# Comparative of Multiplexing Policies for Online Gaming in Terms of QoS Parameters

Jose Saldana, *Student Member, IEEE,* Julián Fernández-Navajas, José Ruiz-Mas, José I. Aznar, *Student Member, IEEE*, Luis Casadesus, and Eduardo Viruete

***Abstract*—This letter compares different policies for multiplexing the traffic of online games. In order to achieve bandwidth savings and to alleviate the high packet rate, headers are compressed and a number of native packets are included into a bigger one, using PPPMux and an L2TP tunnel. Small and controlled delays and jitter are added due to retention at the queue of the multiplexer. The policies are compared using real traffic traces of a popular game, and the results show that the savings are significant, while the impairments are not severe.**

***Index Terms*—First person shooter, multiplexing, network traffic, online gaming, QoS.**

## I. Introduction

ONLINE games via Internet have become a very popular service in the last years. Amongst them, First Person Shooters (FPS) are the ones with the tightest real-time requirements. Players have proven to be very difficult to satisfy [1], so game providers not only have to deploy smart titles, but they must also offer a good network service. Statistical models have been developed for the traffic of many titles [2]. All of them have similar traffic patterns, generating high rates of tiny UDP packets (some tens of bytes). The models are different for client-to-server and server-to-client traffics, as servers generate bigger packets.

In this context, the impact of FPS's traffic on current network infrastructures has also been studied [3]. A conclusion is that the number of packets per second (pps) the router has to manage is a bottleneck which has to be taken into account in addition to the bandwidth limit. The reason is that many existing networks were developed to manage big TCP packets.

Thus, the traffic of FPSs is not optimal for the current network infrastructures due to two reasons: first, an efficiency problem, since small packets entail a big overhead. This problem will become even worse with the adoption of IPv6. Secondly, the amount of pps is directly impaired due to the high packet rate generated by these games.

There exist scenarios where many real-time flows share the same link, so the traffic can be optimized in order to adapt it to the infrastructure. Multiplexing is a well-known solution that has been largely used for real-time services like VoIP [4]. It enables us to group small packets into bigger ones, reducing the amount of pps, and having another interesting consequence: the overhead caused by IP/UDP headers can be significantly reduced, thus obtaining bandwidth savings.

A number of game flows can also share the same link in some scenarios. One of them is an Internet café, where a number (maybe some tens) of users can play together on the same server. Multiplexing can save bandwidth of the Internet access of the café. Other scenario is the one proposed in [5], where *booster boxes* are used to improve the network support for online games. They can be considered as *middleboxes* defined by IETF RFC 3303 [6]. They could be attached to ISP's access routers, aggregating the traffic of many users of a geographical zone and forwarding it to edge routers, which would have to manage less pps. Finally, a third scenario can be a wireless access network (e.g. WiMAX), where the efficiency can be improved by multiplexing game flows.

The quality experienced by players depends on network parameters, mainly delay, packet loss and jitter [7] . The effects of multiplexing have to be studied, since a new delay will be added (the retention time necessary to receive and multiplex a number of packets) and the jitter may be increased. Packet loss will only be indirectly affected because of the packet size increase.

## II. Proposed Methods and Policies

In RFC 4170 the IETF approved Tunneled Compressed RTP [4] in order to multiplex RTP flows. First, ECRTP compressing is applied, and many packets are included into the same one using PPPMux. Finally, a L2TP tunnel is used in order to send the whole multiplexed packet. We have used a similar scheme but, since the traffic is not RTP, we can only compress IP/UDP headers using IPHC or ROHCv2. This method is named TCM (Tunnel-Compress-Multiplex). Fig. 1 presents the structure of a TCM packet. It can be divided into the next parts: Common Header ($CH$=25 bytes): it corresponds to the IP, L2TP and PPP headers; PPPMux Header ($MH$=2 bytes): it is included at the beginning of each compressed packet; Reduced Header ($RH$=4.25 bytes average): it corresponds to the compressed IP/UDP header of each native packet; Payload ($P$): it is the UDP payload of the original packets generated by the application. $NH$ represents the size of a Normal IP/UDP Header (28 bytes).

There are many policies that can be used to decide the number of native packets to be included in each multiplexed one. For instance, a fixed number of packets or a size limit could be used, but they would not take into account the added delays, which have a strong influence on QoS. So in this work we will consider two policies capable of controlling

Manuscript received March 17, 2011. The associate editor coordinating the review of this letter and approving it for publication was Y.-D. Lin.

The authors are with the Communication Technologies Group (GTC) and Aragon Inst. of Engineering Research (I3A), Dpt. IEC. Ada Byron Building. CPS University of Zaragoza, Spain (e-mail: {jsaldana, navajas, jruiz, jiaznar, luis.casadesus, eviruete}@unizar.es).

This work has been partially financed by the CPUFLIPI Project (MICINN TIN2010-17298), MBACToIP Project, of Aragon I+D Agency, and Project Catedra Telefonica, Univ. Zaragoza.

Digital Object Identifier 10.1109/LCOMM.2011.080811.111160

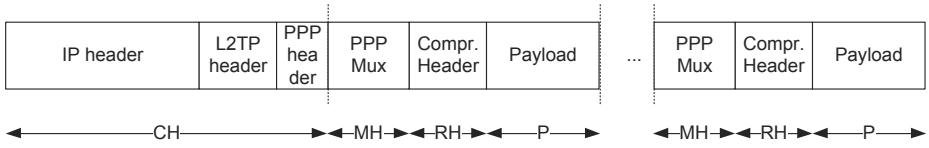



Fig. 1. Structure of a TCM packet.

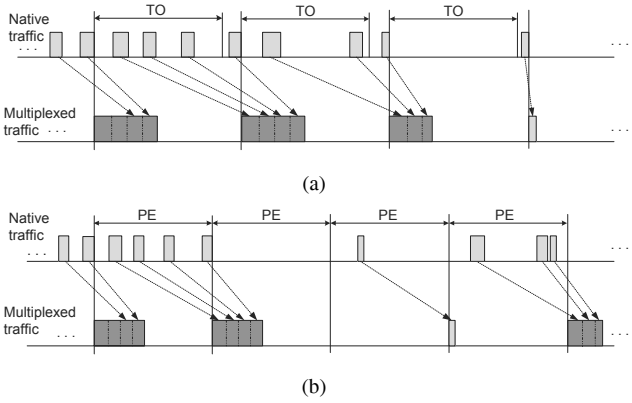



Fig. 2. Schemes of a) *timeout* and b) *period* policies.

the delays, comparing their performance and behaviour: first, *timeout* policy, in Fig. 2(a), sends a multiplexed packet if a native one arrives and the time since the last multiplexed packet departure is above a timeout *TO*. If there is only one packet, it will be sent in its native form, as the use of a tunnel would make it bigger. The other policy is named *period*, shown in Fig. 2(b), where a multiplexed packet is sent at the end of each interval or period *PE*. There are two exceptions: if there is no packet to multiplex, nothing will be sent; and if there is only one packet, it will be sent in its native form.

Next we will obtain the packet rate for each policy, and the bandwidth relationship (*BWR*), i.e. the quotient between the sizes of a multiplexed packet and the native ones it includes. Let $k$ be the number native of packets included into a multiplexed one. $N$ represents the number of players, and $\lambda$ is the amount of pps generated by each player. The expected value of the size of the native packets is:

$$E[k](NH + E[P]) \tag{1}$$

And the expected size of the multiplexed packet will be:

$$Pr(k = 1)(NH + E[P])+$$

$$+Pr(k > 1)(CH+E[k|k > 1](MH+E[RH]+E[P])) \tag{2}$$

So the quotient of (2) and (1) results:

$$BWR = \frac{Pr(k = 1)}{E[k]} + Pr(k > 1)\frac{CH}{E[k](NH + E[P])}+$$

$$+Pr(k > 1)\frac{E[k]|k > 1]}{E[k]}\frac{MH + E[RH] + E[P]}{NH + E[P]} \tag{3}$$

The first term expresses the situation in which a single packet is sent. The second one refers to the sharing of the common header, and it will be reduced as $E[k]$ grows. The third one implies the existence of an asymptote: as the number of packets grows, $Pr(k > 1)$ tends to be 1, and $E[k|k > 1]$ tends to be $E[k]$, so $(MH + E[RH] + E[P])/(NH + E[P])$ is the *BWR* limit.

The values of $Pr(k = 0)$, $Pr(k = 1)$ and $Pr(k > 1)$ depend on the multiplexing policies and also on the statistical distribution of the traffic of the game. The value of $E[k|k > 1]$, taking into account that $E[k|k = 0] = 0$ and $E[k|k = 1] = 1$, is:

$$E[k|k > 1] = \frac{E[k] - Pr(k = 1)}{Pr(k > 1)} \tag{4}$$

So we need to calculate the values of $E[k]$, $Pr(k = 1)$ and $Pr(k > 1)$ for each policy. We will also obtain the packet rate. The most common used models for inter packet times are exponential, deterministic, normal, lognormal and extreme [8]. We will present the calculations for the deterministic distribution, as it is one of the most used. Some games [2], [9] use two possible values for inter-packet times $t_1$ and $t_2$, each one with a probability $p_1$ and $p_2$. In this case, $\lambda = 1/(p_1t_1 + p_2t_2)$. A distribution with a single constant delay is a particular case of this one with $p_1 = 1$ and $p_2 = 0$. We will consider consecutive inter-packet times as independent.

In case of using *timeout* policy, the number of packets arrived in the interval *TO* will be $N\lambda TO$, and if we add the packet that triggers the departure, we obtain $E[k] = N\lambda TO + 1$. The multiplexed packet rate is the inverse of inter-packet time, i.e. $1/(TO + (1/N\lambda))$, which tends to be $1/TO$ as $N\lambda$ grows.

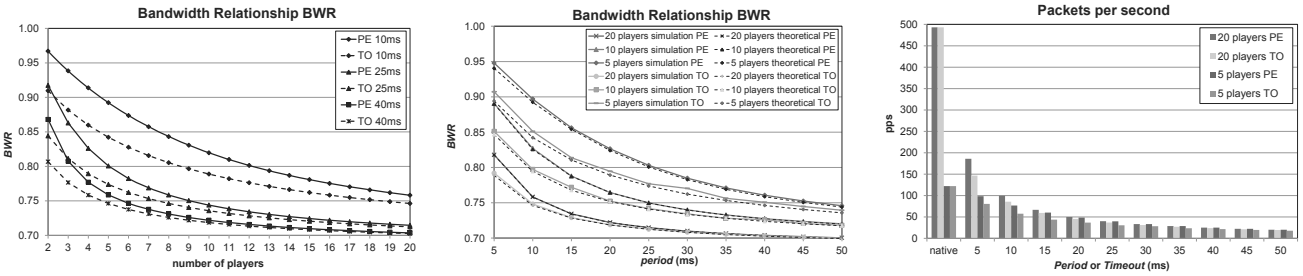
In this case, as $Pr(k = 0)$ is null, we have $Pr(k > 1) = 1-Pr(k = 1)$. If we define $l$ as the number of packets of a user arrived during *TO*, then $Pr(k = 1) = [Pr(l = 0)]^N$, so two cases can be distinguished: if $TO < t_1$ then $Pr(l = 2) = 0$, so we obtain:

$$Pr(l = 1) = E[l] = \lambda TO$$
$$Pr(l = 0) = 1 - Pr(l = 1) = 1 - \lambda TO \tag{5}$$

And if $t_1 \leq TO < t_2$, then the probability of having no packets from a user during *TO*, will correspond to the probability of the timeout interval beginning during the first $t_2 - TO$ seconds of an inter-packet time of duration $t_2$:

$$Pr(l = 0) = p_2\lambda(t_2 - TO) \tag{6}$$

If *period* policy is used, then $E[k] = N\lambda PE$. The multiplexed packet rate will be $Pr(k > 0)/PE$, which tends to be $1/PE$ as $N\lambda$ grows. Now, $Pr(k = 0)$ is not null, so we have to calculate it, and also $Pr(k = 1)$, in order to obtain $Pr(k > 1)$. Assuming that $PE < 2t_1$, and $t_2 < 2t_1$, then only $l = 2$ packets from a player can arrive during a *period*. We




(a) (b) (c)

Fig. 3. a) Theoretical *BWR* for both policies. b) Simulation vs. theoretical *BWR*. c) Simulation pps for both policies.

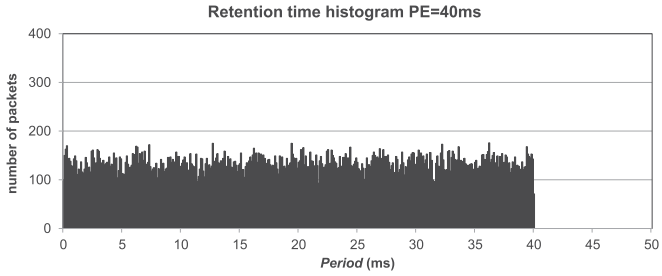



Fig. 4. Retention time histogram using *period* policy for 20 users. *PE*=40ms.

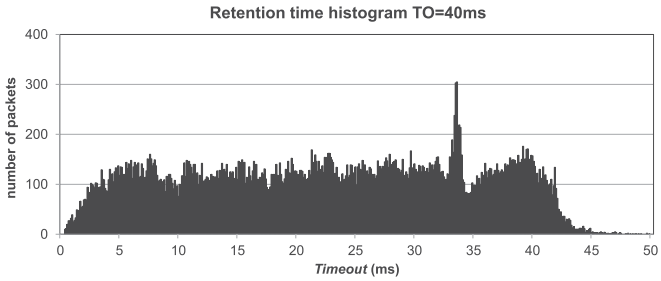



Fig. 5. Retention time histogram using *timeout* policy for 20 users. *TO*=40ms. A peak of 2641 packets for delay=0 has been cut away for clarity.

can distinguish two cases: if $PE < t_1$ then $Pr(l = 2) = 0$, so $Pr(l = 1)$ and $Pr(l = 0)$ are obtained like in (5). When $t_1 \leq PE < t_2$ then $Pr(l = 0)$ is calculated as in (6):

$$Pr(l = 0) = p_2\lambda(t_2 - PE) \tag{7}$$

And knowing that $Pr(l = 0) + Pr(l = 1) + Pr(l = 2) = 1$ and the value of $E[l] = Pr(l = 1) + 2Pr(l = 2) = \lambda PE$, we can find:

$$Pr(l = 1) = \lambda[PE - 2p_1(PE - t_1)] \tag{8}$$

$$Pr(l = 2) = p_1\lambda(PE - t_1) \tag{9}$$

We will have zero packets if none of the users has sent anyone, and one packet if a single user has sent one:

$$Pr(k = 0) = [Pr(l = 0)]^N \tag{10}$$

$$Pr(k = 1) = \binom{N}{1} Pr(l = 1)[Pr(l = 0)]^{N-1} \tag{11}$$

## III. Results and Conclusions

In order to represent the results for *timeout* and *period* policies, we have chosen a popular FPS game: Counter Strike 1.6. We have used client-to-server traffic because it is the one which obtains the greatest savings. Its traffic has a very characteristic behaviour [3], [9]. In OpenGL mode it presents two deterministic inter-packet times of 33 and 50 ms, with $p_1 = p_2 = 0.5$. This behaviour fits the theoretical assumptions of the previous section. In this case $\lambda = 24pps$ and $E[P] = 41$ bytes. Fig. 3(a) shows the theoretical *BWR*. The asymptote is *BWR*=0.68, so the saving is 32%. It can be seen that, as the *period* or the *timeout* grow, both policies tend to be similar. But for small values, *timeout* policy has better results, as it includes one more native packet into the multiplexed one.

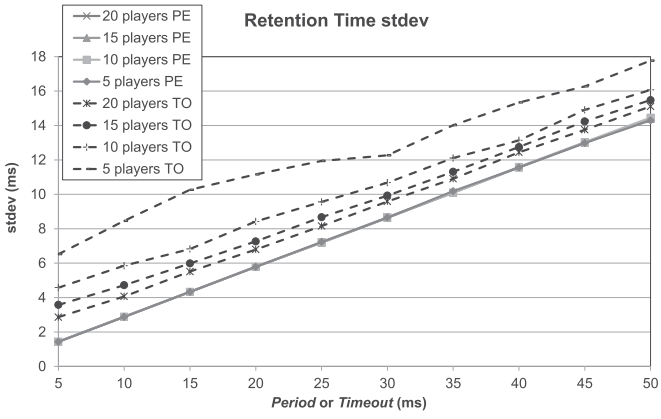



Fig. 6. Standard deviation of retention time using *period* and *timeout* policies.

Simulations have also been carried out, using real traces of 110 sec. of the game under study, obtained from [10]. Fig. 3(b) compares simulation and theoretical results, which fit well, except for a slight difference for small periods and number of players. The main cause is that the inter-packet times of the real traces present small variations around 33 and 50 ms.

Fig. 3(c) presents the reduction in terms of pps obtained in the simulations, which tend to be the inverse of *PE* or *TO*. For 20 players, the rate can be reduced from 493 to 20 pps.

Fig. 4 and 5 show the retention time histogram for a *period* or *timeout* of 40 ms. The average retention time is about half the *period* or *timeout*. This added time is acceptable, as players can tolerate times up to 200-225 ms [11]. While the *period* policy has a uniform distribution and maintains the added delay under an upper bound, *timeout* does not, and presents a tail above 40 ms. It shows a peak at 0 ms caused by the arrivals that trigger the multiplexed packets, and other one

corresponding to inter-packet time of 33 ms. Fig. 6 shows the standard deviation of the retention time, which is considered an impairment parameter in some studies [7]. It is smaller for the *period* policy, and the graphs are the same despite the number of players. This represents an advantage with respect to *timeout*. The small decrease around 30 ms in the *5 players TO* graph is caused by the inter-packet time of 33 ms.

As a summary, we can say that *timeout* policy has a better behaviour in terms of bandwidth, but it lacks an upper bound for the added delay, which *period* policy does have. The jitter impairment is also bigger for *timeout* policy.

## REFERENCES

[1] C. Chambers, W. Feng, S. Sahu, D. Saha, "Measurement-based characterization of a collection of on-line games," in *Proc. 5th ACM SIGCOM conference on Internet Measurement,* 2005.

[2] S. Ratti, B. Hariri, S. Shirmohammadi, "A Survey of first-person shooter gaming traffic on the Internet," *IEEE Internet Computing*, pp. 60–69, Sep./Oct. 2010.

[3] W. Feng, F. Chang, W. Feng, and J. Walpole, "A traffic characterization of popular on-line games," *IEEE/ACM Trans. Netw.*, pp. 488–500, 2005.

[4] B. Thompson, T. Koren, and D. Wing, "Tunneling multiplexed compressed RTP (TCRTP)," IETF RFC 4170, Nov. 2005.

[5] D. Bauer, S. Rooney, and P. Scotton, "Network infrastructure for massively distributed games," in *Proc. 1st Workshop on Network and System Support for Games*, pp. 36–43, 2002.

[6] P. Srisuresh *et al.*, "Middlebox communication architecture and framework," RFC 3303, Aug. 2002.

[7] A. F. Wattimena, R. E. Kooij, J. M. van Vugt, and O. K. Ahmed, "Predicting the perceived quality of a first person shooter: the Quake IV G-model," in *Proc. 5th ACM SIGCOMM Workshop Network and System Support for Games*, 2006.

[8] S. Zander and G. Armitage, "A traffic model for the Xbox game Halo 2," in *Proc. Int. Workshop on Network and OS Support for Digital Audio and Video*, pp 13–18, 2005.

[9] T. Lang, G. Armitage, P. Branch, and H. Choo, "A synthetic traffic model for half-life," in *Proc. Australian Telecom, Networks and Applications Conference*, 2003.

[10] L. Stewart and P. Branch, "HLCS, map: dedust, 5 players," 13 Jan. 2006, Centre for Advanced Internet Architectures SONG Database.

[11] S. Zander and G. Armitage, "Empirically measuring the QoS sensitivity of interactive online game players," in *Proc. Australian Telecommunications Networks & Applications Conference*, Dec. 2004.