Title:

On the Effectiveness of an Optimization Method for the Traffic of TCP-Based Multiplayer Online Games[1]

Author:
Jose Saldana

Affiliation:
Aragon Inst. of Engineering Research (I3A)
University of Zaragoza

Address:
L2.05
Ada Byron Building. EINA
50018 Zaragoza, Spain

Phone:
+34 976 762 698

E-mail:
jsaldana@unizar.es

[1] Some parts of this paper appeared in International Symposium on Performance Evaluation of Computer and Telecommunication Systems SPECTS 2012, Genoa, Italy, 8-11 July 2012. Other parts appeared in Consumer Communications and Networking Conference (CCNC), 2014 IEEE, Las Vegas, pp 447-452, 10–13 Jan 2014.

ON THE EFFECTIVENESS OF TRAFFIC OPTIMIZATION FOR TCP-BASED MULTIPLAYER ONLINE GAMES

ABSTRACT

This paper studies the feasibility of using an optimization method, based on multiplexing and header compression, for the traffic of Massively Multiplayer Online Role Playing Games (MMORPGs) using TCP at the Transport Layer. Different scenarios where a number of flows share a common network path are identified. The adaptation of the multiplexing method is explained, and a formula of the savings is devised. The header compression ratio is obtained using real traces of a popular game and a statistical model of its traffic is used to obtain the bandwidth saving as a function of the number of players and the multiplexing period. The obtained savings can be up to 60 % for IPv4 and 70 % for IPv6. A Mean Opinion Score model from the literature is employed to calculate the limits of the multiplexing period that can be used without harming the user experience.

The interactions between multiplexed and non-multiplexed flows, sharing a bottleneck with different kinds of background traffic, are studied through simulations. As a result of the tests, some limits for the multiplexing period are recommended: the unfairness between players can be low if the value of the multiplexing period is kept under 10 or 20 ms. TCP background flows using *SACK* (Selective Acknowledgment) and *Reno* yield better results, in terms of fairness, than *Tahoe* and *New Reno*. When UDP is used for background traffic, high values of the multiplexing period may stress the unfairness between flows if network congestion is severe.

KEYWORDS

traffic optimization; online games; MMORPG; header compression; multiplexing; subjective quality.

1. Introduction

In the last years we are witnessing the rise of a set of emerging real-time services that use the Internet for the delivery of interactive multimedia applications, as videoconferencing, telemedicine, video vigilance, online gaming, etc. Due to the need of interactivity, updates between the extremes of the communication are sent at a fast pace, and this results in flows composed of high rates of small packets. In addition, some other services also send small packets, but they are not delay-sensitive, *e.g.*, instant messaging, or M2M (Machine to Machine) packets sending collected data in wireless sensor networks. For both the delay-sensitive and delay-insensitive applications, their small data payloads incur significant overhead, since they typically contain a few tens of bytes. Furthermore, the payload-to-header ratio becomes even lower when IPv6 is used, since the basic IPv6 header is twice the size of the IPv4 one.

As a consequence of the increasing use of *"small-packet services"* (as they are often called, [1]), network operators are witnessing a change on the packet size distribution of the traffic mix they have to manage, which also implies a reduction of the overall network efficiency. As an additional problem, the global traffic of some of these services may present a degree of unpredictability, with the consequence of traffic surges appearing at certain moments (*e.g.*, the release of a new game or new content of an existing one) or places (*e.g.*, instant messaging during a sports event, a concert, etc.). This phenomenon, also known as *"flash crowd,"* may jeopardize the stability of the network, resulting in undesired service interruptions.

Therefore, there is a need for mechanisms providing a degree of flexibility, in order to make the networks able to manage these unexpected traffic variations. Between these mechanisms, traffic optimization based on header compression and multiplexing can be effective when the service sends high rates of small payloads. An interesting question arises because some of these emerging services (namely Massively Multiplayer Online Role Playing Games, MMORPGs from now) use TCP for transporting the information between the client and the game server. Taking into account that the game usually generates small payloads, and the generation of high rates of TCP acknowledgement packets (ACKs), the resulting network efficiency of these applications is very low. All in all, we will first review the different techniques for optimizing small-packet services, and then explain the specific problems arising when TCP-based traffic flows are compressed and multiplexed.

1.1. VoIP Optimization

VoIP was the first small-packet service becoming popular. In order to increase its low network efficiency, different traffic optimization techniques (also called multiplexing techniques) were proposed [2]. In fact, a technique for optimizing a number of RTP voice flows sharing a common path was standardized in 2005 by the Internet Engineering Task Force (IETF) [3]. The technique, called TCRTP *(Tunneling Multiplexed Compressed RTP)*, combines header compression, multiplexing and tunneling in order to drastically reduce bandwidth consumption (up to 55 % for certain voice codecs) and the amount of packets per second (by a factor of 20). This technique implies a tradeoff, since these savings are achieved at the cost of *i)* processing power required by header compression; *ii)* a small multiplexing delay necessary for getting a number of packets to be multiplexed. Thus, the effect of optimization techniques on the delay has to be quantified in order to establish the conditions in which subjective quality can be maintained. As shown in [4], where the *E-Model*, developed by the International Telecommunications Union (ITU), for estimating subjective quality was used [5], TCRTP is able to compress traffic while maintaining subjective quality in acceptable levels.

A typical scenario where this optimization can be deployed is presented in Fig. 1: two remote offices of the same company are connected via a tunnel, including a number of simultaneous voice calls, doing a sort of "voice trunking". If a security tunnel is to be employed, this can also be useful in order to share the tunnel overhead between several packets. If a multiplexed bundle includes one packet belonging to each conversation, then the additional delay can be limited to the value of inter-packet time, as shown in Fig. 2.

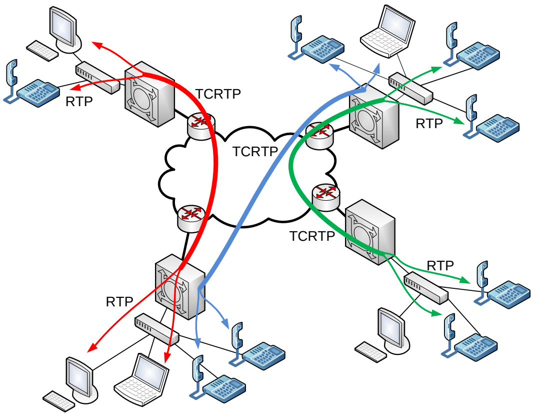


Fig. 1 Scenario where a number of VoIP calls are multiplexed using TCRTP

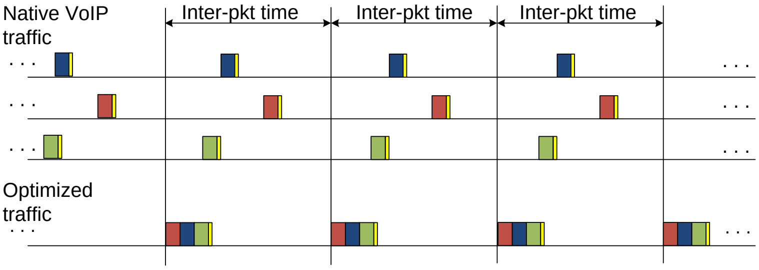


Fig. 2 VoIP multiplexing scheme

1.2. First Person Shooter Games Traffic Optimization

The traffic profile of other emerging services, namely First Person Shooter online games (FPS from now) [6], presents some similarities with that of VoIP, *i.e.*, it consists of high rates of small UDP packets, although FPSs do not use RTP. In an FPS a few tens of players share a virtual world, where they have to achieve a mission or eliminate all the enemies using different guns. Thus, interactivity is crucial: players move and shoot very fast, and network latency is really critical. In fact, the literature has shown that a higher latency can be translated into a higher probability of being shot [7]. These games use UDP, since they prioritize latency versus delivery grant, and they implement different mechanisms (redundancy, prediction) in order to compensate packet loss [8].

Taking into account the similarities between the traffic of VoIP and FPS games, and their increasing popularity, the possibility of broadening the scope of TCRTP optimization has been proposed to the IETF, with the idea of not only considering VoIP, but also UDP-based small-packet services [9]. In [10] the expected savings when optimizing eight different UDP-based FPS games were calculated, showing that a bandwidth reduction of 30 or 35 % can be obtained for client-to-server flows. If IPv6 is used, these figures may rise up to 55 %.

Another fact has to be taken into account: in contrast with some VoIP solutions, commercial online games do not use peer-to-peer but client-server architectures. A central entity is used in order to maintain the status of the game, so the client application uploads the actions of the player, and receives the updated status of the game, including the rest of the players' movements. The reasons for using a client-server architecture include the easiness of maintaining a coherent status, the possibility of billing for the service, an easier control of the latency of each player, etc. Thus, the possibility of finding a number of flows sharing a common network path is higher than in VoIP, since all the flows of a game go to the same destination, *i.e.*, the network of the game provider. In the scenario shown in Fig. 3, a network operator can identify all the flows of a certain game generated in the same town or district, optimize and forward them to the game provider, who would rebuild the packets and deliver them to the game server. The optimizers can be added in different places: DSLAM (Digital Subscriber Line Access Multiplexer), BRAS (Broadband Remote Access Server), eNodeB (Evolved Node B), etc. This is translated into savings in the ISP (Internet Service Provider) aggregation network, in a lower number of packets in the Internet router, and in less bandwidth requirements for the game provider.

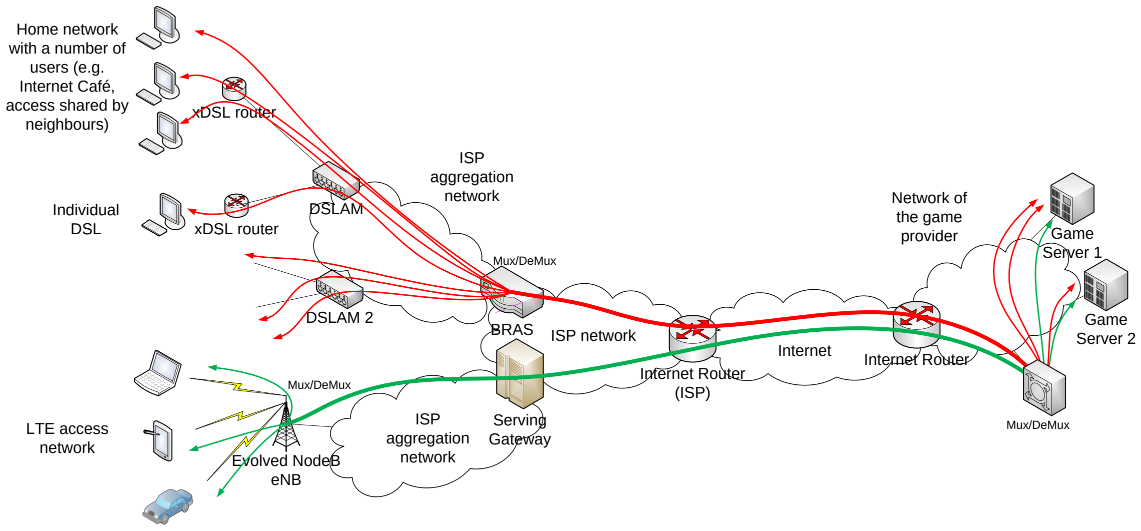


Fig. 3 Scenario where online game traffic is multiplexed

As it has been shown in Fig. 2, the additional delay has an upper bound given by the value of inter-packet time when VoIP flows are optimized. However, online games do not usually send packets with a fixed cadence [11], [12], so a different mechanism has to be employed in order to select the native packets that will travel into a multiplexed one. As proposed in [13], a period *(PE)* can be defined in the optimizer, and all the packets arrived during the interval will travel within the same multiplexed bundle (Fig. 4). This sets an upper bound for the additional multiplexing delay [10], which average value will be *PE*/2. In [14] different studies were conducted with the aim of studying the effect of this additional delay, showing the conditions in which subjective quality can be maintained. A Quality of Experience (QoE) estimator deployed for a game was used [15]. A clear conclusion was devised: when the number of flows to multiplex is high enough, the added delay can be really tiny: as an example, if 20 flows are being multiplexed, bandwidth savings above 30 % are achievable even using a period of 10 ms [10]. In the same situation, the amount of packets per second may pass from 500 to 100.

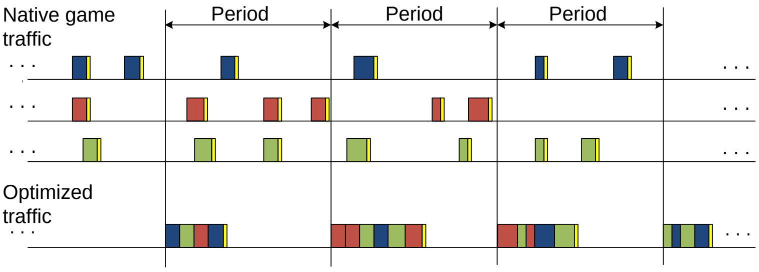


Fig. 4 Online game traffic multiplexing scheme

1.3. Is MMORPG optimization interesting?

We have seen that RTP/UDP and even UDP flows can be optimized. But there are other online game genres that do not use UDP but TCP, namely MMORPGs. In these games, the number of players sharing a virtual world can be huge, not limited to a few tens. They are becoming increasingly popular worldwide, with a special significance in Asia [16]. Some titles have millions of subscribers: as an example, *World of Warcraft*, by *Blizzard*, reported a peak of 12 million registered players [17]. The fact of using TCP for an interactive service may result surprising, and has been studied by scientific literature [18], [19], [20].

The original idea of TCP is that a certain amount of data has to be transferred, so different mechanisms are used in order to send it as soon as possible, while avoiding network congestion and maintaining fairness. In this sense, it can be said that the throughput of TCP is *"network-limited."* However, in an MMORPG, the information is generated as the user plays, so at certain moments it may happen that there is nothing to send. In [20], this behavior was denoted as *"application-limited,"* in contrast to the original *"network-limited"* idea of TCP.

As far as interactivity is concerned, while the level of players' activity and mutual interactions within a typical MMORPG is not as high as in a FPS game, MMORPGs are still considered a "real-time" service, with rather tight requirements in terms of timeliness of packet delivery. Interactivity does matter and delay is seen as bad by players and reduces their scores [21], [22]. There are many differences with FPSs *e.g.*, user behavior, session characteristics and dynamics [23]. This difference has been translated into the use of TCP for information delivery, and this decision prioritizes the reliability of the information transfer with respect to the interactivity.

Nevertheless, the interactivity level of MMORPGs is still high enough so as to make them generate high rates of small TCP packets, especially in the client-to-server direction [22], [24]. Therefore, the possibility of using traffic optimization techniques for these flows has to be explored. The potential savings can be even higher than those obtained with FPSs. The cause is twofold: first, TCP headers (20 bytes) are bigger than UDP ones (8 bytes); and second, the high rates of TCP ACKs (up to 56% of the packets, according to [24]), makes the compression rate be higher, since they do not include a payload, so header compression is applied to the whole packet.

However, new questions arise when considering the optimization of TCP flows. In this case, the problem is not as simple as for UDP flows, which are unidirectional (open loop) and do not depend on the arrival of ACKs for sending a new packet nor retransmit lost packets. When TCP is optimized, the additional multiplexing delay may modify the evolution of the parameters that govern the dynamics of the transport protocol, as *e.g.*, Round Trip Time (RTT).

***

All in all, we see that the optimization of MMORPG flows can result in interesting savings, and their increasing popularity makes it easier to join high numbers of flows to be multiplexed together. The scenarios where high numbers of traffic flows share the same path are the most suitable ones for the deployment of traffic optimization, representing a benefit for all the actors: network providers will see a traffic reduction; game providers will not be affected by network capacity overflow; end users will experience a more stable service, even in congestion times.

However, their optimization is not as straightforward as in the case of UDP-based services, since the modification of the RTT may modify the behavior of TCP. So the contribution of the present work is twofold:

i) the potential savings to be obtained when optimizing these flows are calculated;

ii) the effects of traffic optimization when applied to TCP-based MMORPG flows are studied.

World of Warcraft (WoW), will be mainly used as the application under study, because of the next reasons: first, it is still one of the most popular MMORPGs; second, it follows the typical traffic patterns of this genre; third, it has been largely studied in the literature [12], [22], [23], [24], [25]; and fourth, a Mean Opinion Score (MOS) model has even been developed for it [26].

The rest of the article is organized as follows: the next section summarizes the related work. Section 3 includes a characterization of the traffic of the game under study. The next section details the traffic optimization method, and the expected savings are obtained in Section 5. Section 6 uses a simulation setup to study the specific questions derived from the addition of a multiplexing delay to TCP flows, focusing on the potential unfairness between multiplexed and non-multiplexed flows. The paper ends with the conclusions.

2. Related Work

This paper is focused on the optimization of TCP flows of online games, while maintaining subjective quality. Thus, in order to provide a holistic approach, a number of topics have to be considered in this section: we first summarize the different traffic models for online games that have been presented in the literature. The questions arising when using TCP for a real-time service, and the means for estimating subjective quality in games will be subsequently studied. Finally, the traffic optimization methods based on aggregation at certain network locations will be reviewed.

2.1. Traffic models for online games

A number of traffic models have been developed for FPS games [11]. The traffic of MMORPGs has also been analyzed in [18], [27], [28], and some papers focused on *World of Warcraft* [12], [24], [29] have been presented. These studies, based on traffic traces of a game, deploy a mathematical model for each of the flows, including a statistic characterization of the inter-packet time, and other one of the packet size. The obtained models are usually compared to the original one by means of suitable analytical tools, as *Q-Q plot* (“Q” stands for *quantile*). Regarding packet size, in the case of MMORPGs, some proposals [24] do not model the size of the packet but the APDU (Application Protocol Data Unit) size. This is more accurate, because an APDU may require more than a single packet, or TCP mechanisms may cause the sending of a number of APDUs in the same packet.

MMORPGs include a wide range of possible activities, which does not happen in other game genres like FPSs or Sports ones. In an MMORPG the same player may spend some time picking up flowers, which are required for making a potion which may in turn improve his/her strength when fighting a dragon. Obviously, the traffic profile generated when the player is harvesting flowers is very different from the one observed during the fight. Thus, in order to achieve a better accuracy, advanced models not only propose a statistical distribution for the client and other one for the server traffic, but they take into account the different categories of player's activities, creating a model for each of them [22], [30]. Therefore, they reflect the wide range of activities that can be deployed in an MMORPG, some of them more interactive (*e.g.*, *Player vs Player Combat*) and some of them more relaxed (*e.g.*, *Trading*). This is translated into very different traffic patterns and bandwidth amounts being generated by the same application.

The developed models allow the generation of synthetic traffic, which can be useful for further research, thus avoiding the need of volunteers playing the game while performing network measurements. The generation of game traffic has even been included in traffic generators widely used for research purposes: in [31] a model for *Quake III* was included in a traffic generator (Distributed Internet Traffic Generator, D-ITG), and *World of Warcraft* traffic was also included in [32].

Client-to-server game flows are usually thinner than server-to-client ones. The cause is that the client application only needs to communicate to the server the movements of a single user, whereas the server has to update the client with the game status. In FPSs, the server sends to each player the information of the rest of the players, taking into account that the number is usually limited to a few tens in a single virtual scenario. However, in MMORPGs the same virtual world may be shared by thousands of clients, and this is translated into a significant asymmetry between the two traffic directions, which is stressed if a high number of players are fighting in the same place of the virtual world. In order to make this problem affordable, an *"area of interest"* is usually defined for each virtual character, meaning that only the events happening in that area are transmitted to the client [33]. The increase of the server-to-client traffic, as a function of the number of players in an MMORPG server, was measured in [29].

2.2. The use of TCP for online games

TCP is used by certain game genres because it avoids the loss of information, and also because it makes programming tasks easier. However, using TCP for a service that can be considered as interactive, can be a counterintuitive approach [19]. In general, the objective of TCP is to obtain the maximum throughput, while maintaining fairness and avoiding congestion. This is convenient for transferring a certain amount of data, so TCP mechanisms assume that the throughput is limited by the network. However, many of these mechanisms lose their meaning when used by a game, *i.e.*, an application that generates a very limited data to be transmitted, being the interactivity the most important issue. In fact, some TCP mechanisms (delayed ACK, Nagle algorithm) may even deteriorate the player's experience, as reported in [20].

The problem of the *RTT unfairness* of TCP, (*i.e.*, flows with lower RTTs get more throughput) has been largely studied, and different solutions and TCP improvements have been proposed [34]. This phenomenon has been mainly observed for network-limited traffic, in terms of the throughput obtained by each flow. However, when TCP is used for an interactive service, throughput maximization is not the main objective, since the data are generated on the fly, according to the game dynamics and to the players' actions. Thus, when traffic optimization is employed, it may add a new delay and jitter to the flows, which may share the network with non-optimized ones. The competence between MMORPG flows affected by different latencies was studied in [29] showing that, although the throughput is not affected, the overall RTT and the retransmission rate become worse for the flows experiencing a higher delay.

If these findings are applied to the problem of traffic optimization, it can be assumed that the agents that perform the optimization (*e.g.*, network operators, service providers) may be interested on limiting the potential unfairness between optimized and non-optimized traffic flows.

2.3. Subjective quality models

In order to get an estimation of the user's perception, subjective quality models were first developed for VoIP [5]: the method is based on the development of a number of subjective tests, and then a mathematical function is devised, in order to obtain a MOS, ranging from 1 (bad) to 5 (excellent) giving an idea of user's perception. The threshold value of acceptable quality is usually considered to be about 3.5.

Subjective quality estimators have also been adapted for different games like *Quake IV* [15], or *World of Warcraft* [26]. The problem is that each game presents a different behavior with respect to each concrete network parameter, since different techniques are used by developers for the concealment of network impairments [8]: for example, in [35] it was reported that, while the players of *Quake IV* were surprisingly not aware of packet loss up to 35 %, Microsoft's *Halo* stopped working when packet loss was 4%. As a consequence, each game has to be particularly studied by means of subjective surveys.

2.4. Traffic optimization by means of header compression

Traffic optimization may refer to different techniques aimed to modify the profile or certain flows, in order to adapt them before travelling through the network. For example, a number of TCP flows between the same pair of machines can be optimized with Transport Multiplexing Protocol (TMux) [36], which puts a number of Transport segments into a single TCP packet. Other optimization techniques have been proposed for improving TCP performance in wireless networks: TCP was initially developed for wired networks, so it folds back when packet loss occurs, based on the idea that packet loss happens as a consequence of buffer overflow when the throughput exceeds the link capacity. However, in wireless networks, packets can be lost depending on different factors as transmission power, interference, etc. Thus, different optimization techniques have been proposed (often including cross-layer features) in order to overcome these limitations. Some of these techniques were proposed long ago (some of them are surveyed in [37]), but new techniques are still being proposed, as e.g. [38].

The specific optimization technique issued in this paper refers to the bandwidth reduction which is possible when long-term flows of small packets share a common link in a packet-switched network. In these scenarios, header compression techniques can be employed, in combination with multiplexing. Some of them are specific for UDP flows, but some others can be applied to TCP traffic.

A number of header compression methods have been defined and standardized [39], which are based on the removal of the header fields that are the same for every packet of a flow (*e.g.,* IP addresses, ports, etc.), and also use *delta* compression for reducing the number of bits of the fields presenting an incremental behavior (*e.g.,* a packet sequence number increasing by one may only require a single byte). This requires the use of a *context*, shared by the origin and destination, which stores the value of non-changing header fields, *e.g.,* IP addresses and ports. Logically, context synchronization between the sender and the receiver is of primary importance, so some *refresh* packets are periodically sent. The first method for compressing TCP/IP headers was proposed by Van Jacobson [40]. Later, IPHC (IP Header Compression) [41] also included the possibility of compressing IPv6 and UDP headers. At the same time, cRTP (Compressing IP/UDP/RTP Headers for Low-Speed Serial Links) was defined [42], being also capable of compressing IP/UDP/RTP headers. Some years later, ECRTP (Enhanced Compressed RTP) [43] presented some improvements with respect to cRTP in links with high delay, packet loss and packet reordering. The last compressing algorithm presented was ROHC (RObust Header Compression) [44], which prevents the desynchronization of the context, a problem especially affecting wireless scenarios.

However, a packet with a compressed header cannot travel end-to-end unless tunneled, but the addition of a tunnel header would cancel the saving obtained by header compression. Thus, header compression can be combined with the multiplexing of a number of packets, sharing the common tunneling overhead in order to obtain significant savings [2], [3]. Multiplexing methods were first designed for RTP flows, due to the existence of scenarios where a number of real-time flows may share the same path. The IETF defined TCRTP as RFC 4170 [3], in order to compress headers, using PPPMux (Point to Point Protocol Multiplexing) to include a number of native packets into a multiplexed one. An L2TPv3 (Layer 2 Tunneling Protocol version 3) tunnel was included in order to permit the end-to-end sending of packets. An adaptation of this method for UDP (not RTP/UDP) has been proposed [9], [10]. In [13] a mechanism for selecting the packets to be multiplexed, using a predefined period, was proposed and compared, in terms of delay and jitter, with another one based on a timeout.

These optimization techniques can be used at certain network locations traversed by high numbers of small-packet flows. Thus, a number of packets sharing a common network path can be multiplexed together adding a small latency. In [45] a series of scenarios suitable for traffic optimization were identified, namely:

- The aggregation network of an operator.
- The tunnels between different offices of a company where a VPN (Virtual Private Network) is established, which may include concurrent RTP flows between the same pair of offices.
- All the small-packet flows generated in an Internet Café can be optimized in order to save bandwidth in the access link.
- In some wireless or satellite connections, multiplexing a number of flows before transmission can simultaneously reduce the required bandwidth and the amount of packets per second generated.

The use of proxies for game-supporting infrastructures was proposed in [46] and [47], considering aggregation as a means for traffic optimization, and taking into account the stringent requirements of this concrete service. In [25] the feasibility of a peer-to-peer support for MMORPGs was studied, and one of the conclusions was that message aggregation can reduce the average network latency. In [48] the reservation on part of the path between a game server and a number of clients was explored, discussing the implications for network infrastructure. The authors of [19] explored the problems derived from the use of TCP for MMORPG games, and one of their tests studied the potential gain of sending the data to a group of users in a single TCP connection. For that aim, an imaginary proxy would perform operations on behalf of the server, redistributing the packets to each individual client. They reported an expected bandwidth saving of 40 % if this proxy was present, since many headers and ACK packets would be avoided. However, the authors did not consider header compression, but only grouping a number of TCP flows as a single one.

The effect of optimization techniques on QoE has been studied by means of subjective quality models. For example, in [49] an adaptive multiplexing method for VoIP was proposed, able to maintain the voice quality in acceptable limits, according to the E-Model. This subjective quality estimator was also used in [4] to evaluate the impairments caused by TCRTP optimization. The effect of traffic optimization on the subjective quality of a UDP-based FPS game was explored in [14].

***

All in all, the present work puts together these issues, studying the effect of traffic optimization techniques on the behavior of TCP-based online games. We will specifically focus on the coexistence and competition between optimized and non-optimized flows sharing a link, with the aim of quantifying the impairments experienced by the players whose flows are optimized. Different figures of merit will be used to present the results.

3. MMORPG's TRAFFIC CHARACTERIZATION

In contrast to what happens with FPSs, virtual characters (avatars) in MMORPGs have a long-term life in a persistent virtual world. This allows the players to improve the abilities of their avatars: they can learn new skills or professions, join a guild, earn armors or weapons, etc. In order to distribute the players, the virtual world is replicated in different servers (also known as *shards*), which are independent, *i.e.*, players cannot interact between shards, thus allowing game providers to limit the number of concurrent players in the same virtual world. It can be said that the combination of *sharding* and the use of the *area of interest* makes the problem of maintaining the game state more affordable.

The vast majority of these games can only be played online, so gaming companies have to deploy robust supporting infrastructures. They know that the network is the part of the problem that they cannot control. So they try to minimize the bandwidth requirements (many games can be played with less than 100 kbps) in order to let the game run with every access technology available. This fact increases the possibilities of market penetration, since playing the game will not only be possible in countries with high Internet deployment, but also in emerging economies, where games are also really popular [50]. So they put everything in the hard disk of the player (*e.g.*, *WoW* v4.3.4 folder in a Windows 7 64 bits PC is 18.9 Gb) and minimize the exchange of information with the server. It must be taken into account that these games are played throughout the World, so the game performance has to be independent of the technology employed in each concrete access network.

So everything is placed in the computer of the player, and a TCP bidirectional connection with the server is established, but only the strictly necessary information is transmitted (*e.g.*, the shape of an object or its texture are not transmitted while playing). In fact, this can be seen as a service in which only signaling traffic is exchanged. Nevertheless, the game requires frequent content updates (weekly, monthly) which exchange high amounts of bandwidth, including new scenarios, characters or virtual items. This can be done while the user is playing or not.

3.1. BEHAVIOR PATTERNS AND TRAFFIC UNPREDICTABILITY

The network support of these increasingly popular applications is still an open problem, in which a number of actors participate: first, the game provider, who wants to accomplish the (high) expectations of the players; second, the ISP, who may be accused of being responsible for the (bad) performance of a game; finally, in some developing countries, in which not many people can afford the cost a computer, some businesses (*e.g.*, *Internet Cafés)*, are also very popular [51], and may provide a shared connection to a eventually high number of players [52].

As explained in the Related Work section, a wide range of activities can be performed by the players of a single MMORPG, thus resulting in very different traffic patterns. Some of the activities are performed by a single player (e.g. *trading* with virtual objects or performing *quests)*, but others require a number of players to cooperate. This fact results in an increasing amount of server-to-client traffic, if there are many other players nearby. In [29] it was shown how this traffic was increased with the number of players in the server. In addition, some of the activities are deployed in specific *arenas*, where a predefined number of players (usually grouped in two teams) fight for supremacy (this activity is called *Player vs. Player combat)*. The effect of the number of players was also characterized in that paper.

In addition, the statistical distribution of the activities strongly varies according to the hour of the day. For example, [22] reports that a strong increase of *Raiding* (an activity which consists of accomplishing a mission in cooperation with other players) occurs in the evening, taking into account that this activity may require one or two hours, so it can only be deployed once arrived from work (we should not forget that the average player is 30 years old [16]). As a result, the aggregate traffic presents a high variability depending on the hour of the day and on the day of the week [53]. For example, it was reported that Fridays and Saturdays generate about 50 % of the peak traffic rate compared to other days.

But the problem of dimensioning the network becomes even worse when a new game (or new content in an already existing one) is released. Taking into account the high number of users, the problem of supporting the service after the release is not trivial. The success of a game is not totally predictable and, in fact, some games, as *Diablo II* [46] and more recently *Diablo III* [2], had serious supporting problems in the first months after their release. The problem is not negligible, since game players have been reported to be very difficult to satisfy, *i.e.,* if a server does not match their requirements, they would leave and never return [54].

All in all, it can be seen that the aggregate traffic has a big variability depending on the hour of the day, the day of the week or the release of new content. Some studies have shown that game servers cannot easily share their capacity with other services (*e.g.,* web) [54], as they present similar daily periodic workload peaks. In the same study, the existence of a limit in the packets per second (pps) that a router can manage was highlighted, and it was recommended to consider this pps limit in addition to bandwidth limitation. Consequently, techniques providing bandwidth and workload savings are interesting so as to avoid the need of over provisioning the resources. Therefore, optimization mechanisms can be useful in order to provide some flexibility to the supporting infrastructure: optimization can be activated only when required, with the counterpart of a small additional delay. The benefit is clear: the game provider may maintain the service online even if the offered traffic is above the capacity of its network. Logically, these additional delays have to be kept low enough, in order not to jeopardize user's experience.

3.2. SPECIFIC ISSUES RELATED TO THE USE OF TCP

Normally, UDP is the selected protocol for real-time services. As it does not retransmit lost packets, its behavior is "inelastic", in the sense that it has no feedback of the success of the transmission: it just sends the packets. However, as said in the Introduction, MMORPGs normally use TCP, which is reliable and avoids the loss of any information related to players' actions. So when a packet is lost, the protocol asks for a retransmission. On the other hand, TCP implements a congestion control algorithm based on a sliding window, which increases its rate according to the received ACK packets. But when packet loss is detected, it slows down since it considers that the network capacity limit has been reached.

Being one of the fundamental Internet protocols, TCP is a rather complex, stateful, closed control loop protocol. Its behavior and performance have been studied extensively, as detailed in the Related Work section. For some applications (e-mail, FTP), the achieved throughput is the most important figure of merit. The same happens when updates of game content are downloaded to the player machine: these flows behave as normal file downloads, so maximizing throughput would be the main objective.

In addition to downloads, TCP is used for web browsing. In this case, throughput is important, but in the last years new protocols are being proposed in order to reduce the latency observed by the person who is browsing the Internet. For example, in [55], TCP *Fast Open* was proposed in order to reduce the number of roundtrips required for the initial handshake. In the same line, HTTP/2 has been recently standardized, allowing multiple exchanges in the same connection and thus reducing the latency perceived by the user [56].

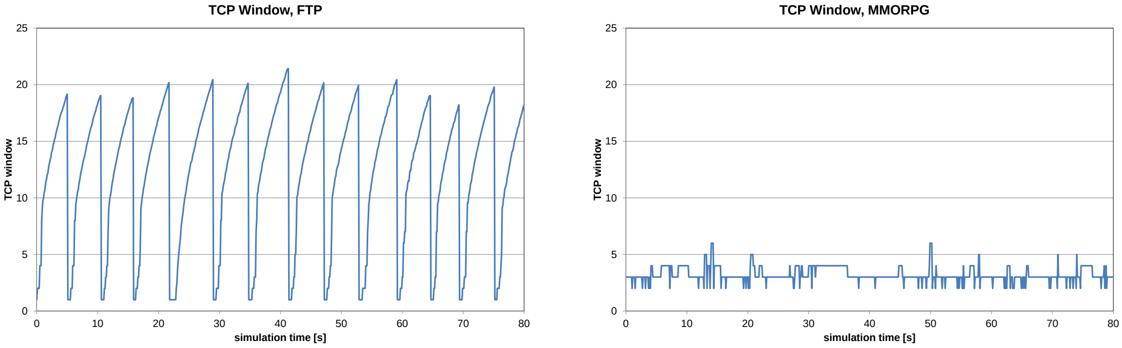
However, at playing time, MMORPG game designers are not concerned about throughput. They just need TCP's ability to reliably deliver the packets from one network host to another. From the players' perspective, the key issue is not the end-to-end throughput, but the end-to-end delay ("lag"), due to its negative effect onto players' in-game performance and, thus, on the perceived quality. In order to avoid additional delays, MMORPGs usually set to 1 the "push" flag of the TCP header [24]. This forces the protocol stack of the player's computer to send the packet as soon as possible. As a result, these games tend to generate small packets. Packets with information are also used as ACKs (piggybacking). Nevertheless, a high number of ACKs without payload are also present. The MMORPG communication does not require a lot of bandwidth; it has been shown that, depending on the situation in the virtual world, traffic demands may rise up to 50 kbps in downlink and 5 kbps in uplink, but that the average demands are typically lower [29].

In order to illustrate the different uses of TCP, Fig. 5 shows the behavior of the sliding TCP window of *a)* the *traditional* use of TCP (network-limited) when downloading a file if the bottleneck is a 1Mbps network, and *b)* the use of TCP for an MMORPG *(World of Warcraft)*, where the limitation is in the application itself, since data is generated according to the player commands. This game has all the

[2] In the case of *Diablo III*, the enterprise was forced to put the game offline, and sent out a letter apologizing for the problems derived of the underestimation of the required resources.

characteristics of a typical MMORPG: it uses a bidirectional TCP, piggybacking ACKs on other packets. In addition, it has different activities with very different traffic patterns.

From the figures, it can be observed that, whereas in *a)* the window grows normally, following the classic pattern of TCP, in *b)* it grows very slowly with frequent drops caused by the absence of data to be sent. As a consequence, its maximum size remains very low.



(a) (b)

Fig. 5 Behavior of TCP window for *a)* network-limited FTP; *b)* application-limited MMORPG *(WoW)*

4. Optimization of MMORPG Traffic

In this section we describe the traffic optimization method used for TCP-based MMORPG flows. We first present the protocol stack, and then the header compression algorithm is explained.

The next list summarizes the specific issues that make this study interesting and different in many aspects from UDP traffic optimization:

- The TCP header length is 20 bytes, whereas UDP one is 8 bytes, so this fact may lead to higher compression rates.
- The presence of a high amount of ACK packets without payload (up to 56 % of the packets [24]) is interesting, since in these packets header compression and packet compression rates are the same.
- Although many FPSs include the possibility of setting up a party in a local network, MMORPGs usually require a connection with the game provider[3]. So users can only play while the server is working, and this makes this service very critical.
- The average session duration of MMORPG is longer than the one in FPS [22], and long-term flows are good for header compression, since all the packets of a flow have the same value for many fields (IP addresses, ports, etc.).
- Since the interactivity is smaller than in FPS, the delay and jitter requirements are different: a higher latency can be tolerated by MMORPG players [57].
- Packet loss is not considered in MMORPGs, since TCP will retransmit lost packets. On the other hand, a retransmission will be translated into additional delay and jitter, which may have an influence on the network impairments used in order to calculate the MOS.
- MMORPGs generate less pps than FPS. This is worse for multiplexing, since a longer period is required for multiplexing the same number of packets.

4.1. Protocol Stack

Fig. 6 shows the protocol stack used to optimize the TCP-based MMORPG traffic. First, header compression is separately applied to the TCP/IP headers of each flow (the payload is not modified). A number of packets belonging to different flows are grouped into a bundle using PPPMux. The bundle is then sent using a suitable tunneling scheme (L2TP).

In order to select the packets to be multiplexed together, a predefined multiplexing period is set in the optimizer. At the end of this period, all the arrived packets, belonging to different flows, are grouped and sent (see Fig. 4). The use of a period sets an upper bound for the added delay [13]. In addition, in order to

[3] Although some private servers (*e.g., TrinityCore*) have been developed for *World or Warcraft*, they are unofficial and based on reverse engineering.

avoid packets bigger than MTU (Maximum Transmission Unit), a size threshold is also defined, and a multiplexed packet is sent whenever the threshold is reached, even if the period has not finished.

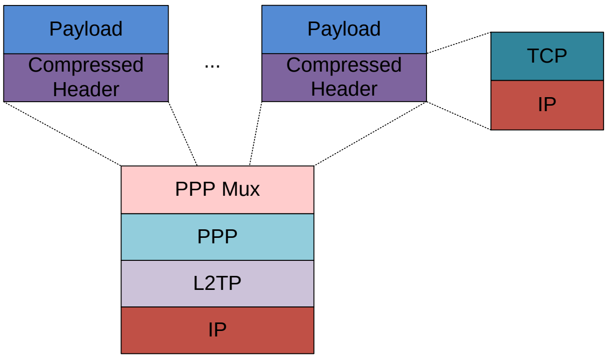


Fig. 6 Protocol stack used for the optimization of TCP/IP traffic

The technique becomes interesting when a high number of flows share the same path, because this permits the sharing of the tunnel overhead between all the packets. The tunnel is established between two machines called *ingress*, where packets are compressed and multiplexed, and *egress*, where they are rebuilt exactly as they were at the ingress. This means that multiplexing is something that happens in the network, so it is transparent to the game client and server. As a result, it can be independently applied for client-to-server and/or server-to-client traffic.

An example of client-to-server MMORPG optimization is shown in Fig. 7 (packet sizes are to scale). An average payload *E*[*P*] of 20 bytes has been used. As it can be observed, the savings obtained are huge, even if a low number of packets (seven and five respectively in the example) are multiplexed. In addition, the optimization of IPv6 packets results in even higher savings, since IPv6 minimum header is twice as big as IPv4 one.

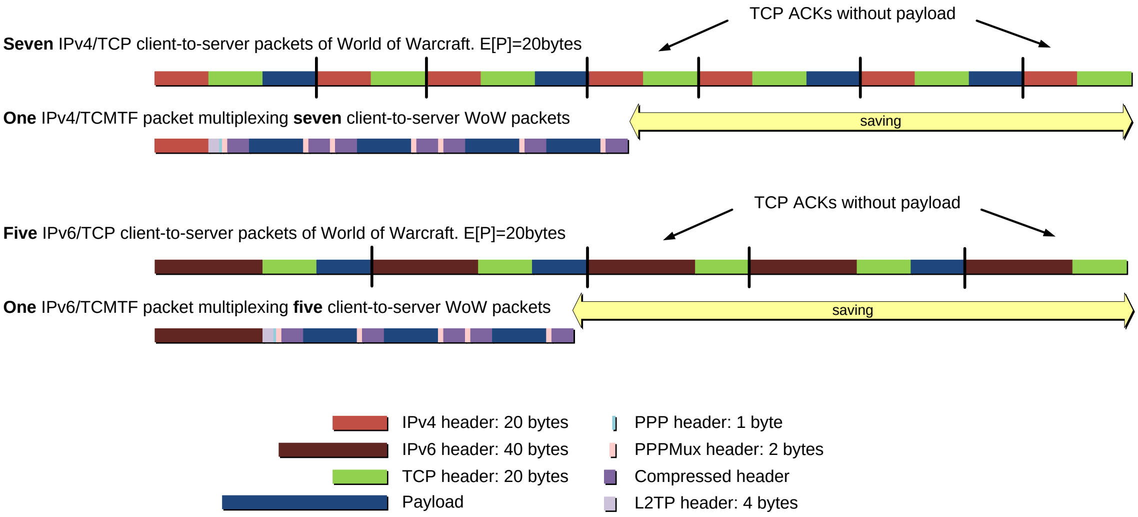

TCP ACKs without payload
Seven IPv4/TCP client-to-server packets of World of Warcraft. E[P]=20bytes
One IPv4/TCMTF packet multiplexing **seven** client-to-server WoW packets
saving
TCP ACKs without payload
Five IPv6/TCP client-to-server packets of World of Warcraft. E[P]=20bytes
One IPv6/TCMTF packet multiplexing **five** client-to-server WoW packets
saving
IPv4 header: 20 bytes
IPv6 header: 40 bytes
TCP header: 20 bytes
Payload
PPP header: 1 byte
PPPMux header: 2 bytes
Compressed header
L2TP header: 4 bytes

Fig. 7 Scheme of a multiplexed packet including a number of native ones (IPv4 and IPv6)

4.2. Header Compression Algorithm

A protocol capable of compressing TCP/IP headers is required, so we may select IPHC [41] or ROHC [44]. Although both use similar compression methods, the latter has been designed to perform well even in links with high RTT and packet loss, as it happens in wireless environments. It sacrifices some amount of compression so as to improve context synchronization guarantees [39]. As the scenarios considered in the present work are wired networks with a very low packet loss rate, IPHC is considered as more adequate in this case.

In order to obtain the expected size of the header, we will briefly summarize the IPHC algorithm, which was adapted from [40], and jointly compresses TCP and IP headers. The fields that are the same for every packet of the flow (*e.g.*, IP address, Port, Protocol, etc.) are denoted as *DEF* fields, and they are only included in non-compressed headers. *Delta* fields (expressed as *"Δfieldname")* can be encoded with a reduced number of bytes, since they are incremental. This is typical of sequence numbers and fields including the value of TCP windows. Finally, *Random* fields (*e.g.*, checksums) do not follow a predefined pattern and cannot be compressed.

The protocol sends two different header types:

- FULL_HEADER: it establishes or refreshes the context of a packet stream, represented by a *context identifier* (CID), *i.e.*, a unique identifier of the flow, necessary to rebuild the packet in the decompressor. The header presents the same size of the original, but it includes the CID value in the second byte of the *total length* field of the IP header. The length of the packet is inferred from lower layer protocols.
- COMPRESSED_TCP: in the rest of the packets, a compressed header is sent. Its scheme can be seen in Fig. 8. The first byte includes the identifier of the context (CID), and the second one is a mask that indicates which fields are present in the header, *e.g.*, if the bit *S* is set to 1, this means that the field *Δsequence (S)* is present. There is an exception: the bit *P* is a copy of the one of the original header. This is the *push* bit, and it indicates if this packet has to be sent immediately. *Random* fields have to be sent normally, and they are included after *TCP checksum*, in the same order as they appear in the original header. *DEF* fields are avoided.

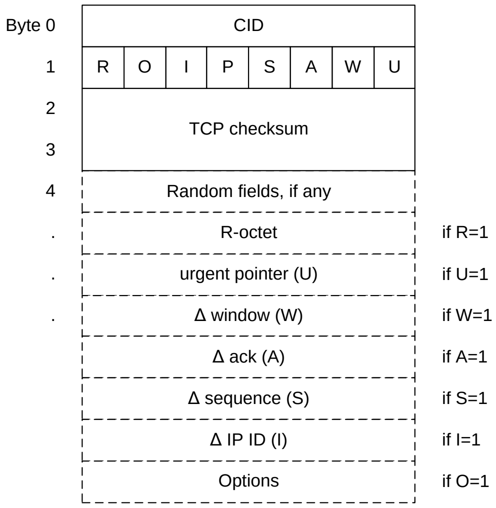


Fig. 8 Header of a COMPRESSED_TCP packet

Ref. [40] also defined a mechanism for including *Full* fields instead of *Delta* ones when necessary: if 8 bits are not enough to express the change in the field (*i.e.*, a change higher than 256), then an extra byte of zeros is included, and next, the full field. So a decision has to be made, depending on the behavior of a field: if the number of times it significantly changes is high, then it will be better to include it as *Random*, thus avoiding the additional byte of zeros.

5. Expected Savings

In this section an analytical calculation of the savings to be obtained when using the proposed optimization method is first presented. Next, the performance of the header compression algorithm is studied, using real traces of the application under test. It should be taken into account that header compression and multiplexing can be seen as independent processes, since a flow can be compressed without using multiplexing, and also a number native flows can be multiplexed without considering header compression. An analytical formula for the expected savings is devised and then simulations with real traces are conducted for obtaining the bandwidth savings and packets per second reduction. A study about the delay limits that have to be taken into account, in order to maintain a good subjective quality concludes the section.

5.1. Calculation of the Bandwidth Savings

In order to obtain a formula for the expected bandwidth savings, we must calculate the expected sum of the sizes of the packets arrived in a period, and also the expected size of the compressed packet. The next variables are defined:

- *NH*: The native header size: It is 40 bytes for TCP/IPv4, and 60 bytes for TCP/IPv6.
- *CH*: The size of the common header of a multiplexed packet, being 25 bytes if IPv4 is used: 20 for IP header, 4 for L2TP and 1 for PPP. For IPv6 it is 45 bytes.
- *MH*: PPPMux header (2 bytes).
- *E*[*P*]: The expected size of the payload, which depends on the application. It must be taken into account that ACK packets without payload can also be compressed, so they will also be considered

in the calculation of the expected value of the payload, with a value of 0. This will make the calculations depend on the implementation and parameters of TCP in the game server and the machine of the player.

- $E[k]$: The average number of native packets included into a multiplexed one.
- $E[RH]$: The expected value of the compressed (or *Reduced*) header.

P, k and *RH* can be considered as independent random variables: the size of the payload *(P)* is independent from the compression ratio of the header, which would determine *RH.* Although not strictly true for high values of the period or the number of players, it can also be assumed that *k* is independent from *P* and *RH* if the size of the multiplexed packet is not near the MTU, since the end of the period (and not the packet size limit) is the factor that triggers the sending of the multiplexed packet.

The expected sum of the size (in bytes) of the native packets arrived in a period will be:

$$bytes_{native} = E[k]\,(NH + E[P]) \qquad (1)$$

And the expected size of the packets once multiplexed is:

$$bytes_{mux} = CH + E[k]\,(MH + E[RH] + E[P]) \qquad (2)$$

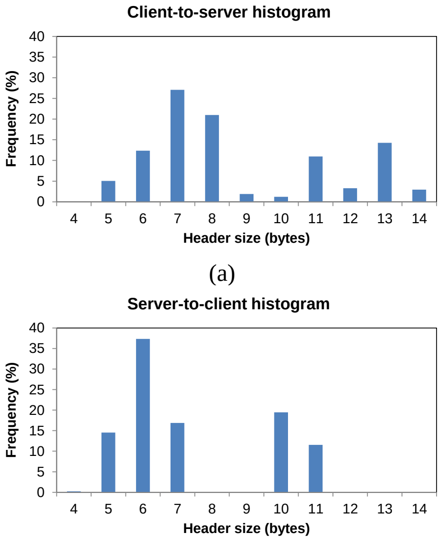
As a consequence, the BandWidth Saving *(BWS)* can be expressed as:

$$BWS = \frac{BW_{native} - BW_{mux}}{BW_{native}} = 1 - \frac{bytes_{mux} / \, period}{bytes_{native} / \, period} = 1 - \frac{CH}{E[k](NH + E[P])} - \frac{MH + E[RH] + E[P]}{NH + E[P]} \qquad (3)$$

The second term represents the sharing of the common tunneling and multiplexing header, and it is reduced as the number of multiplexed packets grows. The third term corresponds to the relationship between compressed and native headers of each packet, so it will be translated into an asymptote for the bandwidth saving, with a fixed value for each game.

At the same time, the number of packets per second is reduced by a factor of $E[k]$, *i.e.*, the average number of multiplexed packets.

Next, in order to calculate the expected value of the compressed header $E[RH]$, we have to obtain the statistical distribution of its size. Traffic traces of the game, obtained from real parties performed in our laboratory, have been used in order to obtain the distribution. We have used a wired connection with a very low packet loss rate, using a Windows 7 64 bits client to play in a *Blizzard* server, and captured 4,000 packets on each direction. The obtained histograms are presented in Fig. 9: the compressed header size for client-to-server packets ranges from 4 to 14 bytes, whereas server-to-client one varies from 4 to 11 bytes. Thus, the expected size of the reduced header is 8.72 bytes for client-to-server packets and 7.37 bytes for server-to-client ones. Taking into account that the original TCP/IP header uses 40 bytes, it can be observed that the saving is significant. The size of the IPv6 headers is one byte smaller, since the IP ID field is not present.



(a)

(b)

Fig. 9 TCP/IPv4 size distribution of the compressed headers.

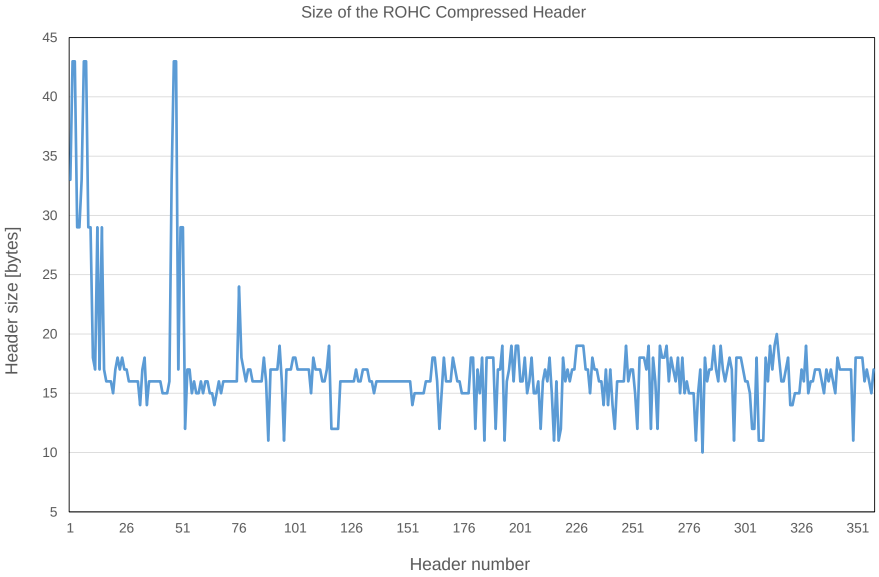
Once the expected sizes of the compressed headers have been estimated, we are now able to present some numerical results of the bandwidth saving asymptote (Table 1), using the expression obtained in (3). In order to get a more general idea of the expected savings, we have calculated the value or the asymptote not only for *World of Warcraft (WoW)*, but also for two more MMORPGs: *ShenZhou Online* [18], by UserJoy Technology; and *Runes of Magic (RoM)*, by Runewalker Entertainment[4]. The values of $E[P]$ have been obtained from the literature or from our own measurements with real traffic traces of the games. Assuming that we are using the same TCP implementation, we can use the calculated value of $E[RH]$ for the three games.

It can be observed that the bandwidth saving is significant for client-to-server traffic, whereas it is lower for the server-to-client direction. Concretely, in the case of client-to-server traffic of *WoW*, the upper bound of the bandwidth saving is roughly 60 % for IPv4 and 70 % for IPv6. The savings are lower for the other two games. On the other hand, the server-to-client saving for this game is very small, because of the big size of the packets. The values for the other games are roughly 20 % for IPv4 and 30 % for IPv6.

client to server				
game	***E[P]***	***pps***	***Saving upper bound, IPv4***	***Saving upper bound, IPv6***
WoW	8.74	9.51	60.07 %	73.15 %
ShenZhou	25	8	45.05 %	59.15 %
RoM	33	4.17	40.11 %	54.06 %
server to client				
game	***E[P]***	***pps***	***Saving upper bound, IPv4***	***Saving upper bound, IPv6***
WoW	314	6.05	8.65 %	13.80 %
ShenZhou	114	8	19.89 %	29.67 %
RoM	99	5.17	22.04 %	32.47 %

Table 1 Upper bound of the bandwidth saving for different games

Finally, in order to corroborate that header compression is able to significantly reduce the size of MMORPG packets, we have run some tests compressing traffic of a real game, using an open source implementation[5] of ROHC [44]. The Bidirectional Optimistic Mode of ROHC is employed. We have deployed a setup where the client-to-server traffic of *World of Warcraft* (version 6.2.0 for Windows) traverses a link where traffic compression is performed. In Fig. 10 the evolution of the size of the compressed headers is shown.



[4] The values for this game are based on empirical measurements deployed in our labs. Although a complete traffic model has not been devised, the average packet size can be easily obtained.

[5] The ROHC implementation can be found here: https://rohc-lib.org/

Fig. 10 Evolution of the size of the compressed headers of client to server traffic of *WoW*

It can be observed that, at the beginning, the savings are low, and the first headers are 43 bytes long: the cause is that these headers are sent uncompressed, and they also include some bytes corresponding to the flow identifier. In addition, when network problems appear, full headers have to be sent again (see packet 51 and subsequent ones). However, it can be seen that the TCP/IP headers can finally be compressed, obtaining an average size of 16.8 bytes (instead of 40). It can be observed that the savings are lower than those expected with IPHC: as said before, ROHC increases robustness at the cost of some overhead and processing requirements [39].

5.2. Estimated Savings with Real Traces

Once the header compression algorithm has been statistically characterized, simulations have been performed so as to obtain packet sizes and packet departure times of both the native and the multiplexed flows. The process can be divided into three stages (Fig. 11), which we will next explain.

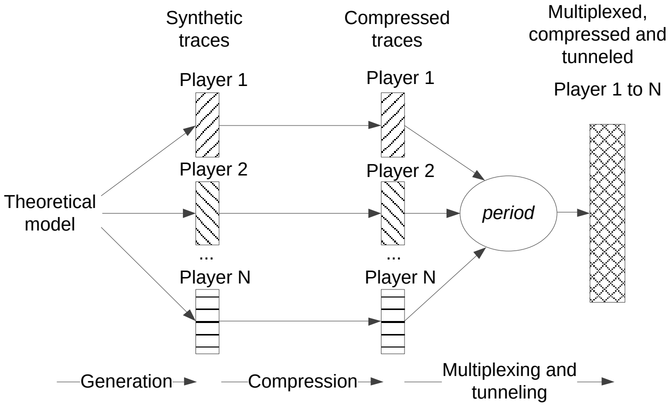


Fig. 11 Stages of the generation of the optimized traffic traces

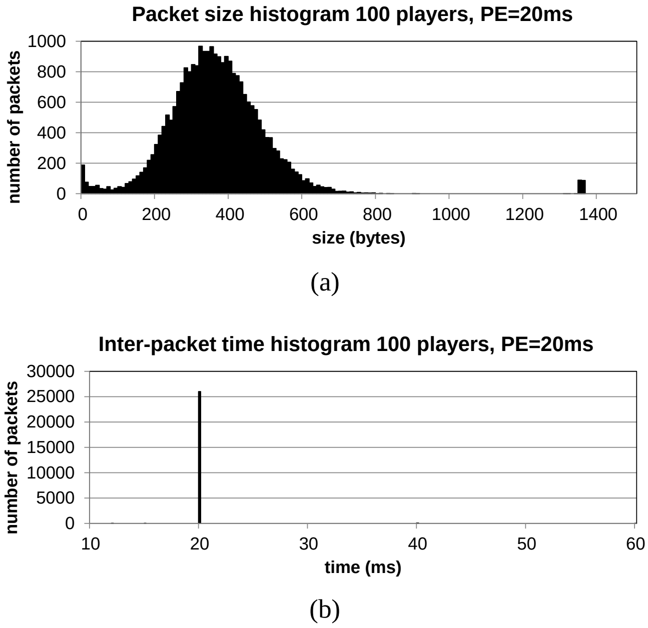
The traffic model for *World of Warcraft* developed in [24] has been used. As explained in the Introduction, this game has been largely used as the example par excellence of MMORPG traffic. In the model, inter-packet time is modeled by a joint distribution of three uniformly distributed variables. The APDU size follows a Weibull distribution for the downlink, and three possible sizes at the uplink. For the obtaining of this model, the authors first removed the ACK packets without payload, which were 56 % of the uplink packets and 28 % of the downlink ones. It must be taken into account that the game also sends ACKs in packets with payload. Following this statistical model, Matlab has been used to generate traffic traces for different numbers of players (in order to have a significant number of packets, 5,000 packets per player are generated) in three steps:

- The APDU and inter-packet times are generated.
- If the APDU is bigger than the MTU, it is divided into a number of packets, which are sent in a burst.
- TCP ACK packets without payload are added, using the rates reported in the model, and the correspondent inter-packet time distribution.

Next, IPHC header compression is applied to each traffic flow, using the statistics obtained in the previous subsection. Compressed packets are then grouped, using a multiplexing period (as shown in Fig. 4). As a result, Fig. 12 and 13 show packet size and inter-packet time histograms when compressing and multiplexing 100 traffic flows using a period of 20 and 60 ms respectively. As expected, the method increases packet size, which average is now 364 for the short period (Fig. 12 a) and 1,080 bytes for the long one (Fig. 13 a). It should be noticed that, as shown in Table 1, the average size for the native traffic is 48.74: 40 bytes corresponding to TCP/IP headers and 8.74 bytes of payload.

A peak around 1,350 bytes appears because a threshold is defined in order to avoid multiplexed packets bigger than the MTU: if the size of the multiplexed bundle reaches this value, the sending is triggered and a new period begins. The peak is small when the period is short (Fig. 12 a), since the probability of reaching the size threshold is low. However, it becomes significant when the period is 60 ms (Fig. 13 a). Regarding inter-packet time, if the period is 20 ms, almost every multiplexed packet departs with this interval (Fig. 12 b). However, when a long period is set, some periods are shorter because the threshold is reached, as shown in Fig. 13 b). Nevertheless, a peak appears in 60 ms (7,462 packets). This means that the vast majority of the periods last 60 ms (the average is in fact 59.2 ms). The peak has been cut for clarity.

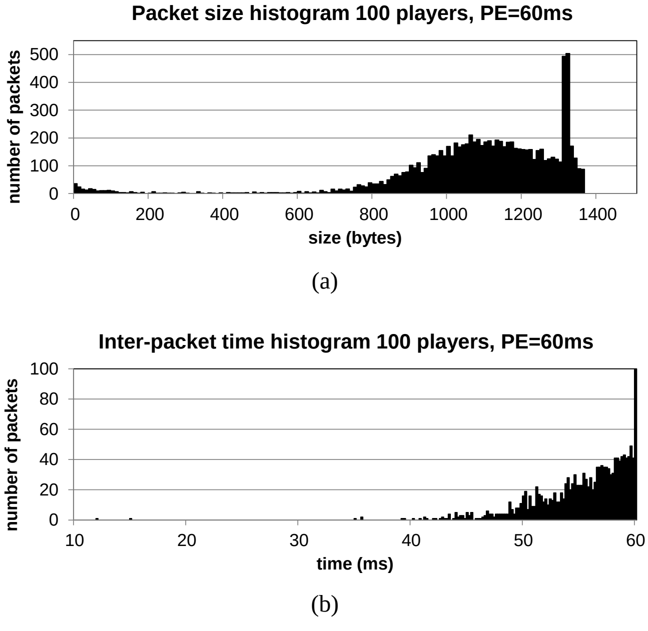
This confirms what we said in Subsection 4.3: if the period is short, it can be assumed that the number of multiplexed packets is independent from the packet size and the compressing ratio. If the period gets longer, the assumption may be translated into a slight inaccuracy of the bandwidth saving estimated by equation (3).



(a)

(b)

Fig. 12 Multiplexed traffic: a) packet size; b) inter-packet time histogram.



(a)

(b)

Fig. 13 Multiplexed traffic: a) packet size; b) inter-packet time histogram.
A peak of 7,462 packets in 60ms has been cut for clarity.

We have seen that client-to-server savings are the highest ones, so we will mainly focus on this traffic. Fig. 14 shows the bandwidth saving when optimizing different numbers of flows, with a period ranging from 10 to 100 ms. It can be observed that the curves present an asymptote around 60 %, as expected according to the results of Table 1. If the number of players is small, a period of 50 ms can be enough so as to obtain bandwidth savings above 50 %. Nevertheless, a saving of 25 % can still be achieved, even with a tiny period of 10 ms. On the other hand, when a big number of flows are multiplexed, bandwidth savings of about 50 % can be obtained even for very small values of the period. Fig. 15 shows the results for IPv6, where bandwidth saving asymptote is above 70 %. Finally, Fig. 16 presents the results in terms of packets per second, which can be reduced from 900 to 10 if a multiplexing period of 100 ms is used.

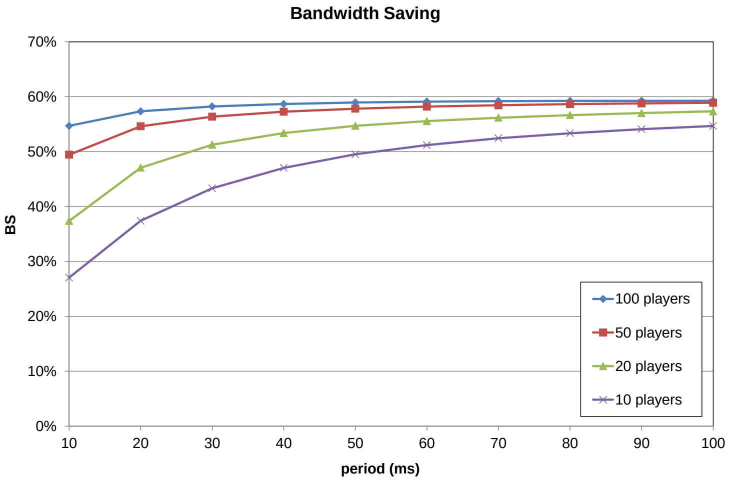


Fig. 14 Bandwidth saving for client-to-server traffic of *WoW* using IPv4.

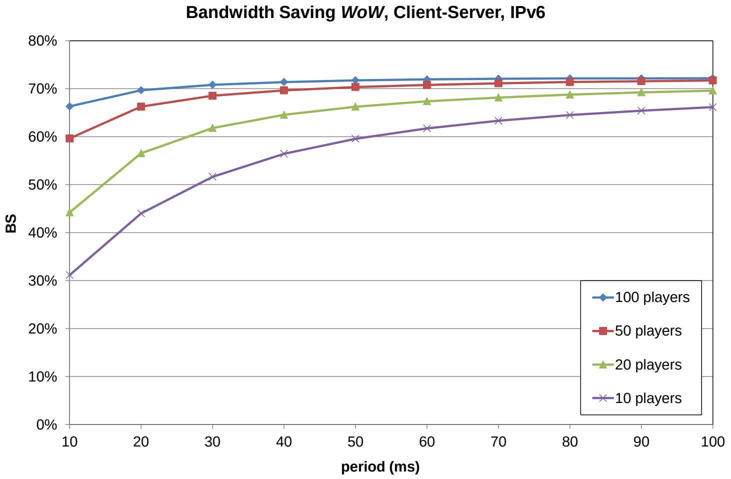


Fig. 15 Bandwidth saving for client-to-server traffic of *WoW* using IPv6.

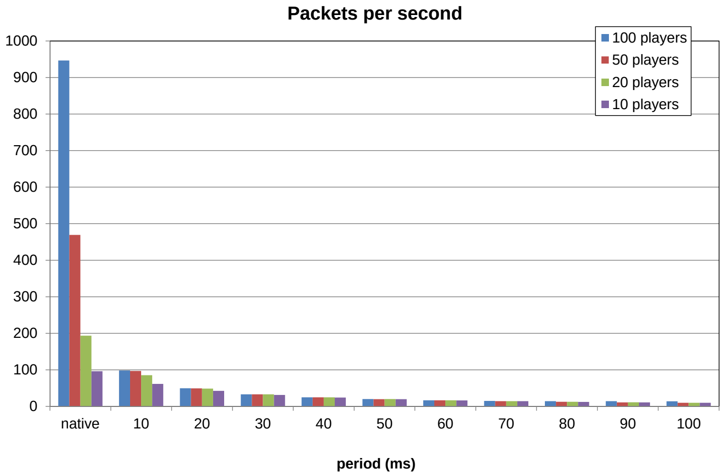


Fig. 16 Packets per second for client-to-server traffic of *WoW*.

If we compare the results with the ones obtained with FPS traffic [10], we can see that higher bandwidth savings can be obtained with MMORPGs, due to the higher compression level of the headers, and also to the presence of TCP ACKs without payload. Regarding the number of flows to multiplex, it is possible to have higher numbers of players, as in these games the scenario is shared by thousands of them.

5.3. Limits Imposed by Subjective Quality requirements

Throughout the paper, we have talked about player's perception of the game. In this subsection we will present some measurements in order to show the limits in which the optimization techniques can be applied while maintaining an acceptable quality. First developed for VoIP [5], subjective quality models have also been proposed for online games. The problem is that each game presents a different behavior with respect to each concrete network parameter [35], since different techniques are used by developers for the concealment of network impairments [8]. As a consequence, each game has to be particularly studied by means of subjective surveys. In this line, we will use the subjective quality model for *WoW* presented in [26].

All the scenarios where traffic optimization has been proposed share a common scheme, similar to the one presented in Fig. 17: packets can be aggregated in a certain part of the network path, but the ingress and the egress of the optimization are never the endpoints themselves, since a single host is not expected to generate a high number of small-packet flows with similar characteristics. As a consequence of this, the multiplexing delay will be seen by the two endpoints as a delay and a jitter added to those already present in the network. If we look at Fig. 4, it is easy to deduce that this multiplexing delay is sawtooth-shaped (Fig. 18), since packets arriving at the beginning of a period will experience a delay equal to *PE*, and packets arriving at the end will experience a very low delay.

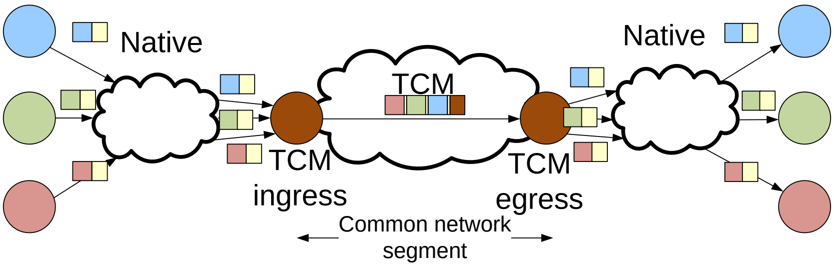


Fig. 17 General scheme of TCM optimization

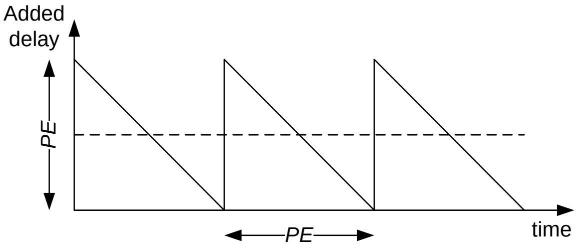


Fig. 18 Sawtooth-shaped delay caused by multiplexing

Thus, the effect of multiplexing can be modelled as the sum of a fixed delay with the value of half the period:

$$delay_{mux} = PE/2 \tag{4}$$

and an additional variable delay, uniformly distributed in the interval (*-PE*/2 , *PE*/2), which standard deviation, as obtained in [14], has the next value:

$$stdev_{mux} = PE/\sqrt{12} \tag{5}$$

In order to have a first idea of the effect of multiplexing on user's experience, we have used the subjective quality model to build Fig. 19. In this case, the MOS has been obtained for different values of network latency, adding the effect of the sawtooth-shaped multiplexing delay. For that aim, we have calculated the sum of $delay_{mux}$ plus the network latency, and we have also added $stdev_{mux}$ to the standard deviation of the network, using root-mean-square for the obtaining of the global jitter. The considered standard deviation of the network delay is 10 ms.

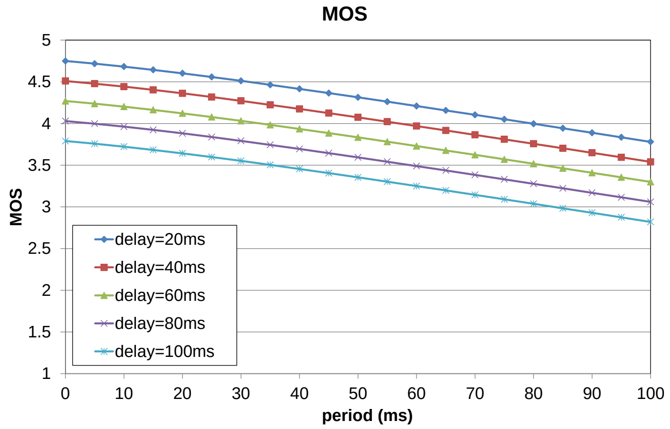


Fig. 19 MOS as a function of network delay and multiplexing period

If we select the value of 3.5 as the threshold for an acceptable MOS, we can see that when network delay is below 40 ms, multiplexing can be applied even with a period of 100 ms. However, if the network latency is 60, 80 or 100 ms, we must use a lower multiplexing period if we want to maintain the subjective quality into acceptable levels. In the previous subsections it has been shown that significant bandwidth savings can be obtained even using a period of 50 or 60 ms, since the savings present an asymptotic behavior and higher periods only provide marginal improvements. All in all, this graph gives us an idea of the values of the period that can be acceptable, so we will use values of the period below 50 ms in subsequent tests.

6. Effect of traffic optimization on TCP behavior

In the previous sections we have explained the multiplexing method, and an estimation of the maximum period to be used has been obtained. The tests have reported significant reductions in terms of bandwidth and packets per second. However, as a counterpart, the packets must be stopped during a short interval in order to get a number of them to be multiplexed together: multiplexing adds a sawtooth-shaped latency, which has been characterized as a fixed delay plus a jitter. In contrast with UDP, TCP is a closed-loop protocol governed by ACK arrivals, so the addition of a new delay will not only have an influence on end-to-end latency, but also on the generation of subsequent packets in the sender.

In this section we want to test the behavior of TCP-based MMORPG flows when competing with other kinds of traffic, as UDP or FTP over different TCP variants. In addition, the traffic will vary depending on the activity deployed by the player: each activity presents a different interactivity level and has its own statistical models (packet size and inter-packet time, for client-server and server-client flows). Taking into account the factors affecting the traffic and the number of parameters involved, we have considered that the best option is to use a simulation tool, able to mimic TCP dynamics, in order to measure how they are affected when a multiplexing delay is added.

We will first explain the traffic model and the simulation scenario. The second subsection is dedicated to the discussion of the figures of merit to be used. Finally, we present the developed tests and discuss the obtained results.

6.1. MMORPG TCP Traffic Generation and Simulation Scenario

In the tests presented in this section we use an NS2 script implementing the traffic model developed in [29]. It is a very complete traffic model including five different player activities (*Trading, Questing, Dungeons, Raiding, Player vs Player Combat*), which change according to different probabilities depending on the hour of the day. The use of this model will allow us to deploy tests comparing the different effect of multiplexing delay depending on the degree of interactivity of the actions the player is performing.

We will only multiplex client-to-server traffic. The reason is twofold: first, client-to-server connections generate smaller packets, which are more suitable to be optimized (see Table 1); second, in many of the scenarios of interest, the uplink is more restrictive than the downlink (*e.g.*, DSL connections).

We have used a dumbbell NS2 simulation scenario (Fig. 20) where a sawtooth-shaped delay is added to *A-A'* flows. The code of the script is available in a public repository[6]. This scenario is typically used when measuring TCP congestion mechanisms (e.g. in [34]), since it allows a number of flows to share a common

[6] https://github.com/josemariasaldana/ns2_scripts

bottleneck. In addition, it would correspond to the scenario shown in Fig. 3, where normal flows share the network with multiplexed ones and with background traffic.

The sawtooth-shaped delay is generated by the dynamic modification of the *O-N* link. Each time a packet appears in this link, a callback function is activated, which modifies the delay of the link, according to the time remaining until the end of the period.

Multiplexing delay only affects packets during a certain part of the network path, namely *O-N*. This fits with the idea that the ingress *(O)* and the egress *(N)* of the optimization tunnel are never located in the endpoints. This scenario will allow us to compare the results obtained by optimized *(A-A')* and non-optimized *(B-B')* flows, in the presence of background traffic *(C-C').*

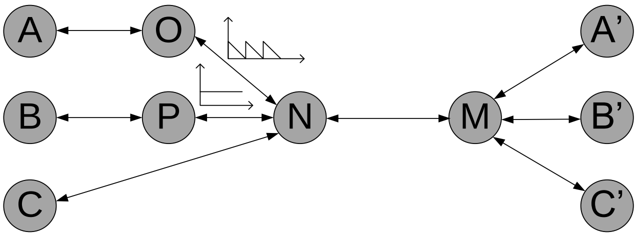


Fig. 20 NS2 simulation scenario

The characteristics of the scenario are:

- A number of MMORPG sessions are established between *A* (clients) and *A'* (servers). The same happens with *B* and *B'*.
- *C-C'* connections are used to create TCP or UDP background traffic.
- By default, the One Way Delay (OWD) of the bottleneck *(N-M)* is set to 20 ms. Links *A-O*, *O-N*, *B-P* and *P-N* have an OWD of 2.5 ms. The rest of the links have a latency of 5 ms. This results in a minimum Round Trip Time (RTT) of 60 ms, which corresponds to an Inter-region typical value [58]. In order to consider other scenarios (*e.g.*, Intra-region), some tests will include different values of the OWD.
- The connection *O-N* adds a sawtooth-shaped multiplexing delay, on the client-to-server direction.
- The default bandwidth of the links is 10 Mbps (a typical value for a local network). The bandwidth of the bottleneck in the uplink is 1 Mbps by default (it would roughly correspond with the capacity of a DSL uplink in many countries). Some tests with higher values are also presented.
- NS2 *FullTCP* is used for the *WoW* flows, since it is bidirectional, to mimic the real traffic of the game, which uses piggybacking, sending the ACKs in data packets [24]. The NS2 parameter *segsperack_* (segments received before generating an ACK) is set to 0 in order to emulate the behavior of the game, which sets to 1 the *PUSH* bit, so as to ask TCP to send the packet as soon as possible.
- TCP *SACK* (Selective Acknowledgment) is used by default for background TCP connections, since it is the most commonly used TCP variant in commercial PCs.
- By default we will use *Questing* traffic, since it is a very common activity in MMORPGs.
- The number of concurrent players in the server is set to 100, using the model in [29].

The simulation tests are repeated three times, and the average values are presented. The 95% confidence intervals are also included.

6.2. Figures of Merit

The proposed simulation setup is used to explore the effect of multiplexing delay on TCP dynamics. The main questions we want to answer are: how does multiplexing delay impair the game traffic? To what extent is fairness between multiplexed and non-multiplexed flows granted? So we will compare flows affected by a sawtooth-shaped delay, against non-delayed flows, sharing the same bottleneck. For that aim, we have first to find some figures of merit able to express our results.

When studying TCP, it is frequent to report the throughput obtained by each flow as the most interesting magnitude, taking into account that the main aim of TCP's initial design is to obtain the maximum possible throughput, while maintaining fairness and avoiding network congestion. As an example, when "RTT unfairness" between different flows is measured [34], the results are mainly reported in terms of throughput.

However, we should remember that in this case we are not studying network-limited flows, in which a certain amount of data has to be transmitted as fast as possible. In contrast, application-limited flows transmit the information while it is generated by the player, so the critical magnitude here is the delay. We cannot forget that we are using TCP for a service in which interactivity does matter, and it may have an influence on the result of the game (players usually talk about "ping" as the RTT). In this sense, the amount of retransmitted packets is also important, since a retransmission causes a significant delay to the game traffic.

As a consequence, we express the results in terms of the next magnitudes:

1) *The RTT parameters estimated by TCP*. In order to govern TCP dynamics (*e.g.*, retransmissions), RTT samples are calculated by TCP and updated frequently, according to network conditions. They are used to compute two different parameters, namely *smoothed RTT* and *RTT variation*, which are subsequently used so as to obtain the value of *Retransmission TimeOut* (RTO) [59]. If the timeout expires, the packet is retransmitted. As a consequence, we will present some of the results in terms of these RTT parameters, taking into account that the multiplexing delay may affect them.

2) *The Retransmission Overhead, i.e., the relationship between the number of bytes generated by the application, and the number of bytes sent by TCP.* As explained in [60], the use of TCP makes it necessary the in-order delivery of packets, so if a packet is lost, subsequent ones will be buffered, and will not be processed by the application until a new copy of the lost one arrives. So it may happen that the information of one of the buffered packets contains a position update of the virtual character, with more recent information than the one in the lost packet. Thus, the loss and the subsequent retransmission of a packet are translated into an additional delay. In order to capture the effect of retransmissions, we will define the *Retransmission Overhead* (*RO*) as the relationship between the amount of bytes sent by TCP retransmissions (headers and ACKs are not counted) and the amount of bytes generated by the application:

$$RO = \frac{bytes_{TCP\_retrans}}{bytes_{application}} = \frac{bytes_{TCP}}{bytes_{application}} - 1 \qquad (6)$$

As an example, if 1,000 bytes are generated by the application, and 1,100 bytes are finally sent in TCP payloads, then the value of *RO* is 0.1 (*i.e.*, 10 % overhead).

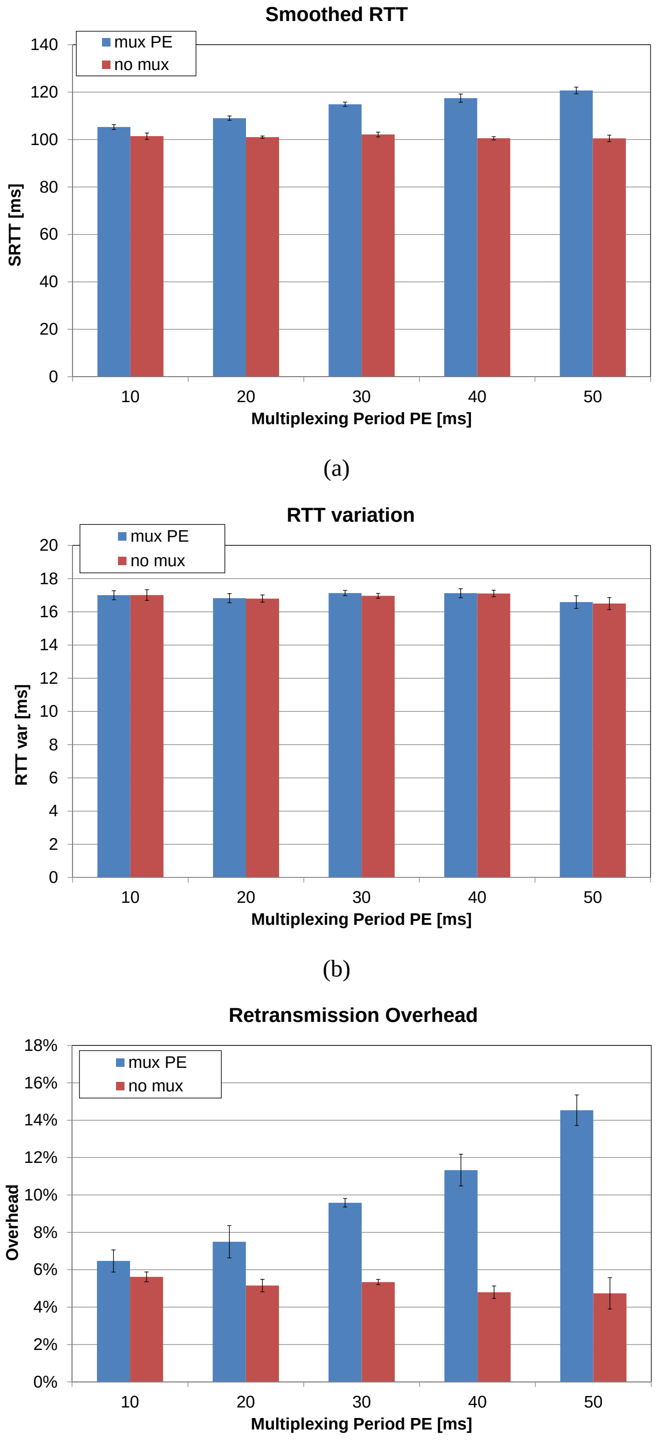
6.3. Deployed tests

6.3.1. Multiplexed and non-multiplexed MMORPG Flows, with FTP Background Traffic

In these tests we want to capture the effect of the sawtooth-shaped multiplexing delay when multiplexed and non-multiplexed MMORPG flows compete in the presence of FTP background traffic. For that aim, 50 game sessions are established between *A* and *A'*, experiencing a multiplexing delay with a period *PE*. Non-multiplexed game sessions (50) are established between *B* and *B'*; finally, one background FTP connection, which tries to get as much throughput as possible, is established from *C* to *C'* (uplink). The simulation time is 200 sec. The values of RTT parameters are calculated this way: the average value of each parameter is first calculated for each flow, between sec. 100 and 200. Then, the average value of the results obtained for each flow is presented.

In Fig. 21 a), we see that the difference between multiplexed and non-multiplexed flows can be appreciated in terms of *smoothed RTT*, since an average extra delay of roughly *PE*/2 is experienced by the multiplexed flows. However, the difference in terms of *RTT variation* is negligible (Fig. 21 b), which means that TCP does not detect a higher value of the jitter in the flows affected by the multiplexing delay.

This increase in the *RTT* is translated into a higher *Retransmission Overhead* (Fig. 21 c) for the multiplexed flows. As a first result, we see that for the lowest values of *PE,* the overhead is not very significant, and the differences are small. However, as the multiplexing period grows, the unfairness between multiplexed and non-multiplexed flows gets more significant. This yields a first conclusion: if the value of *PE* is 30 ms or higher, the unfairness may be noticed by the players.



(a)

(b)

(c)

Fig. 21 *Smoothed RTT*, *RTT variation* and *Retransmission Overhead*, with multiplexing delay *PE* in *A-A'*, and an FTP connection *C-C'*

6.3.2. Influence of TCP variants for background traffic

In this subsection the influence of the TCP variants, used by the background traffic, on the MMORPG flows is explored. Each TCP variant includes a set of different features which makes it more or less "friendly" when coexisting with MMORPG flows. In a previous paper [29] this coexistence was explored, but here we also consider traffic multiplexing as another variable of the problem. The main question we want to answer is: does the TCP variant used for background traffic have any influence on the unfairness between multiplexed and non-multiplexed flows?

For this aim, the tests presented in the previous section are repeated, using other TCP variants (namely *Tahoe, Reno, New Reno* and *Vegas)*, in addition to the one used by default *(SACK)*, for the background FTP traffic. The results, in terms of overhead, are presented in Fig. 22. For the sake of clarity, only the results with *PE*=10 and 30 ms are presented.

When the period is small (Fig. 22, left group), the unfairness between flows is also small, and very similar to the results obtained with TCP *SACK*. The highest difference is obtained with TCP *Tahoe* and *New Reno*, but it only rises to 1 %. TCP *Vegas* does not add any overhead to the game flows: as remarked in [29], its "timid" behavior prevents it from increasing its sending window, thus leaving bandwidth enough for the MMORPG flows, be them multiplexed or not. If the period is higher (Fig. 22, right group), it can be observed that TCP *Reno* behaves in a very similar way to *SACK*, whereas *Tahoe* and *New Reno* tend to stress the unfairness between multiplexed and non-multiplexed flows.

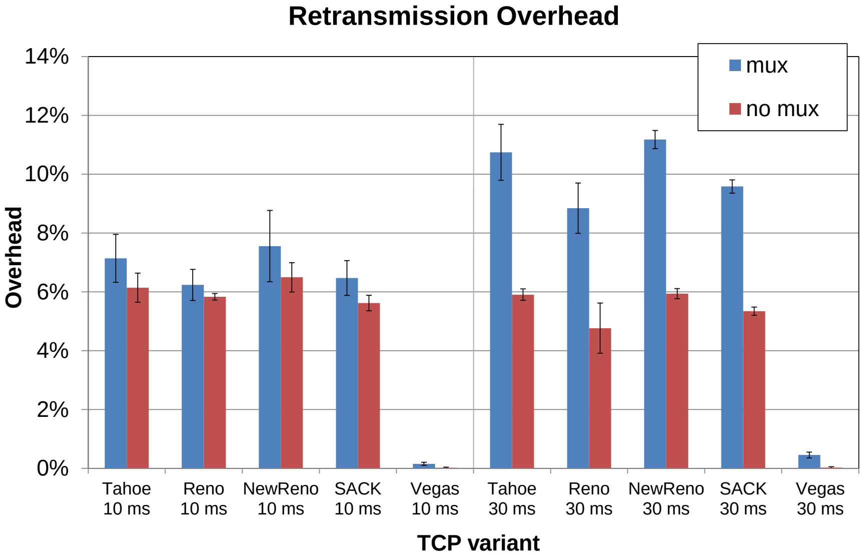


Fig. 22 *Retransmission Overhead* for different TCP variants with *PE*=10 and 30 ms

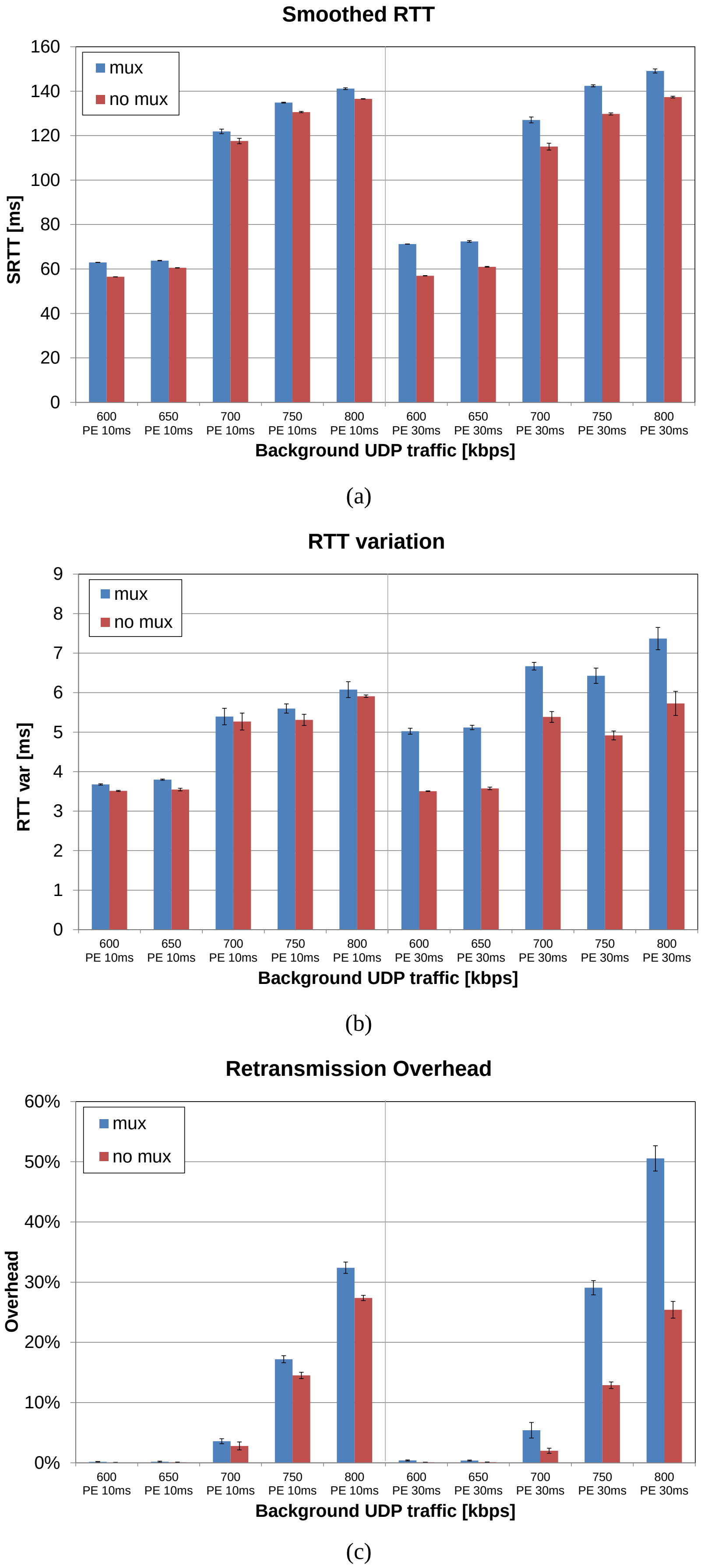
6.3.3. Multiplexed and non-multiplexed MMORPG Flows, with UDP Background Traffic

In this subsection we study the interactions between competing multiplexed and non-multiplexed game flows, in the presence of Constant Bit Rate (CBR) UDP traffic. For that aim, 50 MMORPG multiplexed flows are established between *A* and *A'*; 50 non-multiplexed flows are set between *B* and *B'*; and different amounts of UDP traffic are sent from *C* to *C'*, following this packet size distribution: 50 % of the packets are of 40 bytes; 10 % are of 576 bytes; and the remaining 40 % are 1,500 bytes packets [61].

We have set *PE* to 30 ms for all the tests, and we will vary the amount of background traffic. It should be noticed that the aggregate bandwidth of the MMORPG flows is roughly 350 kbps, so congestion may appear when background traffic approaches is above 650 kbps (the bandwidth in the uplink is 1 Mbps).

The results using *PE*=10 ms and 30 ms are shown in Fig. 23. In contrast to what happened with FTP, the CBR background traffic does not reduce its rate as a consequence of congestion. In terms of *smoothed RTT* (Fig. 23 a), the differences between multiplexed and non-multiplexed flows are not very different from the results obtained with FTP (compare with the third column set of Fig. 20 a). However, the difference between multiplexed and non-multiplexed flows is stressed in terms of *RTT variation* (Fig. 23 b): it can be seen that it was negligible when FTP was used (Fig. 20 b), but it starts to be noticeable when *PE*=10 ms in this case, and it becomes significant for *PE*=30ms. The cause of this is the inelastic behavior of UDP: it does not adapt its rate, whereas TCP does.

In normal conditions, the overhead remains low (see the 600 and 650 kbps column sets of Fig. 23 c), but when the offered traffic gets above the bottleneck capacity (roughly 700 kbps), the overhead rises to very high values, since the CBR flows do not reduce their sending rate. When the period is small (left group of columns), although the overhead is high, this is not translated into a noticeable unfairness, but if the period is long (right group of columns), the overhead rises up to 50 % for multiplexed flows, while it remains significantly lower for non-multiplexed flows. We can conclude that using high values for the multiplexing period may produce unfairness with respect to non-multiplexed flows in case of severe network congestion.



(a)

(b)

(c)

Fig. 23 *Smoothed RTT*, *RTT variation* and *Retransmission Overhead* with multiplexing delay *PE*=10 and 30 ms in *A-A'*, and different amounts of UDP traffic in *C-C'*

6.3.4. Influence of Network Parameters

The simulation setup considers a default value of 20 ms for the OWD of the network and 1 Mbps as the bandwidth of the uplink of the bottleneck. In this subsection, the influence of these two parameters (network delay and bandwidth) will be explored with some new tests. In this subsection, the value of the multiplexing period *PE* is always set to 30 ms.

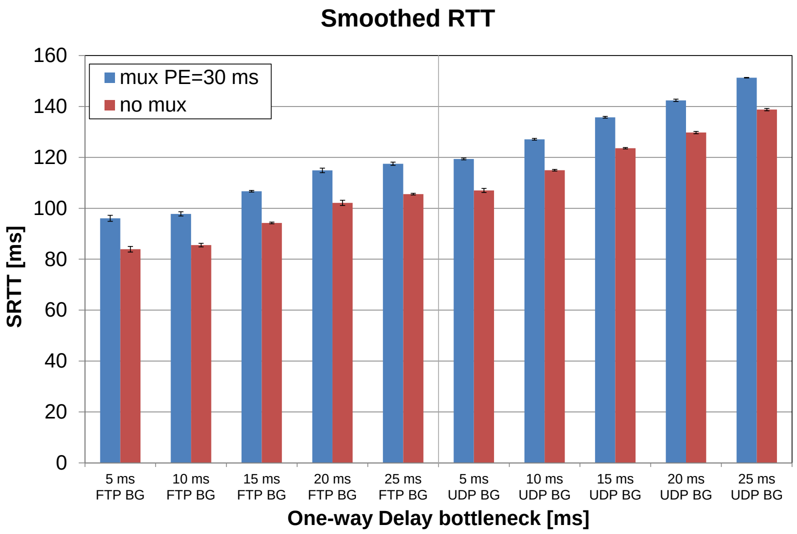
a) Influence of network delay

First of all, different values for the OWD of the bottleneck have been set. The default value is 20 ms for both directions, which results in an RTT of 60 ms. Different values, ranging from 5 to 25 ms have been set in both directions. Fig. 24 shows the results when the game flows share the bottleneck with a single FTP connection, or with a fixed amount of UDP in the uplink. In these tests the amount of UDP traffic is always 750 kbps.

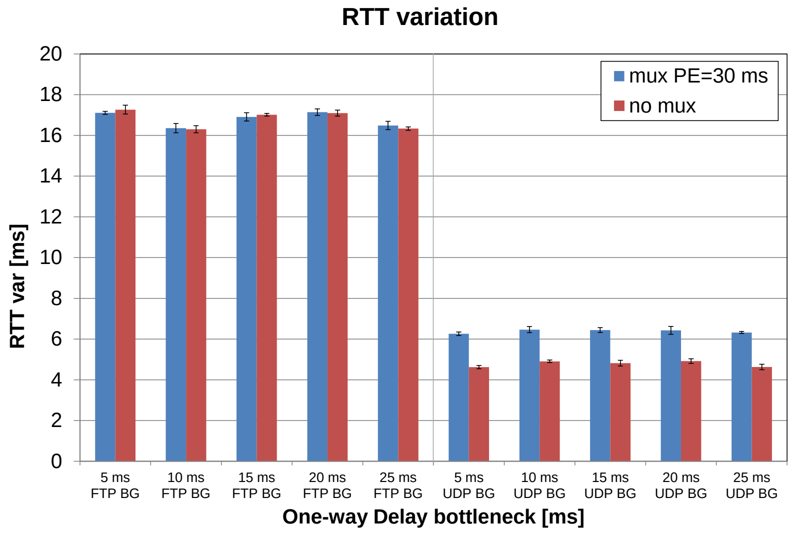
The *smoothed RTT* varies according to the increased latency, as shown in Fig. 24 a), but the unfairness between multiplexed and not multiplexed flows is not modified. This happens in a similar way with FTP and UDP background traffic. The *smoothed RTT* increases linearly with the OWD, and the difference between multiplexed and non-multiplexed flows remains constant.

The *RTT variation* is not affected when the background traffic is FTP, (Fig. 24 b). However, if a fixed amount of UDP traffic is present, the disadvantage of multiplexed flows increases. But the difference shows a non-variant behavior with the OWD.

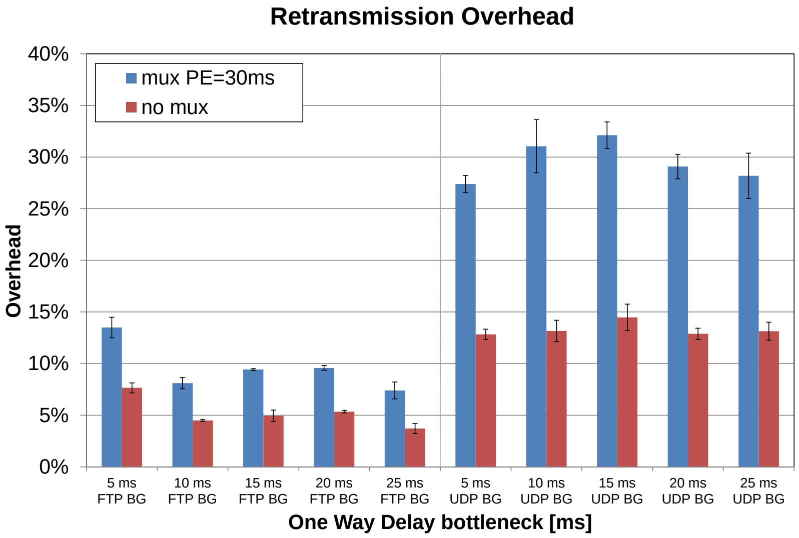
Regarding the *Retransmission Overhead* (Fig. 24 c) with an FTP background flow, it can be observed that the relationship between multiplexed and not multiplexed flows remains constant, but *RO* is significantly higher when network latency is very low (5 ms). The cause of this is that the low latency allows the MMORPG connection to increase its sending window faster, thus resulting in a higher packet loss probability for the game flows. The packet loss rate is above 6% when the OWD is 5ms, but it is about 3% if the OWD is set to 25ms. When UDP background traffic is present, the variations of the *Retransmission Overhead* with OWD are very slight. The cause of this invariant behavior is that the background UDP traffic remains constant despite the variations of the network latency. Regarding the unfairness between multiplexed and non-multiplexed flows, it can be observed that it does not change significantly with the value of the network delay.



(a)



(b)



(c)

Fig. 24 *Smoothed RTT*, *RTT variation* and *Retransmission Overhead* with multiplexing delay *PE*=30ms in *A-A'*, and different values for the OWD, when the bottleneck is shared with an FTP connection, or with with 750 kbps of upload UDP traffic in the uplink *C-C'*.

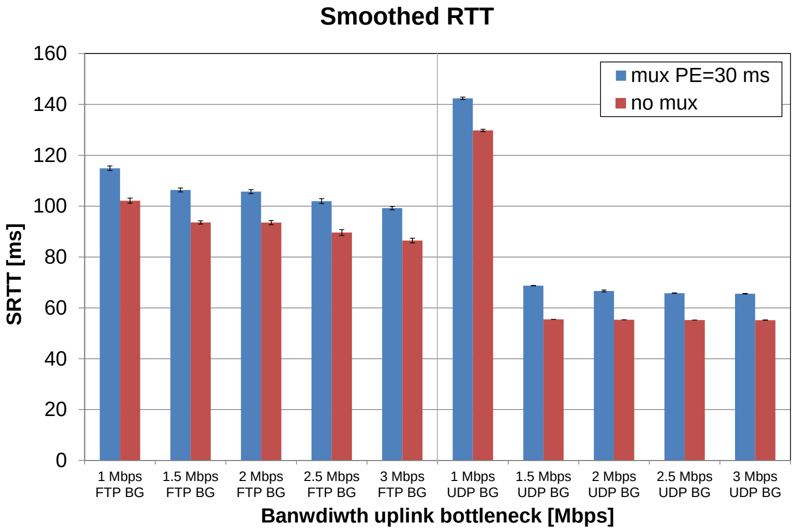
b) Influence of network bandwidth

The aim of this subsection is to measure the effect of the available bandwidth on the unfairness between multiplexed and non-multiplexed flows. Different tests are carried out, in scenarios where background traffic shares the bottleneck with 50 multiplexed and 50 native MMORPG connections. An FTP connection, and a flow of 750 kbps of UDP traffic are used as background traffic. Different values for the bandwidth of the uplink have been set, ranging from 1 to 3 Mbps. The multiplexing period is always 30 ms, and the OWD of the bottleneck is the default one (20 ms).

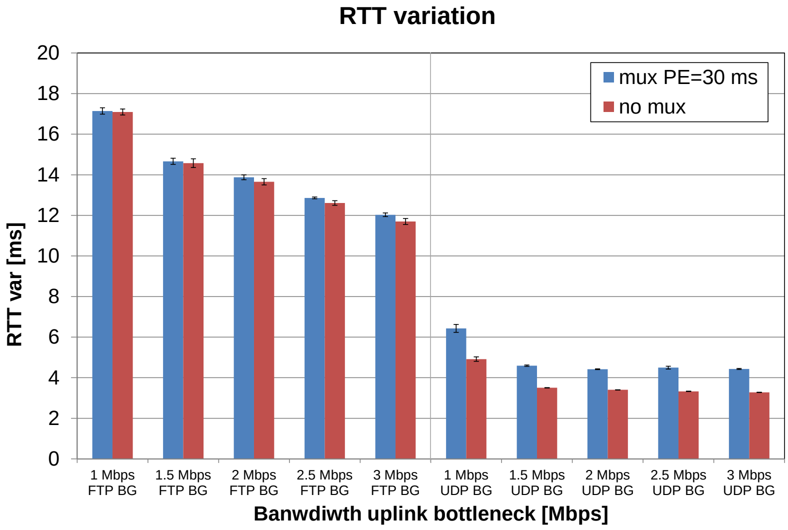
As shown in Fig. 25 a, when a background FTP connection is present, it can be seen that the bandwidth increase is not translated into a significant reduction of the *smoothed RTT*. The cause is that the FTP background traffic gets all the bandwidth left by the MMORPG flows. The unfairness is maintained: the multiplexed flows always experience a 15 ms higher RTT (half the period). When background traffic is UDP, as soon as the bandwidth is significantly higher than the sum of the MMORPG (roughly 350 kbps) and background traffic (750 kbps), the *smoothed RTT* is roughly the RTT (60 ms)

For FTP background traffic, the *RTT variation* gets reduced (Fig. 25 b) as the available bandwidth increases, and this provokes a reduction of the *Retransmission Overhead* (Fig. 25 c). At the same time, the bandwidth increase does not totally avoid retransmissions, since the TCP background traffic tends to use all the available bandwidth. However, the unfairness in terms of *RO* is maintained, being the multiplexed flows the ones with a higher packet loss rate.

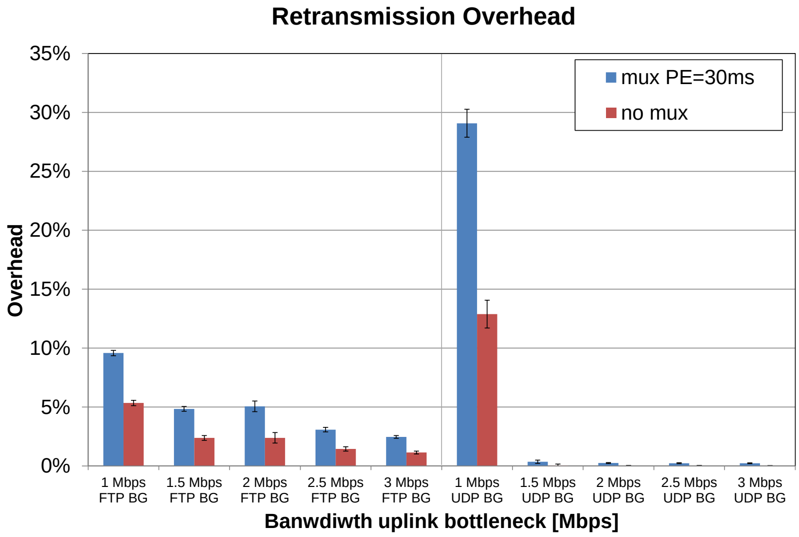
When UDP traffic is present, the variation of the RTT (Fig. 25 b) is tiny (about 4 ms). The unfairness between multiplexed and non-multiplexed flows is again constant despite the changes of the network bandwidth. In addition, the *Retransmission Overhead* (Fig. 25 c) becomes negligible as soon as the bandwidth is enough (more than 1 Mbps), because no packets are lost in the queues.



(a)



(b)



(c)

Fig. 25 *Smoothed RTT*, *RTT variation* and *Retransmission Overhead* with multiplexing delay *PE*=30ms in *A-A'*, and different values for the uplink bandwidth, when the bottleneck is shared with an FTP connection, or with with 750 kbps of upload UDP traffic in the uplink *C-C'*.

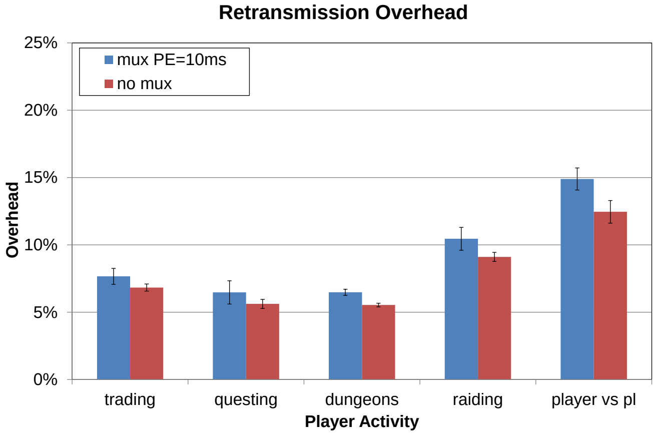
6.3.5. Dependence with Player Activities

All the previous tests have been carried out using the traffic model corresponding to *Questing* activity. However, an MMORPG game may generate very different traffic patterns depending on the action that the player is performing at a certain moment. Thus, in this subsection we present some tests with the aim of studying the different influence of multiplexing, depending on the game activity.

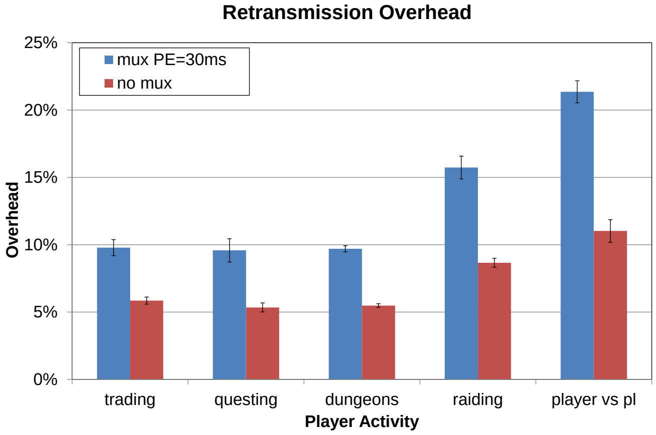
We have used the same scheme as in previous subsections: 50 *A-A'* multiplexed flows share the bottleneck with 50 *B-B'* flows and an FTP upload connection. The statistics of the game flows correspond to *Trading*, *Questing*, *Dungeons*, *Raiding* and *Player vs Player Combat.* The results in terms of overhead are shown in Fig. 26 for *PE*=10 and 30 ms.

In Fig. 26 a), it can be observed that the effect of multiplexing is really low when the period is small (10 ms). In fact, the difference between multiplexed and non-multiplexed flows is less than 1 % for *Trading,* where the player mainly acts alone; it becomes higher for *Questing* and *Dungeons,* where the number of players is low. The effect becomes more noticeable when the interactivity of the game is higher, as it happens in *Player vs. Player* and *Raiding.* In these two activities, the amount of packets per second generated is higher, since a number of players are involved in the action, frequently organized in different teams who have to cooperate or to fight each other in a limited virtual space.

The differences are stressed when the period is higher (30 ms), as shown in Fig. 26 b): a noticeable difference appears in terms of overhead. In *Trading, Questing* and *Dungeons* the difference is now below 5 %, but the unfairness between multiplexed and non-multiplexed flows is higher for the other two activities.



(a)



(b)

Fig. 26 *Retransmission Overhead* for different activities with *a) PE*=10ms; *b) PE*=30 ms

Finally, Fig. 27 represents the quotient between the overhead of multiplexed and non-multiplexed flows. When the period is low (10 ms), only small differences appear, but if a long period is used (50 ms), the overhead of multiplexed flows can be three times as bigger as for non-multiplexed ones. Thus, we confirm the conclusion of subsection 6.3.1, *i.e.*, long multiplexing periods stress the unfairness with respect to non-multiplexed flows. At the same time, it can be seen that the observed tendency is very similar for all the activities, since big differences cannot be observed between them.

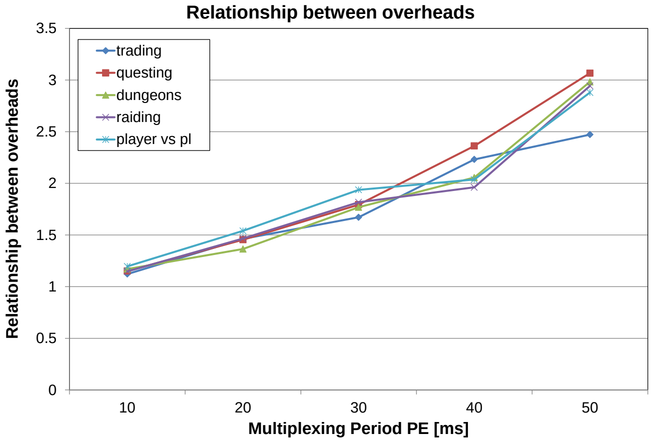


Fig. 27 Quotient between retransmission overhead using different activities and multiplexing periods.

6.4 Discussion of the Results

A number of tests have been deployed to study the limits and the factors that may have an influence on the unfairness between multiplexed and non-multiplexed flows sharing a common bottleneck. Unfairness between players sharing the same virtual world is a problem, since it may reduce the engagement of the impaired players with the game. But the quality experienced by the rest of the players is also reduced, since their game mates become less responsive, and cooperation with skilled avatars, and competition with "powerful" enemies is an essential part of the game experience. In addition, some games employ "lag compensation" mechanisms that may artificially increase the delay of the players with lower RTT, in order to make the game fairer [8]. As a result, the slowest player may slow down the game for all.

For that aim, the NS2 scenario has been used in order to check different factors: the effect of FTP background traffic, using different TCP variants; the effect of inelastic UDP background traffic; the effect of network parameters; finally, the influence of player activities has also been tackled. Although in subsection 5.3, the results indicated that a period of 50 or 60 ms may be tolerable for the players in terms of subjective quality, these values may produce an unfair situation for the players whose traffic is multiplexed. As a first conclusion, we have seen that the unfairness can be low (probably negligible for the players) if the value of the multiplexing period *PE* is kept under 10 or 20 ms.

Regarding the effect of the different TCP variants used by the background traffic, it has been shown that TCP *SACK* and *Reno* behave better than *Tahoe* and *New Reno*, and are able to grant a higher fairness between multiplexed and non-multiplexed flows. As expected [29], *Vegas* is the most “friendly” TCP variant when coexisting with MMORPG flows.

If the background traffic is inelastic (UDP), and the total amount of traffic makes the link become congested, the use of high values for the multiplexing period may produce unfairness with respect to non-multiplexed flows. Since the flows do not fold back, this effect is worse than that observed with TCP.

The effect of network parameters has also been tested. It has been observed that the network latency has a stronger influence if the background traffic is TCP-based. Since TCP depends on ACKs for controlling his sending rate, a very low network latency makes FTP increase its throughput, resulting in a higher packet loss rate for the MMORPG flows. However, when background traffic is based on UDP, the network latency has a minor influence, since the rate of the background traffic remains constant, so only the MMORPG flows are affected. However, it has been observed that the increase of network delay does not have a significant impact on the unfairness between optimized and non-optimized flows.

It has also been shown that the increase of the available bandwidth is more beneficial when no TCP background traffic is present. However, the unfairness between multiplexed and non-multiplexed flows is not affected by bandwidth increase. The TCP mechanisms, when used by network-limited flows, tend to get all the available bandwidth, and this is translated into an unavoidable *Retransmission Overhead* for the MMORPG flows, even when bandwidth capacity is significantly higher than the throughput required by the game flows. However, when only UDP flows are present in the background traffic, the bandwidth increase is translated into a very significant reduction of the retransmissions caused by packet losses.

Some differences have been observed depending on the activities deployed by the player: the activities in which the player is alone *(e.g., Trading)* or interacts with a reduced number of players *(e.g., Dungeons)*, only experience a slight increase in terms of RTT for the multiplexed flows. In terms of overhead, the difference is below 5 % for *Trading, Questing* and *Dungeons*, but the unfairness between multiplexed and non-multiplexed flows is very high for the other two activities, *i.e.,* the most interactive ones *(Raiding* and *Player vs. Player)*.

7. Conclusions

A traffic optimization method based on multiplexing, tunneling and header compression has been applied to the traffic of MMORPGs, in scenarios where a number of flows share a common network path. The issues derived from the use of TCP for an interactive service have been identified and studied. The optimization method has been explained in detail, and an analytical formula of the expected savings has been obtained, showing that there is an asymptote that establishes an upper bound for the bandwidth reduction. Values near the asymptote can be reached if the number of flows to multiplex is high enough.

The behavior of the header compression algorithm has been studied in order to obtain the header compression ratio, using real traces of a popular game *(World of Warcraft)*. Next, a statistical model of the game has been used to calculate the average bandwidth saving as a function of the number of players and the multiplexing period, obtaining very similar results to the analytical ones. The achievable savings are significant, and can be about 60 % for IPv4 and 70 % for IPv6. An important reduction in the amount of packets per second is also observed.

A multiplexing delay is required as a counterpart of bandwidth savings when using traffic optimization. This delay has been characterized as a constant latency and a jitter. A MOS model from the literature has been employed in order to calculate the values of the multiplexing period that can be used without harming the user experience. An NS2 simulation scenario has been created in order to study the interactions between multiplexed and non-multiplexed flows, in the presence of background traffic sharing a bottleneck. The performance of the flows has been compared in terms of RTT and retransmission overhead. Different tests using FTP over different TCP variants, and UDP background traffic have been deployed, showing the different impairments caused by traffic optimization.

Some recommended limits for the multiplexing period have been found: the unfairness can be low if the value of the multiplexing period *PE* is kept under 10 or 20 ms. TCP *SACK* and *Reno* in the background have yielded better results, in terms of fairness, than *Tahoe* and *New Reno*. When UDP is used for background traffic, it has been shown that high values of the multiplexing period may stress the unfairness between flows if network congestion is severe. The effect of network latency and bandwidth have also been measured, and it has been observed that the increase of network delay or bandwidth do not have a significant impact on the unfairness between optimized and non-optimized flows. Finally, the effect of multiplexing has been evaluated, depending on the different activities that the player can perform in the game. It is shown that the multiplexing unfairness is higher for the activities where a large number of players participate together.

ACKNOWLEDGMENTS

This work has been partially financed by the European Social Fund in collaboration with the Government of Aragon; and TAMA Project, Government of Aragon.

REFERENCES

[1] Huawei, Smartphone Solutions White Paper, Issue 2, 2012.07.17.

Available online: http://www.huawei.com/ilink/en/download/HW_193034, [Accessed Apr 2014].

[2] Perkins C (2003) RTP: Audio and Video for the Internet. Addison-Wesley Professional.

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

Available online: http://tools.ietf.org/html/rfc4170, [Accessed Apr 2014].

[4] Saldana J, Fernández-Navajas J, Ruiz-Mas J, Murillo J, Viruete Navarro E, Aznar JI (2012) Evaluating the influence of multiplexing schemes and buffer implementation on perceived VoIP conversation quality. Computer Networks, vol. 56, no. 7: 1893-1919. doi: 10.1016/j.comnet.2012.02.004

[5] The E-model, a computational model for use in transmission planning, ITU-T Recommendation G.107, March 2003.

Available online: http://www.itu.int/rec/T-REC-G.107, [Accessed Apr 2014]

[6] Feng W, Chang F, Feng W, Walpole J (2005) A Traffic Characterization of Popular On-Line Games. IEEE/ACM Trans. Netw. 13, 3: 488-500.

DOI: 10.1109/TNET.2005.850221

[7] Kaiser A, Maggiorini D, Boussetta K, Achir N (2009) On the Objective Evaluation of Real-Time Networked Games. Proc. IEEE Global Telecommunications Conference GLOBECOM. doi: 10.1109/GLOCOM.2009.5426032

[8] Oliveira M, Henderson T (2003) What online gamers really think of the Internet?. In: Proc. 2nd workshop on Network and system support for games (NetGames '03). ACM, New York, NY, USA: 185-193. doi: 10.1145/963900.963918

[9] J. Saldana *et al.* (2014) Tunneling Compressed Multiplexed Traffic Flows (TCMTF), draft-saldana-tsvwg-tcmtf-07.

Available online: http://datatracker.ietf.org/doc/draft-saldana-tsvwg-tcmtf/, [Accessed Sep 2014].

[10] Saldana J, Fernandez-Navajas J, Ruiz-Mas J, Aznar JI, Viruete E, Casadesus L (2011) First Person Shooters: Can a Smarter Network Save Bandwidth without Annoying the Players?. IEEE Communications Magazine, vol. 49, no.11: 190-198. doi: 10.1109/MCOM.2013.6658664

[11] Ratti S, Hariri B, Shirmohammadi S (2010) A Survey of First-Person Shooter Gaming Traffic on the Internet. IEEE Internet Computing, vol. 14, 5: 60-69. doi: 10.1109/MIC.2010.57

[12] Suznjevic M, Stupar I, Matijasevic M (2012) A model and software architecture for MMORPG traffic generation based on player behavior. Multimedia Systems vol. 19, 3: 231-253. doi: 10.1007/s00530-012-0269-x

[13] Saldana J, Fernandez-Navajas J, Ruiz-Mas J, Aznar JI, Casadesus L, Viruete E (2011) Comparative of Multiplexing Policies for Online Gaming in terms of QoS Parameters, IEEE Communications Letters, vol.15, no.10: 1132-1135. doi: 10.1109/LCOMM.2011.080811.111160

[14] Saldana J, Fernandez-Navajas J, Ruiz-Mas J, Viruete Navarro E, Casadesus L (2012) Online FPS Games: Effect of Router Buffer and Multiplexing Techniques on Subjective Quality Estimators, Multimedia Tools and Applications, Vol. 71, 3: 1823-1856. doi: 10.1007/s11042-012-1309-4

[15] Wattimena AF, Kooij RE, van Vugt JM, Ahmed OK (2006) Predicting the perceived quality of a first person shooter: the Quake IV G-model, Proc. NETGAMES'2006. ACM, New York, NY, USA, Article 42. doi: 10.1145/1230040.1230052

[16] Newzoo, Free Global Trend Report 2012-2016.

Available online: http://www.newzoo.com/wp-content/uploads/2011/06/Newzoo_Free_Global_Trend_Report_2012_2016_V2.pdf, [Accessed Apr 2014]

[17] CNet, World of Warcraft subscriber base hits 12 million.

http://www.cnet.com/news/world-of-warcraft-subscriber-base-hits-12-million/, [Accessed Apr 2014].

[18] Chen K, Huang P, Lei C (2006) Game traffic analysis, an MMORPG perspective. Computer Networks, Vol. 50, Issue 16: 3002-3023. doi: 10.1016/j.comnet.2005.11.005

[19] Griwodz C, Halvorsen P (2006) The fun of using TCP for an MMORPG. Proc. international workshop on Network and operating systems support for digital audio and video (NOSSDAV '06). ACM, New York, NY, USA. doi: 10.1145/1378191.1378193

[20] Wu CC, Chen KT, Chen CM, Huang P, Lei CL (2009) On the Challenge and Design of Transport Protocols for MMORPGs, Multimedia Tools and Applications, Vol. 45, No. 1: 7-32. doi: 10.1007/s11042-009-0297-5

[21] Fritsch T, Ritter H, Schiller J (2005) The effect of latency and network limitations on MMORPGs: a field study of everquest2. Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games (NetGames '05). ACM, New York, NY, USA: 1-9. doi: 10.1145/1103599.1103623

[22] Suznjevic M, Dobrijevic O, Matijasevic M (2009) MMORPG Player actions: Network performance, session patterns and latency requirements analysis, Multimedia Tools Appl. 45, 1-3: 191-214. doi: 10.1007/s11042-009-0300-1

[23] Suznjevic M, Dobrijevic O, Matijasevic M (2009) Hack, slash, and chat: a study of players' behavior and communication in MMORPGs. Proceedings 8th Annual Workshop on Network and Systems Support for Games (NetGames '09). IEEE Press, Piscataway, NJ, USA, Article 2, 6 pages. doi: 10.1109/NETGAMES.2009.5446235

[24] Svoboda P, Karner W, Rupp M (2007) Traffic Analysis and Modeling for World of Warcraft, in Proc. ICC' 07, Urbana-Champaign, IL, USA. doi: 10.1109/ICC.2007.270

[25] Miller JL, Crowcroft J (2010) The near-term feasibility of P2P MMOG's, IEEE 9th Annual Workshop on. Network and Systems Support for Games (NetGames). doi: 10.1109/NETGAMES.2010.5679578

[26] Ries M, Svoboda P, Rupp M (2008) Empirical study of subjective quality for Massive Multiplayer Games. Proc. 15th Int. Conf. on Systems, Signals and Image Processing: 181-184. doi: 10.1109/IWSSIP.2008.4604397

[27] Che X, Ip B (2012) Packet-level traffic analysis of online games from the genre characteristics perspective, Journal of Network and Computer Applications, Vol. 35, No. 1: 240–252. doi: 10.1016/j.jnca.2011.08.005

[28] Suznjevic M, Matijasevic M (2012) Player behavior and traffic characterization for MMORPGs: A survey, Multimedia Systems, Vol. 19, No. 3: 199-220. doi: 10.1007/s00530-012-0270-4

[29] Suznjevic M, Saldana J, Matijasevic M, Fernandez-Navajas J, Ruiz-Mas J (2014) Analyzing the effect of TCP and server population on massively multiplayer games, International Journal of Computer Games Technology, Article ID 602403, 17 pages. doi: 10.1155/2014/602403

[30] Zhuang X, Bharambe A, Pang J, Seshan S (2007) Player Dynamics in Massively Multiplayer Online Games, School of Computer Science, Carnegie Mellon University, Pittsburgh, Tech. Rep. CMU-CS-07-158.

Available online: http://ra.adm.cs.cmu.edu/anon/usr/ftp/home/ftp/2007/CMU-CS-07-158.pdf, [Accessed Apr 2014]

[31] Avallone S, Guadagno S, Emma D, Pescapè A, Ventre G (2004) D-ITG distributed internet traffic generator. Proceedings First IEEE International Conference on the Quantitative Evaluation of Systems: 316-317. doi: 10.1109/QEST.2004.1348045

[32] Suznjevic M, Stupar I, Matijasevic M (2011) Traffic modeling of player action categories in a MMORPG, In Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques (SIMUTools '11). ICST, Brussels, Belgium: 280-287.

DOI: 10.4108/icst.simutools.2011.245546

[33] De Vleeschauwer B, Van Den Bossche B, Verdickt T, De Turck F, Dhoedt B, Demeester P (2005) Dynamic microcell assignment for massively multiplayer online gaming, Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games (NetGames '05), pp. 1–7, Hawthorne, NY, USA. doi: 10.1145/1103599.1103611

[34] Marfia G, Palazzi CE, Pau G, Gerla M, Roccetti M (2010) TCP Libra: Derivation, analysis, and comparison with other RTT-fair TCPs. Computer Networks 54, 14: 2327-2344. doi: 10.1016/j.comnet.2010.02.014

[35] Zander S, Armitage G (2004) Empirically Measuring the QoS Sensitivity of Interactive Online Game Players. Australian Telecommunications Networks & Applications Conference (ATNAC2004), Sydney, Australia.

Available online: http://caia.swin.edu.au/pubs/ATNAC04/zander-armitage-ATNAC2004.pdf, [Accessed Apr 2014]

[36] Cameron P, Crocker D, Cohen D, Postel J (1994) RFC 1692: Transport Multiplexing Protocol (TMux). Available online: https://tools.ietf.org/html/rfc1692, [Accessed May 2015].

[37] Zorzi M, Rao R. R. (1999) Perspectives an the impact of error statistics on protocols for wireless networks. Personal Communications, IEEE, 6(5), 32-40. doi: 10.1109/98.799618

[38] Francis B, Narasimhan V, Nayak A, Stojmenovic I. (2012) Techniques for Enhancing TCP Performance in Wireless Networks, Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on, pp.222,230, 18-21 June 2012. doi: 10.1109/ICDCSW.2012.29

[39] Ertekin E, Christou C (2004) Internet protocol header compression, robust header compression, and their applicability in the global information grid, IEEE Communications Magazine, vol. 42: 106-116. doi: 10.1109/MCOM.2004.1362553

[40] Jacobson V (1990) RFC 1144: Compressing TCP/IP Headers for Low-Speed Serial Links. Available online: http://tools.ietf.org/html/rfc1144, [Accessed Apr 2014].

[41] Degermark M, Nordgren B, Pink D (1999) RFC 2507: IP Header Compression,.

Available online: https://tools.ietf.org/html/rfc2507

[42] Casner S et al (1999) RFC 2508: Compressing IP/UDP/RTP Headers for Low-Speed Serial Links.

Available online: https://tools.ietf.org/html/rfc2508, [Accessed Apr 2014].

[43] Koren T et al.(2003) RFC 3545: Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering.

Available online: https://tools.ietf.org/html/rfc3545, [Accessed Apr 2014].

[44] Sandlund K, Pelletier G, Jonsson LE (2010) RFC 5795: The RObust Header Compression (ROHC) Framework.

Available online: http://tools.ietf.org/html/rfc5795, [Accessed Apr 2014].

[45] Saldana J, Fernandez-Navajas J, Ruiz-Mas J, Wing D, Perumal AM, Ramalho M, Camarillo G, Pascual F, Lopez DR, Nunez M, Florez D, Castell JA, de Cola T, Berioli M (2013) Emerging Real-time Services: Optimizing Traffic by Smart Cooperation in the Network. IEEE Communications Magazine, Vol. 51, n. 11: 127-136. doi: 10.1109/MCOM.2013.6658664

[46] Mauve M, Fischer S, Widmer J (2002) A Generic Proxy System for Networked Computer Games. In Proceedings of the 1st workshop on Network and system support for games (NetGames'02): 25-28. ACM, New York doi: 10.1145/566500.566504

[47] Bauer D, Rooney S, Scotton P (2002) Network Infrastructure for Massively Distributed Games. In Proceedings 1st workshop on Network and system support for games (NetGames'02): 36-43 ACM, New York doi: 10.1145/566500.566506

[48] Majewski C, Griwodz C, Halvorsen P (2006) Translating latency requirements into resource requirements for game traffic, Proceedings of the International Network Conference (INC): 113–120, Plymouth, UK.

Available online: http://heim.ifi.uio.no/griff/papers/inc2006a.pdf, [Accessed Apr 2014]

[49] Pereira RM, Tarouco LM (2009) Adaptive Multiplexing Based on E-model for Reducing Network Overhead in Voice over IP Security Ensuring Conversation Quality. Proc. Fourth international Conference on Digital Telecommunications, Washington, DC: 53-58. doi: 10.1109/ICDT.2009.17

[50] Furuholt B, Kristiansen S, Wahid F (2008) Gaming or gaining? Comparing the use of Internet cafés in Indonesia and Tanzania, The International Information & Library Review, 40(2) 129-139. doi: 10.1016/j.iilr.2008.02.001

[51] Batool SH, Mahmood K (2010) Entertainment, communication or academic use? A survey of Internet cafe users in Lahore, Pakistan. Information Development, 26(2) 141-147. doi: 10.1177/0266666910366650

[52] Gurol M, Sevindik T (2007) Profile of Internet Cafe users in Turkey, Telematics and Informatics, Vol. 24, Issue 1: 59-68. doi: 10.1016/j.tele.2005.12.004

[53] Kihl M, Aurelius A, Lagerstedt C (2010) Analysis of World of Warcraft traffic patterns and user behavior, International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT): 218-223. doi: 10.1109/ICUMT.2010.5676634

[54] Chambers C, Feng W, Sahu S, Saha D (2005) Measurement-based Characterization of a Collection of On-line Games. In Proceedings of the 5th ACM SIGCOM conference on Internet Measurement (IMC'05). USENIX Association, Berkeley.

Available online: http://dl.acm.org/citation.cfm?id=1251087, [Accessed Apr 2014]

[55] Radhakrishnan S, Cheng Y, Chu J, Jain A, Raghavan B (2011) TCP fast open, In Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies (CoNEXT '11). ACM, New York, NY, USA, , Article 21, 12 pages. doi: 10.1145/2079296.2079317

[56] Belshe M, Peon R, Thomson M (2015) RFC 7540: Hypertext Transfer Protocol Version 2 (HTTP/2).
Available online: https://tools.ietf.org/html/rfc7540, [Accessed May 2015]

[57] Claypool M, Claypool KL (2006) Latency and player actions in online games. Commun. ACM 49, 11 40-45. doi: 10.1145/1167838.1167860

[58] AT&T Global Network Latency Averages, http://ipnetwork.bgtmo.ip.att.net/pws/global_network_avgs.html

[59] Issariyakul T, Hossain E (2011) Introduction to Network Simulator NS2. Springer.

[60] Chen KT, Huang CY, Huang P, Lei CL (2006) An empirical evaluation of TCP performance in online games. Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology (p. 5). ACM. doi: 10.1145/1178823.1178830

[61] Cooperative Association for Internet Data Analysis, NASA Ames Internet Exchange Packet Length Distributions.

BIOGRAPHIES

Jose Saldana received his B.S. and M.S. in Telecommunications Engineering from University of Zaragoza, in 1998 and 2008, respectively. He received his PhD in Information Technologies in 2011. He is currently a research fellow in the Aragon Institute of Engineering Research (I3A). His research interests focus on Quality of Service in Real-time Multimedia Services, as VoIP and networked online games. He has authored about 50 research articles in scientific journals and conferences, and he also participates in standardization activities within the IETF. He is also a member of the Technical Program Committee of some conferences, as IEEE CCNC and Globecom.