Independent Submission Q. Hu
Internet-Draft B. Carpenter
Intended status: Informational Univ. of Auckland
Expires: April 3, 2011 September 30, 2010

Survey of proposed use cases for the IPv6 flow label
draft-hu-flow-label-cases-02

Abstract

The IPv6 protocol includes a flow label in every packet header, but this field is not used in practice. This paper describes the flow label standard and discusses the implementation issues that it raises. It then describes various published proposals for using the flow label, and shows that most of them are inconsistent with the standard. Methods to address this problem are briefly reviewed. We also question whether the standard should be revised.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

IPv6 is being introduced to overcome the address shortage of the current IPv4 protocol, but it also offers a new feature, i.e., the flow label field in the IPv6 packet header. The flow label is not encrypted by IPsec, and is present in all fragments. However, it is very little used in practice, for reasons discussed below and in [1]. After a short introduction, this document summarizes the current specification of the IPv6 flow label and some open issues about its use in Section 2, and then Section 3 describes and analyses various proposals that have been made for its use. Finally, Section 4 discusses the implications and attempts to draw conclusions.

1.1. A brief history of the flow label

The original proposal for a flow label has been attributed to Dave Clark [2], who proposed that it should contain a pseudo-random value. A flow label field was included in the packet header during the preliminary design of IPv6, which followed an intense period of debate about several competing proposals. The final choice was made in 1994 [3]. In particular, the IETF rejected a proposal known as CATNIP [4], which included so-called 'cache handles' to identify the next hop in high performance routers. Thus CATNIP introduced the notion of a header field that would be shared by all packets belonging to a flow, on a hop-by-hop basis. We recognize this today as a precursor of the MPLS label [5]. However, the IETF decided instead to develop a proposal known as SIPP into IP version 6. SIPP included "labeling of packets belonging to particular traffic 'flows' for which the sender requests special handling, such as non-default quality of service or 'real-time' service" [6]. In 1994, this was a 28-bit flow label field. In 1995 it was down to 24 bits [7] and it was finally reduced to 20 bits [8] to accommodate the IPv6 Traffic Class, which is fully compatible with the IPv4 Type of Service byte.

There was considerable debate in the IETF about the very purpose of the flow label. Was it to be a handle for fast switching, as in

CATNIP, or was it to be meaningful to applications and used to specify quality of service? Must it be set by the sending host, or could it be set by routers? Could it be modified en route, or must it be delivered with no change? Because of these uncertainties, and more urgent work, the flow label was consistently ignored by implementors, and today is set to zero in almost every IPv6 packet. In fact, [8] defined it as "experimental and subject to change." There was considerable preliminary work such as [9], [10], [11] and [12]. The ensuing proposed standard "IPv6 Flow Label Specification" (RFC 3697) [13] intended to clarify this situation by providing precise boundary conditions for use of the flow label. However, this has not proved successful in promoting use of the flow label in

practice, which can still be described quite accurately as a waste of space in every IPv6 packet.

1.2. The flow label and quality of service

Various use cases for the flow label have been proposed, many of them assuming that it should be used principally to support the provision of quality of service (QoS). For many years it has been recognized that real-time Internet traffic requires a different QoS from general data traffic, and this remains true in the era of network neutrality. Thus an alternative to uniform best-effort service is needed, requiring packets to be classified as belonging to a particular class of service or flow. Currently, this leads to a layer violation problem, since a 5-tuple is often used to classify each packet. The 5-tuple includes source and destination addresses, port numbers, and the transport protocol type, so when we want to forward or process packets, we need to extract information from the layer above IP. This may be impossible when packets are encrypted such that port numbers are hidden, or when packets are fragmented, so the layer violation is not an academic concern. The flow label, being exempt from IPsec encryption and being replicated in packet fragments, avoids this difficulty. It has therefore attracted attention from the designers of new approaches to QoS.

2. Flow label definition and issues

2.1. Flow label properties

The flow label field occupies bits 12 through 31 of the IPv6 packet header. It provides a potential way to mark a packet, identify a flow, and look up the corresponding flow state. This field is always present in an IPv6 header, so a phrase such as "a packet with no flow label" refers to a packet whose flow label field contains 20 zero bits, i.e., a flow label whose value is zero.

RFC 3697 [13] standardizes properties of the flow label, including:

- o If the packets are not part of any flow, the flow label value is zero.
- o The 3-tuple {source address, destination address, flow label} uniquely identifies which packets belong to which particular flow.
- o Packets can receive flow-specific treatment if the node has been set up with flow-specific state.
- o The flow label set by the source node must be delivered to the destination node, i.e., it is an end-to-end label.
- o The same pair of source and destination must not use the same flow label value again within a timeout of at least 120 seconds.

One effect of the second of these rules is to avoid the layer violation problem mentioned in Section 1. By using the 3-tuple, we only use the IP layer to classify packets, without needing any transport layer information. This may reduce the lookup time if flow-based treatment is required, and will work even with IPsec encryption and fragmentation. Therefore, for traffic needing other than best-effort service, such as real-time applications, the flow label can be set to different values to represent different flows, and each node forwarding or receiving the packets may provide different flow-specific treatments by looking at the flow label value. This is more fine-grained than differentiated services (Diffserv) [14], [15] but need not be less efficient.

2.2. Dependency prohibition

An additional important rule in the standard [13] effectively forbids any encoding of meaning in the bits of the flow label. To be exact, the standard states that "IPv6 nodes MUST NOT assume any mathematical or other properties of the flow Label values assigned by source nodes." This rule is aimed at the case where a packet from a source using a particular encoding scheme for the flow label reaches a node that is using a different scheme. If by chance the bit pattern in

the flow label is meaningful in both schemes, the receiver would misinterpret the flow label. Therefore, in the absence of other information, the receiver must not assume anything about the meaning of the value of the flow label.

The standard [13] also states "Router performance SHOULD NOT be dependent on the distribution of the Flow Label values. Especially, the Flow Label bits alone make poor material for a hash key." The problem this rule is intended to avoid is that if a source uses one method of choosing flow labels (e.g., counting up from 1), any router that assumes another method (e.g., pseudo-randomness) will be misled.

Note that there is no easy escape from the combination of these two prohibitions, which we will call the dependency prohibition. Unlike Diffserv code points, flow labels are not locally significant within a single administrative domain; they must be preserved end-to-end. In general, a router cannot know whether a particular packet originated in a host supporting a specific usage of the flow label. Therefore, any method that breaks one or both of these rules will only work if there is some way for a router to determine which sources use the same scheme as itself.

2.3. Other issues

[13] does not discuss how to use flow label most effectively. This remains the major open issue, but some authors propose that the label

should be used with reserved bandwidth to achieve customized QoS provision. Coupled with admission control at the edge router, this could limit congestion. However, as we will see below, this is not the only proposed use.

We now introduce some other open issues.

- Unknown flow labels: [16] proposed that when a router receives a datagram with an unknown flow label, it should treat it as zero. However, the standard [13] is silent on this issue. Indeed, some methods of flow state establishment might choose to use an unknown label as the trigger for creating flow state.
- Deleting old flow labels: When a flow finishes, how does the router know the flow label has expired? Should this be based on a timeout, on observation of the transport layer, or on explicit signalling? The standard defines a timeout (120 seconds) after

which a receiving node should discard a previously recorded flow label if there is no more traffic. However, this will be unsatisfactory in the case of a very intermittent flow. In contrast, [17] suggested that a router should send an ICMP message to the source to delete a particular label. The source node can then either send a KEEPALIVE message to the router, or it can allow the router to release that label.

- o Choosing when to set the flow label: For what kinds of application should we set up non-zero flow labels? [16] suggested not setting it for short flows containing few bytes, but using it for long TCP connections and some real-time applications. However, this does not really define clear use cases.
- o Can we modify the flow label? [13] states that the flow label must be delivered unchanged. There are several advantages of immutable flow labels, apart from respecting the standard: the rule is easy to understand, does not require extra processing in routers or a signalling protocol, and allows for very simple host implementations. Also, it is straightforward for hosts and routers to simply ignore the flow label. However, this rule does appear to exclude any MPLS-like or CATNIP-like use for optimized packet switching. Some authors have objected to this feature, suggesting that switches should change the flow label for routing purposes. We will describe these and other proposed mechanisms in next section.

3. Documented proposals for the flow label

In the following, we do not intend to recommend or criticise various proposals. This section shows the variety of proposals that have been published, and whether they are compatible with the existing standard. Most published proposals for the flow label assume that its main purpose is to support QoS, and their flow label mechanisms

are entangled with QoS mechanisms. We describe the proposals in five broad categories, i.e.,

(1) use pseudo-random flow label values for various purposes (for example, to improve routing performance when retrieving cached routing state)
(2) define specific QoS requirements as parameters embedded in the flow label field;
(3) use the flow label to control packet switching;

(4) use the flow label to extend the existing differentiated services QoS architecture;
(5) other uses.

Across these categories, we observe various options to set up the flow label value, described in the following sections. It should be noted that some of these proposals embody novel and perhaps controversial approaches to QoS provision, and these cannot readily be separated from their use of the flow label.

3.1. Specify the flow label as a pseudo-random value

As our first example, [18] specifies a 17-bit pseudo-random value. The figure below shows the proposed flow label structure.

- o The Label Flag (LF) bit: 1 means this type of flow label is present. We note that this encoding is incompatible with the dependency prohibition in [13], since a source that does not use this method may also set the LF bit.
- o The Label type (LT): 2 bits which describe the type of packet.
- o The Label Number (LN): which is randomly generated by the source node.

[18] also describes a signalling process between source, routing and destination nodes based on this label structure and on the IPv6 Traffic Class byte, in order to reserve and release router resources for a given flow within a given class of traffic. The pseudo-random LN value is used to uniquely identify a given flow.

Flow Label Specification (figure simplified from [18])

```
+--+----+-----------------------------+
| 1| 2  |           17 bits           |
+--+----+-----------------------------+
|LF| LT |             LN              |
+--+----+-----------------------------+
```

```
LF    0  Disable
      1  Enable
LT   00  Flow label requested by source
     01  Flow label returned by destination
     10  Flow label for data delivery
     11  Flow label terminates connection
LN       Random number created by source
```

There have been numerous informal discussions of using pseudo-random flow labels to allow load-balancing or at least load-sharing. This would be achieved by including the flow label value among the fields in each packet header used as input to a modulo(N) hash used to select among N alternative paths. However, concerns about the dependency prohibition have generally prevented such proposals from being written up until recently [19].

Another proposal for a pseudo-random flow label value is [20]. This states that off-path spoofing attacks have become a big issue for TCP and other transport-layer applications, and proposes that in IPv6 we should set a random value in the flow label to make the packet header more complex and less easy for the attacker to guess. The two ends of the session will agree on flow label values during the SYN/ACK exchange, but off-path attackers will be unlikely to guess the agreed value. Naturally, on-path attackers who can observe the flow labels in use can trivially defeat this protection. This proposal does not involve using the flow label value to retrieve routing state.

3.2. Specify QoS parameters in the flow label

[21] proposes to utilize the flow label to indicate required QoS parameters in detail. It uses the first few bits of the flow label field as codes to support different approaches, as summarized in following table. Again, this is incompatible with the dependency prohibition in [13], since a source that does not use this method may also set the first two bits to non-zero.

Classification for various approaches (from [21])

Bit Pattern	Approach
00	No QoS requirement (Default QoS value)
01	Pseudo-Random value used for the value of Flow-Label
10	Support for Direct Parametric Representation
1100	Support for the DiffServ Model
1101	Reserved for future use
111	Reserved for future use

This method allows a pseudo-random option, but also adds options for a direct QoS request and for Diffserv. In the direct QoS parameters approach, 18 bits are used to encode requirements for one way delay, IP delay variation, bandwidth and one way packet loss. The proposal appears to assume that RSVP [22] mechanisms are used to actually implement these QoS parameters.

This proposal allows use of flow label for various important QoS models, so the end user and service provider can choose the most suitable model for their situation; [21] claims that "this proposal is simple, scalable, modular and generic implementation to provide for QoS using the IPv6 flow label field".

Similarly, [23] defines the flow label field in five parts, with the first 3 bits used as an approach type. The authors define two approaches: a "random" scheme and a "hybrid" scheme. If the first 3 bits equal "001", the flow label will be used as the random identifier of the flow, but if they equal "101", the remaining bits will include a hybrid QoS requirement for this packet, subdivided into traffic type (stringent or best effort), bandwidth, buffer, and delay requirements. Once again the dependency prohibition in [13] is broken. This proposal also includes throughput monitoring and dynamic capacity allocation. Effectively this proposal uses the flow label both to signal Intserv-like QoS requirements and to classify traffic into Diffserv-like virtual label-switched paths. Packets with a "random" flow label are mapped into a generic (best effort) virtual path.

The authors simulated this architecture for a network of fourteen nodes, using the NS-2 simulator with a weighted fair queueing extension to support their "hybrid" scheme. Their results indicate that the "hybrid" scheme improves capacity utilization for QoS traffic and performance for real-time traffic, rather in the manner of differentiated services, whereas competing traffic using the "random" scheme simply experiences best effort service. It must be

considered, however, that NS-2 does not accurately simulate the performance of typical high-performance routers.

3.3. Use flow label hop-by-hop to control switching

[24] and [25] describe an architectural framework called "IPv6 Label Switching Architecture" (6LSA). In 6LSA, network components identify a flow by looking at the flow label field in the IPv6 packet header; all packets with the same flow label must receive the same treatment and be sent to the same next hop. However, 6LSA resembles MPLS by considering that a label only has meaning between 6LSA routers, and setting the flow label at each hop. If the original source sets a non-zero flow label, there is no mechanism to restore it before delivery, a fundamental breach of [13]. The authors of [24] did at one stage discuss using an IPv6 hop-by-hop option to correct this problem, but this has not been documented. This is a more serious incompatibility than simply breaking the dependency prohibition

Unlike traditional routing algorithms, but like MPLS, 6LSA packets are classified into a Forwarding Equivalence Class (FEC), and routers forward packets on different paths by looking at the FEC. Like previous solutions, the authors divide the flow label field into three parts. The first 3 bits identify the FEC, which will help the router or 6LSA nodes to group the IP packets that receive the same forwarding treatment and forwarding them on the same virtual path. The following 4 bits describe the application type, and the final 13 bits (defined by each node or a group of nodes) specify the hop-specific label. From the table below, we can see the FEC has 6 major categories, each with up to 16 subcategories.

Flow Label Specification (shortened from [24])

FEC (First 3 Bits)	Next 4 Bits	Purpose
No FEC (000)	0000	
Domain Specific (000)	0001 - 1111	

VPN (001)	0001	(IPSec - Tunnel Mode)
	0010	(IPSec - Transport Mode)
	0011	(Special Encryption)
	0100	(VRF)
	0101	(End Network Specific)
	0110 - 1111	(Reserved)

TE Subset/ QoS Enhancement (010)	0001	(DiffServ)
	0010	(RSVP)
. . .		
	1111	(Reserved)

Encapsulation (011)	0001	(IPv6 in IPv6)
	0010	(IPv4 in IPv6)
	0011	(Other in IPv6)
	0100	(Enterprise Specific)
	0101 - 1111	(Reserved)

Enterprise Specific(111)	0000 - 1111	(Reserved)

The authors claim that fast switching using 20-bit labels instead of 128-bit IPv6 addresses will provide memory and processing savings, as well as network management advantages. "It also allows a network

management entity updating available label tables, across the network to reduce man-in-the-middle attacks [sic]" [24].

We note that a similar proposal for QoS-based switching of IPv6 packets [26] is designed to use a hop-by-hop option, which has not so far been allocated by the IETF. Proposals related to this have been discussed by the Telecommunications Industry Association and the ITU-T [27].

We also note that router lookup efficiency was a major concern at the time when Clark first proposed a flow label [2], but with the advent of very large scale integrated circuits capable of rapid lookup in a routing table, most vendors no longer express such concern.

3.4. Diffserv use of IPv6 flow label

[17] uses the flow label field as a replacement for the IPv6 Traffic Class field; this proposal suggests the incoming flow label can indicate the QoS requirement by matching a Diffserv classifier. The authors have used the first three bits to indicate this, and the following 16 bits to indicate a Differentiated Services Per-Hop Behavior Identification code (Diffserv PHB-ID) [28]; the last bit is reserved for future use. This method too breaks the dependency prohibition in [13].

[29] blends the flow label as an MPLS-like switching tag with Diffserv. Unlike 6LSA, the method attempts to bypass the dependency prohibition by using one bit in the Diffserv Code Point [15] to indicate that the flow label is a switching tag. In this way, a router can determine whether the flow label conforms to [13] or to [29]. In [30], the same author proposes using the flow label as a way of compressing IPv6 headers by hashing the addresses into the flow label, again using the Diffserv Code Point to mark the packets accordingly.

3.5. Other uses

We are not aware of any proposals combining the flow label with the other two Internet QoS architectures (Integrated Services [31] and

Next Steps in Signaling (NSIS) [32]), except for recognition that the flow label can be used as a packet filter [22].

[33] proposes a use case whereby certain flows encapsulated in a particular type of IPv4-in-IPv6 tunnel would be distinguished at the remote end of the tunnel by a specific flow label value. This would allow a service provider to deliver a tailored quality of service. This usage appears to be completely compatible with [13].

There has been some discussion of possible flow label use in both the ROLL (Routing Over Low power and Lossy networks) [34] and MEXT (Mobility EXTensions for IPv6) working groups of the IETF. Such uses tend to encode specific local meanings or routing-related tags in the label, so they appear to infringe the dependency prohibition or the immutability of the flow label field. The ROLL group has indeed most recently opted not to use the flow label field for these reasons, despite having to add the undesirable overhead of an IPv6 hop-by-hop option instead [35]. Similarly, MEXT has defined a new mobility option to support flow bindings [36], rather than using the IPv6 flow label field.

4. Discussion

Three aspects of the current standard [13] have caused problems to many designers:

1. The immutability of labels
2. "Nodes MUST NOT assume any mathematical or other properties of the Flow Label"
3. "Router performance SHOULD NOT be dependent on the distribution of the Flow Label values."

Taken together, these rules essentially forbid any encoding of the semantics of a flow, or of any information about its path, in the flow label. This was intentional, in accordance with the stateless nature of the Internet architecture and with the end-to-end principle [37], [38]. It was also felt that QoS encoding via Diffserv was sufficient, and that the requirement for high-speed switching could be met by MPLS. But this means that the majority of the proposals described above breach the standard and the intent of the standard.

The authors often appear to be using the flow label either as an MPLS-like switching handle, or as an encoded QoS signal.

In contrast, a few documents metioned above do appear to respect the rules of RFC 3697. These are [20], [33], [19], [29] and [30].

What would other designers need to do, if they wish to respect RFC 3697? There appear to be two choices. One is to simply accept the existing rules at face value, as in the proposals just listed. This limits the application of the flow label. It can, for example, be used as a nonce or as part of the material for a hash used to share load among alternate paths. It cannot be the only material for such a hash, because of the dependency prohibition. The flow label could also be used consistently with RFC 3697, if an application designer so chose, as a way to associate all packets belonging to a given application session between two hosts, across multiple transport sessions. This, however, would presumably exclude using the flow label to govern routing decisions in any way, and would have widespread implications that have never been explored.

The other choice, for designers who wish to use the flow label to control switching or QoS directly, is to bypass the rules within a given domain (a set of cooperating nodes) in a way that nodes outside the domain cannot detect. In this case, any deviation from RFC 3697 has no possible effect outside the domain in question.

An example scheme to emulate the immutability of labels is as follows. Within the domain, all hosts set the label to zero, the routers set and interpret the label in any way they wish, and the last hop router always sets the label back to zero. If a packet

arrives from outside the domain with a non-zero label, there is a method (such as a special Diffserv code point) to mark this packet so that its label would be ignored and delivered unchanged. An alternative approach would be to define a hop-by-hop option to carry the original flow label across the domain, so that it could be changed within the domain but restored before forwarding the packet beyond the domain.

If a domain allows mutable labels in such a way, it may safely ignore the dependency prohibition, and it may set the bits in the label according to locally defined rules. Within the domain, the label

could be used as desired, and most of the proposed designs discussed above could be "rescued."

However, given the considerable number of designers who have proposed solutions incompatible with RFC 3697, the relatively few designs using the standard rules, and the failure of designs such as ROLL and MEXT to make use of the flow label, it seems reasonable to ask whether the current standard has value.

5. Security Considerations

The flow label is not protected in any way and can be forged by an on-path attacker. Off-path attackers may be able to guess a valid flow label unless a pseudo-random value is used. Specific usage models for the flow label need to allow for these exposures. For further discussion, see [13].

6. IANA Considerations

This document requests no action by IANA.

7. Acknowledgements

An invaluable review of this document was performed by Bob Braden. Helpful comments were made by Sheng Jiang.

This document was produced using the xml2rfc tool [39].

8. Change log

draft-hu-flow-label-cases-00: first I-D version, 2010-04-19

draft-hu-flow-label-cases-01: updated following review comments,

2010-08-18

draft-hu-flow-label-cases-02: updated following more review comments, 2010-09-30

9. Informative References

[1] Amante, S., Carpenter, B., and S. Jiang, "Update to the IPv6 flow label specification", draft-carpenter-6man-flow-update-04 (work in progress), September 2010.

[2] Deering, S., "SIPP Overview and Issues", Minutes of the Joint Sessions of the SIP and PIP Working Groups , November 1993.

[3] Bradner, S. and A. Mankin, "The Recommendation for the IP Next Generation Protocol", RFC 1752, January 1995.

[4] McGovern, M. and R. Ullmann, "CATNIP: Common Architecture for the Internet", RFC 1707, October 1994.

[5] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

[6] Hinden, R., "Simple Internet Protocol Plus White Paper", RFC 1710, October 1994.

[7] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 1883, December 1995.

[8] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[9] Metzler, J. and S. Hauth, "An end-to-end usage of the IPv6 flow label", draft-metzler-ipv6-flowlabel-00 (work in progress), November 2000.

[10] Conta, A. and B. Carpenter, "A proposal for the IPv6 Flow Label Specification", draft-conta-ipv6-flow-label-02 (work in progress), July 2001.

[11] Conta, A. and J. Rajahalme, "Amodel for Diffserv use of the IPv6 Flow Label Specification", draft-conta-diffserv-ipv6-fl-classifier-01 (work in progress), November 2001.

[12] Hagino, J., "Socket API for IPv6 flow label field", draft-itojun-ipv6-flowlabel-api-01 (work in progress),

April 2001.

[13] Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification", RFC 3697, March 2004.

[14] Carpenter, B. and K. Nichols, "Differentiated Services in the Internet", Proc IEEE 90 (9) 1479-1494, 2002.

[15] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[16] Partridge, C., "Using the Flow Label Field in IPv6", RFC 1809, June 1995.

[17] Banerjee, R., "A Modified Specification for use of the IPv6 Flow Label for providing An efficient Quality of Service using hybrid approach", draft-banerjee-flowlabel-ipv6-qos-03 (work in progress), April 2002.

[18] Lin, C., Tseng, P., and W. Hwang, "End-to-End QoS Provisioning by Flow Label in IPv6", JCIS , 2006.

[19] Carpenter, B. and S. Amante, "Using the IPv6 flow label for equal cost multipath routing and link aggregation in tunnels", draft-carpenter-flow-ecmp-02 (work in progress), April 2010.

[20] Blake, S., "Use of the IPv6 Flow Label as a Transport-Layer Nonce to Defend Against Off-Path Spoofing Attacks", draft-blake-ipv6-flow-label-nonce-02 (work in progress), October 2009.

[21] Prakash, B., "Using the 20 bit flow label field in the IPv6 header to indicate desirable quality of service on the internet", University of Colorado (M.Sc. Thesis), 2004.

[22] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[23] Lee, I. and S. Kim, "A QoS Improvement Scheme for Real-Time Traffic Using IPv6 Flow Labels", Lecture Notes in Computer Science Vol. 3043, 2004.

[24] Chakravorty, S., Bush, J., and J. Bound, "IPv6 Label Switching Architecture", draft-chakravorty-6lsa-03 (work in progress), July 2008.

[25] Chakravorty, S., "Challenges of IPv6 Flow Label implementation", Proc IEEE MILCOM2008 , 2008.

[26] Roberts, L. and J. Harford, "In-Band QoS Signaling for IPv6", draft-roberts-inband-qos-ipv6-00 (work in progress), July 2005.

[27] song, j., Adams, J., and J. Joung, "Progress and future development of Flow State Aware standards, and a proposal for alerting nodes or end-systems on data related to a flow", draft-adams-tsvwg-flow-signalling-codepoint-00 (work in progress), June 2008.

[28] Black, D., Brim, S., Carpenter, B., and F. Le Faucheur, "Per Hop Behavior Identification Codes", RFC 3140, June 2001.

[29] Beckman, M., "IPv6 Dynamic Flow Label Switching (FLS)", draft-martinbeckman-ietf-ipv6-fls-ipv6flowswitching-03 (work in progress), March 2007.

[30] Beckman, M., "IPv6 Header Compression via Addressing Mitigation Protocol (IPv6 AMP)", draft-martinbeckman-ietf-ipv6-amp-ipv6hcamp-01 (work in progress), March 2007.

[31] Braden, B., Clark, D., and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633, June 1994.

[32] Hancock, R., Karagiannis, G., Loughney, J., and S. Van den Bosch, "Next Steps in Signaling (NSIS): Framework", RFC 4080, June 2005.

[33] Donley, C. and K. Erichsen, "Using the Flow Label with Dual-Stack Lite", draft-donley-softwire-dslite-flowlabel-00 (work in progress), July 2010.

[34] Winter, T. and P. Thubert, "RPL: IPv6 Routing Protocol for Low power and Lossy Networks", draft-ietf-roll-rpl-07 (work in progress), March 2010.

[35] Winter, T., Thubert, P., and R. Team, "RPL: IPv6 Routing

Protocol for Low power and Lossy Networks",
draft-ietf-roll-rpl-11 (work in progress), July 2010.

[36] Tsirtsis, G., Soliman, H., Montavont, N., Giaretta, G., and K. Kuladinithi, "Flow Bindings in Mobile IPv6 and NEMO Basic Support", draft-ietf-mext-flow-binding-10 (work in progress), September 2010.

[37] Saltzer, J., Reed, D., and D. Clark, "End-To-End Arguments in System Design", ACM TOCS 2 (4) 277-288, 1984.

[38] Kempf, J., Austein, R., and IAB, "The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture", RFC 3724, March 2004.

[39] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

Authors' Addresses

Qinwen Hu
Department of Computer Science
University of Auckland
PB 92019
Auckland, 1142
New Zealand

Email: qhu009@aucklanduni.ac.nz

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland, 1142
New Zealand

Email: brian.e.carpenter@gmail.com