72	345
red TotPkts	19363	19057	19787	18841	17403	16978	9006	0
red TxPkts	18999	18738	19455	18355	16898	16479	8764	0
red ldrops	24	0	0	0	0	0	0	0
red edrops	340	319	332	486	505	499	242	0

a value of around 340 packets when there is no prioritization up to around 500 packets all over the region $CIR \in$ [100 kbps,1 Mbps].

The number of sessions: The total number of sessions as a function of time is given in Fig. 2 for the CIR of 300 kbps (the optimal) and for the worse one (10 Mbps). We do not

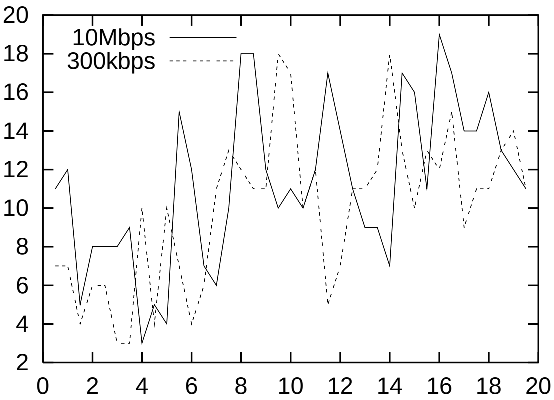

Figure 2. Evolution of total number of sessions, HTTP-type traffic for different CIR

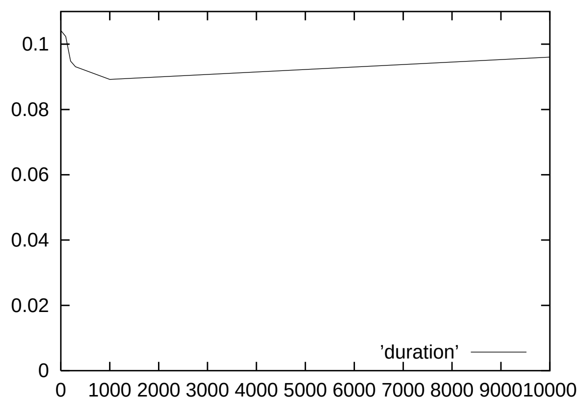

Figure 3. Average transfer duration as a function of CIR in kbps

see a domination of either one of the curves over the other. This will be in contrast with ftp type traffic, as we shall see later. The total number of sessions that completed with CIR of 300 kbps was 2037, where as it was 1769 under a CIR of 10 Mbps (in which all packets are marked green) and 1767 with CIR of 1bps (all packets marked red). We see again the large gain in throughput when implementing our marking approach, this time in terms of sessions per time.

Table 3
Protection of vulnerable packets as a function of CIR

CIR in bps	10^4	3×10^4	10^5	2×10^5	3×10^5	10^6	10^7
lost SYN packets	120	95	53	45	17	78	114
First packets lost	125	119	90	56	37	73	115
Total losses	1699	1612	1476	1286	1088	1290	1577

Session duration: The impact of CIR on the average transfer duration is given in Figure 3. We see that the duration varies very little. Larger durations are obtained when there is no prioritization (all packets are either marked green or all are marked red). The fact that the improvement in the duration of transfers is smaller (in comparison with the gain in the throughput) in the range $CIR \in$ [100 kbps,1 Mbps] can be explained by the fact that our marking scheme has increased considerably the global throughput of the system: more traffic is generated and transmitted per time unit. Therefore, the bottleneck capacity has to be shared among more connections, so that the duration of a session cannot decrease much.

4. Simulating ftp type connections

Simulations are 80 sec long. The longer simulation time is needed due to the much larger variability of the number of active connections as compared to HTTP traffic; we shall see that below. The rate of arrival of bits at the bottleneck is

$$\frac{20 \times 1.04 \times 10^4 \times 8}{0.20} = 8.32 \text{ Mbps}$$

This is obtained as follows: An average packet size is 1040 bytes of which 1000 are data and 40 bytes are an extra header. An average ftp file is assumed to contain 10^4 bytes of data, which means that its total average size (including the extra headers) is approximately $1.04 \times 10^4 \times 8$ bits. The result is obtained by multiplying by the number of source nodes and dividing by the average time between arrivals of files at a node.

Losses: We check the influence of the CIR marking rate on the loss probabilities of the SYN packets and of the first data packet in a connection, as before. We see in Table 3 that we manage to decrease the losses of SYN packets by a factor of seven, and the losses of the first data packet of a connection by a factor of around 3.4, both obtained at CIR of 300 kbps. It is less than the improvement factor obtained for the HTTP traffic. This can be explained by the fact that with ftp traffic, from each source node, there may be more than one simultaneously active sessions. If a new session begins at a node when another one at the same node is already active, then the first packets of the session might be marked red since the edge router cannot distinguish between them and the other packets of the other active connections from the same source. Nevertheless, our marking manages to decrease the amount of losses of vulnerable packets considerably.

We also see that our marking scheme decreases the *global number of losses*. This is exactly the opposite behavior than the one we observed in the case of HTTP traffic. The loss probabilities vary between 1.85%-3% with the lowest loss rate corresponding to 300 kbps.

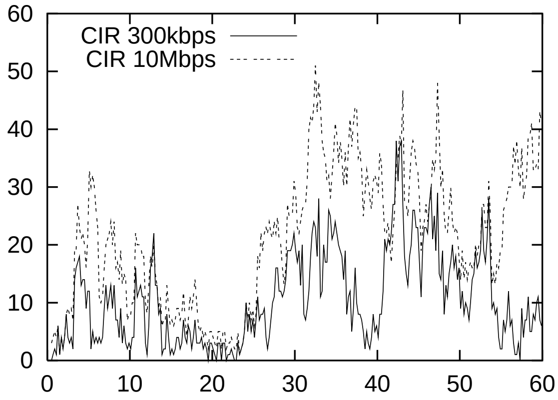

Figure 4. Evolution of total number of sessions, ftp type traffic

We do not provide a rigorous analysis of this phenomenon. Instead, we provide some possible intuitive explanations. When using our marking approach, this "smoothes" the traffic arriving at the queue, since losses occur at larger rates and not at the beginning of connections, when the rate is small. With more smooth traffic, the average queue can be expected to be smaller and therefore the average loss due to RED would be smaller. Moreover, the queue size can be expected to be more variable with the less smooth traffic (where there is no prioritization). Now, due to the convex form of the rejection probability of RED (as a function of the averaged queue length), and making use of Jensen formula, we may expect that a more smooth averaged traffic would give rise to less losses.

Throughput and Goodput: The number of data packets that were successfully transmitted during the simulations was quite independent on the CIR: it was in the average 58285, with a standard deviation of 395 packets (among the different simulations that corresponded to different CIR values). This is due to the fact that arrival rate of sessions does not depend on the CIR (unlike the case of HTTP traffic). In view of the low loss probabilities, the throughput (the amount of traffic transmitted including retransmissions) too, is almost constant as a function of CIR.

The number of sessions: The total number of sessions as a function of time is given in Fig. 4 for the CIR of 300 kbps (the optimal) and for the case of no prioritization (CIR of 10 Mbps). We see from the simulation result that our marking scheme with CIR of 300 kbps gives a better performance: less number of active sessions are present under this marking. This is related to the fact that the session duration in our marking is shorter, as we show later. Indeed, since the arrival rate of sessions is the same independently of CIR (unlike the case of HTTP-type traffic), the average number of session should be proportional to the average duration of a session (the proportionality factor being the arrival rate of sessions, due to Little's formula).

Session duration: In Table 4 we present the average duration of a session as a function the CIR. We see that in the range of 100 kbps till 300 kbps the average duration decreases by a factor between 1/3 (for 100 kbps) and 1/2 (for 300 kbps) with respect to the case of no prioritization. This improvement is much larger than the one we observed for the case of HTTP type traffic (as we saw in Figure 3). This is due, as already discussed, to the fact

Table 4
Average duration of a session as a function of CIR

CIR in bps	10^4	3×10^4	10^5	2×10^5	3×10^5	10^6	10^7
sess. duration	0.252616	0.231077	0.167948	0.149478	0.119509	0.203202	0.225075

that the arrival rate of sessions here does not depend on the marking, whereas in the case of HTTP the arrival rate increases in the above range.

5. Discussions and conclusions

There are a few limitation of our marking approach. The significant improvement that we have obtained would not be obtained in any scenario, and we propose a few guidelines, which we validated through further simulations, to describe its limitations.

(i) Vulnerable packets deteriorate considerably performance since they cause long timeouts. This is especially the case for the loss of a SYN that results in a timeout of 3 sec or of 6 sec. In high speed networks the duration of a file transfer is short (often the whole transfer is much shorter than timeout), so we can expect to gain much by eliminating these long timeouts. In low speed networks, this is no more the case so the gains in our approach become marginal.

(ii) In our simulation, an average file size is 10 kbytes, which is the averaged measured file size in the Internet [13]. This means that around 10% of the packets are SYN packets and further, another 10% of the packets are first in a transfer. Thus in the absence of our approach, around 20% of lost packets would correspond to these types of vulnerable packets, so eliminating these losses can result in a considerable improvement in performance. If we were to use our approach to much longer files, the fraction of vulnerable packets could be much smaller, so that the added value of our approach would be smaller.

We have seen, however, that for average size files and with short link delays, our approach increases performance of both ftp type traffic as well as HTTP type. ftp gets considerable shorter average transmission delay (this can be interpreted as better throughput as perceived by the user) and less losses. With HTTP traffic, our approach allows to increase considerably the system throughput with some further decrease in transmission duration. The improvements can be explained by the fact that our marking approach indeed managed to decrease considerably the losses of some vulnerable packets; through simulations we established this fact for the SYN packet as well as the first packet of each transfer. But this can be expected to hold also for other vulnerable packets, for examples to first transmitted packets after a timeout period.

We saw through simulations that our method was quite robust: large improvements were obtained for a whole range of values of CIR.

The protection of the vulnerable packets was done using a simple "blind" marking approach, based only on the input aggregate rates at the edge routers (rather than on the more costly knowledge of the TCP state of each packet).

Our point of view is somewhat different than that of [10], where each user had the choice of how to use a given amount of tokens in order to mark packets. Implicitly one understands in [10] that all the available budget of tokens should actually be used. This is a plausible scenario in a competitive context where the more tokens one uses, the better one's performance is, so long as the other users do not increase the amount of tokens they have. To further choose optimally the amount of tokens by each user in this competitive scenario, a game theoretic model should be formulated where both prices as well as quality of service should be considered. Our approach is different, in focusing on a social context where the amount of marking (through the choice of CIR) is determined globally and is the same for all users. We showed that optimal or even sub-optimal choices are profitable for both the users as individual (decreasing average delays of connections) as well as for the network in general (increasing global throughput in case of HTTP traffic).

REFERENCES

1. Network Simulator, ns version 2.1, available at http://www.isi.edu/nsnam/ns/
2. S. Blake et al., “An architecture for differentiated services”, RFC2475, Dec. 1998.
3. J.-C. Bolot, “End-to-end delay and loss behavior in the Internet,” *Proc. ACM Sigcomm '93*, pp. 289-298, San Francisco, CA, Sept. 1993.
4. T. Bonald and J. Roberts, “Performance modeling of elastic traffic in overload”, *ACM Sigmetrics*, Cambridge, MA, USA, June, 2001.
5. O. J. Boxma, “Sojourn times in cyclic queues - the influence of the slowest server”, *Computer Performance and Reliability*, G. Iazeolla, P. J. Courtois and O. J. Boxma (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 13-24, 1988.
6. D. Clark and W. Fang, “Explicit allocation of best-effort packet delivery”, *IEEE/ACM Transactions on Networking*, pp. 362-373, Aug. 1998.
7. S. Floyd and V. Jacobson, “Random Early Detection gateways for Congestion Avoidance” *IEEE/ACM Transactions on Networking*, 1(4):25–39, 1993.
8. N. Malouch and Z. Liu, ”Performance Analysis of TCP with RIO Routers”, *IEEE GLOBECOM*, Taipe, Taiwan, Nov. 2002.
9. M. Mellia, I. Stoica and H. Zhang, “TCP-aware packet marking in networks with DiffServ support”, Computer Networks 42, pp 81-100, 2003.
10. W. Noureddine and F. Tobagi, “Improving the performance of interactive TCP applications using service differentiation”, *IEEE Infocom*, N.Y., U.S.A., June 2002.
11. P. Pieda, J. Ethridge, M. Baines and F. Shallwani, “A network simulator differentiated services implementation”, Open IP, Nortel Networks, July 26, 2000.
12. S. Sahu, P. Nain, C. Diot, V. Firoiu and D. F. Towsley”, ”On achievable service differentiation with token bucket marking for TCP”, Proc. ACM SIGMETRICS'00, Santa Clara, CA, June 2000.
13. B. Sikdar, S. Kalyanaraman and K. S. Vastola, “An Integrated Model for the Latency and Steady-State Throughput of TCP Connections”, *Performance Evaluation*, v.46, no.2-3, pp.139-154, September 2001.