QoS-NSLP QSPEC Template

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

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 six 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 "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

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

This Internet-Draft will expire on June 21, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).
Abstract

The QoS NSLP protocol is used to signal QoS reservations and is independent of a specific QoS model (QOSM) such as IntServ or DiffServ. Rather, all information specific to a QOSM is encapsulated in a separate object, the QSPEC. This draft defines a template for the QSPEC, which contains both the QoS description and QSPEC control information. The QSPEC format is defined, as are a number of QSPEC parameters. The QSPEC parameters provide a common language to be re-used in several QOSMs. To a certain extent QSPEC parameters ensure interoperability of QOSMs. Optional QSPEC parameters aim to ensure the extensibility of QoS NSLP to other QOSMs in the future.

The node initiating the NSIS signaling adds an Initiator QSPEC that must not be removed, thereby ensuring the intention of the NSIS initiator is preserved along the signaling path.

Table of Contents

1. Conventions Used in This Document .......................... 3
2. Introduction .................................................... 3
3. Terminology ....................................................... 4
4. QSPEC Parameters, Processing, & Extensibility ................. 6
   4.1 QSPEC Parameters ............................................ 6
   4.2 QSPEC Processing ........................................... 6
   4.3 Example of NSLP/QSPEC Operation .......................... 8
   4.4 Treatment of QSPEC Parameters ............................. 11
      4.4.1 Mandatory and Optional QSPEC Parameters ............ 11
      4.4.2 Read-only and Read-write QSPEC Parameters .......... 11
   4.5 Inability to handle parameters ............................ 12
      4.5.1 Error Conditions ....................................... 12
      4.5.2 Inability to interpret and update parameters ........ 12
   4.6 QSPEC Extensibility ......................................... 13
5. QSPEC Format Overview .......................................... 13
   5.1 QSPEC Control Information ................................ 14
   5.2 QoS Description ............................................. 14
      5.2.1 <QoS Desired> ......................................... 14
      5.2.2 <QoS Available> ...................................... 16
      5.2.3 <QoS Reserved> ........................................ 18
      5.2.4 <Minimum QoS> ....................................... 18
6. QSPEC Procedures & Examples ................................... 18
   6.1 QSPEC Procedures ........................................... 18
      6.1.1 Sender-Initiated Reservations ........................ 19
      6.1.2 Receiver-Initiated Reservations ...................... 20
      6.1.3 Resource Queries .................................... 21
      6.1.4 Bidirectional Reservations ........................... 21
   6.2 QSPEC Examples .............................................. 22
7. QSPEC Functional Specification ................................ 24
   7.1 General QSPEC Formats .................................... 24
   7.2 Parameter Coding ........................................... 27
      7.2.1 <NON QOSM Hop> Parameter ............................ 27
      7.2.2 <Excess Treatment> Parameter ......................... 27
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Introduction

The QoS NSLP establishes and maintains state at nodes along the path of a data flow for the purpose of providing forwarding resources (QoS) for that flow [QoS-SIG]. The design of QoS NSLP is conceptually similar to RSVP [RFC2205], and meets the requirements of [RFC3726].

A QoS-enabled domain supports a particular QoS model (QOSM), which is a method to achieve QoS for a traffic flow. A QOSM incorporates QoS provisioning methods and a QoS architecture. It defines the behavior of the resource management function (RMF), including inputs and outputs, and how QSPEC information is interpreted on traffic.
description, resources required, resources available, and control information required by the RMF. A QOSM also specifies a set of mandatory and optional QSPEC parameters that describe the QoS and how resources will be managed by the RMF. QoS NSLP can support signaling for different QOSMs, such as for IntServ, DiffServ admission control, and those specified in [Y.1541-QOSM, IntSERV-QOSM, RMD-QOSM]. For more information on QOSMs see Section 7.2 and Appendix A.

One of the major differences between RSVP and QoS-NSLP is that QoS-NSLP supports signaling for different QOSMs along the data path, all with one signaling message. For example, the data path may start in a domain supporting DiffServ and end in a domain supporting Y.1541. However, because some typical QoS parameters are standardized and can be reused in different QOSMs, some degree of interoperability between QOSMs exists.

The QSPEC travels in QoS-NSLP messages and is opaque to the QoS NSLP. It is only interpreted by the RMF. The content of the QSPEC is QOSM specific. However, the mandatory parameters in the QSPEC MUST be interpreted by all QNEs, independent of which QOSM they support. Since QoS-NSLP signaling operation can be different for different QOSMs, the QSPEC contains two kinds of information, QSPEC control information and QoS description.

QSPEC control information contains parameters that govern the RMF. An example of QSPEC control information is how the excess traffic is treated in the RMF queuing functions.

The QoS description is composed of QSPEC objects loosely corresponding to the TSpec, RSpec and AdSpec objects specified in RSVP. This is, the QSPEC may contain a description of QoS desired and QoS reserved. It can also collect information about available resources. Going beyond RSVP functionality, the QoS description also allows indicating a range of acceptable QoS by defining a QSPEC object denoting minimum QoS. Usage of these QSPEC objects is not bound to particular message types thus allowing for flexibility. A QSPEC object collecting information about available resources MAY travel in any QoS-NSLP message, for example a QUERY message or a RESERVE message.

3. Terminology

   Mandatory QSPEC parameter: QSPEC parameter that a QNI SHOULD populate if applicable to the underlying QOSM and a QNE MUST interpret, if populated.

   Minimum QoS: Minimum QoS is a QSPEC object that MAY be supported by any QNE. Together with a description of QoS Desired or QoS Available, it allows the QNI to specify a QoS range, i.e. an upper and lower bound. If the QoS Desired cannot be reserved, QNEs are going to decrease the reservation until the minimum QoS is hit.
Optional QSPEC parameter: QSPEC parameter that a QNI SHOULD populate if applicable to the underlying QOSM, and a QNE SHOULD interpret if populated and applicable to the QOSM(s) supported by the QNE. (A QNE MAY ignore if it does not support a QOSM needing the optional QSPEC parameter).

QNE: QoS NSIS Entity, a node supporting QoS NSLP.

QNI: QoS NSIS Initiator, a node initiating QoS-NSLP signaling.

QNR: QoS NSIS Receiver, a node terminating QoS-NSLP signaling.

QoS Description: Describes the actual QoS in QSPEC objects QoS Desired, QoS Available, QoS Reserved, and Minimum QoS. These QSPEC objects are input or output parameters of the RMF. In a valid QSPEC, at least one QSPEC object of the type QoS Desired, QoS Available or QoS Reserved MUST be included.

QoS Available: QSPEC object containing parameters describing the available resources. They are used to collect information along a reservation path.

QoS Desired: QSPEC object containing parameters describing the desired QoS for which the sender requests reservation.

QoS Model (QOSM): A method to achieve QoS for a traffic flow, e.g., IntServ Controlled Load. A QOSM specifies a set of mandatory and optional QSPEC parameters that describe the QoS and how resources will be managed by the RMF. It furthermore specifies how to use QoS NSLP to signal for this QOSM.

QoS Reserved: QSPEC object containing parameters describing the reserved resources and related QoS parameters, for example, bandwidth.

QSPEC Control Information: Control information that is specific to a QSPEC, and contains parameters that govern the RMF.

QSPEC: QSPEC is the object of QoS-NSLP containing all QOSM-specific information.

QSPEC parameter: Any parameter appearing in a QSPEC; includes both QoS description and QSPEC control information parameters, for example, bandwidth, token bucket, and excess treatment parameters.

QSPEC Object: Main building blocks of QoS Description containing a QSPEC parameter set that is input or output of an RMF operation.

Resource Management Function (RMF): Functions that are related to resource management, specific to a QOSM. It processes the QoS
description parameters and QSPEC control parameters.

Read-only Parameter: QSPEC Parameter that is set by initiating or responding QNE and is not changed during the processing of the QSPEC along the path.

Read-write Parameter: QSPEC Parameter that can be changed during the processing of the QSPEC by any QNE along the path.

4. QSPEC Parameters, Processing, & Extensibility

4.1 QSPEC Parameters

The definition of a QOSM includes the specification of how the requested QoS resources will be described and how they will be managed by the RMF. For this purpose, the QOSM specifies a set of QSPEC parameters that describe the QoS and QoS resource control in the RMF. A given QOSM defines which of the mandatory and optional QSPEC parameters it uses, and it MAY define additional optional QSPEC parameters. Mandatory and optional QSPEC parameters provide a common language for QOSM developers to build their QSPECs and are likely to be re-used in several QOSMs. Mandatory and optional QSPEC parameters are defined in this document, and additional optional QSPEC parameters can be defined in separate documents.

As defined in Section 4.6, additional optional QSPEC parameters can be defined in separate Informational documents specific to a given QOSM. For example, optional QSPEC parameters are defined in [RMD-QOSM] and [Y.1541-QOSM].

4.2 QSPEC Processing

The QSPEC is opaque to the QoS-NSLP processing. The QSPEC control information and the QoS description are interpreted and MAY be modified by the RMF in a QNE (see description in [QoS-SIG]).

A QNE MUST support at least one QOSM. A QoS-enabled domain supports a particular QOSM, e.g. DiffServ admission control. If this domain supports QoS-NSLP signaling, its QNEs MUST support the DiffServ admission control QOSM. The QNEs MAY also support additional QOSMs.

The QSPEC contains a QOSM ID, i.e. information on what QOSM is being signaled by the QNI. However, if a QSPEC arrives at a QNE that does not support the QOSM being signaled, it can still understand the QSPEC content, at least to a basic degree. This is because mandatory parameters have been defined as a common language. Therefore, a QNE MUST at least interpret all the mandatory parameters in a QSPEC even if it does not support the corresponding QOSM.

A QoS NSLP message can contain a stack of at most 2. The first on the stack is the Initiator QSPEC. This is a QSPEC provided by the
QNI, which travels end-to-end, and therefore the stack always has at least depth 1. QSPEC parameters MUST NOT be deleted from or added to the Initiator QSPEC. In addition, the stack MAY contain a Local QSPEC stacked on top of the Initiator QSPEC. A QNE only considers the topmost QSPEC.

At the ingress edge of a local QoS domain, a Local QSPEC MAY be pushed on the stack in order to describe the requested resources in a domain-specific manner. Also, the Local QSPEC is popped from the stack at the egress edge of the local QoS domain.

This draft provides a template for the QSPEC, which is needed in order to help define individual QOSMs and in order to promote interoperability between QOSMs. Figure 1 illustrates how the QSPEC is composed of QSPEC control information and QoS description. QoS description in turn is composed of up to four QSPEC objects (not all of them need to be present), namely QoS Desired, QoS Available, QoS Reserved and Minimum QoS. Each of these QSPEC Objects, as well as QSPEC Control Information, consists of a number of mandatory and optional QSPEC parameters.

```
+-------------+---------------------------------------+
| QSPEC Control|              QoS                      |
| Information  |           Description                 |
+-------------+---------------------------------------+
\________________ ______________________/ 
V
\________________ ______________________/  
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
\________________ ______________________/ 
V
```

Figure 1: Structure of the QSPEC

The internal structure of each QSPEC object and the QSPEC control information, with mandatory and optional parameters, is illustrated in Figure 2.

```
+------------------+-----------------+---------------+
| QSPEC/Ctrl Info  | Mandatory QSPEC | Optional QSPEC |
| Object ID       | Parameters      | Parameters    |
+------------------+-----------------+---------------+
```

Figure 2: Structure of QSPEC Objects & Control Information
4.3 Example of NSLP/QSPEC Operation

This Section illustrates the operation and use of the QSPEC within the NSLP. The example configuration is shown in Figure 3.

+----------+      /-------\       /--------\       /--------\ 
| Laptop   |     |   Home  |     |  Cable   |     | DiffServ |
| Computer |-----| Network |-----| Network  |-----| Network  |----+
+----------+     | No QOSM |     |DQOS QOSM |     | RMD QOSM |
\--------/       \--------/       \--------/     
|    /--------\      +----------+  |
|   |  "X"G    |     | Handheld |
|----| Wireless |-----|  Device |
| XG QOSM |     +----------+
\--------/     

Figure 3: Example Configuration to Illustrate QoS-NSLP/QSPEC Operation

In this configuration, a laptop computer and a handheld wireless device are the endpoints for some application that has QoS requirements. Assume initially that the two endpoints are stationary during the application session, later we consider mobile endpoints. For this session, the laptop computer is connected to a home network that has no QoS support. The home network is connected to a CableLabs-type cable access network with dynamic QoS (DQOS) support, such as specified in the ‘CMS to CMS Signaling Specification’ [CMSS] for cable access networks. That network is connected to a DiffServ core network that uses the RMD QOSM [RMD-QOSM]. On the other side of the DiffServ core network is a wireless access network built on generation "X" technology with QoS support as defined by generation "X". And finally the handheld endpoint is connected to the wireless access network.

We assume that the Laptop is the QNI and handheld device is the QNR.

The QNI will populate an Initiator QSPEC to achieve the QoS desired on the path. In this example we consider two different ways to perform sender-initiated signaling for QoS:

Case 1) The QNI sets <QoS Desired>, <QoS Available> and possibly <Minimum QoS> QSPEC objects in the Initiator QSPEC, and initializes <QoS Available> to <QoS Desired>. Since this is a reservation in a heterogenic network with different QOSMs supported in different domains, each QNE on the path reads and interprets those parameters in the Initiator QSPEC that it needs to implement the QOSM within its domain (as described below). Each QNE along the path checks to see if
<QoS Available> resources can be reserved, and if not, the QNE reduces the respective parameter values in <QoS Available> and reserves these values. The minimum parameter values are given in <Minimum QoS>, if populated, otherwise zero if <Minimum QoS> is not included. If one or more parameters in <Available QoS> fails to satisfy the corresponding minimum values in Minimum QoS, the QNE notifies the QNI and the reservation is aborted. Otherwise, the QNR notifies the QNI of the <QoS Available> for the reservation.

Case 2) The QNI populated the Initiator QSPEC with <QoS Desired>. Since this is a reservation in a heterogenic network with different QOSMs supported in different domains, each QNE on the path reads and interprets those parameters in the Initiator QSPEC that it needs to implement the QOSM within its domain (as described below). If a QNE cannot reserve <QoS Desired> resources, the reservation fails.

In both cases, the QNI populates mandatory and optional QSPEC to ensure correct treatment of its traffic in domains down the path. Since the QNI does not know the QOSM used in downstream domains, it includes values for those mandatory and optional QSPEC parameters it cares about. Let us assume the QNI wants to achieve IntServ-like QoS guarantees, and also is interested in what path latency it can achieve. The QNI therefore includes in the QSPEC the QOSM ID for IntServ Controlled Load Service. The QSPEC objects are populated with all parameters necessary for IntServ Controlled Load and additionally the parameter to measure path latency, as follows:

<QoS Desired> = <Token Bucket>
<QoS Available> = <Token Bucket> <Path Latency>

In both cases, each QNE on the path reads and interprets those parameters in the Initiator QSPEC that it needs to implement the QOSM within its domain. It may need additional parameters for its QOSM, which are not specified in the Initiator QSPEC. If possible, these parameters must be inferred from those that are present, according to rules defined in the QOSM implemented by this QNE.

There are three possibilities when a RESERVE message is received at a QNE at a domain border (we illustrate these possibilities in the example):

- the QNE just leaves the QSPEC as-is.

- the QNE can stack a local QSPEC on top of the Initiator QSPEC (this is new in QoS NSLP, RSVP does not do this).

- the QNE can tunnel the Initiator RESERVE message through its domain and issue its own Local RESERVE message. For this new Local RESERVE message, the QNE acts as the QNI, and the QSPEC in the domain is an Initiator QSPEC. This procedure is also used by RSVP in making aggregate reservations, in which case there is not a new intra-domain
(aggregate) RESERVE for each newly arriving interdomain (per-flow) RESERVE, but the aggregate reservation is updated by the border QNE (QNI) as need be. This is also how RMD works [RMD-QOSM].

For example, at the RMD domain, a local RESERVE with its own RMD Initiator QSPEC corresponding to the RMD-QOSM is generated based on the original Initiator QSPEC according to the procedures described in Section 4.5 of [QoS-SIG] and in [RMD-QOSM]. That is, the ingress QNE to the RMD domain must map the QSPEC parameters contained in the original Initiator QSPEC into the RMD QSPEC. The RMD QSPEC for example needs <Bandwidth> and <QoS Class>. <Bandwidth> is generated from the <Token Bucket> parameter. Information on <QoS Class>, however, is not provided. According to the rules laid out in the RMD QOSM, the ingress QNE infers from the fact that an IntServ Controlled Load QOSM was signaled that the EF PHB is appropriate to set the <PHB Class> parameter. These RMD QSPEC parameters are populated in the RMD Initiator QSPEC generated within the RMD domain.

Furthermore, the node at the egress to the RMD domain updates <QoS Available> on behalf of the entire RMD domain if it can. If it cannot, it raises the parameter-specific, ‘not-supported’ flag, warning the QNR that the final value of these parameters in QoS Available is imprecise.

In the XG domain, the Initiator QSPEC is translated into a Local QSPEC using a similar procedure as described above. The Local QSPEC becomes the current QSPEC used within the XG domain, that is, the it becomes the first QSPEC on the stack, and the Initiator QSPEC is second. This saves the QNEs within the XG domain the trouble of re-translation the Initiator QSPEC. At the egress edge of the XG domain, the translated Local QSPEC is popped, and the Initiator QSPEC returns to the number one position.

If the reservation was successful, eventually the RESERVE request arrives at the QNR (otherwise the QNE at which the reservation failed would have aborted the RESERVE and sent an error RESPONSE back to the QNI). The QNR generates a positive RESPONSE with QSPEC objects <QoS Reserved> - and for case 1 - additionally <QoS Available>. The parameters appearing in <QoS Reserved> are the same as in <QoS Desired>, with values copied from <QoS Available> in case 1, and with the original values from <QoS Desired> in case 2. That is, it is not necessary to transport the <QoS Desired> object back to the QNI since the QNI knows what it signaled originally, and the information is not useful for QNEs in the reverse direction. The <QoS Reserved> object should transport all necessary information, although the <QoS Available> and <QoS Reserved> objects may end up transporting some of the same information.

Hence, the QNR populates the following QSPEC objects:
If the handheld device on the right of Figure 3 is mobile, and moves through different "XG" wireless networks, then the QoS might change on the path since different XG wireless networks might support different QOSMs. As a result, QoS-NSLP/QSPEC processing will have to renegotiate the <QoS Available> on the path. From a QSPEC perspective, this is like a new reservation on the new section of the path and is basically the same as any other rerouting event – to the QNEs on the new path it looks like a new reservation. That is, in this mobile scenario, the new segment may support a different QOSM than the old segment, and the QNI would now signal a new reservation (explicitly, or implicitly with the next refreshing RESERVE message) to account for the different QOSM in the XG wireless domain. Further details on rerouting are specified in [QoS-SIG].

For bit-level examples of QSPECs see the documents specifying QOSMs [INTSERV-QOSM, Y.1541-QOSM, RMD-QOSM].

4.4 Treatment of QSPEC Parameters

4.4.1 Mandatory and Optional QSPEC Parameters

Mandatory and optional QSPEC parameters are defined in this document and are applicable to a number of QOSMs. Mandatory QSPEC parameters are treated as follows:

- A QNI SHOULD populate mandatory QSPEC parameters if applicable to the underlying QOSM.
- QNEs MUST interpret mandatory QSPEC parameters, if populated.

Optional QSPEC parameters are treated as follows:

- A QNI SHOULD populate optional QSPEC parameters if applicable to the QOSM for which it is signaling.
- QNEs SHOULD interpret optional QSPEC parameters, if populated and applicable to the QOSM(s) supported by the QNE. (A QNE MAY ignore the optional QSPEC parameter if it does not support a QOSM needing the optional QSPEC parameter).

4.4.2 Read-only and Read-write QSPEC Parameters

Both mandatory and optional QSPEC parameters can be read-only or read-write. Read-write parameters can be changed by any QNE, whereas read-only parameters are fixed by the QNI and/or QNR. For example in a RESERVE message, all parameters in <QoS Available> are read-write parameters, which are updated by intermediate QNEs. Read-only parameters are, for example, all parameters in <QoS Desired> as sent by the QNI.
QoS description parameters can be both read-only or read-write, depending on which QSPEC object, and which message, they appear in. In particular, all parameters in <QoS Desired> and <Minimum QoS> are read-only for all messages. More details are provided in Sec. 7.1.

In the QSPEC Control Information Object, the property of being read-write or read-only is parameter specific.

4.5 Inability to handle parameters

A QNE may not be able to interpret or update the QSPEC or individual parameters for several reasons. For example, the QSPEC cannot be read or interpreted because it is erroneous, or because of a QNE fault. This is an error condition. Another reason is that a parameter type is unknown because it is optional, or a parameter value in QoS Available cannot be updated because QoS NSLP was tunneled to the QNE. These are not error conditions.

4.5.1 Error Conditions

When an RMF cannot interpret the QSPEC because the coding is erroneous, it raises corresponding flags in the QSPEC. The ‘error flags’ are located in each QSPEC Object and in each parameter. If such a flag is set, at least one QNE along the data transmission path between the QNI and QNR cannot interpret a mandatory or optional QSPEC parameter or the QSPEC object for any reason, such as a protocol error, QNE fault, etc. In this case, more detailed error information may be given in the QoS NSLP error message. That is, if possible the RMF must communicate error details to the QoS NSLP processing. QoS NSLP [QoS-SIG] describes how the erroneous message is handled further.

When the error can be located in a particular parameter, the QNE detecting the error raises the error flag in this parameter. Additionally, it raises the error flag in the corresponding QSPEC Object. If the error cannot be located at the parameter level, only the error flag in the QSPEC object is raised.

4.5.2 Inability to interpret and update parameters

When the QOSM ID is not known to a QNE, it MUST interpret at least the mandatory parameters.

Each optional QSPEC parameter has an associated ‘not-supported flag’. If the not-supported flag is set, then at least one QNE along the data transmission path between the QNI and QNR can not support the specified optional parameter, or perhaps the parameter type is understood but the particular parameter value is not standardized. This means the value collected in the corresponding parameter is a lower bound to the "real" value. A QNE MUST be able to set the
not-supported flag if it does not support the optional parameter.

Each QSPEC parameter has an associated ‘tunneled-parameter flag’. When a RESERVE message is tunneled through a domain, QNEs inside the domain cannot update read-write parameters. The egress QNE in a domain has two choices: either it is configured to have the knowledge to update the parameters correctly. Or it cannot update the parameters. In this case it MUST set the tunneled-parameter flag to tell the QNI (or QNR) that the information contained in the read-write parameter is most likely incorrect (or a lower bound).

The formats and semantics of all flags are given in Section 6.1.

4.6 QSPEC Extensibility

Additional optional QSPEC parameters MAY need to be defined in the future. Additional optional QSPEC parameters are defined in separate Informational documents specific to a given QOSM. For example, optional QSPEC parameters are defined in [RMD-QOSM] and [Y.1541-QOSM].

5. QSPEC Format Overview

QSPEC = <QSPEC Version> <QOSM ID> <QSPEC Control Information>

As described above, the QSPEC contains an identifier for the QOSM, the actual resource description (QoS description) as well as QSPEC control information. Note that all QSPEC parameters defined in the following Sections are mandatory QSPEC parameters unless specifically designated as optional QSPEC parameters.

A QSPEC object ID identifies whether the object is <QSPEC Control Information> or <QoS Description>. As described below, the <QoS Description> is further broken down into <QoS Desired>, <QoS Available>, <QoS Reserved>, and <Minimum QoS> objects. A QSPEC parameter ID is assigned to identify each QSPEC parameter defined below.

<QSPEC Version> identifies the QSPEC version number, and <QOSM ID> identifies the particular QOSM being used by the QNI. The <QOSM ID> tells a QNE which parameters to expect. This may simplify processing and error analysis. Furthermore, it may be helpful for a QNE or a domain supporting more than one QOSM to learn which QOSM the QNI would like to have in order to use the most suitable QOSM. Even if a QNE does not support the QOSM it MUST interpret at least the mandatory parameters. Note that more parameters than required by the QOSM can be included by the QNI. QSPEC version and QOSM IDs are assigned by IANA.
5.1 QSPEC Control Information

QSPEC control information is used for signaling QOSM RMF functions not defined in QoS-NSLP. It enables building new RMF functions required by a QOSM within a QoS-NSLP signaling framework, such as specified, for example, in [RMD-QOSM] and [Y.1541-QOSM].

<QSPEC Control Information> = <NON QOSM Hop> <Excess Treatment>

Note that <NON QOSM Hop> is a read-write parameter. <Excess Treatment> is a read-only parameter.

<NON QOSM Hop> is a flag bit telling the QNR (or QNI in a RESPONSE message) whether or not a particular QOSM is supported by each QNE in the path between the QNI and QNR. A QNE sets the <NON QOSM Hop> flag parameter if it does not support the relevant QOSM specification. If the QNR finds this bit set, at least one QNE along the data transmission path between the QNI and QNR can not support the specified QOSM. In a local QSPEC, <NON QOSM Hop> refers to the QoS-NSLP peers of the local QOSM domain.

The <Excess Treatment> parameter describes how the QNE will process excess traffic, that is, out-of-profile traffic. Excess traffic MAY be dropped, shaped and/or remarked. The excess treatment parameter is initially set by the QNI and is read-only.

5.2 QoS Description

The QoS Description is broken down into the following QSPEC objects:

<QoS Description> = <QoS Desired> <QoS Available> <QoS Reserved>

Of these QSPEC objects, QoS Desired, QoS Available and QoS Reserved MUST be supported by QNEs. Minimum QoS MAY be supported.

5.2.1 <QoS Desired>

<QoS Desired> = <Traffic Description> <QoS Class> <Priority>

These parameters describe the resources the QNI desires to reserve and hence this is a read-only QSPEC object. The <QoS Desired> resources that the QNI wishes to reserve are of course directly related to the traffic the QNI is going to inject into the network. Therefore, when used in the <QoS Desired> object, <Traffic Description> refers to traffic injected by the QNI into the network.

<Traffic Description> = <Bandwidth> <Token Bucket>

<Bandwidth> = link bandwidth needed by flow [RFC 2212, RFC 2215]
Note that the Path MTU Discovery (PMTUD) working group is currently specifying a robust method for determining the MTU supported over an end-to-end path. This new method is expected to update RFC1191 and RFC1981, the current standards track protocols for this purpose.

An application MAY like to reserve resources for packets with a particular QoS class, e.g. a DiffServ per-hop behavior (PHB) [RFC2475], or DiffServ-enabled MPLS traffic engineering (DSTE) class type [RFC3564].

<Preemption Priority> is the priority of the new flow compared with the defending priority of previously admitted flows. Once a flow is admitted, the preemption priority becomes irrelevant. <Defending Priority> is used to compare with the preemption priority of new flows. For any specific flow, its preemption priority MUST always be less than or equal to the defending priority. <Admission Priority> and <RPH Priority> provide an essential way to differentiate flows for emergency services, ETS, E911, etc., and assign them a higher admission priority than normal priority flows and best-effort priority flows.

Appropriate security measures need to be taken to prevent abuse of the <Priority> parameters, see Section 8 on Security Considerations.

[Y.1540] defines packet transfer outcomes, as follows:

Successful: packet arrives within the preset waiting time with no errors

Lost: packet fails to arrive within the waiting time

Errored: packet arrives in time, but has one or more bit errors in the header or payload

Packet Loss Ratio (PLR) = total packets lost/total packets sent

Packet Error Ratio (PER) = total errored packets/total packets sent

<Path Latency>, <Path Jitter>, <Path PLR>, and <Path PER> are optional parameters describing the desired path latency, path jitter and path bit error rate respectively. Since these parameters are cumulative, an individual QNE cannot decide whether the desired path latency, etc., is available, and hence they cannot decide whether a
reservation fails. Rather, when these parameters are included in
<Desired QoS>, the QNI SHOULD also include corresponding parameters
in a <QoS Available> QSPEC object in order to facilitate collecting
this information.

5.2.2 <QoS Available>

<QoS Available> = <Traffic Description> <QoS Class> <Priority>
<Path Latency> <Path Jitter> <Path PLR> <Path PER>
<Ctot> <Dtot> <Csum> <Dsum>

When used in the <QoS Available> object, <Traffic Description> refers
to traffic resources available at a QNE in the network.

The <QoS Available> Object collects information on the resources
currently available on the path when it travels in a RESERVE or QUERY
message and hence in this case this QSPEC object is read-write. Each
QNE MUST inspect all parameters of this QSPEC object, and if
resources available to this QNE are less than what a particular
parameter says currently, the QNE MUST adapt this parameter
accordingly. Hence when the message arrives at the recipient of the
message, <QoS Available> reflects the bottleneck of the resources
currently available on a path. It can be used in a QUERY message,
for example, to collect the available resources along a data path.

When <QoS Available> travels in a RESPONSE message, it in fact just
transports the result of a previous measurement performed by a
RESERVE or QUERY message back to the initiator. Therefore in this
case, <QoS Available> is read-only.

The parameters <Token Bucket> and <Bandwidth> provide information,
for example, about the bandwidth available along the path followed by
a data flow. The local parameter is an estimate of the bandwidth the
QNE has available for packets following the path. Computation of the
value of this parameter SHOULD take into account all information
available to the QNE about the path, taking into consideration
administrative and policy controls on bandwidth, as well as physical
resources. The composition rule for this parameter is the MIN
function. The composed value is the minimum of the QNE’s value and
the previously composed value. This quantity, when composed
end-to-end, informs the QNR (or QNI in a RESPONSE message) of the
minimal bandwidth link along the path from QNI to QNR.

The <Path Latency> parameter accumulates the latency of the packet
forwarding process associated with each QNE, where the latency is
defined to be the mean packet delay added by each QNE. This delay
results from speed-of-light propagation delay, from packet processing
limitations, or both. It does not include any variable queuing delay
that may be present. Each QNE MUST add the propagation delay of its
outgoing link, which includes the QNR adding the associated delay for
the egress link. Furthermore, the QNI MUST add the propagation delay
of the ingress link. The composition rule for the <Path Latency> parameter is summation with a clamp of \((2^{32} - 1)\) on the maximum value. This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the minimal packet delay along the path from QNI to QNR. The purpose of this parameter is to provide a minimum path latency for use with services which provide estimates or bounds on additional path delay [RFC 2212]. Together with the queuing delay bound, this parameter gives the application knowledge of both the minimum and maximum packet delivery delay. Knowing both the minimum and maximum latency experienced by data packets allows the receiving application to know the bound on delay variation and de-jitter buffer requirements.

The <Path Jitter> parameter accumulates the jitter of the packet forwarding process associated with each QNE, where the jitter is defined to be the nominal jitter added by each QNE. IP packet jitter, or delay variation, is defined in [RFC3393], Section 3.4 (Type-P-One-way-ipdv), and where the selection function includes the packet with minimum delay such that the distribution is equivalent to 2-point delay variation in [Y.1540]. The suggested evaluation interval is 1 minute. Note that the method to estimate IP delay variation without active measurements requires more study. This jitter results from packet processing limitations, and includes any variable queuing delay which may be present. Each QNE MUST add the jitter of its outgoing link, which includes the QNR adding the associated jitter for the egress link. Furthermore, the QNI MUST add the jitter of the ingress link. The composition method for the <Path Jitter> parameter is the combination of several statistics describing the delay variation distribution with a clamp on the maximum value (note that the methods of accumulation and estimation of nominal QNE jitter are under study). This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the nominal packet jitter along the path from QNI to QNR. The purpose of this parameter is to provide a nominal path jitter for use with services that provide estimates or bounds on additional path delay [RFC2212]. Together with the <Path Latency> and the queuing delay bound, this parameter gives the application knowledge of the typical packet delivery delay variation.

The <Path PLR> parameter accumulates the packet loss rate (PLR) of the packet forwarding process associated with each QNE, where the PLR is defined to be the PLR added by each QNE. Each QNE MUST add the PLR of its outgoing link, which includes the QNR adding the associated PLR for the egress link. Furthermore, the QNI MUST add the PLR of the ingress link. The composition rule for the <Path PLR> parameter is summation with a clamp on the maximum value (this assumes sufficiently low PLR values such that summation error is not significant). This quantity, when composed end-to-end, informs the QNR (or QNI in a RESPONSE message) of the minimal packet PLR along the path from QNI to QNR. As with <Path Jitter>, the method to estimate <Path PLR> requires more study.
<Ctot>, <Dtot>, <Csum>, <Dsum>: Error terms C and D represent how the element’s implementation of the guaranteed service deviates from the fluid model. These two parameters have an additive composition rule. The error term C is the rate-dependent error term. It represents the delay a datagram in the flow might experience due to the rate parameters of the flow. The error term D is the rate-independent, per-element error term and represents the worst case non-rate-based transit time variation through the service element. If the composition function is applied along the entire path to compute the end-to-end sums of C and D (<Ctot> and <Dtot>) and the resulting values are then provided to the QNR (or QNI in a RESPONSE message). <Csum> and <Dsum> are the sums of the parameters C and D between the last reshaping point and the current reshaping point.

5.2.3 <QoS Reserved>

<QoS Reserved> = <Traffic Description> <QoS Class> <Priority> <S>

These parameters describe the QoS reserved by the QNEs along the data path, and hence the QoS reserved QSPEC object is read-write.

<Traffic Description>, <QoS Class> and <Priority> are defined above.

<S> = slack term, which is the difference between desired delay and delay obtained by using bandwidth reservation, and which is used to reduce the resource reservation for a flow [RFC 2212]. This is an optional parameter.

5.2.4 <Minimum QoS>

<Minimum QoS> = <Traffic Description> <QoS Class> <Priority>

<Minimum QoS> does not have an equivalent in RSVP. It allows the QNI to define a range of acceptable QoS levels by including both the desired QoS value and the minimum acceptable QoS in the same message. It is a read-only QSPEC object. The desired QoS is included with a <QoS Desired> and/or a <QoS Available> QSPEC object seeded to the desired QoS value. The minimum acceptable QoS value MAY be coded in the <Minimum QoS> QSPEC object. As the message travels towards the QNR, <QoS Available> is updated by QNEs on the path. If its value drops below the value of <Minimum QoS> the reservation fails and is aborted. When this method is employed, the QNR SHOULD signal back to the QNI the value of <QoS Available> attained in the end, because the reservation MAY need to be adapted accordingly.

6. QSPEC Procedures & Examples

6.1 QSPEC Procedures

While the QSPEC template aims to put minimal restrictions on usage of
QSPEC objects in <QoS Description>, interoperability between QNEs and between QOSMs must be ensured. We therefore give below an exhaustive list of QSPEC object combinations for the message sequences described in QoS NSLP [QOS-SIG]. A specific QOSM may prescribe that only a subset of the procedures listed below may be used.

### 6.1.1 Sender-Initiated Reservations

Here the QNI issues a RESERVE, which is replied to by a RESPONSE. This response is generated either by the QNR or, in case the reservation was unsuccessful, by a QNE. The following possibilities for QSPEC object usage exist:

<table>
<thead>
<tr>
<th>ID</th>
<th>RESERVE</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QoS Desired</td>
<td>QoS Reserved</td>
</tr>
<tr>
<td>2</td>
<td>QoS Desired, QoS Avail.</td>
<td>QoS Reserved, QoS Avail.</td>
</tr>
</tbody>
</table>

(1) If only QoS Desired is included in the RESERVE, the implicit assumption is that exactly these resources must be reserved. If this is not possible the reservation fails. The parameters in QoS Reserved are copied from the parameters in QoS Desired.

(2) When QoS Available is included in the RESERVE also, some parameters will appear only in QoS Available and not in QoS Desired. It is assumed that the value of these parameters is collected for informational purposes only (e.g. path latency).

However, some parameters in QoS Available can be the same as in QoS Desired. For these parameters the implicit message is that the QNI would be satisfied by a reservation with lower parameter values than specified in QoS Desired. For these parameters, the QNI seeds the parameter values in QoS Available to those in QoS Desired (except for cumulative parameters such as <path latency>).

Each QNE downgrades the parameters in QoS Available according to its current capabilities. Reservations in each QNE are hence based on current parameter values in QoS Available (and additionally those parameters that only appear in QoS Desired). The drawback of this approach is that, if the resulting resource reservation becomes gradually smaller towards the QNR, QNEs close to the QNI have an oversized reservation, possibly resulting in unnecessary costs for the user. Of course, in the RESPONSE the QNI learns what the actual reservation is (from the QoS RESERVED object) and can immediately issue a properly sized refreshing RESERVE. The advantage of the approach is that the reservation is performed in half-a-roundtrip time.

The parameter types included in QoS Reserved in the RESPONSE MUST be the same as those in QoS Desired in RESERVE. For those parameters...
that were also included in QoS Available in RESERVE, their value is copied into QoS Desired. For the other parameters, the value is copied from QoS Desired (the reservation would fail if the corresponding QoS could not be reserved).

All parameters in the QoS Available QSPEC object in the RESPONSE are copied with their values from the QoS Available QSPEC object in the RESERVE (irrespective of whether they have also been copied into QoS Desired). Note that the parameters in QoS Available are read-write in the RESERVE message, whereas they are read-only in the RESPONSE.

(3) this case is handled as case (2), except that the reservation fails when QoS Available becomes less than Minimum QoS for one parameter. If a parameter appears in QoS Available but not in Minimum QoS it is assumed that there is no minimum value for this parameter.

Regarding Control Information, the rule is that all parameters that have been included in the RESERVE message by the QNI MUST also be included in the RESPONSE message by the QNR with the value they had when arriving at the QNR. When traveling in the RESPONSE message, all Control Information parameters are read-only.

6.1.2 Receiver-Initiated Reservations

Here the QNR issues a QUERY which is replied to by the QNI with a RESERVE if the reservation was successful. The QNR in turn sends a RESPONSE to the QNI.

<table>
<thead>
<tr>
<th>ID</th>
<th>QUERY</th>
<th>RESERVE</th>
<th>RESPONSE</th>
</tr>
</thead>
</table>

(1) and (2) The idea is that the sender (QNR in this scenario) needs to inform the receiver (QNI in this scenario) about the QoS it desires. To this end the sender sends a QUERY message to the receiver including a QoS Desired QSPEC object. If the QoS is negotiable it additionally includes a (possibly zero) Minimum QoS, as in Case b.

The RESERVE message includes QoS Available if the sender signaled QoS is negotiable (i.e. it included Minimum QoS). If the Minimum QoS received from the sender is non-zero, the QNR also includes Minimum QoS.

(3) This is the "RSVP-style" scenario. The sender (QNR) issues a QUERY with QoS Desired informing the receiver (QNI) about the QoS it desires as above. It also includes a QoS Available object to collect path properties. Note that here, path properties are collected with
the QUERY message, whereas in the previous model (2), path properties were collected in the RESERVE message.

Some parameters in QoS Available may the same as in QoS Desired. For these parameters the implicit message is that the sender would be satisfied by a reservation with lower parameter values than specified in QoS Desired.

It is possible for QoS Available to contain parameters that do not appear in QoS Desired. It is assumed that the value of these parameters is collected for informational purposes only (e.g. path latency).

Parameter values in QoS Available are seeded according to the senders capabilities. Each QNE downgrades or cumulates the parameter values according to its current capabilities.

The receiver (QNI) populates QoS Desired as follows: For those parameters that appear in both QoS Available and QoS Desired in the QUERY message, it takes the (possibly downgraded) parameter values from QoS Available. For those parameters that only appear in QoS Desired, it adopts the parameter values from QoS Desired.

The parameters in the QoS Available QSPEC object in the RESERVE message are copied with their values from the QoS Available QSPEC object in the QUERY message. Note that the parameters in QoS Available are read-write in the QUERY message, whereas they are read-only in the RESERVE message.

The advantage of this model compared to the sender-initiated reservation (model 2) is that the situation of over-reservation in QNEs close to the QNI as described above does not occur. On the other hand, the QUERY may find, for example, a particular bandwidth is not available. When the actual reservation is performed, however, the desired bandwidth may meanwhile have become free. That is, the ‘RSVP style’ may result in a smaller reservation than necessary.

Regarding Control Information in receiver-initiated reservations, the sender includes all Control Information it cares about in the QUERY message. Read-write parameters are updated by QNEs as the QUERY message travels towards the receiver. The receiver includes all Control Information parameters arriving in the QUERY message also in the RESERVE message, as read-only parameters with the value they had when arriving at the receiver.

6.1.3 Resource Queries

Here the QNI issues a QUERY in order to investigate what resources are currently available. The QNR replies with a RESPONSE.
<table>
<thead>
<tr>
<th>ID</th>
<th>QUERY</th>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>QoS Available</td>
<td>QoS Available</td>
</tr>
</tbody>
</table>

Note QoS Available when traveling in the QUERY is read-write, whereas in the RESPONSE it is read-only.

6.1.4 Bidirectional Reservations

On a QSPEC level, bidirectional reservations are no different from uni-directional reservations, since QSPECs for different directions never travel in the same message.

6.2 QSPEC Examples

This Section provides an example QSPEC for DiffServ admission control. The QSPEC for IntServ controlled load service is specified in [INTSERV-QOSM] (note that the QOSMs for IntServ Controlled Load Service and IntServ Guaranteed Service are defined in [RFC2211] and [RFC2212], respectively).

The QSPEC for DiffServ admission control may be composed, for example, of the QSPEC objects <QoS Desired> and <QoS Available>, as well as <QoS Reserved>. Which QSPEC object is present in a particular QSPEC depends on the message type (RESERVE, QUERY etc) in which the QSPEC travels. Parameters in the QSPEC for DiffServ requesting bandwidth for different PHBs are as follows:

Example QSPEC for the DiffServ EF PHB [RFC3297]:

- `<QoS Desired>` = `<Traffic Description>` `<QoS Class>`
- `<Traffic Description>` = `<Token Bucket>`
- `<QoS class>` = `<PHB Class=EF>`
- `<QoS Available>` = `<Token Bucket>`
- `<QoS Reserved>` = `<Token Bucket>`

In general, the EF PHB is a property of the service that is NOT dependent on the input traffic characteristics. A server of rate R and latency E that is compliant with the EF PHB must deliver at least the configured service rate R with at most latency E for any traffic characterization. Therefore, strictly speaking, there is no specific traffic descriptor required to deliver the EF PHB (which by definition is a local per-hop characterization). However, in order to deliver a reasonable end-to-end delay, it is typically assumed that EF traffic is shaped at the ingress. A typical assumption is that input traffic at any ingress is constrained by a single rate token bucket. Therefore, a single rate token bucket is sufficient to signal in QoS-NSLP/QSPEC for the DiffServ-QOSM.
Example QSPEC for the DiffServ AFxy PHB [RFC2597]:

<QSPEC Control Information> = <Excess Treatment>
<QoS Description> = <QoS Desired> <QoS Available> <QoS Reserved>
<QoS Desired> = <Traffic Description> <QoS Class>
<Traffic Description> = <Committed Burst Size (CBS) Token Bucket>
                             <Excess Burst Size (EBS) Token Bucket>
<QoS class> = <PHB Class=AF1x>
<QoS Available> = <CBS Token Bucket> <EBS Token Bucket>
<QoS Reserved> = <CBS Token Bucket> <EBS Token Bucket>

The AF1 PHB class is signaled consisting of the three AF1x PHBs. See [RFC3140] and [RFC2597] for construction of the PHBID for the AF1 PHB class as the concatenation: AF11 recommended DSCP | 8 x 0 bits | 10, i.e., 001010 00000000 10 = 0x2802.

QNEs process two sets of token bucket parameters to implement the DiffServ AF QOSM, one token bucket for the average (CBS) traffic and one token bucket for the burst (EBS) traffic. These 2 token buckets are sufficient to cover most of the ways in which one would distinguish among 3 levels of drop precedence at the queuing mechanics level, as described in the Appendix to [RFC2597].

QoS-NSLP/QSPEC can support signaling the parameters required for the DiffServ marker elements described in [RFC2697] and [RFC2698]. [RFC2697] defines a Single Rate Three Color Marker (srTCM), which can be used as component in a DiffServ traffic conditioner [RFC2475, RFC2474]. The srTCM meters a traffic stream and marks its packets according to three traffic parameters, Committed Information Rate (CIR), Committed Burst Size (CBS), and Excess Burst Size (EBS), to be either green, yellow, or red. A packet is marked green if it does not exceed the CBS, yellow if it does exceed the CBS but not the EBS, and red otherwise.

RFC 2697 and RFC 2698 provide specific procedures, where in essence, RFC 2697 is using two token buckets that run at the same rate.

The <Token Bucket> parameter (see Section 7.2.5) includes values for Token Bucket Rate \([r]\), Token Bucket Size \([b]\), Peak Data Rate \([p]\), Minimum Policed Unit \([m]\), and Maximum Packet Size \([MTU]\). Most DiffServ discussions of token buckets consider only Token Bucket Rate and Token Bucket Size. To realize this sort of basic token bucket, the peak rate value \([p]\) is set to positive infinity, the Minimum Policed Unit \([m]\) to zero, and the Maximum Packet Size \([MTU]\) to a very large number (e.g., the maximum positive 32-bit integer). Most DiffServ implementations can be expected to ignore these three values. Note that [RFC2215] adds \(p, m, \) and \(MTU\) to get a TOKEN_BUCKET_TSPEC, however DiffServ does not use these three added values.

The srTCM [RFC 2697] may be signaled by using the same Committed
Information Rate as the rate \([r]\) for both Token Buckets (#1 and #2) and carrying the Committed Burst Size as the size of Token Bucket #1 and the Excess Burst Size as the size of Token Bucket #2. The trTCM [RFC2698] can be realized by carrying the Committed Information Rate and Committed Burst Size in Token Bucket #1 and the Peak Information Rate and Peak Burst Size in Token Bucket #2. Note that this approach does not capture color-blind versus color-aware configurations of a trTCM. However, the QSPEC carries the traffic description, for which two token buckets are enough, and detailed DiffServ configuration to deal with this is handled via other means.

7. QSPEC Functional Specification

This Section defines the encodings of the QSPEC parameters and QSPEC control information defined in Section 5. We first give the general QSPEC formats and then the formats of the QSPEC objects and parameters.

Note that all QoS Description parameters can be either read-write or read-only, depending on which object and which message they appear in. However, in a given QSPEC object, all objects are either read-write or read-only. In order to simplify keeping track of whether an object is read-write or read-only, a corresponding flag is associated with each object.

Network byte order (‘big-endian’) for all 16- and 32-bit integers, as well as 32-bit floating point numbers, are as specified in [RFC1832, IEEE754, NETWORK-BYTE-ORDER].

7.1 General QSPEC Formats

The format of the QSPEC closely follows that used in GIST [GIST] and QoS NSLP [QoS-SIG]. Every object (and parameter) has the following general format:

- The overall format is Type-Length-Value (in that order).
- Some parts of the type field are set aside for control flags.
- Length has the units of 32-bit words, and measures the length of Value. If there is no Value, Length=0. The Object length excludes the header.
- Value is a whole number of 32-bit words. If there is any padding required, the length and location MUST be defined by the object-specific format information; objects that contain variable length types may need to include additional length subfields to do so.
- Any part of the object used for padding or defined as reserved("r") MUST be set to 0 on transmission and MUST be ignored on reception.
Empty QSPECs and empty QSPEC Objects MUST NOT be used.

Duplicate objects, duplicate parameters, and/or multiple occurrences of a parameter MUST NOT be used.

The Common QSPEC Header is a fixed 4-byte long object containing the QOSM ID and an identifier for the QSPEC Procedure (see Section 6.1):

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Vers. | QOSM ID | QSPEC Proc. | Reserved |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Note that a length field is not necessary since the overall length of the QSPEC is contained in the higher level QoS NSLP data object.

Vers.: Identifies the QSPEC version number. It is assigned by IANA.

QOSM ID: Identifies the particular QOSM being used by the QNI. It is assigned by IANA.

QSPEC Proc.: Is composed of two times 4 bits. The first set of bits identifies the Message Sequence, the second set identifies the QSPEC Object Combination used for this particular message sequence:

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-
|Mes.Sq|Obj.Cmb|
+-+-+-+-+-+-+-+-+-
```

The Message Sequence field can attain the following values:

0: Sender-Initiated Reservations, as defined in Section 6.1.1
1: Receiver-Initiated Reservations, as defined in Section 6.1.2
2: Resource Queries, as defined in Section 6.1.3
The Object Combination field can take the values between 1 and 3 indicated in the tables in Section 6.1.1 to 6.1.3.

The QSPEC Control Information is a variable length object containing one or more parameters. The QSPEC Objects field is a collection of QSPEC objects (QoS Desired, QoS Available, etc.), which share a common format and each contain several parameters.

Both the QSPEC Control Information object and the QSPEC QoS objects share a common header format:

```
+---------------------------------------------------------------+
| R | E | r | r | r | r | Object Type | r | r | r | r | Length |
+---------------------------------------------------------------+
```

R Flag: If set the parameters contained in the object are read-only. Otherwise they are read-write. Note that in the case of Object Type = 0 (Control Information), this value is overwritten by parameter-specific values.

E Flag: Set if an error occurs on object level

Object Type = 0: control information
   = 1: QoS Desired
   = 2: QoS Available
   = 3: QoS Reserved
   = 4: Minimum QoS

The r-flags are reserved.

Each optional or mandatory parameter within an object can be similarly encoded in TLV format using a similar parameter header:

```
+---------------------------------------------------------------+
| M | E | N | T | Parameter ID | r | r | r | r | Length |
+---------------------------------------------------------------+
```

M Flag: When set indicates the subsequent parameter is a mandatory parameter and MUST be interpreted. Otherwise the parameter is optional and can be ignored if not understood.

E Flag: When set indicates an error occurred when this parameter was being interpreted.

N Flag: Not-supported Flag (see Section 4.5). For mandatory parameters the value of this flag is always zero.

T Flag: Tunneled-parameter Flag (see Section 4.5)
Parameter Type: Assigned to each parameter (see below)

7.2 Parameter Coding

Parameters are usually coded individually, for example, the Bandwidth Parameter (Section 7.2.3). However, it is also possible to combine several parameters into one parameter field, which is called "container coding". This coding is useful if either a) the parameters always occur together, as for example the several parameters that jointly make up the token bucket, or b) in order to make coding more efficient because the length of each parameter value is much less than a 32-bit word (as for example described in [RMD-QOSM]).

7.2.1 <NON QOSM Hop> Parameter

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| +---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+---------------+----------------
that none is specified is that there are no guarantees to excess traffic, i.e. a QNE can do whatever it finds suitable.

If ’no metering or policing is permitted’ is signaled, the QNE should accept the <Excess Treatment> parameter set by the sender with special care so that excess traffic should not cause a problem. To request the Null Meter [RFC3290] is especially strong, and should be used with caution.

A NULL metering application [RFC2997] would not include the traffic profile, and conceptually it should be possible to support this with the QSPEC. A QSPEC without a traffic profile is not excluded by the current specification. However, note that the traffic profile is important even in those cases when the excess treatment is not specified, e.g., in negotiating bandwidth for the best effort aggregate. However, a "NULL Service QOSM" would need to be specified where the desired QNE Behavior and the corresponding QSPEC format are described.

As an example behavior for a NULL metering, in the properly configured DiffServ router, the resources are shared between the aggregates by the scheduling disciplines. Thus, if the incoming rate increases, it will influence the state of a queue within that aggregate, while all the other aggregates will be provided sufficient bandwidth resources. NULL metering is useful for best effort and signaling data, where there is no need to meter and police this data as it will be policed implicitly by the allocated bandwidth and, possibly, active queue management mechanism.

7.2.3 <Bandwidth> [RFC 2212, RFC 2215]

```
+-----------------+-+---------------------------+-----------------+-+---------------------------+
|  Bandwidth      |   | (32-bit IEEE floating point number)   |
+-----------------+-+---------------------------+-----------------+-+---------------------------+
```

The <Bandwidth> parameter MUST be nonnegative and is measured in bytes per second and has the same range and suggested representation as the bucket and peak rates of the <Token Bucket>. <Bandwidth> can be represented using single-precision IEEE floating point. The representation MUST be able to express values ranging from 1 byte per second to 40 terabytes per second. For values of this parameter only valid non-negative floating point numbers are allowed. Negative numbers (including "negative zero"), infinities, and NAN’s are not allowed.

A QNE MAY export a local value of zero for this parameter. A network element or application receiving a composed value of zero for this
7.2.4 <Slack Term> Parameter [RFC 2212, RFC 2215]

Slack term S MUST be nonnegative and is measured in microseconds.
The Slack term, S, can be represented as a 32-bit integer. Its value
can range from 0 to (2**32)-1 microseconds.

7.2.5 <Token Bucket> Parameters [RFC 2215]

The <Token Bucket> parameters are represented by three floating
point numbers in single-precision IEEE floating point format followed
by two 32-bit integers in network byte order. The first floating
point value is the rate (r), the second floating point value is the
bucket size (b), the third floating point is the peak rate (p), the
first unsigned integer is the minimum policed unit (m), and the
second unsigned integer is the maximum datagram size (MTU).

Note that the two sets of <Token Bucket> parameters can be
distinguished, as could be needed for example to support DiffServ
applications (see Section 7.2).

Token Bucket #1 Parameter ID = 4
Token Bucket #1: Mandatory QSPEC Parameter

Parameter Values:
Token Bucket #2 Parameter ID = 5
Token Bucket #2: Optional QSPEC Parameter

Parameter Values:

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|0 |1 |2 |3 |

Token Bucket Rate \[ r \] (32-bit IEEE floating point number)
Token Bucket Size \[ b \] (32-bit IEEE floating point number)
Peak Data Rate \[ p \] (32-bit IEEE floating point number)
Minimum Policed Unit \[ m \] (32-bit unsigned integer)
Maximum Packet Size [MTU] (32-bit unsigned integer)

When \( r \), \( b \), and \( p \) terms are represented as IEEE floating point values, the sign bit MUST be zero (all values MUST be non-negative). Exponents less than 127 (i.e., 0) are prohibited. Exponents greater than 162 (i.e., positive 35) are discouraged, except for specifying a peak rate of infinity. Infinity is represented with an exponent of all ones (255) and a sign bit and mantissa of all zeroes.

7.2.6 <QoS Class> Parameters

7.2.6.1 <PHB Class> Parameter [RFC 3140]

As prescribed in RFC 3140, the encoding for a single PHB is the recommended DSCP value for that PHB, left-justified in the 16 bit field, with bits 6 through 15 set to zero.

The registries needed to use RFC 3140 already exist, see [DSCP-REGISTRY, PHBID-CODES-REGISTRY]. Hence, no new registry needs to be created for this purpose.
7.2.6.2 <Y.1541 QoS Class> Parameter [Y.1541]

<table>
<thead>
<tr>
<th>Class</th>
<th>Mean delay</th>
<th>Delay variation</th>
<th>Loss ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&lt;= 100 ms</td>
<td>&lt;= 50 ms</td>
<td>&lt;= 10^-3</td>
</tr>
<tr>
<td>1</td>
<td>&lt;= 400 ms</td>
<td>&lt;= 50 ms</td>
<td>&lt;= 10^-3</td>
</tr>
<tr>
<td>2</td>
<td>&lt;= 100 ms</td>
<td>unspecified</td>
<td>&lt;= 10^-3</td>
</tr>
<tr>
<td>3</td>
<td>&lt;= 400 ms</td>
<td>unspecified</td>
<td>&lt;= 10^-3</td>
</tr>
<tr>
<td>4</td>
<td>&lt;= 1 sec</td>
<td>unspecified</td>
<td>&lt;= 10^-3</td>
</tr>
<tr>
<td>5</td>
<td>unspecified</td>
<td>unspecified</td>
<td>unspecified</td>
</tr>
<tr>
<td>6</td>
<td>&lt;= 100 ms</td>
<td>&lt;= 50 ms</td>
<td>&lt;= 10^-5</td>
</tr>
<tr>
<td>7</td>
<td>&lt;= 400 ms</td>
<td>&lt;= 50 ms</td>
<td>&lt;= 10^-5</td>
</tr>
</tbody>
</table>

Class 0: Mean delay <= 100 ms, delay variation <= 50 ms, loss ratio <= 10^-3. Real-time, highly interactive applications, sensitive to jitter. Application examples include VoIP, Video Teleconference.

Class 1: Mean delay <= 400 ms, delay variation <= 50 ms, loss ratio <= 10^-3. Real-time, interactive applications, sensitive to jitter. Application examples include VoIP, Video Teleconference.

Class 2: Mean delay <= 100 ms, delay variation unspecified, loss ratio <= 10^-3. Highly interactive transaction data. Application examples include signaling.

Class 3: Mean delay <= 400 ms, delay variation unspecified, loss ratio <= 10^-3. Interactive transaction data. Application examples include signaling.

Class 4: Mean delay <= 1 sec, delay variation unspecified, loss ratio <= 10^-3. Low Loss Only applications. Application examples include short transactions, bulk data, video streaming.

Class 5: Mean delay unspecified, delay variation unspecified, loss ratio unspecified. Unspecified applications. Application examples include traditional applications of default IP networks.

Class 6: Mean delay <= 100 ms, delay variation <= 50 ms, loss ratio <= 10^-5. Applications that are highly sensitive to loss, such as television transport, high-capacity TCP transfers, and TDM circuit emulation.

Class 7: Mean delay <= 400 ms, delay variation <= 50 ms, loss ratio <= 10^-5. Applications that are highly sensitive to loss, such as television transport, high-capacity TCP transfers, and TDM circuit emulation.
transport, high-capacity TCP transfers, and TDM circuit emulation.

7.6.2.3 <DSTE Class Type> Parameter [RFC3564]

DSTE class type is defined as follows:

```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|1|E|0|T| 8 |r|r|r|r| 1 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

DSTE Class Type: Indicates the DSTE class type. Values currently allowed are 0, 1, 2, 3, 4, 5, 6, 7.

7.2.7 Priority Parameters

7.2.7.1 <Preemption Priority> & <Defending Priority> Parameters [RFC 3181]

```
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|1|E|0|T| 9 |r|r|r|r| 1 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Preemption Priority: The priority of the new flow compared with the defending priority of previously admitted flows. Higher values represent higher priority.

Defending Priority: Once a flow is admitted, the preemption priority becomes irrelevant. Instead, its defending priority is used to compare with the preemption priority of new flows.

As specified in [RFC3181], <Preemption Priority> and <Defending Priority> are 16-bit integer values and both MUST be populated if the parameter is used.
7.2.7.2 <Admission Priority> Parameter [PRIORITY-RQMTS]

| Admission Priority | Parameter 
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>high priority flow</td>
</tr>
<tr>
<td>1</td>
<td>normal priority flow</td>
</tr>
<tr>
<td>2</td>
<td>best-effort priority flow</td>
</tr>
</tbody>
</table>

High priority flows, normal priority flows, and best-effort priority flows can have access to resources depending on their admission priority value, as described in [PRIORITY-RQMTS], as follows:

 Admission Priority:

0 - high priority flow
1 - normal priority flow
2 - best-effort priority flow

7.2.7.3 <RPH Priority> Parameter [SIP-PRIORITY]

<table>
<thead>
<tr>
<th>RPH Namespace</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>dsn</td>
<td>0</td>
</tr>
<tr>
<td>drsn</td>
<td>1</td>
</tr>
<tr>
<td>q735</td>
<td>2</td>
</tr>
<tr>
<td>ets</td>
<td>3</td>
</tr>
<tr>
<td>wps</td>
<td>4</td>
</tr>
<tr>
<td>not populated</td>
<td>5</td>
</tr>
</tbody>
</table>

[SIP-PRIORITY] defines a resource priority header (RPH) with parameters "RPH Namespace" and "RPH Priority" combination, and if populated is applicable only to flows with high reservation priority, as follows:

 RPH Namespace:

0 - dsn
1 - drsn
2 - q735
3 - ets
4 - wps
5 - not populated

 RPH Priority:

Each namespace has a finite list of relative priority-values. Each is listed here in the order of lowest priority to highest priority:

4 - dsn.routine
3 - dsn.priority
2 - dsn.immediate
1 - dsn.flash
0 - dsn.flash-override

5 - drsn.routine
4 - drsn.priority
3 - drsn.immediate
2 - drsn.flash
1 - drsn.flash-override
0 - drsn.flash-override-override

4 - q735.4
3 - q735.3
2 - q735.2
1 - q735.1
0 - q735.0

4 - ets.4
3 - ets.3
2 - ets.2
1 - ets.1
0 - ets.0

4 - wps.4
3 - wps.3
2 - wps.2
1 - wps.1
0 - wps.0

Note that additional work is needed to communicate these flow priority values to bearer-level network elements [VERTICAL-INTERFACE].

7.2.8 <Path Latency> Parameter [RFC 2210, 2215]

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|E|N|T|           12          |r|r|r|r|          1            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Path Latency (32-bit integer)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Path Latency is a single 32-bit integer in network byte order. The composition rule for the <Path Latency> parameter is summation with a clamp of \((2^{32} - 1)\) on the maximum value. The latencies are average values reported in units of one microsecond. A system with resolution less than one microsecond MUST set unused digits to zero. An individual QNE can advertise a latency value between 1 and \(2^{28}\) (somewhat over two minutes) and the total latency added across all QNEs can range as high as \((2^{32})-2\). If the sum of the different
elements delays exceeds \((2^{*32})-2\), the end-to-end advertised delay
SHOULD be reported as indeterminate. A QNE that cannot accurately
predict the latency of packets it is processing MUST raise the
not-supported flag and either leave the value of Path Latency as is,
or add its best estimate of its lower bound. A raised not-supported
flag indicates the value of Path Latency is a lower bound of the
real Path Latency. The distinguished value \((2^{*32})-1\) is taken to
mean indeterminate latency because the composition function limits
the composed sum to this value, it indicates the range of the
composition calculation was exceeded.

### 7.2.9 <Path Jitter> Parameter

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>N</td>
<td>T</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Path Jitter STAT1(variance) (32-bit integer)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path Jitter STAT2(99.9%-ile) (32-bit integer)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Path Jitter STAT3(reserved) (32-bit integer)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Path Jitter is a set of three 32-bit integers in network byte
order. The Path Jitter parameter is the combination of three
statistics describing the Jitter distribution with a clamp of
\((2^{*32} - 1)\) on the maximum of each value. The jitter STATs are
reported in units of one microsecond. A system with resolution less
than one microsecond MUST set unused digits to zero. An individual
QNE can advertise jitter values between 1 and \(2^{*28}\) (somewhat over
two minutes) and the total jitter computed across all QNEs can range
as high as \((2^{*32})-2\). If the combination of the different element
values exceeds \((2^{*32})-2\), the end-to-end advertised jitter SHOULD be
reported as indeterminate. A QNE that cannot accurately predict the
jitter of packets it is processing MUST raise the not-supported flag
and either leave the value of Path Jitter as is, or add its best
estimate of its STAT values. A raised not-supported flag indicates
the value of Path Jitter is a lower bound of the real Path Jitter.
The distinguished value \((2^{*32})-1\) is taken to mean indeterminate
jitter. A QNE that cannot accurately predict the jitter of packets
it is processing SHOULD set its local parameter to this value.
Because the composition function limits the total to this value,
receipt of this value at a network element or application indicates
that the true path jitter is not known. This MAY happen because one
or more network elements could not supply a value, or because the
range of the composition calculation was exceeded.

NOTE: The Jitter composition function makes use of the <Path Latency>
parameter. Composition functions for loss, latency and jitter may be
found in [Y.1541]. Additional study is in-progress.

7.2.10 <Path PLR> Parameter

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Path PLR is a single 32-bit single precision IEEE floating point number in network byte order. The composition rule for the <Path PLR> parameter is summation with a clamp of 10^-1 on the maximum value. The PLRs are reported in units of 10^-11. A system with resolution less than one microsecond MUST set unused digits to zero. An individual QNE can advertise a PLR value between zero and 10^-2 and the total PLR added across all QNEs can range as high as 10^-1. If the sum of the different elements values exceeds 10^-1, the end-to-end advertised PLR SHOULD be reported as indeterminate. A QNE that cannot accurately predict the PLR of packets it is processing MUST raise the not-supported flag and either leave the value of Path PLR as is, or add its best estimate of its lower bound. A raised not-supported flag indicates the value of Path PLR is a lower bound of the real Path PLR. The distinguished value 10^-1 is taken to mean indeterminate PLR. A QNE which cannot accurately predict the PLR of packets it is processing SHOULD set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path PLR is not known. This MAY happen because one or more network elements could not supply a value, or because the range of the composition calculation was exceeded.

7.2.11 <Path PER> Parameter

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Path PER is a single 32-bit single precision IEEE floating point number in network byte order. The composition rule for the <Path PER> parameter is summation with a clamp of 10^-1 on the maximum value. The PERs are reported in units of 10^-11. A system with resolution less than one microsecond MUST set unused digits to zero. An individual QNE can advertise a PER value between zero and 10^-2 and the total PER added across all QNEs can range as high as 10^-1.
If the sum of the different elements values exceeds $10^{-1}$, the end-to-end advertised PER SHOULD be reported as indeterminate. A QNE that cannot accurately predict the PER of packets it is processing MUST raise the not-supported flag and either leave the value of Path PER as is, or add its best estimate of its lower bound. A raised not-supported flag indicates the value of Path PER is a lower bound of the real Path PER. The distinguished value $10^{-1}$ is taken to mean indeterminate PER. A QNE which cannot accurately predict the PER of packets it is processing SHOULD set its local parameter to this value. Because the composition function limits the composed sum to this value, receipt of this value at a network element or application indicates that the true path PER is not known. This MAY happen because one or more network elements could not supply a value, or because the range of the composition calculation was exceeded.

7.2.12 <Ctot> <Dtot> <Csum> <Dsum> Parameters [RFC 2210, 2212, 2215]

The error term C is measured in units of bytes. An individual QNE can advertise a C value between 1 and $2^{28}$ (a little over 250
megabytes) and the total added over all QNEs can range as high as (2**32)-1. Should the sum of the different QNEs delay exceed (2**32)-1, the end-to-end delay term MUST be set to (2**32)-1. The error term D is measured in units of one microsecond. An individual QNE can advertise a delay value between 1 and 2**28 (somewhat over two minutes) and the total delay added over all QNEs can range as high as (2**32)-1. Should the sum of the different QNEs delay exceed (2**32)-1, the end-to-end delay MUST be set to (2**32)-1.

8. Security Considerations

The priority parameter raises possibilities for Theft of Service Attacks because users could claim an emergency priority for their flows without real need, thereby effectively preventing serious emergency calls to get through. Several options exist for countering such attacks, for example

- only some user groups (e.g. the police) are authorized to set the emergency priority bit
- any user is authorized to employ the emergency priority bit for particular destination addresses (e.g. police)

9. IANA Considerations

This section defines the registries and initial codepoint assignments for the QSPEC template, in accordance with BCP 26 RFC 2434 [RFC2434]. It also defines the procedural requirements to be followed by IANA in allocating new codepoints. Guidelines on the technical criteria to be followed in evaluating requests for new codepoint assignments are given for the overall NSIS protocol suite in a separate NSIS extensibility document [NSIS-EXTENSIBILITY].

This specification allocates the following codepoints in existing registries:

PHB Class Parameter [RFC 3140] (Section 7.2.6.1)

The registries needed to use RFC 3140 already exist [DSCP-REGISTRY, PHBID-CODES-REGISTRY].

This specification creates the following registries with the structures as defined below:

Object Types (12 bits):
The following values are allocated by this specification:
0-4: assigned as specified in Section 7.
The allocation policies for further values are as follows:
5-63: Standards Action
64-127: Private/Experimental Use
128-4095: Reserved
Guidelines on the technical criteria to be followed in evaluating requests for new codepoint assignments are given for the overall NSIS protocol suite in a separate NSIS extensibility document [NSIS-EXTENSIBILITY].

**QSPEC Version (4 bits):**
The following value is allocated by this specification:
0: assigned to Version 0 QSPEC
The allocation policies for further values are as follows:
1-15: Standards Action

**QOSM ID (12 bits):**
The following values are allocated by this specification:
0: IntServ Controlled Load Service QOSM [INTSERV-QOSM]
1: RMD QOSM [RMD-QOSM]
2: Y.1541 QOSM [Y.1541-QOSM]
The allocation policies for further values are as follows:
3-63: Specification Required
64-127: Private/Experimental Use
128-4095: Reserved

**QSPEC Procedure (8 bits):**
Broken down into
**Message Sequence (4 bits):**
The following values are allocated by this specification:
0-2: assigned as specified in Section 7.1
The allocation policies for further values are as follows:
3-15: Standards Action

**Object Combination:**
The following values are allocated by this specification:
0-2: assigned as specified in tables in Section 6.1.1 --&gt; 6.1.3
The allocation policies for further values are as follows:
3-15: Standards Action

**Parameter ID (12 bits):**
The following values are allocated by this specification:
0-18: assigned as specified in Sections 7.2.1 --&gt; 7.2.12.
The allocation policies for further values are as follows:
19-63: Standards Action (for mandatory parameters)
64-127: Specification Required (for optional parameters)
128-255: Private/Experimental Use
255-4095: Reserved

**Excess Treatment Parameter (8 bits):**
The following values are allocated by this specification:
0-3: assigned as specified in Section 7.2.2
The allocation policies for further values are as follows:
4-63: Standards Action
64-255: Reserved
Y.1541 QoS Class Parameter (12 bits):
The following values are allocated by this specification:
0-7: assigned as specified in Section 7.2.6.2
The allocation policies for further values are as follows:
8-63: Standards Action
64-4095: Reserved

DSTE Class Type Parameter (12 bits):
The following values are allocated by this specification:
0-7: assigned as specified in Section 7.2.6.3
The allocation policies for further values are as follows:
8-63: Standards Action
64-4095: Reserved

Admission Priority Parameter (8 bits):
The following values are allocated by this specification:
0-2: assigned as specified in Section 7.2.6.2
The allocation policies for further values are as follows:
3-63: Standards Action
64-255: Reserved

RPH Namespace Parameter (16 bits):
The following values are allocated by this specification:
0-5: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
6-63: Standards Action
64-65535: Reserved

RPH Priority Parameter (8 bits):
dsn namespace:
The following values are allocated by this specification:
0-4: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
5-63: Standards Action
64-255: Reserved
drsn namespace:
The following values are allocated by this specification:
0-5: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
6-63: Standards Action
64-255: Reserved
Q735 namespace:
The following values are allocated by this specification:
0-4: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
5-63: Standards Action
64-255: Reserved
ets namespace:
The following values are allocated by this specification:
0-4: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
5-63: Standards Action
64-255: Reserved

wts namespace:
The following values are allocated by this specification:
0-4: assigned as specified in Section 7.2.7.2
The allocation policies for further values are as follows:
5-63: Standards Action
64-255: Reserved

10. Acknowledgements

The authors would like to thank (in alphabetical order) David Black,
Anna Charny, Matthias Friedrich, Xiaoming Fu, Robert Hancock, Chris
Lang, Jukka Manner, Dave Oran, Tom Phelan, Alexander Sayenko, Bernd
Schloer, Hannes Tschofenig, and Sven van den Bosch for their very
helpful suggestions.

11. Normative References

[GIST] Schulzrinne, H., Hancock, R., "GIST: General Internet
Signaling Transport," work in progress.
[NSIS-EXTENSIBILITY] Loughney, J., "NSIS Extensibility Model", work
in progress.
Signaling," work in progress.
[RFC1832] Srinivasan, R., "XDR: External Data Representation
[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
[RFC2210] Wroclawski, J., "The Use of RSVP with IETF Integrated
[RFC2215] Shenker, S., Wroclawski, J., "General Characterization
Parameters for Integrated Service Network Elements", RFC 2215, Sept.
1997.
[RFC2474] Nichols, K., et. al., "Definition of the Differentiated
Services Field (DS Field) in the IPv4 and IPv6 Headers," RFC 2474,
December 1998.
2597, June 1999.
[RFC2697] Heinanen, J., Guerin, R., "A Single Rate Three Color
12. Informative References


13. Authors' & Contributors' Addresses

Jerry Ash (Editor)
AT&T
Room MT D5-2A01
200 Laurel Avenue
Middletown, NJ 07748, USA
Phone: +1-(732)-420-4578
Fax: +1-(732)-368-8659
Email: gash@att.com

Attila Bader (Editor)
Traffic Lab
Ericsson Research
Ericsson Hungary Ltd.
Laborc u. 1 H-1037
Budapest Hungary
Email: Attila.Bader@ericsson.com

Cornelia Kappler (Editor)
Siemens AG
Siemensdamm 62
Berlin 13627
Germany
Email: cornelia.kappler@siemens.com

Chuck Dvorak
AT&T
Room 2A37
180 Park Avenue, Building 2
Florham Park, NJ 07932
Phone: +1 973-236-6700
Fax: +1 973-236-7453
Email: cdvorak@att.com

Yacine El Mghazli
Alcatel
Route de Nozay
91460 Marcoussis cedex
FRANCE
Phone: +33 1 69 63 41 87
Email: yacine.el_mghazli@alcatel.fr

Georgios Karagiannis
University of Twente
P.O. BOX 217
Appendix A: QoS Models and QSPECs

This Appendix gives a description of QoS Models and QSPECs and explains what is the relation between them. Once these descriptions are contained in a stable form in the appropriate IDs this Appendix will be removed.

QoS NSLP is a generic QoS signaling protocol that can signal for many QOSMs. A QOSM is a particular QoS provisioning method or QoS architecture such as IntServ Controlled Load or Guaranteed Service, DiffServ, or RMD for DiffServ.

The definition of the QOSM is independent from the definition of QoS NSLP. Existing QOSMs do not specify how to use QoS NSLP to signal for them. Therefore, we need to define the QOSM specific signaling functions, as [RMD-QOSM], [INTSERV-QOSM], and [Y.1541-QOSM].
A QOSM MUST include the following information:

- Role of QNEs in this QOSM: E.g., location, frequency, statefulness, etc.
- QSPEC Definition: A QOSM MUST specify the QSPEC, including a value for the QOSM ID, and which QSPEC parameters must be included. Furthermore it needs to explain how QSPEC parameters not used in this QOSM are mapped onto parameters defined therein.
- QSPEC procedures: A QOSM MUST describe which QSPEC procedures are applicable to this QOSM.
- Processing rules in QNEs: It describes how QSPEC info is treated and interpreted in the RMF and QOSM specific processing. E.g., admission control, scheduling, policy control, QoS parameter accumulation (e.g., delay).
- QSPEC example: It includes at least one bit-level QSPEC example.

Appendix B: Mapping of QoS Desired, QoS Available and QoS Reserved of NSIS onto AdSpec, TSpec and RSpec of RSVP IntServ

The union of QoS Desired, QoS Available and QoS Reserved can provide all functionality of the objects specified in RSVP IntServ, however it is difficult to provide an exact mapping.

In RSVP, the Sender TSpec specifies the traffic an application is going to send (e.g. token bucket). The AdSpec can collect path characteristics (e.g. delay). Both are issued by the sender. The receiver sends the FlowSpec which includes a Receiver TSpec describing the resources reserved using the same parameters as the Sender TSpec, as well as a RSpec which provides additional IntServ QoS Model specific parameters, e.g. Rate and Slack.

The RSVP TSpec/AdSpec/RSpec seem quite tailored to receiver-initiated signaling employed by RSVP, and the IntServ QoS Model. E.g. to the knowledge of the authors it is not possible for the sender to specify a desired maximum delay except implicitly and mutably by seeding the AdSpec accordingly. Likewise, the RSpec is only meaningfully sent in the receiver-issued RSVP RESERVE message. For this reason our discussion at this point leads us to a slightly different mapping of necessary functionality to objects, which should result in more flexible signaling models.
Appendix C: Main Changes Since Last Version & Open Issues

C.1 Main Changes Since Version -04

Version -05:
- fixed <QOSM hops> in Sec. 5 and 6.2 as discussed at Interim Meeting
- discarded QSPEC parameter <M> (Maximum packet size) since MTU discovery is expected to be handled by procedure currently defined by PMTUD WG
- added "container QSPEC parameter" in Sec. 6.1 to augment encoding efficiency
- added the ‘tunneled QSPEC parameter flag’ to Sections 5 and 6
- revised Section 6.2.2 on SIP priorities
- added QSPEC procedures for "RSVP-style reservation", resource queries and bidirectional reservations in Sec. 7.1
- reworked Section 7.2

Version -06:
- defined "not-supported flag" and "tunneled parameter flag" (subsumes "optional parameter flag")
- defined "error flag" for error handling
- updated bit error rate (BER) parameter to packet loss ratio (PLR) parameter
- added packet error ratio (PER) parameter
- coding checked by independent expert
- coding updated to include RE flags in QSPEC objects and MENT flags in QSPEC parameters

Version -07:
- added text (from David Black) on DiffServ QSPEC example in Section 6
- re-numbered QSPEC parameter IDs to start with 0 (Section 7)
- expanded IANA Considerations Section 9

Version -08:
- update to ‘RSVP-style’ reservation in Section 6.1.2 to mirror what is done in RSVP
- modified text (from David Black) on DiffServ QSPEC example in Section 6.2
- update to general QSPEC parameter formats in Section 7.1 (length restrictions, etc.)
- re-numbered QSPEC parameter IDs in Section 7.2
- modified <Excess Treatment> parameter values in Section 7.2.2
- update to reservation priority Section 7.2.7
- specify the 3 "STATS" in the <Path Jitter> parameter, Section 7.2.9.4
- minor updates to IANA Considerations Section 9
C.2 Open Issues

None.

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM 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.

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Ash, et. al.       <draft-ietf-nsis-qspec-08.txt>