WIRELESS HOME ENTERTAINMENT CENTER: PROTOCOL COMMUNICATIONS AND ARCHITECTURE

M. Roccetti[1], C. E. Palazzi[1, 2], S. Ferretti[1], and G. Pau[2]
[1]Università di Bologna, Bologna, Italy
[2]University of California - Los Angeles, Los Angeles, CA, USA

Keywords: *Home Network, Computer-Centered Home Entertainment, Enhanced Access Point, Wireless Multimedia, Media Center.*

Definition: *Wireless home entertainment center refers to a device able to handle heterogeneous media and to connect client devices located within the house and the outside world (i.e., the Internet).*

Technologies such as TiVo and Media Center have introduced the concept of Digital Video Recorder (DVR) in millions of homes. Consumers are now able to pause live-TV programs and watch them at their own convenience without the need of obsolete VCRs. These devices and their future evolution can be grouped into the class of Home Entertainment Centers (HECs). Specifically, a HEC can be defined as a hub for all in-home entertainment experiences able to handle heterogeneous media and to perform as a gateway between client devices located within the house and the outside world (i.e., the Internet).

Besides the DVR functionalities, a HEC can be considered as a *magic box* that provides several other services such as IPTV, Web radio, game console, picture viewer, electronic program guide, DVR, CD/DVD/video player, music jukebox, web browser, email handler, instant messenger. Contents related to these services can be locally available or distributed over the Internet to be dynamically retrieved based on the consumer's need with just one click on a button. Several streams are thus produced by these active services, all distributed by the HEC throughout the house.

At the same time, wireless home connectivity is becoming more and more popular, offering mobility, flexibility, and high transmission rates (i.e., 54 Mbps for the IEEE 802.11g and 100 Mbps for the IEEE 802.11n). We can hence assume that every HEC will be endowed with an Access Point (AP) in order to guarantee wireless connectivity to the various devices (e.g., screens, speakers, joypads).

Unfortunately, protocols run on APs have been designed following the belief that the main application run by users on the wireless channel was browsing the Internet and, in general, downloading files. Therefore, buffers and local retransmissions are extensively used, in the aim of providing reliability and high throughput to this kind of applications. On the other hand, these solutions have been demonstrated to be harmful towards real-time applications as they increase the per-packet delivery latency [1], [2], [3]. Simply stated, the current state of the art for in-home wireless connectivity systems does not allow to play online games or watch real time stream videos when another application is simultaneously downloading a big file through the same AP.

We demonstrate here that this applications' coexistence can be achieved with a simple and smart modification to the AP that has a minimal impact on existing architecture and protocols. From this point of view, this technique resemble the journeys taken by hippies in the 1960s and '70s, as key characteristics of hippie trails were those of traveling toward a destination as cheaply and respectfully with the environment as possible. Analogously, the proposed *smart AP* aims at reaching an efficient utilization and coexistence of the various entertainment applications run in a wireless home with a solution that has a little cost and a minimal impact on surrounding architecture and protocols.

Specifically, the AP and the associated HEC are in a strategic position that allows them to gather information about the channel condition and the ongoing traffic. This information can be put to good use to regulate the transmission flow of downloading applications in order to produce a smooth traffic that utilize the channel efficiently but, at the same time, does not create queues at the AP. Furthermore, to ensure an easy deployability, this scheme exploits only existing features of standard protocols and a modified AP.

Digital Entertainment in a Home Networking Scenario

The current market is heading toward a wireless house where all the devices (e.g., computers, televisions, intelligent fridges, etc.) are wirelessly connected to the home network and possibly controlled by the HEC.

In this context, take, for example, a mid-class American household where a family of four people lives: two teenage kids and the hardworking parents. Each family member presumably owns several networked personal portable devices such as PDAs, MP3 players, game consoles and digital cameras. All these being also connected to the home network.

Based on the market trends, we also consider that all those devices are wirelessly connected through a Wi-Fi IEEE 802.11g link to a HEC that controls the in-house media distribution and provides access to the Internet as well as to the cable television and companies providing external services (e.g., the alarm company). We also assume that several family members will be accessing the household network at the same time according to their work or leisure needs. In particular, we can consider family scenario summarized by Figure 1: (a) one teenager is watching the movie "Star Wars", streaming it from the close-by HEC; (b) the other one is playing with his latest online game against a crowd of buddies across the Internet; (c) the dad is having a conversation through an IP based video chat; and (d) the mum is downloading the last U2 greatest hit compilation from the Apple iTunes music store. In the above everyday-life picture is worth noticing that each of the aforementioned employed applications features different requirements in terms of network performance, as well as suffers from very specific problems all due to the best effort nature of the Internet transport protocols.

These are as follows:

• **Video Streaming**. Streaming applications are affected by the jitter phenomenon but are resilient to some packet loss; a network designed mainly for video streaming should minimize the jitter.

• **Video Chat** and **Massive Multiplayer Online Games.** Both this applications require a high degree of interactivity, they greatly suffer from delays and packet jitter while may tolerate some packet loss.

• **iTunes Music download**. A music download activity is typically resilient to jitter and delays but decreases the sending rate in presence of packet losses: it hence does not tolerate any error losses (losses that are not generated because of congestion).

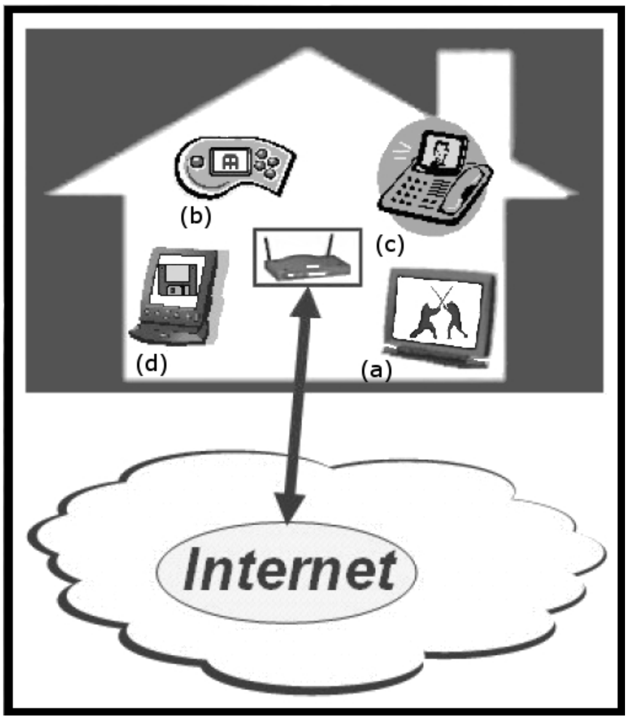

Figure 1. Digital entertainment delivery in a wireless home.

Downloading and Real-time flows on a Wireless HEC: a Difficult Coexistence

Applications can be grouped into two main classes depending on their performance metrics: *downloading* and *real-time*. The first one is concerned with transferring data in a reliable way. Even if data is generally chunked into packets before being transmitted, the performance of these applications are generally measured in terms of how much time is required to have the *whole* file transferred. Examples of this class of applications are file transfer (e.g., FTP, HTTP, SMTP) and remote control (e.g., Telnet). Instead, a second class of applications is concerned with ensuring a quick delivery of *every single* packet transmitted by the application. For these applications, performances are measured in terms of the percentage of packets that reach the destination within a certain time threshold. Interactive on-line games, real-time IP-TV, video/audio chatting, represent typical instances of this class of applications.

Downloading and real-time applications can be distinguished also by the employed transport protocol: TCP or UDP. The former 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 downloading applications. 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 re-established 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 real-time applications.

The lack of congestion control functionalities of UDP had lead the scientific community to wisely consider UDP as unfair toward TCP. Indeed, citing from [4]: *"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."*

However, even if this is true when the available bandwidth is very scarce, the larger and larger bandwidth offered today to home consumers overturns this situation. Indeed, with this broadband connectivity, 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 [5].

Major causes for this problem can be found in the TCP's congestion control functionality. TCP continuously probes for higher transfer rates, eventually queuing packets on the buffer associated with the bottleneck of the connection. Since the same wireless connection might be shared by several devices and applications, it is even more evident how the congestion level and queue lengths can increase, thus delaying the delivery of packets stuck in queue and 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, [6]) 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 [6].

As an example of problems caused by this mixture of causes, we show in Figure 2 the jitter 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. In the considered scenario, the 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 a FTP application (TCP-based) starts to download a file and quickly saturates the channel.

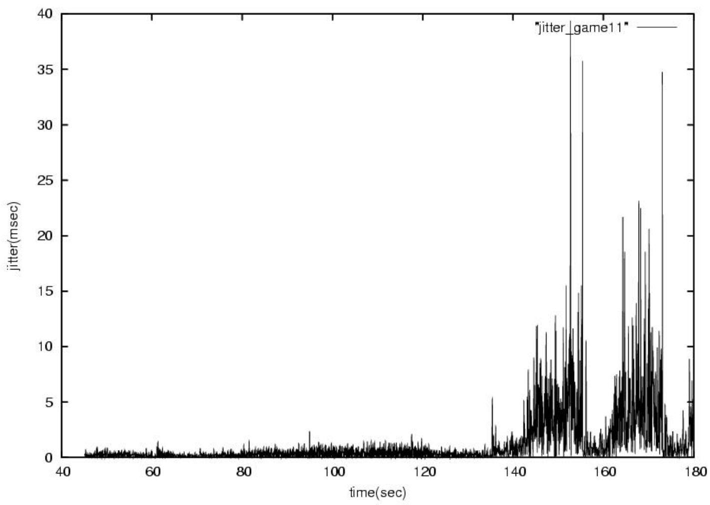

Figure 2. Measured online game jitter; from 135 s, a regular TCP flow is competing for the channel.

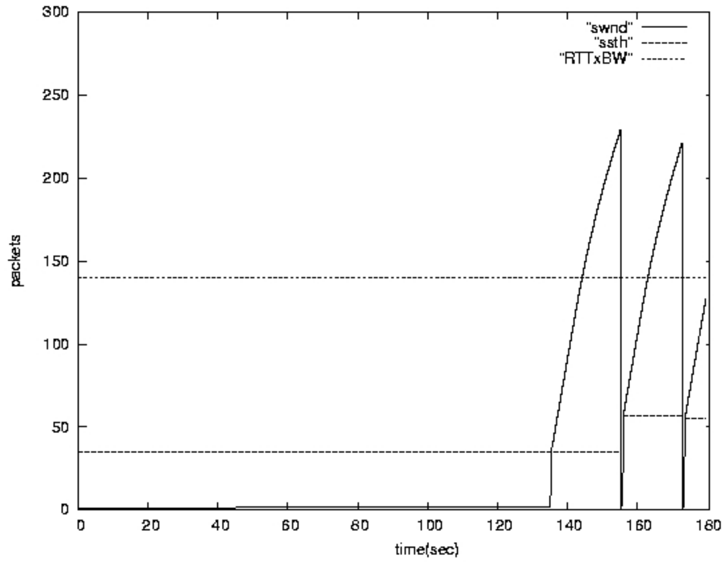

Figure 3. Measured TCP congestion/sending window with a regular TCP flow.

In Figure 2 it is shown that the jitter experienced by online game packets 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 Figure 3 that reports the *congestion/sending window*[1] ("swnd") 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 overlapping Figure 2 and Figure 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].

[1] We use the term "congestion/sending window" to indicate when the sending window exactly corresponds to the congestion window.

Related Work

In the attempt of protecting real-time applications from queuing delays caused by the coexistence with downloading ones, the scientific community has designed the IEEE 802.11e protocol [1], which is a particular version of the well known IEEE 802.11b/g. The IEEE 802.11e is able to assign different priorities to the various flows traversing the AP in order to privilege real time applications with respect to downloading ones. Unfortunately, this protocol requires the source to mark each packet with a priority level value that can then be used to discriminate among flows. Obviously, this poses serious obstacles to its effective deployment since current applications do not have this functionality. Even if in future all the new applications would be developed with the ability to mark packets, still problems will arise if some designer chooses the wrong priority level for her/his novel application or whenever a user will utilize an "old" application.

Another possible solution is represented by the utilization of TCP Vegas in place of the legacy TCP to download files [8]. Indeed, this protocol embodies one of the most cited alternatives to regular TCP in scientific papers and its applicability to the considered problem lies in the fact that TCP Vegas tries to avoid congestion before it happens. In particular, TCP Vegas 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. Nonetheless, 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. If a legacy TCP and a TCP Vegas flows are sharing the same bottleneck, the former fully exploits the available buffer while the latter interprets the consequent RTT trend as an indicator of excessive congestion, thus progressively reducing its sending rate to very low values [9]. In essence, the dramatic efficiency decrease experienced when competing with regular TCP traffic impedes TCP Vegas' factual deployment.

Finally, CLAMP is a protocol that achieves a fair and efficient share of a wireless channel through a distributed algorithm [10]. In particular, the AP and the various wireless nodes cooperate to have transmission flows generated/received by the various nodes limited by the value of their respective fair share. As a result, the total traffic never exceeds the wireless channel capacity thus avoiding queuing and relative delays in packet delivery. Unfortunately, fundamental condition for CLAMP to work is that the AP and *all* the connected nodes have to be modified to incorporate the algorithm. As it is clear, this strict requirement makes CLAMP not factually deployable on a large scale as we cannot assume that in a network there will be no device implementing standard protocols.

Limiting the Advertised Window to Avoid Queuing Delays

Aiming at finding the best solution to the tradeoff relationship existing between TCP throughput and real-time application delays, the two types of traffic should be able to coexist without affecting each other and the employed solution should be easily and factually deployable.

Starting from the last point, i.e., deployability, it is evident how a technique that would exploit existing features of the already utilized protocols could be easily implemented in a real scenario. A possible solution could hence be that of utilizing the advertised window to limit the TCP flow's sending rate and hence the consumed bandwidth.

As it is well known, in fact, the actual sending rate of a TCP flow depends on its current sending window; this value 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) [11]. It is hence evident how the advertised window perfectly embodies a natural upper bound for the sending rate of TCP flows.

Limiting the maximum sending rate of a TCP connection may greatly improve the performance of the HEC [2], [3]. In practice, an optimal tradeoff between the needs for high throughput and low delays could be achieved by maintaining the sending rate of the TCP flows high enough to efficiently utilize the available bandwidth but, at the same time, limited in its growth so as to not utilize buffers. In this way, the throughput is maximized by the absence of packet losses which would halve the congestion window, while the delay is minimized by the absence of queues.

To better understand how limiting the sending window could guarantee the same or even a higher throughput with respect to utilizing regular TCP, we show in Figure 4 a typical saw-tooth shaped sending window of a regular TCP and overlap it with one limited by the advertised window; these two lines also corresponds to the data sending rate. The limited window is more stable than the regular one and, if appropriately chosen, can guarantee the same final throughput. However, as with the limited window the total traffic never exceeds the pipe size, packets will not be queued at the bottleneck.

To put this into practice, we need to address two important issues: how to determine the appropriate upper bound for the sending window and how to factually apply it.

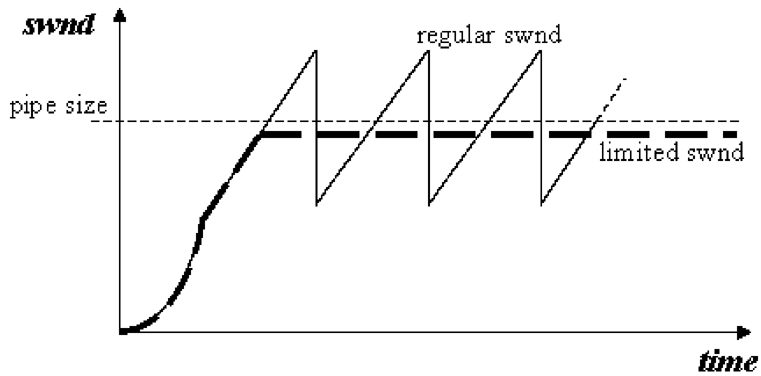

Figure 4. Comparison between regular and limited sending windows (swnd).

Regarding the first point, the most appropriate formula 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 and this transport protocol has no congestion control mechanism. Some smart UDP-based application, however, implements congestion control at the application layer [12].

In any case, to avoid queuing delays, the aggregate bandwidth utilized by TCP flows cannot exceed the total capacity of the bottleneck link diminished by the portion of the channel occupied by the concurrent (UDP-based) real-time traffic.

In essence, the maximum sending rate for each TCP flow at time *t*, namely *maxTCPrate(t)*, is represented by:

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

where *UDPtraffic(t)* corresponds to the amount of bandwidth occupied by UDP-based traffic at time *t*, *#TCPflows(t)* is the concurrent number of TCP flows, and *C* represents the capacity of the bottleneck link.

The second issue that we need to address is how to factually employ this formula in order to have it working in a real scenario. This means i) identifying the location for its implementation, and ii) proposing 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. 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 on which implementing our scheme. 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 which its nominal speed is 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 the number of active TCP connections and the aggregate amount of current UDP traffic. The AP can hence easily compute the best *maxTCPrate(t)* utilizing (1) and accordingly modify the advertised window included in the transiting ACKs.

We refer to this scheme as *Smart Access Point with Limited Advertised Window* (SAP-LAW).

Simulation Assessment

To analyze the wireless home, the NS-2 network simulator (version ns-2.28) was utilized with MAC layer parameters set so as to simulate the IEEE 802.11g. Moreover, the wireless environment was simulated through the Shadowing Model as it represents the most realistic wireless model available in NS-2. Following directions provided by the official NS-2 manual, a home environment partitioned into several rooms was represented through the *path loss exponent* and the *shadowing deviation* parameters equal to 4 and to 9, respectively. The signal attenuation grows with the increase of these two parameters; therefore, by setting the shadowing deviation value with the highest value suggested by the NS-2 manual for an indoor environment (i.e., 9), transmission will suffer from more packet losses.

The HEC is positioned in the middle of the house and endowed with an AP to deliver entertainment to four mobile nodes. Devices inside the house are connected with servers, reachable through the Internet and the bottleneck is situated in correspondence of the last hop (i.e., the wireless connection at home).

As for the simulated traffic, a video stream delivery immediately starts from the HEC to a wirelessly connected screen. Then, at subsequent intervals of 45 seconds other three applications are run, specifically: an online game session, a video-chat conversation, and a persistent download.

Real trace files were used for the movie stream and for the video-chat; these trace files determine packet size and rate for high quality MPEG4 Star Wars IV (for the movie stream) and for two VBR H.263 Lecture Room-Cam (for the video-chat) [13]. Moreover, parameters characterizing the game-generated traffic were inspired by directions provided in scientific literature. Assuming that the player in the house is engaged in the popular first person shooter game Quake Counter Strike with other 25 players through the Internet, game traffic parameters were set as inspired by real measurements reported in [14].

Experimental Results

In order to implement SAP-LAW, the simulated scenario was enhanced by enabling the AP to modify the advertised window (included in returning ACKs) accordingly with (1). In particular, the average UDP-based aggregate traffic was computed through a simple low-pass filter and the new advertised window was determined every 200 ms.

When employing SAP-LAW, the AP is able to keep track of the concurrent real-time traffic and determine the most appropriate advertised window; as a result, the jitter experienced by online game packets is kept low. See, for instance, Figure 5 where C was set equal to 18 Mbps, i.e., 90% of the maximum achievable bandwidth and the maximum jitter value was, in this way, bounded under 10 ms.

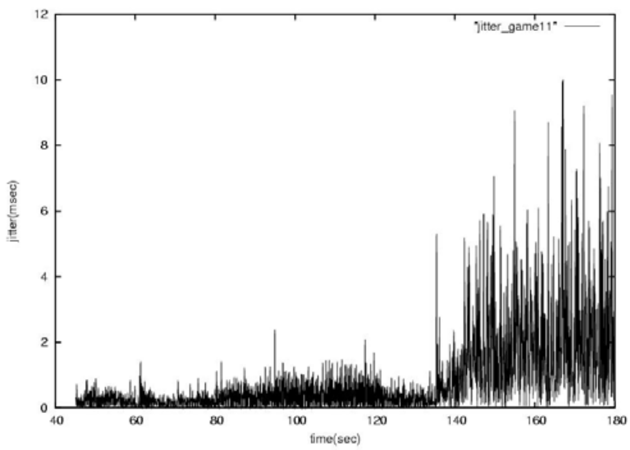

Figure 5. Online game jitter; from 135 s, a TCP flow is competing for the channel; the AP employs SAP-LAW with C = 18 Mbps.

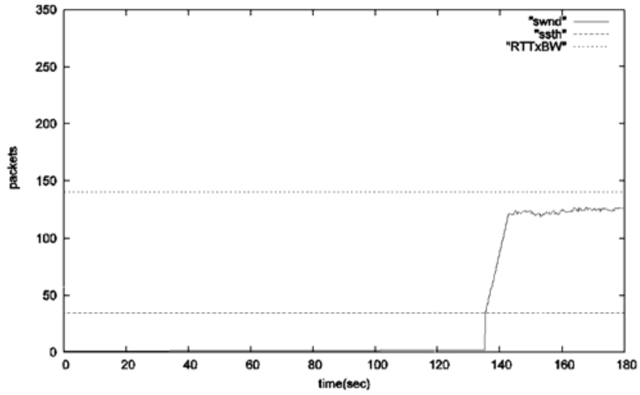

Figure 6. Measured congestion window and advertised window of a TCP flow; the AP employs SAP-LAW with C = 18 Mbps.

At the same time, SAP-LAW guarantees a high TCP throughput by continuously maintaining an appropriately high value of the sending window. To this aim, Figure 6 shows how the advertised window exploited by the TCP limits the sending window, thus eliminating the deleterious saw-tooth shape and guaranteeing a high but also smooth traffic that does not create queues at the bottleneck.

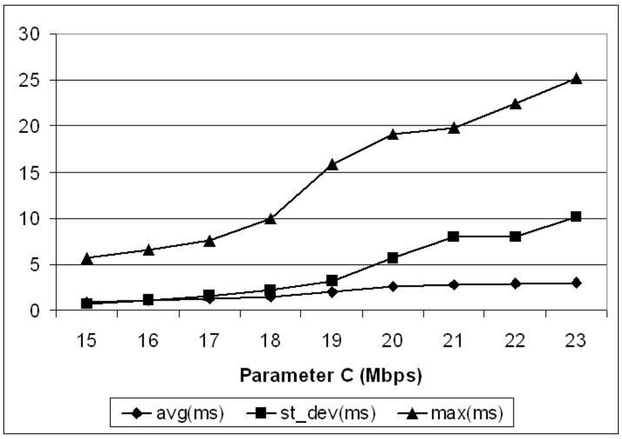

Figure 7. Jitter statistics of the game flow when employing SAP-LAW.

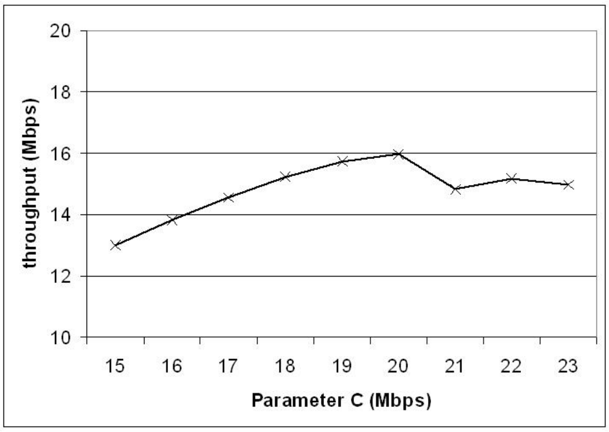

Figure 8. Throughput achieved by the TCP flow when employing SAP-LAW.

Obviously, results obtainable by employing SAP-LAW depends on the chosen *C* value in (1). For a deeper comprehension, various values for the parameter *C* have been tested and results are reported in Figure 7 and Figure 8. In particular, Figure 7 shows the average, the standard deviation, and the maximum value for the jitter experienced by the game flow directed from the server to the client, while Figure 8 presents the throughput trend of the concurrent TCP-based flow.

As clearly shown, both the average and the standard deviation of the online game flow increase with *C*; this is coherent with the fact that higher *C* values decrease the resilience of the scheme to TCP bursts thus leading to some queuing at the AP. Moreover, while the average results very low for all *C* values, the standard deviation sensibly increases with higher values of *C* thus indicating the presence of many peaks of very high delay in the packet delivery. This is confirmed also by the maximum delay values experienced by packets.

Figure 8 demonstrates how the throughput decreases when *C* is set too low. Instead, if *C* is set higher than the maximum achievable throughput on the channel (in this case, 20 Mbps effectively available), then the sender will be allowed to send more packets than those bearable by the bottleneck link causing queuing delays. Thus, it happens that some packets may overflow the buffer and consequent losses cause the reduction of the sending window and average throughput.

Figure 9-11 summarize statistical results obtained by comparing regular protocols (Regular) and SAP-LAW. In particular, considered statistical parameters are the average (Figure 9), the standard deviation (Figure 10), and the maximum value of the jitter experienced by online game packets (Figure 11). Needless to say, the lowest these values, the better the performance. Results obtained by the other real-time applications running in the simulated scenario (i.e., video-stream and video-chat) are coherent with the showed ones and need no further explanation; we hence skip to present their outcomes. Rather, we show in Figure 12 the average throughput achieved by the concurrent (TCP-based) FTP connection. In this case, the highest the value, the better the performance.

As it is evident, employing SAP-LAW conspicuously improves performance both in terms of lowest per-packet delay and achieved throughput. Moreover, SAP-LAW can be easily implemented as it only requires the presence of slightly "smarter" APs. The modifications to the AP are very limited, thus minimally impacting on their cost and, at the same time, SAP-LAW can perfectly coexist with the current Internet and its employed protocols. Considering this and the remarkable achieved results, SAP-LAW is definitely a candidate for enhancing computer-centered home entertainment in a wireless scenario.

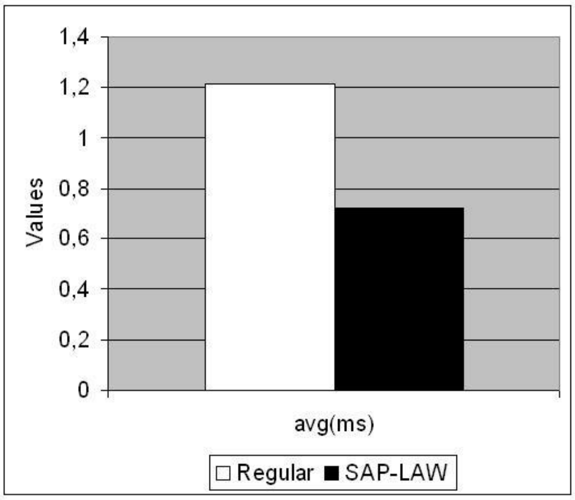

Figure 9. Average of the jitter experienced by online game packets: comparison between regular scheme and SAP-LAW (with C = 18 Mbps).

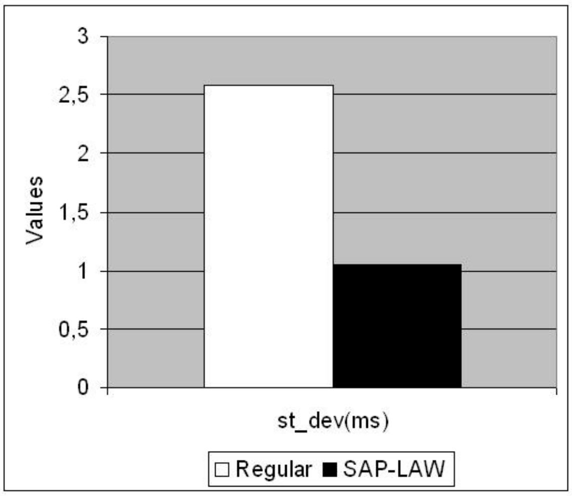

Figure 10. Standard deviation of the jitter experienced by online game packets: comparison between regular scheme and SAP-LAW (with C = 18 Mbps).

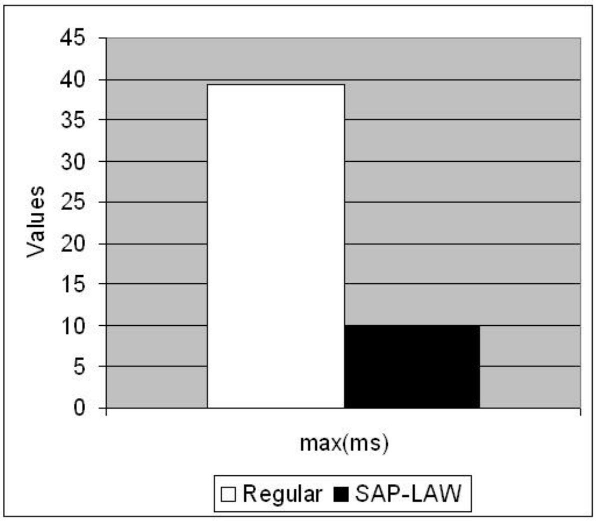

Figure 11. Maximum jitter experienced by online game packets: comparison between regular scheme and SAP-LAW (with C = 18 Mbps).

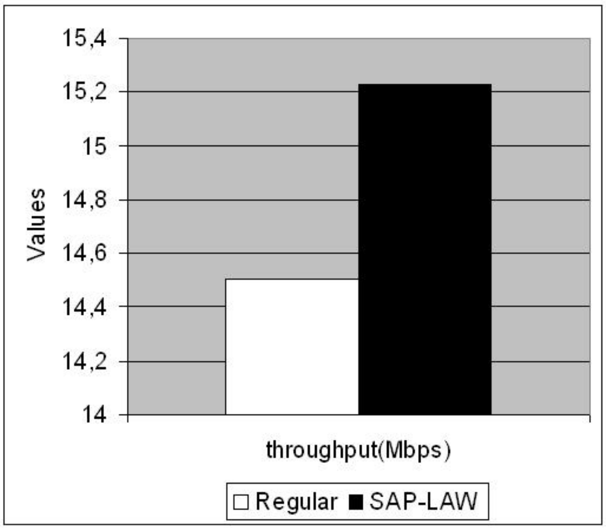

Figure 12. Total throughput achieved by the TCP flow run simultaneously with the real time applications: comparison between regular scheme and SAP-LAW (with C = 18 Mbps).

Summary

Considering a scenario where in-home entertainment is delivered to wireless devices through a HEC, even a single downloading flow 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 even the TCP's lack of buffering control is harmful toward UDP-based applications.

To solve this problem, SAP-LAW utilizes an enhanced AP that does not need to modify existing Internet's protocols. Results showed that SAP-LAW is able to consistently ameliorate the global performance of computer-centered home entertainment services. Moreover, differently from other possible solutions (i.e., IEEE 802.11e, TCP Vegas, CLAMP), 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. It hence emerges as the optimal candidate for enhancing computer-centered home entertainment in a wireless scenario.

Links

1. The TiVo Homepage. http://www.tivo.com/
2. Windows XP Media Center Edition 2005 Home Page. http://www.microsoft.com/windowsxp/mediacenter/
3. Wireless Home Entertainment. http://www.wi-fiplanet.com/columns/article.php/1450881

References

1. 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.
2. C. E. Palazzi, G. Pau, M. Roccetti, S. Ferretti, and M. Gerla, "Wireless Home Entertainment Center: Reducing Last Hop Delays for Real-time Applications," in Proc. of ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2006), Hollywood, CA, USA, Jun 2006.
3. C. E. Palazzi, S. Ferretti, M. Roccetti, G. Pau, and M. Gerla, "*What's in that Magic Box?* The Home Entertainment Center's Special Protocol Potion, Revealed," submitted to IEEE Transactions on Consumer Electronics, Aug 2006.
4. J. F: Kurose and K. W. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, Addison Wesley Longman, Boston MA, USA, 2001.
5. 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 the IEEE 2005 International Conference on Wireless Networks, Communications and Mobile Computing (WIRELESSCOM 2005), Maui, HI, USA, Jun 2005.
6. IEEE, "Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," Specifications, ISO/IEC 8802-11:1999(E), 1999.
7. L. Pantel and 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, May 2002, pp. 23-29.
8. S. Low, L. Peterson, and L. Wang, "Understanding Vegas: a Duality Model," Journal of ACM, Vol. 49, No. 2, 2002, pp. 207-235.
9. G. Marfia, C. E. Palazzi, G. Pau, M. Gerla, M. Y. Sanadidi, and M. Roccetti, "TCP Libra: Exploring RTT-Fairness for TCP," UCLA CSD Technical Report #TR050037, 2005.
10. 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.
11. W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison Wesley, 1994.
12. A. Balk, M. Gerla, M. Sanadidi, and D. Maggiorini, "Adaptive Video Streaming: Pre-encoded MPEG-4 with Bandwidth Scaling," Computer Networks: The International Journal of Computer and Telecommunications Networking, 15, 5, 2004, 415-439.
13. Movie trace files. http://www-tkn.ee.tu-berlin.de/research/trace/ltvt.html
14. J. Färber, "Traffic Modelling for Fast Action Network Games," Multimedia Tools Appl., Vol. 23, No. 1, 2004, pp. 31–46.