XVII IMEKO World Congress
Metrology in the 3rd Millennium
June 22–27, 2003, Dubrovnik, Croatia

ANALYSIS OF SOME QUALITY OF SERVICE PARAMETERS' IMPACT ON END-TO-END DELAY MEASUREMENTS IN VOIP NETWORK

Petar Knezevic[1], Vedran Srsen[2], Kresimir Segaric[2]

Faculty of Electronical Engineering and Computing[1], Coting d.o.o.[2], Zagreb, Croatia

Abstract – Internet users and Internet service providers are using all kinds of tools for monitoring the communication network. Most of the measurements are done by inserting simulated traffic into the network and observing network performances. When used to measure some time-sensitive applications, like Voice over IP (VoIP), these tools in coherence with some network configuration techniques can result with inaccurate measurement result. We discuss some of those configuration techniques and their influence on delay measurements, and introduce possible solutions.

Keywords: QoS, VoIP, delay

1. INTRODUCTION

The Internet is being used to transport voice. The biggest transmigration of voice transport from the Public Switched Telephone Network (PSTN) to the Internet is expected in the next few years, when voice traffic should represent 10 % of the overall Internet traffic [1]. The primary reason for sending voice over the Internet is reduced call price. The main problem of the idea, or, better to say, challenge, is to make the transmigration unnoticeable for the end user. The packet-switching nature of the Internet Protocol (IP) networks is in the opposite of the circuit-switching nature of the PSTN. Voice data communication must be a real time stream, which makes it sensitive to excessive delay, delay variation (jitter) and packet loss. So, to get closer to the solution of the transparent transmigration process, Internet has to accomplish the same level of quality of the transported voice, which includes minimizing the overall delay and maximizing reliability requirements on voice communication. Because of these strict requirements, it is mandatory to treat voice traffic differently than other network traffic [2]. That can be accomplished by applying Quality of Service (QoS) techniques. QoS ensures that VoIP packets receive the preferential treatment they require. QoS techniques provide for following features:

- sufficient bandwidth
- improving loss characteristics
- avoiding network congestion
- setting traffic priorities across the network.

These features are provided by the network elements (e.g. routers), using software attributes for implementing packet classification, priority servicing through an intelligent output interface mechanism (queueing), packet fragmentation and interleaving and packet compression methods.

The accuracy of end-to-end measurements is of great importance, because deployment of real-time services such as VoIP necessitates the knowledge of whether the quality requirements are met in the network. Some QoS techniques can endanger the accuracy of measurements [3]. In this paper, we will present adequate procedures for reducing the impact of queueing QoS techniques on the end-to-end delay measurements.

2. QUEUING TECHNICS

QoS techniques are applied primarily on routers. The most used QoS techniques when beginning the network optimization are queuing policies. Every output router interface has a queue (a buffer) for holding packets that are waiting for transmission. Queuing policies are mostly classified as packet schedulers. Packet schedulers control the order in which packets leave an interface [4]. As we stated in chapter 1, VoIP traffic must be treated differently then the other network traffic. The primary task of queuing policies for VoIP is to make sure that VoIP packets are moved to the front of the queue so they are not delayed by other packet types. For example, if we assume that a VoIP packet will have to wait behind just one data packet, which can be as large as 1500 bytes, on the 64 kbit/s link it would mean that VoIP packet will be delayed 187.5 ms, which is unacceptable. We tested the impact of some simple (e.g. priority queuing) and some more complex techniques (e.g. low latency queueing – LLQ) on the active measuring methods, such as *ping*. Ping application sends packets of configured size and in configured intervals, and receives answers with the same content, so the RTT (round-trip-time) of each packet is calculated by subtracting the arrival time of the response packet and the departure time of the sent packet.

Priority queueing uses a concept of four queues in hierarchical order: high, medium, normal and low priority queue. The traffic defined as high priority receives the benefit of all available resources on the output interface until the queue is empty. Only when the higher priority queue empties the next lower priority queue takes the benefit of the resources.

Low latency queueing uses a method of classifying all outgoing traffic into classes. Traffic that should have

priority is put into *priority class*, traffic that needs guaranteed minimum bandwidth is put into *reserved classes*, and all other traffic that is not classified is put into *default class*. The default class traffic can be given a reserved queue or can be placed in an unreserved default queue where each flow will get an approximately equal share of the unreserved and available bandwidth (fair queue). The scheduler services the queues so that the priority queue traffic is output first unless it exceeds a configured priority bandwidth and this bandwidth is needed by a reserved queue (that is, there is congestion). During periods of congestion, the priority queue is policed at the configured rate so that the priority traffic does not monopolize all the available bandwidth. Low latency queueing is much more appropriate for prioritizing VoIP than priority queueing, because when using priority queueing higher priority traffic can starve the lower priority queues of bandwidth. No bandwidth guarantees are possible. With LLQ we have an ability to give priority to multiple classes and in the same time to reserve bandwidth for non-prioritized traffic. Even though, because there is no mechanism for providing multiple levels of priority, all prioritized traffic is sent through the same priority queue. This sharing of priority queue between applications can introduce jitter, but this topic is beyond the scope of this paper.

The test network configuration is shown in Fig.1.

To simulate network congestion, we started 50 simultaneous TCP (Transport Control Protocol) connections between PC1 and PC2. Routers are interconnected with 64 kbit/s PPP (Point-to-Point Protocol) serial lines. Ping packets are 60 bytes each, and sent with 20 ms interval between them, simulating a VoIP call using G.729 coder [5].

In the first example, the output serial interfaces of routers R1 and R4 are configured with priority queueing putting the VoIP traffic in the high priority queue, and all other traffic in the default (normal) priority queue. In priority queueing, packets from the higher priority queue are serviced first, like in Fig. 2. The numbers of the packets show the order in which packets arrived at the output interface buffer, and the order in which they leave the interface.

After initiating a VoIP call between T1 and T2, we started *ping* test between PC1 and PC2. Measured round-trip-time (RTT) of the *ping* test was 262.40 ms. When we put the ping traffic to the high priority queue along with VoIP traffic, the ping RTT was 125.43 ms. The RTT test, that is conducted by the router that originates VoIP call, in

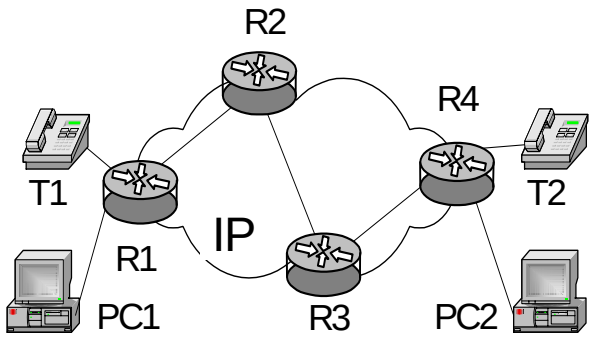

Fig.1. Test network configuration

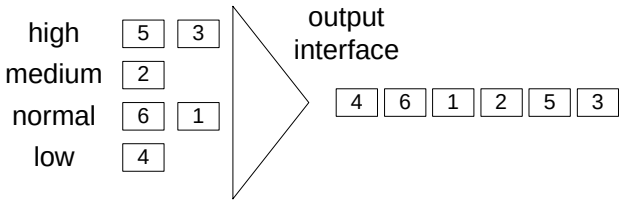

Fig.2. Priority queueing

this case R1, gave a RTT of 129.50 ms. The measurement error e_m caused by wrong treatment of ping packets in the example of priority queuing was

$$e_{m1} = (262.40 - 125.43) / 125.43 = 109.20\ \% \qquad (1)$$

In the first case *ping* test packets are put in the normal queue with all other non-VoIP traffic. This non-VoIP traffic is here represented by TCP traffic we started to simulate network congestion. It is obvious that measurement error e_{m1} is proportional to the amount of traffic that *ping* packet encounters when it arrives in the normal queue buffer. It is sure that this measurement error would be even larger if we started a larger number of TCP connections, but we kept this number constant throughout the test as it was enough to show the amount of the measurement error.

The second test was conducted using LLQ. Fig. 3 shows the basic principles of LLQ.

As mentioned above, in second test we used LLQ on the output interfaces of the routers R1 and R4. The prioritized VoIP class had a 25 kbit/s of bandwidth specified, cause we used the G.729 coder that generates 24 kbit/s of VoIP traffic. The ping packets were matched with all other non-voice traffic in default class, and was served by a fair queue. Measured RTT was 253.45 ms during the initiated VoIP call. After generating a new class for ping traffic, and specifying bandwidth of 24 kbit/s (which is 60 bytes every 20 ms), RTT was 112.07 ms. The measurement error was

$$e_{m2} = (253.45 - 112.07) / 112.07 = 126{,}15\% \qquad (2)$$

The RTT test, which is conducted by the router that originates VoIP call, gave a RTT of 108.00 ms.

These measurements showed that when measuring network performances for special types of QoS-based services the measuring packet probes must be treated from the router

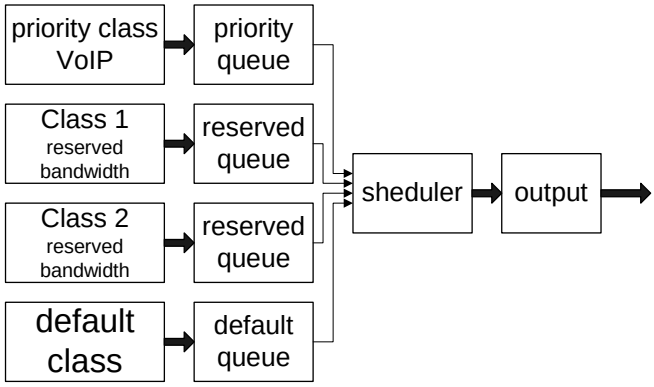

Fig. 3. Low latency queueing

the same way as the traffic of those services. In every other case the results will be highly inaccurate.

3. IMPACT OF OTHER QoS PARAMETERS

In the first chapter of this paper we said that there are also some other QoS techniques that are used for improving network performance. In the case of our test network we used only output interface queueing and packet classification mechanisms. Other QoS parameters that we mentioned in the introduction had no impact because they were not configured on the routers. However, those QoS techniques are highly recommended and used in real-life networks. So, we should take some time and explain what would be their influence on the measurement errors that we observed in our tests.

3.1. Packet fragmentation and interleaving

Even if queueing is working at its best and prioritizing voice traffic, there are times when the priority queue is empty and a packet from another class is serviced. If a priority voice packet arrives in the output queue while these packets are being serviced, the VoIP packet could wait a substantial amount of time before being sent. We described this problem in chapter 2 when VoIP packet waits for 1500-bytes data packet to be sent. This is called a serialization delay, and it should not be larger than 10 ms. Fragmentation is a mechanism that cuts any packet with serialization delay larger than 10 ms into smaller, 10 ms fragments. 10 ms fragments are number of bytes that can be sent over the link in 10 ms. It is obvious that this fragmentation size f_s (in bits) is in direct connection with the link speed l_s:

$$f_s = (0.01s * l_s) \quad (3)$$

Simple fragmentation is insufficient, because if the VoIP packet must wait behind all the fragments of a large data packet, the VoIP packet still will be delayed beyond the end-to-end delay limit. The VoIP packet must be interleaved or inserted in between the data packet fragments. Fig. 4 shows the procedure of fragmentation and interleaving.

3.2. Packet compression

Data compression makes efficient use of bandwidth and increases link throughput by reducing the size of the frame being transported. The type of compression that is used in VoIP networks is called *header compression*. Header compression works in a way that reduces the 40 byte network protocol header to 2 to 4 bytes, there by reducing the bandwidth required per voice call on PPP serial links. The header is compressed at one end of the link and decompressed at the other end.

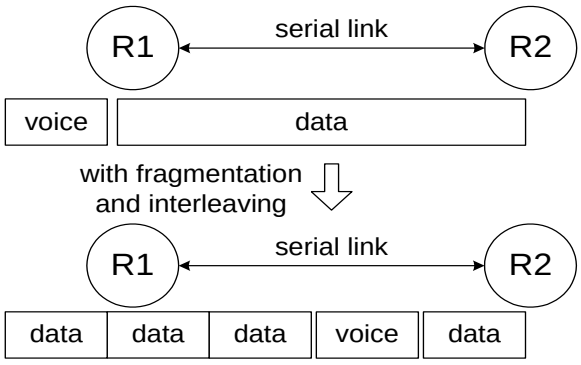

Fig.4. Fragmentation and interleaving

Although both of these QoS techniques have appropriate use in VoIP networks for overall utilization of link bandwidth and making small-packet traffic (e.g. VoIP, telnet,…) more prioritized against large-packet data traffic, they do not have so much influence on the classification of traffic nor on the prioritization of the classified traffic, which are really the main issues of measurement errors when using simulated VoIP probes (like ping packets).

4. CONCLUSIONS

When using active tools for measuring network performances the measurements could potentially be harmed by network elements (e.g. routers) by giving measurement traffic a different priority treatment than observed traffic. In the test examples we can clearly see the effect of the mistreatment of the measurement traffic. The results taken by such measurements that did not take care of the network configuration, in this case queueing techniques on the routers' output interfaces are far from accurate. By applying the same rules that were valid for voice traffic we made one more step forward in making more realistic measurement traffic. The restrictions of these procedures are obvious: all of the actions took place on the network elements. Hence, we had to set the configurations of the routers in order to give the ping traffic the same priority that voice traffic has. Therefore, these procedures can be done strictly by subjects that are in control of the network elements that transport the observed voice traffic (e.g. service providers, companies with local VoIP networks)

These kinds of measurements on high speed links demand bigger accuracy of the ping server application.

REFERENCES

[1] M. Miller, "Voice over IP Technologies: Building the Converged Network", *Hungry Minds*, Indianapolis, IN, 2002.
[2] J. Davidson, "Voice over IP Fundamentals", *Cisco Press*, Indianapolis, IN, 2000.
[3] G. Almes, S. Kalidindi, M. Zekauskas, "A One-way Delay Metric for IPPM", *RFC 2679*, 1999.
[4] S. Keagy, "Integrating Voice and Data Networks", *Cisco Press*, Indianapolis, IN, 2000.
[5] ITU-T Recommendation G.729 "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP)", 1996.

AUTHORS: Ph.D. Petar Knezevic, Faculty of Electrical Engineering and Computing, Unska 3, Zagreb, Croatia, e-mail: petar.knezevic@fer.hr; B.Sc. Vedran Srsen, Mr.Sc. Kresimir Segaric, Coting d.o.o., Unska 2b, Zagreb, Croatia, e-mail: vedran.srsen@coting.hr