What's in that Magic Box? The Home Entertainment Center's Special Protocol Potion, Revealed

C. E. Palazzi, S. Ferretti, M. Roccetti, *Member*, IEEE, G. Pau, *Member*, IEEE, and M. Gerla, *Fellow*, IEEE

Abstract — *Digital entertainment is going to be enjoyed at home through computer-centered services and broadband wireless connectivity. The hub of a home entertainment system will be based on a "magic box" able to handle all sort of multimedia data and deliver them in a fair and efficient way throughout the house. Unfortunately, with current systems, real-time applications (e.g., video streaming, online games) suffer from delays caused by the interference with elastic (e.g., downloading) ones. We provide insight on this problem and reveal the "special protocol potion" we blended to heal home entertainment systems from it. Indeed, our recipe is simply composed by i) an enhanced access point, ii) standard protocol features, and iii) a smart use of the available information on the on-going traffic. This way, we are able to achieve low per-packet delays and high throughputs, thus satisfying the requirements of both real-time and elastic applications. Most important to mention, our special potion is also free from side effects as it does not require modifications to Internet's protocols or architecture*[1].

Index Terms — Home Network, Computer-Centered Home Entertainment, Enhanced Access Point, Wireless Multimedia.

I. INTRODUCTION

The next frontier in home information systems is the convergence of personal computer, game console, digital set-top box, home gateway, and home automation system, into an integrated home server [3]. In this context, digital entertainment is going to play a major role having the way paved by the popular TiVo and Media Center [1], [2]. These are only two exemplars of what should be integrated in the home server becoming a module devoted to the digital entertainment diffusion in the networked home. We name this module *Home Entertainment Center* (HEC) and focus our attention on it. From a consumer's point of view, a HEC is a *magic box* where every game ever thought, every movie ever made, every song ever sung, plus news, sport events and shows, will be available for instant enjoyment with just one click on a button.

A HEC can hence be defined as a hub for all in-home entertainment experiences able to handle heterogeneous media and to exploit the home gateway to connect client devices located within the house and the outside world (i.e., the Internet). Many experts foresee an immediate future where each home gateway will provide wireless connectivity to the various devices through an access point (AP), while transport protocols utilized by applications will remain those that have been in use for the last 25 years: the Transmission Control Protocol (TCP) for elastic (e.g., downloading) applications and the User Datagram Protocol (UDP) for real-time ones (e.g., video/music streaming, online gaming) [6].

Unfortunately, in a home entertainment scenario a new problem emerges, requiring the employment of a specific solution. This problem is commonly synthesized with real consumers' sentences like: *"when I use file-sharing programs, my IP-TV constantly has a delay in airing the images"* or *"if I use the Internet while video-chatting, the video stream stalls a lot"*, as well as depicted like in Fig. 1.

Technical motivations behind this problem are deeply investigated in Section II, yet, in summary, the real fact is that browsing the Internet and downloading files are still considered the main applications run by consumers on the wireless channel. Therefore, to provide reliable delivery and high throughputs to this kind of applications: i) TCP and its congestion control functionality are employed at the transport layer, and ii) buffers and local retransmissions are extensively used at the MAC layer.

However, these solutions have been demonstrated to be harmful toward real-time applications as they increase the per-packet delivery latency [21]. Moreover, the adverse role played by the TCP's congestion control mechanism in this particular context represents an interesting problem as it embodies the reverse of the well known *UDP vs TCP* quarrel by which it is stated that UDP-based flows are aggressive toward TCP-based ones [5]. Indeed, in Section II we demonstrate how persistent TCP-based flows are responsible for performance deterioration of concurrent UDP-based applications. The completely new point of view on the UDP vs TCP quarrel makes even more evident how the HEC is in need of a special protocol *potion* to become the desired magic box for consumers' entertainment.

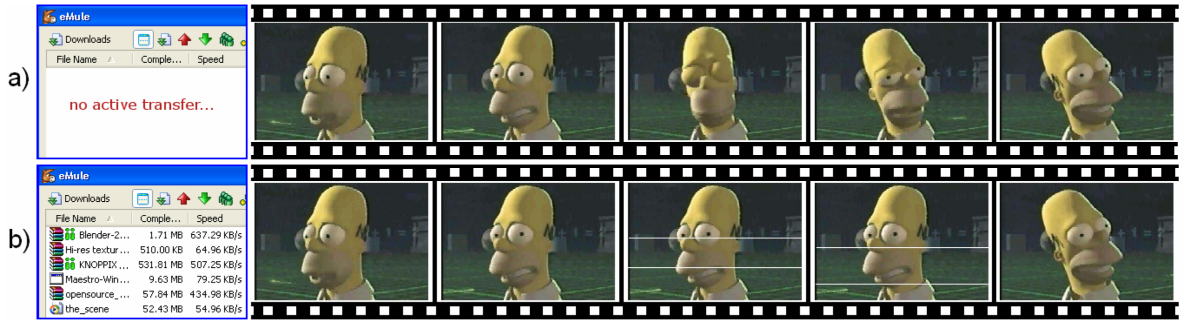
Aiming at providing high downloading throughputs and low per-packet delays, we propose a solution that makes use of a *smart AP*, standard protocol features, and available information on the on-going traffic.

[1] Partial financial support for this work is provided by: the Italian MIUR (under the ICTP/E-Grid, Interlink, MOMA, DAMASCO initiatives); the National Science Foundation (through grants CNS-0435515/ANI-0221528); and STMicroelectronics (under the UC-CoRe Grant MICRO 05-06).

C. E. Palazzi, G. Pau, and M. Gerla are with the Computer Science Department, University of California, Los Angeles, CA 90095, USA (e-mail: {cpalazzi | gpau | gerla}@cs.ucla.edu).

S. Ferretti, and M. Roccetti are with the Dipartimento di Scienze dell'Informazione, Università di Bologna, 40126, Bologna, Italy (e-mail: {sferrett | roccetti}@cs.unibo.it).

Contributed Paper
Manuscript received August 18, 2006



File Name	Comple...	Speed
Blender-2...	1.71 MB	637.29 KB/s
Hi-res textur...	510.00 KB	64.96 KB/s
KNOPPIX ...	531.81 MB	507.25 KB/s
Maestro-Win...	9.63 MB	79.25 KB/s
opensource_...	57.84 MB	434.98 KB/s
the_scene	52.43 MB	54.96 KB/s

Fig. 1. Video stream progression without (a) and with (b) simultaneous downloads.

Indeed, the AP is in a strategic position to gather information about the channel condition and the on-going traffic. Our idea is that this information can be used by the AP to regulate heterogeneous transmission flows and make them coexist efficiently. More in detail, the idea is as follows:

i) the AP snoops all the transiting packets and computes the maximum data rate at which downloading applications can transfer their files without exceeding the factually available bandwidth;
ii) a cross-layer approach may be implemented that allows the AP to modify on-the-fly the advertised window of each transiting TCP packet so as to make it equal to the maximum data rate mentioned in i), and accordingly limit the transfer rate of TCP flows.

This way, downloading applications produce a smooth traffic that efficiently utilizes the available channel without incurring in congestion losses that would degrade their performances and, at the same time, does not create queues at the AP that would increase per-packet delays of real-time applications.

Since we propose neither a new protocol nor a new architecture, our special potion is also highly tolerable from a deployability standpoint. Indeed, we simply design an enhanced AP with a re-engineered packet forwarding functionality. Customers have hence just to get our AP (or the HEC with our AP already included) from their preferred electronics seller and plug-it-in at home. Results gathered on the field showed that performances of entertainment applications in wireless homes will be improved in this simple way, with absolutely no need for updates of Internet's protocols. Comparative results contrasted against alternative approaches, such as TCP Vegas, have been obtained, which showed the superiority of our solution.

The rest of the paper is organized as follows. Section II focuses on the real problem at the basis of this work. In Section III, we discuss our solution to the problem and review the best alternatives available in scientific literature. A simulation assessment is presented in Section IV, while obtained results are shown in Section V. Finally, Section VI concludes this paper.

II. PROBLEM STATEMENT

We provide here further insight on the interference caused by downloading applications to real-time ones over a wireless channel.

Applications can be grouped into two main classes depending on which protocol they use at the transport layer: TCP or UDP. TCP is a protocol that guarantees the reliable and ordered delivery of every packet sent; to this aim it establishes a session and performs retransmissions of lost packets. Since these features, TCP is utilized by elastic applications that involve the download/transmission of files/commands (i.e., FTP, HTTP, SMTP, Telnet). Where not differently stated, with the term TCP we refer to the two most common versions, i.e., TCP New Reno and TCP SACK.

A very important component of TCP is represented by its congestion control functionality. Through it, every TCP flow probes the link with higher and higher data rates eventually filling up the channel. At that point, packets will be queued at the buffer associated with the bottleneck of the link until it overflows causing packet losses. TCP retransmits the lost packets, and halves its sending rate to diminish the congestion level. Finally, the regular increase of the sending rate is reestablished and so forth.

UDP is simpler: packets are immediately sent toward the receiver with a data rate decided by the sender. UDP does not guarantee reliable and ordered delivery of packets but, at the same time, its small overhead and lack of retransmissions make it less prone to generate delays in the packets delivery. For this reason, UDP is usually employed by applications characterized by stringent real-time constraints and that can tolerate sporadic packet losses (i.e., audio/video streaming, online games). The lack of congestion control functionalities of UDP had lead the scientific community to wisely consider UDP as unfair toward TCP. Indeed, citing from [5]: *"Although commonly done today, running multimedia applications over UDP is controversial to say the least. [...] the lack of congestion control in UDP can result in high loss rates between a UDP sender and receiver, and the crowding out of TCP sessions - a potentially serious problem."*

Even if this is true when the available bandwidth is very scarce, the broadband connectivity offered today may overturn this situation.

Larger and larger bandwidths are offered even to home consumers so that the traffic generated by UDP-based applications can be accommodated. Yet, a problem emerges when real-time applications (UDP-based) coexist with downloading ones (TCP-based) on a wireless channel, causing the former to experience a scattered flow progression.

Major causes for this problem can be found in the TCP's congestion control functionality. In particular, TCP continuously probes for higher transfer rates, also queuing packets on the buffer associated with the bottleneck of the connection. If one considers that the same wireless connection might be shared by several devices and applications thus increasing the congestion level and queue lengths, it is even more evident how packets can be delayed in queue, jeopardizing requirements of real-time applications.

This negative situation is further worsened by the following three factors due to the wireless nature of the link. First, the wireless medium allows the transmission of only one packet at a time and is not full-duplex as wired links. Packets have hence to wait their turns to be transmitted. Second, as interference, errors, fading, and mobility may cause packet loss, the IEEE 802.11 MAC layer reacts through local retransmissions (4 at most, [22]) which, in turn, cause subsequent packets to wait in queue until the preceding ones or their retransmissions eventually reach the receiver. Last but not least, the back-off mechanism of the IEEE 802.11 introduces an increasing amount of time before attempting again a transmission [22].

As an example of problems caused by this mixture of causes, we show in Fig. 2 the inter-arrival time experienced by online game packets in a realistic wireless in-home environment, with an AP configured in an off-the-shelf fashion, and a constant inter-departure time of 50 ms. Details for this scenario will be exhaustively discussed in Section IV. At this point, it is sufficient to know that the considered online game application (UDP-based) is started at 45 s, when a video-stream application (UDP-based) was already active, and both lasts for the whole experiment. At 90 s, a video-chat conversation (UDP-based) is added but, still, the traffic generated by the combination of these three applications is far from consuming all the available bandwidth. Instead, at 135 s an FTP (TCP-based) starts to download a file and quickly saturates the channel.

As it is evident in Fig. 2, the inter-arrival time remains regular until the FTP/TCP flow takes action. At that point, a multitude of packets start to experience high delays that cause a scattered progression of the online game flow.

What technically happens is even more evident in Fig. 3 that reports the *congestion/sending window*[2] (“cwnd”) and *slow start threshold* (“ssth”) of the TCP flow, as well as the pipe size, i.e., the Bandwidth-RTT product (“RTTxBW”) of the channel. By comparing Fig. 2 and Fig. 3 we notice that the irregularity in the interarrival-time of online game packets is directly proportional to the size of the congestion/sending window (i.e., the transfer rate). Every time the congestion/sending window exceeds the pipe size, packets are queued at the buffer of the bottleneck link. Delay increments of online game packets can hit also tens of milliseconds thus representing a huge waste of time when trying to deliver real-time information for entertainment services. For instance, transmission delays of interactive online games should be inferior to 100 ms, with a maximum endurable value of 150 ms [7]. (Just to mention, in this configuration the average throughput was 14.51 Mb/s).

[2] We use the term “congestion/sending window” to indicate when the sending window exactly corresponds to the congestion window.

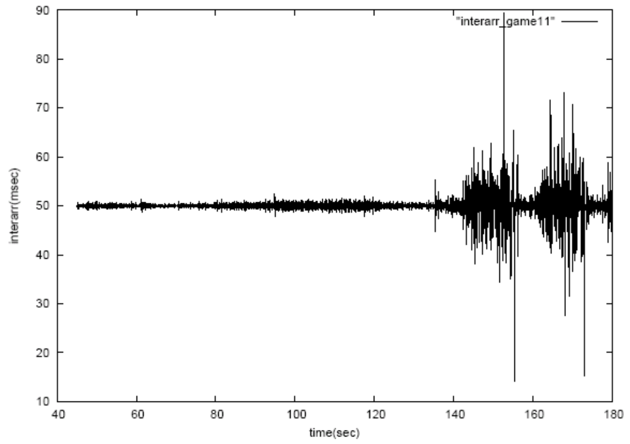


Fig. 2. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. Regular AP and IEEE 802.11g are employed; from 135 s, a FTP/TCP New Reno flow is competing for the channel.

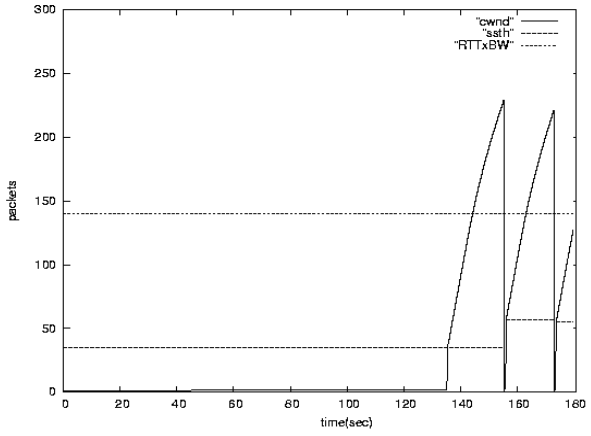


Fig. 3. TCP New Reno’s congestion/sending window with a standard AP.

III. Avoiding Queue Utilization

In this Section we provide two recipes for special potions we have blended to solve the aforementioned problem. Our solutions are finally contrasted with the best known alternative for limiting queuing delays.

We start from a simple, even if only moderately effective, solution that focuses on the MAC layer parameters [21]. Then, we propose a second solution which is based on the coordinated use of an enhanced AP, information available at the AP, and existing features of the TCP. We show how we can mix these ingredients in a simple way and obtain an efficient coexistence among downloading and real-time flows. This is our favorite solution among those we have explored and we name it *Smart Access Point with Limited Advertised Window* (SAP-LAW).

Obviously, several other alternatives can be found in literature that can be used to tackle this problem (e.g., CLAMP, IEEE 802.11e,…). Among them, TCP Vegas has emerged as one of the more promising. Hence, at the end of this Section, we provide a synthesis of this approach and compare it with our solution both in terms of efficacy and factual deployability.

A. Mixing a potion at the MAC layer: MAC-Setting

Our first solution regards the utilization of more appropriate setting for parameters of the IEEE 802.11 MAC protocol. Historically, parameters such as the maximum number of retransmissions and the buffer size were decided in a period when the TCP-based traffic was largely predominant in the Internet. Consequently, the main concerns for designers were reliability and high throughput.

Nowadays, the consumers' massive request for real-time applications partially contradicts the initial assumption that reliability were the most important issue over wireless links: ensuring low per-packet delays has to be considered equally crucial. Therefore, IEEE 802.11's parameters should be modified to make it more sensitive toward the requirements of real-time applications. In particular, the maximum number of local retransmissions could be diminished from 4 to 3 in order to find a solution to the tradeoff between reliability and low delays in packet delivery [21].

Similarly, large buffer sizes at the AP help TCP connections in maintaining a high sending rate for a longer period and diminish the impact of bursty traffic. On the other hand, larger buffers generate longer queuing times when fully utilized, thus jeopardizing the performance achieved by time-sensitive applications. By adjusting the buffer size to an appropriate value we can try to find a compromise between the needs of TCP-based applications and those of UDP-based ones. In particular, the buffer size should not exceeds 50 packets [21].

However, as shown in Section V, by only reducing the maximum number of retransmissions and the buffer size at the MAC layer, we can expect only a slight decrease of delays, which is paid with a loss increase and a consistent throughput reduction.

B. SAP-LAW: Blending Together a Smart Access Point with a Limited Advertisement Window

Here we are aiming at finding the best solution for the tradeoff relationship existing between FTP/TCP throughput and real-time application delays. The two types of traffic should be able to coexist without interfering each other and the employed solution should be easily and factually deployable. An appropriate technique to solve this tradeoff relationship should exploit existing features of legacy protocols in order to be easily deployable. Our most effective potion is inspired by this principle and can be explained by stepping through the following three main points.

1) *The Basic Idea*: The actual sending rate (i.e., the sending window) of a TCP flow is determined as the minimum between the congestion window (continuously recomputed by the sender) and the advertised window (provided by the receiver via returning ACK packets) [5]. Our idea is hence that of exploiting dynamic modifications of the advertised window to limit the growth of the TCP flow's sending rate. Indeed, an optimal tradeoff between throughput and low delays could be achieved by maintaining the sending rate of the TCP flows high enough to efficiently utilize the available bandwidth and, at the same time, limited in its growth so as to not utilize buffers. This way, per-packet delays are minimized by the absence of queues along the route from the sender to the receiver, while the throughput is kept elevated by the absence of packet losses that would halve the congestion window.

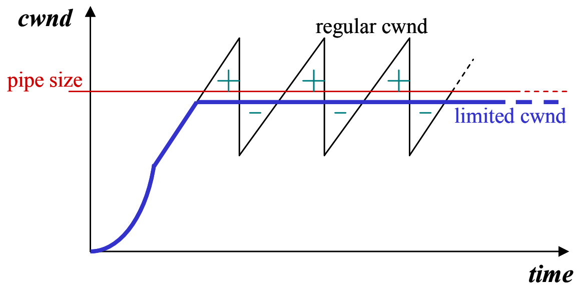


Fig. 4. Comparison between regular and limited sending window.

We provide in Fig. 4 a little evidence that the TCP throughput is not diminished by our proposed solution. Specifically, we show in Fig. 4 a typical saw tooth shaped congestion/sending window of a TCP flow and overlap it with one limited by the advertised window. As it is evident, the latter is more stable since it does not use the buffer at the bottleneck link and consequently experiences no losses. The minus signs in the chart represent situations in which the regular sending window provides TCP with a sending rate that is inferior to the one guaranteed by the limited congestion/sending window. The plus signs represent the inverse situation (generally accompanied by having packets queuing on the buffer preceding the bottleneck link). If the upper bound for the sending window is appropriately chosen, the balance between the plus and minus signs will guarantee to the flow with a limited sending window an equal, or even superior, final throughput with respect to an equivalent regular flow. At the same time, queuing delays will be avoided.

We need now to address two important issues: how to determine an appropriate upper bound and how to practically implement it. Here follow our solutions to these issues.

2) *Determining the Upper Bound*: The most appropriate formula to compute the upper bound can be derived from the two main goals we want to achieve: full utilization of the available bandwidth and no queuing delays.

Real-time traffic generally exploits UDP to transmit at a predetermined rate with no concern about other traffic on the channel. Therefore, to avoid queuing delays, the aggregate bandwidth utilized by TCP flows cannot exceed the total capacity of the bottleneck link at the AP diminished by the portion of the channel already occupied by the concurrent real-time traffic.

If we call *UDPtraffic* the amount of bandwidth occupied by the UDP-based traffic, *#TCPflows* the concurrent number of

TCP flows, and *C* the capacity of the bottleneck link, we can formulate the upper bound for the sending rate allowed to each TCP flows at time *t* as follows:

$$maxTCPrate\,(t) = \frac{(C - UDPtraffic\,(t))}{\#\,TCPflows\,(t)} \qquad (1)$$

3) *Practical Implementation*: To practically employ eq. (1) we have to i) identify the location for its implementation, and ii) propose a method to compute the value of the various variables.

Regarding i), the advertised window is generally imposed by the receiver; however, this could not represent the most suitable place to set it. As it is evident from eq. (1), determining the most appropriate value for the advertised window requires a comprehensive knowledge about all the flows that are transiting through the bottleneck. Since all flows have to pass through the AP, this represents the most appropriate node to host the intelligence of our solution. Indeed, the AP is integrated with the HEC and the mechanism can take advantage of this to retrieve all the necessary information.

Focusing on ii), in any commercial operating system it is possible to know which kind of connection is in use and its nominal speed just by looking at the status of the network interface. Through snooping the channel or exploiting information known at the HEC we can also infer both the number of active TCP connections and the aggregate amount of concurrent UDP traffic. The AP can hence easily compute the best *maxTCPrate(t)* utilizing eq. (1) and modify the advertised window included in the transiting ACKs accordingly.

C. The incapability to do magic: the TCP Vegas Solution

Our work would not be complete if we did not compare our special potion with the best existing alternatives in avoiding excessive buffer utilization such as TCP Vegas, IEEE 802.11e, and CLAMP [8], [22], [20]. We focus here only on TCP Vegas as this protocol emerges as the best-endowed to avoid queuing. Indeed, this transport protocol embodies one of the most cited alternatives to regular TCP in scientific papers. Its applicability to the considered problem lies in the fact that TCP Vegas tries to avoid congestion before it happens. In particular, it augments its congestion window until buffers along the path between sender and receiver have a low utilization, whereas it reduces its congestion window when queuing is sensed. Therefore, TCP Vegas perfectly fits real-time applications' need for low buffer utilization.

In Section V we will show TCP Vegas' outcomes in our considered scenario; instead, here we provide more insight into TCP Vegas' behavior for the sake of the reader's comprehension.

While regular TCP utilizes packet loss to determine network congestion, TCP Vegas is sensitive to end-to-end queuing delay. With TCP Vegas, the sender monitors every round trip time (RTT) the difference between its expected rate and the actually achieved one. The difference is compared with a couple of parameters, namely α and β, to determine whether the congestion window has to be incremented or decremented (by 1) during the next RTT [8], [9]. In essence, parameters α and β determine the amount of buffer that can be utilized by transiting flows and can be seen as knobs able to move the tradeoff between the per-packet delay and the total throughput toward one direction or the other. If the buffer at the bottleneck is large enough then TCP Vegas reaches an equilibrium. In this case, TCP Vegas flows should experience zero packet losses, a stable congestion/sending window, and a buffer utilization that is proportional to the number of TCP Vegas flows sharing the same bottleneck.

Nonetheless, TCP Vegas is not capable to do the magic in the context of the HEC without heavy side effects. Indeed, even if TCP Vegas has been proven to fairly share the channel with other TCP Vegas flows, it behaves too conservatively in presence of simultaneous regular TCP flows. TCP fully exploits the available buffer and TCP Vegas interprets the consequent RTT trend as an indicator of excessive congestion, thus progressively reducing its sending rate to very low values [10]. In essence, the dramatic efficiency decrease experienced when competing with regular TCP traffic impedes TCP Vegas' factual deployment.

IV. Simulation Assessment

To evaluate the efficacy of our potion and the existing best alternative (TCP Vegas) we have built an experimental scenario utilizing the NS-2 simulator [12]. In particular, we intend to analyze a general house environment with four wirelessly connected devices and the home server that incorporates also the HEC and the AP. The distance between each device and the AP is 10m and the MAC layer parameters have been set accordingly to the IEEE802.11g standard allowing us to reach a maximum bandwidth of circa 20 Mb/s. This represents a reasonable value over the declared 54 Mb/s even in the real world [13].

For the wireless medium we chose the *Shadowing Model* to realistically simulate signal fading. We followed the directions provided by the official NS-2 manual to represent a home environment partitioned into several rooms. Specifically, the *path loss exponent* and the *shadowing deviation* parameters were set to the worst possible case suggested for an indoor environment, i.e., 4 and 9 respectively.

As recently demonstrated by measurements on a real OC48 link, the available capacity in the Internet core is generally larger than the aggregate utilized by transiting flows [14]. Moreover, tools are available to consumers to verify that their connection is factually supporting the high speed advertised by the provider [15]-[17]. We can hence assume the bottleneck located at the edge of the path connecting a sender and a receiver: the 20 Mb/s effectively available over the wireless link. Over this topology, several kinds of application were simulated that used different transport protocols, starting times, and RTTs (as shown by Table I). Just to mention, the video-stream had almost no propagation latency since the video was supposed to be stored at the HEC.

In order to uplift the trustworthiness degree of the simulations, we have exploited real trace files for the video-stream and for the video-chat. Specifically, adopted trace files correspond respectively to high quality MPEG4 *Star Wars IV* for the movie, and two VBR H.263 Lecture Room-Cam for the video-chat, available in [18].

Parameters characterizing the game-generated traffic were chosen following directions provided by scientific literature related to this field. We assumed that the user in the house were engaged in one of the very popular first person shooter games, e.g. Quake Counter Strike, with other ~25 players, geographically away from each other and connected through the Internet. To model the packet size and inter-arrival time of the traffic generated by online games, we used some of the approximations suggested in [19], which are based on real game platform measurements. Game events were hence generated at the client side every 60 ms; whereas the server was transmitting back game state updates every 50 ms toward the client. Moreover, game packet sizes generated by client and server were set to 42 Bytes and 200 Bytes, respectively.

TABLE I
SIMULATED APPLICATION FLOWS

Flow Type	Transp. Prot.	Start	End	RTT
Video-Stream	UDP	0 s	180 s	~0 ms
Online Game	UDP	45 s	180 s	40 ms
Video-Chat	UDP	90 s	180 s	60 ms
FTP	TCP	135 s	180 s	80 ms

TABLE II
TUNABLE PARAMETERS IN THE SIMULATED CONFIGURATIONS

Parameter	Values	Comment
MAC data retransmissions	1, 2, 3, **4**	default value = 4
MAC queue size (pkts)	50, **100**	common default values

Simulation experiments have been replicated to examine the effects generated by different configurations of the parameters involved in the considered scenario (see Table II). Where not differently stated, simulations were run utilizing some realistic default values for the simulative parameters. These values are written in bold in Table II.

We focus our attention on the most significative results with respect to the problem experienced by online game packets. However, no significant information is retained since the per-packet delay and jitter for all the considered real-time applications showed a homogeneous trend. Aimed at finding the best solution to provide both high throughput and low per-packet delay we use the following metrics:

- the inter-arrival time and the jitter experienced by packets of one of the simulated real-time applications;
- the throughput achieved by the FTP/TCP application.

In the next Section we test the various special potions to find the most successful one and verify the presence of any side effect.

V. RESULTS

Exploiting the metrics highlighted in Section IV, we discuss in the following subsections the most relevant results from the extensive set of simulations we have run for each discussed solution. We start with a comparison of the various outcomes and continue with detailed results on each technique.

A. Preview of Results

We conducted a comparative measurement of the various solutions and statistical outcomes of the online game traffic, plus the average throughput achieved by the concurrent TCP connection, are reported in Fig. 5. As it is evident, the TCP Vegas solution and our SAP-LAW are the two approaches that would guarantee the best performances. Indeed, they both show the lowest values for average, variance, and maximum jitter experienced by the online game stream, as well as high TCP's throughput.

However, even if TCP Vegas' outcomes result slightly better than those achieved by SAP-LAW, the former cannot be actually deployed in the Internet since it is not able to efficiently coexist with the legacy TCP [10]. Conversely, SAP-LAW can be easily implemented as it only involves the presence of slightly "smarter" APs. Modifications required at the AP are very limited, thus minimally impacting on their cost. Moreover, our scheme utilizes only existing features of currently employed protocols and is hence perfectly compatible with the Internet. Considering this fundamental advantage and the remarkable results achieved, SAP-LAW represents the optimal candidate for enhancing HECs in wireless home scenarios.

To better understand results shown in Fig. 14, in the rest of this Section we provide further insight on the outcomes of the various solutions explored.

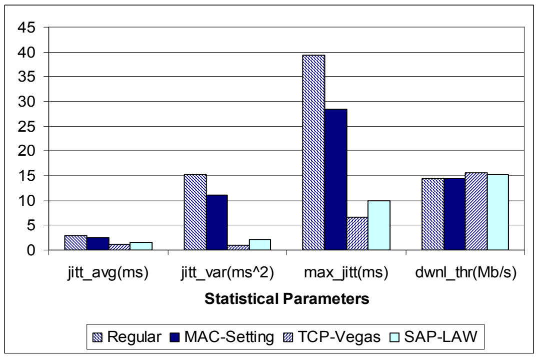


Fig. 5. Statistical outcomes for the compared schemes during the simulative interval 135–180 s.

B. Setting MAC Layer Parameters: Outcomes

Utilizing different buffer sizes and/or maximum number of retransmissions at the MAC layer amounts to the simplest solution among those we have explored. However, this solution leads to just a moderate improvement of performances achieved by the various real-time applications at the cost of a reasonable throughput decrease for downloading flows.

This is clearly visible in Fig. 6 that shows a (moderate) reduction of the height of the inter-arrival time peaks with respect to the regular case (compare values on the y-axis with those in Fig. 2). This result was achieved by changing the normal setting of the AP (buffer size = 100 packets, maximum number of retransmissions at the MAC layer = 4) with 50 packets of buffer size and 3 retransmissions, at most, at the MAC layer. The rationale behind this decision can be found also in Fig. 7 that shows downloading flow's throughputs achieved in our simulations with various combinations of the considered parameters. The throughput achieved with this solution has been reduced with respect to the regular case but it still represents a decent value if compared to other configurations (i.e., 1 or 2 maximum number of retransmissions) and corresponds to 14.37 Mb/s.

Finally, Fig. 8 shows how the congestion/sending window peaks of the concurrent TCP flow do not exceed the pipe size as much as they were doing in Fig. 3 since they have been reduced in height as an effect of the utilized configuration of the MAC layer parameters.

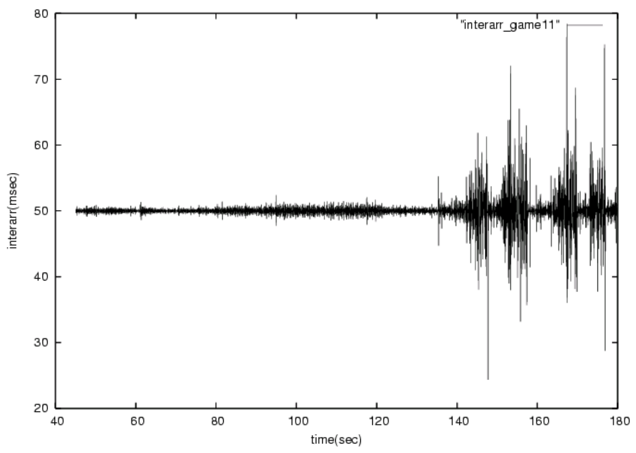


Fig. 6. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. Max 3 retransmissions at the MAC layer and buffer size = 50 packets; from 135 s, a FTP/TCP New Reno flow is competing for the channel.

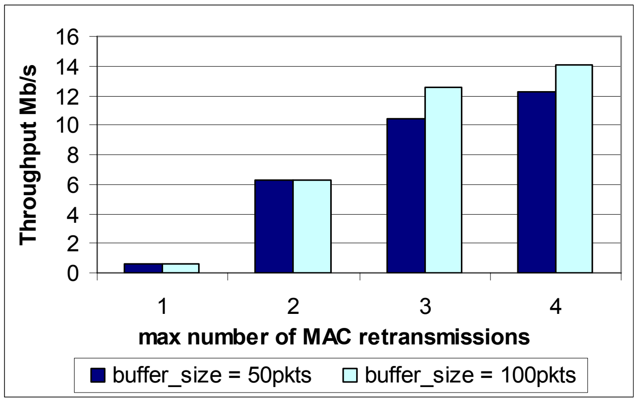


Fig. 7. FTP total throughput with different MAC buffer sizes.

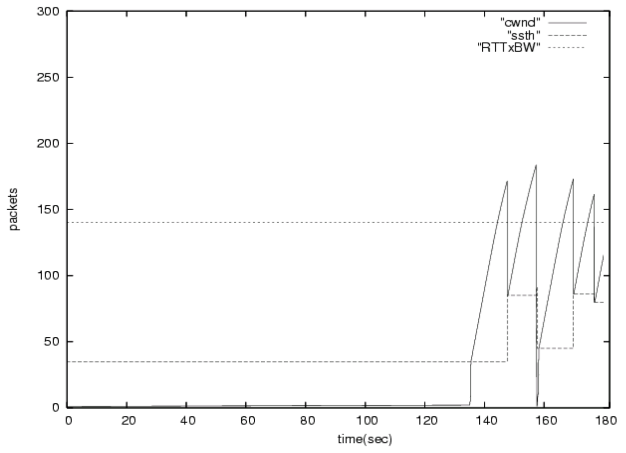


Fig. 8. TCP New Reno's congestion/sending window; max 3 retransmissions at the MAC layer and buffer size = 50 packets.

C. Utilizing SAP-LAW: Outcomes

As shown by Fig. 9, SAP-LAW is able to sensibly reduce the variation of the inter-arrival time suffered by online game packets in our considered scenario (compare values on the y-axis with those in Fig. 2). More in detail, Fig. 9 was obtained with parameter C in eq. (1) set to 18 Mb/s (the 90% of the maximum achievable bandwidth in our scenario).

Statistics (i.e. average, variance, and maximum value) about the jitter experienced by online game packets and the throughput trend of the concurrent TCP flow are shown in Fig. 10 as obtained by varying the value of parameter C. As expected, to higher values of C corresponds higher utilization of the buffer at the AP and hence higher average, standard deviation, and maximum value of the per-packet delays experienced by the online game flow. Moreover, an appropriate setting of C also helps in preserving, or even augmenting, the throughput. In particular, with C = 18 Mb/s, SAP-LAW allowed the TCP flow to achieve an average throughput of 15.23 Mb/s.

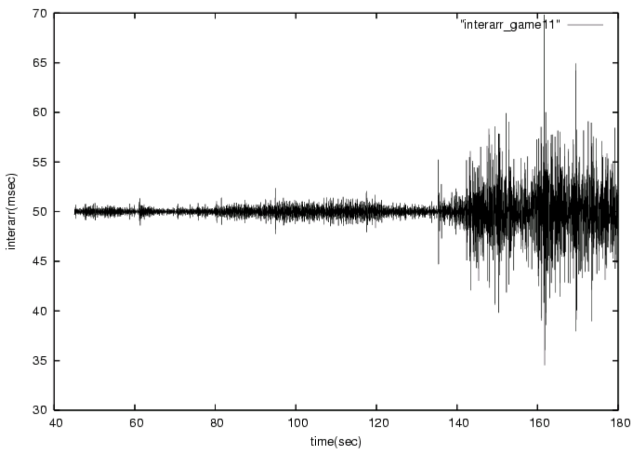


Fig. 9. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. SAP-LAW and regular IEEE 802.11g are employed; from 135 s, a FTP/TCP New Reno flow is competing for the channel.

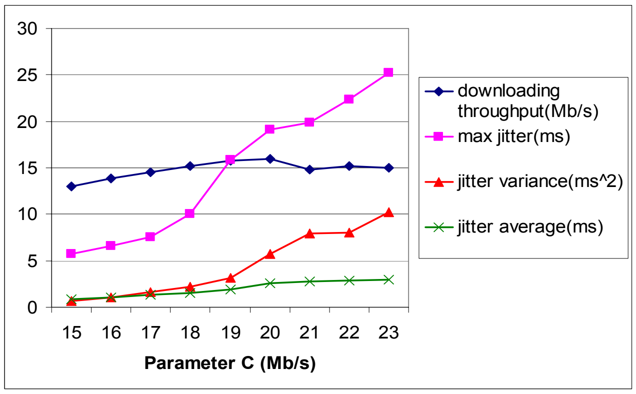


Fig. 10. Throughput achieved by the FTP/TCP New Reno flow and jitter statistics of the game flow when employing SAP-LAW and regular 802.11g setting.

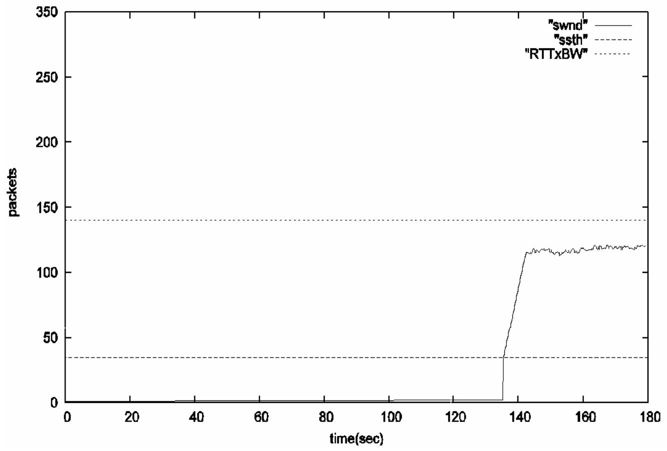


Fig. 11. TCP New Reno's sending window; SAP-LAW, C = 18 Mb/s, and regular 802.11g setting.

As for the precedent solution, even in this case we have to analyze the TCP's sending window as it represents the determining factor in generating packet delivery delays and TCP's throughput. Therefore, Fig. 11 shows the sending window ("swnd") generated by SAP-LAW as the minimum between the congestion window and the advertised window, with the latter dynamically computed by exploiting eq. (1). The aim is that of keeping the sending window high enough to achieve an high throughput but, at the same time, lower than the pipe size so as to not generate excessive queuing delays. As can be seen in Fig. 11, this result is perfectly achieved by SAP-LAW thus explaining the good performance obtained by our scheme in terms of low per-packet delay and high throughput.

D. TCP Vegas in Place of Regular TCP: Outcomes

Here we discuss the effect of utilizing TCP Vegas in place of regular TCP to download files in the wireless home scenario. In particular, we can appreciate in Fig. 12 how TCP Vegas' ability in avoiding queuing is reflected in a limited range extension of interarrival-times experienced by the concurrent online game flow (compare values on the y-axis with those in Fig. 2).

Such a result was achieved by using α = 3 and β = 7 and generated also a high average TCP throughput (15.57 Mb/s as can be seen in Fig. 13). The corresponding congestion/sending window is depicted in Fig. 14 and demonstrates how TCP Vegas' transfer rate is kept lower than the pipe size for the presence of other traffic, thus avoiding congestion and queues.

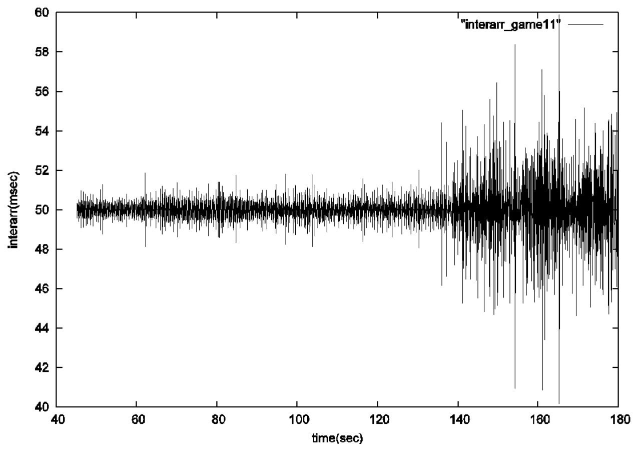


Fig. 12. Measured inter-arrival time of online game packets with 50 ms of inter-departing time. Regular AP and IEEE 802.11g are employed; from 135 s, a FTP/TCP Vegas flow with α = 3 and β = 7 is competing for the channel.

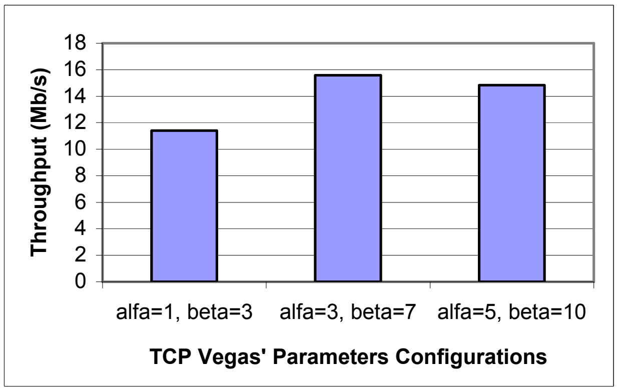


Fig. 13. FTP/TCP Vegas' total throughput with different setting of parameters α and β.

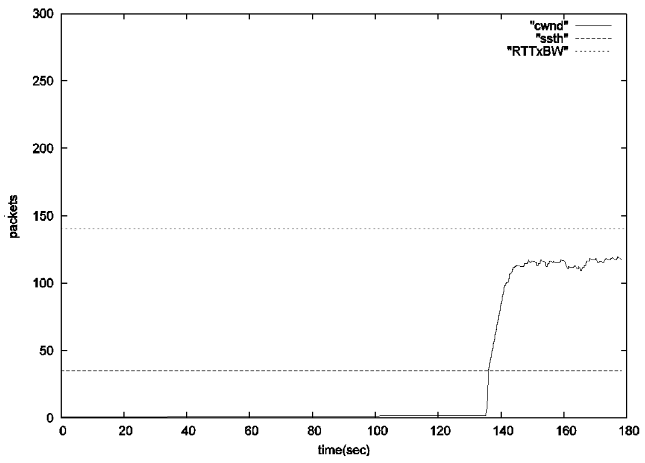


Fig. 14. TCP Vegas' congestion/sending window with α = 3 and β = 7; regular AP and 802.11g setting.

Unfortunately, TCP Vegas suffers from three main drawbacks that are well known in the scientific community and that pose drastic limits to its factual deployability. First, setting its parameters is not a trivial task and depends on many factors such as the buffer size at the bottleneck and the number of flows sharing that link. In particular, the last factor continuously changes while it is not possible to continuously adapt α and β. Second, TCP Vegas has been shown to be rather unstable [11]. Third, TCP Vegas behaves very poorly in terms of throughput when coexisting with the legacy TCP as shown, for instance, in [10].

VI. Conclusion

We took into account a scenario where in-home entertainment is delivered to wireless devices through a HEC.

Our analysis focused on the mutual influence among concurrent transmissions of two kind of different streams through a wireless HEC. The former being TCP-based elastic (e.g., downloading) applications and the latter being UDP-based real-time (e.g., video streaming and online gaming) applications.

We showed how even a single persistent TCP connection can conspicuously increase the queuing delay suffered by concurrent real-time applications. This constitutes the reverse of the well known argument by which UDP's lack of congestion control would harm TCP, whereas we have shown how the TCP's lack of buffering control is harmful as well toward UDP-based applications.

To solve this problem, we proposed SAP-LAW, a solution consisting in an enhanced AP that does not need to modify existing Internet's protocols. We compared SAP-LAW to two other possible solutions. The first one is an home made technique based on the idea of optimizing parameters at the MAC layer, while the second amounts to the use of TCP Vegas. Results showed that SAP-LAW and TCP Vegas achieved similar results and were both able to consistently ameliorate the global performance of computer-centered home entertainment services. However, while TCP Vegas cannot coexist with legacy TCP, SAP-LAW is fully compatible with the Internet and requires only the plugging-in of an enhanced AP with no protocol modifications at the Internet side.

Therefore, SAP-LAW emerges as the perfect *magic potion* to reduce consumers' frustrations with scattered progression of real-time flows when concurrent downloading applications are heavily competing for the channel usage.

References

[1] The TiVo Homepage. http://www.tivo.com/

[2] Windows XP Media Center Edition 2005 Home Page. http://www.microsoft.com/windowsxp/mediacenter/

[3] I. Han, H.-S. Park, Y.-K. Jeong, K.-R. Park, "An Integrated Home Server for Communication, Broadcast Reception, and Home Automation," IEEE Transactions on Consumer Electronics, vol. 52, no. 1, pp. 104-109, Feb 2006.

[4] IEEE Standard for Information Technology, "Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Amendment: Medium Access Control (MAC) Quality of Service Enhancements", P802.11e/D13.0, Jan 2005.

[5] J. F: Kurose, K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*, Addison Wesley Longman, Boston MA, USA, 2001.

[6] L. L. Peterson, B. S. Davie, *Computer Networks, a System Approach*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[7] L. Pantel, L. C. Wolf, "On the Impact of Delay on Real-Time Multiplayer Games," *in Proc. of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Miami, FL, USA, pp. 23-29 May 2002.

[8] S. Low, L. Peterson, L. Wang, "Understanding Vegas: a Duality Model," Journal of ACM, vol. 49, no. 2, pp. 207-235, 2002.

[9] L. Brakmo, S. O'Malley, L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," *in Proc. of the SIGCOMM '94 Symposium*, London, UK, pp. 24-35, Aug 1994.

[10] G. Marfia, C. E. Palazzi, G. Pau, M. Gerla, M. Y. Sanadidi, M. Roccetti, "TCP Libra: Exploring RTT-Fairness for TCP," UCLA CSD Technical Report #TR050037, 2005.

[11] C. Jin, D. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell, J. C. Doyle, W. C. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, and S. Singh, "Fast TCP: From Background Theory to Experiments," IEEE Network, vol. 19, no. 1, pp. 4-11, 2005.

[12] The Network Simulator, NS-2. http://www.isi.edu/nsnam/ns/

[13] A. L. Wijesinha, Y. Song, M. Krishnan, V. Mathur, J. Ahn, V. Shyamasundar, "Throughput Measurement for UDP Traffic in an IEEE 802.11g WLAN, " *in Proc. of 6th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)*, Towson, MD, USA, pp. 220-225, May 2005.

[14] H. Jiang, C. Dovrolis, "Why is the Internet traffic bursty in short (sub-RTT) time scales?" *in Proc. of ACM SIGMETRICS 2005*, Banff, AL, Canada, pp. 241 - 252, Jun 2005.

[15] Broadband Speed Tests. http://www.dslreports.com/stest

[16] Verizon Online DSL. http://www.verizon.com/dsl/

[17] AT&T Worldnet DSL Service. http://www.att.net/dsl/

[18] Movie Trace Files. http://www-tkn.ee.tu-berlin.de/research/trace/ltvt.html

[19] J. Farber, "Traffic Modelling for Fast Action Network Games," Multimedia Tools and Applications, vol. 23, no. 1, pp. 31-46, 2004.

[20] L. L. H. Andrew, S. V. Hanly, R. G. Mukhtar, "CLAMP: Active Queue Management at Wireless Access Points," in *Proc. of the 11th European Wireless Conference 2005*, Cyprus, Apr 2005.

[21] C. E. Palazzi, G. Pau, M. Roccetti, M. Gerla, "In-Home Online Entertainment: Analyzing the Impact of the Wireless MAC-Transport Protocols Interference," *in Proc. of IEEE International Conference on Wireless Networks, Communications and Mobile Computing (WIRELESSCOM 2005)*, Maui, HI, USA, pp. 516- 521, Jun 2005.

[22] IEEE, "Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," Specifications, ISO/IEC 8802-11:1999(E), 1999.

Claudio E. Palazzi received a PhD from the University of Bologna in 2006. His research interests include wireless multimedia entertainment. (Photo N/A).

Stefano Ferretti is currently an assistant professor at the University of Bologna. His research interests include distributed multimedia systems. (Photo N/A).

Marco Roccetti is a professor of Computer Science at the University of Bologna His work focuses on computer-centered entertainment system and wireless multimedia. (Photo N/A).

Giovanni Pau is currently a research scientist at UCLA. His area of expertise includes mobile computer network environment. (Photo N/A).

Mario Gerla is a professor at UCLA. His area of interest is focused on analysis, design, and control of computer communication networks. (Photo N/A).