Overloaded IP Node With Implemented Priority Queuing QoS Mechanism

Ilija Efnusev[1] and Toni Janevski[2]

***Abstract* – Although QoS in IP is reality now days, its implementation with all of its power is still negligible. More research before putting it into force is needed. Although the research in this paper is just a tiny part of the whole QoS mechanism, still it is an integral part of the whole idea and brings valuable conclusions.**

This paper analyses the priority queuing simulated on a single IP operating node overloaded of incoming traffic. The incoming traffic is divided in different flows with different priorities, which simulates the priority queuing mechanism. The simulation results show the different output traffic parameters for each flow.

***Keywords* – Quality of service (QoS), IP, Simulation, Priority Queuing, Overloaded**

I. INTRODUCTION

With the computerization of the modern world, new applications are emerging every day. Most of this applications demand remote access, shared resources and globalization of information. This indicates that remote transfer of information is becoming more demanding. The transfer of information needs to be faster and faster.

In the moment we can say that IP won the battle for global transfer technology. As this technology was created to be based on "best effort" transfer of information, the fast developing information world brought new challenges in the IP technology [1]. With the new applications, new services emerge. In these services, speed is not the only limit. Different front-end services demand variety of requirements to be met: maximum Packet Delay, maximum Jitter, minimal Bit Rate, maximum Packet Loss etc. This resulted in IP protocol improvement with the new QoS mechanisms [2].

II. QOS ARCHITECTURE

The QoS architecture needs to satisfy different parameters, on different network levels and in different network parts. This brings us to a complex QoS architecture that has to control the specified parameters of a service requested by the end-user application over different network elements. This means that signaling and traffic admission and control has to be established between network elements.

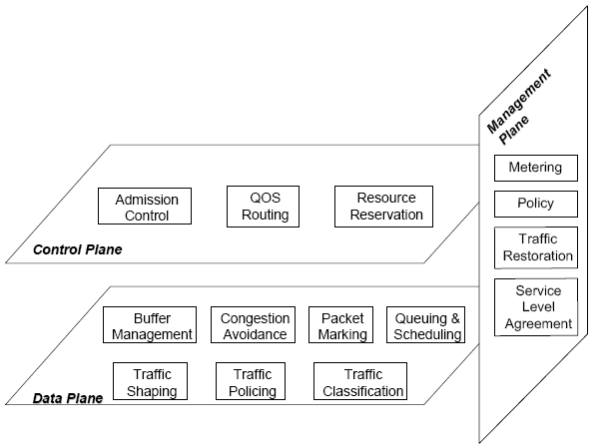

Fig. 1. QoS architectural framework

The complex QoS architecture has been a target of many researchers during the last couple of years. Finally in May 2004 ITU-T came out with the Y.1291 recommendation for the QoS architecture [2]. According to this recommendation the QoS building blocks are organized into three planes, as showed on Fig 1.

The QoS mechanism may refer only to a specific network node (like the Packet marking) or to a network segment (like the QoS routing). This type of architecture requires signaling between network nodes no matter of which part of the network those nodes belong to. The signaling between the QoS segments belonging to the Data plane block is closed in the node (device) and is chosen by the manufacturer, while the signaling in the Control or Management plane is between two or more nodes, so it requires a signaling protocol. The QoS architecture is just a logical framework, so it puts no restrictions of that how the different QoS mechanisms and signaling between them will be realized. Simulating the whole QoS architectural framework is a complicated and demanding process, so only the queuing and scheduling mechanisms will be analyzed in this paper.

III. QUEUING AND SCHEDULING QOS MECHANISM

This mechanism controls which of the incoming packets to transmit to an outgoing link. The incoming traffic could be considered as part of a queuing system, consisted of multiple queues and a scheduler. So by controlling the queuing and scheduling of packets the QoS requirements could be met. This mechanism can still be divided into several approaches:

[1]Ilija Efnusev is Quality of Service Specialist in T-Mobile Macedonia AD. Skopje, Total Quality Management Department, Orce Nikolov bb. Skopje Macedonia, ilija.efnushev@t-mobile.com.mk

[2] Toni Janevski is Prof. Dr. at the "St. Cyril and Methodius" University in Skopje, Faculty of Electrical Engineering and Information Technologies, Dept. of Telecommunications, Karpos 2 bb. Skopje, Macedonia, tonij@etf.ukim.edu.mk

- **First-In, First-Out (FIFO) queuing:** This is when packets are placed into a single queue and they are served in the same order as they arrive in the queue.
- **Fair queuing:** This is when packets are classified into flows and then assigned to queues dedicated to the respective flows. Queues are serviced in round robin method, where empty queues are skipped. This is also referred to as a per-flow or flow-based queuing, where statistically every flow gets the same attention.
- **Priority queuing:** This is when packets are first classified and then placed into different priority queues. Packets are scheduled from the head of a given queue only if all queues of higher priority are empty. Within each of the priority queues, packets are scheduled in first-in, first-out order.
- **Weighted fair queuing:** This is when packets are classified into flows and assigned to queues dedicated to respective flows. A queue is assigned a percentage of output bandwidth according to the bandwidth need of the corresponding flow. By distinguishing variable length packets, this approach also prevents flows with larger packets from being allocated more bandwidth than those with smaller packets.
- **Class-based queuing:** This is when packets are classified into various service classes and then assigned to queues assigned to the service classes, respectively. Each queue can be assigned a different percentage of the output bandwidth and is serviced in round robin. Empty queues are skipped

IV. SIMULATION MODEL

The simulations were made by custom programmed simulator in C. This simulator was made with the purpose of studying the queuing and scheduling QoS mechanisms. The basic model of the simulated node is given on Fig. 2

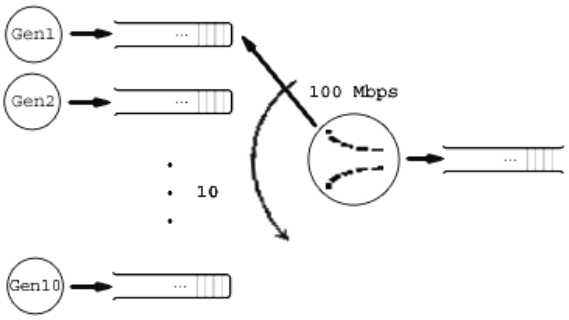

Fig. 2 Simulation Model

A. Simulator characteristics

The custom made simulator has the following characteristics:

- 10 buffers with 10 kB of buffer space
- Packets in each buffer are served by the FIFO queuing.
- If the buffer is full incoming packets are dropped.
- Each buffer together with the generator simulates one flow defined in the ITU QoS architecture.
- The buffers have different priority. The first buffer has the highest, and the 10-th has the lowest priority.
- A buffer with lower priority is only served if all the buffers with higher priority are empty.
- The entering traffic is equally distributed in flows, depending on the number of flows used (ex: If the incoming traffic is 100 Mbps and we have 4 flows, each flow would have speed of 25 Mbps)
- The Node works with speed of 100 Mbps, and serves packet by packet.
- The output buffer is unlimited
- The generators are defined with by their speed. They generate packets with Packet length between 50 and 1500 Bytes (the size of the packet exponentially distributed), and inter-arrival times also exponentially distributed. This simulation model is according the propositions in the reference paper [3].

B. Measured parameters

The simulator measures the following parameters:

- **Transmitted Flow Bit Rate**, measured in %. This is a parameter that shows the % of the Mean Bit Rate of the Flow at the output of the node, compared to the Total Bit rate at the nodes output, according to equation (1).

$$TransmittedFlowBitRate = \frac{OutputFlowMeanBitRate}{TotalNodeOutpuBitRate}100\% \quad (1)$$

- **Lost Packets**, measured in %. This shows the number of packets lost compared to the total incoming packets, per flow. See equation (2).

$$PercentOfLostPackets = \frac{FlowsLostPackets}{Total\ \text{Flows}\ Generated\ \text{Packets}}100\% \quad (2)$$

- **Mean Delay**. This is the average delay of the packets in the flow, calculated as the time difference at the input and output time of the packet in the Node, formula (3). This includes the waiting time in the buffer, plus the serving time by the node.

$$MeanDelay = \frac{\sum_{n=0}^{n=N} (t_{(output)n} - t_{(input)n})}{N} \quad (3)$$

- **Jitter**. This shows the time deviation of the packets at the output from the mean expected output time. This parameter is calculated according to the formula (4).

$$Jitter = \frac{\sum_{n=0}^{n=N} |\ MeanDelay - (t_{(output)n} - t_{(input)n})\ |}{N} \quad (4)$$

V. SIMULATION RESULTS

These simulations were made with the constraint that incoming traffic speed is double than the serving nodes speed. So depending on the number of flows, each flows bit rate is 200Mbps/N, where N is the number of flows. Each flow can be considered as a service which needs different QoS parameters to be satisfied. This simulates what will happen if there is no flow control between the nodes in the network and in one moment the incoming traffic doubles.

Each of the measured parameters in the simulation will be discussed separately.

A. Transmitted Flow Bit Rate

This parameter actually shows which % of the output mean bit rate (the serving rate of the node) has each Flow at the output buffer of the node. The measured results are shown in Fig. 3. From the figure it is clear that the total output Bit Rate (utilization of the output link) is bigger when traffic is segmented in more flows. This is due the fact that from time to time if there is only one buffer, there is a probability that it might be empty although the packets on the input are coming with bigger speed than they are served. While when there are more buffers the node starts processing immediately the packets in the next buffers. Because the packets from lower priority queues are served only when all of the queues with higher priority are empty it can be seen that lower priority flows are rarely served. So they have low transmission rate, as we will se later on Fig.4 they have great packet loss. This leads to a conclusion that, segmenting the flows in more than four flows makes the node unstable, and with terrible performance when overloaded.

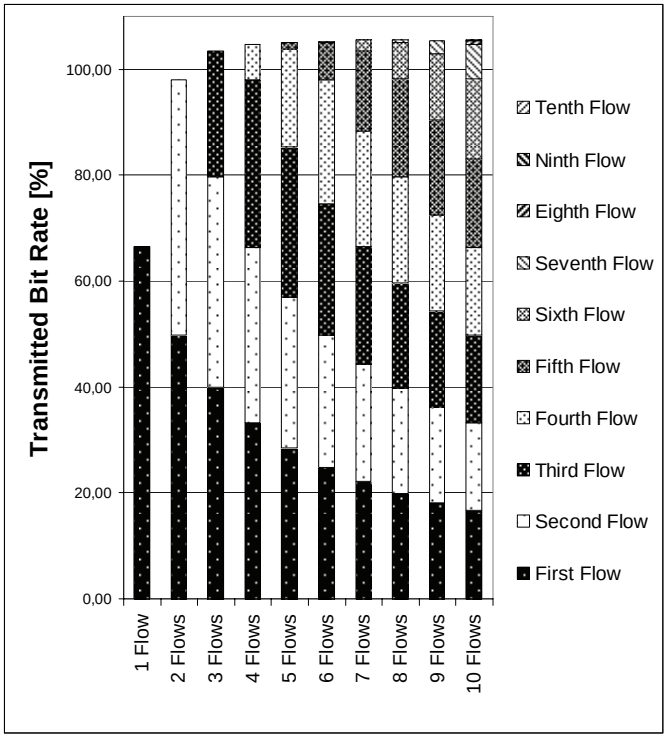

Fig. 3 Transmitted Bit Rate [%]

B. Packet Loss

This parameter shows the percent of the packets that are lost from each flow. Packets get lost if the buffer of the flow is full and new packet is generated by that flow. The results are shown on Fig. 4. As it can be seen in the figure when traffic is separated in more than 4 flows there is packet loss of over 90% in the flows with lower priorities. This means that these flows are blocked and the node doesn't transmit their traffic further. This brings us to the conclusion that flows with lower priorities would have to implement a mechanism of packet retransmission. If the overloading of the node is just for a short time this packets would be retransmitted and the services defined in this flows will continue functioning with some delay. But if the overloading of the node is continuous for these services it would seem like the node is down.

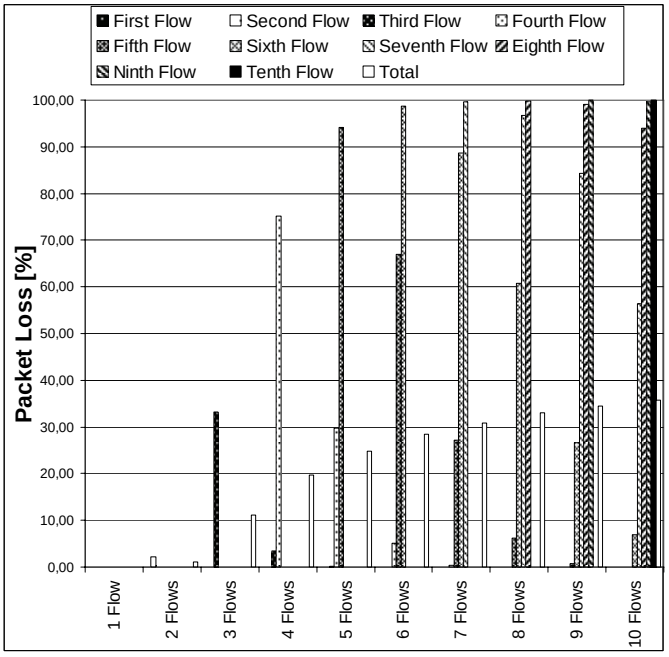

Fig. 4 Packet Loss [%]

B. Mean Delay

This parameter shows the mean delay that the packets experience when served in the node. The delay is as a result of the waiting time in the buffer and the time needed for the packet to be served by the node. The mean delay of each flow is shown on Fig. 5 given in [ms] on a logarithmic scale. Flows with lower priority have enormous delay when the node is overloaded due to their waiting time in the buffer. The highest priority flows tend to the average serving time of the node according to equation (5).

$$ServingTime = \frac{AveragePacketSize[bit]}{NodeOutputSpeed[bit/s]} \qquad (5)$$

From the figure it is clear that the services defined in flows with lower priorities have to be tolerant to packet delay because if we consider that this is just the delay at one node

the total delay in a network would be n times bigger, where n is the number of nodes. When there are more than 4 flows, delays are unbearable for any existing IP service.

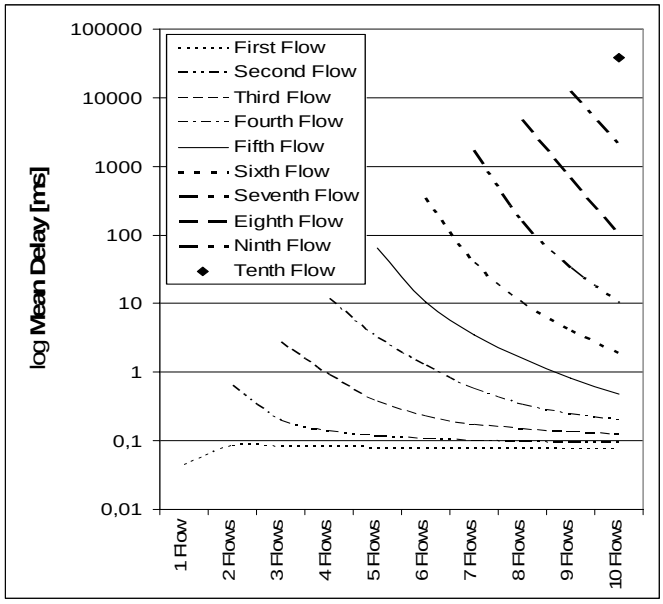

Fig. 5 Mean delay [ms]

B. Jitter

This parameter actually is the mean deviation from the delay of the packets served by the node. The results of the simulation are shown on Fig. 6 in [ms] on a logarithmic scale. The jitter is extreme for flows with lower priority. Services like internet browsing or file download are resistant to jitter and delay but when there are more than 4 flows the jitter is unbearable even for these services. Other real-time services like video and audio streaming are intolerant to jitter so this kind of services must be defined in flows of highest priority.

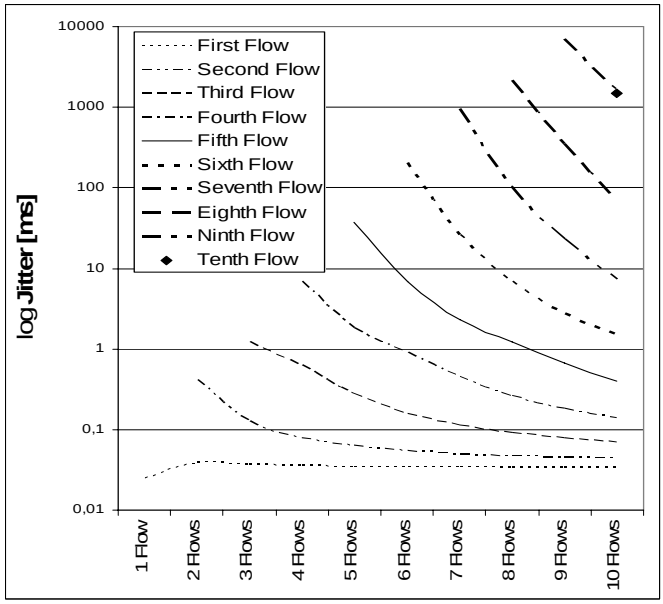

Fig. 6 Jitter [ms]

VI. CONCLUSION

As it can be seen from the simulation results, this kind of implementation of QoS on node level shows deteriorated performance in lower priority flows, when the node is overloaded. This is especially visible when the number of flows is grater (more different service levels are defined). So by the simple implementation of QoS and dividing the traffic in different flows, the low priority flows won't show any performance. As it can be seen in the simulations results, there is a packet loss of over 90% at almost half of the flows. But this is not the only problem. All of the other parameters are also extremely degraded as well. Looking at all the parameters together it can be said that four is the optimal number of flows with different QoS parameters that should be defined. Besides this, all parameters also suggest that a simple implementation of the QoS mechanisms on node level needs to be combined with other QoS mechanisms: flow control, QoS routing and other, to achieve performance and sustain service quality. With this paper it is clearly shown that implementing QoS in just part of the network is not a solution. There has to be a continuity of QoS mechanisms across the network and between networks, so that QoS would really work. Service providers have to become conscious of the power that QoS brings and upgrade their systems to support the QoS mechanisms.

ACKNOWLEDGEMENT

I would like to express my gratitude to the professor Toni Janevski and the Telecommunication Institute at the Faculty of Electrical Engineering and Information Technologies at "St. Cyril and Methodius" University in Skopje, for the shown cooperation and help in my research.

REFERENCES

[1] Toni Janevski, Traffic Analysis and Design of Wireless IP Networks, Artech House Inc. 2003

[2] ITU-T, An architectural framework for support of Quality of Service (QoS) in packet networks (Ref. No.: Y.1291)

[3] Zafer Sahinoglu and Sirin Tekinay, Multimedia Networks: Self-Similar Traffic and Network Performance, New Jersey Institute of Technology.