

IPv6 Working Group
Internet Draft
draft_qos.txt

Rahul Banerjee
N. Preethi
M. Sethuraman
BITS, Pilani (India)
January 2002

Design and Implementation of the Quality-of-Service in
IPv6 using the modified Hop-by-Hop Extension header -
A Transitional Mechanism

Status of This Memo

This document is an Internet Draft and is subject to all provisions of Section 10 of RFC 2026. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet Drafts

Internet Drafts are draft documents valid for a maximum of 6 months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet Drafts as reference material or to cite them other than as a "work in progress".

The list of current Internet Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright (C) The Internet Society (2002). All Rights Reserved.

Abstract

This paper proposes a temporary solution to the QoS implementation in IPv6, the design of which uses the Hop-by-Hop Extension header and not the 20-bit flow label field in the IPv6 main header. This paper deals extensively with Integrated Services type of QoS model (like the one supported by RSVP) and gives the definition of the important TLV options that will be needed to specify the Type of QoS and the corresponding resource requirements in the Hop-by-Hop Extension Header. This design can also support the Differentiated Services type of QoS model, which is dealt in a brief manner. The paper also elaborates on the data structures that will be required at the routers and finally gives the algorithm that the source and the router should follow while trying to implement this design.

Table of Contents

1. Introduction.....	4
2. Motivation for using the Hop-by-Hop extension header for implementing QoS.....	4
3. The Hop-by-Hop extension header.....	4
4. Type - Length - Value (TLV) Options.....	5
4.1 Introduction.....	5
4.2 Already defined TLV options.....	6
4.2.1 Pad1 option.....	6
4.2.2 PadN option.....	6
4.2.3 The router alert option.....	6
5. Using the TLV options to implement QoS.....	7
5.1 QoS models and their representation in the options field.....	7
5.2 The IntServ Model.....	8
5.2.1 The QoS Identifier.....	8
5.2.2 Resource Identifier.....	9
5.2.3 Resource required list.....	10
5.3 The Different TYPES OF FLOW in the IntServ Model.....	10
5.3.1 Guaranteed Flow Service.....	10
5.3.1.1 Option Definition.....	10
5.3.1.2 Explanation.....	11
5.3.1.3 The Resource Required List and its Specification.....	11
5.3.1.4 Examples.....	11
5.3.2 Controlled Load Service.....	11
5.3.2.1 Option Definition.....	12
5.3.2.2 Explanation.....	12
5.3.2.3 The Resource Required List and its Specification.....	12
5.3.2.4 Examples.....	12
5.4 Overview of some important facts.....	13
5.5 No Flow Management.....	13
5.5.1 Option Definition.....	13
6. At the Router.....	13
6.1 The AllottedQoS table.....	14
6.2 Resource Required List.....	14
6.3 Defining the different Resource Identifiers.....	14
6.4 Template for the AllottedQoS table entry.....	14
7. Overview of the whole design.....	15
7.1 Function of the source.....	15
7.2 Function of each relevant intermediate router.....	15
7.2.1 Initial Processing.....	15
7.2.2 Searching for the entry.....	15
7.2.3 New Entry.....	15
8. Conclusion.....	16

Acknowledgements.....17
References.....17
Disclaimer.....17
Author Information.....17
Full Copyright Statement.....18

1. Introduction

This paper talks about the design and gives an overview of the implementation details of Quality of Service (QoS) in IPv6. Though IPv6 main header has a 20-bit flow label field for QoS implementation purposes, it has not yet been exploited. This paper doesn't talk of using those 20 bits but explores the possibility of using the hop-by-hop extension header to implement QoS. This design is based on the Integrated Services model and can act as an effective temporary solution till the specifications to use the 20-bit flow control field in the IPv6 main header is developed.

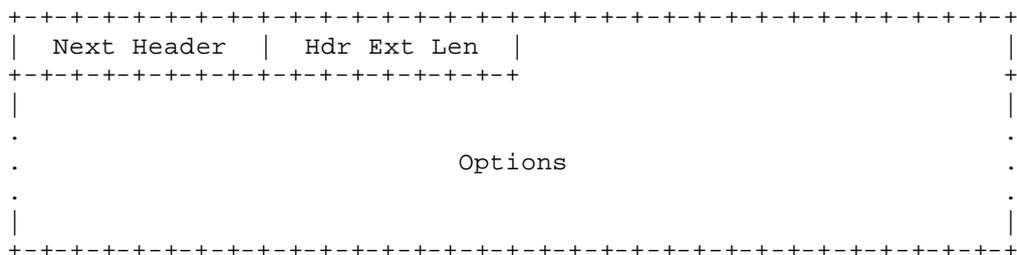
2. Motivation for using the hop-by-hop extension header implementing QoS

To implement any model of QoS, all the routers en-route have to be requested for the particular resources required and it is important that they give their consent on the same. The hop-by-hop extension header is one that will be processed by all the routers en-route to the destination. So all the routers in the path will see any information that is embedded in this header.

The TLV options in the hop-by-hop extension header have not yet been fully exploited. By exploiting those options to our convenience, it is possible to specify the requisite information for each flow (i.e. the type and the resources required) to all the intermediate routers. The individual routers can send appropriate messages to the source if it cannot meet the resource requirements.

3. The Hop-by-Hop Extension header

Before using the Hop-by-Hop Extension header, it is necessary to know about it in detail. According to RFC 2460 - the formal Specification for IPv6, the Hop-by-Hop Extension Header is used to carry optional information that must be examined by every node along a packet's delivery path. It is identified by a Next Header value of 0 (Zero) in the IPv6 header, and has the following format:



The third-highest-order bit of the Option Type specifies whether or not the Option Data of that option can change en-route to the packet's final destination.

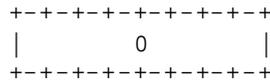
- 0 - Option Data does not change en-route
- 1 - Option Data may change en-route

The three high-order bits described above are treated as part of the Option Type, not independent of the Option Type. That is, a full 8-bit Option Type, not just the low-order 5 bits of an Option Type, identifies a particular option.

4.2 The Already defined TLV options

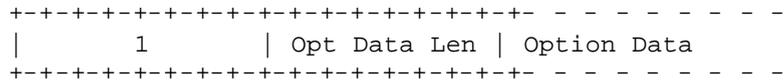
The only hop-by-hop options defined in RFC 2460 (IPv6 Specification) are the Pad1 and PadN options specified as follows:

4.2.1 Pad1 option



The format of the Pad1 option is special in the sense that it does not have length and value fields. The Pad1 option is used to insert one octet of padding into the Options area of a header. [RFC 2460].

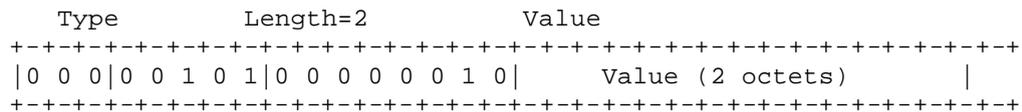
4.2.2 PadN option



The PadN option is used to insert two or more octets of padding into the Options area of a header. For N octets of padding, the Opt Data Len field contains the value N-2, and the Option Data consists of N-2 zero-valued octets. [RFC 2460]

4.2.3 The router alert option

This option has been defined in the RFC 2711 - titled Router Alert Option and has the following format:



The first three bits of the first byte are zero and the value 5 in the remaining five bits is the Hop-by-Hop Option Type number. By zeroing all three, this specification requires that, nodes not recognizing this option type should skip over this option and continue processing the header and that the option must not change en route.

The above 3 are the options that have been defined in RFCs. The rest of the values for the option type of the hop-by-hop options header haven't been defined yet. [RFC 2711]

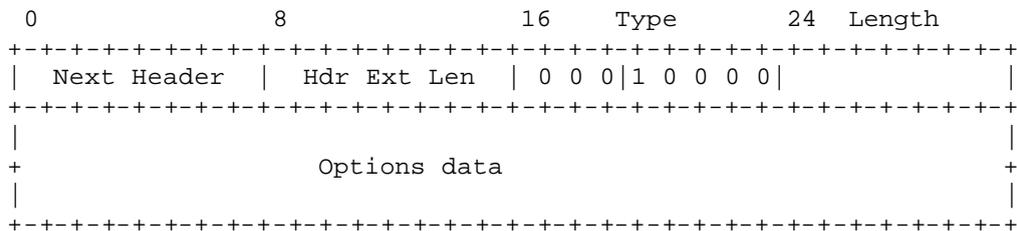
5. Using the TLV options to implement QoS

This design hopes to exploit the remaining non-defined and possible values of the option type in the Hop-by-Hop options header, (after leaving some values for future use) to indicate some important QoS types.

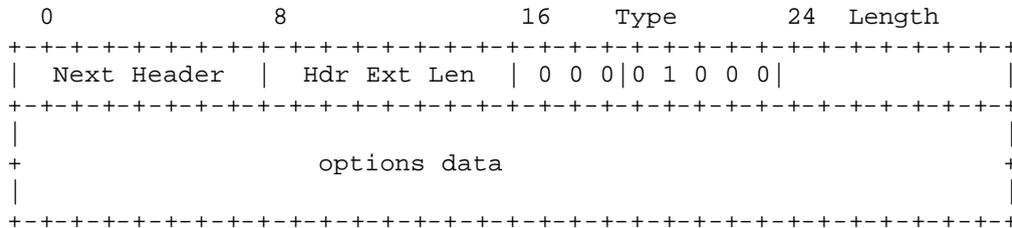
5.1 QoS Models and their representation in the options field

Since this work focuses to provide a transitional mechanism for providing QoS-support (by complementing the 20-bit flow control field in the IPv6 base header), it deals with a Integrated Services (IntServ) model like that supported by RSVP [Paul et al.], wherein each and every flow needs to specify its TYPE and the RESOURCES that it needs en-route. (This design can also support the Differentiated Services (DiffServ) model of QoS, in which case each flow is aggregated to a particular class of traffic. This design can then act as a substitute for the concept behind the Traffic Class bits (8-bit field) in the IPv6 base header.)

The source tells the routers that it is using the Integrated Services model by setting the nineteenth bit of the first 32 bits.



The Differentiated Services (DiffServ) feature, if required, can be specified by setting the twentieth bit of the first 32 bits.

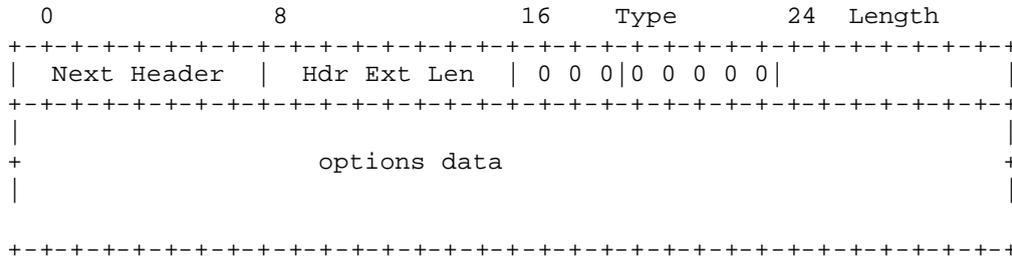


This report talks extensively of the IntServ model only and hopes that the DiffServ model gets developed in the future.

5.2 The IntServ Model

The two main Types of flows in the IntServ model are [Paul et al]
Guaranteed flow service
Controlled Load Service

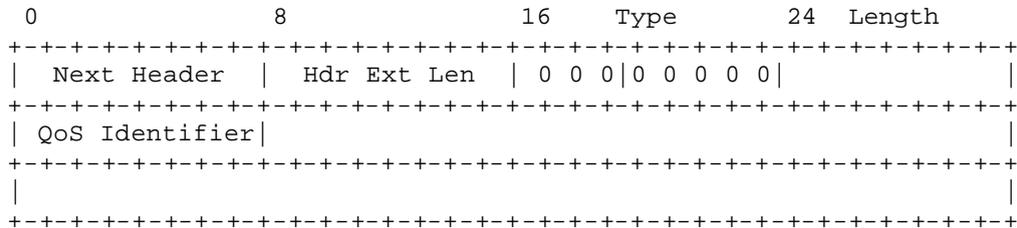
The last three bits of the Type field i.e. the bits numbered 21, 22, 23 are used to represent one of these types.



There are a total of 8 possible combinations out of which the IntServ model uses two. The rest can be can be exploited by the DiffServ model and future use.

5.2.1 The QoS Identifier

This is an 8-bit identifier and occupies the first byte in the options data field as shown in the figure below. There might be many applications from the same source wherein each one has its own flow specifications. So there arises a need to uniquely identify each such flow. The QoS identifier does this job. A particular source can establish a maximum of 256 connections that need QoS guarantee.



5.2.2 Resource Identifier

This is a 4-bit identifier that specifies the type of the resource needed by a particular flow. The different types of resources needed are indicated using these identifiers in a list. This list follows the QoS Identifier in the option data field, which in turn is followed by a list of 32 bit values that specify the amount of resource required for each of the resource types. Some of the identified resource types are:

0000 - End of List Identifier

This is a special identifier that specifies the end of the resource-required list (brief explanation in section 5.2.3).

0001 - Constant Data Transfer Rate

This identifies the Constant Bandwidth required and the value is given in a 32-bit field specified in Kbps (Kilo bits per second). (Max value = 512 GBps)

0010 - Average Data Transfer Rate

This identifies the Average Bandwidth required and the value is given in a 32-bit field in Kbps (Kilo bits per second).

0011 - Maximum Data Transfer Rate

This identifies the Maximum Bandwidth required and the value is given in a 32-bit field specified in Kilobits per second (Kbps).

0100 - Minimum Delay Requirement

This identifies the Minimum Delay that the application demands and the required value is given in a 32-bit field specified in nanoseconds. (Max value = 4.3 sec)

0101 - Average Delay Requirement

This identifies the Average end-to-end delay that the application can tolerate and the value is given in a 32-bit field specified in nanoseconds.

5.3.1.2 Explanation

The first 3 bits being 1,0,0 say that if the router is not able to recognize the option type, it should discard the packet and, regardless of whether or not the packet's Destination Address was a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type and the value of the option data field should not be changed en route by any routers.

The value of 18 in the 5 bits numbered 19,20,21,22,23 defines this QoS type of IntServ and Guaranteed Service. The numeric decimal value specifying this type is 146.

5.3.1.3 The Resource Required List and its Specification

- a. Constant Bandwidth Requirement: The bit value of 0001 after the QoS identifier is the identifier for this and the first 32-bit value gives the amount of bytes to be reserved.
- b. Minimum delay Requirement: The deterministic minimal delay nanoseconds. The identifier is 0010 and the second 32-bit value corresponds to this.
The 0000 identifier ends this list.

5.3.1.4 Examples

Interactive applications like Videoconferencing/Audio Conferencing or real time applications.

5.3.2) Controlled Load Service

This service is meant for RTT (Real Time tolerant) or soft Real Time applications, which have an average bandwidth requirement and an indeterminate end-to-end delay for an arbitrary packet. [Paul et al]. These RTI applications demand weak bounds on the maximum delay over the network. Occasional packet loss is acceptable. For example, consider video applications which use buffering.

The required resource reservations can be

- a. Average bandwidth for the application traffic
- b. Buffer requirement at each relevant intermediate router

2. QoS identifier for that particular flow from the source.
3. Information regarding whether it is the IntServ Model or the Diffserv Model.
Enum MODEL_ID{
INTSERV=0, // the IntServ Model
DIFFSERV=1 // the DiffServ Model
};
4. List of resources allotted to that entry (i.e.) an array of values like the following.
Struct RESOURCE_ALLOCATED{
Short int Res_identifier; //the 4 bit identifier of the resource
Int Res_allocated; //the 32 bit value of the allocated resource
};

6.2 Resource Required List

The list of resources will be an array of pointers to the structure RESOURCE_ALLOCATED as declared below.

```
Struct RESOURCE_ALLOCATED *res_allocated[MAX];
```

This array will be maintained for each source address. The QoS Identifier will be the array subscript for each source. The pointer value stored acts as the head of the list of the resources allotted for that particular QoS identifier.

6.3 Defining the different Resource Identifiers

```
enum RES_ID{  
ENDOFLLIST=0, // End of List Identifier  
CONSTBW =1, // Constant Data Transfer Rate  
AVBW =2, // Average Data Transfer Rate  
MAXBW =3, // Maximum Data Transfer Rate  
MINDELAY =4, // Minimum Delay Requirement  
AVDELAY =5, // Average Delay Requirement  
BUFFREQ =6 // Buffer Requirement  
};
```

6.4 Template for the AllottedQos table entry

```
#define MAX 256 //maximum of 256 QoS Ids for every source  
typedef struct {  
struct sockaddr_in6 *srcaddr; //the source IPv6 address  
struct RESOURCE_ALLOCATED *res_allocated[MAX]; //a pointer which  
//acts as the head for each of the lists i.e. for each of the  
//0..MAX QoS Identifiers for the particular source address.  
MODEL_ID model; // IntServ or DiffServ  
}ALLOTTEDQOS_TABLE;
```

7. Overview of the whole design.

This section describes the whole process by taking an example. Consider any application (like Videoconferencing or Video/Audio on Demand) that needs some specified QoS.

7.1 Function of the Source

It gets a unique QoS Identifier for that particular flow and fills it in the Hop-by-Hop header.

It specifies the IntServ model by setting the appropriate bit.

The source application then fills in the resource-required list and the corresponding 32 bit values (the amount of each resource needed) in the options data part of the Hop-by-Hop header.

This packet is put on the network and it reaches the intermediate routers.

7.2 Function of each relevant intermediate router

7.2.1 Initial Processing

It gets the option type value from the header.

Checks if its the default (no QoS required which is indicated by a value of all bits being 0 in the 5 bits numbered 19,20,21,22,23

If its not the default QoS, it gets the QoS identifier from the first byte of the options data field.

7.2.2 Searching for the entry

1. The ALLOTTED_QOS 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 QoS Identifier got during the Initial Processing stage. (the array index for the res_allocated structure is the corresponding QoS Identifier and this pointer is NULL if its a new entry).
3. If the entry already exists, the IPv6 packet is processed so that the reserved QoS is met.
4. If the entry is not found, a new entry is made in the ALLOTTED_QOS table for the source and the QoS Identifier and further processing of this new entry is done as follows.

7.2.3 New Entry

1. The router now checks if its the IntServ Model or the Diffserv Model by checking the appropriate bits in the options type field and stores this information in the model variable of type MODEL_ID in the ALLOTTED_QOS table.

2. The router then gets the Resources required list and their corresponding values from the options data field and updates the res_allocated array structure.
3. It then checks with the QoS Routing table, to find out if this reservation is possible. If yes, it updates the new entry in the ALLOTTED_QOS table in the memory or else this entry is removed.
4. 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 happens at all the intermediate routers between the source and the destination.

8. Conclusion

This work has dealt extensively with the design of the Integrated Services model of Quality of Service in IPv6 using the Hop-by-Hop Extensions Header. This is being suggested primarily as a transitional mechanism / solution although it has a definite potential to qualify as an effective QoS support measure.

Acknowledgements

Authors acknowledge technical inputs and support from the members of the "Project IPv6@BITS" at the Birla Institute of Technology and Science, Pilani, India, Dr. Latif Ladid of Ericsson Telebit, (Luxembourg); Dr. Torstern 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) and Dr. Merce and Dr. Paulo at the EC.

References

- [RFC 2460] RFC 2460, Internet Protocol version 6 Specification.
- [RFC 2117] RFC 2117, Router Alert Option in IPv6.
- [Paul et al] QoS in Data Networks, Protocols and Standards by Arindam Paul.
- [RFC 2676] RFC 2676, QoS Routing Mechanisms and OSPF Extensions.
- [NGNI-MMI-QoS: D1] Rahul Banerjee (BITS), Juan Quemada (UPM), P. Lorenz (UHA), Torsten Braun (UoB), Bernardo Martinez (Versaware): "Use of Various Parameters for Attaining QoS in IPv6-based Multimedia Internetworks", Nov. 15, 2001 available at <http://IPv6.bits-pilani.ac.in/ngni/>.

Disclaimer

The views and specification here are those of the authors and are not necessarily those of their employers. The authors and their employers specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this specification.

Author Information

Rahul Banerjee
3256, Centre for Software Development
BITS, Pilani - 333031
Rajasthan, India.

Phone: +91-159-7645073 Ext. 335
Email: rahul@bits-pilani.ac.in

Full Copyright Statement

Copyright (C) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.