Multiplexing VoIP Packets over Wireless Mesh Networks: A Survey

Mosleh M. Abualhaj[1], Manjur Kolhar[2] and Kefaya Qaddoum[1], Ahmad Adel Abu-Shareha[3]
[1] Faculty of Information Technology, Al-Ahliyya Amman University
Amman, Jordan
[e-mail:{m.abualhaj, k.qaddoum}@ammanu.edu.jo]
[2] Faculty of Arts and Science, Prince Sattam Bin Abdulaziz University
Wadi Ad Dwaser, Kingdom of Saudi Arabia
[e-mail: m.kolhar@psau.edu.sa]
[3] Faculty of Information Technology, Middle East University
Amman, Jordan
[e-mail: aabuahareha@meu.edu.jo]
*Corresponding author: Mosleh M Abualhaj

Received January 23, 2016; revised April 26, 2016; revised June 8, 2016; accepted July 1, 2016; published August 31, 2016

Abstract

Wireless mesh networks (WMNs) have been increasingly applied in private and public networks during the last decade. In a different context, voice over IP (VoIP) has emerged as a new technology for making voice calls around the world over IP networks and is replacing traditional telecommunication systems. The popularity of the two technologies motivated the deployment of VoIP over WMNs. However, VoIP over WMNs suffers from inefficient bandwidth utilization because of two reasons: i) attaching 40-byte RTP/UDP/IP header to a small VoIP payload (e.g., 10 bytes) and ii) 841 µs delay overhead of each packet in WMNs. Among several solutions, VoIP packet multiplexing is the most prominent one. This technique combines several VoIP packets in one header. In this study, we will survey all the VoIP multiplexing methods over WMNs. This study provides a clear understanding of the VoIP bandwidth utilization problem over WMNs, discusses the general approaches in which packet multiplexing methods could be performed, provides a detailed study of present multiplexing techniques, shows the aspects that hinder the VoIP multiplexing methods, discusses the factors affected by VoIP multiplexing schemes, shows the merits and demerits of different multiplexing approaches, provides guidelines for designing a new improved multiplexing technique, and provides directions for future research. This study contributes by providing guidance for designing a suitable and robust method to multiplex VoIP packets over WMNs.

Keywords: VoIP, VoIP Packet Multiplexing, Wireless Mesh Network, Bandwidth Utilization.

1. Introduction

Voice over IP (VoIP) has grown tremendously in the last few years. According to [1], the amount of VoIP traffic over the Internet in 2015 is around 156 petabytes per month. In 2013, 214 billion minutes of VoIP calls were run on the Skype application alone [2]. The main driver behind this tremendous adoption of VoIP is that VoIP calls are low cost and are sometimes free. A second driver is the ability to integrate other services with voice such as voicemail to email transcription, text chatting, video, and …etc. Another vital driver is that VoIP efficiently improves bandwidth utilization when compared to the traditional PSTN (public switched telephone network) system. Whereas, the 64-Kb/s channel in a PSTN can accommodate many VoIP calls, which require less than 10 Kb/s [3], [4].

Wireless local area network (WLAN), specifically IEEE 802.11 standard, has been increasingly applied in various networks, from business networks to home networks [5]. To install an IEEE 802.11 network, every access point (AP) needs to be connected to another network through a wired link. This wired connection imposes extra installation complexity and high installation costs, especially in environments that need several APs. To handle this issue, IEEE 802.11s [a.k.a wireless mesh networks (WMNs)] emerged as an extension to IEEE 802.11. Instead of using several wired APs as in an IEEE 802.11 network, WMNs require only one AP to connect to a wired link; the wireless mesh routers are used to extend the coverage area [6], [7], [8]. Seamless installation, scalability, turnaround device failure, and low troubleshooting costs have promoted the wide utilization of WMNs, thereby replacing the typical IEEE 802.11 standard [9]. With this distribution, the deployment of VoIP over WMN is an attractive solution that has gained considerable attention in all sectors [10].

However, VoIP over WMNs is facing two main problems. First, the QoS of voice is degrading over WMN due to packet loss, which is the number of lost packets before they reached the destination, delay, which is the time consumed to transmit the packet between a caller and a callee, and jitter, which is the variation of delay. The increasing of these three factors (packet loss, delay and jitter) in WMNs is due to i) the high susceptible to channel interference, ii) the limitation of the avilable bandwidth, iii) the increasing in channel contention because of increasing the typical transmission-control protocol (TCP) traffic, such as Web and e-mail traffic, iv) the access delay which is the time between arriving the VoIP packet to the AP until it is either successfully transmitted over the WMN or dropped because it has waits long to be transmitted, and v) using distributed coordination function (DCF) channel, which does not consider the sensitivity of VoIP to delay, since point coordination function (PCF) cannot be used [8], [11], [12], [13]. Apart from QoS, a key problem is the inefficient bandwidth utilization resulting from large header overhead (as discussed in section 2.2), thereby limiting the WMN transmission capacity (number of simultaneous calls) [14], [15], [16]. Many efforts have been made to solve this issue. One of these efforts is VoIP packet multiplexing, in which multiple VoIP packets are combined in one header [15], [17], [18]. This work discusses the present packet multiplexing methods of VoIP packets over WMNs. The work provides a clear understanding of the inefficient bandwidth utilization of VoIP over WMN and the impact of packet multiplexing on improve bandwidth utilization and how it reflects to VoIP QoS. In addition, this work provides some general directions for designing a robust multiplexing technique. To the best of our knowledge, this survey is the first one that focuses on VoIP multiplexing methods over WMNs.

The rest of this paper is organized as follows: Section 2 briefly describes WMNs, VoIP, and multiplexing VoIP packets. Section 3 analyzes the present multiplexing methods of VoIP packets over WMNs. Section 4 discusses the present multiplexing methods in general. Section 5 provides guidelines for designing a multiplexing method and introduces suggestions for future research. Finally, Section 6 elaborates the conclusions.

2. Background

This section highlights the topics related to this work in order to provide a better understanding to the reader. These topics include WMNs, VoIP over WMNs, and VoIP packet multiplexing.

2.1 WMNs

In the last decade, WMN has become a new multi-hop wireless standard that extended from the IEEE 802.11 standard. In WMN, the two IEEE 802.11 devices, namely, AP and client, are turned into mesh APs (MAPs) and mesh clients, respectively. In addition, WMN introduces a new device called mesh points (MPs). A mesh client can be any end device with a wireless network interface card, such as laptops and mobile phones. MAP is a device connected to the wired network in one side and broadcasts the data wirelessly on the other side. An MP can be viewed as a WMN router and is used along with MAP to extend the range of the wireless network [19], [20]. **Fig. 1** shows the general architecture of WMN.

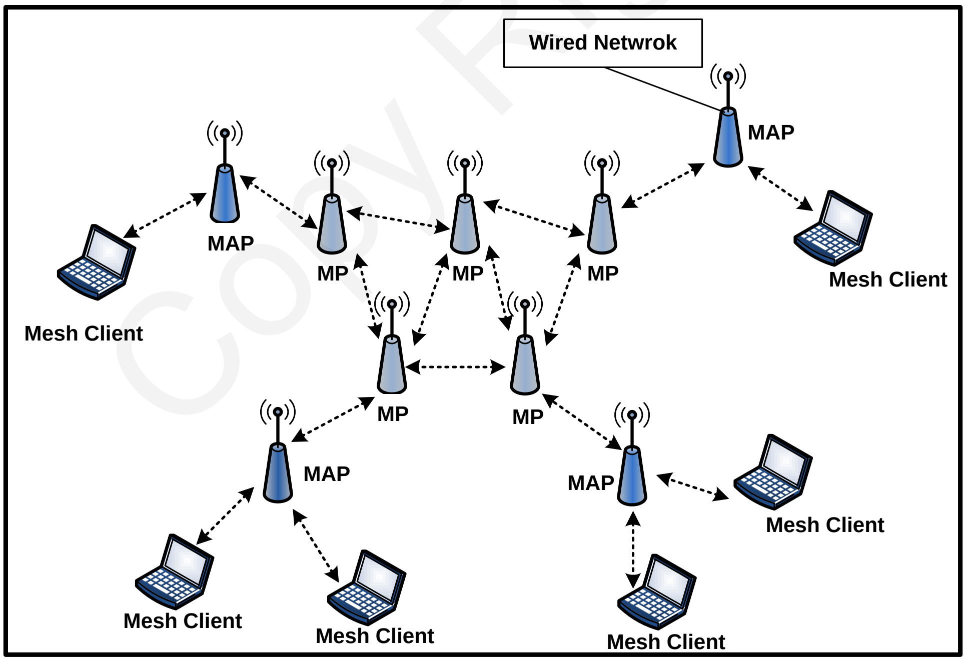

Fig. 1. Architecture of WMN

With regard to the WMN frame format, the frame header of the original IEEE 802.11 standard was extended to include new control fields for mesh technology. These new fields are embedded in the IEEE 802.11 frame body, as shown in **Fig. 2**. The total size of the mesh header is 40 bytes for IEEE 802.11 plus mesh control fields of up to 24 bytes, thereby resulting in 64 bytes of mesh header. The header overhead is considered to be large especially when compared with small packet applications, such as VoIP as we will see in the next subsection [20], [21].

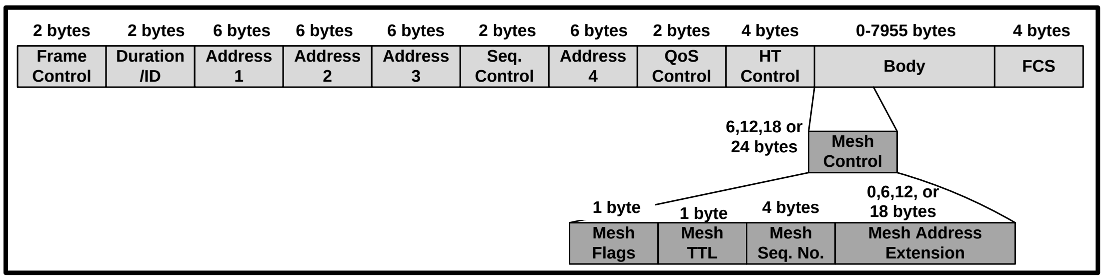

Fig. 2. WMN frame format

2.2 VoIP over WMNs

VoIP is a technology that allows voice calls through an IP network. VoIP quality is highly sensitive to delay and packet loss (high delay and packet loss result in poor VoIP quality). To travel over an IP network, the voice is captured and digitized by hardware or software called a codec. The codec then compresses and converts the digitized data to small size frames (voice packet payload). Each frame is produced in a specific time period. Enlarging the frame size will increase the time period. Such enlargement will increase the end-to-end delay, thereby resulting in poor VoIP quality. Accordingly, the frame is limited to specific small sizes. The frame size and time period vary depending on the type of codec. **Table 1** shows some of the voice codecs. For example, the G.729 codec produces a 20-byte packet payload (2 frames) in a 20 ms duration. The produced frame is then attached to a 40-byte RTP/UDP/IP header and passed down to the lower layer [data link layer (DLL)], which is attached to another header depending on the technology [18], [22], [23], [24].

Table 1. Voice codecs

Codec	Frame size	Frame size
LPC	20 ms	14 B
G.726	5 ms	15 B
G.723.1	30 ms	20 B
G.728	5 ms	10 B
G.729	10 ms	10 B

In the case of WMN technology, in addition to the 40-byte RTP/UDP/IP header, a header of up to 64-bytes (as shown in the previous section) is attached to the voice frame, thereby resulting in a 104-byte header. If an 11 Mb/s network is considered, then the transmission time of 104 bytes voice packet is 104*8/11= 76 µs. Each packet should wait 50 µs distributed interframe space (DIFS) and typically 310 µs backoff (BO) time to make sure that the channel is idle before transmitting the packet. In addition, each packet is followed by 10 µs short inter-frame space (SIFS) and 11 µs acknowledgment (ACK) to process a received frame and ACK the frame, respectively. Further, 192 µs physical layer header [144 µs preamble and 48 µs physical layer convergence protocol (PLCP)] is attached to each packet and ACK. Accordingly, the physical layer header of the data, ACK, physical layer header of ACK, backoff (BO) time, SIFS, and DIFS add around 765 µs overhead to the 76 µs mesh header, thereby resulting in around 841 µs overhead of each packet. Meanwhile, the transmission time of a 20-byte G.729 payload in the 11 Mb/s network is only 20*8/11=14 µs. Therefore, the voice data consume less than 2% of the transmission time [20], [21], [25], [26], [27]. Accordingly, a large header overhead is added to each packet, thereby wasting bandwidth resources. **Fig. 3** shows the overhead of sending one VoIP packet assuming a 20-byte G.729. VoIP packet multiplexing is one of the techniques for alleviating the header overhead problem [15], [17], [18]. This technique will be discussed in the next subsection.

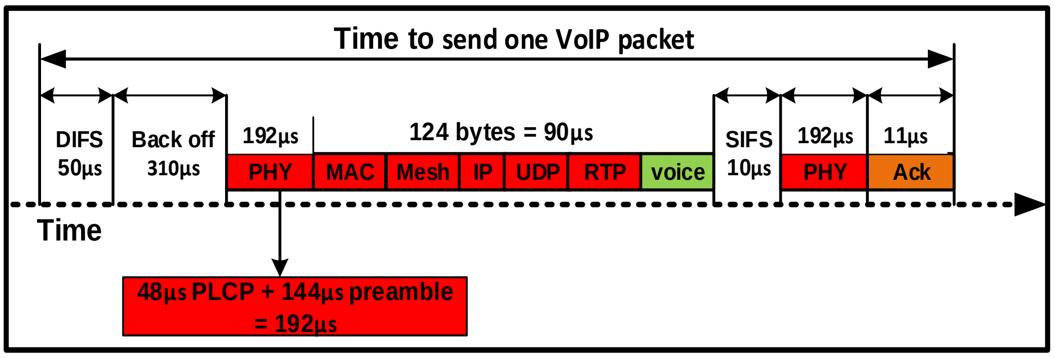

Fig. 3. Time to send one VoIP packet

2.3 Packet Multiplexing

The basic philosophy of packet multiplexing is to combine several VoIP packets in one header. The multiplexing entity at the sender side checks the packet destination and then combines the packets that travel in the same route in one header, thereby producing one multiplexed packet (***mux-pkt***). Upon receiving the ***mux-pkt***, the de-multiplexing entity at the receiver side separates and restores the combined packets. **Fig. 4** depicts the general packet multiplexing/de-multiplexing process. Packet multiplexing is one of the main techniques for handling the large overhead problem caused by carrying a VoIP packet over WMNs. **Fig. 5** demonstrates the multiplexing of three packets into one packet. The top part of the figure (**Fig. 5**) shows the time needed to transmit three packets separately. As discussed in section 2.2, 841 µs overhead is required to transmit each packet, with a total of 2523 µs (841 * 3) to transmit the three packets. The bottom part of the figure (**Fig. 5**) shows the time needed to transmit the mux-pkt that resulting from multiplexing the three packets. The ***mux-pkt*** requires only 841 µs, instead of 841 µs for each of the three separate packets. Therefore, a considerable header overhead reduction can be achieved using packet multiplexing methods. Clearly, multiplexing more packets in a ***mux-pkt*** (larger ***mux-pkt*** size) achieves better overhead reduction and provides higher bandwidth utilization. Packet multiplexing can occur at the DLL, network, transport, or even in the application layer. The high layer causes high bandwidth utilization because the header overhead of the multiplexed packets will be reduced [8], [15], [28].

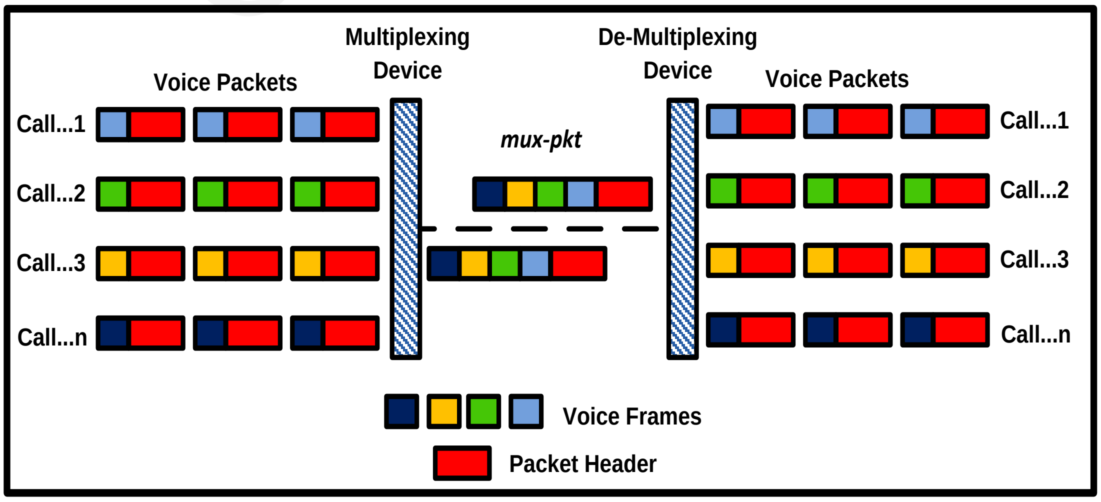

Fig. 4. Multiplexing/de-multiplexing process

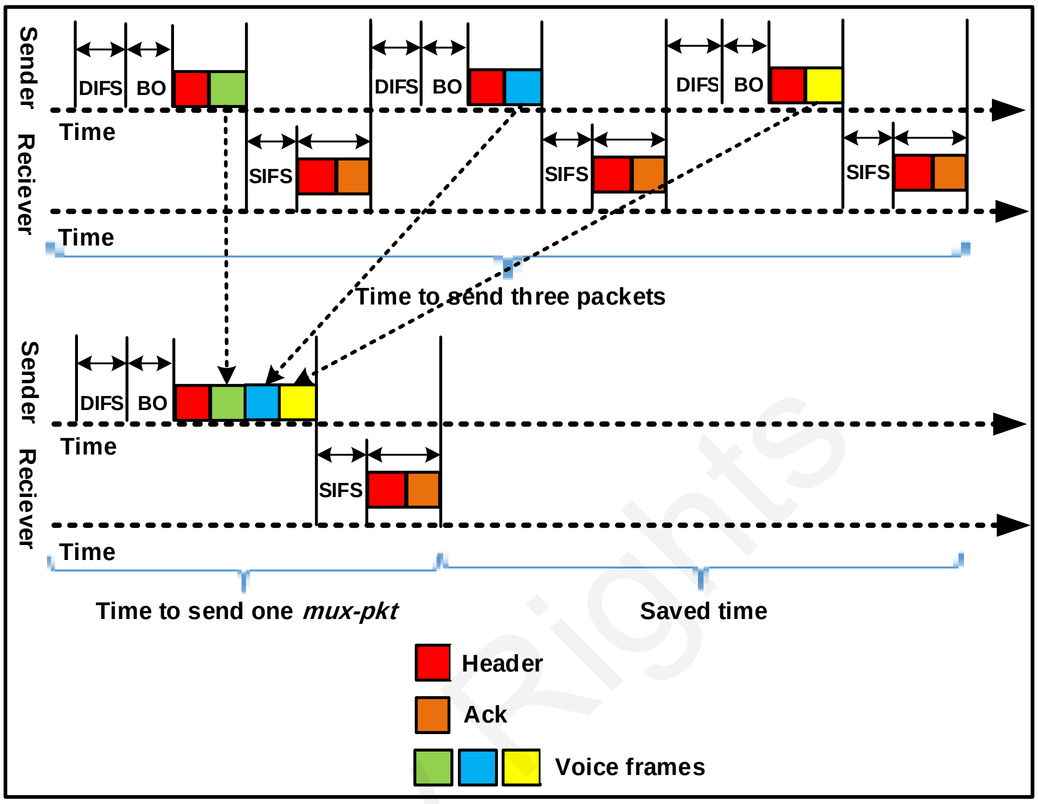

Fig. 5. Reducing header overhead by packet multiplexing

3. VoIP Multiplexing Methods Over WMNs

This section discusses the current multiplexing methods of WMNs. First, the general multiplexing approaches over WMNs are introduced. Then, the multiplexing methods are divided into two groups: adaptive and non-adaptive multiplexing methods. Finally, a brief summary of all the multiplexing methods is provided.

3.1 Multiplexing Approaches

Different general approaches of packet multiplexing methods over WMNs are available. This subsection discusses these approaches, these are: hop-to-hop, end-to-end, adaptive and non-adaptive multiplexing.

3.1.1 Hop-to-hop multiplexing

This approach performs packet multiplexing/de-multiplexing and enforces multiplexing delay at all hops throughout the route between source and destination. The end-to-end delay increases because of the multiplexing/de-multiplexing process at each hop. However, hop-to-hop approach achieves high bandwidth utilization because the possibility of multiplexing more packets increases with the number of hops. In addition, the route between the two hops limits the maximum size of ***mux-pkt***. In particular, a high-quality route carries large ***mux-pkt***, and a low-quality route carries small ***mux-pkt*,** thereby improving the bandwidth utilization of multiplexing methods [14], [29].

3.1.2 End-to-end multiplexing

This approach performs packet multiplexing at the source hop and packet de-multiplexing at the destination hop only. Therefore, the end-to-end delay that results from multiplexing/de-multiplexing process is less than that in the hop-to-hop approach. However, this approach achieves lower bandwidth utilization than the previous approach does because packet multiplexing occurs at the source hop only. In addition, the ***mux-pkt*** size is limited by the weakest link; hence, the bandwidth utilization of multiplexing methods is degraded [14], [29].

3.1.3 Adaptive multiplexing

In this approach the ***mux-pkt*** size is controlled and changed adaptively based on the link quality parameters, including; load, end-to-end delay, bit error rate, and congestion. Whereas, in poor quality links large ***mux-pkt*** is more susceptibility to noise, interference, distortion or bit synchronization errors, which increases the bit error rate. Therefore, increases the packet size increases its possibility to get damaged and, thus, dropping. On the other hand, in high quality links large ***mux-pkt*** reduces the header overhead and improves bandwidth utilization. Accordingly, this approach bounded the ***mux-pkt*** size by link quality parameters [7], [10], [15].

3.1.4 Non-adaptive multiplexing

In this approach, ***mux-pkt*** size is bounded by certain thresholds, such as time period, number of packets, and size. The actual link quality status is ignored, and, thus, the ***mux-pkt*** size might not suit the link quality; hence, the performance of the multiplexing method might be degraded [7], [15]. In all approaches, the ***mux-pkt*** size should be less than the maximum transmission unit (MTU) size, which is 2300 bytes in WMNs [30].

3.2 Present Multiplexing Methods Over WMNs

This section introduces the present multiplexing methods over WMNs. The methods are divided into two main groups: adaptive and non-adaptive multiplexing methods. As mentioned above, the main difference between the adaptive multiplexing and non-adaptive multiplexing is that the ***mux-pkt*** size in the adaptive multiplexing is changing adaptively based on the link quality, while in the non-adaptive multiplexing is bounded by certain thresholds. However, the general multiplexing process of both adaptive multiplexing and non-adaptive multiplexing is similar. **Fig. 6** shows the general process of a multiplexing method. In most papers, other contributions in addition to packet multiplexing were provided, such as header compression, mobility, and routing. In this work, we will mention these contributions but will not discuss them. Additional details on these contributions can be found in the original papers, given that this work highlights and discusses only packet multiplexing methods.

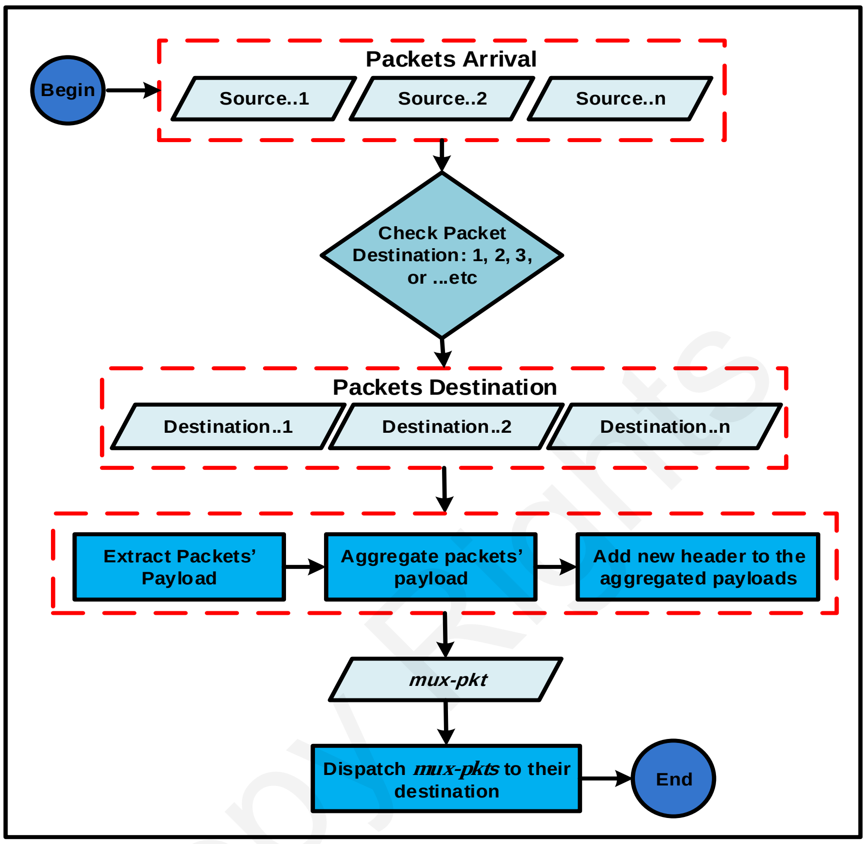
Begin
Packets Arrival
Source..1
Source..2
Source..n
Check Packet Destination: 1, 2, 3, or ...etc
Packets Destination
Destination..1
Destination..2
Destination..n
Extract Packets' Payload
Aggregate packets' payload
Add new header to the aggregated payloads
mux-pkt
Dispatch mux-pkts to their destination
End

Fig. 6. The general multiplexing process

3.2.1 Non-adaptive Multiplexing Methods

The studies in [31], [32] proposed a hop-by-hop multiplexing mechanism that works at the DLL layer. Three configurable parameters are used to control the packet size and the multiplexing process: **$SIZE_{min}$**, which is the minimum size of the ***mux-pkt***; **$SIZE_{max}$**, which is the maximum size of the ***mux-pkt***; and **MAX_{delay}**, which is the maximum waiting time to multiplex the packets. The multiplexing process works as follows: If available packet size > **$SIZE_{min}$**, then the packets are multiplexed while ***mux-pkt*** size < **$SIZE_{max}$**. If waiting time > **MAX_{delay}**, then the packets are multiplexed while ***mux-pkt*** size < **$SIZE_{max}$**. Accordingly, the ***mux-pkt*** is produced by attaching a 20-byte IP header to the ***mux-pkts***. At least two packets should be available to accomplish multiplexing. Finally, the ***mux-pkt*** is transmitted to its destination. The de-multiplexer uses the information in the IP header to find if this packet is ***mux-pkt*** and to extract the multiplexed packets within the ***mux-pkt***. As recommended by these works, **$SIZE_{min}$** and **MAX_{delay}** should trade off bandwidth utilization and delay. For example, increasing **MAX_{delay}** to multiplex more packets increases the end-to-end delay. By contrast, decreasing **MAX_{delay}** to reduce the end-to-end delay decreases the ***mux-pkt*** size, thereby reducing the bandwidth utilization. The result for the tested cases showed that the bandwidth utilization was improved to more than four times that in the traditional situation (without multiplexing). The QoS (delay, jitter, and packet loss) is acceptable when the number of connections is around 80 with “no multiplexing” scenario and 354 with “multiplexing” scenario. If the number of connections exceeds these connections, then the QoS is degraded. In terms of mean opinion score (MOS), the “no multiplexing” scenario shows better MOS because this scenario imposes less delay than the “multiplexing” scenario does.

In [33], Ganguly, Samrat et al. proposed a hop-by-hop multiplexing method that multiplexes multiple packets from different flows that travel in the same route in one ***mux-pkt***. First, the network is monitored to determine the hop(s) at which the flows converge (multiple flows pass through the same hop simultaneously). If the interference of that hop is within a certain level, then the hop is chosen as the multiplexer. Otherwise, another hop with multiple converged flows and acceptable interference is chosen as the multiplexer. Afterwards, the network is monitored to find the hop where the flows are diverging. This hop is chosen as the de-multiplexer. After determining the multiplexer and de-multiplexer, the packets are combined in a single ***mux-pkt*** at the DLL layer up to a certain size. Then, the aggregated packet is transmitted through a route determined based on the shortest-hop routing method. Upon reaching the de-multiplexer, packets are de-multiplexed and further transmitted to their destination. In addition to multiplexing, this invention compresses the 40-byte RTP/UDP/IP header to 26 bytes, thereby improving the bandwidth utilization.

Kekre, H. B. et al. [26] combined packet multiplexing and multicasting (**M-M**). The primary idea of **M-M** method is to compress the RTP/UDP/IP header of the VoIP packet and attach a mini-header with a unique ID to distinguish the multiplexed packets at the de-multiplexer. Then, several packets are multiplexed in one large packet at the network layer. Finally, the ***mux-pkt*** is multicast by a single transmission to all stations in a multicast group. The multiplexer multiplexes the packets for certain fixed period (T). The value of T should be chosen to compromise between bandwidth efficiency and end-to-end delay.

The authors mentioned that the packets can be unintentionally captured and read by a certain node by multicasting the packets to all nodes. To solve this issue, the authors proposed the encryption of the voice packets. Although encrypted, the packet is still more susceptible to being hacked than in the other methods without multicasting. In addition, the encryption/decryption process adds extra delay and degrades the overall quality. Multicasting the packets to all nodes can also cause large processing overhead on the nodes with a data not intended to them.

Khyati Marwah et al. [34] improved the e-model of the International Telecommunication Union (ITU) to measure the effect of VoIP packet multiplexing on VoIP quality over WMNs. They extended the rating factor (R-factor) equation by including the multiplexing delay A_d parameter. The proposed e-model was implemented over the end-to-end multiplexing method, and several packets were multiplexed at the MAC layer. In the evaluation test, the number of multiplexed packets in the ***mux-pkt*** increased gradually. Results showed that increasing the number of multiplexed packets increases the multiplexing delay, thereby increasing the end-to-end delay and influencing the R-factor. The results also showed that, if the number of multiplexed packets increases, then the MOS decreases. Increasing delay and decreasing the MOS have negative effects on the QoS of VoIP. Accordingly, the packet multiplexing is acceptable up to a certain threshold to avoid the low QoS of VoIP. However, the proposed e-model extension is implemented only in fixed ***mux-pkt*** size. Therefore, the authors fail to demonstrate packet multiplexing by including adaptive packet multiplexing. In addition, the effect of other factors on VoIP QoS, such as packet loss and jitter, were not explored.

In summary, this subsection discusses the non-adaptive multiplexing methods. The work in [34] only measures the effect of packet multiplexing on VoIP quality over WMNs. In [26], although the work proposes to bound the ***mux-pkt*** size with T time period, no experiments are performed to evaluate the proposed method. In addition, multicasting the packets to all stations causes unneccesary overhead on the network and reduces its perfomance. The works in [31] and [32] bound the ***mux-pkt*** size with three parameters: SIZEmin, SIZEmax, and MAXdelay. Accordingly, the ***mux-pkt*** size can be selected properly. However, these works perform the multiplexing/de-multiplexing process at each hop; hence, the end-to-end delay increases, thereby degrading the VoIP quality. Notably, the bandwidth utilization improves. The work in [33] proposes a noticeable method that performs the multiplexing/de-multiplexing process only at specific hops where packets converge/diverge. As a result, the delay that results from multiplexing/de-multiplexing process at each hop is reduced, and high bandwidth utilization is maintained. In addition, the multiplexing/de-multiplexing hops are chosen to be within a specific interference level to ensure an improved multiplexing/de-multiplexing process. Therefore, this method can be selected as the best among the non-adaptive multiplexing methods. However, the non-adaptive multiplexing methods usually use certain fixed values to determine the ***mux-pkt*** size, thereby degrading the performance of the multiplexing method. In other words, the aforementioned network conditions (interference and load) are changing continuously, especially in wireless networks. Therefore, when the network conditions are good (low interference and low load) the ***mux-pkt*** size may increase, thus improving bandwidth utilization. By contrast, when the network conditions are poor (high interference and high load) the ***mux-pkt*** size may decrease, thereby reducing the delay caused by the multiplexing process time and the packet loss induced by the large ***mux-pkt*** size. Consequently, the VoIP conversation quality is improved. Therefore, the ***mux-pkt*** size in a packet multiplexing method should change adaptively according to the network conditions. Such feature is incorporated in the adaptive multiplexing methods that will be discussed in the following subsection [7], [10], [15].

3.2.2 Adaptive Multiplexing Methods

Niculescu, Dragos et al. [35] proposed a multiplexing method that multiplexes VoIP packets at the network layer. The multiplexing occurs at every hop from source to destination unless it reaches the MTU. However, this method is not solely a hop-to-hop multiplexing method because it does not force a multiplexing delay at each hop throughout the path. This method forces a variable multiplexing delay (based on the link status) on the ingress hop only and uses the natural queuing delay to multiplex at the intermediate hops. Therefore, similar to end-to-end multiplexing methods, the proposed method reduces the end-to-end delay that results from hop-by-hop multiplexing. Moreover, similar to hop-by-hop multiplexing methods, this method improves bandwidth utilization by performing multiplexing at each hop. The detailed multiplexing algorithm is shown in **Fig. 7**. In addition to packet multiplexing, a header compression and routing algorithm for selecting the best route was proposed in this work. The result for the tested cases showed that the bandwidth utilization improved slightly compared with that in the traditional situation. The method also does not impose any additional delay because it uses “the wait for the MAC availability” for packet aggregation.

```
P:  Packet being queued at a node.
P': Packet with the same next hop as P.
A:  Multiplexed  packet being appeared.
miniPackets: number of packets from the
same flow that have multiplexed at the ingress
(corresponds to the delay budget available for
the flow);
MTU: maximum transmission unit= number of
voice packets that can be fit in 1500 bytes.

Find queue of P;
1: if size (queue) > miniPackets
       add all packets from flow (P);
       if size (A) < MTU
          find a queue with the same dest
          go to 1;
       else
          send  A directly to destination;
     else
       if size (MTU)
          do find the flows (P' >=) miniPackets
          and add miniPackets from them
          while size (A) > MTU
       else
           send A to the next hop
```

Fig. 7. Multiplexing algorithm

Another method [36] that combines the features of hop-by-hop and end-to-end multiplexing methods was proposed by Kim, Kyungtae, and Sangjin Hong. Their proposed method forces delay only at the ingress hop and takes advantage of the queuing delay at the intermediate hops. The multiplexing method is as follows: First, the packets from the same flow are multiplexed together at the ingress hop (or edge multiplexer) in a ***mux-pkt***, within a variable period of time for each flow. Then, the resultant ***mux-pkt***s are multiplexed together at the DLL layer at the edge multiplexer. Afterwards, the backhaul multiplexer, which works between hops only, multiplexes the resultant packets in one large packet. If the size of this packet reaches a specific size, then it is transfered directly to the destination to save resources; otherwise, it is transfered to the next hop for further multiplexing. A header compression algorithm was also proposed in this work. However, the proposed header compression is different from that in [35]. The result for the tested cases showed that the bandwidth utilization improved by nearly four times that in the traditional situation. Furthermore, the method does not impose any additional delay owing to its use of "the wait for the MAC availability" for packet aggregation.

Kim, Kyungtae et al. [38] proposed to multiplex multiple VoIP frames at the DLL layer. The ***mux-pkt*** size changes adaptively based on two thresholds: time and size thresholds. An algorithm was proposed to show where the delay should occur and the duration of the delay in each hop. Therefore, a better result can be obtained by this approach than when the typical hop-to-hop approach is used. If one of the two thresholds is reached, then the ***mux-pkt*** is transmitted to its destination. The detailed multiplexing algorithm is shown in **Fig. 8**. The result for the tested cases showed that the bandwidth utilization improved by more than four times that in the traditional situation. The QoS (jitter and packet loss) improved slightly compared with that in the traditional method.

```
minPackets: number of packets to be multiplexed.
maxTime: maximum Waiting time to multiplex.
P: first packet queued at node.
P': number of packets with the same destination as P.
P": number of packets with the same next hop as P.
A: multiplexed packet,
MTU: maximum transmission unit.

{Ingress w/forced delay and intermediate nodes using natural MAC}
repeat
  if P nand P' > minPackets then
     while P' and A < MTU do
        multiplex all packet from P';
     end while
     if A > MTU then
        send A to destination directly and reset timer (maxTime)
        continue;
     end if
     while P" and A < MTU do
        multiplex all packet from P";
       end while
   else if P" and time (maxTime) is expired then
       while A < MTU do
          multiplex all packet from P";
          send A to destination directly and reset timer (maxTime)
     end while
   end if
until VoIP packets
```

Fig. 8. Multiplexing algorithm

The same multiplexing method in [37] was exploited by Ganguly, Samrat et al. in [38] with minor modification. Specifically, multiple VoIP packets were multiplexed at the network layer instead of the DLL layer. In addition to packet multiplexing, a header compression technique was proposed, which increases the number of running calls and improves bandwidth utilization. This work provides two other contributions unrelated to bandwidth utilization. First, their proposed method maintains the VoIP quality by sending the packets through the route with less delay and packet loss. Second, a method that supports mobility is used, in which the call is kept stable (without disruption) when moving to other APs while the call is ongoing. In [9], the same multiplexing method and header compression were combined together and proposed as a patent. The result for the tested cases showed that the bandwidth utilization improved by two to three times that in the traditional situation. The QoS (jitter and packet loss) is improved compared with that in the traditional method.

The work in [39] implemented a hop-to-hop multiplexing method that adaptively multiplexes the packets at the DLL layer based on the link status. The optimal ***mux-pkt*** size is calculated at each hop using a specific equation of signal-to-noise ratio (SNR). Four parameters are used to control the ***mux-pkt*** size and the multiplexing process: $\mathbf{SIZE_{min}}$, which is the minimum size of the ***mux-pkt***; $\mathbf{SIZE_{max}}$, which is the maximum size of the ***mux-pkt*** and is determined as a function of SNR; $\mathbf{MAX_{delay}}$, which is the maximum waiting time to multiplex the packets; and **SIZE$_{factor}$**, which compromises between packet loss caused by frame errors and congestion. **SIZE$_{max}$** is determined as a function of SNR. If available packet size > **SIZE$_{min}$**, then the packets are multiplexed while ***mux-pkt*** size < **SIZE$_{max}$** * **SIZE$_{factor}$**. If waiting time > **MAX$_{delay}$**, then the packets are multiplexed while ***mux-pkt*** size < **SIZE$_{max}$**. After accomplishing multiplexing, a 20-byte IP header is attached to each ***mux-pkt***. At least two packets should be available to perform the multiplexing process. The result for the tested cases showed that the bandwidth utilization was improved by more than three times that in the traditional situation. In addition, the QoS (delay, jitter, and packet loss) is acceptable when the number of connections is around 40 with “no aggregation” scenario and 120 with “aggregation” scenario. If the number of connections exceeds these connections or traffic is low, then the QoS is degraded.

The works in [40] and [41] proposed a method that combines packet multiplexing and packet differentiation. Upon receiving the packets, the hop differentiates them using the differentiated service code point (DSCP) field of the IP. These packets are then distributed into four queues; one of the queues is for VoIP traffic. Each queue uses a packet multiplexing method to save bandwidth. For the VoIP, the work proposes an adaptive multiplexing method that works at the DLL layer. This method imposes a multiplexing delay on the ingress hop only and uses the natural queuing delay for multiplexing at the intermediate hops. Therefore, this method reduces the end-to-end delay that results from hop-by-hop methods and improves bandwidth utilization by performing multiplexing at each hop. To calculate the optimal ***mux-pkt*** size, the measurable routing metrics and the number of stations in range were used in a specific formula. The ***mux-pkt*** is produced and transmitted when multiplexing period is expired or optimal size is reached. The result for the tested cases showed that the bandwidth utilization was improved by around four times that in the traditional situation. The method imposes less delay when the number of flows is small and higher delay when the number of flows increases. However, the packet loss is less than that in the traditional method.

In [42], the author proposed a novel multiplexing method that multiplexes the VoIP packets from the same call in one ***mux-pkt***. Typically, VoIP packets from the same call are multiplexed by elongating the voice payload at the source of the call, thereby increasing the packetization delay. Unlike the typical methods, this work proposed to multiplex the VoIP packets from the same call at each router from source to destination with zero multiplexing delay. The resulted ***mux-pkt*** shares the same UDP and IP header. The multiplexing is performed based on the congestion level of the router. The basic philosophy of this method is that the current bandwidth is sufficient if no congestion is detected. Therefore, performing multiplexing is unnecessary. By contrast, if congestion is detected and the packets are waiting in the queue, then the method uses this waiting time to multiplex the packets from the same call without any additional multiplexing delay. Clearly, the ***mux-pkt*** size is adaptive based on the congestion level in the router. For example, if light congestion is detected on the router, then the number of multiplexed packets is low; if high congestion is detected on the router, then the number of multiplexed packets is high. The result showed that the number of calls was double those in the traditional situation, without any additional delay.

Hasegawa, Jun et al. [43] proposed a unique method that combines both packet multiplexing and network coding. Their proposed technique utilizes some network features to improve the network transmission capacity. The proposed method is called bidirectional packet multiplexing and coding (BiPAC). BiPAC performs multiplexing and network coding in each hop and inserts a mini-header between the DLL and IP headers to restore multiplexed packets. Among many network coding techniques, XOR-based network coding was used by BiPAC because this coding is efficient in WMNs. Encoding increases packet overhead because the packet IDs of all XORed packets must be added as a header for network coding. To reduce header overhead, BiPAC multiplexes the packets and then performs the XOR operation on the multiplexed packets. The result for the tested cases showed that the bandwidth utilization was improved by nearly four and a half times that in the traditional situation. The method imposes high delay when the number of flows is small and nearly the same delay when the number of flows increases. However, the packet loss is less than that in the traditional method.

The work in [10] proposed a hop-to-hop multiplexing method that multiplexes several packets at the network layer. The method was implemented over DCF channel because it is more common than PCF channel. The author argued that the proposed multiplexing method was implemented to run over IP level routing, and thus, the method can be implemented to run over DLL layer routing. A mini-header separates the packets within the ***mux-pkt***. The receiver uses the information in the mini-header to extract the packets from the ***mux-pkt.*** The size of ***mux-pkt*** changes adaptively based on the link status. The signal-to-noise and interference ratio (SNIR) for measuring connection quality was used in a certain function to calculate the optimal ***mux-pkt*** size. In [30], [44], and [45], similar multiplexing method was adopted and implemented under different WMN scenarios. The result for the tested cases showed that the bandwidth utilization improved by more than three times that in the traditional situation. The QoS (delay, jitter, and packet loss) is improved compared with that in the traditional method.

The work in [46] combined RObust header compression (ROHC) and packet multiplexing of VoIP packets. A series of testing and evaluation found that ROHC and packet multiplexing highly improved bandwidth utilization and decreased end-to-end delay. However, the bandwidth utilization was degraded because of the processing time of ROHC and packet multiplexing. To reduce the processing time, this work proposed a specific-purpose processor model instead of a general-purpose processor. The proposed model handles ROHC and packet multiplexing. Result showed that the performance improved when using the proposed processor model than when the typical processor is used.

In summary, this subsection discusses the adaptive multiplexing methods. The adaptive multiplexing methods can be divided into two groups. The first group is composed of hop-to-hop methods, including the works in [10], [30], [39], [42], [43], [44], and [45]. The second group comprised hybrid (hop-to-hop/end-to-end) methods, including the works in [36], [37], [38], [40], and [41].

In the first group (hop-to-hop adaptive multiplexing methods), the works in [10], [30], [45], and [46] use the SNIR in a specific equation to evaluate the network performance and thus determine the mux-pkt size. The works in [42] and [43] use size and delay in a specific equation to determine the mux-pkt size. The work in [39] uses SNR and delay in a specific equation to determine the mux-pkt size. In [39], two quality measures (SNR and delay) are used to evaluate the link status, thereby providing a more suitable mux-pkt size as evidenced by the result. Such a result is obtained because the work in [39] improves the bandwidth utilization and keeps acceptable VoIP quality. However, in general, the hop-to-hop adaptive group provides suitable mux-pkt size to a high extent because the methods in this group use the network conditions to calculate the size. In addition, this group achieves high bandwidth utilization because the possibility of multiplexing more packets increases with the number of hops. However, multiplexing packets on every hop in the route imposes additional delay and thus degrades the voice quality. Accordingly, a packet multiplexing method should improve bandwidth utilization and avoid imposing additional delay, as in the second group (hybrid adaptive multiplexing methods) summarized below [37], [38], [40].

In the second group (hybrid adaptive multiplexing methods), the work in [30] uses MTU to determine the ***mux-pkt*** size. The work in [36] uses size to determine the ***mux-pkt*** size. The works in [37], [38], and [9] use size and delay to determine the ***mux-pkt*** size. The works in [40] and [41] use routing metrics and number of stations in specific equation to determine the ***mux-pkt*** size. In [35] and [36], the works depend on the MTU and size, respectively, to determine the ***mux-pkt*** size. These works also ignored the link status. Therefore, the resulting ***mux-pkt*** size might be unsuitable for the network, thereby potentially reducing the efficiency of bandwidth utilization and degrading the voice quality, as discussed earlier. In [40] and [41], the works may also reduce the efficiency of bandwidth utilization and degrade voice quality because they use routing metrics and number of stations to determine the ***mux-pkt*** size. These methods also did not consider the link status. The works in [37], [38], and [9] use delay in addition to size and thus resulting in more suitable ***mux-pkt*** size. Accordingly, voice quality and bandwidth utilization are better. In addition, they maintain the VoIP quality by sending the packets though the route with less delay and packet loss. However, contrary to the work in [32], which performs the multiplexing at the MAC layer, the work in [38] performs multiplexing at the network layer. The latter work has been patented in [9]. Therefore, more efficient header compression, better bandwidth utilization, and improved voice quality are obtained. Accordingly, the work in [38] and its patent [9] may be considered the best available VoIP multiplexing methods over WMNs. However, in general, the hybrid adaptive group provides a suitable ***mux-pkt*** size to a high extent because they use the network conditions to calculate the size. In addition, this group forces a variable multiplexing delay (based on the link status) on the ingress hop only and uses the natural queuing delay for multiplexing at the intermediate hops. Therefore, similar to end-to-end multiplexing methods, these methods reduce the end-to-end delay that results from the hop-by-hop multiplexing. Moreover, similar to hop-by-hop multiplexing methods, they improve bandwidth utilization by performing multiplexing at each hop. By contrast, this group shows bandwidth utilization that is close to that gained by the hop-by-hop method group. Therefore, using hybrid adaptive multiplexing methods is more favorable for WMNs [37], [38], [40].

This section introduces the present methods of multiplexing VoIP packet over WMNs. The multiplexing methods were divided into adaptive and non-adaptive methods. In non-adaptive methods, the ***mux-pkt*** size was determined based on specific thresholds: delay and size. In adaptive methods, the ***mux-pkt*** size was determined based on the link quality parameters: delay, size, SNR, SNIR, routing metrics, and number of stations. **Table 2** summarizes the adaptive and non-adaptive methods.

Table 2. Summary of adaptive and non-adaptive multiplexing methods

Method	**Adaptive/non-adaptive**	**Hop-to-hop/ end-to-end**	**Multiplexing layer**	***mux-pkt* size determinator**	**Other contribution(s)**
Ref [31]	Non-adaptive	Hop-to-hop	MAC layer	Delay Size	N/A
Ref [32]	Non-adaptive	Hop-to-hop	MAC layer	Delay Size	N/A
Ref [33]	Non-adaptive	Hop-to-hop	MAC layer	Size	Header compression
Ref [26]	Non-adaptive	End-to-end	Network layer	Delay	Header compression
Ref [34]	Non-adaptive	End-to-end	MAC layer	Size	N/A

Ref [35]	Adaptive	Hybrid: hop-to-hop/ end-to-end	Network layer	MTU	Header compression Routing
Ref [36]	Adaptive	Hybrid: hop-to-hop/ end-to-end	MAC layer	Size	Header compression Routing
Ref [37]	Adaptive	Hybrid: hop-to-hop/ end-to-end	MAC layer	Delay Size	N/A
Ref [38]	Adaptive	Hybrid: hop-to-hop/ end-to-end	Network layer	Delay Size	Header compression Routing Mobility
Ref [9]	Adaptive	Hybrid: hop-to-hop/ end-to-end	Network layer	Delay Size	Header compression
Ref [39]	Adaptive	Hop-to-hop	MAC layer	SNR Delay	N/A
Ref [40]	Adaptive	Hybrid: hop-to-hop/ end-to-end	MAC layer	Routing metrics Number of stations	Packet differentiation
Ref [41]	Adaptive	Hybrid: hop-to-hop/ end-to-end	MAC layer	Routing metrics Number of stations	Packet differentiation
Ref [42]	Adaptive	Hop-to-hop	MAC layer	Size Delay	N/A
Ref [43]	Adaptive	Hop-to-hop	MAC layer	Size Delay	Network coding
Ref [10]	Adaptive	Hop-to-hop	Network layer	SNIR	N/A
Ref [30]	Adaptive	Hop-to-hop	Network layer	SNIR	N/A
Ref [44]	Adaptive	Hop-to-hop	Network layer	SNIR	N/A
Ref [45]	Adaptive	Hop-to-hop	Network layer	SNIR	N/A
Ref [46]	Ref [46] proposes a processor model used to reduce the multiplexing process time and can be applied to any multiplexing method				

4. Discussion

4.1 Effect of packet multiplexing on bandwidth utilization

The main purpose of the VoIP packet multiplexing methods is to improve network bandwidth utilization. Multiplexing methods achieve this aim in several aspects. First, on the one hand, the typical VoIP packet payload size between 10 and 30 bytes depends on the codec. On the other hand, up to 104-byte header is added to each payload. Accordingly, the header overhead, which is the relative ratio between the header size and the packet size, is between around 77.5% to 92%. **Fig. 9** demonstrates the header overhead with different payload sizes. When multiplexing several packets in one header, the header overhead decreases depending on the number of multiplexed packets in the ***mux-pkt***, as shown in **Fig. 10**. Second, as discussed in Section 2.2, 841 µs delay is induced by each small packet. Therefore, when multiplexing several small packets in one large packet, the 841 µs delay is shortened in this large packet, thereby increasing the number of transmitted packets. Third, the capacity (number of concurrent calls) of the link will increase because of two reasons: i) the reduced header overhead, which saves the bandwidth and allows the link to accommodate more calls; and ii) the shortened 841 µs delay of each small packet to one large packets, which reduces the link busy time and allows more calls to be accommodated in the link [15], [47], [48], [49]. All the aforementioned factors (header overhead, delay of each packet, and capacity) reflect the bandwidth utilization. On the basis of these factors, multiplexing methods highly improve bandwidth utilization. **Fig. 11** shows the capacity of the method in [38], because this method has been selected by the author as one of the best methods. As we can see, the capacity with packet multiplexing is greater than the capacity without multiplexing. However, the capacity decreases when the number of hops increases.

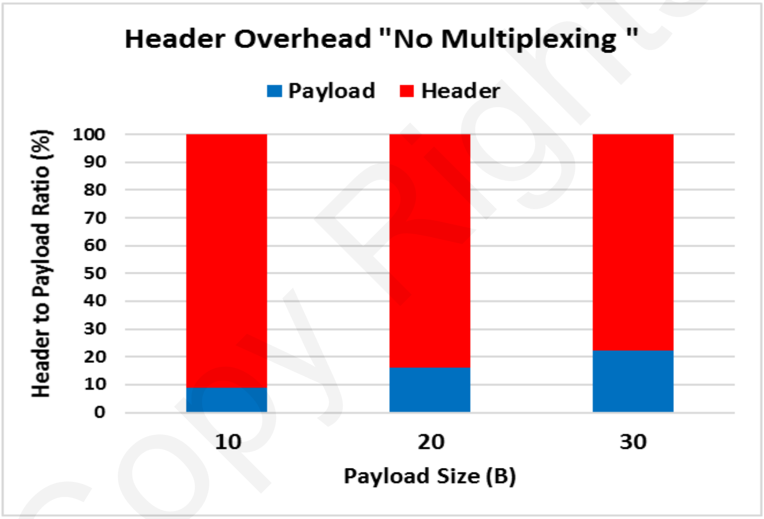

Fig. 9. Header overhead ratio without multiplexing

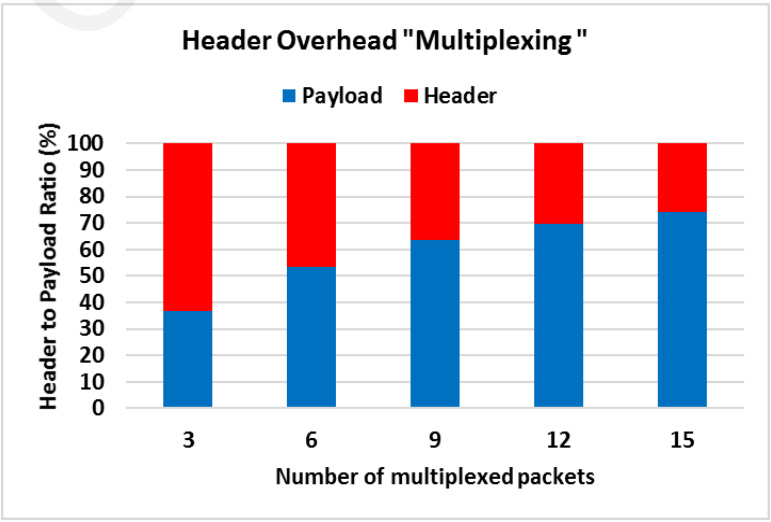

Fig. 10. Header overhead ratio with multiplexing

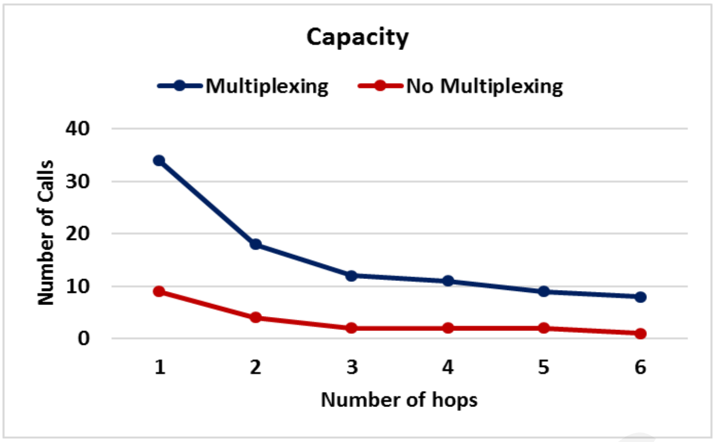

Fig. 11. Number of calls with and without multiplexing

4.2 Effect of packet multiplexing on VoIP quality

VoIP technology must provide high conversation quality to ensure user satisfaction. Several factors affect the quality of VoIP. The first factor is delay. According to the ITU, the acceptable end-to-end delay should be within 150 µs. In addition to the natural delay (packetization, transmission, processing, and queuing delay), the multiplexing methods increase the end-to-end delay because of the imposed multiplexing period. Furthermore, the delay increases on the basis of the processing time that results from multiplexing/de-multiplexing operation. However, the multiplexing methods compensate for this extra delay by decreasing the queuing delay through processing and forwarding one large packet instead of many small packets. In addition, the 841 µs delay is shortened to one large packet instead of several small packets. Multiplexing methods might increase or decrease the delay. Therefore, a mechanism to estimate the amount of time to hold the packet for aggregation and compensate the delay is needed [15], [16], [50], [51], [52]. The second factor is jitter. A high jitter corresponds to a low voice quality. The main causes of jitter are queuing time, contention, and the use of different routes. Multiplexing methods insignificantly affect the jitter in that they improve it slightly. The possible reasons are as follows: i) packet multiplexing decreases the number of packets and thus reduces the contention; ii) packets need to be delayed for some time to accomplish multiplexing; and iii) fewer routes will be traversed because one large packet will be sent instead of multiple small packets [7] , [37], [38]. The third factor is packet loss. For VoIP, long delayed packets are considered lost. The acceptable packet loss ratio depends on the codec and packet size. Different codecs use different packet loss concealment (PLC) algorithms to fill the lost packet with approximate packet. However, less than 4% packet loss is generally acceptable. Multiplexing methods decrease packet loss by improving buffer utilization. This phenomenon is driven by replacing the small packet header with one header of large packet; thus, the buffer takes in more packets. However, a larger packet is more susceptible to damage in high error rate links. Accordingly, a larger packet is more likely to be lost, thereby increasing packet loss rate. In addition, a lost large packet is considered lost bursty traffic. Thus, PLC algorithms are less effective [8], [10], [50], [53]. Therefore, the number of packets to be aggregated into the ***mux-pkt*** is an important factor and requires a suitable mechanism to be calculated. R-score is a scale to measure voice quality from 1 to 100. The result in [38], as it has been elected by the author as one of the best methods, shows decreasing in packet loss, nearly same delay and jitter in comparison ot traditional method.

4.3 Handicaps of packet multiplexing

Several issues limit VoIP packet multiplexing over WMNs. First, packet multiplexing degrades the QoS of VoIP. As discussed above, the delay, jitter, and packet loss will increase if the multiplexing method is unsuitable [7], [8], [18], [37], [38], [54]. Second, bandwidth utilization is ineffective in the case of few calls and multi-path routing because fewer packets are available for multiplexing [52], [55]. Third, multiplexing multiple packets from different streams requires the same QoS to all streams, thereby eliminating the traffic prioritization feature [52], [55]. Finally, multiplexing/de-multiplexing operation increases the processing overhead on the hops [46]. However, all these obstacles can be alleviated by choosing suitable criteria for designing a multiplexing method, as discussed in Section 5. Designing a suitable multiplexing method that improves bandwidth utilization is possible but remains a challenge for researchers. **Table 3** summarizes the elements affected by packet multiplexing.

Table 3. Elements affected by packet multiplexing

Element	Effect
Header overhead	Reduces the header overhead because multiple packets are combined in a single header instead of a header to each packet
Bandwidth utilization	Improves the bandwidth utilization because the header overhead is reduced
Capacity	Increases the capacity because the header overhead is reduced and the 841 µs delay to one packet is shortened. Therefore, more calls can be accommodated.
Delay	On the one hand, extra delay can occur because of the multiplexing/de-multiplexing processing time and the multiplexing delay period. On the other hand, the queuing delay decreases by processing and forwarding one large packet instead of many small packets. In addition, the 841 µs delay overhead of each small packet will be shortened to one large packet.
Packet loss	On the one hand, packet loss will decrease because of improved buffer utilization, thereby allowing the buffer to take in more packets. On the other hand, larger packet is more susceptible to damage and thus more susceptible to packet loss. Accordingly, in the environments that have high error rate, packet multiplexing will increase the packet loss. In other environments that have less error rate, packet multiplexing will decrease the packet loss.
Traffic	Reduces the number of packets because multiple packets are multiplexed together.
Routing (forwarding) process	Improves packet forwarding process because the number of packets is reduced.
Network overload	Reduces the header overhead and improves bandwidth utilization. In addition, reducing the traffic accelerates the process of packet forwarding, thereby reducing network overload.
Congestion	Reduces the header overhead and improves bandwidth utilization. In addition, reducing the traffic accelerates the process of packet forwarding, thereby reducing netwrok overload and congestion consequently.

5. Guidelines and Potential Research

The main goal of VoIP packet multiplexing is to maximize the link capacity while enhancing or at least maintaining the same VoIP QoS. Several issues should be considered when designing a multiplexing method to achieve this goal. First, an important issue is to choose between adaptive method and non-adaptive method. The ***mux-pkt*** size is the main important factor to consider when designing a multiplexing method [7]. As discussed earlier, the ***mux-pkt*** size in non-adaptive multiplexing methods is bounded by certain thresholds and does not consider the link status. This design might cause improper ***mux-pkt*** size. The improper ***mux-pkt*** size causes many problems. For example, i) a large packet size increases packet loss in high error rate links, as shown in the discussion earlier; and ii) in stable links, a small ***mux-pkt*** size degrades the bandwidth utilization efficiency. By contrast, adaptive multiplexing methods change the ***mux-pkt*** size adaptively based on the link status. Therefore, these methods provide proper ***mux-pkt*** size and thus result in better bandwidth utilization and fewer packets compared with those in non-adaptive multiplexing methods [7], [8], [10], [15]. Accordingly, the adaptive multiplexing methods are preferred over non-adaptive multiplexing ones. Second, in the adaptive multiplexing methods, ***mux-pkt*** size should be calculated based on a robust equation that considers suitable quality parameters. Third, the multiplexing layer should be selected properly. The multiplexing layer highly affects the bandwidth utilization. In case of application layer, multiple voice frames combined in one packet at the sender side reduce the header overhead effectively. This reduction leads to a significant increase in the packetization delay. Moreover, the ***mux-pkt*** will be sensitive to bursty packet loss. In case of network layer, efficient header compression might be applied along with the packet multiplexing, thereby highly improving bandwidth utilization. In addition, multiplexing can be adjusted to suit the network situation. In case of DLL (MAC) layer, multiplexing can be adjusted to suit the network situation and obtain a greater header overhead reduction than in the network layer. However, the bandwidth utilization that results from combining packet multiplexing and header compression at the network layer is higher than that obtained by reducing header overhead at the DLL. Accordingly, multiplexing at the network layer achieves better bandwidth utilization because packet multiplexing and header compression are combined. The multiplexing can be adjusted to suit the network situation, which makes this method the suitable choice. However, its protocol fields need to be changed during the encapsulation process. In addition, WMN MAPs usually operate at the DLL layer and does not work at higher layers [7], [51], [56]. Fourth, the hop-to-hop methods achieve better bandwidth utilization than end-to-end methods do. By contrast, end-to-end methods impose less delay than hop-to-hop methods do. Therefore, combining the two techniques obtains better results. The combination can be achieved by forcing multiplexing delay only on the ingress hop and using the natural queuing delay for multiplexing at the intermediate hops. Specifically, the additional delay is imposed only at the ingress/egress node as in the end-to-end methods, and multiplexing is conducted at each hop to achieve bandwidth utilization close to that in hop-to-hop methods [14], [29], [35]. A robust and efficient multiplexing method can be designed when these issues are considered. However, other trends are not investigated in this work. This limitation may be reason that the performance of the multiplexing methods appears more improved over the other methods. These trends can be addressed in future research. Some of these trends are as follows:

- One of the techniques for improving VoIP bandwidth utilization is header compression. Header compression successfully compressed 40-byte IP/UDP/RTP headers to 2 bytes in some compression techniques. The key point to achieve this high compression is to utilize the duplicated fields in the IP/UDP/RTP headers of the consecutive packets. In some multiplexing methods, several VoIP packets are combined within the ***mux-pkt***, each with separate IP/UDP/RTP headers. Some current works apply the typical header compression on these headers. However, a new header compression technique that operates with the multiplexing methods can be developed; such a technique can utilize the redundant fields in the IP/UDP/RTP headers within the ***mux-pkt*** [9], [57], [58].

- Fuzzy logic is a widely used technique and has been used for intelligence control. This technique is used to design robust systems when specific factors are available, such as nonlinearity, parameter uncertainty, and measurement and modeling imprecision. Fuzzy logic theory provides an adjacent controller design approach based on expert knowledge, which is close to human decision making. This theory also helps in modeling complicated non-linear systems. Fuzzy logic has been extensively applied in computer networks (e.g., network congestion control [59]), and showed unexpected and complete control performance in accuracy, transient response, robustness, and stability. Accordingly, the efficiency of applying fuzzy logic with VoIP packet multiplexing is worth investigating [60].
- As discussed above, the adaptive methods achieved better performance because they use the link quality to calculate the optimal ***mux-pkt*** size. The quality metrics used by the current adaptive method formulas include delay, SNR, SNIR, routing metrics, and number of stations. However, other metrics, such as packet loss, jitter, and route errors, should be analyzed and considered when designing the adaptive method formulas.

6. Conclusion

The prominent characteristics of making voice calls over an IP network have encouraged users and service providers to migrate their traditional telecommunication systems to VoIP telecommunication systems. WMNs have been widely applied in all sectors, including homes, schools, universities, and companies. Thus, combining VoIP and WMN is an inevitable issue in all sectors. Inefficient bandwidth utilization is one of the serious problems that hinder the propagation of VoIP over WMNs. Packet multiplexing combines several VoIP packets in one header and is one of the main methods for handling bandwidth utilization problems. In this work, we surveyed the entire current packet multiplexing methods of VoIP packets over WMNs. This study provided a detailed investigation of the present multiplexing methods over WMNs. This study also provided a clear understanding of the bandwidth utilization problem, guidelines for designing a new multiplexing method, and directions for future research. We plan to extend our study by including all the VoIP packet multiplexing methods over wired and wireless networks (e.g., 802.11 standard).

References

[1] http://www.statista.com/statistics/267183/forecast-for-the-worldwide-voip-traffic/, November-2015.
[2] http://www.trutower.com/2014/01/13/skype-voip-app-calling-statistics-telegeography/, November- 2015.
[3] Devi, G. Usha, et al., "VoIP over Mobile Wi-Fi Hotspot," *Indian Journal of Science and Technology 8.S2*, 195-199, 2015. Article (CrossRef Link)
[4] Avula, Mallikarjun, Sang-Gon Lee, and Seong-Moo Yoo, “Security Framework for Hybrid Wireless Mesh Protocol in Wireless Mesh Networks,” *TIIS 8.6*, 1982-2004, 2014. Article (CrossRef Link)
[5] Agrawal, Dharma and Qing-An Zeng, “Introduction to wireless and mobile systems,” *Cengage Learning*, 4th edition, 2015.

[6] Yen, Li–Hsing, and Kuo–Wei Huang, "Link–preserving interference–minimisation channel assignment in multi–radio wireless mesh networks," *International Journal of Ad Hoc and Ubiquitous Computing 18*, no. 4, 222-233, 2015. Article (CrossRef Link)

[7] Zulu, Docas Dudu, "Packet aggregation for voice over internet protocol on wireless mesh networks," *PhD diss.*, University of the Western Cape, 2012.

[8] Dely, Peter, "Adaptive Aggregation of Voice over IP in Wireless Mesh Networks," *Master diss*, Karlstad University, 2007.

[9] Kim, Kyungtae, and Rauf Izmailov, "On Packet Aggregation and Header Compression Mechanisms for Improving VoIP Quality in Mesh Networks," U.S. Patent Application 11/683,641, filed March 8, 2007.

[10] Okech JM, Odhiambo MO and Kurien A, "Packet VoIP Aggregation: A Mechanism to Improve the Performance of VoIP in Wireless Mesh Networks (WMNs)", *IST Transactions of Electrical and Electronic Systems-Theory and Applications*, Vol. 1, No. 1 (2), pp.28-36, March 2012.

[11] Wang, Wei, Soung Chang Liew and Victor OK Li, "Solutions to performance problems in VoIP over a 802.11 wireless LAN," *Vehicular Technology*, IEEE Transactions on 54.1, 366-384, 2005. Article (CrossRef Link)

[12] Wang, Ping, Hai Jiang and Weihua Zhuang, "IEEE 802.11 e enhancement for voice service," *Wireless Communications, IEEE 13.1*, 30-35, 2006. Article (CrossRef Link)

[13] Olariu, Cristian, et al., "A Delay-aware Packet Prioritisation Mechanism for Voice over IP in Wireless Mesh Networks," in *Proc. of 2016 IEEE Wireless Communications and Networking Conference (WCNC)*, Doha, Qatar 3-6 April 2016. IEEE, 2016.

[14] Azevêdo Filho, Paulo H., Marcos F. Caetano and Jacir L. Bordim, "A packet aggregation mechanism for real time applications over wireless networks," *International Journal of Networking and Computing 2*, no. 1,18-40, 2012. Article (CrossRef Link)

[15] Vulkan, Csaba, Attila Rakos, Zoltan Vincze and Arpad Drozdy, "Reducing overhead on voice traffic," U.S. Patent 8,824,304, issued September 2, 2014.

[16] Abu-Alhaj, Mosleh M., S. K. Manjur, R. Sureswaran, Tat-Chee Wan, Imad J. Mohamad and Ahmed M. Manasrah, "ITTP: A New Transport Protocol for VoIP Applications," *International Journal of Innovative Computing*, Information and Control (IJICIC) 8, 2012.

[17] Abu-Alhaj, Mosleh M., "ITTP-MUX:An efficient multiplexing mechanism to improve voip applications bandwidth utilization," *International Journal of Innovative Computing, Information and Control (IJICIC)*, Vol 11, No. 6, pp. 1–1503-0010, 2015.

[18] Abu-Alhaj, Mosleh M., Manjur S. Kolhar, Lingeswari V. Chandra, O. Abouabdalla and Ahmed M. Manasrah, "Delta-Multiplexing: A Novel Technique to Improve VoIP Bandwidth Utilization between VoIP Gateways," in *Proc. of Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pp. 329-335. IEEE, 2010. Article (CrossRef Link)

[19] Jahanshahi, Mohsen and Alireza Talebi Barmi, "Multicast routing protocols in wireless mesh networks: a survey," Computing 96, no. 11, 1029-1057, 2014. Article (CrossRef Link)

[20] IEEE Computer Society, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” 2012.

[21] Hiertz, Guido R., Dee Denteneer, Sebastian Max, Rakesh Taori, Javier Cardona, Lars Berlemann and Bernhard Walke, "IEEE 802.11 s: the WLAN mesh standard," *Wireless Communications*, IEEE 17, no. 1, 104-111, 2010. Article (CrossRef Link)

[22] Manjur, S. K., Mosleh Abu-Alhaj, Omar Abouabdalla, Tat-Chee Wan, Rahmat Budiarto and Ahmed M. Manasrah, "Conference gateway for heterogeneous clients: Real time switching clients and interasterisk exchange clients," *INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING INFORMATION AND CONTROL 7*, no. 1, 395-406, 2011.

[23] Saldana, Jose, Julián Fernández-Navajas, José Ruiz-Mas, Jenifer Murillo, Eduardo Viruete Navarro and José I. Aznar, "Evaluating the influence of multiplexing schemes and buffer implementation on perceived VoIP conversation quality," *Computer Networks 56*, no. 7, 1893-1919, 2012. Article (CrossRef Link)

[24] Saldana, Jose, Dan Wing, Julian Fernandez-Navajas, Jose Ruiz-Mas, Muthu AM Perumal and Gonzalo Camarillo, "Widening the scope of a standard: Real time flows tunneling, compressing and multiplexing," in *Proc. of Communications (ICC), 2012 IEEE International Conference on*, pp. 6906-6910. IEEE, 2012. Article (CrossRef Link)

[25] Wang, Wei, Soung Chang Liew and Victor OK Li, "Solutions to performance problems in VoIP over a 802.11 wireless LAN," *Vehicular Technology, IEEE Transactions on 54*, no. 1, 366-384, 2005.

[26] Kekre, H. B., V. A. Bharadi, R. S. Bansode and Vikas Kaul, "Performance problems of voip in 802.11 wireless mesh networks & their solutions," in *Proc. of the International Conference & Workshop on Emerging Trends in Technology*, pp. 891-898. ACM, 2011.

[27] Yun, Sangki, Hyogon Kim, Heejo Lee and Inhye Kang, "100+ VoIP calls on 802.11 b: The power of combining voice frame aggregation and uplink-downlink bandwidth control in wireless LANs," *Selected Areas in Communications, IEEE Journal on 25*, no. 4, 689-698, 2007. Article (CrossRef Link)

[28] AbuAlhaj, Mosleh, Manjur S. Kolhar, M. Halaiyqah, O. Abouabdalla and R. Sureswaran, "Multiplexing SIP applications voice packets between SWVG gateways," in *Proc. of International Conference on Computer Engineering and Applications (ICCEA 2009)*, 2009.

[29] Jung, Ji-Young, Hyun-Sik Kang and Jung-Ryun Lee, "Performance evaluation of packet aggregation scheme for VoIP service in wireless multi-hop network," *Ad Hoc Networks 11*, no. 3, 1037-1045, 2013. Article (CrossRef Link)

[30] Okech, James, Yskandar Hamam, Anish Kurien, Thomas Olwal and Marcel Odhiambo, "A Dynamic Packet Aggregation Scheme for VoIP in Wireless Mesh Networks," in *Proc. of International Journal Of Computer Science*, 2008.

[31] Bayer, Nico, Marcel Cavalcanti De Castro, Peter Dely, Andreas Kassler, Yevgeni Koucheryavy, Piotr Mitoraj and Dirk Staehle, "VoIP service performance optimization in pre-IEEE 802.11 s Wireless Mesh Networks," in *Proc. of Circuits and Systems for Communications, 2008. ICCSC 2008. 4th IEEE International Conference on*, pp. 75-79. IEEE, 2008. Article (CrossRef Link)

[32] Castro, Marcel C., Peter Dely, Jonas Karlsson and Andreas Kassler, "Capacity increase for voice over IP traffic through packet aggregation in wireless multihop mesh networks," in *Proc. of Future Generation Communication and Networking (FGCN 2007)*, vol. 2, pp. 350-355. IEEE, 2007. Article (CrossRef Link)

[33] Ganguly, Samrat, Dragos Niculescu, Kyung Tae Kim and Rauf Izmailov, "Improved voic-over-internet-protocol method," *U.S. Patent Application 11/307,948*, filed February 28, 2006.

[34] Marwah, Khyati, and Gurpal Singh, "VoIP over WMN: Effect of packet aggregation," *International Journal on Computer Science and Engineering 3*, no. 6, 2323-2331, 2011.

[35] Niculescu, Dragos, Samrat Ganguly, Kyungtae Kim and Rauf Izmailov, "Performance of VoIP in a 802.11 Wireless Mesh Network," in *Proc. of INFOCOM*, 2006. Article (CrossRef Link)

[36] Kim, Kyungtae, and Sangjin Hong, "VoMESH: voice over wireless mesh networks," in *Proc. of Wireless Communications and Networking Conference*, 2006. WCNC 2006. IEEE, vol. 1, pp. 193-198. IEEE, 2006. Article (CrossRef Link)

[37] Kim, Kyungtae, Samrat Ganguly, Rauf Izmailov and Sangjin Hong, "On packet aggregation mechanisms for improving VoIP quality in mesh networks," in *Proc. of Vehicular Technology Conference*, 2006. VTC 2006-Spring. IEEE 63rd, vol. 2, pp. 891-895. IEEE, 2006. Article (CrossRef Link)

[38] Ganguly, Samrat, Vishnu Navda, Kyungtae Kim, Anand Kashyap, Dragos Niculescu, Rauf Izmailov, Sangjin Hong and Samir R. Das, "Performance optimizations for deploying voip services in mesh networks," *Selected Areas in Communications*, IEEE Journal on 24, no. 11, 2147-2158, 2006. Article (CrossRef Link)

[39] Kassler, Andreas, Marcel Castro and Peter Dely, "Voip packet aggregation based on link quality metric for multihop wireless mesh networks," in *Proc. of the Future Telecommunication Conference*, Beijing, China. 2007.

[40] Riggio, Roberto, Daniele Miorandi, Francesco De Pellegrini, Fabrizio Granelli and Imrich Chlamtac, "A traffic aggregation and differentiation scheme for enhanced QoS in IEEE 802.11-based Wireless Mesh Networks," *Computer communications 31*, no. 7, 1290-1300, 2008. Article (CrossRef Link)
[41] Riggio, Roberto, Francesco De Pellegrini, Nicola Scalabrino, Pan Li, Yuguang Fang and Imrich Chlamtac, "Performance of a novel adaptive traffic aggregation scheme for wireless mesh networks," in *Proc. of Military Communications Conference*, 2007. MILCOM 2007. IEEE, pp. 1-7. IEEE, 2007. Article (CrossRef Link)
[42] Hyogon, K. I. M., Y. U. N. Sangki and L. E. E. Heejo, "Boosting VoIP capacity of wireless mesh networks through lazy frame aggregation," *IEICE transactions on communications 90*, no. 5, 1283-1285, 2007.
[43] Hasegawa, Jun, Hiroyuki Yomo, Yoshihisa Kondo, Peter Davis, Ryutaro Suzuki, Sadao Obana and Katsumi Sakakibara, "Bidirectional packet aggregation and coding for VoIP transmission in wireless multi-hop networks." in *Proc. of Communications, 2009. ICC'09. IEEE International Conference on*, pp. 1-6. IEEE, 2009. Article (CrossRef Link)
[44] Olwal, T., J. Okech, Y. Hamam, A. Kurien and M. Odhiambo, "Increasing supported VoIP flows in WMNs through link-based aggregation," *IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries*, 2008.
[45] Okech, J., A. Kurien and M. O. Odhiambo, "Link-based VoIP aggregation in mesh networks," in *Proc. of New Technologies, Mobility and Security, 2008. NTMS'08.*, pp. 1-5. IEEE, 2008. Article (CrossRef Link)
[46] Jung, Sangkil, Sangjin Hong and Peom Park, "Effect of robust header compression (rohc) and packet aggregation on multi-hop wireless mesh networks," in *Proc. of Computer and Information Technology, 2006. CIT'06. The Sixth IEEE International Conference on*, pp. 91-91. IEEE, 2006. Article (CrossRef Link)
[47] Blanco, Fernando, Dan Wing, Julian Navajas, Muthu Perumal and Jose Saldana, "Tunneling Compressing and Multiplexing (TCM) Traffic Flows. Reference Model," 2015.
[48] Jun XIAO et al., "Analysis and Simulation for VoIP Capacity in IEEE 802.11 WLAN," *Journal of Computational Information Systems 8*: 19, 7955–7962, 2012.
[49] Di Stasi, Giovanni, et al., "Combining multi-path forwarding and packet aggregation for improved network performance in wireless mesh networks," *Computer Networks 64*, 26-37, 2014. Article (CrossRef Link)
[50] Shin, Sangho and Henning Schulzrinne, "Measurement and Analysis of the VoIP Capacity in IEEE 802.11 WLAN," *Mobile Computing, IEEE Transactions on 8*, no. 9, 1265-1279, 2009. Article (CrossRef Link)
[51] Sze, H. P., Soung C. Liew, Jack YB Lee and Danny Yip, "A multiplexing scheme for H. 323 voice-over-IP applications," *Selected Areas in Communications, IEEE Journal on 20*, no. 7, 1360-1368, 2002. Article (CrossRef Link)
[52] Subbiah, Barani, Senthil Sengodan and Jarno Rajahalme, "RTP payload multiplexing between IP telephony gateways," in *Proc. of Global Telecommunications Conference, 1999. GLOBECOM'99*, vol. 2, pp. 1121-1127. IEEE, 1999. Article (CrossRef Link)
[53] Shin, Sangho and Henning Schulzrinne, "Experimental measurement of the capacity for VoIP traffic in IEEE 802.11 WLANs," in *Proc. of INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 2018-2026. IEEE, 2007. Article (CrossRef Link)
[54] Charfi, Emna, et al., "Dynamic frame aggregation scheduler for multimedia applications in IEEE 802.11 n networks," *Transactions on Emerging Telecommunications Technologies*, 2015. Article (CrossRef Link)
[55] Mosleh M. Abualhaj, Internet Telephony Transport Protocol (Ittp): An efficient transport protocol for voip applications. Diss. University Sains Malaysia, 2012.
[56] Hoshi, Tohru, Keiko Tanigawa and Kohji Tsukada, "Proposal of a method of voice stream multiplexing for IP telephony systems," in *Proc. of Internet Workshop, 1999. IWS 99*, pp. 182-188. IEEE, 1999. Article (CrossRef Link)

[57] Sandlund, K., G. Pelletier and L. E. Jonsson, The RObust Header Compression (ROHC) Framework. No. RFC 5795. 2010.

[58] Koren, T., S. Casner, J. Geevarghese, B. Thompson and P. Ruddy, Enhanced Compressed RTP (CRTP) for links with high delay, packet loss and reordering. No. RFC 3545. 2003.

[59] Baklizi, Mahmoud, Hussein Abdel-Jaber, Ahmad Adel Abu-Shareha, Mosleh M. Abualhaj and Sureswaran Ramadass, "Fuzzy logic controller of gentle random early detection based on average queue length and delay rate," *International Journal of Fuzzy Systems 16*, no. 1, 9-19, 2014.

[60] Liu, Jungang and Oliver WW Yang, "Using Fuzzy Logic Control to Provide Intelligent Traffic Management Service for High-Speed Networks," *Network and Service Management, IEEE Transactions on 10*, no. 2, 148-161, 2013. Article (CrossRef Link)

Dr. Mosleh M. Abu-Alhaj is a senior lecturer in Al-Ahliyya Amman University. He received his first degree in Computer Science from Philadelphia University, Jordan, in July 2004, master degree in Computer Information System from the Arab Academy for Banking and Financial Sciences (AABFS), Jordan in July 2007, and doctorate degree in Computer Networks from Universiti Sains Malaysia (USM) in 2011. His research area of interest includes VoIP, Multimedia Networking, and Congestion Control. Dr. Mosleh, also does consultancy services in the above research areas.

Dr. Manjur Kolhar (m.kolhar@psau.edu.sa) received his Bachelor of Science from KUD, INDIA in 1999 and Master in Computer Applications system from KUD, India in 2001, received PhD degree from National Advanced IPv6 Centre (NAV6) in Universiti Sains Malaysia (USM). In 2010. He has published more than 25 research papers in International Journals and Conferences of high repute. His research interest includes advanced Computer networks and security and cloud computing resource management.

Dr. Kefaya Qaddoum is a senior lecturer in Al-Ahliyya Amman University. She received his first degree in Computer Science from Philadelphia University, Jordan, in July 2003, master degree in Computer Information System from Philadelphia University July 2008, and doctorate degree in A.I from Warwick University, UK in 2013. Her research area of interest includes A.I, Machine Learning, Data Mining and Networking.

Dr. Ahmad Adel Abu-Shareha is an assistant professor and the head of departments of Computer Information Systems (CIS). He received his first degree in Computer Science from Al Al-Bayt University (AABU), Jordan, 2004, master degree from Universiti Sains Malaysia (USM) – Malaysia, 2006, and his Ph.D degree from Universiti Sains Malaysia (USM) – Malaysia, 2012. His research focuses on data mining and artificial intelligent.