Article can be accessed online at http://www.publishingindia.com

ENHANCING THE EFFICIENCY OF MULTIHOP IPV6 TUNNELS USING HEADER COMPRESSION OVER VARYING PACKET SIZES

Dipti Chauhan*, Jay Kumar Jain, Sanjay Sharma*****

***Department of Mathematics & Computer Applications, MANIT, Bhopal, Madhya Pradesh, India.*
Email: diptichauhan09@gmail.com
***Department of Mathematics & Computer Applications, MANIT, Bhopal, Madhya Pradesh, India.*
Email: jayjain.research@gmail.com
****Department of Mathematics & Computer Applications, MANIT, Bhopal, Madhya Pradesh, India.*
Email: drssharma66@gmail.com

Abstract *IPv6 is the next generation Internet protocol which will eventually replace IPv4, but until this happens these two protocols need to coexist for a long time. IPv6 offers several benefits over IPv4, still the adoption rate is very low over worldwide. In all the widely used applications such as email, messaging services, file transfer, web, audio streams, video on demand etc, the IP traffic is common and generates the highest overhead with IPv6 and IPv4 headers of size 40 and 20 bytes, respectively. This overhead could be even more in case of tunnel where one packet is encapsulated inside another packet. This high header overhead could degrade the performance of network especially over wireless links where resources are scarce. Header Compression (HC) results in improved network performance and better utilisation of resources. In this paper we are improving the efficiency of IPv6 tunnels by compressing the IPv6 header of the packet. Here tunnel is a multihop tunnel between edge routers. Performance of HC is checked over two different network conditions: Wireless and Wired Networks. Simulations are carried out over Qualnet 5.1 simulator, different performance parameter like throughput, delay, jitter and packet delivery ratio is been calculated over different packet sizes.*

Keywords: *Bandwidth, Compression, Decompression, IPv6, Multihop, Packet Size*

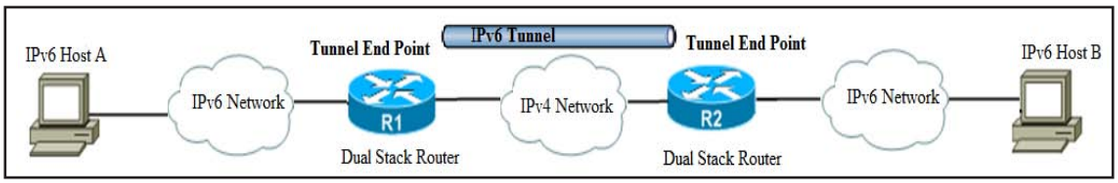
INTRODUCTION

Internet Protocol (IPv4) is the most widely used addressing protocol used over the communication networks. It is a 32 bit addressing protocol, which can address up to 232 devices, i.e. it can address up to 4.3 billion devices (Rey, n.d.). But with the exponential growth rate of internet it is almost impossible to sustain with IPv4. It is expected that by year 2020 there will be about 50 billion devices online (Infographic, n.d.). The problem of address exhaustion has long been anticipated, and with the advent of new devices, applications and technologies it is impossible to sustain with internet protocol IPv4. In 1998 Internet Engineering Task Force (IETF) standardised the next generation internet protocol, known as IPv6, to overcome the problem of address exhaustion which was faced by IPv4. IPv6 (Hinden & Deering, n.d.) is a 128 bit addressing protocol which can address up to 2128 devices, which is much more in comparison with IPv4. These two protocols are not compatible with each other, i.e. IPv4 hosts and routers cannot directly communicate with IPv6 devices and vice versa (Chauhan & Sharma, 2014). Different solutions have been proposed from the past to assist the migration towards IPv6 networks like dual stack, tunneling, and header translation (Raicu & Zeadally, 2003). In this paper we are dealing with enhancing the efficiency of tunneling techniques for enabling the smooth interoperation of the two protocols.

The main issue evolved with both the protocols is different header format, IPv6 uses different header than IPv4 and the size of IPv6 header is double the size of IPv4. The length of IPv4 header is 20 bytes when no option is specified, and IPv6 uses a fixed length header of 40 bytes. This header overhead could affect the performance of network especially over wireless links where resources are scarce, and even more in the case IPv6 tunnels, where one packet is encapsulated inside the other one (Rawat, Bonnin, & Toutain, 2008). IPv6 tunnels are used when the IPv6 sender wants to communicate with an IPv6 host on the other network but the intermediate

network is based on IPv4 and vice versa. In order to enable such a communication, IPv6 tunneling mechanism is used, which will carry IPv6 packet through IPv4 backbone, by encapsulating an IPv6 packet inside an IPv4 packet and sent across the tunnel (Wu *et al*., 2013). Fig. 1 depicts the scenario for IPv6 tunneling mechanism.

Fig. 1: IPv6 Tunneling Scenario



Here an IPv6 host A on IPv6 network wants to communicate with IPv6 host B, via an IPv4 backbone; R1and R2 are dual stack edge routers for the networks. Here an IPv6 tunnel is created between R1 and R2, which will carry an IPv6 packet through IPv4 backbone. At the sender side an IPv6 packet is encapsulated inside an IPv4 packet and sent across the tunnel, at the receiver IPv4 packet is striped off, and the packet is delivered to the intended destination. The main overhead involved with tunneling mechanism is tunneling header overhead which results in high bandwidth usage reduce network efficiency, especially when these tunnels are used over wireless or radio links, where bandwidth remains a scarce resource (Rawat, Bonnin, Minaburo, & Toutain, 2007).

Headers serve very useful purpose over end to end communication, but are of very less importance over a single hop. So it's better to compress these headers due to the redundancy in header fields of the same packet as well as consecutive packets of the same packet stream (Effnet, 2004a). Header Compression is the process of compression excess protocol headers and sends it over a link and then decompresses it at the other end of the link, which would result in many cases more than 90% savings, and thus save the bandwidth and use the expensive resources efficiently (Effnet, 2004b). In this paper we are using header compression technique over IPv6 header of the packet as it is of largest length of 40 bytes. We are using it in context of IPv6 tunnels where IPv6 packet is encapsulated inside the IPv4 packet.

The contributions of this paper are as follows:

1. For context synchronisation, in case of any error in the packet, it does not affect the decompression of subsequent packets, which was a major issue with standard header compression algorithms. Here we have sent dynamic fields of header with every compressed packet, this results in less compression gain but there is no need of context synchronisation in case of packet loss, which was a major issue with other header compression algorithms.
2. We have reduced the Compression / Decompression (C/D) cycles by storing the C/D entity only at the dual stack edge routers, which reduces C/D cycles.
3. IPv6 Header Compression scheme is implemented over multihop IPv6 tunnels. We have compressed the 40 bytes of IPv6 header to 6 bytes which results in improved network performance and enhance the efficiency of IPv6 tunneling mechanism.

Rest of the paper is structured as follows: second section discusses about the literature survey, third section describes about the proposed methodology, fourth section describes the simulation parameters and scenario. Results are discussed in fifth section, and sixth section concludes the paper.

Literature Survey

Jung and Hong (2009) proposed a profile-based network and hardware co-simulation method to investigate the overall performance and real-timing characteristics of a wireless mesh network (WMN) affected by hardware capabilities, speed, and complexity. Here Robust Header Compression (ROHC) and packet aggregation is used for reliable data transmission over wireless links. Dimitriadis, Karapantazis, & Pavlidou (2007) discuss the performance of two header compression mechanisms, Compressed RTP (CRTP) and Enhanced CRTP, over satellite links. Here comparison is done on the basis of two parameters compression ratio and packet loss. To model this, test bed was developed in the Free BSD operating system that models the behaviour of the satellite channel and allows the assessment of these schemes on real voice traffic. Results show that ECRTP performs better than CRTP. Fortuna & Ricardo (2009) discussed the use of ROHC's U mode for VOIP application over 802.11 networks. The study reported that the maximum gain of the ROHC's U-mode when applied to VoIP over IEEE 802.11 is about 23 percent for medium or better voice quality. Yoon *et al*. (2011) proposed a new approach for header compression over IP over tactical data link. Results show that the proposed header compression method outperforms 90.7% for IP packet transmission and maximum 95.8% for tactical data link message compared to existing Link-16. The paper "Impacts of IPv6 on Robust Header Compression in LTE Mobile Networks" (ICNS, 2012) discussed the impact of IPv6 on the Radio Access Networks, i.e., to look into the impacts on the e-Node B in the LTE architecture. LTE uses ROHC with IPv4 in its architecture; the implementation of IPv6 introduces concerns related to expanded packet

headers as the size of packet header doubled from 20 Bytes (IPv4) to at least 40 Bytes (IPv6). ROHC is used in LTE architecture and results show that better compression gains can be achieved in case of IPv6 headers. Tatiana, Frank, Fitzek, Nethi, Arildsen, & Perrucci (2006) focused on the header compression for wireless technologies with fixed link layer packet types. A new header compression algorithm is proposed and is implemented for Bluetooth technology. The performance is checked for two different scenarios and result is compared with IP header compression (RFC 2508). Simulation shows that new algorithm performs better than IP header compression.

Literature shows that most of the research work has been focused on header compression in context of Wireless Networks, MANET's, Mobile IP, and IEEE 802.15.4 standard, i.e. for 6LOWPAN. However there are no studies to the best of our knowledge on examining the impact of header compression for IPv6 tunneling mechanism, so we have compared this with standard tunneling technique. A comparison is made between compressed and uncompressed network, results shows that compression of IPv6 header results in better resource utilisation and better bandwidth savings.

Proposed Methodology

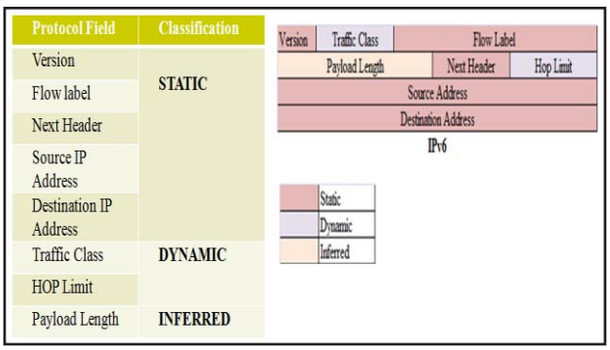
This section describes the proposed methodology for IPv6 header compression. We have compressed only the IPv6 header of the packet as IPv6 packet is largest length of 40 bytes. To do this we have classified the IPv6 header in to the following different categories (Chauhan & Sharma, 2015). Fig. 2 shows the classification of IPv6 header.

STATIC: Static fields are the header fields which remain unchanged during the life time of a header. These fields are sent only with uncompressed packets.

DYNAMIC: These are the fields which change in a specified pattern or randomly. These fields are compressed efficiently.

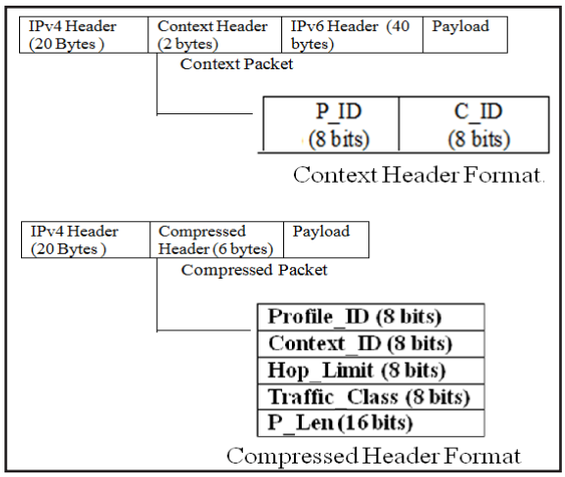
INFERRED: These fields are inferred from the lower layers in the protocol stack.

Fig. 2: IPv6 Header Classification



Based up on this information we are initially sending few packets uncompressed called as context packets to establish context between sender and receiver, once context is established we are sending compressed header. Fig. 3 shows the header formats for context and compressed header. This is an end-to-end header compression, which means that compression and decompression are performed by the edge routers of source and destination nodes. Intermediate nodes must be able to forward the compressed packet without decompressing. The packet will be decompressed only at the destination router. The compressor/decompressor (C/D) entity is stored at the dual stack routers, which reduces C/D cycles.

Fig. 3: Header Formats



We have sent dynamic and inferred fields along with the compressed header. Here there is no need of context updation, since the dynamic information is carried in the compressed header. This which would result in lower compression gain, but if a packet is lost, it does affect the successful decompression of subsequent packets. Following is the algorithm which needs to be followed while doing header compression and decompression.

Algorithm:

Step1: Router A Receives the IPv6 packet:

Step1.1: Determines the next hop on the basis of routing table entries.

Step1.2: Next hop will be tunnels address.

Step 2: At this point IPv6 packet is encapsulated inside IPv4 packet. At network layer we have added a new parameter called tunnel algo to use:

Tunnel algo to use = 0 {Normal Tunneling method}

Tunnel algo to use=1 {Compression tunneling method}

& Number of uncompressed packets to send = n

We will send n uncompressed packet in the tunnel by adding two extra bytes of context header in the packet.

Step 2.1: IPv4 Source Address: Tunnels source address.

Step 2.2: IPv4 destination Address: Tunnels end point.

Step 3: Initially send first few packets uncompressed to maintain context and the static information. Make an entry with a specific context id in the send table. Save all static header info in this table for this specific context id. Also add 2 bytes for profile id and context id in the uncompressed packet before sending it.

Step 3.1: Once the uncompressed packet is received at destination edge router, take out the context id, profile id and static packet info for current profile and save the info in receive table.

Step 3.2: Once context is established then go to step 4.

Step 4: At this point compression algorithm works:

Step 4.1: Replace the IPv6 header with 6 bytes of compressed header, before encapsulation.

Step 4.2: Compressed packet will be encapsulated inside the IPv4 packet.

Step 4.3: Further packets will be sent through the tunnel.

Step 5: At the destination, (i.e. tunnel end point):

Step 5.1: IPv4 header is striped off.

Step 5.2: For n uncompressed packets

First remove the context header after reading the values of p_id and c_id.

Read static info from the IPv6 packet and stores the information for corresponding c_id

At $(n+1)^{Th}$ packet

Read compressed header to get p_id and c_id.

Check the static entry for this corresponding c_id.

Make a new IPv6 header based on this static and dynamic information.

Step 5.3: Based on these static header values and dynamic values coming in current header, reconstruct the original ipv6 header.

Step 6: IPv6 packet is routed onto the IPv6 LAN toward the destination address as specified in the IPv6 packet. Fig. 4 shows the flowchart for steps at sender side:

Fig. 4: Flowchart at Sender Side

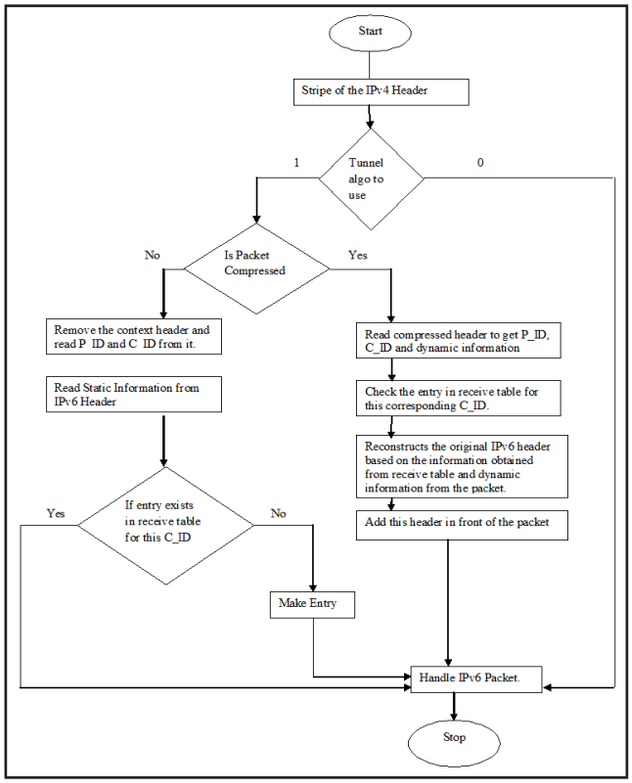
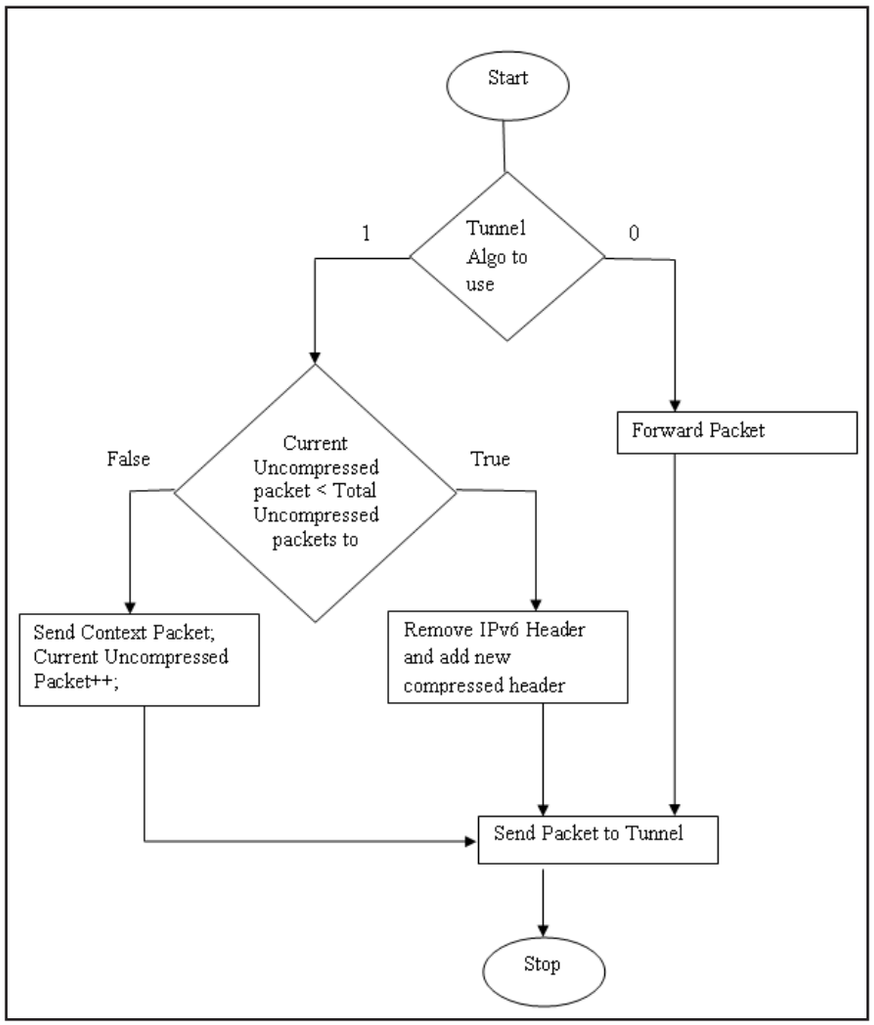


Fig. 5 shows the flowchart for steps at receiver side:

Fig. 5: Flowchart at Receiver Side



Using IPv6 header compression we have compressed 40 bytes of IPv6 header to 6 bytes, thus improving the overall network efficiency. The benefit of this approach is that there is no need of context updation in case of packet loss, since we have sent dynamic information with every packet. Thus error in one packet will not affect the subsequent packets. Result shows that IPv6 header compression results in better bandwidth savings which affect the other network parameters, also its impact can be seen in the results.

Simulation Test Bed

In networks, implementation of a new protocol is very difficult in the real world. Simulator is a very important tool for testing the performance and specifying the network conditions. We can evaluate basic behaviour of a network and test combinations of network features that are likely to work. Various open source and licensed are available like NS-2, NS-3, Opnet, GNS 3, Exata/Cyber, Qualnet etc. In order to test the performance of our algorithm we have tested it over Qualnet 5.1 simulator. QualNet 5.1 is licensed software that provides a comprehensive environment for designing protocols, creating and animating network scenarios, and analysing their performance (Qualnet 5.1 User's Guide). In this paper we have tested our algorithm over wired and wireless network over different packet sizes. Fig. 6 represents the scenario for wireless network; here tunnel is a Multihop wireless tunnel. A Field configuration of 1500m x 1500m is used for the scenario. MAC protocol for wireless network is 802.11. We have used 4 constant bit rate applications to generate the traffic in the network. The sending rate is 100 packets per second.

Fig. 6: Scenario for Wireless Network

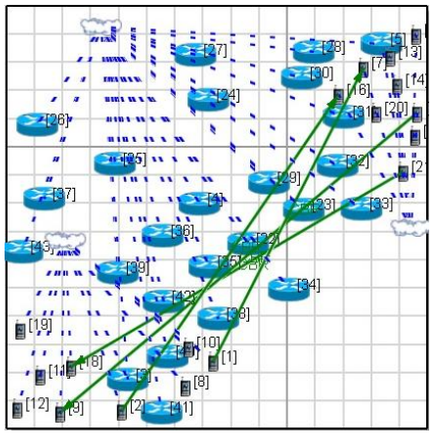
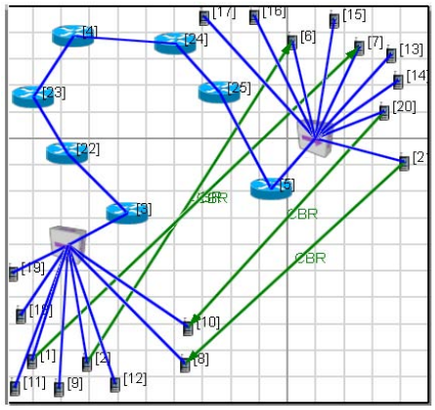


Fig. 7: Scenario for Wired Network



In this scenario we have three subnets out of which 2 subnets are IPv6 only and one is IPv4 only subnet. Here Router 3 and Router 5 are dual stack routers on each IPv6 subnet. Here an IPv6 host wants to communicate to another IPv6 host via an IPv4 backbone. To enable this communication, a wireless tunnel is created between both the dual stack routers which will carry IPv6 packet encapsulated inside an IPv4 packet. Here tunnel is a Multihop wireless tunnel, and all the intermediate routers are IPv4 only routers i.e. they understand IPv4 only packets and discard IPv6 packets. This is an end-to-end algorithm, i.e. compression and decompression is performed only at the edge routers i.e. at router 3 and router 5, and other routers forward the packets without performing compression and decompression. Here we have varied the packet size of the packet from 64 bytes to 2048 bytes.

Fig. 7 represents the scenario for wired network; here tunnel is a multihop wired tunnel. A Field configuration of 1500m x 1500m is used for the scenario. MAC protocol for wired network is 802.3. We have used 4 constant bit rate applications to generate the traffic in the network. The sending rate is 100 packets per second. Here Router 3 and Router 5 are dual stack routers on IPv6 subnet. Here an IPv6 host wants to communicate to another IPv6 host via an IPv4 backbone. To enable this communication, a wired tunnel is created between both the dual stack routers which will carry IPv6 packet encapsulated inside an IPv4 packet. Here tunnel is a multihop wired tunnel, and all the intermediate routers are IPv4 only routers i.e. they understand IPv4 only packets and discard IPv6 packets. This is an end-to-end algorithm, i.e. compression and decompression is performed only at the edge routers i.e. at router 3 and router 5, and other routers forward the packets without performing compression and decompression. Here we have varied the packet size of the packet from 64 bytes to 2048 bytes.

Results & Discussions

This section discusses about the results which are simulated on Qualnet 5.1 simulator, and analysed on the basis of varying packet size of wired and wireless network. Header compression is applied over the IPv6 header of the packet over multihop tunnel and 4 Constant Bit rate (CBR) applications are used on varying packet size of 64, 128, 256, 512, 1024, and 2048 Bytes. We have done comparisons for compressed and uncompressed networks. The metric based analysis for multihop wireless tunnel is shown in Figs. 8 to 11 and metric based analysis for multihop wired tunnel is shown in Figs. 12 to 15.

CASE-I Multihop Wireless Tunnel: Wireless Scenario

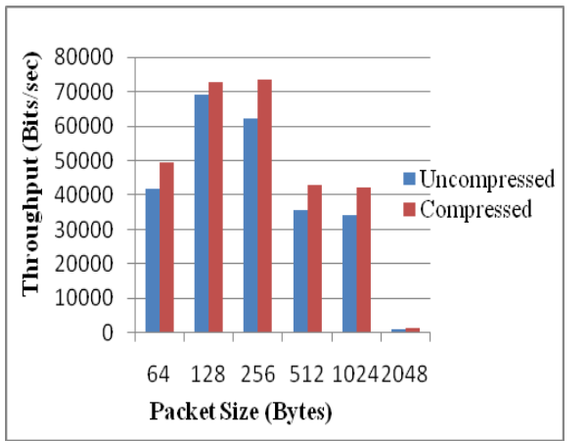
Throughput: Throughput is defined as the number of packets successfully delivered to the destination in a particular time unit. The relationship between packet size and throughput is that throughput is proportional to packet size, i.e. as packet size increases throughput increases, but for large packet sizes it is harder to deliver a packet to the intended destination because of channel conditions, and will reduce the throughput. Throughput is calculated in bits/seconds. The formula for throughput is given as

$$Throughput(T) = 8 \times \frac{(Total\ No.\ of\ bytes\ received)}{(Times\ last\ packet\ sent - time\ first\ packet\ sent)}$$

It is evident from Fig. 8 that while comparing throughput of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, throughput increases for packet size of 64 and 128 bytes, and decreases when the packet size is 256, 512, 1024 bytes. When packet size is increased to 2048 bytes there is exceptionally degradation in throughput as packet loss is more in this case. This is because of wireless network where there is always scarcity of resources. As the packet sizes increases, more

bits are transmitted over a link and hence due to congestion more packets are lost which results in reduced throughput. In this case newly modified compressed protocol gives better throughput than the standard technique, since bits transmitted are reduced, so the chances of packet drop is reduced. Statistics shows that the maximum improvement in throughput of 31% is achieved when the packet size is 2048 bytes, and significant improvement is observed with other packet sizes too.

Fig. 8: Throughput Vs Packet Size



Average End-to-end Delay: Average end-to-end delay is the time taken by a packet sent from a sender to receiver. It includes all the delays like queuing delay, propagation delay, transmission delay, processing delay etc. It is measured in seconds. The formula for delay calculation is given as:

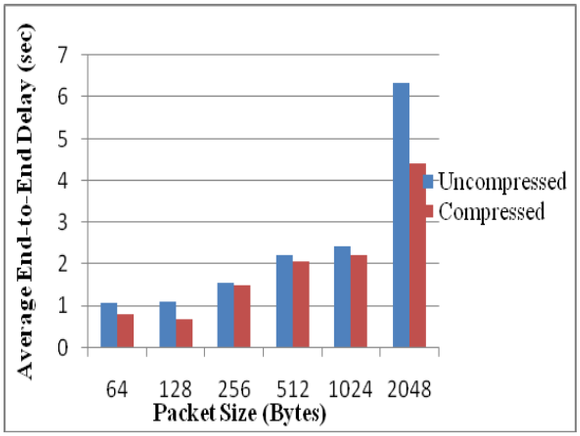
$$Average\ end-to-end\ delay = \frac{(total\ transmission\ of\ all\ received\ packets)}{Number\ of\ packets\ received}$$

where

Transmission delay of a packet = (Time packet received at server – time packet transmitted at client)

It is evident from Fig. 9 that while comparing end-to-end delay of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, end-to-end delay increases as packet size increases, due to the limited resource constraint in wireless network. Delay is exceptionally very high even for small packets since tunnel is a multihop wireless tunnel, because of multiple hops a data packet has to go through, the more time it takes to reach its destination node, and due to wireless nature resource is a constraint in wireless network. Such a high jitter is not tolerable in real network. Impact of packet size is directly proportional to delay, as packet size increases, delay increases. Table 2 shows variation of end-to-end delay with respect to varying packet sizes. Statistics shows that the maximum improvement in end-to-end delay of 38% is achieved when the packet size is 128 bytes, and significant improvement is observed with other packet sizes too.

Fig. 9: Average End-to-End Delay Vs Packet Size



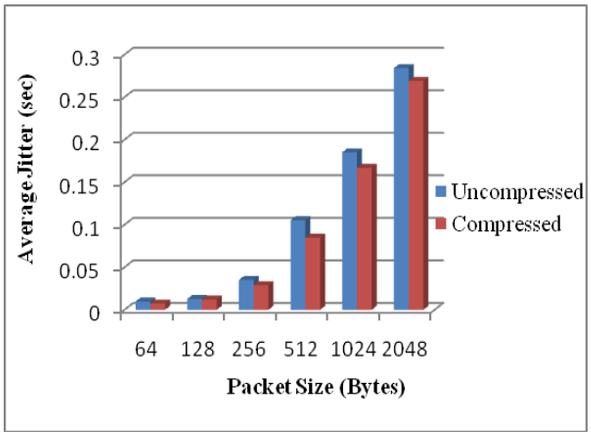
Average Jitter: Jitter can be defined as the variation between the inter arrival time between two consecutive packets. In other words sender sends the packets at a particular rate but due to channel conditions and network congestions packets arrive at the receiver at variable rate. Jitter affects the quality of application especially over real time applications. Jitter should be negligible more jitter could affect the quality of any application. Jitter can be calculated only if at least two packets have been received. The formula for Jitter calculation is given as:

$$Average\ jitter = \frac{Total\ packet\ jitter\ for\ al\ received\ packets}{(number\ of\ packets\ received - 1}$$

where

Packet Jitter = (Transmission delay of current packet – transmission delay of previous packet)

Fig. 10: Average Jitter Vs Packet Size

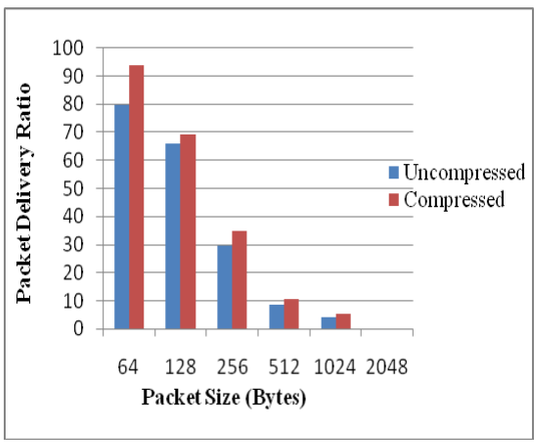


It is evident from Fig. 10 that while comparing jitter of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, jitter increases as packet size increases, and is high when packet size is 2048 bytes. Results show that jitter is very less when packet is small i.e. for 64, 128 and 256 bytes, but it significantly increases as the packet size increases to 512, 1024 and 2048 bytes. This is because tunnel is a multihop wireless tunnel and it exhibits the property of wireless network. We are experiencing less delay in case of compressed network, as we are reducing the overall size of the packet. Impact of packet size is directly proportional to jitter, as packet size increases, jitter increases. Table 3 shows variation of jitter with respect to varying packet sizes. Statistics shows that the maximum improvement in jitter of 25% is achieved when the packet size is small of 64 bytes, and significant improvement is observed with other packet sizes too.

Packet Delivery Ratio: Packet Delivery ratio is the ratio of total number of packets received at the destination to the total number of packets sent by the source. The formula for packet delivery ratio (PDR) is given as:

$$Packet\ delivery\ Ratio = \frac{(Total\ number\ of\ \ packets\ received)}{(Total\ number\ of\ \ packetssent)} \times 100$$

Fig. 11: Packet Delivery Ratio Vs Packet Size

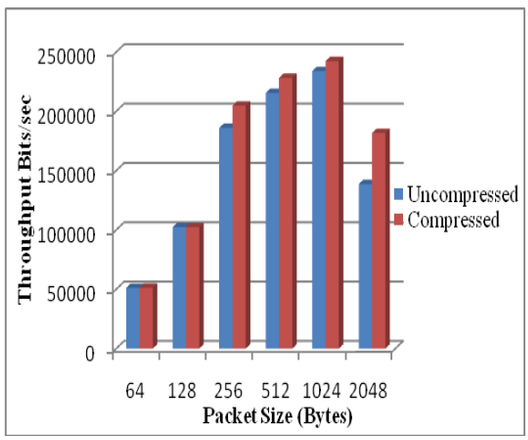


It is evident from Fig. 11 that while comparing PDR of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, PDR decreases as packet size increases. Packet delivery ratio is high for small packet size, as packet size increases PDR decreases. Here the case is of wireless networks, packet loss is high because of limited resources and power constraints. But we are getting increased PDR in case of compressed networks, which results in better resource utilisation. PDR is very less when the packet size is 512 and onwards. This is because due to large packet size, more bits are transferred over the link which results in congestion. More packets are lost in the network and hence PDR decreases. In all the cases of packet sizes, newly developed protocol gives better PDR than the standard technique. Here we can say that impact of packet size is inversely proportional to PDR, i.e. as packet size increases, PDR decreases considerably. Statistics shows that the maximum improvement in Packet delivery ratio of 34% is achieved when the packet size is 1024 bytes, and significant improvement is observed with other packet sizes too.

CASE-II Multihop Wired Tunnel: Wired Scenario

Throughput: Throughput is the average rate of successful packet delivery over a communication channel. It is very important metric since it shows the overall performance of the network. The comparison of performance for throughput of the uncompressed and compressed network is shown in Fig. 12. It is evident from figure that while comparing throughput of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, throughput increases for packet size of 64 and 128 bytes, and decreases when the packet size is 256, 512, 1024 bytes. When packet size is increased to 2048 bytes there is exceptionally degradation in throughput as packet loss is more in this case. This is because of wireless network where there is always scarcity of resources. As the packet sizes increases, more bits are transmitted over a link and hence due to congestion more packets are lost which results in reduced throughput. In this case newly modified compressed protocol gives better throughput than the standard technique, since bits transmitted are reduced, so the chances of packet drop is reduced. Statistics shows that the maximum improvement in throughput of 31% is achieved when the packet size is 2048 bytes, and significant improvement is observed with other packet sizes too.

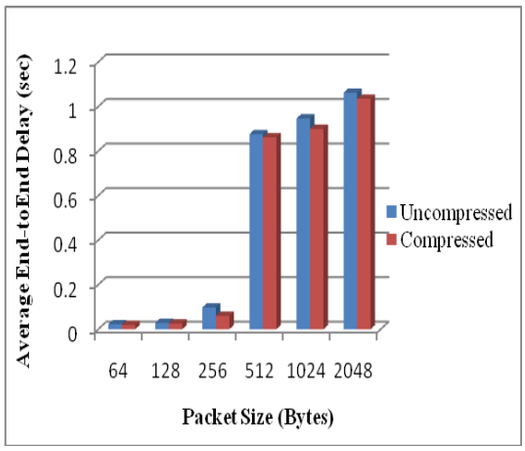
Fig. 12: Throughput Vs Packet Size



Average End-to-End Delay: Average end-to-end delay includes all possible delays from the moment the packet is

generated to the moment it is received by the destination node. Generally, there are three factors affecting end-to-end delay of a packet: Route discovery time, Buffering waiting time, the length of routing path. The comparison of performance for end-to-end delay of the uncompressed and compressed network is shown in Fig. 13. It is evident from the figure that while comparing end-to-end delay of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, delay increases as the packet size increases. End-to-end delay less for small packet sizes of 64, 128, and 256 bytes, and increases significantly with packet size 512, 1024 and 2048 bytes. This is because tunnel is a multihop wired tunnel, and delay is less in wired network as compared to wireless. We are getting a dedicated link for transmission between the hosts and routers. Bandwidth is not a constraint in wired network. But still we are experiencing less delay in case of uncompressed networks due to reduced number of bits transmitted over the link, it takes less time for processing a packet. Table 6 shows variation of end-to-end delay with respect to varying packet sizes. Statistics shows that the maximum improvement in end-to-end delay of 38% is achieved when the packet size is 256 bytes, and significant improvement is observed with other packet sizes too.

Fig. 13: Average End-to-End Delay Vs Packet Size



Average Jitter: Jitter is the variation in the inter arrival time between two consecutive packets. More the value of jitter, affects the quality of the application. Jitter is a serious problem in real time communication like IP telephony and videoconferencing. The comparison of performance for jitter of the uncompressed and compressed network is shown in Fig. 14. It is evident from the figure that while comparing jitter of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, jitter increases as packet size increases, even for large packet sizes of 2048 bytes jitter is very less. Jitter is almost negligible here in every case because of wired network, tunnel is a multihop wired tunnel and exhibits the property of wired network, still we are getting better results in case of compressed network. Table 7 shows variation of jitter with respect to varying packet sizes. Statistics shows that the maximum improvement in jitter of 20% is achieved when the packet size is 256 bytes, and significant improvement is observed with other packet sizes too.

Fig. 14: Average Jitter Vs Packet Size

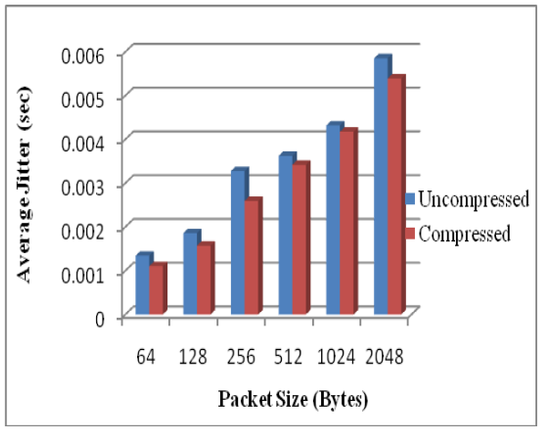
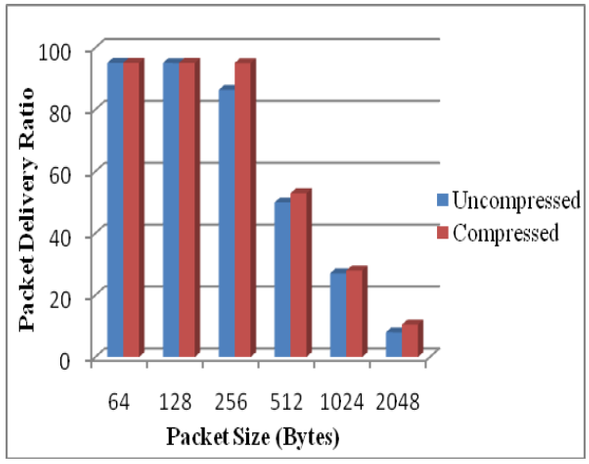


Fig. 15: Packet Delivery Ratio Vs Packet Size



Packet Delivery Ratio: Packet delivery ratio is defined as ratio of the data packets delivered at the destination to those generated by the CBR sources. Packet delivery ratio is very important metric since it shows the loss rate, which in turn affects the maximum throughput of the network. The comparison of performance for PDR of the uncompressed and compressed network is shown in Fig. 15. It is evident from the figure that while comparing PDR of uncompressed and compressed network with varying packet sizes of 64, 128, 256, 512, 1024, and 2048 bytes, PDR is high for small packets, as packet size increases PDR decreases, this is because for big packets more number of bits are transmitted over the link, which results in congestion. As a result more packets are dropped which degrades the PDR, which affects the other parameters. In all the cases of packet sizes, newly

developed protocol gives better PDR than the standard technique, this is due to reduced number of bits transmitted which results in improved packet loss. Here we can say that impact of packet size is inversely proportional to PDR, i.e. as packet size increases, PDR decreases considerably. Table 8 shows variation of PDR with respect to varying packet sizes. Statistics shows that the maximum improvement in PDR of 31% is achieved when the packet size is 2048 bytes, and significant improvement is observed with other packet sizes too.

Conclusion

Tunneling is the most widely used solution for enabling the smooth interoperation between IPv4 and IPv6 protocols. Tunneling comes with several shortcomings like header overhead which results in high bandwidth usage, packet reordering, inefficient routing etc. In this paper we are improving the efficiency IPv6 tunneling mechanism by using IPv6 header compression, as IPv6 is of largest length of 40 bytes. In this paper we have proposed a novel approach for IPv6 header compression over IPv6 tunneling mechanism for multihop tunnel. Using this approach we have compressed 40 bytes of IPv6 header up to 6 bytes. Here we have discussed the results for multihop wired and wireless tunnel over varying packet sizes. We have compared the compressed network with uncompressed network. Simulations are carried out over Qualnet 5.1 simulator. Four parameters are taken into consideration: Throughput, Average end-to-end delay, Average Jitter, and Packet delivery ratio. Results show that compressed network performs better in every case than uncompressed tunneling whatever the network is either wired or wireless. IPv6 header compression results in better bandwidth utilisation by reducing the header overhead which improves the overall performance of the network. Currently the profile specified is IPv6 only profile, in future IPv6/TCP and IPv6/UDP will be added to our work.

References

Chauhan, D., & Sharma, S. (2014). A Survey on next generation internet protocol: IPv6. *International Journal of Electronics & Industrial Engineering (IJEEE),* 2(2), 125-128.

Chauhan, D., & Sharma, S. (2015). Addressing the bandwidth issue in end-to-end header compression over ipv6 tunneling mechanism. *International Journal of Computer Network and Information Security*, 9(5), 39-45

Dimitriadis, G., Karapantazis, S., & Pavlidou, F. (2007). *Comparison of header compression schemes over satellite links*. Proceedings International Workshop on IP Networking over Next-generation Satellite Systems (INNSS'07), Budapest, Hungary. 2007.

Effnet, A. B. (2004a). An introduction to IP header compression. Retrieved from http://www.effnet.com.

Effnet, A. B. (2004b). *The concept of robust header compression, ROHC*. Bromma, White Paper.

Fortuna, P. & Ricardo, M. (2009) Header compressed VoIP in IEEE 802.11. *Wireless Communications*, 16(3), 69-75.

Hinden, R., & Deering, S. (1998). *Internet Protocol Version 6 (IPv6) Addressing Architecture*. RFC 3513. Retrieved from http://tools.ietf.org/html/rfc3513

ICNS. (2012). *Impacts of IPv6 on Robust Header Compression in LTE Mobile Networks*. The 8th International Conference on Networking and Services ICNS 2012.

Infographic (n.d.), World IPv6 Launch Retrieved from http://www.worldipv6launch.org/infographic/

Jung, S., & Hong, S. (2009). Network/hardware cross-layer evaluation for ROHC and packet aggregation on wireless mesh networks. *Wireless Networks*, 15(8), 1086-1101.

Qualnet 5.1 User's Guide, Scalable Network Technologies. Retrieved from http://web.scalable-networks.com/content/ qualnet.

Raicu, I., & Zeadally, S. (2003). *Evaluating IPv4 to IPv6 transition mechanisms*. 10th International Conference on Telecommunications, 2, pp. 1091-1098.

Rawat, P., Bonnin, J. M., & Toutain, L. (2008). *Designing a tunneling header compression (TuCP) for tunneling over IP*. IEEE International Symposium on Wireless Communication Systems. 2008. ISWCS'08.

Rawat, P., Bonnin, J. M., Minaburo, A., & Toutain, L. (2007). An End-2-End Tunnel Header Compression Solution for Nested Mobile Networks. *International Conference on the Latest Advances in Networks*, ICLAN'2007, Paris, France, December.

Rey, M.D. (1981). *California 90291 Internet Protocol, Darpa Internet Program, Protocol Specification*. RFC 791.

Tatiana, K., Frank, M., Fitzek, H. P., Nethi, S., Arildsen, T., & Perrucci, G. P. (2006). *Novel IP Header Compression Technique for Wireless Technologies with Fixed Link Layer Packet Types*. IEEE GLOBECOM 2006, (pp. 1-5).

Wu, P. (2013). *Transition from IPv4 to IPv6: A state-of-the-art survey*. Communications Surveys & Tutorials, 15(3), 1407-1424.

Yoon, Y., Park, S., Lee, H., Kim, J. S., Jee, S. B. (2011). Header Compression Method and Its Performance for IP over Tactical Data Link. *International Journal of Energy, Information & Communications*, 2(2), 61-77.