Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 69 (2014) 655 – 664

Procedia Engineering

www.elsevier.com/locate/procedia

24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013

Configuration of Quality of Service Parameters in Communication Networks

Hrvoje Kozačinski*, Petar Knežević

Faculty of Electrical Engineering and Computing, Unska 3, Zagreb 10000, Croatia

Abstract

Quality of Service in communication networks is a request for minimal required performance needed for transferring IP packets in a network. The performance can be ensured with a set of mechanisms that guarantee required performance through shaping and policing of packet traffic. Their goal is to prevent congestion, packet delays and packet loss. The goal of this paper is to introduce some of the commonly used Quality of Service mechanisms and analyze their impact on network traffic in order to improve its performance, mainly packet delay. The mechanisms are analyzed trough a simple packet transfer simulator that offers different combinations of mechanisms to be used in the simulation, allowing different packet transfer scenario setups. After a simulation is complete, the simulator offers results along with relevant packet transfer graphs. The impact of Quality of Service mechanisms is analyzed and interpreted trough those results. It is shown that combining Quality of Service mechanisms in packet transfer generally gives better results. Also, different combinations of mechanisms give different results, allowing more flexibility in improving packet transfer in a desired way, but also making it more predictable and controllable.

Keywords: QoS; Quality of Service; Communication Network; Token Bucket; Packet Transfer

1. Introduction

Quality of Service (QoS) is a set of requirements that guarantee certain performance in packet transfer. A good implementation of QoS usually minimizes packet delay and reduces the number of discarded packets. To ensure

* Corresponding author. Tel.: +38-591-973-9345.
E-mail address: hrvoje.kozacinski@fer.hr

doi:10.1016/j.proeng.2014.03.039

such performance, QoS has a set of mechanisms that help regulate the transfer, so that the performance requirements are met.

A communication network usually consists of many communication nodes and routes between them. The information is sent trough the network within IP packets. One node sends the packets - a sender, the other receives them - a receiver. They are not necessarily connected together, in most cases there are nodes and routes between them. Because of that, it is possible that not all packets are sent trough the same route and following errors can occur:

- Packets arriving in wrong order via different routes.
- Jitter when packets arrive in the wrong order, but are delay sensitive. The effect can be heard and seen in video streaming or VoIP communication.
- Errors in transfer. Packets can be lost in transfer, or they can also get stuck on congested routes and get delayed. When they are delay sensitive and are delayed too much, they become unusable.

In real life situations, the receiver is usually slower than the sender. Senders are clients that connect to servers (receivers). Clearly, all clients cannot be handled in the exact same time and need to wait for the server response. This results in slower transfer, packet delay and possible packet loss. There are different approaches to solving poor transfer performance, mainly by introducing different QoS mechanisms in communication networks that help reduce packet delay and packet loss [1,6]. The problem this paper analyzes is how exactly QoS mechanisms impact a communication network performance in attempt to improve packet transfer, primarily by reducing packet delays and packet loss.

2. Quality of service mechanisms

Some of the main QoS mechanisms that are used in communication networks to help improve performance will be explained in this paragraph, along with their desired effects and shortcomings.

Retransmission: A mechanism responsible for resending packets lost in the transmission, thus preventing loss of information. It helps with acquiring the complete information, but with a cost - increased transfer time and packet delay[7].

Buffer: A container used to temporarily store packets that the receiver cannot process upon arrival, preventing retransmission which would cause bigger time delay and unneeded bandwidth usage. This, however, does not solve packet delay entirely. The bigger the buffer, the more packets need to wait for retrieval (extending their delay) if the buffer is getting full, unless other mechanisms are applied [9].

Buffer collecting strategies: Packets can be collected from the buffer in different ways. The most common way is to check for packets from the beginning of the buffer and collect the first available packet. The downside of this solution is that packets further in the buffer have prolonged delay times while they wait to be collected. Different approaches try to improve packet collecting in some way. For instance, a round robin buffer collecting strategy collects packets in a round fashion, to correct the flaw of the first approach, evening the delay times.

Packet priority: A mechanism that classifies packets by their importance. Important packets get higher priority than those that are less important and will be processed first by the receiver. This classification helps reducing packet loss and delays of packets with higher priority, but by increasing packet delay of lower priority packets[10].

Policing mechanisms: Mechanisms used to discard packets from the buffer to prevent its congestion, which would result in retransmission of packets that cannot enter the buffer. Instead, discarded packets are sent in retransmission. This allows new packets to enter the buffer, assuming they have the newest information, more relevant to the receiver than the discarded packets. The most common policing mechanisms are RED (*Random Early Detection;* also *Discard* or *Drop*)[11] and WRED (*Weighted* RED, uses packet priority to help decide which packets to

discard). If the priority mechanism is used, WRED can discard less significant packets to prevent congestion and allow future higher priority packets to be received [5,6].

Shaping mechanisms: Mechanisms used to shape the sending flow of packets to prevent buffer congestion, packet drop and retransmission, avoiding unnecessary bandwidth usage and packet delays. The most common are the *Token Bucket* and *Leaky Bucket* algorithms [2,4].

3. The simulator of QoS mechanisms

The goal of this paper is to give theoretical insight of Quality of Service and its mechanisms, but also to back it up with a simple simulation to further illustrate the use of mechanisms, and the effect they have on packet transfer. The main focus is on the following mechanisms: retransmission, buffer, buffer collecting, packet priority with 3 priorities (0-2, 0 being the highest) and the Token Bucket algorithm, as they are supported by the simulator.

Their impact on packet transfer will be observed and measured in eight transfer scenarios. In every scenario, the basic transfer parameters will remain constant:

- Buffer size: 60
- Packets to send: 140
- Sender speed: 1
- Receiver speed: 3
- Tokens: 60 (maximum token capacity for the Token Bucket algorithm)

The simulation covers the basic case of one sender and one receiver connected together. This is because the implementation of all the mechanisms and parameters of a communication network is very complex. It is also simplified deliberately to better illustrate the use of Quality of Service mechanisms and their effect on packet transfer so they can be better understood and interpreted.

After the simulation is completed, three graphs are generated: two packet transfer delay graphs (with and without marked priorities) that show the time delay of each packet and a buffer capacity graph that shows how the capacity changed in the buffer during the transfer.

The simulator also displays relevant transfer data and generates log files of:

- Total transfer time
- Number of discarded packets
- Number of discarded packets depending on priorities
- Number of packets discarded in retransmission
- Average packet time delay inside the buffer
- Average packet time delay inside the buffer depending on packet priorities

Each scenario was run multiple times and average values are presented. This is necessary to reduce measuring errors caused by outside interferences, because the simulation was run on a personal computer which contained other running processes.

4. Simulation scenarios

The simulation is divided in 8 different scenarios, depending on the combination of QoS mechanisms used, as seen in Table 1. All scenarios use retransmission and the buffer mechanism with one of the buffer collecting strategies. Each scenario will be explained in more detail, along with its results, through one typical simulation of each scenario.

Table 1. Simulation scenarios and mechanisms used.

Scenarios	Priority	Buffer collecting	Token Bucket
Scenario 1	No	First available	No
Scenario 2	Yes	First available	No
Scenario 3	No	First available	Yes
Scenario 4	Yes	First available	Yes
Scenario 5	No	Round robin	No
Scenario 6	Yes	Round robin	No
Scenario 7	No	Round robin	Yes
Scenario 8	Yes	Round robin	Yes

4.1. Scenario 1

The first scenario uses as few mechanisms as possible, the only unavoidable mechanisms being the retransmission and the buffer mechanisms along with the buffer collecting strategy that collects the first available packet in the buffer.

Delays of packets can be seen on Figure 1. The graph marks different packet priorities with different colors, but because the priority mechanism isn't activated, packet priority has no effect on delay times. There are three distinct lines on the graph, one horizontal line and two lines with linear growth. The horizontal line is caused by the buffer collecting strategy, which collects the first packet in the buffer it comes across, resulting in low time delays. The first linear growth is caused by the delay the packets have while waiting in the buffer. The second linear growth is the result of retransmission that started after approximately 85 packets were sent and the buffer was congested. The retransmission time delay appears smaller, but it is in fact larger. This graph shows only time delays of packets while waiting in the buffer. Packets that went into retransmission were sent, rejected, then retransmitted again and then waited in the buffer, so their total time delay is larger than that of the packets that immediately went into the buffer.

Another relevant graph is the buffer capacity graph, shown in Figure 2. The buffer capacity was exponentially filled to the point of congestion, at approximately 135 ms. After that, surplus packets were discarded and sent into retransmission. At that point, the receiver collected packets, but the sender filled the empty buffer spot quickly, as is shown on the graph with the horizontal line spanning to 200 ms in transfer. After that the drop in sending rate because of the slow retransmission rate that helped lower the buffer capacity, until all packets were collected from the buffer.

The total transfer time was 507 ms, total of 38 packets were dropped in transfer (12 priority 0, 12 priority 1 and 14 priority 2). Average packet time delay is 161 ms (164 ms priority 0, 154 ms priority 1 and 168 ms priority 2). The results clearly show that without a priority mechanism all packets are treated equally. This is the basic scenario on which all others will try and improve.

4.2. Scenario 2

This scenario adds the priority mechanism to scenario 1, to observe and measure its effects on packet transfer, seen on Figure 3. Packets are clearly distinguished by their priority, priority 0 has the lowest time delay, followed by priority 1 and priority 2 with the biggest time delay. A linear growth after the 85th packet is again the result of retransmission. At that point, packet priority is leveled and begins to lose its importance.

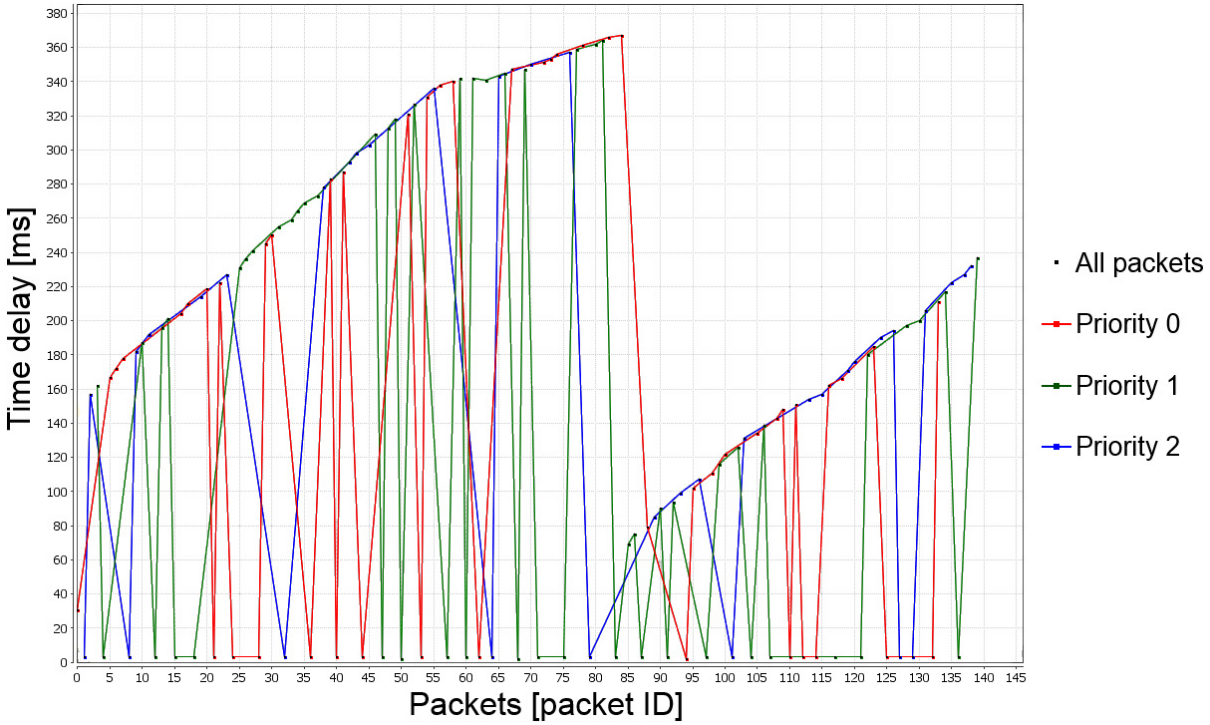

Fig. 1. Packet delay during transfer graph, scenario 1.

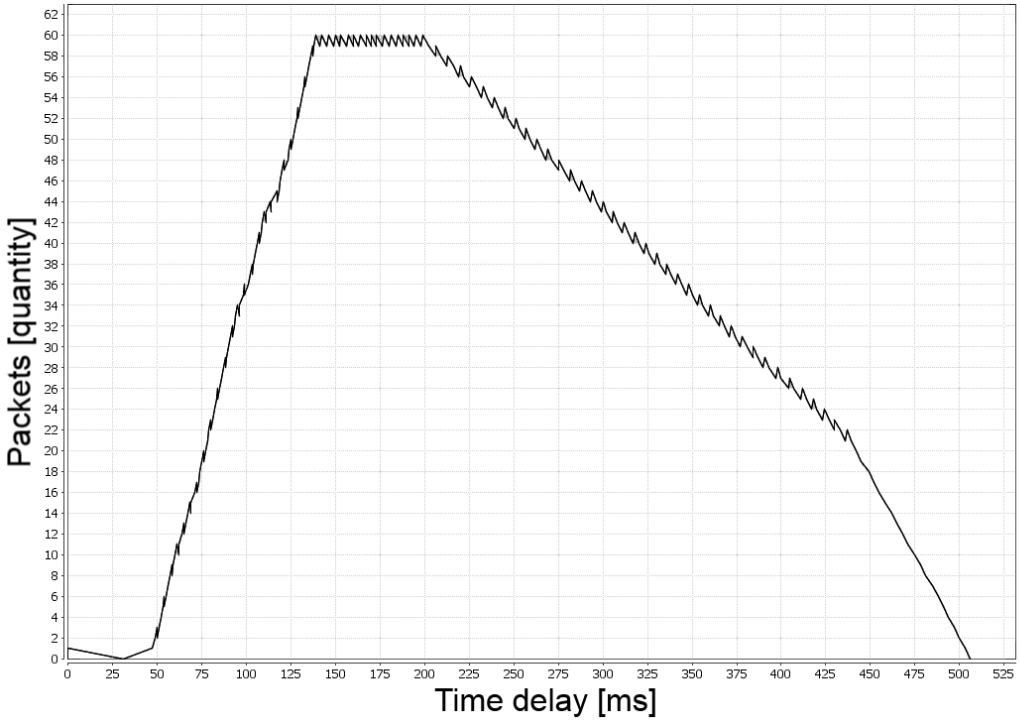

Fig. 2. Buffer capacity during transfer graph, scenario 1.

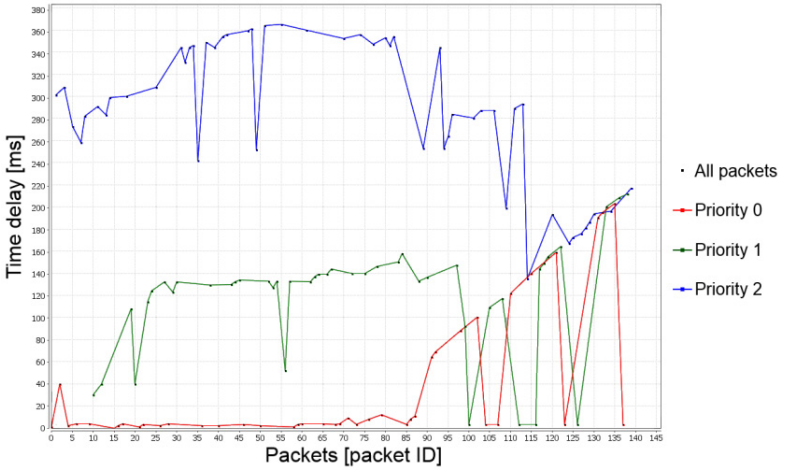

Fig. 3. Packet delay during transfer graph, scenario 2.

The total transfer time was 491 ms, total of 36 packets were dropped in transfer (10 priority 0, 10 priority 1 and 16 priority 2). Average packet time delay is 155 ms (34 ms priority 0, 118 ms priority 1 and 287 ms priority 2). The priority mechanism helped with both the dropped packets and packet time delays, sacrificing priority 2 drop rate and time delay while improving priority 0 and 1.

The buffer capacity behavior is similar to that of scenario 1.

4.3. Scenario 3

This scenario adds the Token Bucket algorithm to scenario 1, to observe and measure its effect on packet transfer, seen on Figure 4. The idea behind the Token Bucket algorithm is to control the rate at which the packets are sent. The algorithm periodically fills a container (bucket) with tokens, up to a set capacity (usually around the buffer capacity level). A packet can only be sent if there are sufficient tokens in the bucket to "pay" for sending. At first, with enough tokens at the beginning of transfer, packets can be sent in a large burst, until the container has no more tokens. After that, the periodical appearance of tokens controls the send rate. This helps buffer congestion, reduces the need for retransmission and because of that also reduces packet drop rates and delay times [2,3].

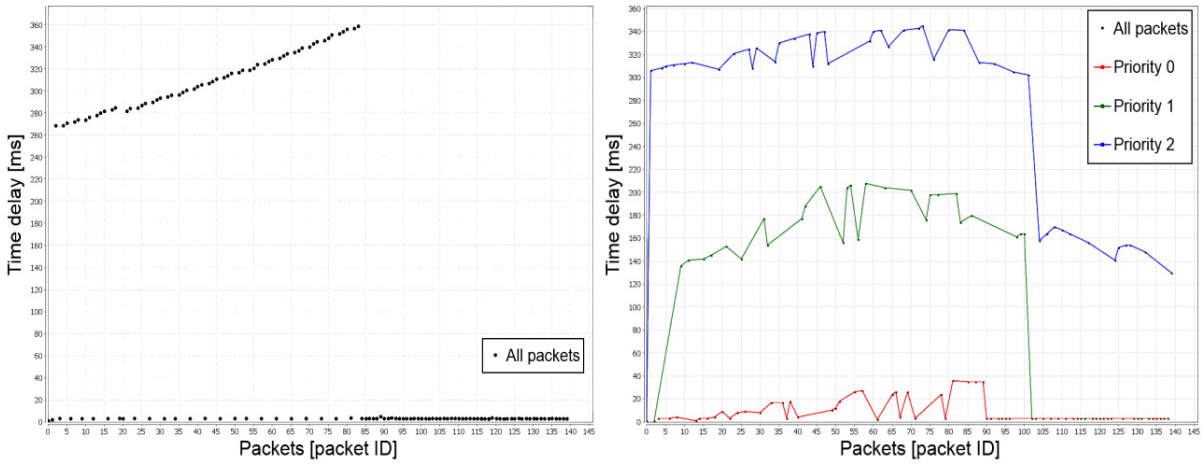

Fig. 4. Packet delay during transfer graph, (left) scenario 3; (right) scenario 4.

The total transfer time was 450 ms and because of the Token Bucket algorithm, no packets were dropped in transfer. Average packet time delay is 132 ms (126 ms priority 0, 170 ms priority 1 and 105 ms priority 2). The shaping mechanism helped with packet drops, but this is an idealized example, where the algorithms tokens are refilled at the same rate the receiver can receive a packet. In a real life situation, the algorithm would require a careful parameter calibration to get as close to the idealized example as possible. Higher sending rate would cause packet drops and lower rate would cause bigger packet time delay.

The buffer capacity graph looks similar to the graphs in scenarios 1 and 2, only the horizontal line where the sender and the receiver compete while the buffer is at full capacity lasts longer and ends with a smooth linear drop.

4.4. Scenario 4

In this scenario both the priority mechanism and the shaping mechanism are combined. The effect this has on packet transfer is visible on Figure 4.

The total transfer time was 446 ms and no packets were dropped in transfer. Average packet time delay is 127 ms (9 ms priority 0, 115 ms priority 1 and 272 ms priority 2). In this case, combining both mechanisms really paid off, especially when observing the priority 0 delay times. The time delay drop at the end of the graph is the result of the Token Bucket algorithm that reduced the send rate once it ran out of tokens.

4.5. Scenario 5

In this scenario, and all the following scenarios, the buffer collecting strategy is changed. Packets will be collected by using the round robin strategy. This scenario has no additional mechanisms. The effect this collecting strategy has on packet transfer can be seen in Figure 5.

The idea of the round robin collecting strategy is to improve the basic collecting strategy. Because of the way the packets are collected, the strategy evens the time delays across all the packets that enter the buffer. In the graph this is shown as the horizontal line in the middle. The two linear growths are because of the fact that the sender is filling the buffer faster than the receiver can collect the packets and because of the inevitable retransmission once the buffer is congested.

The total transfer time was 486 ms, total of 41 packets were dropped in transfer (19 priority 0, 12 priority 1 and 10 priority 2). Average packet time delay is 166 ms (182 ms priority 0, 157 ms priority 1 and 158 ms priority 2).

The buffer capacity graph is similar to the same graph in scenario 1.

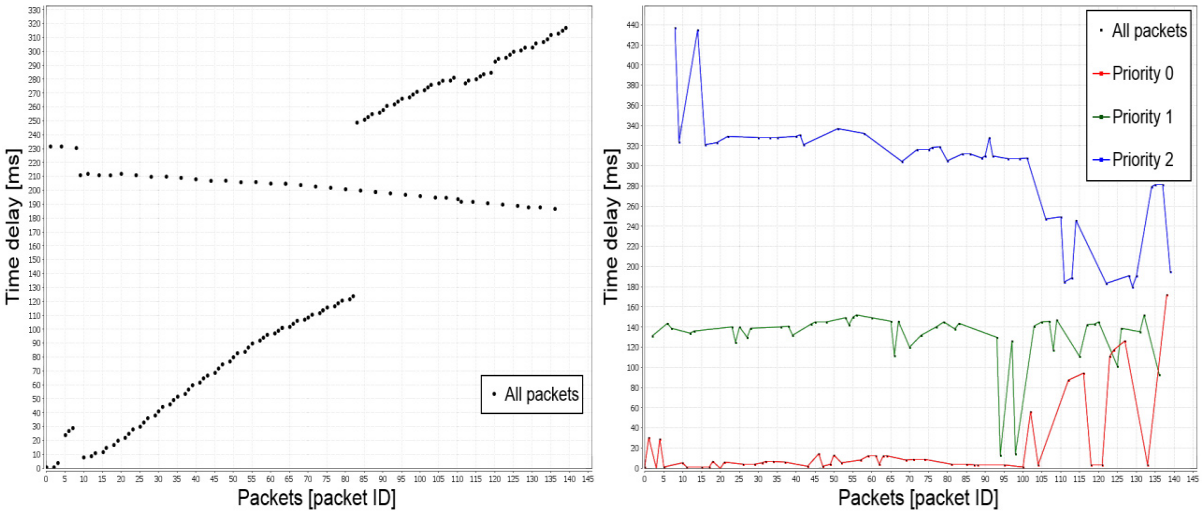

Fig. 5. Packet delay during transfer graph, (left) scenario 5; (right) scenario 6.

4.6. Scenario 6

This scenario uses the round robin collecting strategy along with the packet priority mechanism. Packet transfer delay times are shown in Figure 5.

The total transfer time was 484 ms, total of 24 packets were dropped in transfer (7 priority 0, 10 priority 1 and 7 priority 2). Average packet time delay is 140 ms (20 ms priority 0, 131 ms priority 1 and 294 ms priority 2). The priority mechanism helped with packet time delays, sacrificing priority 2 time delay while improving priority 0 and 1. The priority does not seem to have an effect on packet drop rate in this example.

The buffer capacity graph is similar to the same graph in scenario 1.

4.7. Scenario 7

This scenario uses the Token Bucket algorithm along with the round robin strategy. Packet transfer delay times are shown in Figure 6.

The graph looks similar to the graph in scenario 5, except there is no additional delay caused by retransmission since no packets are dropped.

The total transfer time was 463 ms and because of the Token Bucket algorithm, no packets were dropped in transfer. Average packet time delay is 127 ms (141 ms priority 0, 120 ms priority 1 and 123 ms priority 2).

The buffer capacity graph is similar to the one in scenario 3.

4.8. Scenario 8

In this scenario both the priority mechanism and the shaping mechanism are combined with the round robin buffer collection strategy. The effect this has on packet transfer is visible on Figure 6.

The total transfer time was 450 ms and no packets were dropped in transfer. Average packet time delay is 133 ms (8 ms priority 0, 120 ms priority 1 and 270 ms priority 2). In this case, combining both mechanisms also paid off, like in scenario 4, especially when observing the priority 0 delay times. The time delay drop at the end of the graph is the result of the Token Bucket algorithm that reduced the send rate once it ran out of tokens.

The buffer capacity graph is similar to the one in scenario 3.

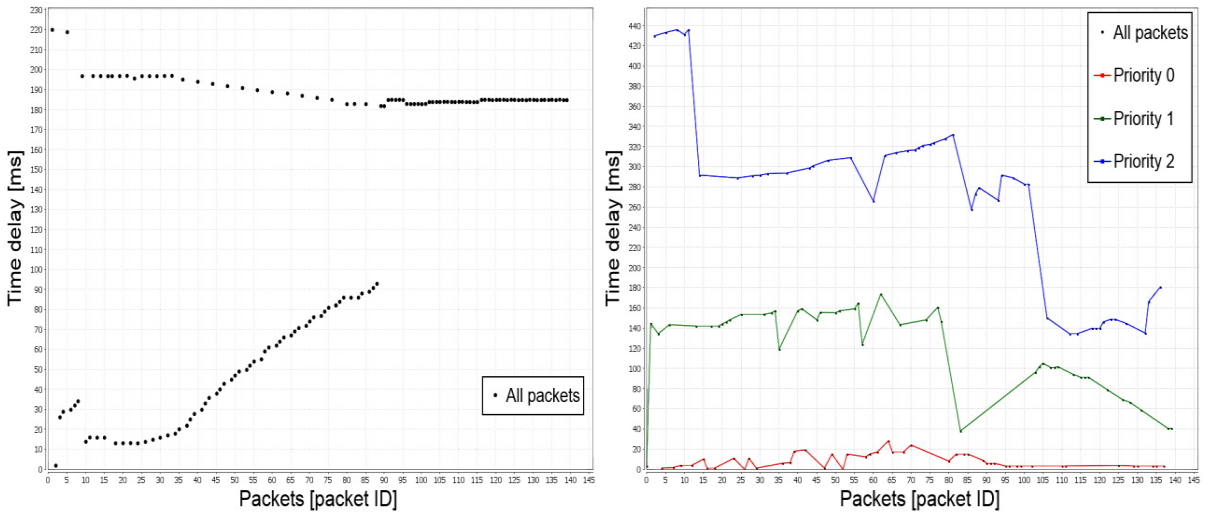

Fig. 6. Packet delay during transfer graph, (left) scenario 7; (right) scenario 8.

5. Simulation results

After all the scenarios were run 25 times, average values of observed parameters were calculated. The most interesting parameters to analyze are the total transfer times, given in Figure 7, and average packet time delay depending on packet priority, given in Figure 8.

If only average transfer times are observed, the conclusion is that scenario 5 (round robin, no priority, no Token Bucket) has the lowest transfer time of 469.72 ms. The highest transfer time can be seen in scenario 7 (round robin, no priority, Token Bucket) with 576.76 ms, followed closely by scenario 8 (575.8 ms) and scenario 4 (574.24 ms). The conclusion that can be drawn from this figure is that if lowest possible transfer time is desired in a communication network, only a buffer and the round robin buffer collection strategy will suffice.

On the other hand, if average packet time delay depending on packet priority for each scenario is observed, there are two distinct patterns. The flat columns of scenarios 1, 3, 5 and 7 are the result of the scenarios not having the priority mechanism, and are not too relevant for this observation. Other scenarios have more interesting results. The lowest time delay for priority 0 can be seen in scenario 4 (5.04 ms), with scenario 8 close by (6.32 ms). They also have the lowest time delay of priority 1 packets (110.16 ms and 109.72 ms respectively). However, scenarios 2 and 6 have lower time delays for priority 2 packets (319.84 ms and 325.24 ms versus 340.2 ms of scenario 4 and 335.88 ms of scenario 8).

For best priority 0 packet time delay, the priority mechanism should be used along with the Token Bucket algorithm, regardless of the buffer collecting strategy. Round robin strategy does give a bit higher priority 0 delay, but has better delays for other priorities, which can also be relevant, depending on the situation.

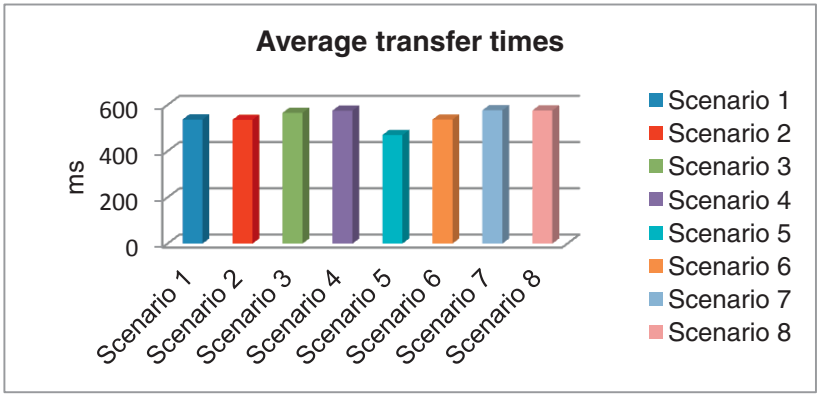

Fig. 7. Average transfer times for all scenarios.

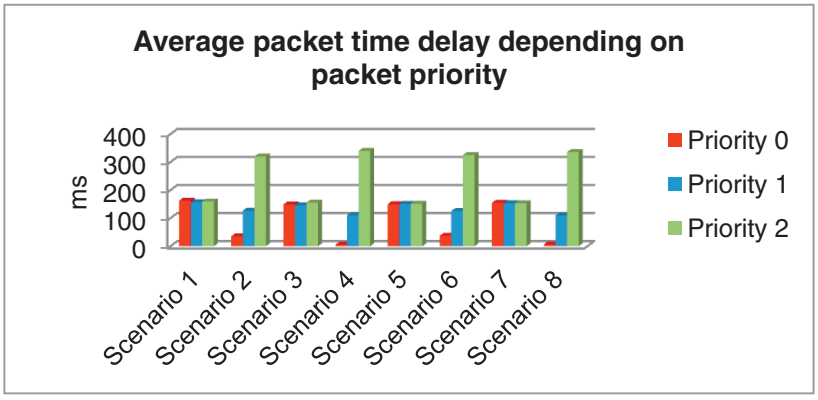

Fig. 8. Average packet time delay depending on packet priority.

6. Conclusion

The main problem of this paper was IP packet delay, the impact it has on communication networks and how it can be reduced to improve Quality of Service. This can be achieved by implementing QoS mechanisms. A simulator was used to further study the impact each of the mechanisms has on packet delays, both alone and combined with other mechanisms.

The results show that a clear solution to QoS in communication networks does not exist. Instead, the solution greatly depends on the network itself and the individual needs for that particular network. The results have also shown that combining the QoS mechanisms can give various results, depending on which mechanisms are used. If only fast packet transfer is needed, regardless of packet priority, the round robin buffer collection strategy can be enough. If, on the other hand, packets have different priorities or some are more sensitive to delays than others, priority mechanism should be used combined with the Token Bucket mechanism.

Although an ultimate solution does not exist, a careful selection of mechanisms can provide a good solution, because using the QoS mechanisms makes the packet transfer more clear, more predictable and more manageable, eliminating or lessening the undesirable effects in network transfer. This also allows declaring minimal performance guarantees necessary when establishing QoS in a communication network[8].

The simulator itself, like all software solutions, could keep on improving. Further improvements include adding new QoS mechanisms in the simulation, like the Leaky Bucket algorithm shaping mechanism and policing mechanisms (RED and WRED explained at the beginning of the paper). Retransmission could be more accurate by sending packets immediately after they get discarded, not at the end of transfer. Also, packet priorities could be more configurable, allowing a person to choose a number of priorities and the ratios between them for better simulating different situations where packet priority may differ in transfer. Other major improvements include creating a simulator that can show real time results and be configured on the fly, while measuring drop rates and peak delays. This can give more insight on how the transfer behaves in a more dynamic environment using different QoS mechanisms or under different network conditions.

All possible improvements aside, this simulator was designed to be simple on purpose, so it can be more illustrative. The idea was to show as clearly as possible what individual and combined effects QoS mechanisms have on packet transfer and to measure and analyze them more easily.

References

[1] *System i Networking Quality of Service (Version 6 Release 1)*, IBM Corp., 2009., *Available from:* http://publib.boulder.ibm.com/infocenter/iseries/v6r1m0/topic/rzak8/rzak8.pdf, *Accessed:* 2013.7.1.

[2] Ali Mantar, H. (2000). *The Token Bucket (Leaky Bucket) Model*, Syracuse University, *Available from:* http://qbone.internet2.edu/bb/Bucket.doc, *Accessed:* 2013.7.1.

[3] *Queue Management*, Rutgers University, *Available from:* http://www.research.rutgers.edu/~xili/cs352/queue-management.ppt, *Accessed:* 2013.5.14.

[4] Domokos Botond, B. (2001). *Traffic Shaping*, *Available from:* http://ftp.utcluj.ro/pub/users/tarc/biro.ppt, *Accessed:2013.7.2.*

[5] Francis-Cobley, P. (2004). *Congestion Control*, *Available from:* http://ftp.utcluj.ro/pub/users/cemil/prc/CONGESTION%20CONTROL.ppt, *Accessed:* 2013.5.15.

[6] *Quality of Service Networking*, Cisco Press, *Available from:* http://docwiki.cisco.com/wiki/Quality_of_Service_Networking, *Accessed:* 2013.7.2.

[7] Anonymous, (2010). *Komunikacijske mreže, 7. Transportni sloj. Transportni protokoli u Internetu* (*notes from the lesson*)

[8] Kozacinski, H. & Knezevic, P. (2011). Configuration of Quality of Service Parameters in Communication Networks, Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium, 23-26th November 2011, Vienna, Austria, Volume 22, No. 1, ISSN 1726-9679, ISBN 978-3-901509-83-4, Katalinic, B. (Ed.), pp. 1369-1370, Published by DAAAM International Vienna, Vienna

[9] Gettys, J. & Nichols, K. (2011). *Bufferbloat: Dark Buffers in the Internet*, Queue - Virtualization, Volume 9, Issue 11.

[10] Ke, P., Li, Y., Ni, F. (2013). *Multiple Services Scheduling with Priority Queuing Model*, IJCIT, Volume 3, Issue 1, *Available from:* http://ijcit.org/ijcit_papers/vol3no1/IJCIT-120702.pdf, *Accessed:* 2013.11.23.

[11] Patil, G., McClean, S., Gaurav, R. (2011). *Drop Tail and Red Queue Management with Small Buffers: Stability and HOPF Bifurcation*, ICTACT, Volume 2, Issue 2, *Available from:* http://eprints.ulster.ac.uk/21987/1/jct_Spl_Paper_339_344.pdf, *Accessed:* 2013.11.25.