INFLUENCIA DEL BUFFER DEL ROUTER EN LA MULTIPLEXIÓN DE JUEGOS ONLINE

José Mª Saldaña, Julián Fernández-Navajas, José Ruiz-Mas, José I. Aznar,
Eduardo Viruete Navarro, Luis Casadesus
{jsaldana, navajas, jruiz, jiaznar, eviruete, luis.casadesus}@unizar.es
Grupo de Tecnologías de las Comunicaciones – Instituto de Investigación en Ingeniería de Aragón
Dpt. IEC. Centro Politécnico Superior Universidad de Zaragoza
Edif. Ada Byron, 50018, Zaragoza

***Summary*- This work presents a study of the behaviour of a Tunneling, Compressing and Multiplexing technique, named TCM, which is able to save bandwidth when many players of the same online game share the same path, as it happens e.g. in an Internet Café. First, a summary of the main characteristics of the system is presented, and then the results of some tests carried out with real machines emulating different buffer policies are shown. Real traces of a commercial First Person Shooter game have been used. The results show significant bandwidth savings, so the user's experience can be improved in certain cases. A final conclusion is that the best parameters for the use of the protocol have to be empirically obtained in each case.**

I. INTRODUCCIÓN

Desde la mitad de los años 90, los denominados *cibercafés* han dado a muchos usuarios la oportunidad de acceder a los servicios de Internet. Hoy en día, aún representan una proporción importante del modo en que los usuarios se conectan a Internet en algunos países [1]. Algunos estudios sobre el perfil de los usuarios de esos establecimientos [2], han concluido que una de las actividades que más se desarrollan en ellos son los juegos *online*. Dos de los géneros más populares son los *First Person Shooter* (FPS) y los *Massive Multiplayer Online Role Playing Game* (MMORPG). Los *cibercafés* están presentes en todo el mundo, pero tienen especial importancia en algunos países en vías de desarrollo [3].

Este escenario, donde varias máquinas comparten la misma conexión a Internet, puede tener características muy variadas: diferentes tecnologías de red, diferentes equipos para conectarse, distintas topologías de red, etc. Sin embargo, todos coinciden en que el ancho de banda suele ser un recurso escaso que debe administrarse bien.

Los juegos FPS generan altas tasas de pequeños paquetes UDP, por lo que si hay varias máquinas generando tráfico, el *router* puede tener problemas para gestionar todos los paquetes. En [4] se presentó una caracterización del tráfico de los FPS, concluyendo que gran parte de los equipos de red están diseñados para gestionar paquetes TCP de gran tamaño. De ahí que su capacidad de proceso pueda suponer un cuello de botella si reciben demasiados paquetes por segundo. Los juegos MMORPG necesitan menos ancho de banda [5] y sus requerimientos de tiempo real no son tan grandes, por lo que no se ven afectados de la misma manera que los FPS. Por esta razón, nos ocuparemos de estos últimos en este trabajo.

En este escenario, un agente local (Fig. 1) puede agrupar paquetes de distintos usuarios, multiplexándolos en paquetes más grandes, y logrando así tanto una reducción en el número de paquetes por segundo como un ahorro de ancho de banda, ya que los paquetes pequeños presentan una baja eficiencia. Esta técnica se ha usado desde hace tiempo para otros servicios multimedia, como por ejemplo *Voice over IP* (VoIP). El agente local puede situarse en una máquina dedicada, o en el *router*, o incluso en el ordenador de uno de los jugadores.

En [6] se presentó un método denominado TCM *(Tunneling, Compressing, Multiplexing)* que, añadiendo pequeños retardos, es capaz de ahorrar un 30% del ancho de banda en el tráfico del cliente al servidor para muchos juegos FPS, llegando hasta el 50% para algunos títulos. Lógicamente, el tamaño de los paquetes aumenta al multiplexar.

En este trabajo estudiaremos el efecto que puede tener el comportamiento del *buffer* del *router* cuando se usa esta técnica. Por un lado, el ahorro de ancho de banda puede ser beneficioso. Pero por otro, el aumento del tamaño de los paquetes puede perjudicar a la calidad para ciertas políticas que penalicen a los paquetes grandes.

El resto del trabajo está organizado de la siguiente forma: la siguiente sección presenta los trabajos relacionados. La sección III resume el comportamiento del sistema. La sección IV presenta las pruebas y resultados, y el trabajo se cierra con las conclusiones.

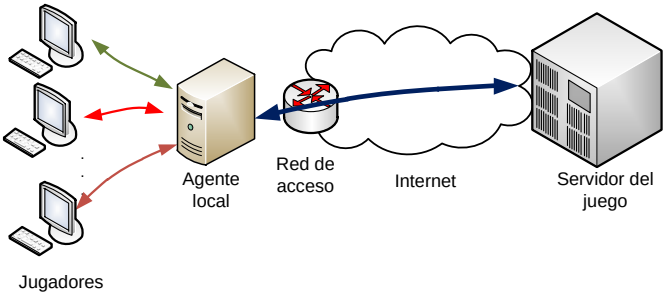

Fig. 1. El agente local multiplexa varios flujos

II. TRABAJOS RELACIONADOS

El escenario que estamos considerando puede basarse en diferentes tecnologías, y lo mismo ocurre con el *router* de acceso, puesto que hay una gran variedad. El problema de dimensionar el *buffer* fue estudiado en la revisión presentada por Dhamdere y Drovolis en [7], donde se explica que hasta hace unos años se aceptaba la regla no escrita de usar el producto retardo-ancho de banda para dimensionarlo; pero esta regla está siendo sustituida por el llamado *"Stanford model"*, que propone *buffer* más pequeños. En el mismo trabajo, los autores proponían el uso de *buffer* limitados en tiempo, una política que penaliza a los paquetes grandes pero resulta interesante para flujos multimedia, ya que mantiene los retardos por debajo de una cota superior. En este trabajo compararemos esta propuesta con los *buffer* grandes.

Respecto al tráfico de servicios en tiempo real, como VoIP, videoconferencia o juegos *online*, podemos decir que los requerimientos de tiempo real hacen que estos programas envíen una gran cantidad de paquetes pequeños por segundo, con lo que tienen poca eficiencia. Se han propuesto soluciones basadas en multiplexión, que incluso se han estandarizado [8] para escenarios donde varios tráficos de tiempo real comparten la misma ruta, como sucede con el servicio VoIP (*trunking*). Se puede incluir en un mismo paquete un mayor número de muestras, añadiendo solamente un tiempo de retención, que se corresponde con el tiempo entre paquetes. Por tanto, a mayor número de flujos multiplexados, mayor eficiencia.

Con relación al tráfico de los juegos FPS, se puede decir que, a pesar de estar desarrollados por diferentes empresas, todos tienen unos patrones de tráfico similares [9]. Por un lado, en el sentido cliente a servidor, cada jugador genera paquetes pequeños (algunas decenas de bytes) a altas tasas. Estos tráficos suelen ser independientes del número de jugadores, ya que cada uno sólo tiene que comunicar sus acciones al servidor [4], [10]. Por otro lado, los paquetes que envía el servidor a los clientes son más grandes, al incluir la información del resto de jugadores: su tamaño depende del número de participantes. En [6] se propuso un método para multiplexar, comprimir y enviar en un túnel el tráfico de cliente a servidor de cierto número de jugadores. Se adaptó el esquema de [8], pero utilizando diferentes algoritmos de compresión, ya que no se utiliza RTP.

Respecto a los juegos MMORPG, recientemente se ha llevado a cabo un estudio usando el tráfico de uno de ellos [11], concluyendo que los esquemas P2P no sirven para este tipo de juegos. Otra conclusión, relacionada con este trabajo, es que agregar mensajes antes de transmitirlos puede reducir tanto el ancho de banda como el retardo para las arquitecturas cliente-servidor y P2P.

III. RESUMEN DEL FUNCIONAMIENTO DE TCM

En este apartado resumiremos brevemente el funcionamiento del algoritmo TCM, mostrando el ahorro que puede proporcionar. Como se ve en la Fig. 1, la idea principal es añadir un agente local que multiplexa en un solo paquete todos los que llegan durante un periodo denominado *T* (Fig. 2). En el caso de haber llegado un solo paquete, se envía en su forma original, ya que el túnel aumentaría su tamaño. La Fig. 3 muestra el esquema de un paquete multiplexado: en primer lugar se aplica a las cabeceras IP/UDP un protocolo de compresión, como IPHC o

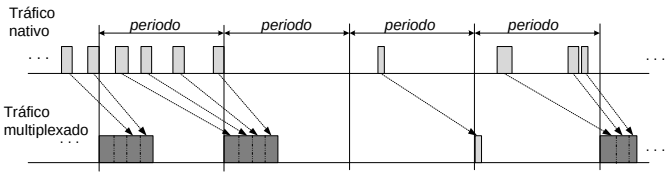

Fig. 2. Método de multiplexión

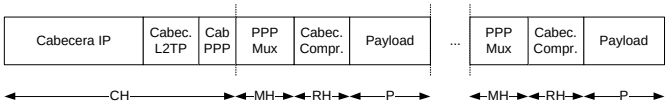

Fig. 3. Esquema de un paquete TCM

ROHCv2; después se utiliza PPPMux y finalmente el paquete multiplexado se envía mediante un túnel L2TP.

Lógicamente, se añaden dos nuevos retardos: en primer lugar, un tiempo de retención que será en media la mitad del periodo. En segundo lugar, un tiempo de procesado que en un caso similar [12] se midió en 1 ms. Los retardos de transmisión en la red local se consideran despreciables, al ser estas redes normalmente más rápidas que Internet.

La relación de anchos de banda *(BandWidth Relationship,* BWR) original y multiplexado para TCM es [6]:

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

Donde *k* representa el número de paquetes que han llegado durante un periodo. *CH, MH, RH* y *P* son los tamaños representados en la Fig. 3, y *NH* se refiere al tamaño de una cabecera IP/UDP, que será 28 bytes para IPv4, y 48 para IPv6. El primer término expresa el caso en que sólo se envía un paquete; el segundo da cuenta de cómo los paquetes multiplexados comparten la cabecera común, y se reduce según crece *E[k]*. El tercer término expresa la asíntota para el *BWR* para los casos en que el periodo y el número de jugadores son lo suficientemente grandes, con lo que $Pr(k>1) \approx 1$ y $E[k|k>1] \approx E[k]$.

Se ha seleccionado para las pruebas el juego *Half Life Counter Strike 1.6*. Aunque tiene ya varios años, todavía sigue siendo popular, representativo del tráfico de los FPS y además existen muchos estudios sobre su comportamiento [9], [10], [13]. La Fig. 4 representa el *BWR* teórico para diferentes números de usuarios y periodos, mostrando su comportamiento asintótico. Como puede observarse, se puede conseguir un 30% de ahorro de ancho de banda usando IPv4.

IV. PRUEBAS Y RESULTADOS

El objetivo de las pruebas que se presentan es el estudio de la influencia mutua entre las políticas del *buffer* y la técnica TCM. Aunque parezca que el *buffer* no influye directamente en TCM, ya que los paquetes primero se multiplexan y luego se envían al *router*, existe una relación que nos dice que cuanto mayor es el periodo, mayor tamaño tendrán los paquetes enviados. El comportamiento de los paquetes en el *buffer* dependerá de sus políticas, que influirán de manera diferente para distintos tamaños de paquete. Se puede indicar además que, como TCM ahorra ancho de banda, reducirá el tráfico total que llega al *router*.

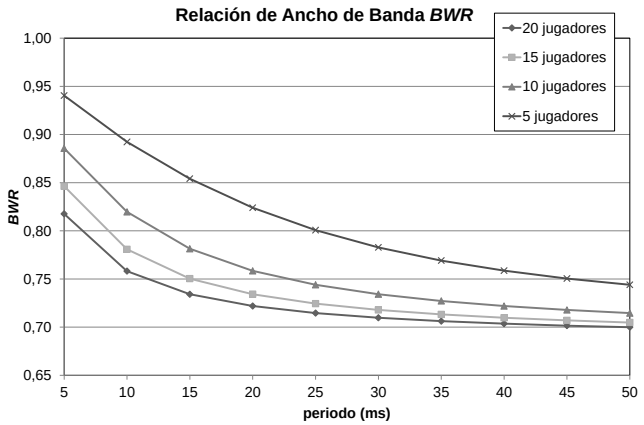

Fig. 4. BWR para Counter Strike 1.6. usando IPv4

Las trazas de tráfico se han obtenido de [14], e incluyen solamente tráfico de juego activo. Se han combinado distintas trazas para obtener la de 20 usuarios, como se indica en [6]. La Fig.5 muestra el escenario usado en las pruebas. Primero, el tráfico del juego y el de fondo se envían usando el generador JTG, que es capaz de enviar las trazas exactamente como eran originalmente, ya que lee los tiempos y tamaños de sendos ficheros. Por tanto, no se ha necesitado usar un modelo para el tráfico. La distribución de los tamaños de paquetes para el tráfico de fondo es: el 50% son de 40 bytes, el 10% de 576, y el 40% restante de 1500 [15].

Ambos tráficos comparten el mismo enlace de acceso, emulado mediante la herramienta Linux *tc (traffic control)*, que permite limitar el ancho de banda a nivel *eth* y también definir el tamaño del *buffer*. El límite se ha establecido en 1 Mbps. El parámetro *burst* se ha establecido en 5000 bytes. El tamaño del *buffer* se define según el tiempo máximo de un paquete en él. Se han usado dos diferentes: un *buffer* de alta capacidad, con un tiempo máximo de 500 ms, y otro limitado en tiempo, de 50 ms.

El tráfico, una vez recogido, se procesa *offline* para añadir un retardo de red, resultante de la suma de uno fijo de 20 ms, asociado a la distancia geográfica, y otro lognormal de media 20 ms y varianza 5. También se añade un retardo de procesado de 5 ms, que da cuenta de los retardos en el multiplexor y demultiplexor [12].

En [13] se llevó a cabo un estudio sobre los usuarios de *Half Life*, y se llegó a la conclusión de que los jugadores no aceptarían retardos mayores de 225-250 ms. Otros estudios más recientes [16] han concluido que se puede lograr una calidad aceptable con retardos de 200 ms para algunos juegos. Por tanto, los retardos añadidos por TCM pueden ser asumibles. Con respecto a las pérdidas, su comportamiento depende del juego: algunos dejan de funcionar a partir de un 4% de pérdidas, mientras que otros funcionan bien incluso con un 35 % [16]. Hemos usado estos valores para los límites de las gráficas.

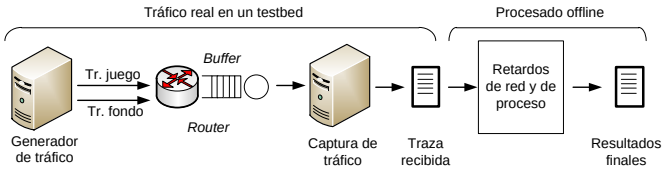

Fig. 5. Esquema de las pruebas

A continuación presentaremos algunas gráficas del retardo en un sentido *(One Way Delay*, OWD) y pérdidas, usando diferentes valores de tráfico de fondo para saturar el *router*. Lógicamente, la multiplexión interesará sólo cuando el tráfico del juego tenga que competir con tráficos de fondo importantes. Para cada *buffer* se han usado tres tráficos: el *nativo*, en el que no hay multiplexión, y otros dos con T = 25 ms y T = 50 ms respectivamente.

La Fig. 6 muestra los resultados del *buffer* de alta capacidad. Se puede ver que aparece un pequeño incremento en el OWD al multiplexar, debido al tiempo de retención (la mitad del periodo) y al de proceso. El ancho de banda del tráfico nativo es de 319 kbps a nivel *eth*. Cuando el tráfico total rebasa el límite de ancho de banda, los retardos aumentan considerablemente. También se puede ver que el ahorro de ancho de banda (unos 120 kbps) se traduce en una mayor cantidad de tráfico de fondo que se puede soportar manteniendo los retardos en un nivel aceptable.

La Fig. 7 se ha obtenido usando el *buffer* limitado en tiempo. Los efectos del retardo son los mismos que para la Fig. 6, pero el uso de este *buffer* tiene la ventaja de mantener el OWD por debajo de 160 ms independientemente del tráfico de fondo.

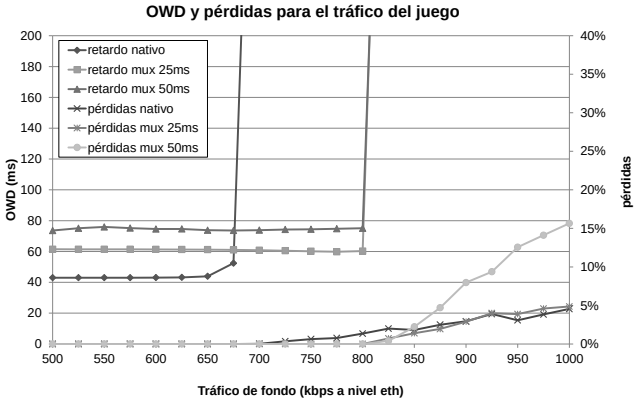

Fig. 6. Retardo y pérdidas para el *buffer* de alta capacidad

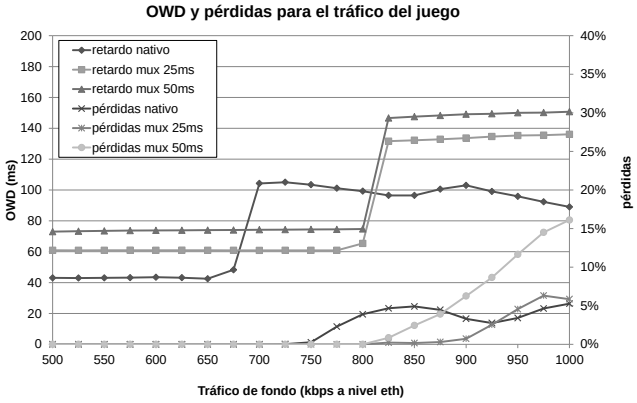

Fig. 7. Retardo y pérdidas para el *buffer* limitado en tiempo

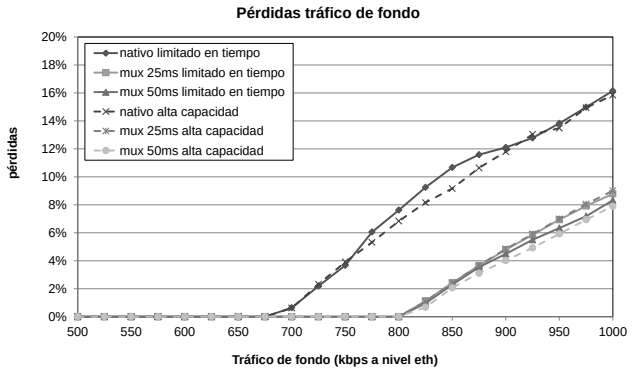

Fig. 8. Pérdidas del tráfico de fondo para ambos *buffer*

Hay otro efecto interesante relacionado con las pérdidas: la utilización del periodo de 25 ms da mejores resultados que el uso de tráfico nativo a causa del ahorro de ancho de banda, siendo además mejor que usar un periodo de 50 ms. Podemos discutir este resultado mirando a la gráfica de 20 jugadores de la Fig. 4: los valores de BWR para 25 y 50 ms son muy similares, por estar muy cerca de la asíntota. De hecho, la diferencia en términos de ancho de banda es de 6 kbps. Pero si calculamos el tamaño medio de paquete, vemos que en el primer caso son 608 bytes, mientras que en el segundo son 1192. Por tanto, si la política del *buffer* penaliza los paquetes grandes, será mejor no usar un valor grande para el periodo.

Pero a partir de 925 kbps de tráfico de fondo se puede ver que el tráfico nativo tiene menos pérdidas que el multiplexado, para los dos *buffer*. La causa es que los paquetes pequeños tienen menos probabilidad de descarte. Vemos, por tanto, que hay situaciones en que multiplexar puede aumentar las pérdidas.

Una primera conclusión es que la multiplexión afecta de distinta manera al retardo y las pérdidas del tráfico del juego según sea la política del *buffer* del *router.*

Finalmente, analizaremos los resultados para el tráfico de fondo. La Fig. 8 muestra las pérdidas para el tráfico de fondo usando ambos *buffer*. Las líneas discontinuas representan los valores del de alta capacidad. Vemos que los resultados son muy similares. El ahorro de ancho de banda se traduce en una probabilidad de pérdidas menor, por lo que observamos que multiplexar es siempre beneficioso para el tráfico de fondo. Podemos concluir que las políticas del *buffer* no tienen una influencia directa en las pérdidas para el tráfico de fondo.

V. Conclusiones

Se ha presentado una comparativa del comportamiento de flujos TCM en función de las políticas de *buffer*, mostrando que la mejor solución no es siempre la que ahorra más ancho de banda. El tamaño de los paquetes se debe tener en cuenta, ya que algunas políticas penalizan los paquetes grandes.

Teniendo en cuenta la gran variedad de *router* que se pueden encontrar en el escenario considerado, las medidas presentadas ilustran la necesidad de particularizar la solución para cada caso concreto: cada red tendrá distintos retardos, diferentes comportamientos respecto a las pérdidas, variadas distribuciones de tráfico de fondo y un límite de paquetes por segundo que el *router* es capaz de gestionar. Por tanto, la técnica TCM nos puede ayudar a adaptar el tráfico al comportamiento de la red. Si dicha red está mejor preparada para bajas tasas de paquetes grandes, podemos modificar el tráfico para adaptarlo a cada tecnología.

Este trabajo forma parte de uno más amplio en el que se abordan temas tales como la caracterización de *router* comerciales, el estudio del tráfico de otros juegos y las distintas posibilidades de ubicación para el multiplexor y demultiplexor, con la finalidad de proporcionar a los fabricantes una técnica flexible que pueda ser útil para mejorar la experiencia del usuario en escenarios en que muchos jugadores comparten la misma ruta.

Agradecimientos

Este trabajo ha sido financiado parcialmente por el Proyecto CPUFLIPI (MICINN TIN2010-17298), por el Proyecto MBACToIP, de la Agencia I+D del Gobierno de Aragón e Ibercaja Obra Social, y por el Proyecto NDCIPI-QQoE de la Cátedra Telefónica, de la Univ. de Zaragoza.

Referencias

[1] S. H. Batool and K. Mahmood, “Entertainment, communication or academic use? A survey of Internet cafe users in Lahore, Pakistan,” Information Development vol. 26. pp 141-147, May 2010.

[2] M. Gurol and T. Sevindik, “Profile of Internet Cafe users in Turkey,” Telematics and Informatics, Vol. 24, Issue 1, pp 59-68, Feb. 2007.

[3] B. Furuholt, S. Kristiansen, F. Wahid, “Gaming or gaining? Comparing the use of Internet cafes in Indonesia and Tanzania,” The Intern. Inform. & Library Review, Volume 40, Issue 2, pp 129-139, Jun. 2008.

[4] W. Feng, F. Chang, W. Feng, J. Walpole, “A Traffic Characterization of Popular On-Line Games,” IEEE/ACM Trans. Netw., pp. 488-500, 2005.

[5] K. Chen, P. Huang, C. Lei: Game traffic analysis: An MMORPG perspective. In Proceedings of the international workshop on Network and operating systems support for digital audio and video (NOSSDAV’05), pp. 19-24. ACM, New York, 2005.

[6] J. Saldana, J. Murillo, J. Fernández-Navajas, J. Ruiz-Mas, J. I. Aznar, E. Viruete, “Bandwidth efficiency improvement for online games by the use of tunneling, compressing and multiplexing techniques”, unpublished.

[7] A. Dhamdhere and C. Dovrolis, “Open issues in router buffer sizing,” Comput. Commun. Rev., vol. 36, no. 1, pp. 87-92, Jan. 2006.

[8] B. Thompson, T. Koren, D. Wing. “RFC 4170: Tunneling Multiplexed Compressed RTP” Nov. 2005.

[9] 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.

[10] T. Lang, G. Armitage, P. Branch, H. Choo, “A Synthetic Traffic Model for Half-Life,” In Australian Telecom, Networks and Applications Conference (ATNAC) Melbourne, 2003.

[11] J. L. Miller, J. Crowcroft, "The Near-Term Feasibility of P2P MMOGs," in Proc. of International Workshop on Network and Systems Support for Games (NetGames), 2010.

[12] H. Sze, C. Liew, J. Lee, D. Yip: A Multiplexing Scheme for H.323 Voice-Over-IP Applications. IEEE J. Select. Areas Commun. Vol. 20, pp. 1360-1368, 2002.

[13] T. Henderson, "Latency and User Behaviour on a Multiplayer Game Server", Lect. Notes Comp. Science, Springer Berlin/ Heidelberg, pp 1-13, vol 2233, 2001.

[14] L. Stewart, P. Branch: HLCS, Map: dedust, 5 players, 13Jan2006. Centre for Advanced Internet Architectures SONG Database, http://caia.swin.edu.au/sitcrc/ hlcs_130106_1_ dedust_5_fragment .tar.gz

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

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