Optimization Of IPv4 Packet's Headers

Fahim A. Ahmed Ghanem[1], Vilas M. Thakare[2]

[1]Research Student, School of Computational Sciences,
Swami Ramanand Teerth Marathwada University, Nanded, India,

[2]Professor and Head of Computer Science, Faculty of Engineering & Technology
Post Graduate Department Of Computer Science, SGB, Amravati University, Amravati.

ABSTRACT

This paper aims to optimize the current IP packet structure by reducing the size of the headers, increasing the size of the data, reducing packet processing time and speed up the data transmission.

KEYWORDS: TCP/IP, UDP, IPv4, Ethernet Frame.

1. Introduction

The current IPv4 Packet's scheme has several fields to be carried and transmitted in each packet even if the source, the destination and the payload of the packets are same. These fields utilize packet space and thus decrease the size of user data. Therefore the processing of each packet requires more time and delay of end-to-end transmission. While data transmission, data has to be broken down into similar structures of the packets, which are reassembled to the original data chunk once it reaches the destination. A packet is also called a datagram, a segment, a block, a cell or a frame, depending on the protocol.

2. Objective Of This Study

This paper aims to optimize data transmission by redesigning OSI packet structure by dividing packets into master packet and slave packet, decreasing the size of packet's header and increase the size of the data while maintaining the same current packet size. This optimization aims to reduce the processing time as a result of reducing the contents of processed headers.

3. OSI Model Layers

Before we start with packet optimization, it will be helpful to have a quick review of the OSI model packet's construction. In this section we will have a look through OSI model. There are two types of header fields:

a. Variable Fields: fields that it is information from packet to packet and has to be repeated in each packet.
b. Invariable Fields (Tagged with Inv.): field that it is information doesn't vary from packet to other and doesn't have to be repeated in each packet.

Open Systems Interconnection (OSI) model (ISO/IEC 7498-1) is a product of the Open Systems Interconnection effort at the International organization for Standardization. It is a prescription of characterizing and standardizing the functions of a communications system in terms of abstraction. Similar communication functions are grouped into logical layers. A layer serves the layer above it and is served by the layer below it.

There are seven interconnection layers start from up to down (layer 7 to layer 1) prospectively:

Application forwards user data to Presentation layer which forwards it to Session layer. Session layer forwards data user to Transportation layer. Transport layer adds the header and sent the segment to Network layer which adds header and forward the packet to Data link layer. Data link adds the header and forward the frame to Physical layer.

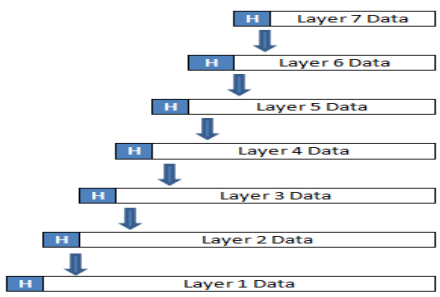

Fig. 1 OSI 7 layers' hierarchy

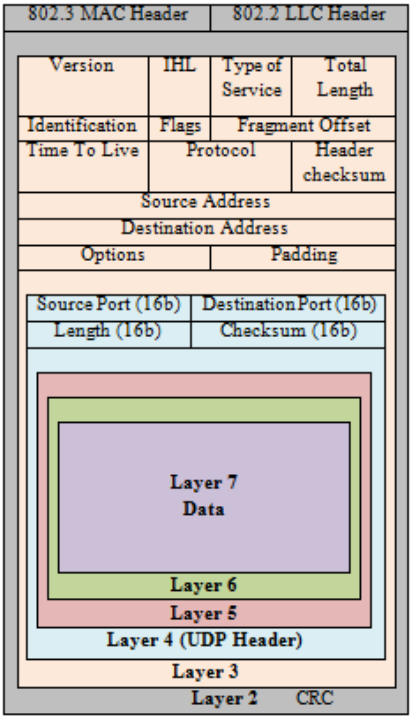

Fig. 2 full frame (UDP)

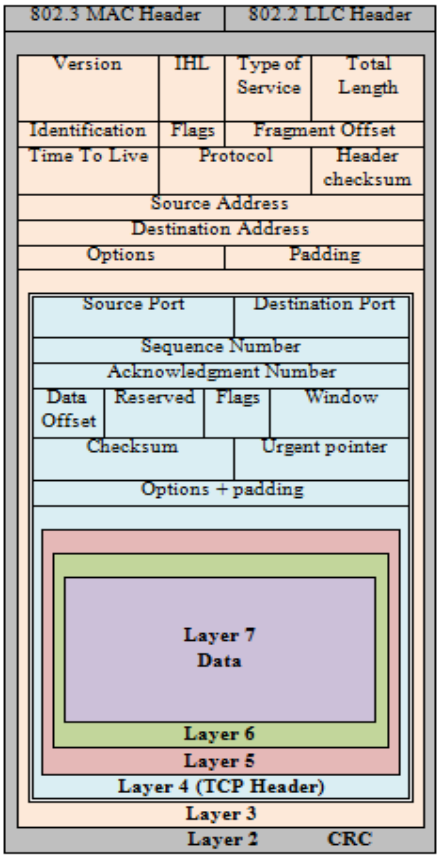

Fig. 3 full frame (TCP)

In this paper, we will concentrate on layer 4, layer 3 and layer 2 headers. We will ignore layer 7, layer 6 and layer 5 headers as it is variables from application to other.

3.1 Transport Layer Headers

Transport layer contains either TCP or UDP headers:

3.1.1 TCP Header

The Transmission Control Protocol (TCP) is one of the core protocols of the Internet Protocol Suite provides reliable, ordered delivery of a stream of bytes from a program on one computer to another program on another computer. TCP is the protocol that major Internet applications such as the World Wide Web, email, remote administration and file transfer rely on.

Below description displays the contents of TCP header contents:

a. Source Port (inv.): the source port number.
b. Destination Port (inv.): the destination port number.
c. Sequence Number: the sequence number of the first data octet in this segment.
d. Acknowledgment Number: if the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive.
e. Data offset (inv.): this indicates where the data begins.
f. Reserved (inv.): reserved for future use. Must be zero.
g. Control Bits:
   1 URG: Urgent Pointer field significant
   2 ACK: Acknowledgment field significant
   3 PSH: Push Function
   4 RST: Reset the connection
   5 SYN: Synchronize sequence numbers
   6 FIN: No more data from sender
   7 Window: the number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.
h. Checksum: the checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text.

i. Urgent Pointer (inv.): this field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment.
j. Options (inv.): variable, Options may occupy space at the end of the TCP header and are a multiple of 8 bits in length.
k. Padding (inv.): variable, the TCP header padding is used to ensure that the TCP header ends.

TCP header varies from 20 bytes up to 60 bytes.

3.1.2 UDP Header

UDP is a connectionless protocol that provides an unreliable data services [1]. UDP is one of the core members of the Internet Protocol Suite, the set of network protocols used for the Internet. With UDP, computer applications can send messages to other hosts on an Internet Protocol (IP) network without requiring prior communications to set up special transmission channels or data paths. UDP applications must generally be willing to accept some loss, errors or duplication. Some applications such as TFTP may add rudimentary reliability mechanisms into the application layer as needed [2]. UDP provides application multiplexing (via port numbers) and integrity verification (via checksum) of the header and payload [3]**.** UDP is suitable for purposes where error checking and correction is either not necessary or performed in the application, avoiding the overhead of such processing at the network interface level [4]**.**

Below description displays the contents of UDP header contents:

a. Source port (inv.): It identifies the sender's port.
b. Destination port (inv.): it identifies the receiver's port.
c. Length, it specifies the length in bytes of the entire datagram: header and data.
d. Checksum: It is used for error-checking of the header and data. If no checksum is generated by the transmitter, the field uses the value all-zeros [5]**.**

UDP header varies from 4 to 8 bytes.

3.2 Network Layer Header

Network layer header consists of 13 fields, which 12 are required, the 13th field is optional. Network layer header size is varies from 20 bytes to 60 bytes.

Below description explains the contents of network layer header contents:

a. Version (inv.): indicates the version of IPv4 currently used.
b. IPv4 Header Length (IHL): indicates the datagram header length in 32-bit words
c. Type-of-Service (inv.): specifies how an upper-layer protocol would like a current datagram to be handled, and assigns datagrams various levels of importance.
d. Total Length: specifies the length, in bytes, of the entire IPv4 packet, including the data and header.
e. Identification: contains an integer that identifies the current datagram.
f. Flags (inv.): Consists of a 3-bit field of which the two low-order (least-significant) bits control fragmentation. The low-order bit specifies whether the packet can be fragmented. The middle bit specifies whether the packet is the last fragment in a series of fragmented packets. The third or high-order bit is not used.
g. Fragment Offset (inv.): indicates the position of the fragment's data relative to the beginning of the data in the original datagram, which allows the destination IPv4 process to properly reconstruct the original datagram.
h. Time-to-Live (inv.): maintains a counter that gradually decrements down to zero, at which point the datagram is discarded. This keeps packets from looping endlessly.
i. Protocol (inv.): indicates which upper-layer protocol receives incoming packets after IPv4 processing is complete.
j. Header Checksum: helps ensure IPv4 header integrity.
k. Source Address (inv.): specifies the sending node.
l. Destination Address (inv.): specifies the receiving node.
m. Options (inv.): Allows IPv4 to support various options, such as security.

The minimum IPv4 header is 20 byte and the maximum is 40 byte.

3.3 Data Link Header and Tailor (802.3 IEEE)

Data link frame header and Tailor consist of 4 fields. It is chunk of data that is packaged for transmission over network [6].

Ethernet II framing (also known as DIX Ethernet) named after DEC, Intel and Xerox, the major participants in its design. Below description display the contents of Ethernet header:

a. Destination Address (Inv.): It specifies either a single recipient node (unicast mode), a group of recipient nodes (multicast mode), or the set of all recipient nodes (broadcast mode).
b. Source Address (Inv.): sender's globally unique node address. This may be used by the network layer protocol to identify the sender, but usually other mechanisms are used (e.g. arp). Its main function is to allow address learning which may be used to configure the filter tables in a bridge.
c. LLC Header (Inv.): it contains DSAP, SSAP and control byte.
d. CRC: this field is added at the end of the frame (trailer) and provides error detection in the case where line errors (or transmission collisions in Ethernet) result in corruption of the MAC frame. Any frame with an invalid CRC is discarded by the MAC receiver without further processing. The MAC protocol does not provide any indication that a frame has been discarded due to an invalid CRC.

The size of Ethernet header along with CRC is 21 bytes.

As per the above overview of OSI models and based on the traffic type, Ethernet frame can contains two types of segment headers: TCP header or UDP header. The accumulated headers within Ethernet frame (TCP header + L3 header + L2 header and tailor) varies from 81bytes to 121 bytes.

In sample case, if we have a TCP/IP sentence contains of 50,000 bytes, this sentence will be divided into small chunks called frames and the maximum size of each frame will not exceed 1518 bytes. Headers will consume 121 byte out of 1518 and the remaining 1379 bytes will be available for data, so 50,000 bytes will be transmitted over 37 packets as a result of dividing 50,000 by 1379.

The accumulated headers within Ethernet frame (UDP header + L3 header + L2 header and tailor) varies from 45 bytes to 69 bytes.

The maximum size of Ethernet packet will not exceed 1518 bytes. Headers consume 69 bytes out of 1518 and the remaining 1449 bytes will be available for data. According to the above data, we observe that we are facing two obstacles: first one is the size of the headers and the other one is the processing of those headers. Headers consume some packet's size and most of the data in the header sections is repeated in all packets and it is processed in each middle device till it reaches the destination.

4. Steps Of Optimization

Optimization steps are divided into two stages:

1- Headers Optimization:
   It contains creation of master and slave packet.
2- Packet Transmission:
   Packet's processing divided into two steps
   a. Processing of Master packet.
   b. Processing of Slave packet.

4.1 Headers Optimization

4.1.1 Proposed Master Packet

The proposed Master packet contains all header fields (Variables, Invariable, along with ID and Tag Fields) but without user data field.

4.1.1.1 Transport Layer Master Segment

The proposed transport layer Master segment contains all Transport layer header fields (Variables, Invariable, along with ID and Tag Fields) but without user data field.

4.1.1.1.1 TCP Master Segment

The proposed TCP Master segment is created when transport layer receives first packet in the session from session layer in order to be transmitted to the desired destination.

The contents of Master TCP segment are same contents of OSI TCP header with extra two fields:

a. ID: 1 bit indicates the type of the packet, as value 1 indicates Master segment and value 0 indicate slave packet.
b. Tag: 31 bits, is a unique ID for each session, the

value of tag will be maintained same till session is closed.

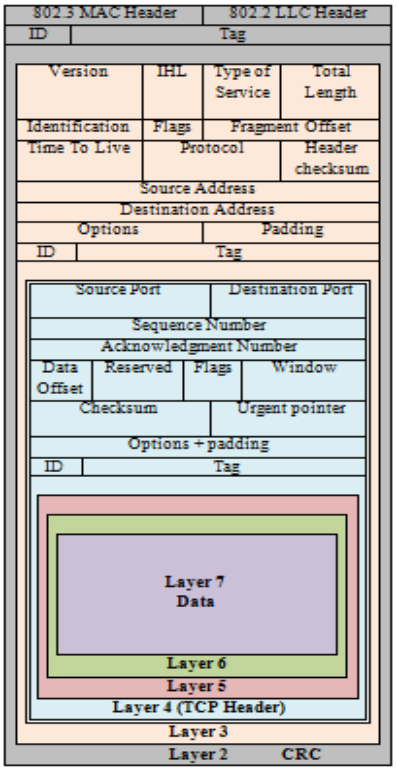

Fig. 4 Proposed full master TCP frame

4.1.1.1.2 UDP Master Segment

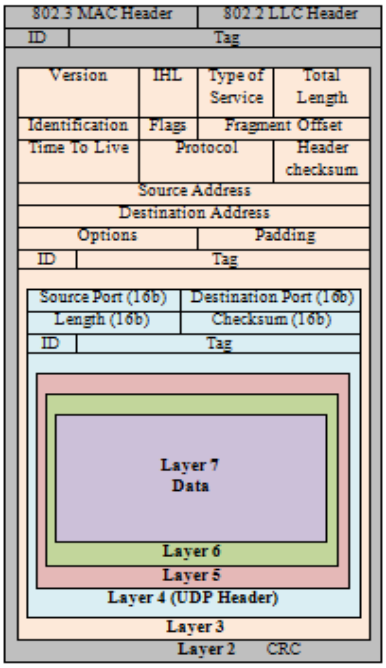

Fig. 5 Proposed full master UDP frame

The proposed UDP Master segment is created when transport layer receives first packet for each session from session layer in order to be transmitted to the desired destination. The contents of Master UDP segment are same as the contents of OSI UDP header with extra ID and Tag fields. The roles of these two fields are same as the role of TCP master segment.

4.1.1.2 Network Layer Master Packet

The proposed Network layer header is added to the segment received from transport layer. The contents of Master packet are same the contents of network layer header of OSI model with extra ID and Tag fields. The roles of these two fields are same as the role of transport layer for Master segment.

4.1.1.3 Data Link Master Frame

The proposed Ethernet header is added to the packet received from network layer. The contents of Master frame header are same the contents of Ethernet header of OSI model with extra ID and Tag fields. The roles of these two fields are same as its role in transport layer Master segment.

The size of Master Frame (TCP header) is the size of the layer 4, layer 3, layer 2 headers and tailor only without user Data which varies from 96 bytes (without optional fields) to 133 bytes (with optional fields). The size of Master (UDP headed) is the size of the layer 4, layer 3, layer 2 headers and tailor only without Data which varies from 57 bytes to 81 bytes.

4.1.2 Proposed Slave Packet

Slave packet contains only invariable header fields in addition to ID and Tag fields along with user data.

4.1.2.1 Transport Layer Slave Segment

The proposed transport layer packet contains invariable header fields in addition to ID and Tag fields along with user data.

4.1.2.1.1 TCP Slave Segment
The size of the header is 134 bits equals to 17 bytes.

4.1.2.1.2 UDP Slave packet

The proposed UDP packet header contains only invariable header fields in addition to ID and Tag Fields along with data. The size of the header is 64 bits equals to 8 bytes.

4.1.2.2 Network Layer Slave packet

The proposed layer 3 header contains only invariable header fields in addition to ID and Tag Fields along with user data. The size of the layer 3 header is 84 bits equals to 11 bytes.

4.1.2.3 Ethernet Slave Frame

The proposed layer 2 Frame contains only invariable header and Tailor fields in addition to ID and Tag Fields along with user data. The size of Ethernet header and tailor is 46 bits equals to 6 bytes.

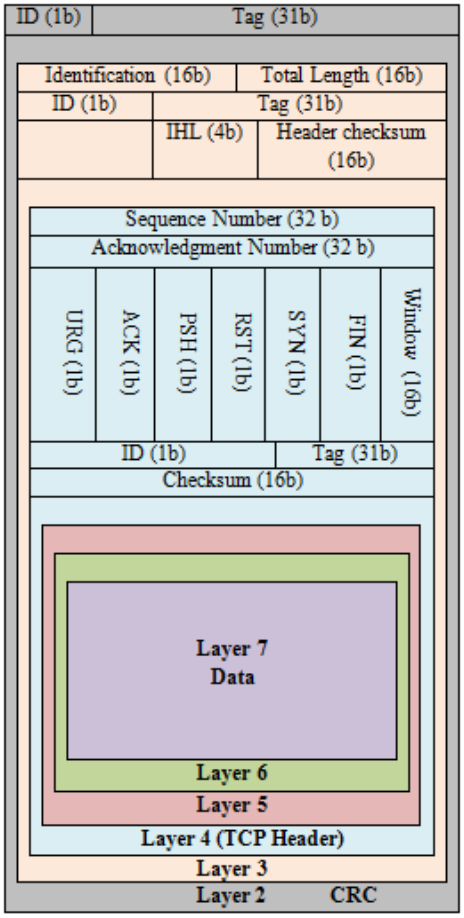

Fig. 6 Proposed full slave TCP frame

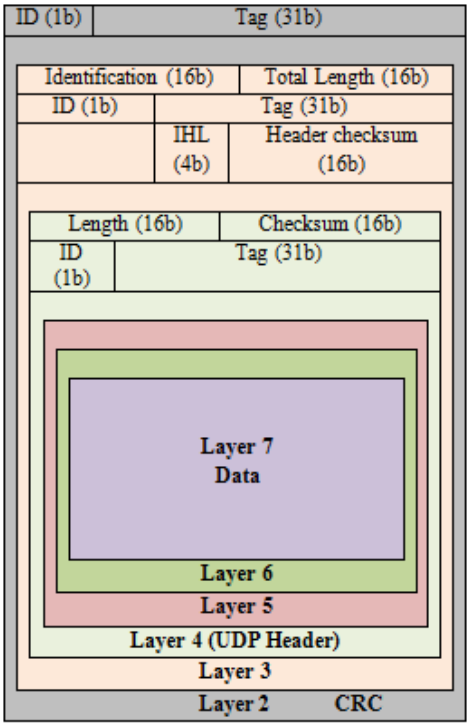

Fig. 7 Proposed full slave UDP frame

4.2 Packet Transmission

Packet transmission includes the process of transmission of master and slave packets from source to destination.

4.2.1 Master Packet's Transmission

4.2.1.1 Master TCP Segment, Packet and Frame Transmission

a. Application layer of the source station forwards user data to Presentation which forwards it to Session which also forwards it to transport layer.
b. Transport layer buffers the received user data, creates one master TCP segment and forward it to network layer.
c. Network layer adds Network layer master header to the received segment and send the packet to Ethernet layer.
d. Ethernet layer adds Ethernet master header and tailor and sends the frame to physical layer.
e. Physical layer adds it is header sends it to another station or middle devices towards destination.

f. Ethernet layer in layer 2 device checks the packet ID and identify whether it is Master packet or Slave packet by value of ID (1= master and 0 = slave).
g. If the value ID is 1, Ethernet layer processes Master packet as the processing of OSI model packet based on source, destination and content etc. and decide the next hop MAC.
h. Ethernet layer cashes the frame processing result, save it in the forwarding table along with frame's Tag and send the frame towards the destination.
i. If the next hop is layer 2, steps *f, g, and h* will be repeated.
j. If the next hop is layer 3 device, layer 3 checks the packet ID and identify whether it is Master Packet or Slave Packet by value of ID (1= master and 0 = slave).
k. If value ID is 1, layer 3, layer 3 processes master packet as OSI model processing based on source, destination and content etc. and decide the next hop IP.
l. Layers 3 cashes the processing result, save it in the forwarding table along with packet's Tag and forwards the packet towards the destination.
m. If the next hop is layer 3, steps *j, k,* and *l* will be repeated.
n. If the next hop is layer 4, layer 4 processes the segment as per layer 2 and layer 3 and forwards it to next hop or layer.
o. Steps f – n will be repeated till packet reaches the destination.
p. Once Transport layer of source station receives the acknowledgement about receiving Master packet from the destination, it starts creating and sending slave segments.
Note: master TCP packet was sent only one time per each session.

4.2.1.2 Master UDP Segment, Packet and Frame Transmission

Transmission of master UDP packet takes steps from *a - n* of TCP packet but not step o, because UDP packet doesn't rely on acknowledgment.

Note: master UDP packet being transmitted every 5 seconds in order to insure continuously delivering of packets in case there is any transmission or processing issue in the middle devices.

4.2.2 Slave Packet's Transmission

4.2.2.1 Slave TCP Segment, Packet and Frame Transmission

a. Source transport layer creates either TCP or UDP slave header and sends the segment to Network layer.
b. Network layer adds it is slave header and sends the packet to Ethernet layer.
c. Ethernet layer adds it is slave header and tailor and sends it to physical layer.
d. Physical layer adds it is header and forwards it towards the destination.
e. Ethernet layer of the destination station or middle devices checks the packet ID and identify whether it is Master Frame or Slave Frame by value ID (1= master and 0 = slave).
f. If the value ID is 0, layer 2 matches the Tag with forwarding table information and forwards it to the next hope without any extra processing.
g. If there is no information about the Tag in the forwarding table, layer 2 simply drops the packet.
h. If the next hope is layer 2, step *e* and *f* will be repeated.
i. If the next hope is layer 3 device of the destination station, layer 3 will check the packet ID and identify whether it is Master Frame or Slave Frame by value ID (1= master and 0 = slave).
j. If the value ID is 0, layer 3 matches packet Tag with forwarding table information and forwards it to the next hope without any extra processing.
k. If there is no information about the Tag in the forwarding table, layer 3 simply drops the packet.
l. If the next hope is layer 3, step *i* and *j* will be repeated.
m. If the next hop is layer 4, layer 4 processes the segment like layer 2 and layer 3 and forwards it to next layer.
n. Steps *e* to *m* were repeated till packet reaches the destination.
o. Packet acknowledgment was sent as OSI model.

4.2.2.2 Slave UDP Segment, Packet and Segment Transmission:

Transmission of slave UDP segment, packet and frame takes is same as steps from *a - n* of TCP packet

but not step *o*, because UDP packet doesn't rely on acknowledgment.

Note: master UDP packet was sent every 5 seconds in order to insure continuously delivering of the packets in case there is any transmission or processing issue in the middle devices.

Slave Ethernet packet (TCP header + L3 header + L2 header and tailor) is fixed to 282 bits equals to 36 bytes whether master packet contains optional fields or not.

Slave Ethernet packet (UDP header + L3 header + L2 header and tailor) is fixed to 212 bits equals to 7 bytes whether master packet contains optional fields or not.

5. Verification

Optimization of Packet header and packet transmission optimization gives ability to transmit more user data along with small headers and less processing time which increase the efficiency of the existing and feature network infrastructure.

Below two tables and four charts illustrate the comparison between ISO model and new proposed optimized packet where we observe the decreased size of Ethernet packet headers using TCP layer 4 and using UDP layer 4. The total size of Ethernet headers for optimized Ethernet (TCP header + IP header + Ethernet header and tailor) varies from 28.09% to 55.73% of OSI headers. The total size of Ethernet headers for optimized Ethernet (UDP header + IP header + Ethernet header and tailor) varies from 36.2 till 55.55% of OSI headers.

Table 1: comparison between OSI headers (TCP) and optimized packet headers

	IOS (TCP) without optional	Optimized	IOS (TCP) without optional	Optimized
TCP Header	20	17	60	17
IP Header	20	11	40	11
Frame Header	21	6	21	6
Total headers (bytes)	61	34	121	34

Table 2: comparison between OSI headers (UDP) and optimized packet headers

	IOS (TCP) without optional	Optimized	IOS (TCP) without optional	Optimized
TCP Header	4	8	8	8
IP Header	20	11	40	11
Frame Header	21	6	21	6
Total headers (bytes)	45	25	69	25

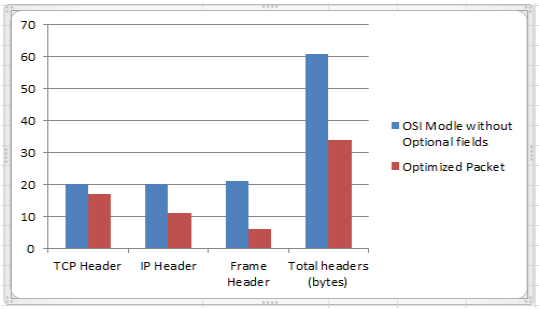

Fig. 8 comparison chart between OSI headers (TCP) without optional fields and optimized packet headers

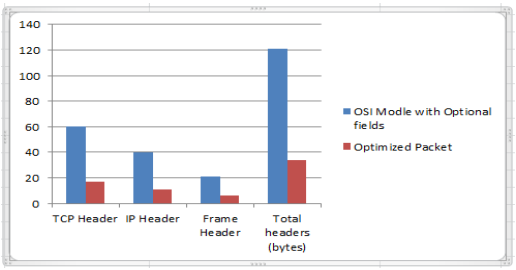

Fig. 9 comparison chart between OSI headers (TCP) with optional fields and optimized packet headers

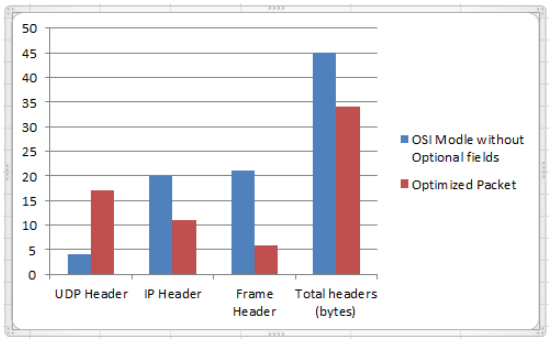

Fig. 10 comparison chart between OSI headers (UDP) without optional fields and optimized packet headers

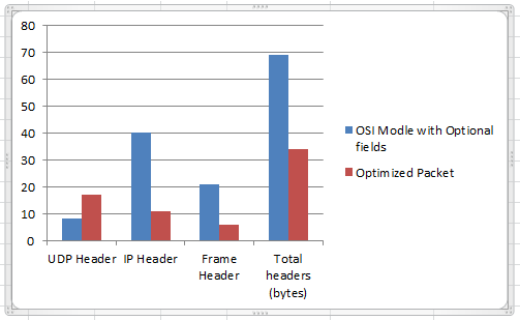

Fig. 11 comparison chart between OSI headers (UDP) with optional fields and optimized packet headers

6. Conclusion

Referring to figures 8 through 11, the new proposed IP packet structure and IP packet transmission decreases the size of headers between 44% up to 71%. Decreasing the headers' space give an opportunity to increase the size of user data while maintaining same frame size. Moreover, reducing the size of the headers will reduce the processing times as middle devices or layers will only process Master packet, cash the processing result and apply the result to all Slaves packets which belong to same session.
The proposed optimization solution can be applied to the current and new network infrastructure in order to increase the efficiency of the networks.

7. References

[1] Michael A. Gallo, William M. Hancock, "Computer Communications and Networking Technologies", New Delhi, India, Cengage Learning India Private Limited.

[2] Forouzan, B.A. "TCP/IP: Protocol Suite, 1st edition", New Delhi, India, Tata McGraw-Hill Publishing Company Ltd.

[3] Clark, M.P., "Data Networks IP and the Internet, 1st edition", West Sussex, England, John Wiley & Sons Ltd.

[4] Andrew S. Tanenbaum, "Computer Networks 4th Edition", Amsterdam, the Netherland, Prentice Hall PTR.

[5] Barrie Sosinsky, "Networking", New Delhi, India Wily India Pvt. Ltd. 2010.

[6] Kurose, J. F., Ross, K. W. "Computer Networking: A Top-Down Approach (5th ed.)", Boston, MA, Pearson Education. ISBN 978-0-13-136548-3

Mr. Fahim A. Ahmed Ghanem is a Ph.D. student at School of Computational Sciences, SWAMI RAMANAND TEERTH MARATHWADA UNIVERSITY, Nanded, India. He is doing research in field of IP Network Optimization. He received MSC-IT degree from SMU, India 2005 – 2007. He is a dual CCIE R&S and SP. He works as a Senior Network Engineer in mobily co. KSA.

(Supervisor) Mr. Vilas Mahadeorao Thakare, has received his M.E. degree in field of Advance Electronics, received his M.Sc. degree in field of Applied Electronics, Diploma in Computer Management. He was obtained his PhD in field of Computer Science.