IPv6 Working Group Rahul Banerjee
Internet Draft Sumeshwar Paul Malhotra
Mahaveer M
BITS, Pilani (India)
April 2002

A Modified Specification for use of the IPv6 Flow Label for providing An efficient Quality of Service using a hybrid approach.
draft-banerjee-flowlabel-ipv6-qos-03.txt

Obsoletes 00, 01, 02 versions of this draft.

Status of This Memo

Abstract

This memo suggests a pragmatic specification for defining the 20-bit Flow Label field using a hybrid approach that includes options to provide IntServ as well as DiffServ based support for IPv6 Quality of Service. It also compares various suggested approaches for defining the 20-bit Flow Label field in IPv6 Base Header based on RFC 2460 (December 1998) and few other drafts. Addressing the IPv6-Multicast-QoS issues also becomes possible as a consequence. This draft clearly specifies exactly when and how various options are to be used; and in case of the MFC, exactly how a specific action might be taken by the suggested implementation. Thus the resultant mechanism is fully implementable and unambiguous as even the lower-level details have been worked out as may be required for actual implementations. The draft also has a pointer to an experimental QoS scheme called MultServ.

Table of Contents

1. Introduction

This draft addresses the design and implementation-specific issues pertaining to the Quality of Service (QoS) support in the Flow Label field of the IPv6 Base Header. It provides support for IntServ and DiffServ Quality-of-Service. Though the IPv6 Base Header has a 20-bit Flow Label field for QoS implementation purposes, it has not yet been exploited. Very few Internet Drafts address these long-standing issues and attempt to present solutions in the form of a clear specification of the 20-bit Flow Label in IPv6. This work attempts to provide an analysis of these definitions and subsequently suggests a modified IPv6 Flow Label specification, which in view of the authors can provide an efficient Quality-of-Service.

2. IPv6 Flow Labels

The IPv6 Flow Label [RFC 2460] is defined as a 20-bit field in the IPv6 header which may be used by a source to label sequences of packets for which it requests special handling by the IPv6 routers, such as non-default quality of service or "real-time" service. The nature of that special handling might be conveyed to the routers by a control protocol, such as RSVP, or by information within the flow's packets themselves, e.g., in a hop-by-hop option.

The characteristics of IPv6 flows and Flow Labels are given in the Appendix A.1

3. Issues related with IPv6 Flow Label

According to RFC 1809, the IPv6 specification originally left open a number of issues, of which the following are important.

3.1 What should a router do with Flow Labels for which it has no state?

[RFC 1809] and the author's view suggest that the default rule should be that if a router receives a datagram with an unknown Flow Label, it treats the datagram as if the Flow Label is zero. Unknown flow labels may also occur if a router crashes and loses its state. As part of forwarding, the router will examine any hop-by-hop options and learn if the datagram requires special handling. The options could include simply the information that the datagram is to be dropped if the Flow Label is unknown or could contain the flow state the router should have.

3.2 How does an internetwork flush old Flow Labels?

Stale Flow Labels can occur in a number of ways, even if we assume that the source always sends a message deleting a Flow Label when the source finishes using a Flow.

1. The deletion message may be lost before reaching all routers.

2. Furthermore, the source may crash before it can send out a Flow Label deletion message.

The authors of the document suggest the following approach as a solution to this problem:

1. The MRU (Most Recently Used) algorithm should be used for maintaining the Flow Labels. At any point of time, the most recently used Labels alone will be kept and the remaining should be flushed.

2. Before flushing a label, the router should send an ICMP message to the source saying that the particular label is going to be flushed. So the source should send a KEEPALIVE Message to the router saying not to flush the Flow Label in case the source requires the Flow Label to be used again. On the other hand, if the source agrees with the router to delete the Flow Label, it should send a GOAHEAD Message to the router. On receiving the GOAHEAD Message, the router immediately deletes the label for that particular source. These messages are also sent to all the intermediate routers, so that, those routers can as well flush the Flow Labels for that particular source.

3. In case, the router does not receive any consent from the source, it will re-send the ICMP message for at most two or three times. If the router does not receive any reply from the source, it can flush the particular Label assuming that the Flow Label was not important for the source or any other intermediate router. The intermediate routers will also delete that Flow Label as they didn't receive any message from the source. The policy of sending the ICMP message to the source two or three times ensures the proper behavior of the method of flushing Flow Labels in case of packet loss. This method assumes that the ICMP message would not be lost all the three times. Hence, if the router doesn't receive any reply from the source even after sending the ICMP message three times, it deletes the label.

3.3 Which datagrams should carry non-zero Flow Labels?

According to RFC 1809, following were some points of basic agreement.

1. Small exchanges of data should have a zero Flow Label since it is not worth creating a flow for a few datagrams.

2. Real-time flows must always have a Flow Label.

One option specified in [RFC 1809] is to use Flow Labels for all long-term TCP connections. The option is not feasible in the view of the authors as it will force all the applications on that particular connection to use the Flow Labels which in turn will force routing vendors to deal with cache explosion issue.

3.4 Mutable/Non-mutable IPv6 Flow Label

The Flow Labels should be non-mutable because of the following reasons:

1. Using mutable Flow Labels would require certain negotiation mechanism between neighboring routers, or a certain setup through router management or configuration, to make sure that the values or the changes made to the Flow Label are known to all the routers on the path of the packets, in which the Flow Label changes. On the other hand, the non-mutable Flow Labels certainly have the advantage of the simplicity implied by such a characteristic.

2. A mutable Flow Label characteristic goes against the IPv6 specification of the Flow Label explained in section 2 and the IPv6 Flow Label characteristics explained in the coming sections.

3.5 Filtering using Flow Label

If, at all, any filtering has to be done based on the Flow Label field in the IPv6 header, the expectation is that the IPv6 Flow Label field carries a predictable or well-determined value. This is not the case if the Flow Label has randomly chosen values.

Supporting the arguments given in [draft-conta-ipv6-flow-label-02.txt], the authors of this document suggest that the problem of not being able to configure load-filtering rules, which are based or are including the Flow Label, can be resolved by relaxing IPv6 specification of having a random number in the Flow Label field. Exactly how can it be done has been suggested later.

4. A modified specification for the IPv6 Flow Label and related implementation mechanism: A hybrid approach suggested by this work

4.1 Overview

Appendix A.2 gives a comparison on various approaches suggested in [draft-conta-ipv6-flow-label-02.txt] on defining the 20-bit Flow Label.

This section specifies a modified Flow Label for IPv6 for providing efficient Quality of Service that utilizes the results of some of the works referred in Appendix A.2, extends some of these suggested mechanisms and finally presents an integrated hybrid approach.

4.2 Definition of first three bits of the Flow Label

The hybrid approach suggested in this section includes various approaches which are mentioned in Appendix A.2. The 20-bits of the Flow Label should be defined in an appropriate manner so that various approaches can be included to produce a more efficient hybrid solution. Hence, for this purpose, the first three bits of the IPv6 Flow Label are used to define the approach used and the next 17 bits are used to define the format used in a particular approach.

Following is the bit pattern for the first 3 bits of Flow Label that defines the type of the approach used:

0 0 0   Default.

0 0 1   A random number is used to define the Flow Label.

0 1 0   The value given in the Hop-by-Hop extension header is used instead of the Flow Label.

0 1 1   PHB ID.

1 0 0   A format that includes the port number and the protocol in the Flow Label is used.

1 0 1   A new definition explained later in this section is used.

1 1 0   Reserved for future use.

1 1 1   Reserved for future use.

This definition of Flow Label includes IntServ, DiffServ and other approaches for defining the Flow Label. A further explanation of these options is provided in the remaining part of this section. The default value specifies that the datagram does not need any special Quality of Service.

4.3 Defining the remaining 17 bits of the IPv6 Flow Label

The remaining 17 bits of the IPv6 Flow Label are defined based on the approach defined in the first three bits of the Flow Label.

4.3.1 Random Number

As specified in IPv6 specification, a random number can be used to define the Flow Label. Here a 17-bit random number can be used. The random numbers can be generated in the range from 1 to 1FFFF. Keeping the IPv6 specifications in mind, the authors of this document believe that the random number can be used as one of the approaches. As other approaches are defined in the Flow Label, this random number approach may not be used whenever not feasible or efficient to do so.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 0 1|     Pseudo - Random value       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.3.2 Using Hop-by-Hop extension header

As defined in [draft-banerjee-ipv6-quality-service-02.txt], Hop-by-Hop extension header can be used for defining the Flow Label in case IntServ is used. In this case the value in the 20-bit Flow Label is ignored. The modified Hop-by-Hop extension has been suggested and defined in the reference [draft-banerjee-ipv6-quality-service-02.txt]. In that draft, the Hop-by-Hop extension header has been defined to be used with IntServ. This mechanism applies to define for DiffServ as well.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 0|         Don't care              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.3.3 Using PHB ID

This defines the DiffServ with MF classifier. In that case the format of the Flow Label will be as shown below:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 1|     DiffServ IPv6 Flow Label    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

As suggested in [draft-conta-ipv6-flow-label-02.txt], this Flow Label can be a PHB ID (Per Hop Behavior Identification Code). In this case, 16-bit PHB ID will be used and the remaining 1 bit is reserved for future use.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 1 1|  Per Hop Behavior Ident. Code |R|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

'R' is reserved.

Packets coming into the provider network can be policed based on the Flow Label. The provider, based on the SLAs, SLSs, TCAs, TCSs agreed with the client, configures MF classifiers. This draft specifies the classifier which is little different from the one suggested in the [draft-conta-ipv6-flow-label-02.txt]. The classifier looks like:

```
C  = (SA/SAPrefix, DA/DAPrefix, Flow-Label).
Or
C` = (SA/SAPrefix, DA/DAPrefix, Flow-Label-Min: Range).
```

The range here specifies the difference between the maximum and the minimum Flow Label. The significance of using the range instead of Maximum Flow Label is the reduced number of bits. Definitely the difference between the two values can be specified in a lesser number of bits as compared to the value itself.

```
Flow-Label-Classifier:

IPv6SourceAddressValue/Prefix:  10:11:12:13:14:15:16:17:18::1/128
IPv6DestAddressValue/Prefix:    1:2:3:4:5:6:7:8::2/128
IPv6 Flow Label:                50

Or

IPv6SourceAddressValue/Prefix:  10:11:12:13:14:15:16:17:18::1/128
IPv6DestAddressValue/Prefix:    1:2:3:4:5:6:7:8::2/128
IPv6 Flow Label:Range:          10:20
```

Incoming Packet header (SA, DA, Flow Label) is matched against classification rules table entry (C or C`).

4.3.4 Using the Port Number and the Protocol

This approach defines Flow Label by including the server port number and the host-to-host protocol. The "Server Port Number" is the port number assigned to the server side of the client/server applications. As specified in [draft-conta-ipv6-flow-label-02.txt], this approach reserves 16 bits for the port number and 1 bit for the protocol with the remaining bits reserved for the future use.

```
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1 0 0|     TCP Server port number     |0|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 0 0|     UDP Server port number    |1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

But this approach puts the restriction on the protocol to be used by any application.

As most of the application seeking Real-time service use TCP or UDP as the transport layer protocol, this approach would work fine in most of the cases. In case the application requires to use any other host-to-host protocol, the other methods for specifying the Flow Label, discussed in this section can be used. Anyhow, this method for specifying the port number and the protocol can be exploited further in the future to remove any limitations.

4.3.5 A new structure and mechanism for the use of the Flow Label

This section describes an innovative approach to define the 20-bit Flow Label field in IPv6 header. By the optimal use of the bits in the Flow Label, this approach includes various Quality of Service parameters in the IPv6 Flow Label that may be requested by any application. The various Quality of Service parameters are:

1. Bandwidth
2. Delay or Latency
3. Jitter
4. Packet Loss
5. Buffer Requirements

As packet loss and the jitter are often desired to be of minimum value by any application, these two parameters may not be defined in the Flow Label field itself. Instead, if needed, the Hop-by-Hop EH space can be effectively used to specify these parameters. Bits thus saved in the Flow Label can be effectively used for more demanding purposes. The Quality of Service parameters that are to be included in the Flow Label are:

1. Bandwidth (to be expressed in multiples of kbps).
2. Delay   (to be expressed in nanoseconds).
3. Buffer requirements (to be expressed in bytes).

As there are only 17 bits left, the optimal use of the bits is very important so as to obtain the maximum information out of those 17 bits. The first bit out of these 17 bits is used to differentiate between the hard real time and soft real time applications. This bit is set to 0 for soft real time applications and it is set to 1 for hard real time applications.

Soft Real time applications:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|1 0 1|0|      Flow Label               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This service is meant for RTT (Real Time Tolerant) or soft real time applications, which have an average bandwidth requirement and an intermediate end-to-end delay for an arbitrary packet. Even if the minimum or maximum values specified in the Flow Label are not exactly met, the application can afford to manage with the QoS provided.

Hard Real time applications:

```
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1 0 1|1|      Flow Label               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This service is meant for RTI (Real Time Intolerant) or hard real rime applications, which demand minimal latency and jitter. For example, a multicast real time application (videoconferencing). Delay is unacceptable and ends should be brought as close as possible.

For this videoconference (DTVC) case, the required resource reservations are

a. Constant bandwidth for the application traffic.
b. Deterministic Minimum delay that can be tolerated.

These types of applications can decrease delay by increasing demands for bandwidth. The minimum or maximum values specified in the Flow Label have to be exactly met for these kind of applications.

After keeping one bit for Hard/Soft real time applications, we are left with 16 bits for defining the Flow Label. The remaining part of this section discusses how to represent the values of bandwidth, delay and buffer requirements.

1. Bandwidth

This definition specifies 6 bits out of the 16 bits to be used for specifying the bandwidth value.

Each value in these six bits corresponds to a pre-defined value for bandwidth. Further explanation about this is given at the end of this section.

2. Buffer Requirements

This definition specifies next 5 bits out of the 16 bits to be used for specifying the buffer value.

Each value in these six bits corresponds to a pre-defined value for buffer requirement. Further explanation about this is given at the end of this section.

3. Delay

This definition specifies last 5 bits out of the 16 bits to be used for specifying the delay value.

Each value in these six bits corresponds to a pre-defined value for delay.

The approach described here is a DiffServ based mechanism for providing the QoS as any packet received by any router is classified based on the MF Classifier which is a triplet consisting of the source address, destination address and (bandwidth, buffer and delay). The packet that arrives at the router is examined for the values specified in bandwidth, buffer and delay fields and is matched with the classifiers corresponding to which the packet is provided with the QoS. The classifier looks like:

C = (src address, dest address, flow label);

Where flow label = (bandwidth, buffer, delay)

MF Classifier	Bandwidth	Buffer	Delay
0, 0, 0	32 kbps	512 bytes	4 ns
0, 0, 1	32 kbps	512 bytes	8 ns
.			
.			
.			
63, 31, 31	64 tbps	1 tbytes	8 sec

5. A possible mechanism for the implementation of the above design.

This section describes one possible mechanism that will allow immediate and practicable implementation of the above design.

5.1 Data structures required (at the router).

The data structures are specific to the implementations. Different implementations can choose their own data structures that will be required to implement the above design.

Any router that tries to implement QoS maintains a QoS routing table and keeps track of the QoS available to each destination through the required number of hops [RFC 2676]. Apart from this table, the router needs to keep track of the allotted QoS to each and every flow. This table is the ALLOTTED_QOS_TABLE.

1. Defining the different approaches.

```
enum MODEL_ID {
  RANDNUM=1,       // the random number method
  HOPBYHOP=2,      // the hop-by-hop extension header method
  PHB_ID=3,        // the multi-field classifier
  PORT_PROT=4,     // port/protocol method
  HYBRID=5         // the hybrid approach
};
```

2. Defining the different Resource Identifiers.

```
enum RES_ID {
  BANDWIDTH=0,     // bandwidth requirement
  DELAY=1,         // delay requirement
  BUFFER=2,        // buffer requirement
};
```

3. Defining the value of the resource.

```
typedef unsigned int RES_VAL;

struct RESOURCE {
  RES_ID res_identifier; // identifier of the resource
  RES_VAL res_value;     // 32-bit value of the resource
};
```

4. Defining the Quality of Service.

```
struct QOS_INFO {
  MODEL model_id;
  RESOURCE resource;
};
```

5. Defining the port/protocol and the flow label.

```
struct port_protocol {
  unsigned port;      // port number
  unsigned protocol;  // protocol
};

union format {
    unsigned flowlabel;             // 20-bit Flow Label value
    struct port_protocol port_prot;
};
```

6. Defining the packet information.

```
        struct PACKET_INFO {
           struct sockaddr_in6 src_addr;
```

```
          struct sockaddr_in6 dest_addr;
          union format format_value;
        };
```

7. Defining the Alloted QoS table.

```
   struct ALLOTED_QOS_TABLE {
     struct PACKET_INFO packet;
     struct QOS_INFO qos;
   };
```

5.2 Function of the Source

The application specifies the desired QoS and the Flow Label field in the IPv6 header is filled based on the QoS asked by the application. The application has the flexibility of specifying which format it wants to use for getting the desired QoS. It can specify any of the formats described in this document. The packet is then put on the network and it reaches the intermediate routers

5.3 Function of each relevant intermediate router

5.3.1 Initial Processing (Checks for default service)

It gets the format used by the packet by reading the first three bits of the Flow Label. In case the first three bits are 000 or 110 or 111, it represents the default service. No specific treatment is required for this particular packet. In this case, no further processing of the packet is required and the default QoS is provided to the packet. If the value given in the first three bits is 010, no further processing is done and the router knows that the required QoS is specified in the hop-by-hop extension header.

5.3.2 Searching for the entry (In case of non-default service)

1. The ALLOTTED_QOS_TABLE table is searched based on the source address.
2. If an entry is found, then for that particular source, a search is made based on the PACKET_INFO structure defined above. If all the information stored exactly matches with the information contained in the incoming packet, the IPv6 packet is processed so that the reserved QoS is met.

5.3.3 New Entry

1. **If an entry is not found, a new entry is made in the ALLOTTED_QOS_TABLE table for the source and further processing** of this new entry is done as follows.

2. All the relevant structures defined above are filled based on the information contained in the packet. Information about the packet is stored in the PACKET_INFO structure.

3. It reads the desired QoS from the packet's header. If the format specifies that a random number is used in the Flow Label field, it reads the RANDOM_NUMBER table. It reads the specified QoS from the table and maintains that in the QOS_INFO structure after updating the RESOURCE structure. It then moves onto step 7.
4. If the format specifies that PHB ID is used in the Flow Label field, it reads the Flow Label and the packet is classified based on the MF classifier described in the previous section and it moves on to the step 7.
5. If the value in the Flow Label field specifies that the PORT/PROTOCOL field is used in defining the QoS required by the packet, it fills the RESOURCE structure and the QOS_INFO structure and moves onto step 7.
6. If the value in the Flow Label field specifies that the hybrid approach is used where the packet specifies the values of the bandwidth, delay and buffer requirement. The packet is classified based on the MF classifier described in the previous section and it moves on to the step 7.
7. It then checks with the QoS Routing table, to find out if the desired QoS is possible to be provided to the packet. If yes, it updates the new entry in the ALLOTTED_QOS_TABLE table in the memory or else this entry is removed.
8. If any relevant router en-route is not able to guarantee the requested QoS, an ICMPv6 message is sent to the source and the other routers (that had guaranteed the QoS) are also notified of the same so that they delete the corresponding entry from their QoS tables.

This process executes at all the intermediate routers between the source and the destination.

6. When to use which approach?

1. Random Number: This approach supports the pure IntServ based model. So if the network uses only IntServ model for QoS, using random numbers in Flow Label is a valid option. But in some conditions it is not desirable to use random numbers in Flow Label. If the network is required to have a deterministic behavior, using random numbers is not a good option as it increases the unpredictability. Again, if any load filtering rules have to be designed based on or using the Flow Label, random numbers should not be used as the value in the Flow Label can not be predicted.

2. PHB ID: This approach supports the pure DiffServ based model. So if the network is designed so as to support DiffServ model for QoS, using PHB ID in flow label and using MF classifier as described

in the previous sections is a valid option.

3. Hybrid: Again, if the network supports DiffServ model for QoS, using this approach is a valid option. Here the application should be

capable of providing the exact values of bandwidth, delay and buffer requirement it needs.

4. Hop-by-Hop: For using this approach, the application should be capable of specifying the values of QoS parameters. So if the application has these details and the values asked by the application are not supported by the hybrid approach, this approach should be used.

5. Port-Protocol method: If the network is designed so as to perform some load filtering based on the port number or the protocol, this approach is a valid option.

7. Where other approaches differ in defining the Flow Label from the proposed approach

Few internet drafts have differentiated between the control and forwarding plane. [draft-ietf-ipv6-flow-label-00.txt] defines the Control plane as part of an IP node taking care of control functions, such as routing protocols and flow establishment protocols and Forwarding plane as part of an IP node receiving and forwarding IP packets; also known as the "datapath". Having a separation of control plane and forwarding plane does have an advantage as explained in that draft. But it may not be completely beneficial as the TCP/IP architecture itself is not fully layered. Moreover this approach might require some changes in the existing architecture as opposed to the proposed solution given in this draft.

8. Security Considerations

The specifications of this draft do not raise any new security issues. The Flow Label field in the IPv6 header cannot be encrypted because of the known reasons. If encrypted, each in between router has to decrypt the header for providing the required QoS to the packet. As the QoS specification requires minimum delay for the packet, decrypting each packet's header at each router will not be a good idea because of the time required in processing the packet.

9. Conclusion

This report has dealt extensively with all the suggested formats for defining the 20-bit IPv6 Flow Label and finally has suggested a hybrid approach for efficiently defining the 20-bit IPv6 Flow Label.

One of the major reasons why the current solution proposed in this draft provides choice for IntServ/DiffServ based quality of service is the fact that a few representative research experiments in many places including those in Europe (www.bits-pilani.ac.in/ngni) have shown that while

DiffServ is definitely an attractive solution due to its scalability, IntServ has been found to be fair and reasonably efficient under a real life situation constraints that were stimulated in these experiments.

In the meanwhile, yet another Quality of Service approach is gradually evolving (Appendix A.3) that aims to provide a seamless application transparency based solution to provide end-to-end quality of service support. Inspired from the initiative in the distributed operating system research and policy-based QoS mechanisms,this approach is still evolving and refined. It is hoped that once this approach becomes verifiable and viable, an alternate protocol independent quality of service strategy shall be possible to be implemented in the near future.

The emphasis of this work is to result into a practically acceptable specification that could be effectively used for a reasonably long period of time for implementing IPv6 Quality of Service that so far has been elusive in absence of a clear, verifiable and complete specification. A separate ID is under preparation specifically building upon these specifications so as to explicitly address the scalability issues related to the IPv6-Multicast-QoS.

Appendix

A.1. Characteristics of IPv6 flows and Flow Labels

The characteristics of IPv6 flows and Flow Labels as given in RFC 2460 are rearranged as follows:

(a) A flow is uniquely identified by the combination of a source address and a non-zero Flow Label.

(b) Packets that do not belong to a flow carry a Flow Label of zero.

(c) A Flow Label is assigned to a flow by the Flow's source node.

(d) New Flow Labels must be chosen (pseudo) randomly and uniformly from the range 1 to FFFFF hex. The purpose of the random allocation is to make any set of bits within the Flow Label field suitable for use as a hash key by routers, for looking up the state associated with the flow.

(e) All packets belonging to the same flow must be sent with the same source address, destination address, and Flow Label.

(f) If packets of flow include a Hop-by-Hop options header, then they all must be originated with the same Hop-by-Hop options

(g) If packets of a flow include a routing header, then they all must be originated with the same contents in all extension headers up to and including the routing header.
header contents.

(h) The maximum's lifetime of any flow-handling state established along a flow's path must be specified as part of the description of the state-establishment mechanism, e.g., the resource reservation protocol or the flow-setup hop-by-hop option.

(i) The source must not reuse a Flow Label for a new flow within the maximum lifetime of any flow-handling state that might have been established for the prior use of that Flow Label.

A.2. Comparison of already suggested approaches in defining the IPv6 Flow Label format

This section discusses the already suggested approaches in [draft-conta-ipv6-flow-label-02.txt] for defining the 20-bit Flow Label. It discusses the advantages and disadvantages of these approaches. Finally it tells about accepting or not preferring these approaches and includes the accepted approaches (with modifications wherever required) in the final definition of the Flow Label discussed in the next section.

A.2.1 First approach

Following format can be used for the Flow Label:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 0 |   Pseudo - Random value                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 1 |   DiffServ IPv6 Flow Label              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The DiffServ IPv6 Flow Label is a number that is constructed based on the Differentiated services "Per Hop Behavior Identification Code".

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 1 |   Per Hop Behavior Ident. Code|  Res. |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The "Res" bits are reserved.

The PHB ID is either directly derived from a standard differentiated services code point, or it is an "IANA Assigned Value".

Advantages:

Preserves compatibility with the random number method of selecting a Flow Label value defined in IPv6 specification.

Captures the differentiated services treatment intended to be applied to the packet.

Unlike the value of the traffic class field, it is not locally mapped and hence suitable for use in an end-to-end header field.

Disadvantages:

It captures less information than the port number and protocol number normally used in multi field classifier.

A.2.2 Second Approach

DiffServ with multi field classifier can be used in a more efficient and practical manner as an alternative to IntServ and RSVP. The Flow

Label classifier is basically a 3-element tuple - source and destination address and IPv6 Flow Label.

The classifier can be defined in any of the following two ways:

C = (SA, SAPrefix, DA, DAPrefix, Flow Label).

C` = (SA, SAPrefix, DA, DAPrefix, Flow Label min: Flow Label max).

Incoming packet header (SA, DA, Flow Label) is matched with classification rules table entry C or C`.

Advantages:

Helps the IPv6 Flow Label to achieve, as it is supposed, in a more efficient processing of packets in QoS engines in IPv6 forwarding devices.

Disadvantages:

When packets are transmitted, the end nodes have to force the correct Flow Label in the IPv6 headers of outgoing packets or the first hop routers have to do this job. To accomplish these rules, these routers will be configured with MF classifiers. This puts extra computations to be done by the routers.

A.2.3 Third approach

Includes the algorithmic mapping of the port numbers and protocol into the Flow Label. It reserves 12 bits for the port number and 8 bits for the protocol.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Server port number   | H-to-H protocol|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Advantages:

Classification rule is 5 or 6 element tuple format of a DiffServ MF classifier, containing the source and the destination address, the source and the destination ports, the host-to-host protocol. So no new classification rule format is needed.

Disadvantages:

It cannot differentiate among multiple instances of the same application running on the same two communication end nodes.

The reduced number of bits (12 out of 16) limits the value of ports. 12 bits can represent only the "IANA well-known ports", that is from 1 to 1023 and a subset of "IANA registered ports", that is from 1024 to 4095. Registered ports have values between 1024 and 65535.

A.2.4 Fourth approach

The field occupied by host-to-host protocol could be reduced to 1, as TCP and UDP are the only well known protocols.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    TCP Server port number      |Res |0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    UDP Server port number      |Res |1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The "Res" bits are reserved.

The "TCP Server Port Number" or "UDP Server Port Number" is the 16-bit port number assigned to the server side of the client/server application.

Advantages:

Again the classification field is a 5 or 6 element tuple. So no new classification rule is needed.

This approach keeps 16 bits for the port number so that all the "IANA well-known ports" and "IANA registered ports" can be accommodated in these 16 bits.

Disadvantages:

This approach, too, cannot differentiate among multiple instances of the same application running on the same two communication end nodes.

Reserving only 1 bit for the protocol field in the Flow Label restricts the use of any protocol other than TCP and UDP.

A.2.5 Fifth approach

Header length format:

Another possible solution is to store the length of IPv6 headers length that is the length of the IPv6 Base Headers and IPv6 extension headers preceding the host-to-host or transport header. The length of IPv6 headers in the Flow Label value would provide the information, which a DiffServ QoS engine classifier could use to locate and fetch the source and destination ports and apply those along with the source and destination address and host-to-host protocol from the Flow Label, to match the source and destination address, the source and destination ports and the protocol identifier elements of a DiffServ MF classifier.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Length of IPv6 headers| H-to-H protocol|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Advantages:

"Length of IPv6 headers" allows skipping the IPv6 headers to access directly the host-by-host header for other purposes. This format is useful for classifying packets that are not TCP or UDP, and have no source and destination ports.

Disadvantages:

IPv6 header does not include "Total Headers Length" field. So introducing this new field in the Flow Label puts extra computation to be done that may result in the processing delays.

Including "Length of IPv6 headers" in the Flow Label does not carry any significance in case ESP is used for IP Security.

This approach is not preferred because of the reasons given above. Again, it does not carry any direct advantage in keeping the "Length of IPv6 headers" in the Flow Label.

A.3. Recent works in progress

An emerging packet switched QoS approach for providing end-to-end quality of service transparent to the application programs is in the verge of becoming a realistic solution for the IPv6 based WAN-QoS requirements. Known as MultServ, this approach finds its inspiration from the initiatives and the results of the distributed operating system research. Some fundamental initial work has been done by the IPv6-QoS research group at the Center for Software Development, BITS, Pilani (India).(http://ipv6.bits-pilani.ac.in/ngni/NGNI-MMI-QoS-D4-v1.3-secure.pdf). It is expected that an IETF document shall soon be submitted to the QoS community for their inputs and review of the emergent approach.

A.4. QoS through policy based protocol implementation

For quite sometime now , an interesting and promising approach that is generic in nature has been suggested and even implemented in parts in terms of quality of service. This approach called policy based control protocol has already one standardized protocol known as Common Open Policy Service (COPS). COPS implementation has been available in several newer routers. Ths policy based quality of service framework permits the network administrators to define QoS Policies that explicitly define rules pertaining to handling aggregated flows at a network node known as the Policy Enforcement Point (PEP). The policy servers known as the Policy Decision Point (PDP) computes or determine the exact QoS enforcement action to be taken on the policy-classified packets to be executed at the PEPs. Although very useful, this approach exhibits certain basic flaws. For instance, PDPs could be the point of failures and building redundancy by providing more PDPs may lead to network degradation (due to possible overheads and synchronisation issues) unless it is very carefully designed. [Qos_pol113]

Acutally this policy based QoS solution augments the DiffServ approach, since in this case the PDPs are expected to map the flow information to specific DiffServ traffic conditioning action meta data which is communicated back to PEP; which thereafter uses this information for future processing. However this approach has one advantage that qualifies for an honourable slot in the QoS strategies and that is because such a mechanism does not require the application themselves to be QoS aware. This also happens to be the strong point of the MultServ approach, but it does not operate on the client-server methodology.

The Quality of Service has one aspect called C&A (Charging and Accounting) which the commercial providers of the service require to support in case they have to charge their customers on the basis of QoS requirements. As of now, most of these service providers either do not provide QoS or provide certain flat tariff rates based on the explicit choices made by the customers that requires the customers to be QoS aware. All this is due to the fact that there is no C&A provision in the majority of the proposed mechanisms pertaining to QoS.

The management of the QoS capable networks (QoS WANs) is yet another area that has not been adequately addressed by most of the existing proposed QoS mechanisms (with or without IPv6). The key problem here is that since the routers do offer a variety of packet handling mechanisms, the operator has to specifically select and combine the required traffic conditioning components at the Edge Routers and even at the Core Routers at the service provider's end. Although the aggregated end-to-end flow can be implemented in such cases, the task to define the exact router configuration remains an increasing complex job particularlyy in wide area heterogeneous networks. A related issue is scalability of management of such QoS-capable networks.

The abovementioned issues are the two areas that are specifically being attempted to be addressed as built-in features of the MultServ quality of service mechanism, which may eventually be implemented in IPv6 WANs and which will not require any major change in the basic protocol itself.

Acknowledgements

Authors acknowledge technical inputs and support from the members of the "Project IPv6@BITS" as well as the graduate students registered in EA C451 Internetworking Technology course at the Birla Institute of Technology & Science, Pilani, India, Dr. Latif Ladid of Ericsson Telebit, (Luxembourg); Dr. Torsten Braun of University of Bern (Switzerland); Dr. Pascal Lorenz of I.U.T. at the University of Haute Alsace, Colmar (France); Dr. S. Rao of Telscom A.G. (Switzerland); Dr. Bernardo Martinez of Versaware Inc. (Spain); Dr. Juan Quemada of UPM, Madrid (Spain); Dr. Merce and Dr. Paulo Desousa at the EC; Dr. Zoubir Mammeri of IRIT (France) and Dr. Brian Carpenter of IBM. The IPv6-QoS team wishes to explicitly acknowledge the support from Dr. S.Venkateswaran of BITS, Pilani (India).

Authors gratefully acknowledge the works of many dedicated brains at the IETF, ETSI and elsewhere, sections or extracts of which have helped us to shape this document.

References

[RFC 2460]   S. Deering and Bob Hinden, "The Internet Protocol Specification", RFC 2460, Internet Protocol version 6 Specification.

[RFC 1809]   C. Partridge, RFC 1809, "Using the Flow Label Field in IPv6".

[RFC 2676]   RFC 2676, QoS Routing Mechanisms and OSPF Extensions.

[RFC 1633]   RFC 1633, Integrated Services in the Internet Architecture: an overview.

[RFC 2475]   RFC 2475, An Architecture for Differentiated Services.

[RFC 2676]   RFC 2676, QoS Routing Mechanisms and OSPF Extensions.

[Qos_pol113] QoS Forum: "Whitepaper in QoS Policy", available at the URL: http://www.gt-er.cg.org.br/sgt-qos/documents/qospol_v11.pdf

References to the works in progress

[draft-banerjee-ipv6-quality-service-02.txt] Rahul Banerjee, N.Preethi, M. Sethuraman, "Design and Implementation of the Quality-of-Service in IPv6 using the modified Hop-by-Hop Extension header - A Practicable Mechanism".

[draft-conta-ipv6-flow-label-02.txt] A. Conta, B. Carpenter, "A proposal for the IPv6 Flow Label".

[draft-rajahalme-ipv6-flow-label-00.txt] J. Rajahalme, A. Conta, "An IPv6 Flow Label Specification".

[draft-banerjee-flowlabel-ipv6-qos-02.txt] Rahul Banerjee, Sumeshwar Paul Malhotra, Mahaveer M, "A Modified Specification for use of the IPv6 Flow Label for providing an efficient Quality of Service using a hybrid approach".

[draft-jagadeesan-rad-approach-service-01.txt] Harshavardhan Jagadeesan, Tuhina Singh, "A Radical Approach in providing Quality-of-Service over the Internet using the 20-bit IPv6 Flow Label field".

[NGNI-MMI-QoS: D1] Rahul Banerjee (BITS), Juan Quemda (UPM), P. Lorenz (UHA), Torsten Braun (UoB), Bernardo Martinez (Versaware): "Use of Various Parameters for Attaining QoS in IPv6-based Multimedia Internetworks", Feb. 2002 readily available at the URL: http://ipv6.bits-pilani.ac.in/ngni/.

[NGNI-MMI-QoS: D3] Rahul Banerjee (BITS), Juan Quemada (UPM), P. Lorenz (UHA), Torsten Braun (UoB), Bernardo Martinez (Versaware): "Quality of Service Directions, Bench Marking and Roadmaps for IPv6 Oriented NGN Multimedia Internetworks". http://ipv6.bits-pilani.ac.in/ngni/.

[NGNI-MMI-QoS: D4] Rahul Banerjee (BITS), Juan Quemada (UPM), P. Lorenz (UHA), Torsten Braun (UoB), Bernardo Martinez (Versaware): http://ipv6.bits-pilani.ac.in/ngni/NGNI-MMI-QoS-D4-v1.3-secure.pdf

Disclaimer

correct or incorrect implementation or use of this specification.

Authors Information

Rahul Banerjee
3256, Center for Software Development
BITS, Pilani û 333031, Rajasthan, India.
Phone: +91-159-7645073 Ext. 335
Email: rahul@bits-pilani.ac.in

Sumeshwar Paul Malhotra
3256, Center for Software Development
BITS, Pilani û 333031, Rajasthan, India.
Email: f1998035@bits-pilani.ac.in

Mahaveer M
3256, Center for Software Development
BITS, Pilani û 333031, Rajasthan, India.
Email: f1998046@bits-pilani.ac.in

Full Copyright Statement