Switched Multi-Hop FCFS Networks- the Influence of Traffic Shapers on Soft Real-Time Performance

Master Thesis in Embedded and Intelligent Systems

Shashank Sharma

&

Master Thesis in Information Technology

Syed Hasnain Raza Tirmazi

Supervisor: Magnus Jonsson & Mattias Weckstén

School of Information Science, Computer and Electrical Engineering
Halmstad University
Preface

This thesis is a part of the Master programme in Embedded and Intelligent Systems, and Information Technology which is carried out in Halmstad University Sweden for the spring semester 2010.

We are thankful to Halmstad University for providing us with a good working environment and resources for completing our thesis project. Our supervisors, Magnus Jonson and Mattias Weckstén, guided us in proper way to complete this thesis work which would have been impossible without their help, moral support and guidance; and that was very valuable in different stages of the work.

Also, we would like to thank our family and friends for their support and strength they provided us during the programme.

Name: Shashank Sharma
Name: Syed Hasnain Raza Tirmazi
Halmstad University, June 2010
Switched Multi-Hop FCFS Networks- the Influence of Traffic Shapers on Real-Time Performance
Introduction

Abstract

In the past 10 years, the bandwidths and processing capabilities of the networks have increased dramatically. The number of real-time applications using these networks has also increased. The large number of real-time packets might, in a switched multi-hop network, lead to unpredictable traffic patterns. This is not a problem when the traffic intensity is low, but if the same network is used by a large number of users simultaneously, the overall performance of the network degrades. In fact, unpredictable delays in the delivery of the message can adversely affect the execution of the tasks dependent on these messages, even if we take into account the soft real-time performance.

In this paper, we investigate the effect of traffic shapers on soft real-time performance. We will consider a switched multi-hop network with FCFS queues. We will implement two versions of the network simulator. One version will be without traffic shaper and the other version will use a traffic shaper. By comparing the results (for average delay, deadline miss ratio etc.) from both the versions, we will try to conclude if it is really beneficial to use traffic shapers for soft real-time performance. Leaky bucket and token bucket algorithms are the most popular ones for traffic shaper implementation. We will consider leaky bucket algorithm for our analysis. We analyse different versions of the leaky bucket and present the trade-off’s involved.

Keywords:
Traffic Shaper, Leaky Bucket, Tree Topology, FCFS queuing, Soft Real-time, Multihop, Jitter
# Contents

PREFACE ................................................................................................................................. 3  
ABSTRACT ............................................................................................................................. 5  
CONTENTS ............................................................................................................................ 6  
1 INTRODUCTION ...................................................................................................................... 7  
  1.1 MOTIVATION .................................................................................................................. 8  
2 NETWORK MODEL ................................................................................................................. 10  
  2.1 DISCUSSION ON TOPOLOGY USED ............................................................................. 10  
  2.2 TYPES OF PHYSICAL TOPOLOGY ............................................................................. 10  
  2.3 TREE TOPOLOGY FEATURES – CONCLUSION .......................................................... 14  
3 TRAFFIC SHAPERS ................................................................................................................ 15  
  3.1 OVERVIEW OF LEAKY BUCKET ............................................................................. 16  
  3.2 FUNCTIONALITY OF LEAKY BUCKET ..................................................................... 16  
4 ASSUMPTIONS AND APPROACH .......................................................................................... 19  
5 RELATED WORK ..................................................................................................................... 21  
6 DISCUSSIONS AND SIMULATION RESULTS ........................................................................ 23  
  6.1 SIMULATION ANALYSIS BY IMPLEMENTING TRAFFIC SHAPER ON PER FLOW BASIS WITH RELEASE TIME FOR PACKETS (TRAFFIC SHAPER USED BY US) ........................................................................................................... 25  
  6.2 NETWORK WITH TRAFFIC SHAPER IN ALL NODES VS NETWORK WITHOUT TRAFFIC SHAPER .......................................................... 26  
  6.3 ANALYSIS OF TRAFFIC SHAPER WITH DIFFERENT RATES ........................................ 29  
  6.4 PERFORMANCE OF TRAFFIC SHAPER ONLY AT THE SOURCE NODES VS TRAFFIC SHAPER IN ALL SWITCHES ........................................................................................................... 31  
  6.5 ANALYSIS OF TRAFFIC SHAPER WITH PER PORT, PER SWITCH AT SOURCE NODES AND TRAFFIC SHAPER AT ALTERNATE DEPTH ........................................................................................................... 33  
  6.6 ANALYSIS BY IMPLEMENTING TRAFFIC SHAPER ON PER FLOW BASIS .................. 33  
  6.7 FINAL DISCUSSION .......................................................................................................... 33  
7 CONCLUSION .......................................................................................................................... 34  
8 REFERENCES .......................................................................................................................... 35  
APPENDIX .................................................................................................................................. 36
Introduction

1 Introduction

Today, real-time systems have become a part and parcel of our daily life. Everything from audio to video, and from rockets to satellites is classed as real-time application. In some cases, however, the real-time communication is of utmost importance, as even a $1 ms$ delay can cause a catastrophe. For example, a rocket might whirlwind back to the earth’s crust! In some other cases, the results might not be so devastating. When missing a deadline does not lead to a catastrophic effect, the system is a soft real-time system, whereas the deadline critical systems are called “hard” real-time systems. We focus soft real-time systems in our paper.

Although soft real-time systems do not have hard real-time requirements, the packet delays/drops will certainly affect the quality of service desired by the end user. Our idea is that the regulated traffic should yield better results than turbulent traffic. To exploit this idea, we would see the effect of traffic shapers on soft real-time systems. In multihop networks, the probability of traffic transforming into bursty traffic is quite high. We keep the network architecture simple with binary tree topology in our evaluations. Also, since most of the switches support FCFS scheduling, we also consider FCFS packet switches.

The whole thesis work may be subdivided into four main subtasks.

1. Investigation for the type of traffic shaper to be used.
2. Implementing a network simulator, first without a traffic shaper implementation
3. Implementing the network simulator with a traffic shaper.
4. Comparison of results of steps 2 and 3 above. Final report and conclusions.

The rest of the paper is organized as follows: in section 1.1, we explain the problem in slightly detailed form. This section gives a clear idea about our motivation. In section 2, we describe the various topologies, network architecture and conclude with the type of topology best suited for our analysis. In section 3, we present the functioning and algorithm of the leaky bucket traffic shaper. We also reason, in brief, about our choice of traffic shaper (leaky bucket instead of token bucket). In section 4, we present the assumptions and approach for our analysis. After this, we also present some details on related work and try to compare/criticize our work with the previous works in section 5, while the simulation results are presented in section 6. Finally, in section 7, we present the conclusion and possibility for future work.
1.1 Motivation

At the time of channel establishment, each intermediate node checks whether it will be able to accept packets at the rate declared by the sender or not. If the users, at all times, obey the traffic agreement, then the traffic remains well organized but, in the real world, there may be several faulty users. They may send packets aggressively and exceed the rate declared by them at the time of channel establishment. This gives rise to non-conforming traffic and events like packet dropping become more frequent. Also, due to network load fluctuations, the jitter problem becomes worse. Jitter is the difference between the maximum and the minimum inter arrival times of the packets. Even if we can characterize the traffic at the entrance of the network, the network load fluctuations can cause traffic distortion inside the network. Due to this, we are going to use traffic shapers at each switch to keep the traffic conforming.

Figure 1 explains the problem [5]

Figure 1. Organized traffic transforming into bursty traffic

As we can see in figure 1, at the entrance to network we have a well organized traffic. The packets are sent at a fixed rate in the network. Let us suppose we have a network load
Introduction

fluctuation and this causes a delay of the first packet at the time of transmission from switch 1. Due to sudden load fluctuation, only the first packet is delayed and the other four packets pass as they are, i.e. without any delay. In a multihop scenario, this condition can be repeated several times. In figure 1, we see that the time spacing between the packets has reduced drastically and, in fact, the traffic becomes quite bursty in nature after switch 4. This makes the queue population and packet delay very undeterministic. As a potential solution to this problem, we propose to use traffic shapers at each switch throughout the network.
2 Network Model

We consider a multihop network. A multihop network can support various topologies, e.g. bus, star, ring, mesh, tree etc. We restrict our case to tree topology. All the network components, i.e. all nodes and switches, support FCFS queuing. The frames are processed on a first come first served basis. Traffic shapers are employed at each port in all the switches. We also assume unidirectional flow of packets from one switch to another switch and each physical link has a fixed bit rate.

2.1 Discussion on Topology used

Here, we try to motivate the topology (TREE topology) used by us. First, we compare the basic topologies in brief and then we conclude with the topology best suited for our case, which is tree topology.

The topology of a network is the configuration of cables, computer and other peripherals. Topology is of two types:

i. Physical topology
ii. Logical topology

2.2 Types of Physical Topology

The followings are the types of topologies used in the networks:

- Linear bus topology
- Star topology
- Tree topology
- Ring topology
- Mesh topology

Linear Bus Topology

In linear bus topology, as shown in figure 2, all nodes are connected with a linear cable. All the nodes use a common backbone to connect all devices. A device wanting to communicate with another device on the network sends a broadcast message onto the wire that all other devices see, but only the intended recipient actually accepts and processes the message.
Network Model

Figure 2. Linear Bus Topology

Advantages:

a) Easy to connect a computer with a linear bus.

b) It requires less cable length than a star topology.

Disadvantages:

a) In case of break down in main cable, the entire network will be shut down.

b) At both ends of the backbone cable, terminators are required.

c) In case of shutdown, it is very difficult to identify the problem.

d) It is not suitable for large buildings.

Star Topology

In star topology, as shown in figure 3, each node is connected directly to a central network. Data passes through the hub, switch or concentrator/router before continuing to its destination. These things control all the functions of the network. It acts as a repeater for the data flow.

Figure 3. Star Topology
Switched Multi-Hop FCFS Networks - the Influence of Traffic Shapers on Real-Time Performance

Advantages:

a) Easy to install.
b) There is no disruption in network when connecting or removing the devices.
c) Easy to detect faults and remove parts.

Disadvantages:

a) As compared to the linear topology, more cable length is required.
b) Attached nodes are disabled due to failure of hub or switch.
c) More expensive than linear bus topology.

Tree or Expanded Star Topology

All the characteristics of linear bus and star topologies combine in tree topology, as shown in figure 4. A group of star configured workstations connected to a linear bus backbone cable exist within it. We can expand the existing network easily to meet our needs. In its simplest form, only hub devices connect directly to the tree bus and each hub functions as the root of a tree of devices.

Figure 4. Tree or Expended Star Topology

Advantages:

a) Easy to connect a computer with a linear bus.
b) It requires less cable length than a star topology.
Network Model

Disadvantages:

a) Because it is dependent on the trunk, which is the main backbone of the network, if that trunk has to fail, then the entire network would fail.

b) Troubleshooting can be difficult in tree topology, as finding a faulty switch is troublesome in a big network.

Mesh Topology

In a mesh topology, there are several possible paths available from source to destination, as shown in figure 5. Mesh topology is being used in WANs (Wide Area Networks). If a device in a mesh network is connected with every other device, then it is called “full mesh”. Beside this partial mesh, networks also exist in which some devices connect only indirectly to others.

![Mesh Topology Diagram]

Figure 5. Mesh Topology

Advantages:

a) It is possible to transmit data from one node to many other nodes at the same time.

b) Without disruption to current users, network can easily be expanded.

Disadvantages:

a) In this kind of arrangement, where every node of the network is connected to every other node, many of the connections serve no major purpose. So many of the network connections leads to the redundancy.

b) As compared to other LAN topologies, more cables are required.
2.3 Tree Topology Features – Conclusion

Hierarchical pattern is followed by tree topology. Each level is connected to the next higher level in a symmetrical pattern. In the hierarchy, each level follows a certain pattern in connecting the nodes. The top-most level in tree might have only one node or two nodes and the following level in the hierarchy might have a few more nodes which work on the point to point connectivity, and the third level also has asymmetrical node-to-node pattern, and each of these levels are connected to the root level in the hierarchy. Imagining a tree that has branches extending out in various directions, and all these branches need the roots and the tree trunk to survive. Similarly, a tree structured network is very similar to this and that is why it is called the “tree” topology.

Features:

- In the tree network, there will be at least three levels of hierarchy in the tree network topology and they all work based on the root node.
- The tree topology functions by taking into account the total number of nodes present in the network. It does not matter how many nodes are there on each level. Nodes can be added to any level of the hierarchy and there are no limitations for the number of nodes.
- There might be bottlenecks in the higher levels in the hierarchy.

Conclusion:

If a packet is transmitted by the root node, it is received by all the nodes at the same time. This increases the efficiency of the overall network. The tree network topology can be extended easily to function and there are no limitations as to how much it can be extended. We can easily add additional root nodes and they can be interconnected within one single network. So, with respect to the above mentioned features, and also taking into account our network assumptions, we choose tree topology for our network model.
Traffic Shapers

3 Traffic Shapers

Broadly, real-time traffic can be classified into two types:

- Hard real-time traffic
- Soft real-time traffic

Hard real-time traffic does not expect the packets to miss their deadlines. In other words, it has a zero tolerance for loss of packets. On the other hand, soft real-time is not so stringent and can allow for occasional packet losses and deadline misses. For example, audio and video sources generate soft real-time traffic. While video applications can tolerate up to 1% packet loss [1], audio is said to tolerate between 5 to 50% packet loss, depending on quality desired [2] [3]. A packet loss normally may occur due to:

- Buffer overflow
- Late packet arrival

Although soft real-time can tolerate some packet loss, it still demands a certain QoS. The QoS requirement, in turn, depends on the quality desired by the end users.

Typical QoS requirements are:

- Max tolerable packet loss
- Deadline miss ratio
- Average delay

These are simulated and results presented later in the paper.

We will use traffic shaper (leaky bucket) in the network and study its effect on soft real-time performance. Although token bucket is also a popular algorithm for traffic shaping, it is most suitable when we wish to analyze bursty traffic. In fact, ‘Venkat Antharam’ and ‘Takis Konstantopoulos’ [4] have proved that a chain of token buckets is the optimal solution for regulating source traffic in a communication network.

The traffic shaper (both leaky and token) works by delaying the packets until the time such that it conforms to the predetermined traffic profile. For our purpose, we use leaky bucket. Leaky bucket imposes a hard limit on the rate at which packets are sent. Sometimes, the network may experience idle periods, but will not experience bursts when using the leaky bucket algorithm. This makes the packet flow predictable and controlled.
3.1 Overview of Leaky Bucket

The main guiding principle of leaky bucket algorithm is to maintain a constant period between the incoming (somewhat turbulent) packets. The packets may come in a burst (due to network load fluctuations) or only slight variation from the regulated traffic – the leaky bucket traffic shaper results in a smooth and regulated outflow of packets at a fixed constant rate. This fixed constant rate, in turn, depends on the traffic contract between the two end nodes. The leaky bucket mechanism is dynamic in the sense that the outflow rate of the packets can be changed in case of changes in the requirements of the end node.

3.2 Functionality of Leaky Bucket

Figure 6. Leaky Bucket Queuing Model

Figure 6 explains the functionality of leaky bucket. The main buffer overflow results in packet drop/discard. The following algorithm describes the functioning of leaky bucket. The algorithm could also be understood with the help of flowchart in figure 7.

- Let $g$ be the rate at which the authorization grants arrive at the grant buffer. This rate is set equal to the desired outflow rate of the packets to keep the traffic regulated.
- The packets from the source node arrive in the main buffer.
- The HOL (head of line) packet checks for the availability of grant in the grant buffer. If no grant is available, then it will have to wait for the grant.
- Each packet in the main buffer gets a grant every $1/g$ seconds.
- Every $1/g$ seconds, maximum $i$ packets can be assigned grants, depending upon the availability of the grants in the grant buffer. Even if $i$ packets can get grants at once, they are still transmitted at a rate of $g$ only, and not as a burst. Here, $i \leq w$ ($w$=size of the grant buffer).
- The number of packets that can be stored in the main buffer depends on the size of the main buffer.
Traffic Shapers

![Leaky Bucket Flow Chart](image)

Figure 7. Leaky Bucket Flow Chart

$t_{exp}$  Expected time of packet arrival  
$t_{arr}$  Actual time of packet arrival  
g  Packet transmitted rate (hole size of bucket)  

The above flowchart is explained with the help of following example:
Table 1 shows an example of behaviour of leaky bucket. In table 1, we have assumed the rate of leaky bucket as 10. From table 1 we can see that how the expected time of next packet is calculated. Whenever a packet arrives, the expected time of next packet is calculated as a sum of arrival time of current packet and time interval which has to be maintained between two packets. If a packet arrives earlier than the expected time, then it should be put in a buffer. The packets are sent out of the buffer only after fixed intervals of time.

<table>
<thead>
<tr>
<th>Packet</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_{arr} (ms)</td>
<td>0</td>
<td>10</td>
<td>18</td>
<td>29</td>
<td>42</td>
<td>50</td>
</tr>
<tr>
<td>t_{exp} (ms)</td>
<td>0</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>52</td>
</tr>
<tr>
<td>Result, t_{exp} (ms)</td>
<td>(0+10)</td>
<td>(10+10)</td>
<td>(20+10)</td>
<td>(30+10)</td>
<td>(42+10)</td>
<td>(52+10)</td>
</tr>
</tbody>
</table>
4 Assumptions and Approach

We assume a multihop network with node nodes and SW switches connected by a packet switched network. In this network, we can enable the structuring of different network topologies and different types of configurations but, in our case, we are using tree topology with the computation nodes in the leaves and switch nodes in the rest of the tree. This is a balanced binary tree with $n=2^N$ leaf nodes and, thus, $2^{N+1} - 1$ nodes. We are using the FCFS queuing technique for all the nodes and switches in the network. In this technique, the frames are taken from the queue in the order of arrival. Our network employs deadlock free routing. We also assume that all the resource scheduling is performed off-line and the only type of real-time control is of the release time for the packets from the nodes. The switches are output buffered and they support “store and forward” and handle packets in FCFS order. Propagation, packetization and switching delays are known and fixed. To be able to handle packets with real-time constraints, it is necessary that the nodes in the system are clock synchronized. The traffic shapers are employed at each port in all the switches without any alteration in network protocols. We are using leaky bucket traffic shaper to regulate our traffic. Message generation rate is fixed.

The switches are output buffered. They handle the flows on a per port basis. The traffic shaper is implemented for each real-time channel. All traffic flows are characterized by a source node, $S_i$, a destination node, $D_i$, traffic generation rate or period $Period_i$ (which may or may not be different per flow) and deadline $Deadline_i$. The message length varies between 1-6 for all the channels. The following expression completely defines each real-time channel:

$$RCh_i = \{S_i, D_i, Period_i, Deadline_i\}.$$  

$RCh_i$ denotes the real-time channel with $1 \leq i \leq M_{max}$, where $M_{max}$ is the number of real-time channels.

We have made the comparison for these parameters:

i. Average delay

ii. Deadline miss ratio

iii. Number of packets dropped in the network due to buffer overflow

iv. Total number of packets received
Switched Multi-Hop FCFS Networks- the Influence of Traffic Shapers on Real-Time Performance

While all the other statistics are saved only at the receiver end, the number of packets dropped is taken into account in the entire network at each switch. Each switch has three ports (as we have assumed binary tree topology), and each port maintains its own queue for the packets. In the traffic shaper implementation, each queue at the port is assumed to consist of several virtual queues, as shown in figure 8. The number of virtual queues is equal to the number of real-time channels in the network.

![Virtual queue representation](image)

**Figure 8. Virtual queue representation**

Packets are still stored and transmitted in the FCFS fashion. Of all the first packets, in all the virtual queues, the packet which is due for the release at the current time is transmitted.
Related Work

5 Related Work

When we talk about a network, then there are some problems like delays, jitter problems and some other QoS regarding issues are taken into account. These are minimal in point-to-point simple network but, when we go to the multihop network, then these problems are more critical and we have to overcome these issues. To avoid bursty traffic and different kinds of delays, we have to regulate our traffic and, for this purpose, there are different kinds of traffic shapers used in the network.

Our work is on switched multihop network with FCFS queuing technique and we are analyzing the influence of traffic shapers on real-time network. In this regard, some work is already being undertaken by some researchers and, in the following lines, we will relate our work with other researchers and try to investigate the influence of traffic shapers in a real-time network.

In [8], the author has developed two real-time analytical frameworks for networks with FCFS queuing in the switches. Two variants are proposed: one with FCFS queuing in the source nodes [10] [6], and one with EDF queuing in the source nodes [10]. The switched Ethernet is used, but the general approaches are not limited to switched Ethernet networks and can easily be modified to other similar packet-switched networks. It was found to be less complex in implementation than other scheduling techniques or traffic shaper implementation. Also, hybrid scheduling is discussed. It is suggested that, by using FCFS queuing in switches, we can avoid any modification in switches and also we can increase network utilization. The deadline sorted queues are suggested in this paper. The analysis is based on both fixed sized frames with same bit rate in switches and variable sized in frames with different bit rate in switches.

This work is a generic method for calculations of the delays in the form of EDF scheduling in a multi-hop switched network. The analysis takes into account situations that will not occur during runtime and is, hence, somewhat pessimistic.

We also use FCFS queuing as it is supported by most of the network components and no modification is needed in switches. However, in addition to this, we also include traffic shapers at each port of the switch.

In [9], the authors present a method for calculation of the packet delay in multi-hop switched networks when the packet offset is taken into account. In their scenario, the only traffic shaping is performed in the end nodes. A suggested solution is to consider the packet offsets
Switched Multi-Hop FCFS Networks- the Influence of Traffic Shapers on Real-Time Performance

which limit the existence of packets to small time windows, which reduce the pessimism because of the increased knowledge of where and when a packet will exist. The argument for this solution is that a designer of a real-time system might prefer stability and predictability over flexibility and performance.

In our analysis, we do not use any offset for packets and employ traffic shapers at all the switches. We assume that source nodes are traffic agreement compliant and we want to address the issue of burstiness in our evaluation.

In [11], the authors present the implementation of traffic shaping on switched Ethernet technology, to achieve reliable packet transmission with bounded transmission delay. In a network, every connected node can send the traffic whenever it wants; to avoid this situation, every node must shape its traffic accordingly. The measurements are taken to analyze the properties of switches and delays that can be achieved by using these switches. Continuing this work for hard real-time communication, in [12], they present ideas on offloading the traffic shaping process and the receive process into the firmware of a network interface card (NIC).

We consider the propagation delay of a packet as a constant and take into account only the queuing delay in switches. In our case, also the node can send packets at fixed intervals but traffic is treated/shaped in the switches to see the effect of traffic shapers on soft real-time performance.

Lucian Ioan and Graziela Niculescu [7] developed two mathematical models for leaky bucket performance measure. They analyzed ATM networks and developed one general mathematical model for the leaky bucket mechanism. They considered poisson arrival process for packets and grants. They developed another model for the on-off sources. They realized that buffer capacities directly influence the cell loss and delay. Large buffer capacities result in low packet loss but higher delay, and vice-versa, by using a higher grant rate. So they concluded that, to meet the QoS requirements, there has to be a trade-off between the buffer capacities and the grant rate. Our work can be seen as an extension of this work. We use the leaky bucket algorithm for soft real-time traffic with FCFS queuing in the switches.
6 Discussions and Simulation Results

The following are the parameters that we use and analyse in our simulations. Here, we will discuss these parameters in slight detail and motivate our choice of QoS parameters for our analysis.

- **Average delay**
  Average delay is one of the most common and useful parameters for analysing network behaviour. Some of the other parameters, like deadline miss ratio, could be approximately deduced from the average delay analysis. The average delay could also, to some extent, give an idea about the buffer requirements at switches in the network. If the average delay is high and increasing quite rapidly, then buffer requirements will be greater, as compared to the less rapid increase in the average delay. We plot total average delay, which is the average of average delays for the number of simulations.

- **Deadline miss ratio**
  Deadline miss ratio is one of the most important parameters for the analysis of real-time systems. While hard real-time systems have a very strict limitation on deadline miss ratio, soft real-time systems could tolerate variable amount of deadline miss ratio, depending upon the type and quality of service desired. In the analysis below, we will see the various trade-offs involved with high or low deadline miss ratios. We have presented the average miss ratio for number of simulation runs.

- **Total Packets Dropped**
  This is an important parameter, especially with respect to traffic shapers. The amount of packets dropped in the network due to buffer overflow gives a direct indication about the buffer requirements in the network and, hence, the cost involved. The higher the buffer requirement, the higher is the cost involved. With total packets dropped, we also present the total number of received packets.

Simulation Results

We did the experiments by passing various parameters. First we describe them in brief. $K_{\text{max}}$ is the number of times the simulation is run to get a better average. $M_{\text{max}}$ is the maximum number of real time channels in our experiment. Simulation length is the time, in $\mu$s, for which one simulation runs. Period and deadline are the period and deadline of the real time channels respectively. Queue length is the buffer capacity at each port for each switch. We
present our results for binary tree topology of depth 3. For all simulations, the x-axis denotes the number of real time channels. We set the above mentioned parameters to the following values. $K_{\text{max}} = 100$, $M_{\text{max}} = 40$, $\text{Simulation length} = 100000$, $\text{Period = Deadline} = 4000$. We have assumed queue length (buffer capacity) of 8. We have assumed packet duration of 125 $\mu$s which is slightly longer than one maximum sized Ethernet frame at 100Mbit/s.
Discussions and Simulation Results

6.1 Simulation analysis by implementing traffic shaper on per flow basis with release time for packets (traffic shaper used by us)

In continuation of the above, we give release times to the packets, so each packet is released only when the current time is greater than, or equal to, the release time of the corresponding packet. With each hop, the release time is updated to new value, based on the interval we wish to maintain between two packets of same channel. This update in time is independent of any other packet’s release time.

This will now be explained in a little more detail: if a packet belonging to any channel gets delayed by some time due to any reason, then the next packet of the same channel will not be affected by this delay. This will still be released at its scheduled release time. This results in a deterministic delay of the packets and, also, the buffer requirements in the network become much more predictable and fewer as compared to the network without a traffic shaper. To optimize the costs involved, this could also be done at alternate depths. This will involve more packet drops but less deadline miss ratio (Result in appendix (b)).
6.2 Network with traffic shaper in all nodes vs network without traffic shaper

In this section we compare the performance of the network with and without traffic shaper implementation.

![Comparison of average delay for network with and without traffic shaper](image)

Figure 9. Average delay of the network with traffic shaper vs without traffic shaper

Figure 9 shows the comparison of average delay for network with and without traffic shaper. From figure 9, we notice that for network without a traffic shaper, the delay rises from 500 to 3500. This shows that the rise in average delay is almost 7 times.

For a network with a traffic shaper, we can see that up to 29 channels, the delay increases from 1,500 and 2,000: barely a doubling of the rise in delay. If we look at the overall figure 9, then we can see that the delay is confined between 1200 and 3700, signifying only 3 times rise as compared to 7 times rise without using traffic shaper.

A horizontal curve of average delay for a large number of logical channels denotes a deterministic delay. Hence, with the use of a traffic shaper, we could have deterministic traffic. This helps a lot in estimation of the buffer requirements at switches in the network and also at the receiver.
Discussions and Simulation Results

Figure 10 below shows the deadline miss ratio comparison.

![Figure 10. Deadline miss ratio of the network with and without traffic shaper](image)

By analysing figure 10, it seems that the traffic shaper line leads to more deadline misses. If we inspect this closely, we can see that the deadline miss ratio is almost 0.005 till 25 channels, and then increases. The deadline miss ratio is higher in leaky bucket implementations because the packets are intentionally delayed in the network. Here, we have used an interval of (6*125) between each packet. If we decrease this interval, the deadline miss ratio will also come down. This point becomes clearer when we analyse different rates of traffic shaper.
From figure 11(a), we can readily notice that the packets dropped in the network are significantly fewer when we use a traffic shaper. In fact, the packet drop is almost 0 for 17 channels when we use a traffic shaper but, without the traffic shaper, the packet drop starts increasing after 9 channels. We can also see the improvement for the total number of packets dropped over 40 channels, in figure 11(b).
6.3 Analysis of traffic shaper with different rates

In this section we analyse the trade-offs involved with different leaky bucket rates.

![Average delay with different rates of leaky bucket](image1)

Figure 12. Average delay with different rates of leaky bucket

![Deadline miss ratio with different rates of leaky bucket](image2)

Figure 13. Deadline miss ratio with different rates of leaky bucket
Switched Multi-Hop FCFS Networks- the Influence of Traffic Shapers on Real-Time Performance

By analysing figures 12 and 13, we can clearly see that, as we increase the rate of the traffic shaper, the delay becomes more and more deterministic. The delay is confined to a smaller interval at higher traffic shaper rates. Correspondingly, the deadline miss ratio increases with the increase in average delay.

![Figure 14. Packet drop with different rates of leaky bucket](image)

For the number of packets dropped curve in figure 14(a), we can clearly see the knee shifting to the right as we increase the rate of the traffic shaper. We can also notice a slight increase in throughput with increase in traffic shaper rate, in figure 14(b).
6.4 Performance of traffic shaper only at the source nodes vs traffic shaper in all switches

Characterizing traffic before entering into the network is quite popular commercially as traffic shaper is not used in all the switches. In this section, we compare its performance with traffic shaper implementation in all switches of the network.

Figure 15. Average delay comparison for traffic shaper at source nodes and all switches

Figure 16. Deadline miss ratio comparison for traffic shaper at source nodes and all switches
We see that in figure 15, when we use traffic shaper only at the source, the average delay increases rapidly with the increase in the number of channels, and this also has a direct effect on the deadline miss ratio, seen in figure 16. The average delay increases as packets are allowed to be sent only at fixed intervals from the source, i.e. some delay is introduced in the beginning and then the packets again get delayed in the networks in the single FCFS queues. These 2 delays combine together and make the overall average delay quite high and a high miss ratio. On the other hand, this has a very positive effect on the number of packets dropped due to buffer overflow as seen in figure 17(a). This is because the packets are sent out from the source nodes after fixed intervals and, after that, packets are continuously removed from the queues in the rest of the network. In fact, the higher the interval between two packets, the lower is the number of packets dropped. This type of traffic shaper, only at the source, clearly gives a high importance to packet drop while jeopardizing the deadline miss ratio. From figure 17(b) it is clear that the throughput is less for the case when we use traffic shaper only at the source nodes, as compared to the case for without traffic shaper and with traffic shaper in all switches. This has very low buffer requirements and can be very cost effective. On the other hand, our implementation of traffic shaper in the entire network gives a much better deadline miss ratio while still keeping the number of packets dropped quite low.
Discussions and Simulation Results

6.5 Analysis of traffic shaper with per port, per switch at source nodes and traffic shaper at alternate depth
In the previous section, we presented a discussion and results of per port per switch with a traffic shaper applied only at the source nodes. We experimented on it by implementing the traffic shaper at alternate depths also. This resulted in more average delay and, hence, more deadline miss ratio. This happens because we are just delaying each and every packet for a fixed interval. This approach may be able to remove burstiness, but it cannot make the delay deterministic. For this reason, we did not consider this traffic shaper for our final version. We could have improved its performance slightly by introducing delay only on the uplink ports and letting the downlink ports function without delay. This could have resulted in a better deadline miss ratio.

6.6 Analysis by implementing traffic shaper on per flow basis
This traffic shaper functions by maintaining a fixed interval between all the packets of the same channel. Physically, all the switches in the network have one queue but, logically, they have several virtual queues, one for each channel. In this implementation, each channel has knowledge about when it transmitted a packet. The next packet in the virtual queue is released only after a fixed interval after that previously transmitted packet. This arrangement could give somewhat deterministic delays, but it also has a potential disadvantage associated with it. If any packet of any channel gets delayed in the network (for any reason), then all the subsequent packets of that channel will also be delayed by at least the same amount. This means that, even if the subsequent packets were compliant with the traffic agreement, they will be delayed and might even miss their deadlines.

6.7 Final Discussion
Leaky bucket is mostly used in ATM networks and token bucket is mostly used in IP networks. Conventional leaky bucket puts a hard limit on the amount of data that could be sent in a given time unit. This makes leaky bucket vulnerable to a drawback. The end user (sender) cannot save bit-rate when they have nothing to send. Even if the packets are arriving after long periods of idleness, they will still be sent after being spread out in fixed time intervals. The token bucket traffic shaper is quite dynamic. It can support bursts after periods of idleness (for token bucket description please refer to appendix (a)). The main merit of leaky bucket is that the packet delays are deterministic.

The typical application of leaky bucket [14] and token bucket [13] is seen to be quite different.
7 Conclusion

A simple leaky bucket algorithm applied on per port per switch helps greatly in reducing the packet drops, but fares poorly for deadline miss ratio. In the same manner, keeping a constant interval between the packets of the same flow could lead to very predictable delays but poor deadline miss ratio. This is because, if one packet gets delayed for any reason, all the packets of the same flow also become delayed by at least the same amount, thereby deteriorating the deadline miss ratio. A traffic shaper only at the source is also a good option if only the packet drop is taken into account. With our version of the traffic shaper implemented in all switches, we showed that we could greatly improve the deadline miss ratio while still keeping the number of packets dropped and, hence, the buffer requirements, quite low. The rate of leaky bucket plays an important role. Depending on which QoS we aim for, we could have different rates of traffic shaper. For example, if the interval between the packets is kept quite high, the delay is much more deterministic, but takes its toll on deadline miss ratio. On the other hand, if the interval between the packets is kept quite low, the delay is less deterministic but the deadline miss ratio improves. There is, therefore, a trade-off involved with the traffic shaper rate. The selection of traffic shaper rate depends on what kind of QoS we desire. Implementing a traffic shaper only at alternate depths is also a commercially viable solution.
8 References


Appendix

A. Token bucket

Figure 18 shows the working of token bucket. At the heart of this scheme is token generator. Tokens are generated at a fixed rate and stored in the token bucket. Sufficient number of token is attached to each incoming packet. If the tokens are not available, the packets have to wait until sufficient numbers of tokens are available. If large number of tokens have been generated when no packets has arrived, then a burst of packets could be transmitted as soon as they arrive. Depending on the size of the token bucket, only a certain maximum number of tokens could be stored in the token bucket. The additional/extra tokens are discarded. This also means that this algorithm supports only a certain maximum burst size.

Figure 18. Token bucket
Appendix

B. Analysis of Traffic shaper applied at alternate depths:

Here, for this analysis, we had the following parameters: Depth = 7; Period = 5000; children=2; rate of leaky bucket = 6.

From the below graphs, we can see that implementing traffic shaper at alternate depths comes with its own trade-offs.

![Figure 19. Average delay comparison for traffic shaper at all switches and at alternate depths](image)

From figure 19 we see that the delay is still quite predictable and has almost the same rise in delay over 50 channels.
From figure 20 we can conclude that the traffic shaper at alternate depth scores higher for deadline miss ratio, since at alternate depth it will send packets without trying to maintain any interval between the packets of the same channel. So the packets are not held in the queues for long time and are sent as soon as the outgoing link is free. This reduces the delay and hence, also the deadline miss ratio.
The successive depths of the topology have traffic shaper and then, no traffic shaper. The nodes without traffic shaper will send the packets whenever the outgoing link is free. The packets in the nodes with traffic shaper hold the packets until their release time. The higher incoming rate of the packets and lower outgoing rate results in buffer overflow quite soon as compared to traffic shaper in all switches as is clear from figure 21(a). Figure 21(b) shows that traffic shaper in all switches maintains higher throughput than traffic shaper in alternate depth.
C. Analysis of Traffic shaper, with children= 4 and depth = 3 in the tree topology:
In this section we compare the QoS parameters for the case when each node in the tree has four children.

![Graph](image1)

Figure 22. Average delay with and without traffic shaper for children=4

![Graph](image2)

Figure 23. Deadline miss ratio with and without traffic shaper for children=4
Figure 24. Packet drop with and without traffic shaper for children=4

The results in the graphs shown in figures 22, 23, 24(a) and 24(b), with 4 children in the tree topology, show conformance with the results we got for binary tree topology and same explanation/discussion as in 6.2, applies here also.