TFMC: a TCP-Friendly Multiplexing Control Scheme for VoIP Flow Transmission

Abdelbasset Trad
INRIA
2004 Route des Lucioles, BP-93
06902 Sophia-Antipolis, France
Email: Abdelbasset.Trad@sophia.inria.fr
Telephone: (+33) 4 92 38 50 22
Fax: (+33) 4 92 38 79 78

Hossam Afifi
INT-INRIA
9 rue Charles Fourier
91011 Evry, France
Email: Hossam.Afifi@int-evry.fr
Telephone: (+33) 1 60 76 47 08
Fax: (+33) 1 60 76 42 91

Abstract— **We investigate the performance limitations in the case of a large number of long distance voice over IP calls originating from different sources and transported through a best-effort IP network. We focus on the potentially negative effects of protocol header overhead, which is twice the voice payload generated by the high-compression audio codecs. These negative effects range from the inefficient use of bandwidth to the network congestion caused by the large number of short voice packets flowing into the IP network, which degrades the real-time transmission performance and creates a fairness problem because of TCP traffic being suppressed. The approach developed in this paper is to adapt the transport protocol and the way in which it interacts with the network in order to support voice flows competing with TCP traffic in the Internet environment. We propose a new multiplexing scheme of RTP voice streams called TFMC (TCP-Friendly Multiplexing Control) that uses TCP-friendly equation-based congestion control for unresponsive flows. TFMC sender adapts the throughput generated by protocol header in response to congestion while maintaining a steady voice data throughput in order to achieve requirements of voice over IP applications where a smooth sending rate is of importance. We discuss through analytical and simulation results the TFMC scheme performance. Simulations show that the proposed equation for adjusting the number of multiplexed packets achieves efficient voice flow transmission and improves packet delay while being fair to TCP.**

Keywords— **VoIP, header overhead, RTP flow multiplexing, TCP friendliness.**

I. Introduction and Motivation

The flexibility of the Internet architecture has expedited the convergence of data communications (packet switched), and voice-based communications (traditionally circuit switched) into a single IP-based core architecture. The integration of voice data transmissions over the Internet offers an opportunity for large savings in communication cost. From the user's viewpoint, making long-distance telephone calls via the Internet results in substantial cost reduction since international charge imposed by telecommunication companies are bypassed [1]. For the network operator, IP telephony have two advantages. First, audio codecs based on advanced voice-compression techniques, can be used to generate low bit data rates (less than 10 kbps). The second advantage is due to efficiency achieved by connectionless mode where bandwidth is consumed only when voice packets are delivered. Although IP telephony systems originally came into the spotlight as networks that could enable cheap long-distance and international telephone calls, attention is now being focused on IP telephony systems as networks that will replace the telephone network and become the base for the next generation of multimedia communications [2]. A single converged network for transport of voice and data will be used. The quality of current Internet real-time voice transmission is not satisfactory because of the Internet's delivery and scheduling mechanisms, originally designed to support elastic data applications. Latency sensitive traffic, such as voice and audio have unacceptable performance if long delays are incurred. A bounded delay on the voice delivery can be achieved either by restricting the offered load (e.g., by blocking voice calls) or by adding QoS mechanisms to the Internet, i.e., service differentiation and traffic engineering which are not yet widely deployed due to their complexity and scalability problems. In this paper, we consider that a major performance limitation for voice transmission over the Internet is the inefficient use of network resources, such as buffers and bandwidth due to packet header overhead and hence propose improvements. We develop a new approach where the network participates in controlling its own resource utilization, in order to ensure acceptable performance for both real-time and elastic traffic sharing network resources. The proposed solution adapts both the transport protocol, by supporting RTP flow multiplexing, and the way in which it interacts with the network state, by regulating the number of RTP streams multiplexed in one UDP packet. We introduce a new adaptive congestion control scheme for RTP voice traffic called TFMC (TCP-Friendly Multiplexing Control). TFMC is applied at network edges; it adjusts the number of UDP voice packets transmitted between two peer gateways to the congestion level of the network. That is, based on the feedback information from the destination gateway, the sender gateway would reduce the number of multiplexed RTP packets during normal load situations and increase it otherwise. With TFMC scheme, no QoS guarantees can be made, but the user-perceived quality is improved because delay is reduced. The paper is organized as follows. Section 2 describes the process of voice transmission through IP networks. Section 3 states the problem. In Section 4 we discuss related work. Section 5 presents the TFMC scheme and shows its performance, and Section 6 concludes the paper.

II. Voice Transmission through IP Networks

In order to reduce bandwidth usage, low-bit-rate voice codecs are used in IP telephony systems (Table I). The commonly used are G.723.1 [3], G.729A [4] and GSM [5]. When silence suppression scheme is employed, the codecs

TABLE I
LOW BIT RATES GENERATED BY THREE STANDARD IP TELEPHONY SPEECH CODECS

Codec	G.723.1	G.729A	GSM
Bit rate (kbps)	5.3/6.3	8	13.2
Frame interval (ms)	30	10	20
Lookahead delay (ms)	7.5	5	0
Total encoding delay (ms)	37.5	15	20
Decoding delay (ms)	18.75	7.5	20
IPv4/UDP/RTP header (bytes)	40	40	40
Payload (bytes)	20/24	20	33
IP bandwidth (kbps)	15.96/16.96	24	29.2

then operate in two states: a silent state at zero bit-rate and an active state at the compressed bit-rate. Regardless of the state, the frame period and frame size are still fixed [1]. For voice, samples representing 20 ms are considered the maximum duration for the payload. This duration is a compromise between bandwidth requirements and quality. Smaller payloads demand higher bandwidth per channel band, because the header length remains at 40 bytes. However, if payloads are increased, the overall delay of the transmission will increase, and the system will be more susceptible to the loss of individual packets by the network. If an 8 kbps algorithm such as G.729A is used, the total bandwidth required to transmit each voice channel would be 24 kbps (as shown in Table I). After the digitization and compression operations, the voice frames are transmitted through the IP network. Since VoIP applications care more about time of delivery than reliability, they typically use the UDP transport protocol. UDP does not provide the underlying support for quality of service, it gives more flexibility in addressing application-specific requirements. Most of the current real-time applications over the Internet are based on UDP and the Real-time Transport Protocol (RTP/RTCP) [6][7] which is rather an application layer protocol. Although RTP does not contain any mechanism that guarantees the timely delivery of data, it provides a mechanism to time-stamp packets so that random delays resulting from other network traffic can be compensated by the use of buffers at the destination. RTP is currently widely used for multimedia communication and particularly for IP telephony in the Internet.

III. Problem Statement

We consider that a major performance limitation for voice transmission over the Internet is the inefficient use of network resources, such as buffers and bandwidth due to packet header overhead, especially when a large number of voice communications are transmitted between two edge gateways of an IP backbone network.

A. Studied Architecture

We focus on the case of a large number of voice sources at an access network, sharing a common path and destined to different users in remote networks. IP backbone networks represent an important part of the end-to-end path for long distance VoIP calls, including calls that are serviced by a combination of a switched telephone network in the local area and the Internet for the long haul (Figure 1). In

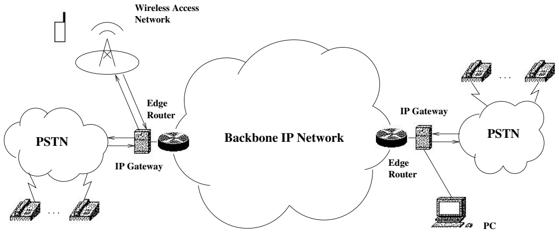

Fig. 1. Heterogeneous mix of data and real-time services transmitted on the same IP infrastructure

the considered architecture, we suppose the existence of a gateway between the communicating entities. Nevertheless, research into IP telephony is going toward an end-to-end voice communication without going through a telephony gateway [8]. In addition to phone-to-phone communication, PSTN/Internet gateways can be used to provide RTP supported phone-to-PC and PC-to-phone communications since RTP is integrated into the H.323 [9] and SIP [10] protocol stack. Call signaling protocols can be used in conjunction with data transport protocols to build a complete multimedia architecture. Typically, these architectures will include protocols such as the RTP for transporting real-time data and providing QoS feedback, the Media Gateway Control Protocol (MEGACO) [11] for controlling gateways, and the Session Description Protocol (SDP) [12] for describing multimedia sessions. In this paper, we deal with the transport mechanism of RTP voice flows when a large number of active calls occur between a pair of gateways.

B. Performance Limitations

High-compression voice codecs used in IP telephony systems improve bandwidth efficiency enormously. However, the payloads are relatively small compared to the additional overhead imposed by the network to pass the audio data between the sender and the receiver. This overhead results in a decreased usable band. For instance, a typical IP voice packet consists of a header of 40 bytes. Compared with the typical payload size of only 10-33 bytes for each audio frame, the header overhead is clearly very substantial. Current VoIP applications tackle this problem by embedding multiple audio frames into a single packet at the source to increase the ratio of payload to header size. This approach has the benefit of reducing the overall data rate of a call. But, since an audio frame is generated only after raw audio signals in a frame period are captured and encoded, packing an additional audio frame will add another frame period to the assembly delay. Together with the existing network delay, the resultant end-to-end delays may become unacceptable [1]. In local area networks where bandwidth is abundant, VoIP applications can send each audio frame in a separate RTP packet to minimize packetization delay. However, in case of Internet telephony gateways with multiple RTP streams, the bandwidth that the header occupies must be taken into consideration. Especially on backbone facilities where costs are high (e.g., some global cross-sections). For example, carrying Voice over IP headers for the entire voice load of a large network with 300 million or more calls per day could consume on the order of 20-40 gigabits per second on the backbone network for headers alone [13]. From another view point, UDP traffic is unresponsive to congestion and therefore can completely monopolize the available bandwidth. Thus, Internet load increases because of large numbers of short voice UDP packets, with 100 packets flowing every second in both directions for each call into the IP network, eventually resulting in large delay, jitter and packet loss [2]. Performance problems will be experienced, in this case, by all voice calls and also by other traffic (i.e., TCP traffic) sharing the best-effort IP network.

IV. Summary of Related Work

In this section, we briefly describe the related contributions: the IP/UDP/RTP header compression standard, the multiplexing schemes of RTP voice flows and the TCP-friendly congestion control mechanism for unresponsive flows. We have coupled the two latter mechanisms to develop our multiplexing scheme.

A. IP/UDP/RTP Header Compression

An IP/UDP/RTP header compression scheme for low speed serial links was proposed by Casner and Jacobson [14]. Their scheme reduces the IP/UDP/RTP header size from 40 bytes to a minimum of 2 bytes on a link-by-link basis. Casner's algorithm takes advantages of two properties in RTP streams. First, most of the fields in the IP, UDP and RTP headers do not change over the lifetime of an RTP session. These constant-value fields can be represented by fewer bits with a session context during transmission. Second, RTP header fields like sequence number and timestamp are increased by a constant amount for successive packets in a stream. Hence, differential coding can be applied to compress these fields into few bits. Recently, a multiplexing scheme for RTP streams was proposed in [15], it combined the IP/UDP/RTP compression with a point-to-point protocol (PPP) multiplexing scheme [16] to form a method for end-to-end tunneling of multiplexed RTP packets. However, this proposal operates in the link layer and can only be used over PPP links.

B. RTP Multiplexing Schemes for Voice Flows

Multiplexing aims at the reduction of the overhead associated with Internet protocol layers and traffic load (i.e., number of packets) on routers. The basic assumption for the multiplexing is that, at any time, there is more than one user communicating with the same remote location. Various approaches for multiplexing VoIP streams between peer gateways have been proposed [1][8][17][18][19]. The basic idea is to multiplex voice calls between two gateways into a single packet, instead of using a separate connection and thus separate packets for each. In [8], the authors propose to multiplex RTP streams sharing the same destination gateway into one UDP packet at a multiplexing interval period. This solution is simple and there is no additional header for multiplexing; it is effective for both header overhead and number of packet reduction. In typical case, 40% of the bandwidth is saved for 8 multiplexed G.723.1 encoded voice streams by header overhead reduction and number of voice packets also decreases 1 by 8. In [18], the proposed scheme consists in merging several audio payloads coming from different users into a single packet. Only one IP/UDP/RTP header and a new RTP mini-header is added to the packet. The mini-header is required to delineate the multiplexed packets. The protocol adds 16 bits of overhead per multiplexed user. Although this scheme allows a big reduction of the header, it adds some protocol complexity and requires modifications of the RTP packet format defined in [6]. In [19], a light-weight data driven multiplexing approach is introduced. The basic idea is to replace the IP/UDP/RTP header of each packet with a mini-header at the edge router. The mini-header is a two-byte tag that replaces the original header at the ingress router, and will be used to reconstruct the original header at the egress router using a mapping table kept at each of the access routers. A control signaling protocol is also proposed to exchange simple control signals between peer entities. The main drawback of these schemes is the added delay. Two types of delay are incurred. Multiplexing processing delay occurs in the gateway given that the generation of a multiplexed packet is triggered by the expiration of the multiplexing period or by the arrivals of enough audio packets to fit into the MTU. The multiplexing period is decided by the implementor of an IP gateway. If the chosen multiplexing period is small, the additional delay becomes small but the number of users in a multiplexed channel also becomes small [8]. It is possible to adjust the multiplexing period according to the number of existing connections so that the system can support all the voice calls with the smallest possible delay to attain the best-possible conversation quality. Multiplexing is likely to reduce losses on the Internet. First, because improved efficiency results in a reduced overall bit rate for the voice stream and thus the necessary bandwidth. Secondly, many routers drop packets not because of link congestion and buffer overflow, but because of processing overload. A burst of small packets can overwhelm the processors on a typical router, causing loss.

C. TCP-Friendly Rate Control for Unresponsive Flows

Unresponsive flows are flows that do not use end-to-end congestion control and, in particular, that do not reduce their load on the network when subjected to packet drops. This behavior can result in both unfairness and congestion collapse for the Internet. TCP-friendly equation-based rate control for unresponsive flows inside best-effort networks was introduced [20] to ensure proper congestion avoidance for multimedia applications using unresponsive transport protocols, i.e., UDP and RTP, while coping with the real-time needs of these applications. That is, in contrast with the behavior of TCP, to smoothly find available bandwidth, increase the sending rate slowly in response to a decrease in the loss event rate and to do not halve the sending rate in response to a single loss. The basic decision in designing equation-based congestion control is to choose the underlying control equation. In [21], a TCP-friendly flow is characterized by an upper bound of its arrival rate. This bound is given by the maximum overall sending rate for a TCP connection with a given packet drop rate, packet size, and round trip time. Given a packet drop rate of p, the maximum sending rate of a TCP connection, in absence of timeouts, for packets of S bytes with a fairly constant round-trip time, including queueing delays, of R seconds is given by (1):

$$T(Bps) \leq c * \frac{B}{R * \sqrt{p}} \tag{1}$$

with

$$c = 1.5 * \sqrt{\frac{2}{3}}$$

Based on the TCP-friendly rate control mechanism, many schemes have been developed in order to provide appropriate congestion control for real-time applications in the Internet environment. In [22], the authors propose a receiver-based multicast congestion control scheme where the receivers compute round-trip times, estimate the packet loss rate p, and use (1) to compute the rate at which they should receive data. In [23], the authors propose an end-to-end rate adaptation scheme suitable for unicast applications, that adjusts the transmission rate of multimedia applications to the congestion level of the network. That is, based on the estimation of the loss rate and the round-trip times obtained from the regular information of RTCP reports [6], the sender increases the transmission rate during network underload periods and reduces this rate during congestion periods, while avoiding an aggressive adaptation behavior.

V. Proposed TCP-Friendly Multiplexing Control Scheme for RTP Voice Flows

The common principle for the TCP-friendly rate control schemes proposed in [21][22][23] is that they incorporate congestion control mechanism at the application level of one user, i.e., they instruct the applications at the end systems to adapt the bandwidth share they are utilizing to the network congestion state. In contrast with these schemes, our TFMC (TCP-Friendly Multiplexing Control) scheme is applied at the transport level on a flow formed by multiplexed RTP voice flows originating from different sources between two edge IP gateways.

A. TFMC Scheme Overview

TFMC scheme is based on the idea of multiplexing an adjustable number of voice packets, this number is adapted to the current network congestion state. Since voice over IP applications use standard voice codecs which are unresponsive to congestion indication, TFMC adapts the control traffic (i.e., packet headers) of unresponsive voice flows to the network congestion state using TCP-friendly equation-based rate control mechanism. TFMC uses a simple multiplexing scheme described in [8], where header overhead is reduced through multiplexing several RTP streams destined to the same gateway into one UDP packet. Multiplexed packets are generated from different sources and occur at the same instant into the sending gateway. We assume that at any time there exist enough voice packets to fit into the multiplexed packet (i.e., there are enough simultaneous calls sending voice data). The multiplex is formed by linking a series of RTP streams and an IP-UDP header (Figure 2).

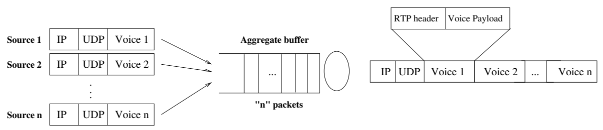

Fig. 2. Packets from different RTP flows multiplexed into one UDP packet

Multiplexed packets are sent using one of the UDP ports designated for the multiplexed streams. At fixed time intervals, the receiver computes the loss rate observed during the previous interval. The sender, based on the receiver's feedback information, updates its sending rate by adjusting the number of packets to multiplex (Figure 3). We assume

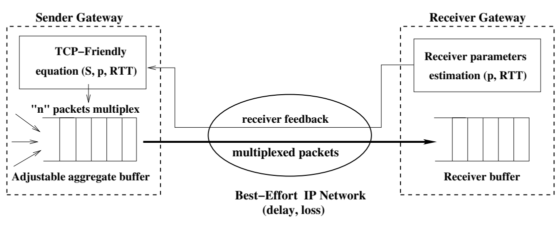

Fig. 3. Adjusting the multiplexing buffer size in response to traffic fluctuations using the TCP-friendly congestion control mechanism

that the delay and the packet loss rate are affected by the packet size.

1) Sender Gateway Functionality: Basically, the sender estimates the values for the round-trip time R, the retransmit timeout value t_{RTO} and the mean payload size of voice flows. The sender and receiver use sequence numbers of a multiplexed RTP flow to estimate the round trip time in the network. Every time the receiver gateway sends a feedback message, it echoes back to the sender the sequence number of one RTP flow received in most recent multiplexed packet, and the time since that packet was received. The retransmit timeout value, t_{RTO}, can be estimated from R. In practice the simple empirical heuristic of $t_{RTO} = 4R$ works reasonably well to provide fairness with TCP [26]. The payload size of voice frames varies depending on the speech codec used by one source. Typical payload size is 10-33 bytes for each voice frame (Table I). To take into account this variation, TFMC sender estimates the mean RTP payload based on the payload size of coded voice frames that occur at the sender gateway in the time interval between two receiver feedbacks. TFMC sender uses the TCP throughput equation (2) developed in [24] as the sending rate approximation. Equation (2) roughly describes TCP's sending rate as a function of the loss event rate, round-trip time and packet size. This equation reflects TCP's retransmit timeout behavior, as this dominates TCP throughput at higher loss rates [25].

$$T(Bps) = \frac{S}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \quad (2)$$

Initially the number of packets to multiplex, m, is set to 1 and the sending rate is determined by the rate of existing voice flows coming at the sender gateway. When a feedback message is received, the sender changes the number of packets to multiplex based on the information obtained from the receiver gateway. Assuming that at a given instant the gateway is sending a flow of m_1 multiplexed voice packets having a packet size of $S(m_1)$ and a total throughput rate of T_1 bytes/sec, then after receiving the feedback message from the receiver gateway the sender measures the round-trip time estimate, updates the retransmission timeout value. The loss rate obtained from the receiver gateway, p_1, and the measured round-trip time, R_1, are then fed into the throughput equation (2), to give the new acceptable sending rate T_2. The sender gateway then adjusts the number of multiplexed RTP packets to match the calculated rate T_2 while conserving the rate of RTP packets transmitted in the previous time interval using the equation (3) that we have introduced:

$$m_1.\frac{T_1}{S(m_1)} = m_2.\frac{T_2}{S(m_2)} \quad (3)$$

where $S(m)$ is the size in bytes of a multiplexed packet formed by m RTP voice packets and one IP-UDP header, given by the formula below.

$$S(m) = h_{ip} + h_{udp} + m * (h_{rtp} + Payload) \quad (4)$$

The proposed equation (3) determines the number of packets to multiplex according to the network congestion state. The sender will increase the number of multiplexed RTP packets, m_2, if there was a high traffic load during the previous time interval indicated by a calculated sending rate T_2 less than the actual sending rate T_1. Otherwise, the sender will decrease the number of multiplexed packets during normal load periods. Actually, TFMC sender adapts the throughput generated by protocol header in response to congestion while maintaining a steady voice data throughput in order to achieve requirements of voice over IP applications where a smooth sending rate is of importance. TFMC responds to changes of network congestion by adjusting the number of multiplexed packets, and hence adjusting the number of transmitted packet headers.

2) Receiver Gateway Functionality: The receiver gateway periodically sends a feedback message reporting the loss event rate p to the sender. Every time the receiver sends a feedback message, it echoes back to the sender the sequence number of one RTP flow received in most recent multiplexed packet, and the time since that packet was received. The receiver keeps a packet history in order to detect loss of multiplexed packets. Packet loss is detected using RTP sequence numbers related to one multiplexed flow. The estimated loss rate measure the loss event rate rather than the packet loss rate. A loss event can consist of several packets lost within a round-trip time, as it is discussed in [26]. In the studied case the sender gateway is transmitting at a high rate (many packets per RTT), but the receiver sends one feedback message per several multiplexed flows. This avoids bursts of control packets and improves the scalability of TFMC scheme.

3) Discussion of the Variable Size of TFMC Packets: Originally, TCP-friendly rate control mechanism was designed for applications that use fixed packet size, and vary their sending rate in packets per second in response to congestion. TCP-friendly rate control mechanism should not be used for applications that vary their packet size instead of their packet rate in response to congestion [25]. Varying the packet size during the time interval between two estimations of the sending rate distorts packet-based measurement of the loss event. TFMC adapts its sending rate by adjusting the number of multiplexed packets, consequently the packet size is varied. However, TFMC varies the packet size only after the estimation of the sending rate using the TCP-friendly throughput equation and keeps this size fixed until the next feedback message. Therefore, TFMC sending rate estimation is quite accurate.

4) Saving Bandwidth by TFMC Scheme: Without multiplexing, the bandwidth required for the transmission of n RTP voice packets is given by:

$$B_n = n * (h_{ip} + h_{udp} + h_{rtp} + Payload) \quad (5)$$

With our multiplexing scheme, the bandwidth required for the transmission of the same amount of voice data by multiplexing m RTP voice packets into one UDP packet (assuming that n is a multiple of m) is given by:

$$B_{n,m} = \frac{n}{m} * (h_{ip} + h_{udp} + m * (h_{rtp} + Payload)) \quad (6)$$

The bandwidth saved by multiplexing m RTP voice packets can be then calculated from (7):

$$\beta_m = \frac{B_n - B_{n,m}}{B_n} = \frac{(1 - \frac{1}{m}) * (h_{ip} + h_{udp})}{h_{ip} + h_{udp} + h_{rtp} + Payload} \quad (7)$$

Figure 4 illustrates the percentage of bandwidth saving vs. number of multiplexed packets for a typical payload size of 20 bytes.

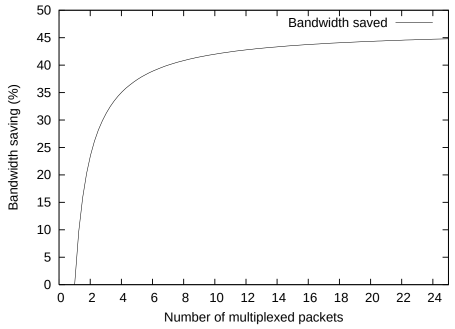

Fig. 4. Bandwidth saving as the number of multiplexed packets increases

We notice that the bandwidth saved increases quite significantly when the number of multiplexed packets is varied from 1 to 10 packets. With 10 multiplexed packets having a payload of 20 bytes, a significant bandwidth saving of 42% is achieved. Note that the multiplexed-packet length is bounded by the MTU.

5) Comparison of TFMC Scheme with Header Reduction Schemes: The IP/UDP/RTP header compression scheme proposed in [14] reduces header size from 40 bytes to a minimum of 2 bytes in the best case (one condition is that UDP checksum from the source is disabled). The IP/UDP/RTP header compression method also relies on the the link layer for exchanging control messages in order to preserve a lossless compression: the IP header checksum is elided assuming that the link layer is providing good error detection (e.g., PPP's CRC). The total length field (in the IP header) is eliminated since it is considered as redundant with the length provided by the link layer. While this method offers a full restoration of the IP/UDP/RTP header, it is link-by-link based as opposed to our transport layer scheme that provides a simple header reduction method (concatenating voice packets from different RTP streams into a single UDP packet). TFMC scheme is not supported by the link layer, it is applied at the transport layer in which packets may traverse several links. Thus, TFMC is more suitable for latency sensitive voice traffic. Compared to the multiplexing schemes proposed in [1][8][17][18][19], our multiplexing scheme is characterized by the variability of the number of multiplexed packets. This variability is controlled by the proposed equation (3) that uses TCP-friendly rate control mechanism for unresponsive flows. By varying the number of multiplexed packets using equation (3), TFMC sender adapts the throughput generated by protocol header in response to congestion while maintaining a steady voice data throughput. The number of multiplexed packets represent a compromise between the additional multiplexing delay and the number of users in a multiplexed channel (as explained in Section IV.B). By using TCP-friendly rate control, TFMC responds to changes of network congestion by adjusting the number of multiplexed packets. Hence, the smallest possible delay will be obtained. Moreover, being TCP-friendly reduces the network congestion and consequently reduces loss of voice packets. This will help to attain the best-possible conversation quality.

B. Simulation Based Evaluation of the TFMC Scheme

We investigate, through simulations performed in network simulator (ns-2), the performance of TFMC scheme in terms of bandwidth utilization and delay of voice packets as well as the behavior of TCP flow sharing a bottleneck link with voice flows. This link have a capacity of 10 Mbps and a propagation delay of 10 ms. The routers use drop-tail queuing. Voice sources are connected between two nodes representing the TFMC sender and the TFMC receiver (Figure 5). Packet voice streams are generated using a constant bit rate (CBR) source of 24 kbps (which represents the throughput generated by voice packets using G.729A codec considering the headers). A TCP Reno connection

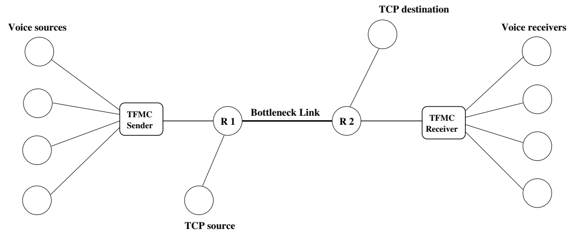

Fig. 5. TFMC voice flow competing with TCP cross traffic

with a packet size of 1000 bytes is initiated at the time 0 seconds for the duration of the hole simulation. At the time 20 seconds, 200 audio sources connected to the TFMC sender start the generation of CBR flows destined to the receiving nodes connected to the TFMC receiver. Figure 6 represents the throughput of the TCP flow and the total throughput of non-multiplexed voice flows measured on the bottleneck link. One observation is that TCP throughput was dramatically decreased when the audio sources have started transmission. Unresponsive UDP traffic monopolizes the available bandwidth and cause starvation for TCP traffic competing in the same congested drop-tail queue. TCP then reduces its sending window in response to congestion. However, when TFMC is used, we notice that voice flows get less bandwidth than TCP flow (Figure 7). TCP flow maintained a steady sending rate (more than 5 Mbits/s) after the audio sources have started the transmission. This can be explained by the decrease in the number of voice

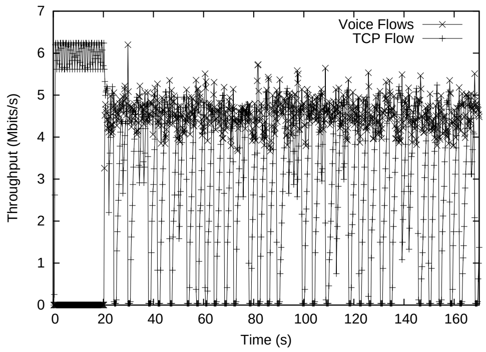

Fig. 6. Throughput of non-multiplexed voice flows competing with TCP traffic

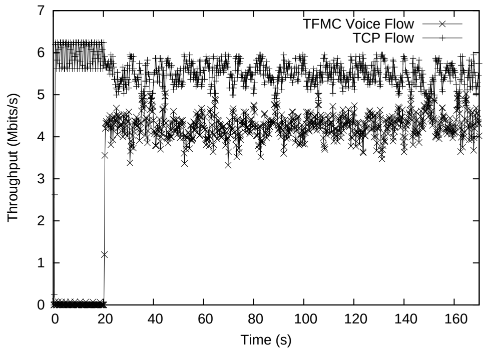

Fig. 7. Throughput of TFMC voice flow coexisting with TCP traffic

packets through multiplexing achieved by TFMC in order to have a TCP-friendly behavior. TFMC flow is smoother than non-multiplexed voice flows and it requires less network bandwidth, this is because of header overhead reduction.

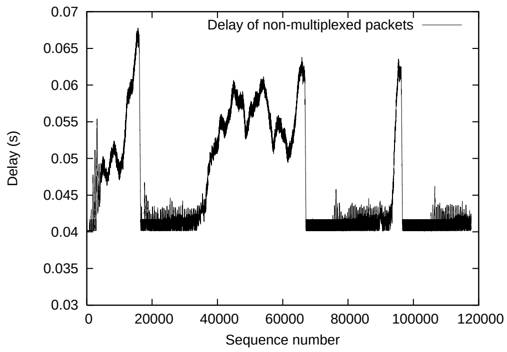

Fig. 8. Delay of voice packets when TFMC mechanism is not used

Figure 8 illustrates the measured delay of one voice flow when multiplexing is not applied and Figure 9 represents this delay when TFMC is used. An interesting result is that TFMC, globally, achieves better delay than non-multiplexed flows (delay reduced by 20 ms for some packets) and ensure less variations in the delay. Maintaining low delay variation will reduce the delay jitter and this will ameliorate the perceived voice quality. This result can be explained by the fact that TFMC decreases dynamically the header traffic

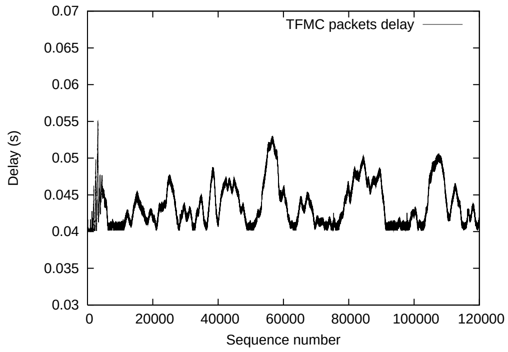

Fig. 9. Delay of voice packets improved when TFMC mechanism is used

load and the number of packets at routers according to the network congestion state, therefore it decreases congestion and queueing delays. The adaptability of the scheme to the network conditions results in a reduced packet voice delay (< 55 ms).

VI. Conclusion

We have proposed a new adaptive voice flow multiplexing scheme called TFMC (TCP-Friendly Multiplexing Control), which tries to keep the transmission protocol overhead to a minimum while maintaining a steady voice data throughput. TFMC combines RTP voice flow multiplexing and the TCP-friendly congestion control mechanism. Simulation results show that TFMC achieves performance goals of voice flows (reduced delay and delay jitter) as well as efficient network utilization, and fairness with TCP traffic. Our scheme is scalable because no changes are needed at core routers and minimal control messages are used: one feedback message per several multiplexed voice flows. In addition, no modifications of the RTP packet format are required, thus it can be easily implemented and deployed.

References

[1] H. P. Sze, S. C. Liew, J. Y. B. Lee, and C. S. Yip, *A Multiplexing Scheme for H.323 Voice-Over-IP Applications*, IEEE Journal on Selected Areas in Communications, Vol. 20, No. 7, September 2002.

[2] T. Hoshi, K. Tanigawa, and T. Takahashi, *IP Telephony Gateway and Its Media Stream Control*, Hitachi Review, Vol. 48, No. 4, 1999.

[3] ITU, *Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s*, ITU-T Recommendation G.723.1, March 1996.

[4] ITU, *C source Code and Test Vectors for Implementation Verification of the G.729 Reduced Complexity 8 kbit/s CS-ACELP Speech Coder*, ITU-T Recommendation G.729 Annex A, November 1996.

[5] ETSI, *Digital Cellular Telecommunications System Full Rate Speech Transcoding*, ETSI GSM 6.10 version 5.1.1, May 1998.

[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, Request for Comments 1889, January 1996.

[7] H. Schulzrinne, *RTP Profile for Audio and Video Conferences with Minimal Control*, Request for Comments 1890, January 1996.

[8] K. Tanigawa, T. Hoshi, and K. Tsukada, *Simple RTP Multiplexing Transfer Methods for VoIP*, Internet Draft draft-tanigawa-rtp-multiplex-01.txt, May 1999.

[9] ITU, *Packet-Based Multimedia Communications Systems*, ITU-T Recommendation H.323, November 2000.

[10] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson , R. Sparks, M. Handley and E. Schooler, *SIP: Session Initiation Protocol*, Request for Comments 3261, June 2002.
[11] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen and J. Segers, *Megaco Protocol Version 1.0*, RFC 3015, November 2000.
[12] M. Handley and V. Jacobson, *SDP: Session Description Protocol*, RFC 2327, April 1998.
[13] J. Ash, B. Goode, J. Hand, and R. Zhang, *Requirements for End-to-End VoIP Header Compression*, Internet Draft, Network Working Group, February 2003.
[14] S. Casner and V. Jacobson, *Compressing IP/UDP/RTP Headers for Low Speed Serial Links*, in Proc. IETF RFC 2508, February 1999.
[15] B. Thomspon, T. Koren, and D. Wing, *Tunneling Multiplexed Compressed RTP ('TCRTP')*, Internet Draft, February 2002.
[16] R. Pazhyannur, I. Ali, and C. Fox, *PPP multiplexing*, IETF RFC 3153, August 2001.
[17] J. Rosenberg and H. Schulzrinne, *Issues and Options for an Aggregation Service within RTP*, Internet Draft, IETF Audio/Video Transport Working Group, May 1997.
[18] J. Rosenberg, and H. Schulzrinne, *An RTP payload Format for User Multiplexing*, Internet Draft, Audio/Video Transport Working Group, November 1998.
[19] K. El-Khatib, G. Luo, G. Bochmann, and P. Feng, *Multiplexing Scheme for RTP Flows Between Access Routers*, Internet Draft, Audio/Video Transport Working Group, April 2000.
[20] J. Mahdavi and S. Floyd, *TCP-Friendly Unicast Rate-Based Flow Control*, note sent to end2end-interest mailing list, January 1997.
[21] S. Floyd, and K. Fall, *Promoting the Use of End-to-End Congestion Control in the Internet*, IEEE/ACM Transactions on Networking, August 1999.
[22] T. Turletti, S. Parisis, and J. Bolot, *Experiments with a Layered Transmission Scheme over the Internet*, Technical Report RR-3296, INRIA, France.
[23] D. Sisalem and H. Schulzrinne, *The Loss-Delay based Adjustment Algorithm: a TCP-friendly Adaptation Scheme*, in Proceedings of NOSSDAV'98, pp. 215-226, July 1998.
[24] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, *Modeling TCP Throughput: A Simple Model and its Empirical Validation*, Proc. ACM SIGCOMM 1998.
[25] M. Handley, S. Floyd, J. Padhye and J. Widmer, *TCP Friendly Rate Control (TFRC): Protocol Specification*, Request for Comments 3448, January 2003.
[26] S. Floyd, M. Handley, J. Padhye and J. Widmer, *Equation-Based Congestion Control for Unicast Applications*, in Proceedings of ACM SIGCOMM'2000, pp. 43-56, August 2000.