MAJOR IMPROVEMENTS IN TCP PERFORMANCE OVER SATELLITE AND RADIO

Dr. Lawrence G. Roberts
CEO, Anagran Inc.

This research is funded in part by DARPA contract # N66001-5-9-8904

ABSTRACT

TCP performance deteriorates rapidly with packet loss and delay. The response to packet loss is inherent in TCP's design so as relieve congestion in the network but if the packet loss is due to channel noise it is an incorrect response. Also, the reduced throughput due to large delays is an unfortunate and unfair side effect of the TCP process. If one can determine the highest rate that does not cause congestion, then TCP could be sent without any substantial throughput degradation. This paper introduces a new signaling message, exchanged as a flow is setup, which allows the network to specify the highest rate it can support for the flow, thereby allowing TCP to proceed immediately (skipping slow-start) at this rate. Since the network will tell the source if it needs to slow down with a message, there is no need for packet discards and modest rates of lost packets can be considered to be caused by noise and thus retransmitted without slowing down. This protocol has been standardized by the Telecommunications Industry Association as TIA 1039 and is specified to operate with both IPv4 and IPv6 (with encryption). The paper describes the operation and shows the major improvements possible.

FLOW MANAGEMENT

When I started the prototype Internet, the ARPANET, in 1969, memory was so expensive that there was no hope of a router keeping track of any information about the ongoing flows. Thus, the first routers only looked at the packet destination and ignored any flow information. A flow is the sequence of packets that form one communication between two entities, a message, a web access, a phone call, etc. In IP each flow is uniquely defined by the 5-tupple, source and destination address and ports plus the protocol. Thus it is easy to identify a flow with an exact match of these fields. However, memory continued to be too expensive to keep flow information for 20-30 years so the practice of just routing individual packets became ingrained as the only economic possibility.

Now however, memory has become so inexpensive that it is not only economic to keep lots of information on every active flow; its QoS requirements, rate, delay, loss, precedence, route and much more. When this is done, it is also possible to reduce the cost and complexity of the router because the route, QoS, DOS, and other processing need not be redone for each packet, the first packet tells the whole story. Thus, a class of IP routers called Flow Routers has developed where flow information is maintained which allows the flow router to manage the QoS of each individual flow far better than a simple 6 bit DiffServ code in each packet can describe. For (UDP) streaming media like VoIP and IPTV, the correct strategy is to totally reject the lowest priority flow if too many attempt to fill a channel. This is called Call Acceptance Control (CAC) much the same as in the telephone network. The other type of flow, TCP, allows the network to control its rate so that the network can adapt to increased traffic. Today the network achieves this control by discarding packets if there is too much traffic and the computer sending the TCP flow then adapts by slowing down its flow rate. TCP has been the dominant traffic over the last 30 years and its ability to adapt to network conditions has been critical for the stability of the network. However, TCP was designed 30 years ago for 9.6-56 Kb modems over a low error rate network and now with broadband access and error-prone radio links, TCP is far from optimal. As the network gains capability to measure and control individual flow rates, it becomes feasible for the router to help change how the TCP rate is controlled and thus avoid many of the serious problems TCP is now creating.

TCP PROBLEM AREAS

The largest problem with TCP is that it interprets all lost packets as being caused by network congestion, and thus it cuts its rate in half. It then climbs back up in rate until another packet is lost and again cuts its rate in half. This is effective if all lost packets are being caused by network congestion but as we migrate more and more to wireless links or access, the cause of packet loss will often be due to radio noise, not congestion. TCP degrades rapidly with packet loss and by the time it reaches 20-30% TCP gives up and times out. Before that, its file delivery time increases by over 100:1. Thus, improving the communication between the routers and the sender as to available recourses and what rate will work can have major benefits.

A second major problem with TCP is that as the round trip time (RTT) from the sender to the receiver and back increases, TCP operates slower and slower. This is because TCP allows a certain size block of data to be

sent before it must receive an acknowledgement. Thus, as the RTT increases, users are restricted to slower and slower rates, considerably reducing throughput and introducing a strong unfairness to the network based on the distance between nodes.

A third problem with TCP is that even at the same distance and over the same path, two users may receive totally different rates, an unfairness that has been accentuated by P2P traffic in recent years. The longer a flow continues, the better chance it has to acquire a large fraction of a links bandwidth, thus squeezing other users who may be doing short web accesses. This unfairness can be very serious on the final access link, be it cable, WiFi, or DSL. Here, the router can improve things easily by monitoring the rate of each flow and controlling the rates to be as fair as possible in each payment class.

A fourth problem with TCP is that TCP will keep increasing its rate until some router is forced to discard packets to control the rate. These router discards themselves hurt TCP throughput just like radio noise and TCP throughput and efficiency is inherently limited by the very nature of the control method. In some more extreme cases, the routers will discard over 30% of the packets causing TCP to timeout and try again. This is clearly very inefficient and annoying.

FLOW BASED SIGNALING

In order to overcome both the TCP problems and overload problems with UDP, it would be valuable to signal a QoS request across the network in the first packet of a flow and thus:

1. Inform the network about the requirements of the flow
2. Allow the network to modify the request to specify if it can support the request, or if it cannot, what it can support
3. To verify that the user has paid for or been authorized to use the level of QoS requested (rate, priority, delay, loss)
4. To establish a TCP rate so that slow start and packet discarding are unnecessary

A QoS Signaling protocol was developed by the author in response to the DOD need to make both TCP and Video work as efficiently as possible over satellites and radio links. The primary group responsible for satellite standards, the Telecommunications Industry Association, considered this protocol and approved it as a TIA standard in 2005. The protocol is TIA 1039. Since then the ITU has taken up consideration of the protocol. Since TIA 1039 is complete, its function and impact will be considered here.

TIA 1039

The presumption in TIA 1039 is that semiconductor technology advances now allow routers to be designed which can examine and modify in-band QoS signaling at line rate, and thereafter support the QoS (rate, priority, delay variance, and loss) that has been negotiated. As previously mentioned this can not only be done today, but it also eliminates the need to route every packet and thus is actually less expensive than previous routers. Even if all routers are not currently designed to support these capabilities, we must plan protocols that permit the advancement of the network as new technology permits us to improve function and lower cost. At the same time, the protocol must work with a mixture of old and new routers so that a transition can be phased over time.

IN-BAND VS. OUT-OF-BAND SIGNALING

Historically, all QoS signaling has been done out-of-band starting with the telephone network (SS7) and also for ATM and Frame Relay. ATM's failure was largely due to the signaling being out-of-band since the CPU capacity to process the call setup message was slow enough that it became the limiting factor in setting up short calls or flows. In general, if the CPU can process X calls per second and the trunk capacity is Y bits per second, then if the average call size is less than Y/X bits, then the CPU will limit the capacity of the switch. For ATM, this allowed the switches to be efficient for six minute 64 kilobit/second voice calls, but totally inadequate for short IP flows such as occur with compressed voice or in WWW transactions. Since both the CPU and the trunk speeds have improved with semiconductor trends, this problem remains and has about the same impact on IP routers as it did on ATM switches. If the signaling is out-of-band then only large flows can be supported. However, the improvements in semiconductor and memory technology have been quite sufficient to permit powerful in-band signaling to be processed at line rate. To do this, the signaling information should be in a simple, fixed position format so as to facilitate hardware processing. However, this still permits a QoS request with all the features that ATM had plus adding new requirements like Preemption Priority which is required for military and emergency services and also valuable in the office or home. So long as the hardware to process and manage the QoS is significantly less expensive than the hardware to support a high speed trunk, the result will scale as trunks get faster, even if parallel hardware is required. In fact, since the technology trend for memory is faster than the trend

for trunk hardware speed, the situation actually improves with time and keeping flow information is becoming a lower and lower percentage of the total cost. As a result of these trends, TIA 1039 is structured to operate totally with in-band signaling so that it can scale smoothly as networks grow faster and still support flow of any size.

SIGNALING MESSAGE

The TIA 1039 signaling message is a 16 bytes block with the following QoS fields:

1. Available Rate – the rate that the network sets to balance its load, typically for file transfer where any rate will work (TCP)
2. Guaranteed Rate – a rate set by the sender that is required to support the flow in real time, typically for voice, video and streaming media (UDP)
3. Preemption Priority – a priority that specifies which flows should be kept intact and which dropped if there are too many requests or the network loses capacity
4. Delay Priority – indicates the relative requirement for low delay variance or more tolerance for delay variance.
5. Burst Tolerance – indicates how much the sender can burst over the requested rate

The rates are 16 bit floating point numbers (same format as ATM) so as to allow a very large range of rates. Guaranteed Rate is used in two ways, either as a firm guarantee until closed as may be desired for a leased line replacement, or as a semi-firm guarantee called Max Rate that the network will make every attempt to support but that will be dropped when unused for a short period and that may be preempted by higher priority flows. Max Rate is the correct type to use for a video or voice call and should be set to the maximum rate expected during the flow. In all cases, unused bandwidth will be filled in with other traffic having lower delay priority, but will be made available within one packet time whenever the Max Rate or Guaranteed Rate flows need their capacity.

TCP OPERATION (EXPLICIT RATE OPERATION)

Available Rate flow setup provides a major improvement in TCP because it allows the sender to ask for the highest rate he can use and then each protocol supporting router lowers the rate to the rate it is able to support for all he flows in the same class. In IPv4 TCP, the QoS messages are placed in the data field of the TCP 3-way startup exchange (SYN, SYN/ACK, ACK). In IPv6, the QoS block is placed in a hop-by-hop option field of the same TCP startup messages so that IPSEC does not encrypt it. This way no extra packets are required and the QoS in both directions can be requested and a response received. The forward request is reduced in rate and perhaps in delay and preemption priorities as each router determines what capacity is available. When received at the far end, the receiver responds with the negotiated QoS information so that the sender knows what rate, delay priority, and preemption priority he has available. He can then bypass slowstart and start sending at the negotiated rate. If packets are lost he knows that this is due to line noise, not congestion, so he can continue sending at the full rate using selective retransmission to replace the lost packets. If the network determines that it must lower the rate due to lost capacity or increased load, it can forward a signaling packet that demands a lower rate to the receiver and the receiver returns the message to the sender. The sender must then lower the rate to the new rate. The sender is also allowed to request a new higher rate, if available, every 128 packets. Figure 1 shows the process.

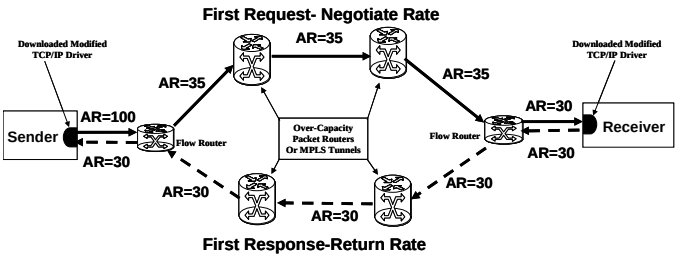

Figure 1 – TCP Rate Negotiation

As shown in Figure 1, the edge routers should be routers that support the TIA 1039 protocol, typically flow routers. The core routers need not understand the protocol and as a result, will pass the QoS message without change. So long as they are over-capacity core routers or the path is a guaranteed rate MPLS path between the edge routers, the edge routers will be the only place where the QoS message needs to be processed since this is where the congestion occurs. However, if there is a heavily loaded intermediate link like a peering link or satellite gateway, then both ends of those links also need protocol sensitive routers. The operation in figure 1 is that the sender sends his first SYN packet with the DiffServ code set to “QoSSignal” and the QoS message attached at the end requesting 100 Mbps. The first flow router marks the rate down to what it can support, 35 Mbps. The core routers do not touch the message. The final edge router marks it down again to 30 Mbps and forwards it to the receiver. The receiver returns the QoS message intact along with his rate

request in the SYN/ACK packet. It returns to the sender and the sender's TCP driver notes that it has been approved 30 Mbps and both edge routers have seen and marked the packet. The sender then sets TCP to send at 30 Mbps returning the final ACK with the receivers QoS message response and starts sending data. If packets are noted lost, the sender continues at 30 Mbps, retransmitting the lost packets with the standard selective retransmit protocol. If a rate change is needed by the network it signals it around to the sender. Every 128 packets the sender can request a higher rate.

The result of this rate negotiation across the network allows TCP to work at optimal throughput. The network does not need to discard any packets to signal the rate and the process of determining the optimal rate is far superior to the "bang-bang servo" type operation of TCP with discards or with EIN marks. The sender finds the optimal rate in minimal time and the network can control all rates down as far as needed as fast as possible if an emergency arises. If a trunk breaks, some packets may be lost, but the network can quickly notify all users to slow down as far as necessary. Also, all users at all distances receive a fair rate. However the largest gain is that the user gets to send a file in a small fraction of the time previously required. Figure 2 shows the impact of the example this for sending an 8 MB file cross country with a 2% packet loss rate. Using 100 Mbps interfaces, the delivery time reduces from 28 seconds to 1 second.

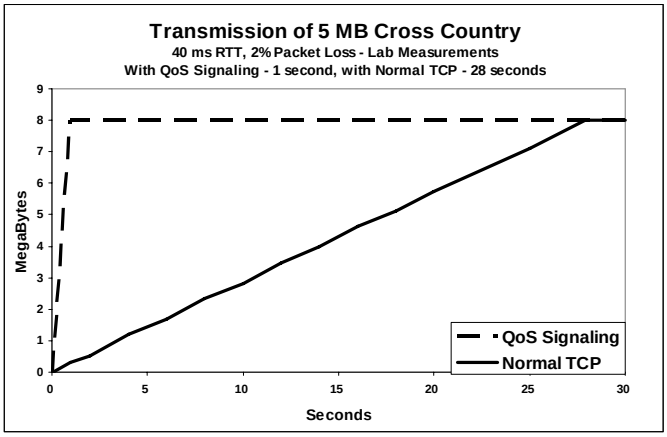

Figure 2 – Time to Receive an 8 MB File

TCP/IP DRIVER SOFTWARE

To implement this new protocol, the software driver in the sender and receiver computers needs to be upgraded to support this protocol. This has been done for Linux systems and based on the DARPA support for this activity, is being made available freely as freeware. The new driver will eventually be available as a simple download, but until that is setup, it must be loaded into Linux. The driver supports IPv4 and IPv6, available rate and max rate (TCP and UDP). When a TCP request is made, the driver adds the QoS request and marks the packet's DiffServ as "QoSSignal". If the receiver also has an upgraded driver, and the two edge routers have marked the QoS message appropriately, then the receiver returns the QoS message and its request in the SYN/ACK. If the sender receives a correct response from the receiver indicating that the edge routers and the receiver are QoS capable, then it will jump to the agreed rate. For moderate error rates the sender will proceed at the agreed rate using selective retransmission. However, to protect against a network path that has a congested peering link or similar problem which does not have protocol capable routers, the sender will back off like normal TCP if the error rate exceeds a preset level. Typically this would be set at 10% or in a poor radio area somewhat higher. However, a router that is overloaded will quickly progress to 20% discards so that this provides good protection against an overlooked and overloaded network path. This, the driver should be able to be used all the time and will only proceed to skip slowstart if the network and the receiver are properly protocol sensitive.

EXPERIMENTAL RESULTS

In order to determine the actual benefit of this new protocol, experiments have been run with transmissions between two Linux servers over a 100 Mbps path including a Linux server which introduces any delay and loss desired. Then both the standard Linux TCP and the TIA 1039 modified Linux TCP drivers were used to send various length files using FTP and recording the times for each transmission.

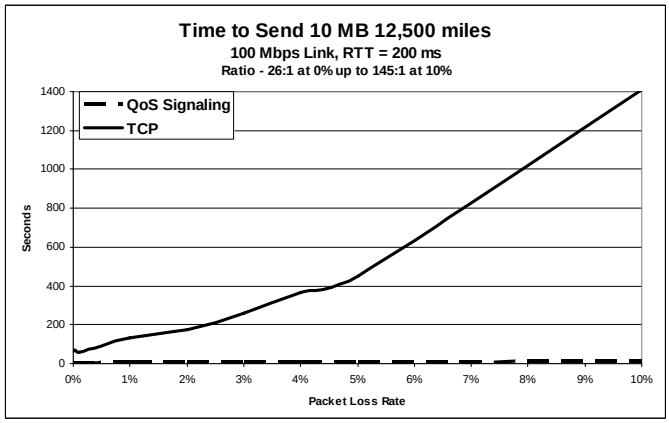

Figure 3 – Transmission time with various loss rates

As can be seen in Figure 3, the Explicit Rate protocol has only a slight transmission time increase from 2 to10 seconds whereas normal TCP increases from 66 to 1411 seconds. The improvement is about 62:1 over the whole range of packet loss and grows to 145:1.

FORMULAS DERIVED FROM EXPERIMENTS

The following formulas were derived as the best fit to the observed date from many measurements of many file sizes, delays, and packet loss rates. For these equations, the term ***loss*** is the packet loss rate fraction (both directions) and ***delay*** is the round trip delay in milliseconds. **CR** is the channel rate in Mbps. The resulting rate, R is in Mbps.

1. Normal TCP
   R=CR/ (1+delay/2)/ (1+loss/.007)
2. Explicit Rate TCP
   R=CR/ (1+delay/60)/ (1+loss/.12)

These equations project that the ratio between Normal and Explicit Rate TCP increases from 1 with no delay and no loss to 224:1 with a 10% loss rate and two satellite hops (1 second). However, the gain is already 10:1 with a .2% loss rate and a cross US round trip delay of 40 ms. Normal router control of TCP will always introduce at least 1% packet loss just to control the rate. Thus the Explicit Rate signaling would improve normal cross country TCP rates by 18:1, even without any radio or satellite links. The equations also show that TCP is very sensitive to both packet loss and delay, with the time of transmission increasing rapidly as the loss increases above 0.5% and as the delay increases beyond 2.5 ms (156 miles of fiber). The change to Explicit Rate signaling halves the sensitivity to packet loss and almost eliminates the impact of delay for distances under 3,200 miles.

Fairness with Distance

From the formulas above, one can also examine the fairness of normal TCP and Explicit Rate TCP with distance. In both cases, TCP loses throughput as the distance increases, but the Explicit Rate TCP is about twice as fair in general. For example for a user in California accessing a server nearby vs. one in Japan accessing the same server, the RTT would be 80 ms and the Japan user would receive the same file in 1.7 times the time with Explicit Rate instead of 20 times the time with normal TCP. Thus the improvement in fairness is 12:1. Actually, with flow routers in the path, they would further improve the fairness because all users going through the router near the server would receive about the same rate.

JITTER REMOVAL

A third benefit of the Explicit Rate protocol is that it tells each user a rate to send at that permits the user to send continuously. Typically TCP sends in bursts and then waits for acknowledgements before proceeding. This creates very bursty traffic for the network. However when the users are able to send continuously, the network traffic will become extremely smooth since everyone is sending at fixed rates. The only jitter would be from users coming and going, a much more Poisson process which smoothes easily.

FUTURE WORK

The current implantation of the Explicit Rate driver does not achieve the improvement with packet loss that should be feasible theoretically. If the TCP driver could hold the sending rate constant in the face of packet loss, the only throughput reduction that should be seen would be the impact of the retransmissions, not delay, which should result in R=CR/(1+loss). Currently the results are much worse than this, but still much better than unmodified TCP. Unfortunately, in modifying the current Linux TCP stack, it is hard to totally fix all of the convoluted old code. Further work in totally revising the code should reduce the impact of both delay and packet loss, allowing improvements of up to 400:1 over normal TCP.

CONCLUSION

The negotiation across the network of an explicit rate to send TCP transmissions, provides a major improvement in file transmission time if there is packet loss or delay in the path. TIA 1039 protocol defines a standard for this process. If there are satellite hops or radio links in the path, as there almost always is for DOD to the front, the gain averages 40:1 and grows to 140:1. Future work should increase the gain much further.

APPENDIX – QOS SIGNALING PROTOCOL

The following is a compact summary of the referenced QoS Signaling Protocol.

IPV4 PACKET FORMAT

For IPv4, a "Start Packet" is sent at the start of each flow. It contains the QoS Structure in the data section and must have the same source address, destination address, protocol, source port, and destination port as the remainder of the flow. This makes it part of the flow. The Diffserv code for all signaling packets will be set to "QoSSignal", code 000011. This tells the routers and the receiver that it is a Start Packet. It should not contain any additional data except the 16-byte QoS Structure(s). All other packet in the flow should use DiffServ code "QoSFlow", code 000111.

<table>
<tr><td colspan="6">Source Address</td></tr>
<tr><td colspan="3">Available Rate</td><td colspan="3">Guaranteed Rate</td></tr>
<tr><td>PP</td><td colspan="2">D</td><td>C</td><td>T</td><td>C</td><td>B</td></tr>
<tr><td colspan="2">QoS</td><td>M</td><td colspan="4">Source Port</td></tr>
</table>

Figure 4 – IPV4 QoS Structure

The QoS Structure will appear in the TCP or UDP data section. For TCP, the Forward QoS Structure will typically be sent in the SYN packet. The QoS capable routers in the flow path should modify the QoS parameters as required. When the destination receives the Start Packet request in a TCP SYN packet, it responds with a SYN/ACK packet with two QoS Structures. The first 16 byte QoS Structure is the Response to the forward path request and the second QoS Structure is the Reverse path QoS request. Then the sender responds with an ACK packet as in the normal 3-way TCP handshake, including the QoS structure of the reverse path response. Mid flow updates (AR or GR Rate reductions) are accomplished by QoS capable routers by simply modifying a TCP packet setting Diffserv set to "QoSSignal" and inserting a QoS Structure. The QoS Structures are removed by the QoS sub-layer before the packets are passed to the IP stack. If the flow is a UDP flow, then the QoS Structure will appear in the UDP data section.

IPV4 QOS STRUCTURE FIELDS

The IPv4 Request/Response QoS Structure is 16-bytes with 12 fields as follows:

- **SOURCE ADDRESS: (**32 bits**)** Set to 0 on original request. Set to the prior real source address to re-start a flow after a move or the sender's source address in a response.
- **AR**: (16 bits) Available Rate – floating point rate for network assigned rates The 15 lower bits are in ATM format.
- **GR**: (16 bits) Guaranteed Rate –floating point rate for requested guaranteed rate. The 15 lower bits are in ATM format.
- **PP**: (8 bits) Preemption Priority – indicates the override or preemption priority of the flow – 64 levels – 0=lowest, 63=highest in the high order 6 bits. The two low order bits are reserved.
- **DP:** (8 bits) Delay Priority in 64 levels. – 0=lowest priority, 63=highest priority. The highest level gets priority in transmission, reducing delay variation.
- **CD:** (4 bits) Change/Direction field – 0=Forward no action required, 1=Request, 2=Response returning agreed parameters to the sender, 3=Confirmation of the negotiated parameters, 4=Missed Start Packet, 5= Renegotiate request every 128 packets. 6=Resend Start Packet, 7=Close for a Guaranteed Rate flow. No other values are specified at this time.
- **TP**: (4 bits) Type of flow – 0=Available Rate (TCP), 1=Composite Rate where GR is requested and AR is assigned by the network, 2=Maximum Rate, 3=Guaranteed Rate as specified in the GR field. No other values are specified at this time.
- **CH:** (4 bits) Charging information. – 0 = Forward charging, 1 = Reverse charging No other values are specified at this time.
- **BT**: (4 bits) Burst Tolerance – the time (at approved rate) a flow is permitted to exceed its rate (GR+AR) before packets are discarded. Time=2^ (-BT-1) seconds
- **QOS VERSION:** (12 bits) QoS Protocol Version Field – (initially set to 1)
- **M:** (4 bits) Modified marker. Set to 0 by sender on request. Set to 1 by routers if any field changed during a request or renegotiate.
- **SOURCE PORT** (16 bits) Set to 0 on original request set to the real source port when re-starting a flow after a move or the sender's source port in a response.

IPV6 PACKET FORMAT

For IPv6 where everything after the hop-by-hop options is encrypted, the QoS Structure must be a hop-by-hop option since it must be seen by the routers. It cannot be in a separate Start Packet as with IPv4. The IPv6 header will have the Next Protocol Field set to zero to indicate there is a hop-by-hop option next. The QoS option need not be the first hop-by-hop option. Hop-by-hop options in IPv6 have a standard format and the first 4 bytes are needed to describe the length, the next protocol field, the option type (QoS) and the Option Length. Then the QoS fields are in the next 8 bytes, the same format and fields as in IPv4. The last four bytes are used for the QoS Version, M, and a reserved space. In the response message, the reserved space is used for the flow label of the forward flow.

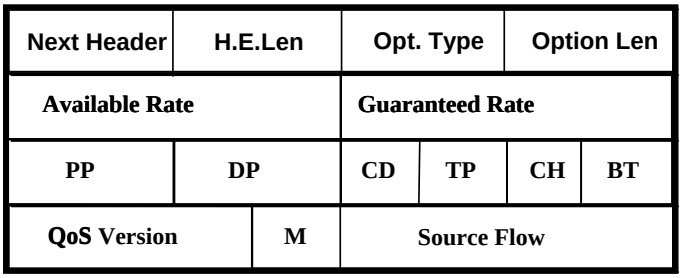

Figure 5 – IPV6 QoS Structure

The IPv6 QoS Request Option Field is 16-bytes with 15 fields as follows:

- **NEXT HEADER**: (8 bits) Identifies the type of header immediately following this option
- **HEADER EXTENSION LENGTH**: (8 bits) Header Extension Length is set to 1 to indicate that the QoS extension field since it is exactly 16 bytes long (8-bytes plus 1 additional 8-byte segment).
- **OPTION TYPE**: (8 bits) Option Type indicates the QoS code to be selected. It must start with '001' since it of type "skip if not recognized" and subject to "change each hop". No options are currently assigned with the 001 header. Thus, until such time that IANA formally assigns an option code, the default code for experimental use is selected as 00100000 or in decimal, 32.
- **OPTION LENGTH**: (8 bits) Option Length is set to 12 (12 more bytes).
- **QOS FIELDS** – Same as IPv4
- **SOURCE FLOW LABEL**: (16 bits) - Set to zero on request.

IPv6 Response Packet

The IPv6 QoS Structure Response is the same format as the request except for the last field:

- **SOURCE FLOW LABEL:** (20 bits) – Set to the Flow Label of the forward flow (using the M field)

IPV6 CONFIRMATION AND CLOSE PACKETS

For Guaranteed Rate flows, the format of the IPv6 confirmation packet is identical to the IPv6 request packet except for the CD field. CD is set to 3. All the QoS parameters received in the response are forwarded as received with no change. This allows all the routers to drop excess capacity reservations. The close packet is also the same except CD=7

IPV6 RENEGOTIATE PACKET

The format of the IPv6 renegotiate packet is identical to the IPv6 request packet except for the CD field. CD is set to 5. All the QoS parameters received in the response are forwarded as received with no change except the available rate (AR) and CD. This allows the sender to request an increased rate or a router to reduce a rate.