Online FPS Games: Effect of Router Buffer and Multiplexing Techniques on Subjective Quality Estimators

Jose Saldana, Julián Fernández-Navajas, José Ruiz-Mas, Eduardo Viruete Navarro, Luis Casadesus

Communication Technologies Group (GTC) – Aragon Inst. of Engineering Research (I3A)

Dpt. IEC. Ada Byron Building. EINA University of Zaragoza

50018 Zaragoza, Spain

Phone +34 976 762 698

Fax +34 976 762 111

{jsaldana, navajas, jruiz, eviruete, luis.casadesus}@unizar.es

Note: A preliminary version of this work was published in "The Effect of Router Buffer Size on Subjective Gaming Quality Estimators based on Delay and Jitter," Proceedings CCNC 2012- 4th IEEE International Workshop on Digital Entertainment, Networked Virtual Environments, and Creative Technology (DENVECT), pp. 502-506, Las Vegas. Jan 2012. Some other parts were published in the article "Influence of Online Games Traffic Multiplexing and Router Buffer on Subjective Quality," in pp. 482-486 of the Proceedings of the same Workshop. This work has been partially financed by CPUFLIPI Project (MICINN TIN2010-17298), MBACToIP Project, of Aragon I+D Agency and Ibercaja Obra Social, and NDCIPI-QQoE Project of Catedra Telefonica, Univ. of Zaragoza.

Abstract: First Person Shooters are a genre of online games in which users demand a high interactivity, because the actions and the movements are very fast. They usually generate high rates of small packets which have to be delivered to the server within a deadline. When the traffic of a number of players shares the same link, these flows can be aggregated in order to save bandwidth. Certain multiplexing techniques are able to merge a number of packets, in a similar way to voice trunking, creating a bundle which is transmitted using a tunnel. In addition, the headers of the original packets can be compressed by means of standard algorithms.

The characteristics of the buffers of the routers which deliver these bundled packets may have a strong influence on the network impairments (mainly delay, jitter and packet loss) which determine the quality of the game. A subjective quality estimator has been used in order to study the mutual influence of the buffer and multiplexing techniques.

Taking into account that there exist buffers which size is measured in terms of bytes, and others measured in packets, both kinds of buffers have been tested, using different sizes. Traces from real game parties have been merged in order to obtain the traffic of 20 simultaneous players sharing the same Internet access. The delay and jitter produced by the buffer of the access router have been obtained using simulations. In general, the quality is expected to be reduced as the background traffic grows, but the results show an anomalous region in which the quality rises with the background traffic amount.

Small buffers present better subjective quality results than bigger ones. When the total traffic amount gets above the available bandwidth, the buffers measured in bytes add to the packets a fixed delay, which grows with buffer size. They present a jitter peak when the offered traffic is roughly the link capacity. On the other hand, buffers which size is measured in packets add a smaller delay, but they increase packet loss for gaming traffic.

The obtained results illustrate the need of knowing the characteristics of the buffer in order to make the correct decision about traffic multiplexing. As a conclusion, it would be interesting for game developers to identify the behaviour of the router buffer so as to adapt the traffic to it.

Keywords*: gaming; delay; multiplexing; compressing; measurement; network games; Quality of Experience; First Person Shooter*

1. Introduction

At the beginning, the Internet was designed as a best-effort network, which means that it delivers information without any delay guarantees. Many years later, it has grown up and it is used to provide new services, sometimes with real-time requirements. One of the first real-time services deployed was VoIP, which is nowadays widely used, and is replacing many traditional telephony systems.

The problem of using a best-effort network for providing a real-time interactive service has been largely discussed for VoIP, as the users of traditional telephony are not likely to change to a new technology unless the offered quality is similar to the one they are used to have. Thus, many studies were carried out, trying to identify the different network impairments, in order to quantify their effect on the quality perceived by users. As a result, some tools have been developed. One of them is ITU's E-Model [1]: first, a battery of surveys is deployed and then a mathematical model is developed. The model takes as input parameters delay, packet loss and other ones such as the codec used, and it provides a quality estimator without having to repeat the surveys each time a new VoIP system is installed.

In the last years some new real-time services have arisen, and among them online gaming has become one of the most popular. Developers release new titles, but some old ones still maintain their popularity. Online games have also been developed for game consoles, thus increasing the number of potential players. Games can be divided into a number of genres. One of them, which has become very popular is MMORPG (Massive Multiplayer Online Role Playing Games). Some titles have millions of subscribers [2], and thousands of them share the same virtual world. These games typically use TCP packets so as to transmit the actions of the player to the server, and to make the player application aware of the movements of the rest of the players.

First Person Shooters (FPS from now) are another game genre. They are played by a smaller number of players (typically some dozens). Every player controls an avatar, which has to accomplish a mission or kill the enemies, using a weapon that can be improved according to the score of the game. Their specific characteristic is that the movements of the players are very fast, so the interactivity requirements are more stringent than the ones for MMORPGs. In this paper we will study them, since they are the ones with the tightest real-time requirements. The network has to be quick enough so as to satisfy the stringent delay requirements of this kind of services.

FPSs use client-server architectures. Due to interactive requirements, clients generate high rates of small UDP packets which are sent to the server. Once the server has calculated the next state, this state is sent to the client application in order to update the state of the players. If delay is not controlled, some undesired effects can appear: "shooting around the corner" [3], unfairness caused by a shot lost in the network [4], etc. Some games have implemented mechanisms in order to tackle these problems [3]: prediction, buffering or time distortion are some of them.

We will use the term KPI (Key Performance Indicator) referring to the network parameters that define the quality of a connection. They are mainly delay, packet loss and jitter. Players are usually behind residential Internet access networks, which may have a low speed at the uplink. If background traffic is present, the uplink can become a bottleneck between the LAN and the backbone network, and the main contribution to the delay can then be caused by the router buffer [4]. But there is another problem, highlighted in [5] and [6], related to the amount of packets per second (pps from now): some routers were designed for managing big packets generated by traditional applications such as web browsing, e-mail or P2P, but they may have problems when dealing with high rates of small packets, because of processing capacity limitations. As residential accesses use mid or low-end routers, this implies some limitations for traffic delivery.

The characteristics of real-time traffic, i.e. high rates of tiny packets, make it suitable for multiplexing. This solution, which was first used for VoIP, consists of merging a number of small packets into a bigger one. Individual packets' headers are compressed, and a common header is added to the multiplexed bundle. But in order not to add undesired delays, this is only useful when the traffic of a number of users shares the same link. The utility of these techniques for FPS games in some scenarios, as Internet cafés or the network of a game provider, where many players share the same access, was explained in [7].

On the one hand, multiplexing mitigates two problems: first, it reduces the overhead, thus improving the efficiency of real-time flows. These services usually generate small packets including a number of headers which may represent a high percentage of its size. For example, an RTP VoIP packet includes 40 bytes corresponding to IP/UDP/RTP headers, or a client-to-server packet of an FPS game includes 28 bytes of IP/UDP headers. As the length of the payload is about some tens of bytes, the efficiency is very low. The other problem that can be solved by multiplexing is the router limitation in pps. Logically, if a number of packets are included into a bigger one, the total amount of packets is reduced. On the other hand, multiplexing modifies KPIs: since a number of packets have to be multiplexed, a new delay will be added, corresponding to the time the packets are retained in the multiplexer. An additional jitter will also appear, since some packets will have to wait more than others in order to be sent by the multiplexer.

In order to blend the effects of the different KPIs, some studies [4], [8], [9] have made an effort in order to deploy quality estimators similar to the E-Model, but adapted to a specific game. As we will see in the next section, the first step they take is to identify the network parameters that mainly impair the experienced quality, which are then properly combined so as to obtain a MOS (Mean Opinion Score) formula. Some of these studies have found that delay and jitter are the most important parameters, instead of considering delay and packet loss as the main KPIs.

This fact is very interesting and has some implications related to the scenario where players are, i.e. they are usually connected to the Internet via an access network with a limited bandwidth and a low or mid-range router. These routers may present different behavior depending on access bandwidth and the implementation of the output buffer [10].

In this paper we also study the combined effects of the buffer of the access router, and traffic multiplexing, when using quality estimators based on delay and jitter. On the one hand, multiplexing saves bandwidth, and this will imply a global reduction of the delay and the jitter if the buffer is busy. On the other hand, multiplexing techniques add delay and jitter, mainly caused by retention time at the multiplexer. As a consequence, an interesting trade-off arises, which will be studied in the paper.

The paper is organized as follows: section 2 discusses the related work about quality estimators for online games, online games traffic multiplexing and buffer sizing. The methodology for the tests is explained in section 3. The next section covers the tests and results, and the paper ends with the conclusions.

2. Related Work

Different related topics have to be considered in this study. First, some works related to the subjective quality evaluation for online games will be summarized. Second, we will consider the multiplexing of real-time flows, more concretely from online games. Finally, the influence of the buffer of the access router has to be considered, as it plays a central role in the considered scenario.

2.1. Subjective Quality Estimation

The E-Model [1] is a widely accepted method which has proven useful for designing and dimensioning VoIP systems. It defines some impairment factors which are finally combined to obtain a MOS estimation formula. Although different network impairments are somewhat related (e.g., a packet with extremely high delay is equivalent to a loss packet), these estimators consider them separately. The main network KPIs that have an influence on the calculation are delay and packet loss. Jitter is not directly considered, as de-jitter buffers are usually implemented in VoIP applications. They add a delay and also discard packets that arrive too late to be correctly played.

Regarding online games, the problem of obtaining estimators for the subjective quality has been issued in many works. In [11] only the delay was considered, and it was denoted as SRT (System Response Time), which was defined as the time necessary to detect a user event, to process it, and to display the new game state at the output device of the user. A simple MOS estimation formula was then calculated, which had a linear dependence on the SRT.

In [12] the authors studied the different delay upper bounds reported in the literature, which are between 150 and 250 ms. A set of experiments was carried out, using an emulator so as to add controlled delays and packet losses, and asking real players to fill in some questionnaires regarding their perceived quality. Two FPS games were tested, obtaining a similar behaviour with respect to delay, but the effect of packet loss was very different: while *Halo* stopped working when packet loss was around 4%, the users of *Quake III* did not experience a quality degradation even with packet loss of about 35%. This means that this game has a very good packet loss concealment system. The work separately studied the effect of delay and loss, so a MOS estimation formula was not developed. Some of the conclusions of this work were that delay has a bigger influence than loss. The study did not consider the jitter as a KPI, as the authors considered that its effect on perceived quality was significantly smaller than the latency impairment.

In [13] a survey was conducted and also a practical evaluation of four games: two FPSs, a sports game and an RTS (Real Time Strategy) game. Only delay and jitter were considered as KPIs, while packet loss was left for future work. The results showed very different MOS impairments of the KPI for each game.

The first MOS model for an FPS game (*Quake IV*), adapted from the VoIP E-Model, was presented in [8]. It is based on delay and jitter, which is measured as the standard deviation of the delay. Packet loss is not considered, taking into account the very good packet loss concealment algorithm that the game includes, which hides this bad network effect to the players. A polynomial formula for MOS estimation was obtained using multi-dimensional regression analysis. This is the model we are going to use in the current work.

Ref. [4] presented a formula which included weighting factors for delay, jitter and packet loss. These factors are different for each game. However, since this work was carried out inside a company, these parameters are not publicly available.

A similar analysis was carried out in [9] and a formula for estimating the MOS of an MMORPG was developed (*World of Warcraft*). Different combinations of loss and jitter were added using an emulator, and a group of users were invited to play the game and fill in some questionnaires regarding the perceived quality.

Finally, some works have studied the relationship between subjective quality and multiplexing, mainly for VoIP. Ref. [14] presented a multiplexing method based on the E-Model for reducing overhead when using IPsec. Also in [15] our group compared the conversation quality obtained when using native RTP, TCRTP (*Tunneling Compressed RTP*, RFC 4170) [16] and the multiplexing proposal of [17]. Bandwidth saving was reported as an important factor, but another conclusion was that packet size has to be taken into account, as some buffers penalize big packets, i.e. packet loss probability increases with packet size.

2.2. Multiplexing Real-Time Flows

The IETF defined TCRTP [16] as a combination of protocols in order to multiplex a number of RTP flows. This standard, defined in 2005, has three layers: First, a header compression algorithm is applied: the IETF has defined a number of algorithms in order to reduce the headers of packets belonging to the same flow. When a real-time session is established between two hosts, the packets of the flow have the same value in a number of header fields: i.e. the origin and destination IP addresses, source and destination ports, etc. This fact can be used in order to avoid the sending of some of the fields: in a hop-by-hop way, each origin and destination may store the value of the repeated fields, known as the *context*, and this allows significant reductions of the header size. The used algorithm for RTP compression is called *Enhanced Compressed RTP* [18].

The second layer is multiplexing, which is deployed by PPPMux [19], a protocol capable of multiplexing a number of different packets into a single one. And thirdly, if a tunnelling scheme is also used, then the header compression can be used end to end. Therefore, the standard also includes an L2TPv3 tunnelling scheme for the implementation of this layer.

Other non-standard proposals for multiplexing RTP flows are [17] and [20]. They were mainly designed for VoIP, since this is a very popular real-time service which presents a high overhead.

In [21] our group first proposed TCM (*Tunneling, Compressing and Multiplexing*), which is an adaptation of the TCRTP scheme to the traffic of non-RTP services, such as FPS games. It would permit the use of different compression algorithms, since these games do not send RTP packets. This proposal has been improved and proposed to the IETF [22]. We will use this method in the current paper, in order to study the effect of the buffer behaviour on the different KPIs.

2.3. The Influence of the Router Buffer

In the last years, many studies related to the buffer size problem have been deployed. A very complete review of the published works can be found in [23]. The problem is normally considered for backbone routers managing a number of TCP flows. As a summary, we can say that the traditional *rule of thumb* used to calculate the buffer size was the bandwidth-delay product (BDP). In 2004, Appenzeller et al. deployed the so-called *Stanford model* [24], which proposed the use of *small* buffers, calculating their size as the quotient of BDP and the square root of the number of TCP flows. A tendency to reduce the size of the buffer can be observed, and some works [25] propose the use of *tiny* buffers, considering that a capacity of some dozens of packets, or some tens of kB, is enough so as to obtain link utilization of about 80-90%. Finally, some works [26], [27] consider the combined effect of TCP and UDP on *tiny* buffers, and an anomalous region where packet loss grows with buffer size was found. The problem of the coexistence of UDP and TCP flows was also addressed in [28]. In these works, the idea of reducing the size of the buffer was also reported as adequate for real-time UDP flows.

In the present work we will not consider backbone routers, as in the considered scenario the access to the Internet is usually provided by mid or even low-end routers, which are smaller and simpler. Different buffers will be tested and compared in order to study the influence of buffer size and implementation on the performance of real-time services. We will consider drop-tail FIFO *tiny* buffers of some tens of kB, or a small number of packets, as bigger ones would increase the delay, making the experienced quality of real-time services unacceptable.

This topic is related to the two other ones that have been issued in this Section, i.e. the behaviour of the router buffer has a big influence on experienced quality, and of course multiplexing can strongly modify the traffic. Our group has studied the relationship between these three problems in some previous works: in [15] we compared the behaviour of VoIP multiplexing when using different buffers. In [7], the influence of multiplexing techniques on the traffic of a game was also studied, in terms of delay and packet loss, using a

high capacity buffer and also a time-limited one. The buffer size was found to be a very important parameter which has to be taken into account in order to select a correct value for the multiplexing period.

The present work studies in more depth these relationships, i.e. the mutual influence of multiplexing schemes and the buffer of the router. But the results will not only be based on objective network parameters (delay, packet loss and jitter), but subjective quality estimators will also be used, in order to get a better idea of the user's perspective of the quality of the game. The study is centred on the impairments for real-time traffic when it shares a buffer with other traffic flows.

3. Test Methodology

In this section we will consider a number of topics that are necessary in order to understand the deployed tests. First, we will present the scenarios considered in this study. Next, we will briefly comment on some previous results obtained using VoIP, another real-time application which traffic has some similarities with the one of FPS games. Then, the traffic of the selected game and the background traffic will be described. Next, we will describe the different characteristics to consider in the router buffer and the formula used to estimate the MOS. Then, some specific considerations for multiplexing tests are detailed, and the section concludes with a detailed explain of the simulations.

3.1. Scenarios of Interest

The scenario we are considering can be seen in Fig. 1. A number of hosts are connected to a server via the same access router. If we consider gaming service, we can find this scenario in different networks: one of them is the network of an Internet café. Gaming has been reported as one of the main activities deployed by the users of these businesses [29], which are still a very popular way for connecting to the Internet [30] [31]. This scenario was more deeply studied in [32]. Another network that fits this scenario is a game provider using proxies so as to improve the quality experienced by the users [21]. The main difference is that a network provider may use high-end routers. Another example can be an access networks based on WiMAX: these wireless technologies can be used to provide Internet access to groups of people (e.g. a rural zone), if a wireless link has a smaller cost than a wired one. In these cases, the traffic of many players can share the wireless link, and bandwidth savings can be interesting [33].

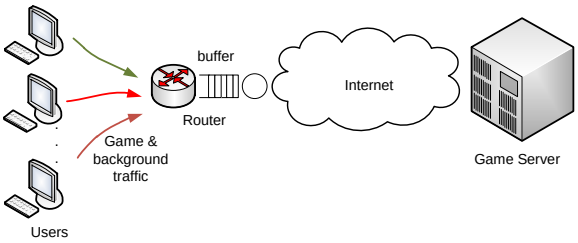


Figure 1. Scenario considered.

In this scenario, the Internet access will be shared by many services, some of them real-time. The total traffic offered to the router may vary and in some moments it can even be over the link capacity. In these moments, the router will add delays and discard packets, and the buffer size and policy will have a strong influence on the distribution of these impairments. In particular, some policies are packet-size aware, as they penalize different packet sizes in different manners.

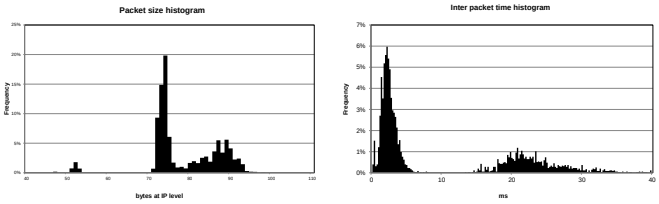
3.2. Previous Results using VoIP

In previous works, our group studied the influence of the buffer size on another real-time service, namely VoIP [34], and the results showed that the MOS has a monotonically decreasing behavior as the background traffic grows. In that case, jitter was not considered as a KPI, as a de-jitter buffer is used, which adds some delay and discards packets that arrive out of time. This can be done as the codec and consequently the packet rate, are known. In that study the VoIP native traffic showed a good behavior when using a small buffer, as small packets have less probability of being discarded than big ones. Acceptable MOS results were obtained even when the offered traffic was over the bandwidth limit. In VoIP the MOS is normally considered acceptable when it is over 3.5.

3.3. Traffic Details

Taking into account that online games are not normally open applications, we do not know whether they use a de-jitter buffer or not. As we have seen, delay and jitter are the KPIs considered to affect certain games, so in this study we will measure their effect. We will consider the same scenario used for VoIP service, i.e. we have a number of players sharing the access network.

In order to test the most stringent real-time applications, we have selected an FPS game, concretely one for which a MOS model exists [8]: *Quake IV*. First, we needed traffic traces of the game under study. We obtained them from the CAIA project [35], where many online games have been analyzed. The traces were obtained from real parties in controlled conditions, and are very well documented. They contain a number of packets which is the product of 5,000 and the number of players. Fig. 2 shows the histograms for packet size and inter-packet time for the selected game. The average size of the packets at IP level is 79.5 bytes. The game generates 64 packets per second average. So the total bandwidth generated by each player is roughly 40.7 kbps. The number of players considered in the tests is 20, so the total game traffic is 814 kbps.



(a) (b)

Figure 2. Packet size (a) and inter-packet time (b) histograms for *Quake IV*.

For obtaining each point of the results graphs, presented in the next section, 400 seconds of the application traffic have been sent, sharing the buffer with different amounts of background traffic. A background traffic distribution with Poisson inter-packet times and three packet sizes is used: 50% of the packets are of 40 bytes at IP level (header plus payload), 10% have 576 bytes and the rest are of 1,500 bytes [36].

We will study the most stringent link, which in our case is the uplink, as many access technologies are asymmetric and the bandwidth of the uplink is significantly smaller than the one for the downlink. We have first isolated client-to-server traffic from the traces under study. Then, traces including different numbers of players have been combined it in order to obtain the desired number of users, as done in [32].

3.4. Used Buffers

We will consider an Internet access with uplink bandwidth of 2 Mbps and 3 Mbps. Two buffer implementations are used: one defines its size in bytes (*byte-sized*) and the other in packets (*packet-sized*). In fact, some manufacturers measure the buffer size in bytes and others do it in packets. For example, in [37] the routers of two manufacturers are compared, and one of them gives this information in packets, while the other gives it in milliseconds, which is equivalent to bytes, as the two parameters are related by the line speed. We will consider drop-tail FIFO buffers, as they are the most common ones in low or mid-end routers, which are the ones we will find in the scenario. Different sizes will be also considered for each implementation: 10, 20, 50 and 100 kB for *byte-sized* buffers, and 16, 33, 83 and 166 packets for *packet-sized* ones. The sizes are roughly equivalent, considering average packet size of 600 bytes. They can be considered as *tiny* buffers [23]. Some limitations of buffers will also be considered in some tests: i.e. the processing capacity of the router may imply a limitation on the number of pps it is able to manage, and this will have some consequences when real-time and background traffic share the queue.

3.5. MOS Estimation

In order to present the results of the estimation of the perceived quality, we have selected a MOS model from the literature. The formula proposed in that model [8] first calculates a *network impairment* parameter named *X*, which depends on the value in milliseconds of RTT (*ping_average)* and *jitter_average* of the arrived packets:

$$X = 0.104 * ping\_average + jitter\_average \qquad (1)$$

The model was obtained using LAN parties, to which different amounts of delay (from 0 to 320 ms) and jitter (from 0 to 160 ms) were added using a network delay emulator. Thus, the model considers as negligible the value of the delay and jitter which can be present in a LAN. The jitter is measured as the standard deviation of the delay. Jitter is measured in different ways in the literature, and there is not a clear consensus on its definition. In VoIP it can be measured as inter-packet delay variation but, in the case of the considered game, it cannot be measured that way, as inter-packet time is not fixed (Fig. 2b).

The MOS estimation formula depends on *X:*

$$MOS = -0.00000587\ X^3 + 0.00139\ X^2 - 0.114\ X + 4.37 \qquad (2)$$

As said in Section 2, packet loss under a threshold is not considered to significantly affect the quality, as this game has a very good packet loss concealment algorithm. This also happened in *Quake III*, which surprisingly, could work properly even with packet loss near to 35% [12]. These games implement different techniques for compensating network impairments (e.g., prediction, interpolation, time distortion, etc.), as studied in [3]. Although some studies use the VoIP scale and consider MOS acceptable when it is above 3.5, some others consider that a value of 3 can be good enough, and that gamers will try to find another server when MOS is roughly 2 [13].

3.6. Specific Details of Multiplexing Tests

Some of the tests consider multiplexing of game traffic. In this subsection we will explain the specific issues that have to be considered in this case. A multiplexer is added to the local network (Fig. 3a), being able to multiplex the traffic of the users using TCM, as explained in [32]. This feature can be included in a specific device, and may also be deployed by the computer of a player, or may even be embedded in the router.

As we have seen, we have to obtain the total RTT, which will then be introduced as the *ping_average* parameter in the quality model. This delay has three components:

- Multiplexing delay: as we are only multiplexing the traffic for the uplink, we will only have $delay_{mux}$ for the traffic that goes from the client to the server. It includes the retention time plus the processing time in the multiplexer. Since the system only multiplexes client-to-server traffic, there is no demultiplexing delay when the traffic arrives to the client.
- Queuing delay in the router ($delay_{router}$): it will only be present in the uplink, since we are passing from a fast LAN to a slower access network. Queuing delay at the uplink router buffer can be significant, as seen in the introduction, if background traffic is present. In the downlink packets pass from a slower to a faster network, so queuing delay can be considered negligible.
- Network round trip time ($delay_{network}$): it includes the network delay, plus processing times in the demultiplexer and the game server.

A period *PE* is defined in the multiplexer in order to periodically send a packet including all the native game packets which have arrived. As can be seen in Fig. 3b, this will add a delay, and also a jitter, since the added delay will be different depending on the moment in which the packet is received: if it arrives at the beginning of a period, it will be delayed much more than if it arrives at the end.

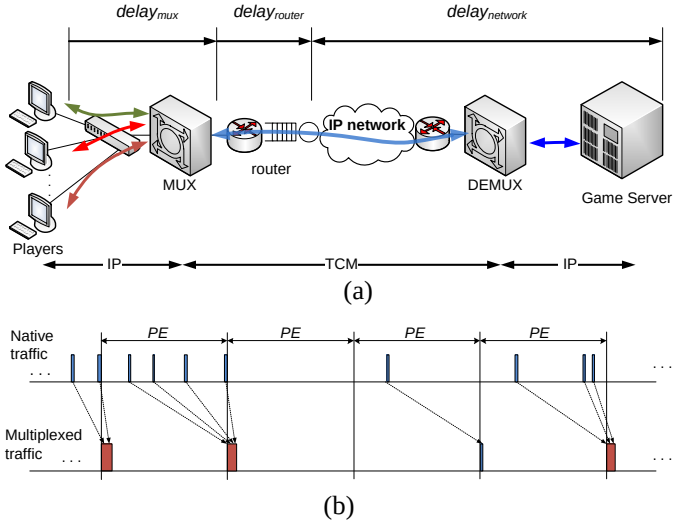


Figure 3. Scenario used in the multiplexing tests (a) and multiplexing method (b).

Logically, if a big value is set for the period, bandwidth saving will increase but, on the other hand, retention time will grow. This fact implies a series of trade-offs: if *PE* is increased, bandwidth is reduced, so the buffer will discard fewer packets. But on the other hand, retention time will grow, and so will do the jitter. But there is still another factor which has to be taken into account: packet size. Multiplexed packets are bigger than native ones and this will have an influence, depending on the implementation of the router buffer. Finally, the reduction in terms of pps can also have an influence in some cases.

Multiplexing delay will be half the period PE on average, considering that packet arrivals will be distributed during the period. In order to calculate the standard deviation, we will make the simplification of considering packet arrivals as uniformly distributed during the period PE, taking into account that the number of traffic flows to multiplex is big, and that inter packet time is variable (Fig. 2b). The variance of a uniformly distributed variable in an interval *(a,b)* is $(b\text{-}a)^2/12$. As the added delay varies from 0 to *PE*, the standard deviation of the delay added at the multiplexer will be:

$$\sigma_{delay\_mux} = PE/\sqrt{12} \quad (3)$$

Root-mean-square can be used to calculate the addition of standard deviations of uncorrelated variables. In order to obtain the total standard deviation of the delay, we have to take into account that σ_{delay_mux} and σ_{delay_router} are correlated, since the variation of *PE* will modify the total traffic offered to the router, and consequently the delay added by the buffer. So the standard deviation has been calculated as the difference between packet arrival time at the multiplexer and departure time from the router buffer.

$$\sigma_{delay\_mux\_router} = \sqrt{\sigma_{delay\_mux}^{\ 2} + \sigma_{delay\_router}^{\ 2} + \mathrm{cov}_{m+r}} \quad (4)$$

We have observed that $\sigma_{delay\_mux\_router}$ is smaller than the root-mean-square of σ_{delay_mux} and σ_{delay_router} so this implies that the covariance (cov_{m+r}) of $delay_{mux}$ and $delay_{router}$ is negative. This happens when higher than average values of a variable correspond to lower than average values of the other variable. In our case, this could be expected: when we use higher values of *PE*, the variance at the multiplexer grows, but the total traffic offered to the router is reduced, so the variance of queuing delay will be reduced as well.

We have considered that network delay is independent of the two other ones. Network delay with an average of 30 ms and a variance of 5 has been added. So the total jitter has been calculated as:

$$\sigma_{delay\_total} = \sqrt{\sigma_{delay\_mux\_router}^{\ 2} + \sigma_{delay\_network}^{\ 2}} \quad (5)$$

3.7. Simulation Details

A Matlab-based simulation has been used in order to introduce the obtained game traces and the background traffic in the same drop-tail buffer. Delay is measured as round trip time. Jitter is the standard deviation of the delay. The delay is composed by the buffer delay plus an additional lognormal RTT (Round Trip Time) of 30 ms with a variance of 5, as suggested in the model of [38]. This can be a typical network delay for a regional scenario [39], and the response time of the game server. The acceptable delay boundary for certain FPS games has been reported to be around 200 or 225 ms [12].

Fig. 4 shows the scheme of the Matlab simulations. First, traffic traces are separated in order to be Tunneled, Compressed and Multiplexed with TCM. At the same time, background traffic traces are generated using a statistical distribution, as commented above. The traces are then merged, obtaining the packet sizes and arrival times at the input of the buffer. Next, the buffer behaviour is simulated: the application takes the packets one by one, inserts them in the buffer if possible and, at the same time, packets are sent according to the router output rate, taking into account the different parameters and limitations (mainly rate and pps). An output trace file is obtained, which adds the corresponding delay to each packet that has been accepted. Finally, the results are calculated from this trace, taking into account the packets that have been accepted or dropped.

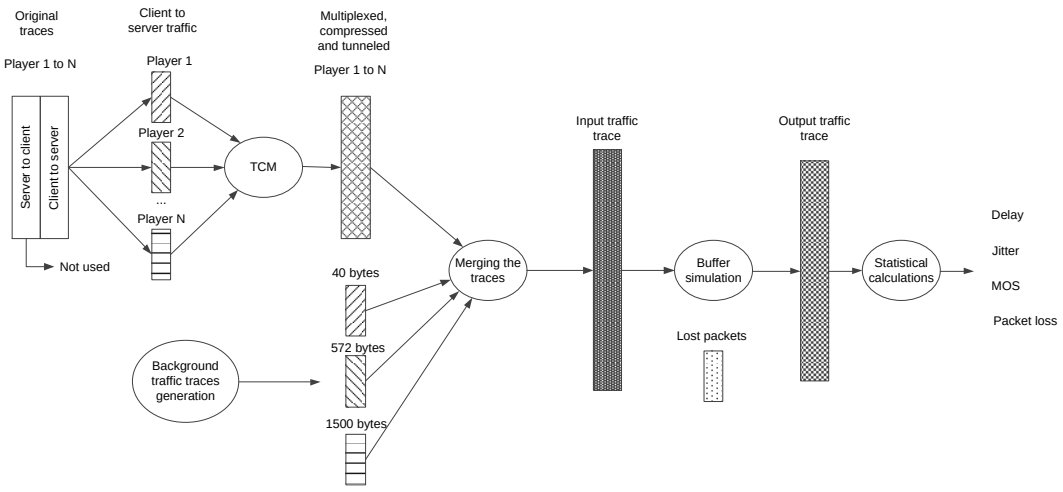


Figure 4. Scheme of Matlab simulations.

4. Tests and Results

In this section, a comparative study of the buffer effects on the different KPIs will be performed. In each subsection we will first present RTT, jitter and MOS for each option, taking into account that the MOS depends on the other two KPIs. The graphs for packet loss will be presented separately, since the MOS does not directly depend on them, as it has been previously explained. According to Section 3.4, two different buffer implementations (*byte-sized* and *packet-sized*) will be compared. To improve the clarity of the graphs, we have decided not to use the same scale in the Y-axis of the RTT and jitter results for different buffers, so the reader has to be aware of this.

4.1. Buffer Size and Buffer Implementation Comparative Study

The first study presents the results of a 2 Mbps connection, using *byte-sized* (Fig. 5.a) and *packet-sized* (Fig. 5.b) buffer implementations. It can be observed that the X-axis ranges from 0 to 3 Mbps of background traffic. Although the bandwidth limit is 2 Mbps, we have used that range in the graphs because some interesting effects can be observed. This also allows us to better compare the graphs with subsequent results (Fig. 7 and 8).

First, it can be seen that when the bandwidth limit is reached, which happens when background traffic is roughly 1,200 kbps, the delay grows up depending on the buffer size and implementation. In this case, the smaller the buffer, the smaller the added delay. Regarding *byte-sized* buffers (Fig. 5.1.a), it can be observed that, when total traffic is above bandwidth limit, the added delay remains constant even if the total offered traffic grows. The cause for this is that the buffer is always almost full, so all the accepted packets experience the same delay, which is equivalent to the quotient of the buffer size and the bandwidth. We can see that the *100 kB* buffer cannot be used if the total offered traffic is above the bandwidth limit, because it would add a very big delay. On the other hand, the buffer of *50 kB* is near the limit of acceptable values (220 ms), which means that the buffer size limit for acceptable delays is roughly this value.

If the buffer is *packet-sized* (Fig. 5.1.b), we can see that the added delays are smaller, and grow more slowly than the ones of the other buffer. In addition, the delay continuously grows with the increase of background traffic. The reason for this is that the average size of the offered traffic grows with background traffic, as the game traffic remains the same and the average size of background packets is around 675 bytes. As packet loss is the same for every packet size (Fig. 6 b), the outgoing traffic size distribution is the same as the one for offered traffic. So in conclusion, the delay will be increased, since the average size of outgoing packets grows with the increase of the background traffic amount.

Fig. 5.2 a) shows the jitter for the *byte-sized* buffer. For the smallest buffer, it shows a good behaviour but, if bigger buffers are used, a peak appears when the offered traffic is roughly the bandwidth limit. This is caused by the saturation of the access link, which makes the buffer occupation grow. But if the offered traffic is above the bandwidth limit, then the jitter gets reduced, because the buffer is always full, so all the packets will experience the same delay.

If the *packet-sized* buffer is used (Fig. 5.2b), we see that the jitter peak is smaller: as the delay is smaller, its standard deviation also gets reduced. On the other hand, it does not fall down after the bandwidth limit, as it happened with the other buffer, as the delay keeps on growing.

Fig. 5.3 presents the MOS, estimated by means of Eq. (2). The graphs show a surprising behaviour when compared to the MOS of VoIP, which has a monotonically decreasing behaviour [34]. When the background traffic is small, the graph is as it could be expected: the bigger the background traffic, the worse the experienced quality. And when we are near the bandwidth limit, the graphs become worse, due to the jitter increase. But surprisingly, when the offered traffic goes above the bandwidth limit, the experienced quality grows up as the background traffic increases, achieving better values. This counter-intuitive "valley" behaviour is caused by the jitter reduction shown in Fig. 5.2. As we have said, the jitter is significantly reduced because the buffer is always full, so the delay is roughly constant for all the accepted packets.

If we consider acceptable MOS values around 3, in Fig. 5.3.a it can be seen that only the smallest buffer (*10 kB*) achieves acceptable MOS results despite the offered traffic. The rest of buffer sizes maintain acceptable MOS levels until the total offered traffic is around 90% of the bandwidth limit.

In Fig. 5.3.b, it can be seen that the smallest buffers show a monotonic decrease with background traffic, caused by the increase of delay and jitter above the bandwidth limit. In contrast, the biggest buffers present a "valley" behavior.

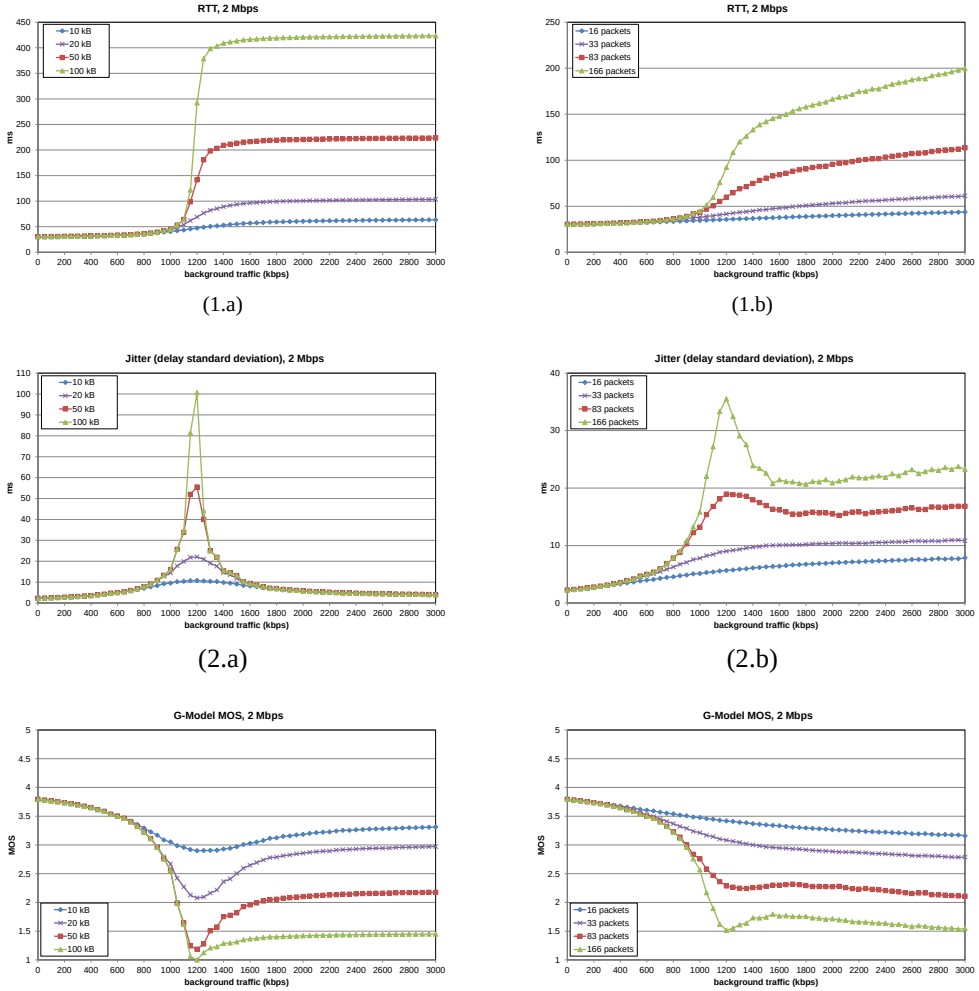


Figure 5. RTT (1), jitter (2) and MOS (3) as a function of background traffic, for *byte-sized* (a) and *packet-sized* (b) buffers with 2Mbps bandwidth

Fig. 6 shows packet loss for each packet flow, using the *10 kB* and *16 packet* buffers. For the *byte-sized* buffer (a), it can be seen that small packets have a clear advantage, as the probability of not having enough place at the queue increases with size. This is good for game traffic, as packets are small. On the other hand, if the *packet-sized* buffer is used (b), the loss probability is independent of packet size, as it can be seen in the figure. In this case, the traffic of the game does not have any advantage because of its small packet size. In the graph we observe that traffic over the bandwidth limit produces a packet loss rate over 35%, so this would make MOS be reduced for this buffer, although the G-model formula does not include the effect of packet loss.

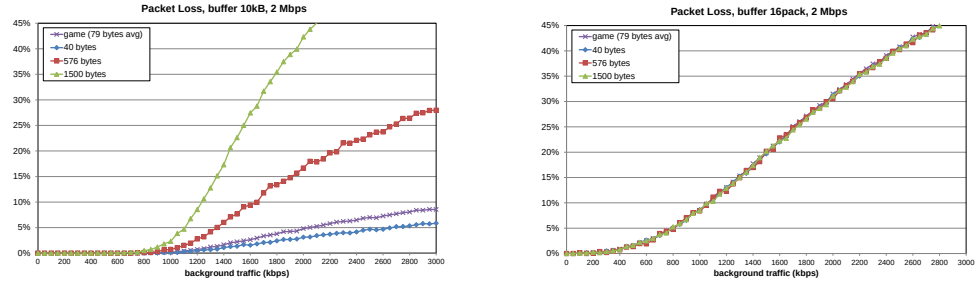


Figure 6. Packet loss as a function of background traffic, for *byte-sized* (a) and *packet-sized* (b) buffers with 2Mbps bandwidth

4.2. Effect of Bandwidth Increase

This subsection presents the results using a bandwidth of 3 Mbps. Regarding RTT, Fig. 7.1.a) has a similar behavior as the one of 2 Mbps (Fig. 5.1.a), with delay growing up when the bandwidth limit is reached. But in

this case the higher bandwidth limit makes the delay become smaller. So in this case, with a buffer of *50 kB* we will obtain acceptable delay values (around 150 ms). A similar reduction can be observed when comparing Fig. 7.1.b) and 5.1.b).

If we compare the jitter in Fig. 5.2.a) and Fig. 7.2.a), we can see that the peak for the *100 kB* graph has been reduced from 100 to 70 ms, and this decrease can be observed for the rest of buffer sizes. This means that there is a relationship between the bandwidth limit and the maximum buffer size allowed: the bigger the bandwidth of the access network, the bigger the buffer allowed. A significant reduction is also observed for the *packet-sized* buffer (Fig 7.2.b).

Finally, we see that the MOS achieves better results if bandwidth is increased (Fig 7.3). For the *20 kB* buffer, if the offered traffic grows above the bandwidth limit, acceptable MOS values can be reached again. This effect is caused by the decrease of the jitter.

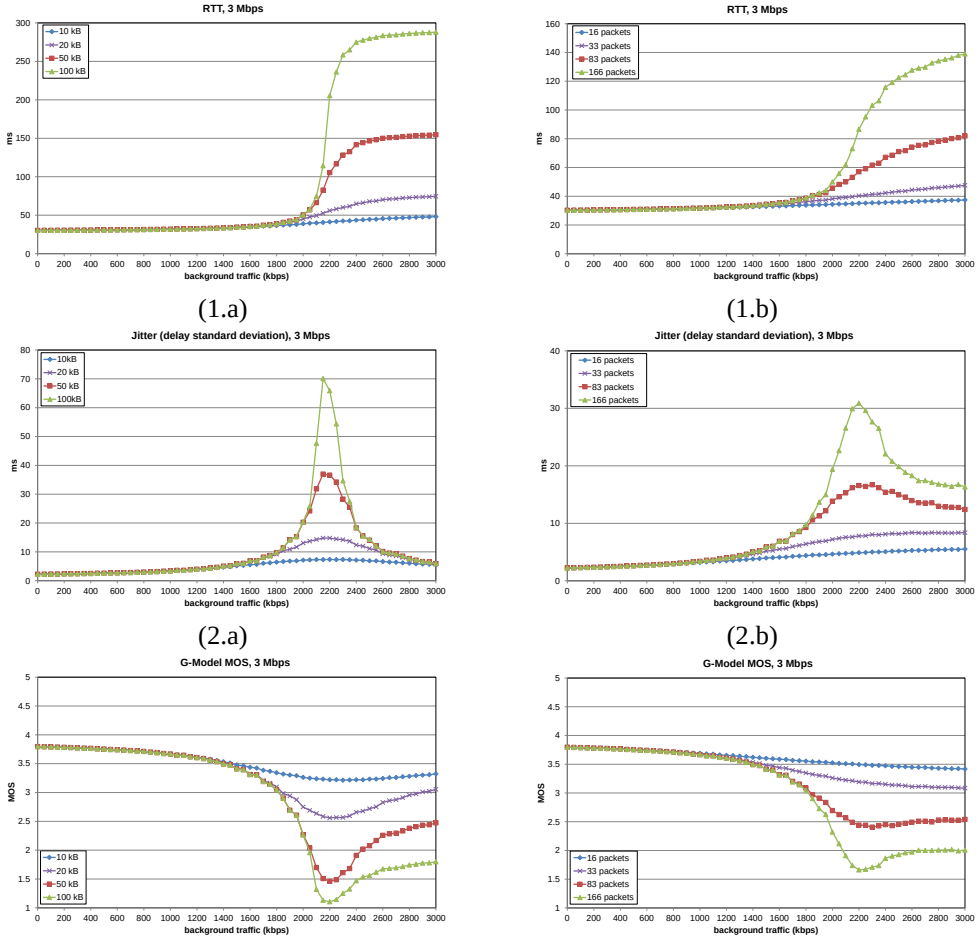


Figure 7. RTT (1), jitter (2) and MOS (3) as a function of background traffic, for *byte-sized* (a) and *packet-sized* (b) buffers with 3 Mbps bandwidth

Fig. 8 presents packet loss for the different traffic types. The same effects observed in Fig. 6 can be seen. The only difference is that, for the same buffer, the packet loss percentage is smaller, as bandwidth is bigger.

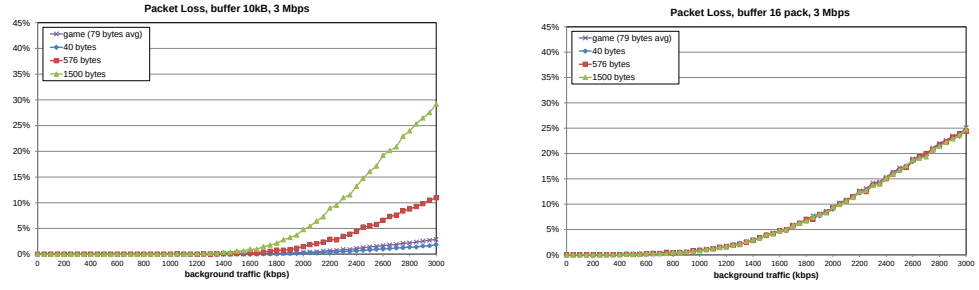


Figure 8. Packet loss as a function of background traffic, for *byte-sized* (a) and *packet-sized* (b) buffers with 3 Mbps bandwidth

4.3. Effect of Multiplexing

In this subsection, measurements have been carried out using native traffic traces, and also with multiplexed ones with *PE*=5 ms and 15 ms. On behalf of clarity, only the biggest and smallest buffers for each implementation have been used (*10 kB, 100 kB, 16 packets* and *166 packets*). In this subsection the bandwidth of the Internet access is 2 Mbps.

Fig. 9 shows RTT, jitter and MOS for each buffer. First, we can observe that, when the link capacity is exceeded, the delay grows (Fig. 9.1). The behaviour is similar to the one observed in Fig. 5. For *small* buffers, it does not grow to unacceptable values, but for *big* ones we observe that the delay grows dramatically, and also a peak appears in the jitter graphs (Fig. 9.2). From the figures, we can observe the consequences of the trade-off: on the one hand, multiplexing adds a small fixed delay mainly caused by retention time at the multiplexer, which makes the total delay be slightly bigger when background traffic is small. On the other hand, if we use multiplexing, bandwidth saving is about 200 kbps, so the delay growth appears when background traffic is 1,400 kbps instead of 1,200 kbps. This saving could become more significant if a narrower buffer was used. The jitter goes down again after the peak because the buffer is always full, so buffering delay will experience very little variations. Fig. 9.2.b. shows a smaller peak for the native traffic jitter than Fig. 9.2.a., because in this case the average packet size is smaller, as a big number of native packets are present, so queuing delay will also be smaller.

Fig. 9.3 shows the MOS. For *small* buffers, the graphs first go down because of the jitter peak, and after that they grow a little. For *10 kB* buffer the difference is small, whereas for *16 packets* buffer multiplexing is worse than native traffic. When using *big* buffers, the graphs go down later for multiplexed traffic, so it can support a bigger amount of background traffic, because of bandwidth saving.

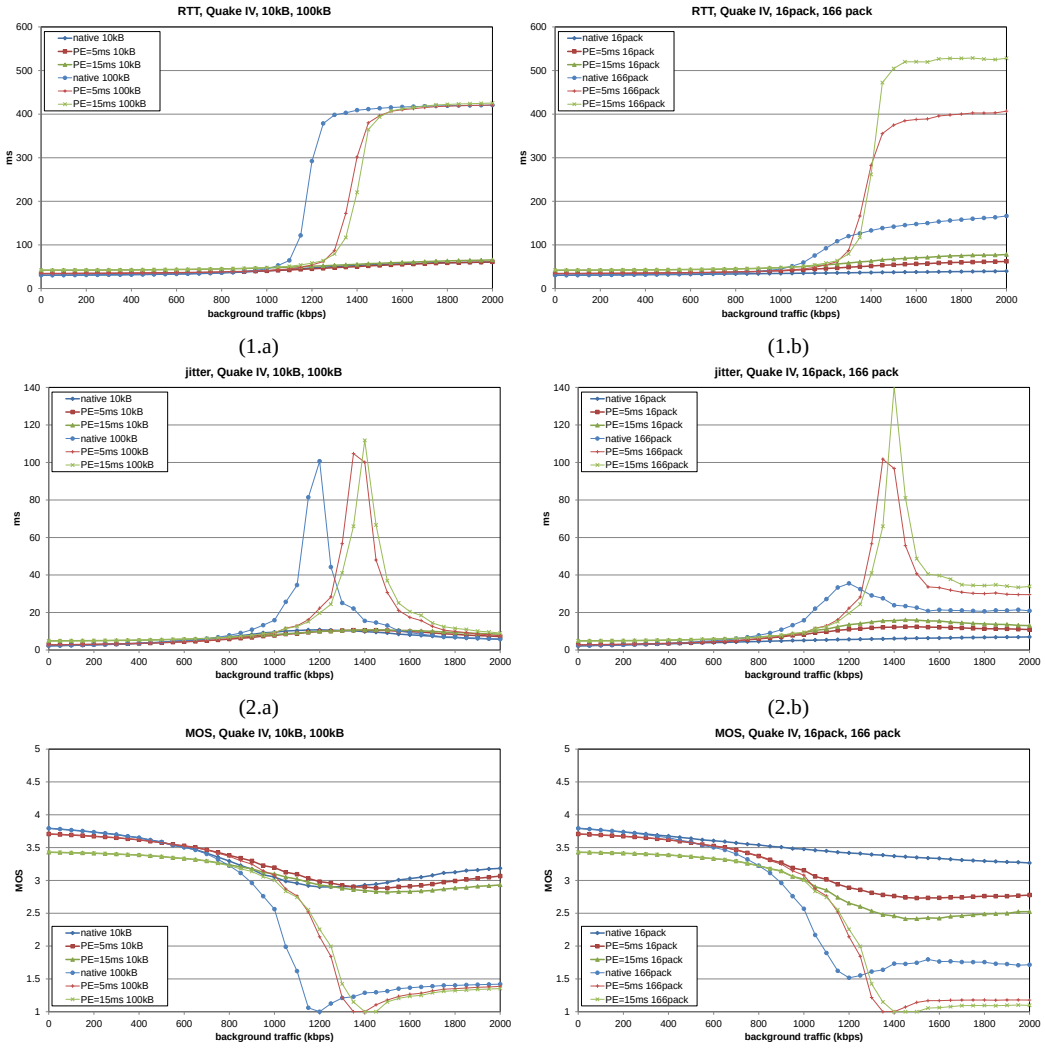
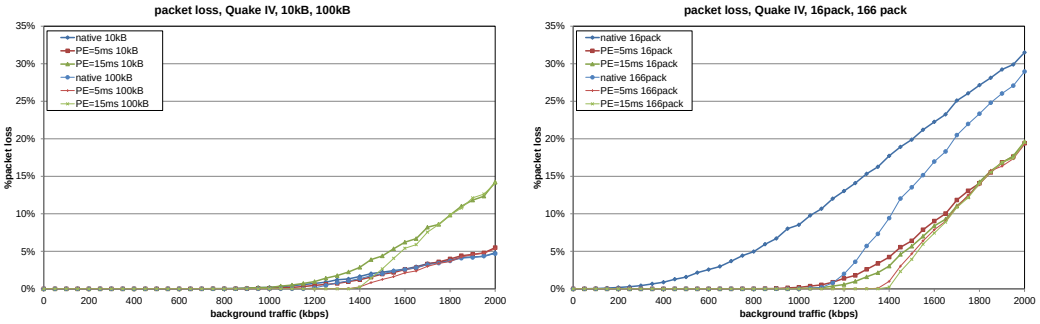


Figure 9. RTT (1), jitter (2) and MOS (3) for *10 kB, 100 kB* (a), *16 packets* and *166 packets* (b) buffers

Regarding packet loss (Fig. 10), we can observe that *big* buffers start losing packets exactly when the bandwidth limit is reached, whereas *small* ones start before reaching the limit. Another effect is related to the modification of packet size caused by multiplexing, which represents a trade-off: if the buffer is *byte-sized*, then big packets have a bigger probability of being discarded. So multiplexing with *PE*=15 ms obtains worse results than *PE*=5 ms, despite the fact that we are saving more bandwidth. If the buffer is *packet-sized*, then all the packets have the same probability of being discarded, so packet loss is increased with respect to *byte-sized* buffers. In this case, packet loss is significantly reduced when multiplexing. This advantage is caused by the pps reduction. As a conclusion, we can say that *small* buffers are more adequate to maintain delay and jitter at acceptable levels, but they increase packet loss.



(a) (b)

Figure 10. Packet loss for *10 kB, 100 kB* (a), *16 packets* and *166 packets* (b) buffers

4.4. Other Limitations of the Router

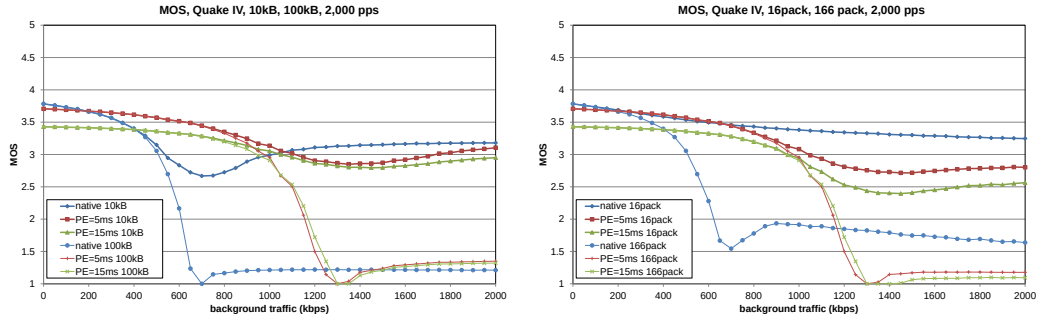
Finally, Fig. 11 illustrates another limitation of the router, i.e., the number of pps it can manage. Low-end routers, which are frequently used for providing residential access, may have a low processing capacity, so they may experience some problems when dealing with high amounts of pps [5] [6].

In this case, we have only included MOS graphs. If compared with Fig. 9.3., we see that the behavior becomes clearly worse if the router has a limitation of 2,000 pps (Fig 11.1). Although the maximum offered amount of pps is 1,652, this limitation has a very bad effect on MOS.

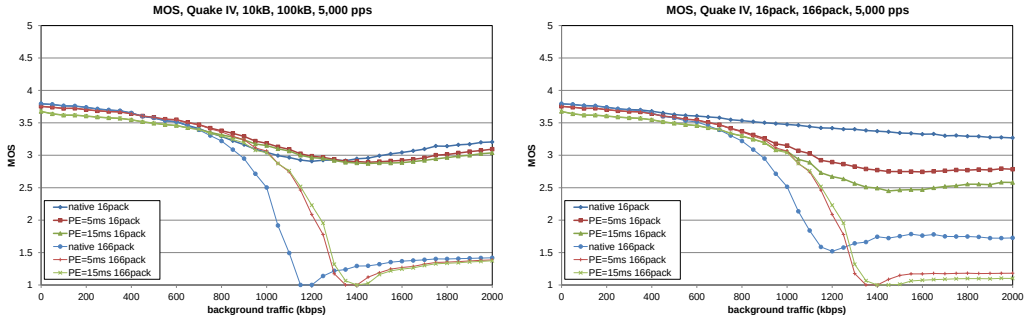
For the *byte-sized* buffer (Fig 11.1.a), multiplexed traffic maintains quality above 3 for bigger amounts of background traffic, while the "valley" in the graph of the native traffic using the small buffer appears when a smaller amount of background traffic is present.

If the *packet-sized* buffer is used (Fig. 11.1.b), then the native traffic using the smallest buffers is the one that presents the best results. In conclusion, we see that although multiplexing is good for one buffer implementation, it is bad for the other one. This illustrates that, in addition to the bandwidth limit of our connection, the limitation of pps has to be considered.

However, if the amount of packets per second that the router can manage is bigger, this limitation does not have any influence. Fig 11.2 (a) and (b) show the behaviour when the router has a limit of 5,000 pps. We can see that the differences with respect to Fig. 9.3. are negligible in this case. Thus, we see that this limitation is only important when the router has a processing capacity in the order of the amount of pps received.



(1.a) (1.b)



(2.a) (2.b)

Figure 11. MOS for *byte-sized 10 kB, 100 kB* (a) and *packet-sized 16 packets, 166 packets* (b) buffers with a limitation of 2,000 pps (1) and 5,000 pps (2).

4.5. Discussion of the Results

First of all, we must say that the obtained delay results are not surprising: as the buffer gets smaller and the bandwidth gets bigger, the buffer can be emptied more quickly. But the jitter presents a peak, which is reduced if the offered traffic is bigger than the bandwidth of the connection, because in this case the buffer is always full. This causes the surprising behaviour of the MOS graphs. Although the results have been obtained using a concrete quality model, this behaviour can be expected to be similar for other subjective quality estimators based on delay and jitter.

On the other hand, packet loss does not affect the game traffic. With the *byte-sized* buffer, it has been measured to be small, as drop-tail FIFO policy penalizes big packets, and the ones of the game are small. But if a *packet-sized* buffer is used, then the game traffic no longer has an advantage, since packet loss probability is independent of packet size in this case.

To sum up the results, we can highlight the importance of integrating network parameters into a MOS value. We cannot simplify the problem and study each one of the KPIs separately. In the case of the application under study, they are delay and jitter, which are affected by the bandwidth of the access network, as expected. But the results show that the buffer size and its implementation have a strong influence too. It has also been shown that a bigger bandwidth permits bigger output buffers. An application which runs well in a local environment may experience problems when using an access network to interact with a game server located on the Internet, so buffer size is a crucial parameter which has to be configured taking into account all these relationships, especially for commercial access routers.

5. Conclusions

This work has studied the effect of the router buffer on the perceived quality for certain online FPS games. Previous studies deploying subjective tests with gamers have shown that their main KPIs are delay and jitter. Packet loss is not considered unless its amount gets very high. The influence of these parameters has been studied and in addition, the influence of a traffic multiplexing method has been considered.

Simulations have been conducted in order to obtain the delay and jitter produced by the buffer of the access router, using different amounts of background traffic. Two buffer implementations, each one with different buffer sizes, have been tested in order to study the mutual influences of the router buffer, multiplexing and subjective quality. Network delay and jitter have also been added. The traces of the game have been obtained from real parties which were properly combined in order to obtain the traffic of 20 simultaneous players sharing the same Internet access. The results show a jitter peak that causes a “valley” in the MOS graph, obtaining an anomalous region in which the MOS grows with the background traffic amount.

Small buffers present better results than bigger ones, and there is a relationship between the recommended buffer size and the connection bandwidth, which has to be taken into account. Buffers measured in bytes add a fixed delay, corresponding to their size, when the total bandwidth gets above the limit. They present a jitter peak when the offered traffic is equivalent to the link capacity. Buffers which size is measured in packets add less delay, but they increase packet loss, as all packets have the same probability of being discarded.

Finally, low-end buffers with a very low limit in packets per second have also been tested, and in this case multiplexing has shown an advantage in one case, but obtains worse results in the other. So this illustrates the need of knowing the characteristics of the buffer in order to make the correct decision about multiplexing the traffic or not. It would be interesting for game developers to identify the behaviour of the router buffer so as to adapt the traffic to it.

References

[1] ITU-T Recommendation G.107 (2011) The E-model, a computational model for use in transmission planning.

[2] Suznjevic M, Stupar I, Matijasevic M (2011) Traffic modeling of player action categories in a MMORPG. In Proceedings of 4th International Conference on Simulation Tools and Techniques (SIMUTools) 2011, Barcelona, Spain.

[3] Oliveira M, Henderson T (2003) What online gamers really think of the Internet?. In Proceedings of the 2nd Workshop on Network and system support for games (NetGames '03). ACM, New York, NY, USA, pp. 185-193.

[4] Ubicom White Paper (2005) OPScore, or Online Playability Score: A Metric for Playability of Online Games with Network Impairments. http://www.kevingee.biz/wp-content/uploads/2011/04/IP3K-DWP-OPSCORE-10.pdf, Accessed 3 Feb. 2012.

[5] Yu J, Al-Ajarmeh I (2007) Call Admission Control and Traffic Engineering of VoIP. In Proceedings Second International Conference on Digital Telecommunications, IEEE ICDT, San Jose, Cal, pp. 11.

[6] Feng W, Chang F, Feng W, Walpole J (2005) A Traffic Characterization of Popular On-Line Games. IEEE/ACM Transactions on Networking, Volume 13 Issue 3, pp. 488-500.

[7] Saldana J, Fernandez-Navajas J, Ruiz-Mas J, Aznar J.I, Viruete E, Casadesus L (2011) Influence of the Router Buffer on Online Games Traffic Multiplexing. In Proceedings International Symposium on Performance Evaluation of Computer and Telecommunications Systems SPECTS 2011, The Hague, Netherlands, pp. 253-258.

[8] Wattimena A.F, Kooij R.E, van Vugt J.M, Ahmed O.K (2006) Predicting the perceived quality of a first person shooter: the Quake IV G-model. In Proceedings 5th SIGCOMM workshop on Network and system support for games (NetGames '06), ACM, New York, NY, USA.

[9] Ries M, Svoboda P, Rupp M (2008) Empirical study of subjective quality for Massive Multiplayer Games. Systems, Signals and Image Processing, 2008. In Proceedings IWSSIP 2008. 15th International Conference on, pp.181-184.

[10] Dischinger M, Haeberlen A, Gummadi K.P, Saroiu S (2007) Characterizing residential broadband networks. Proceedings 7th ACM SIGCOMM Conference on Internet measurement (IMC '07). ACM, New York, NY, USA, pp. 43-56.

[11] Schaefer C, Enderes T, Ritter H, Zitterbart M (2002) Subjective quality assessment for multiplayer real-time games. In Proceedings 1st Workshop on Network and system support for games (NetGames '02). ACM, New York, NY, USA, pp. 74-78.

[12] Zander S, Armitage G (2004) Empirically measuring the QoS sensitivity of interactive online game players. In Proceedings of the Australian Telecommunications Networks and Applications Conference 2004 (Bondi Beach, Sydney, Australia, Dec. 8--10). ATNAC, Australia, 2004, pp. 511-518.

[13] Dick M, Wellnitz O, Wolf L (2005) Analysis of factors affecting players' performance and perception in multiplayer games. In Proceedings 4th ACM SIGCOMM Workshop on Network and system support for games (NetGames '05). ACM, New York, NY, USA, pp. 1-7.

[14] Pereira R.M, Tarouco L.M (2009) Adaptive Multiplexing Based on E-model for Reducing Network Overhead in Voice over IP Security Ensuring Conversation Quality. In Proceedings Fourth international Conference on Digital Telecommunications, Washington DC, pp. 53-58.

[15] Saldana J, Murillo J, Fernandez-Navajas J, Ruiz-Mas J, Viruete, E, Aznar, J.I (2011) Evaluation of Multiplexing and Buffer Policies Influence on VoIP Conversation Quality. In Proceedings CCNC 2011-3rd IEEE Workshop on Digital Entertainment, Networked Virtual Environments and Creative Technology (DENVECT), pp 1147-1151, Las Vegas.

[16] Thompson B, Koren T, Wing D, RFC 4170: Tunneling Multiplexed Compressed RTP (TCRTP), Nov. 2005.

[17] Sze H, Liew C, Lee J, Yip D (2002) A Multiplexing Scheme for H.323 Voice-Over-IP Applications. IEEE Journal on Selected Areas in Communications, vol. 20, pp. 1360-1368.

[18] Koren T, Casner S, Geevarghese J, Thompson B, Ruddy P, RFC 3545, Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering, Jul. 2003.

[19] Pazhyannur R., Ali I. Fox C., PPP Multiplexing (PPPMux), RFC 3153, Aug. 2001.

[20] Trad A, Afifi H (2003) Adaptive Multiplexing Scheme for Voice Flow Transmission Across Best-Effort IP Networks. INRIA Research Report 4929.

[21] Saldana J, Fernández-Navajas J, Ruiz-Mas J, Aznar J I, Viruete E, Casadesus L (2011) First Person Shooters: Can a Smarter Network Save Bandwidth without Annoying the Players?. IEEE Communications Magazine, Consumer Communications and Networking Series, vol. 49, no. 11, pp. 190-198.

[22] Saldana J, Wing D, Fernandez-Navajas J, Ruiz-Mas J, Perumal M. A.M., Camarillo G (2012) Widening the Scope of a Standard: Real Time Flows Tunneling, Compressing and Multiplexing. In Proceedings IEEE ICC 2012, Workshop on Telecommunications: from Research to Standards, June 10-11, 2012, Ottawa, Canada.

[23] Vishwanath A, Sivaraman V, Thottan M. (2009) Perspectives on router buffer sizing: recent results and open problems. SIGCOMM Computer Communication Review 39, 2, pp. 34-39.

[24] Appenzeller G., Keslassy I., McKeown N. (2004) Sizing router buffers. In Computer Communication Review, pp. 281–292, New York, USA. ACM Press.

[25] Enachescu M, Ganjali Y, Goel A, McKeown N, Roughgarden T (2005) Part III: routers with very small buffers. SIGCOMM Computer Communication Review 35, 3, pp. 83-90.

[26] Vishwanath A, Sivaraman V (2008) Routers with Very Small Buffers: Anomalous Loss Performance for Mixed Real-Time and TCP Traffic. In Proceedings IEEE IWQoS, Enschede, The Netherlands, Jun. 2008, pp. 80-89.

[27] Vishwanath A, Sivaraman V, Rouskas G.N (2009) Considerations for Sizing Buffers in Optical Packet Switched Networks. In IEEE INFOCOM 2009, Brazil.

[28] Palazzi C.E, Ferretti S, Roccetti M, Pau G, Gerla M (2006) What's in that Magic Box? The Home Entertainment Center's Special Protocol Potion, Revealed, IEEE Transactions on Consumer Electronics, IEEE Consumer Electronics Society, vol. 52, no. 4, pp. 1280-1288.

[29] Furuholt B, Kristiansen S, Wahid F (2008) Gaming or gaining? Comparing the use of Internet cafes in Indonesia and Tanzania. The Internet Information and Library Review, Volume 40, Issue 2, pp. 129-139.

[30] Gurol M, Sevindik T (2007) Profile of Internet Cafe users in Turkey. Telematics and Informatics, Vol. 24, Issue 1, pp. 59-68.

[31] Batool S.H, Mahmood K (2010) Entertainment, communication or academic use? A survey of Internet cafe users in Lahore, Pakistan, Information Development, vol. 26. Pp. 141-147.

[32] Saldana J, Murillo J, Fernandez-Navajas J, Ruiz-Mas J, Aznar J.I, Viruete E (2011) Bandwidth Efficiency Improvement for Online Games by the use of Tunneling, Compressing and Multiplexing Techniques. In Proceedings Int. Symp. Performance Evaluation of Computer and Telecommunications Systems SPECTS 2011, pp. 227-234, The Hague, Netherlands.

[33] Nuaymi, L, Bouida N, Lahbil N, Godlewski P (2007) Headers Overhead Estimation, Header Suppression and Header Compression in WiMAX. In Proceedings Wireless and Mobile Computing, Networking and Communications, 2007. WiMOB 2007. Third IEEE International Conference on, p.17.

[34] Saldana J, Murillo J, Fernandez-Navajas J, Ruiz-Mas J, Viruete E, Aznar J.I (2011) QoS and Admission Probability Study for a SIP-Based Central Managed IP Telephony System. In Proceedings New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference.

[35] Stewart L, Branch P (2006) HLCS, Map: dedust, 5 players, 13Jan2006. Centre for Advanced Internet Architectures SONG Database, http://caia.swin.edu.au/sitcrc/hlcs_130106_1_dedust_5_fragment.tar.gz, Accessed 5 April 2011.

[36] Cooperative Association for Internet Data Analysis: NASA Ames Internet Exchange Packet Length Distributions.

[37] Sommers J, Barford P, Greenberg A, Willinger W (2008) An SLA perspective on the router buffer sizing problem. SIGMETRICS Performance Evaluation Review 35, 4, pp 40-51.

[38] Kaune S, Pussep K, Leng C, Kovacevic A, Tyson G, Steinmetz R (2009) Modeling the internet delay space based on geographical locations. In Proceedings of the 2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP '09). IEEE Computer Society, Washington, DC, USA, pp. 301-310.

[39] AT&T Global IP Network, http://ipnetwork.bgtmo.ip.att.net/pws, Accessed 3 Feb. 2012.