International Journal of Innovative Computing, Information and Control
Volume **11**, Number **6**, December **2015**
ICIC International ©2015 ISSN 1349-4198
pp. 2063–2073

ITTP-MUX: AN EFFICIENT MULTIPLEXING MECHANISM TO IMPROVE VOIP APPLICATIONS BANDWIDTH UTILIZATION

Mosleh M Abualhaj

Department of Networks and Information Security
Faculty of Information Technology
Al-Ahliyya Amman University
Amman, Jordan
m.abualhaj@ammanu.edu.jo

Received March 2015; revised August 2015

Abstract. *In recent years, the field of telecommunications began to move toward Voice over Internet Protocol (VoIP) technology. VoIP technology applications generate packets with small payload sizes to minimize packetization delay. That is, increasing the header overhead wastes the network link bandwidth. Packet multiplexing is a mechanism to improve the exploitation of network link bandwidth. Various multiplexing mechanisms are proposed to improve link bandwidth exploitation when using RTP/UDP protocols. This paper proposes an efficient multiplexing mechanism, called ITTP Multiplexing (ITTP-Mux). Unlike previous mechanisms, ITTP-Mux mechanism multiplexes VoIP packets when using ITTP protocol, not RTP/UDP. In addition, the ITTP-Mux mechanism assembles VoIP packets, which exist in the same path, in a single IP header instead of an IP header to each packet. Therefore, header overhead is lessened and network link bandwidth is saved. ITTP-Mux also adds 1-byte mini-header to each packet to distinguish the assembled packets. The proposed ITTP-Mux mechanism is simulated and compared with the traditional ITTP protocol (without multiplexing) using five factors, namely, number of calls, goodput, header overhead, bandwidth usage, and saved bandwidth. Based on all these factors, ITTP-Mux mechanism outperforms the traditional ITTP protocol. For example, the result shows that bandwidth usage is improved by up to 29.1% in the tested cases.*

Keywords: VoIP, ITTP, Packet multiplexing, Network bandwidth

1. **Introduction.** The telecommunications sector underwent a considerable revolution in the past decade. The voice telecommunications sector is changing from traditional cellular and landline systems to Voice over Internet Protocol (VoIP) [1]. The unexpected demand for VoIP has brought about a popularity explosion, which has induced most telecommunication service providers to adopt VoIP technology. Businesses, universities, and enterprises have also adopted VoIP technology and invested in its development [2]. The main reason for this unexpected demand is the ability of VoIP to allow local and overseas calls to be made anywhere in the world at rates cheaper than those from traditional cellular and landline systems [2,3]. However, numerous obstacles hinder the development of VoIP technology. Besides the potential QoS issues, VoIP technology suffers from the inefficient bandwidth exploitation of network links because of the considerable header overhead of the VoIP protocol [4-6].

Many VoIP protocols are used in the communication process, such as media transfer protocols [7,8]. Media transfer protocols, such as Real-time Transport Protocol (RTP) and Internet Telephony Transport Protocol (ITTP), are used to transfer voice data between call parties [4,9]. The 12-byte RTP typically works in conjunction with the 28-byte UDP/IP (8-byte UDP and 20-byte IP) to transfer VoIP data. However, the 40-byte

RTP/UDP/IP adds a substantial header overhead to the typical 10- to 30-byte VoIP packet payload, causing inefficient bandwidth exploitation [6,8,10]. To solve this problem, the 6-byte ITTP was designed as a dedicated transport protocol to carry VoIP data, thus replacing RTP/UDP protocols [4,9]. The typical ITTP VoIP packet consists of a 26-byte ITTP/IP header. Although ITTP lessens the header overhead the RTP/UDP protocols cause, the header overhead is still substantial compared with the 10- to 30-byte VoIP packet payload. Thus, the bandwidth links are consumed by carrying header (non-useful) data.

A considerable number of VoIP packets move over network links [6,11]. Therefore, VoIP packets consume a significant amount of network link bandwidth. However, network links, especially WAN links, are costly enough to be worthy of efficient bandwidth exploitation. Hence, efficient bandwidth exploitation is a main priority of VoIP technology [4,5]. Several mechanisms are developed to improve VoIP bandwidth exploitation [6,11]. Packet multiplexing is one of these mechanisms [6,12,13]. The principal idea of packet multiplexing is to assemble multiple payloads of packets in one header, which lessens header overhead and improves bandwidth exploitation [6,14]. This study presents a new multiplexing mechanism that assembles a number of ITTP VoIP payload packets from multiple sources existing in the same path of only one IP header. Thereby, header overhead is lessened and bandwidth exploitation is improved.

The rest of this paper is organized as follows. In Section 2, we introduce the background. In Section 3, we display related works. In Section 4, we explain the proposed multiplexing mechanism, namely, ITTP Multiplexing (ITTP-Mux). In Section 5, we show the simulation details of ITTP-Mux, as well as present and discuss the results. Finally, we conclude the paper in Section 6.

2. **Background.** In this section, we review the voice codec, which is relevant to this work.

A voice codec converts voice data from analog to digital by compressing digital voice data and converting these to small frames (VoIP packet payload) that vary between 10 and 30 bytes depending on the codec [7,10]. Table 1 shows examples of the VoIP codecs. A codec generates small frame sizes because a frame is generated only after voice signals are captured and encoded in the time period of a frame. The time period of the frame and the network delay should be within the acceptable end-to-end delay. Increasing the frame size will increase the time period of the frame, and thus increase the end-to-end delay. Accordingly, a small time period for a frame period produces a small frame size [5,6].

3. **Related Works.** Several multiplexing mechanisms were proposed to save bandwidth. This section discusses some of these mechanisms.

TABLE 1. Examples of VoIP codecs

Codec	*Frame size (ms)*	*Frame size (B)*	*Bit rate (kbps)*
G. 728	*5*	*10*	*16*
G. 726	*5*	*15*	*24*
LPC	*20*	*14*	*5.6*
G. 723.1	*30*	*24*	*6.3*
G. 729	*10*	*10*	*8*
G. 723.1	*30*	*20*	*5.3*

Hoshi et al. proposed one of the first VoIP stream multiplexing mechanisms in 1999 [15]. The proposed mechanism assembles a number of VoIP streams from multiple sources that exist in the same path in only one UDP/IP header. The multiplexing technique falls between the UDP and RTP layers and requires no extra header. The resulting multiplexed packet is composed of one UDP/IP header and multiple RTP headers with a VoIP frame. Therefore, this method lessens the header overhead that results from the 28-byte UDP/IP attached to each packet. The implementation of this mechanism showed that bandwidth employment was improved by 40%.

Another multiplexing mechanism was proposed by Subbiah et al. in 1999 [16]. Unlike previous mechanisms, this mechanism assembles a number of VoIP streams from multiple sources that exist in the same path in only one RTP/UDP/IP header. The multiplexing mechanism of this mechanism falls between the RTP layer and the voice payload. Therefore, a new extra header is required. A 2-byte mini-header is added to each audio frame to distinguish the multiplexed voice frames. The resulting multiplexed packet is composed of one RTP/UDP/IP header and multiple mini-headers with a VoIP frame. This method lessens the header overhead that results from the 40-byte RTP/UDP/IP attached to each packet. The implementation of this mechanism showed that the header overhead was lessened by 50% to 80%, depending on the VoIP frame size.

Sze et al. addressed the VoIP stream multiplexing mechanism in 2002 [5] by proposing a mechanism that combines header compression and stream multiplexing. Initially, the RTP header in each VoIP packet is replaced with a 2-byte compressed mini-header. The resulting chunks are then multiplexed into one UDP/IP header. The multiplexer and the demultiplexer create mapping tables to ensure that the compressed mini-headers are rebuilt at the demultiplexer. The resulting multiplexed packet is composed of one UDP/IP header and multiple compressed headers with a VoIP frame. This method significantly lessens the header overhead. The implementation of this mechanism showed that bandwidth employment was improved by 72%.

A mechanism that combined packet multiplexing and compression, named MuxComp, was proposed by Abu-Alhaj et al. in 2009 [3]. MuxComp mechanism proposes a framework to multiplex voice packets and then compress the resulting multiplexed packet. The mechanism consists of two separate entities: MuxCmp and DCmpDMux. First, the MuxCmp entity at the sender side combines several VoIP packets from different sources that are destined to the same destination in one packet. The MuxCmp entity then compresses the resulting multiplexed packet to lessen the overall packet size. At the receiver side, the DCmpDMux entity decompresses the received packets, demultiplexes each packet to the original packets, and dispatches these packets to their destinations. Multiplexing multiple VoIP packets lessens the header overhead and provides the opportunity to compress the packets again, which decreases overall packet size and lowers the consumed bandwidth.

In 2010, Abu-Alhaj et al. proposed the Delta-multiplexing mechanism [14], which combines packet multiplexing and payload compression. Delta-multiplexing lessens the header overhead and assembles a number of VoIP streams in one UDP/IP header. This mechanism also lessens the size of the packet payload by transferring the difference of the successive packet payloads. A 2-byte mini-header is needed to return the packet payloads to its size. The 2-byte mini-header falls between the UDP layer and the RTP layer. The resulting multiplexed packet is composed of one UDP/IP header and multiple mini-headers with an RTP header and a compressed voice frame. The Delta-multiplexing mechanism greatly lessens the header overhead. The implementation of this mechanism showed a bandwidth saving between 68% and 72%, depending on the VoIP frame size.

The aforementioned VoIP stream multiplexing mechanisms significantly improved VoIP bandwidth exploitation. These mechanisms were proposed to save bandwidth when using

RTP/UDP protocols. However, none of the multiplexing mechanisms were implemented with the ITTP protocol as a new protocol. In this paper, we propose a new multiplexing mechanism called ITTP-Mux. Unlike the aforementioned multiplexing mechanisms, ITTP-Mux is implemented over the ITTP protocol. The ITTP-Mux mechanism also adds the multiplexing layer between the ITTP and the IP layers, which multiplex the ITTP streams in a single IP header and keeps a separate ITTP header in each packet of each stream. Furthermore, ITTP-Mux adds 1-byte mini-header with stream ID to each packet to distinguish the packets in the multiplexed streams.

4. **ITTP-Mux.** This section presents the design of the proposed ITTP-Mux mechanism. The ITTP-Mux mechanism achieves competent bandwidth exploitation. The principal idea of the ITTP-Mux mechanism is to assemble several ITTP VoIP packets from different streams to the same VoIP gateway into a single IP stream. This method lessens the packet header overhead that results from the IP header attached to each packet, thus improving bandwidth exploitation. Figure 1 shows a scenario where the ITTP-Mux mechanism achieves high bandwidth exploitation.

4.1. **ITTP-Mux architecture.** The ITTP-Mux architecture consists of a Stream Multiplexer (S-Mux) located in the sender gateway and a Stream Demultiplexer (S-DMux) located in the receiver gateway. S-Mux performs stream packet multiplexing, whereas S-DMux performs stream packet demultiplexing.

4.1.1. *Stream packet multiplexing.* The S-Mux at the sender gateway performs a set of procedures to achieve packet multiplexing. It initially collects the packets received at the sender gateway and checks their destinations to assemble the payload of packets destined to the same destination gateway. S-Mux then extracts the packet payload, which consists of the ITTP header and an audio frame. Subsequently, a 1-byte header, called the mini-header, is attached to the extracted payload, which constitutes a small packet called the mini-packet. Figure 2 shows the format of the mini-packet. The resulting mini-packets are assembled in one IP header, which constitutes a multiplexed packet called mux-packet. Figure 3 shows the format of the mux-packet. Finally, the mux-packets are dispatched to their destination gateways. Figure 4 shows the S-Mux in the sender gateway. Figure 5 shows a flowchart of the internal process of the S-Mux in the sender gateway.

4.1.2. *Stream packet demultiplexing.* The S-DMux at the receiver gateway performs a set of procedures to achieve packet demultiplexing. First, it collects the mux-packets received from the sender gateway. Then it de-assembles the received mux-packets by inspecting

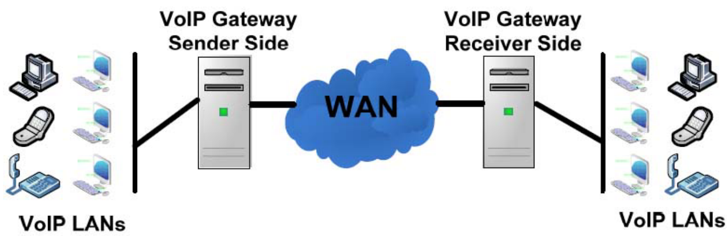

FIGURE 1. ITTP-Mux scenario

FIGURE 2. Mini-packet format

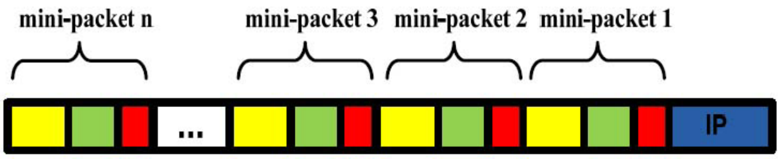

FIGURE 3. Mux-packet format

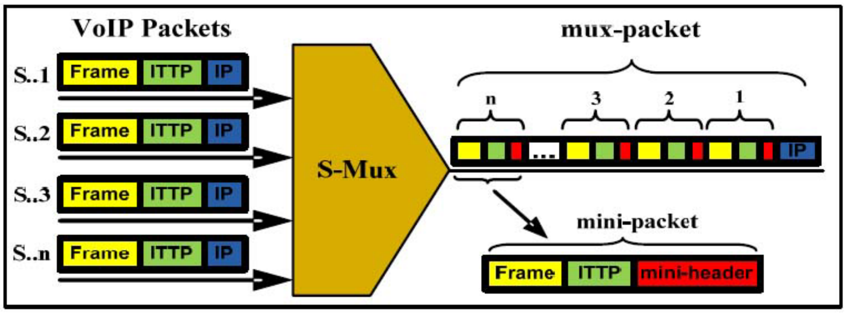

FIGURE 4. ITTP-Mux mechanism multiplexer (S-Mux)

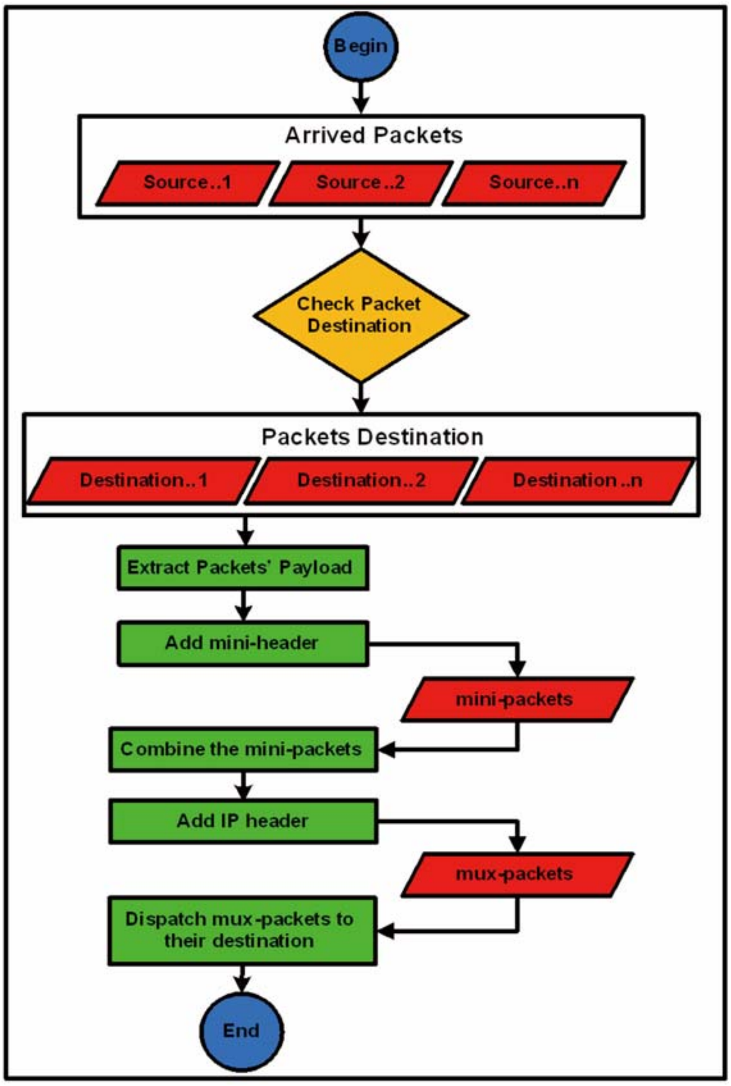

FIGURE 5. S-Mux internal process

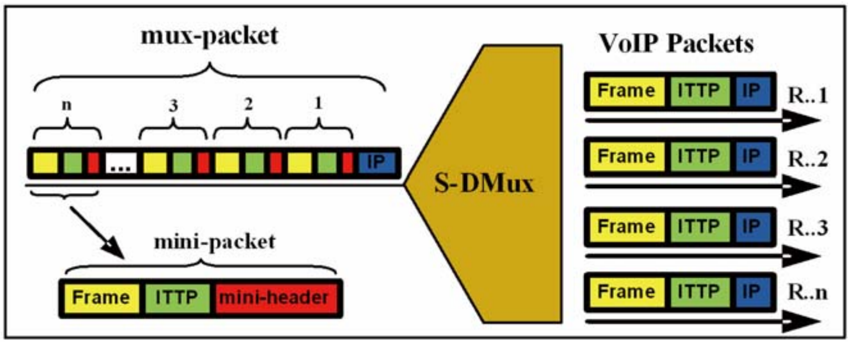

FIGURE 6. ITTP-Mux mechanism demultiplexer (S-DMux)

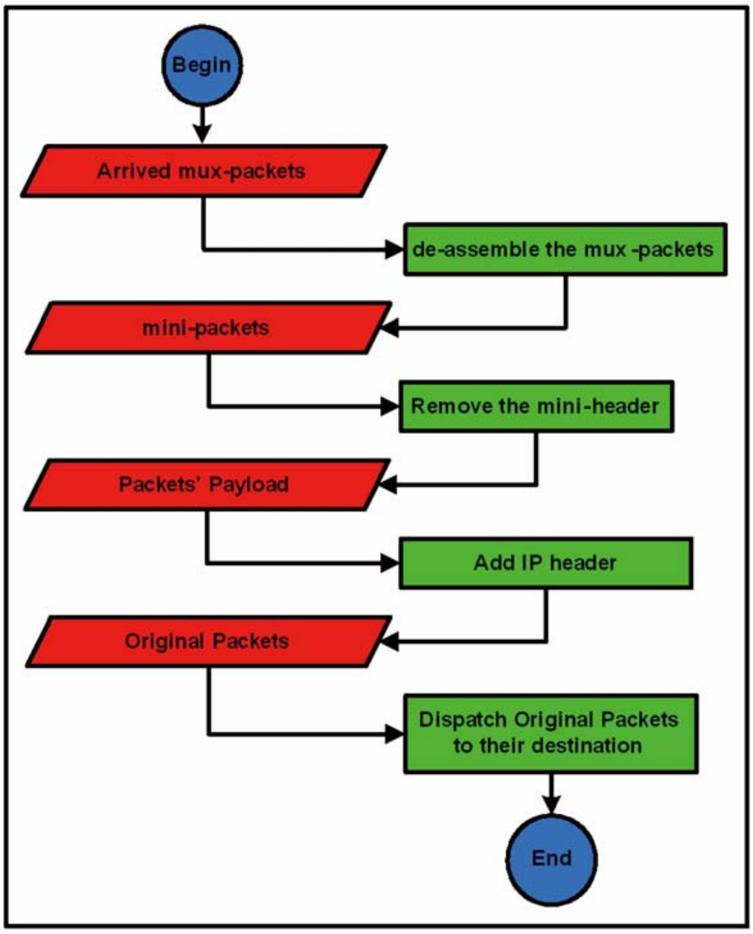

FIGURE 7. S-DMux internal process

the mini-header to recover the individual mini-packet. Afterward, the S-DMux eliminates the mini-header of each mini-packet and reconstructs the original packet by adding the IP header based on the information in the mini-header. Finally, the original packets are sent to their destinations. Figure 6 shows the S-DMux in the receiver gateway. Figure 7 shows a flowchart of the internal process of the S-DMux in the receiver gateway.

4.2. **Mini-header.** The principal idea of the proposed ITTP-Mux mechanism is to remove the IP header of each packet and assemble the packet payload destined to the same VoIP gateway into a single mux-packet of one IP header. Before assembling these payloads, a 1-byte mini-header is added to each payload instead of the IP header, which constitutes the mini-packet. The ITTP-Mux at the receiver gateway uses this mini-header to differentiate the mini-packets inside the mux-packet.

The mini-header consists of only one field called Stream ID (SID). The ITTP-Mux uses SID at the receiver gateway to retrieve the destination address of the mini-packet. The SID in a single mux-packet is unique. A mapping between the mini-packet destination address and the corresponding SID is kept in a state table at the sender and receiver gateways during call set-up.

The SID field size can vary based on the number of assembled mini-packets inside a mux-packet. A 1-byte SID can assemble 256 mini-packets inside one mux-packet, which clearly replaces the 20-byte IP header with the 1-byte mini-header, thus lessening the header overhead problem and saving bandwidth.

4.3. **Call set-up and SID selection.** Phone calls over the IP network require a call setup [17,18]. The call setup establishes a session between call ends. The call setup process requires a small adjustment to conform the multiplexing in the VoIP streams between the VoIP gateways. The adjustment in the call setup process selects a SID for each stream and maps the SID with the stream destination address. The SID selection procedure is conducted as follows.

1. After initiating the session between the two call ends, the S-Mux in the sender gateway checks for the existence of a connection with the receiver gateway:
   (a) If a connection exists with free SID, then a SID is selected and reserved for that stream.
   (b) If the two VoIP gateways have no connection or if all SIDs of the existing connections are reserved, then a new connection is established between the two gateways.
2. After selecting a SID for the call, the S-Mux keeps the SID and the callee address (IP address and port number) in a table called state table. Meanwhile, the S-Mux at the sender gateway sends the SID and the address information to the receiver gateway.
3. Once the receiver gateway receives the SID and the address, it also keeps them in a state table, which will be used when the voice data are being transmitted.

The process of SID selection must be conducted in both directions of the call. Table 2 shows the address information and the corresponding SID inside the state table.

TABLE 2. State table

VoIP Gateway Sender Side		*VoIP Gateway Receiver Side*	
SID	*Callee address (IP address:port number)*	*SID*	*Callee address (IP address:port number)*
70	*10.207.160.1:4040*	*70*	*10.207.160.1:4040*
13	*10.207.160.2:4041*	*13*	*10.207.160.2:4041*
90	*10.207.160.5:4051*	*90*	*10.207.160.5:4051*
92	*10.207.160.6:4055*	*92*	*10.207.160.6:4055*

5. **Implementation and Performance Analysis.** This section demonstrates the simulation model in which the ITTP-Mux mechanism was evaluated. It compares the bandwidth usage efficiency of the ITTP-Mux mechanism with the conventional ITTP protocol (without multiplexing). The bandwidth usage efficiency of the ITTP-Mux mechanism and the ITTP protocol was evaluated and compared based on three main factors: number of calls, goodput, and header overhead.

5.1. **Simulation model.** A simulation model was used to evaluate the efficiency of ITTP-Mux mechanism bandwidth usage and was compared with the ITTP protocol. The simulation model architecture contains two components of the ITTP-Mux mechanism: S-Mux, which is located at the sender gateway, and S-DMux, which is located at the receiver gateway. Each component uses a queue with a maximum size of 50. The S-Mux and S-DMux are connected with a WAN link, which is simulated as a first-in first-out queue. The processes of S-Mux and S-DMux are explained in Sections 4.1.1 and 4.1.2 respectively. The G.729 codec is assumed, in which the audio frame size is equal to 10 bytes.

The S-Mux multiplexes the received ITTP packets every T ms, where T is the multiplexing period. The maximum acceptable delay for a voice conversation is 150 ms. A short multiplexing period lessens delay but increases header overhead because the S-Mux assembles few mini-packets in each mux-packet. By contrast, a large multiplexing period increases delay, but lessens header overhead [5,6]. A trade-off between header overhead and delay should be considered. In this simulation, we assume the multiplexing period of 10 ms, which is similar to previous research.

The number of calls, goodput, and header overhead was investigated at various bandwidths between 100 and 500 kbps. The number of synchronous streams was increased for each link bandwidth. The start of packet dropping indicates that the link is overwhelmed. Therefore, the number of synchronous streams for each link is equal to the number of streams before packet dropping starts. Thus, the number of calls, goodput, and header overhead was calculated accordingly.

5.2. **Number of calls.** The number of synchronous calls of different bandwidths is shown in this section. Figure 8 explores the number of synchronous calls for ITTP-Mux mechanism and ITTP. The number of synchronous calls when using ITTP-Mux mechanism is greater than the number of synchronous calls when using ITTP. In addition, the difference in the number of synchronous calls increases when the available bandwidth increases.

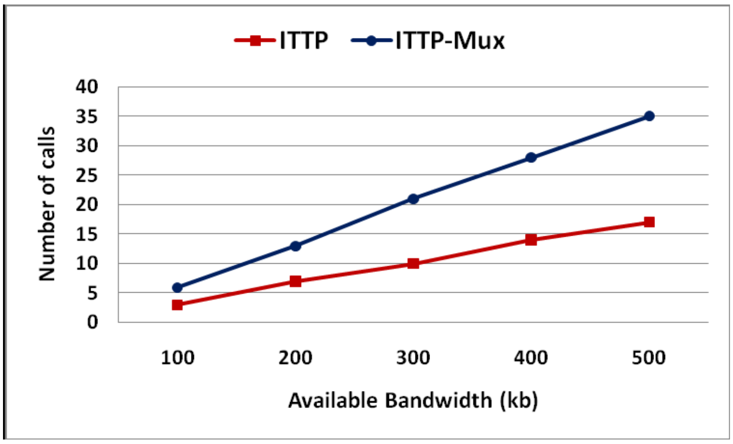

FIGURE 8. Number of calls

5.3. **Goodput.** Goodput was another factor used to evaluate the ITTP-Mux mechanism; it was compared with ITTP. Goodput represents the amount of actual data (packet payload) that the destination received. Equation (1) is used to calculate goodput:

$$Goodput = \frac{\sum \boldsymbol{R}_{Pkt} * \boldsymbol{Pkt}_{ps} * 8}{1000} \tag{1}$$

where $\boldsymbol{R}_{Pkt}$ is the received packet, and $\boldsymbol{Pkt}_{ps}$ is the received packet payload size. Figure 9 shows the goodput of the ITTP-Mux mechanism and the ITTP protocol. Better bandwidth exploitation was attained when using the ITTP-Mux mechanism than when using the ITTP protocol. Bandwidth exploitation increases with the available bandwidth because most of the link bandwidth is used to convey the voice frames when using the ITTP-Mux mechanism. By contrast, most of the link bandwidth is used to convey the protocol header when using the ITTP protocol.

5.4. **Header overhead.** In this subsection, header overhead was used to evaluate the ITTP-Mux mechanism compared with the ITTP protocol. The header overhead ratio of a specific bandwidth is the relative ratio between the sum of the header size of the packets and the sum of the entire packet size of the packets. Figure 10 shows the header overhead ratio of the ITTP-Mux mechanism compared with the ITTP protocol. The ITTP-Mux mechanism shows a considerable decrease in header overhead compared with the ITTP protocol. The header overhead ratio decreases when the available bandwidth increases because the ITTP and IP protocols add 26 bytes of header to each payload. By contrast, the ITTP-Mux mechanism only adds 1 byte of header to each payload and 20 bytes of IP header to the entire mux-packet.

5.5. **Bandwidth usage.** In this subsection, the bandwidth usage, which is opposite to the header overhead, was used to evaluate the ITTP-Mux mechanism compared with the

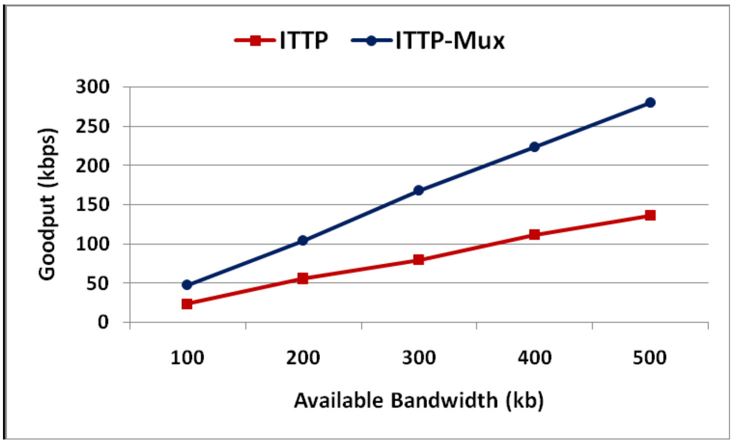

FIGURE 9. Goodput

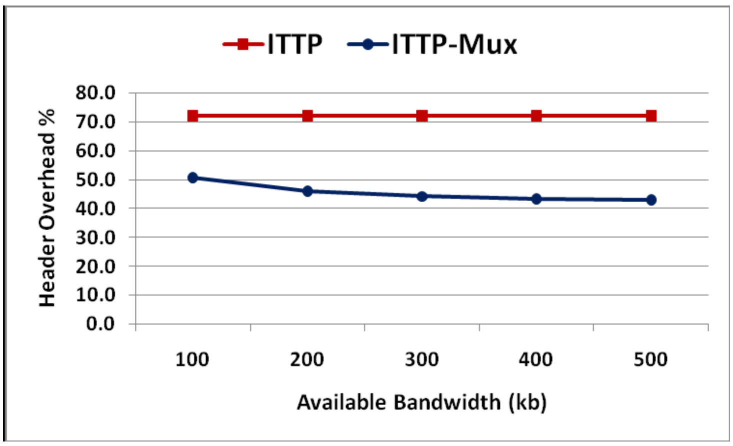

FIGURE 10. Header overhead

ITTP protocol. The bandwidth usage ratio of a specific bandwidth is the relative ratio between the sum of the payload size of the packets and the sum of the entire packet size of the packets. Figure 11 shows the bandwidth usage ratio of the ITTP-Mux mechanism compared with the ITTP protocol. The ITTP-Mux mechanism shows a considerable improvement in bandwidth usage compared with the ITTP protocol. The bandwidth usage ratio also increases when the available bandwidth increases because the ITTP-Mux mechanism only adds 1 byte of header to each payload and 20 bytes of IP header to the entire mux-packet. By contrast, the ITTP and IP protocols add 26 bytes of header to each payload.

5.6. **Saved bandwidth.** Saved bandwidth, the last factor used to evaluate the ITTP-Mux mechanism, was also compared with ITTP. This factor shows bandwidth saving when using the ITTP-Mux mechanism instead of the ITTP protocol (without multiplexing). As illustrated in Figure 12, the ITTP-Mux mechanism shows considerable bandwidth saving, which starts at more than 40% of the bandwidth when the number of calls is five and increases with the number of calls. This result is attributed to the considerable bandwidth wasted in transferring header data when using the ITTP, as shown previously.

6. **Conclusions.** VoIP emerged in the last decade as a new technology in the telecommunications industry. VoIP technology applications transmit VoIP packets in small sizes,

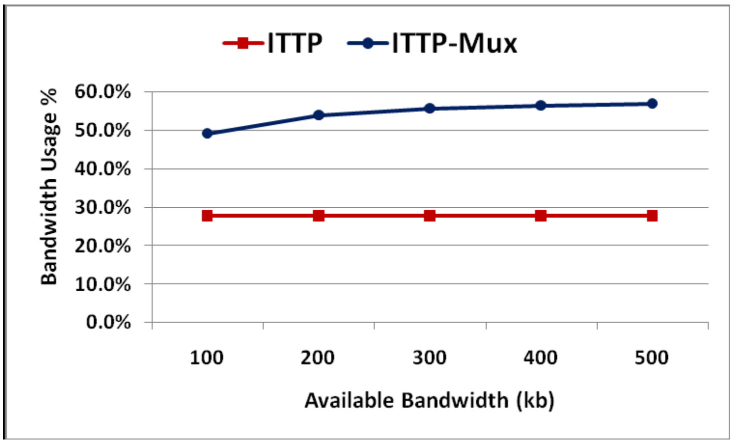

FIGURE 11. Bandwidth usage

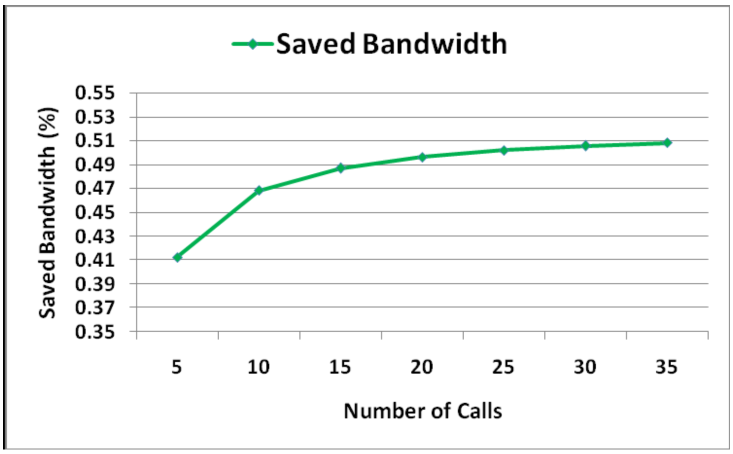

FIGURE 12. Saved bandwidth

which increase header overhead of the packets and result in inefficient bandwidth exploitation. In this paper, we proposed an efficient multiplexing mechanism called ITTP-Mux. ITTP-Mux assembles the VoIP packets from different sources that exist in the same single packet. The ITTP-performance of the mux mechanism was evaluated based on the number of calls, goodput, header overhead, bandwidth exploitation, and saved bandwidth. Based on these five parameters, the ITTP-Mux mechanism outperformed the conventional mechanism (ITTP without multiplexing). The five parameters reflect the bandwidth exploitation efficiency. In general, the simulation result showed that ITTP-Mux improves bandwidth usage by up to 29.1%.

REFERENCES

[1] X. Wang, J. Lin, Y. Sun, H. Gan and L. Yao, Applying feature extraction of speech recognition on VoIP auditing, *International Journal of Innovative Computing, Information and Control*, vol.5, no.7, pp.1851-1856, 2009.

[2] M. SK, M. M. Abu-Alhaj, O. Abouabdalla, T.-C. Wan, R. Budiarto and A. M. Manasrah, Conference gateway for heterogeneous clients: Real time switching clients and interasterisk exchange clients, *International Journal of Innovative Computing, Information and Control*, vol.7, no.1, pp.395-406, 2011.

[3] M. M. Abu-Alhaj, M. S. Kolhar, M. Halaiyqah, O. Abouabdalla and R. Sureswaran, MuxComp – A new architecture to improve VoIP bandwidth utilization, *International Conference on Future Networks*, 2009.

[4] M. M. Abu-Alhaj, M. SK, R. Sureswaran, T.-C. Wan, I. J. Mohamad and A. M. Manasrah, ITTP: A new transport protocol for VoIP applications, *International Journal of Innovative Computing, Information and Control*, vol.8, no.3(A), pp.1879-1895, 2012.

[5] H. P. Sze, S. C. Liew, J. Y. B. Lee and D. C. S. Yip, A multiplexing scheme for H.323 voice-over-IP applications, *IEEE Journal on Selected Areas in Communications*, vol.20, no.7, pp.1360-1368, 2002.

[6] U. P. Daniel, N. C. Agbanusi and K. J. Danjuma, A survey of bandwidth optimization techniques and patterns in VoIP services and applications, *International Journal of Computer Science Issues (IJCSI)*, vol.11, no.2, 2014.

[7] M. M. Abu-Alhaj, M. S. Kolhar, M. Halaiyqah, O. Abouabdalla and R. Sureswaran, Multiplexing SIP applications voice packets between SWVG gateways, *Proc. of the 2009 International Conference on Computer Engineering and Applications*, pp.237-241, 2009.

[8] M. M. Abu-Alhaj, A. Manasrah, M. Baklizi, N. Abdullah and L. V. Chandra, Transport layer protocols taxonomy from voice over IP perspective, *Advanced Computing*, vol.2, no.4, 2011.

[9] A. Kayed, M. M. Abu-Alhaj and M. Alharibat, Performance comparison of IAX and ITTP VoIP protocols, *International Journal of Academic Research*, vol.5, no.2, pp.186-193, 2013.

[10] A. Froehlich, *CCNA Voice Study Guide*, Sybex, 2010.

[11] A. Munther, R. R. Othman, M. M. Abu-Alhaj, M. Anbar, S. Nizam and I. Alalousi, P2P-MCS-S: A new technique to improve MCS conferencing system bandwidth utilization, *Journal of Convergence Information Technology*, vol.9, no.4, 2014.

[12] S. Boland, Multiplexing VoIP streams for conferencing and selective playback of audio streams, *U.S. Patent No. 7,936,705*, 2011.

[13] J. Poscher, IP multiplexing from many IP hosts, *U.S. Patent Application 13/141,459*.

[14] M. M. Abu-Alhaj, M. S. Kolhar, L. V. Chandra, O. Abouabdalla and A. M. Manasrah, Delta-multiplexing: A novel technique to improve VoIP bandwidth utilization between VoIP gateways, *IEEE the 10th International Conference on Computer and Information Technology*, 2010.

[15] T. Hoshi, K. Tanigawa and K. Tsukada, Proposal of a method of voice stream multiplexing for IP telephony systems, *Internet Workshop*, 1999.

[16] B. Subbiah, S. Sengodan and J. Rajahalme, RTP payload multiplexing between IP telephony gateways, *Global Telecommunications Conference*, 1999.

[17] M. S. Kolhar, M. M. Abu-Alhaj, O. Abouabdalla, T. C. Wan and A. M. Manasrah, Comparative evaluation and analysis of IAX and RSW, *International Journal of Computer Science and Information Security (IJCSIS)*, vol.6, no.3, 2009.

[18] A. N. Jaber et al., Session initiation protocol security: A brief review, *Journal of Computer Science*, vol.8, no.3, pp.348-357, 2012.