Network Working Group
Request for Comments: 1809
Category: Informational

C. Partridge
BBN Systems and Technologies

June 1995

Using the Flow Label Field in IPv6

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

The purpose of this memo is to distill various opinions and suggestions of the End-to-End Research Group regarding the handling of Flow Labels into a set of suggestions for IPv6. This memo is for information purposes only and is not one of the IPv6 specifications. Distribution of this memo is unlimited.

Introduction

This memo originated as the report of a discussion at an End-to-End Research Group meeting in November 1994. At that meeting the group discussed several issues regarding how to manage flow identifiers in IPv6. A report of the meeting was then circulated to the IPv6 community. Feedback from that community resulted in changes to this memo and in changes to the IPv6 specification to fix some minor problems the End-to-End Group had raised.

While many of the ideas in this memo have found their way into the IPv6 specification, the explanation of why various design decisions were made have not. This memo is intended to provide some additional context for interested parties.

Brief Description of the Flow Label

The current draft of the IPv6 specification states that every IPv6 header contains a 24-bit Flow Label. (Originally the specification called for a 28-bit Flow ID field, which included the flow label and a 4-bit priority field. The priority field is now distinct, for reasons discussed at the end of this memo).

The Flow Label is a pseudo-random number between 1 and FFFFFF (hex) that is unique when combined with the source address. The zero Flow Label is reserved to say that no Flow Label is being used. The specification requires that a source must not reuse a Flow Label value until all state information for the previous use of the Flow Label has been flushed from all routers in the internet.

The specification further requires that all datagrams with the same (non-zero) Flow Label must have the same Destination Address, Hop-by-Hop Options header, Routing Header and Source Address contents. The notion is that by simply looking up the Flow Label in a table, the router can decide how to route and forward the datagram without examining the rest of the header.

Flow Label Issues

The IPv6 specification originally left open a number of questions, of which these three were among the most important:

1. What should a router do if a datagram with a (non-zero) Flow Label arrives and the router has no state for that Flow Label?

2. How does an internet flush old Flow Labels?

3. Which datagrams should carry (non-zero) Flow Labels?

This memo summarizes the End-to-End Group’s attempts to answer these questions.

What Does a Router Do With Flow Labels for Which It Has No State?

If a datagram with a non-zero Flow Label arrives at a router and the router discovers it has no state information for that Flow Label, what is the correct thing for the router to do?

The IPv6 specification allows routers to ignore Flow Labels and also allows for the possibility that IPv6 datagrams may carry flow setup information in their options. Unknown Flow Labels may also occur if a router crashes and loses its state. During a recovery period, the router will receive datagrams with Flow Labels it does not know, but this is arguably not an error, but rather a part of the recovery period. Finally, if the controversial suggestion that each TCP connection be assigned a separate Flow Label is adopted, it may be necessary to manage Flow Labels using an LRU cache (to avoid Flow Label cache overflow in routers), in which case an active but infrequently used flow’s state may have been intentionally discarded.

In any case, it is clear that treating this situation as an error and, say dropping the datagram and sending an ICMP message, is inappropriate. Indeed, it seems likely that in most cases, simply forwarding the datagram as one would a datagram with a zero Flow Label would give better service to the flow than dropping the datagram.

Of course, there will be situations in which routing the datagram as if its Flow Label were zero will cause the wrong result. An example is a router which has two paths to the datagram's destination, one via a high-bandwidth satellite link and the other via a low-bandwidth terrestrial link. A high bandwidth flow obviously should be routed via the high-bandwidth link, but if the router loses the flow state, the router may route the traffic via the low-bandwidth link, with the potential for the flow's traffic to swamp the low-bandwidth link. It seems likely, however, these situations will be exceptions rather than the rule. So it seems reasonable to handle these situations using options that indicate that if the flow state is absent, the datagram needs special handling. (The options may be Hop-by-Hop or only handled at some routers, depending on the flow's needs).

It would clearly be desirable to have some method for signalling to end systems that the flow state has been lost and needs to be refreshed. One possibility is to add a state-lost bit to the Flow Label field, however there is sensitivity to eating into the precious 24-bits of the field. Other possibilities include adding options to the datagram to indicate its Flow Label was unknown or sending an ICMP message back to the flow source.

In summary, the view is that the default rule should be that if a router receives a datagram with an unknown Flow Label, it treats the datagram as if the Flow Label is zero. As part of forwarding, the router will examine any hop-by-hop options and learn if the the datagram requires special handling. The options could include simply the information that the datagram is to be dropped if the Flow Label is unknown or could contain the flow state the router should have. There is clearly room here for experimentation with option design.

Flushing Old Flow Labels

The flow mechanism assumes that state associated with a given Flow Label is somehow deposited in routers, so they know how to handle datagrams that carry the Flow Label. A serious problem is how to flush Flow Labels that are no longer being used (stale Flow Labels) from the routers.

Stale Flow Labels can happen a number of ways, even if we assume that the source always sends a message deleting a Flow Label when the

source finishes using a Flow. An internet may have partioned since the flow was created. Or the deletion message may be lost before reaching all routers. Furthermore, the source may crash before it can send out a Flow Label deletion message. The point here is that we cannot expect the source (or, for the same reasons, a third party) always to clear out stale Flow Labels. Rather, routers will have to find some mechanism to flush Flow Labels themselves.

The obvious mechanism is to use a timer. Routers should discard Flow Labels whose state has not been refreshed within some period of time. At the same time, a source that crashes must observe a quiet time, during which it creates no flows, until it knows that all Flow Labels from its previous life must have expired. (Sources can avoid quiet time restrictions by keeping information about active Flow Labels in stable storage that survives crashes). This is precisely how TCP initial sequence numbers are managed and it seems the same mechanism should work well for Flow Labels.

Exactly how the Flow Label and its state should be refreshed needs some study. There are two obvious options. The source could periodically send out a special refresh message (such as an RSVP Path message) to explicitly refresh the Flow Label and its state. Or, the router could treat every datagram that carries the Flow Label as an implicit refresh or sources could send explicit refresh options. The choice is between periodically handling a special update message and doing an extra computation on each datagram (namely noting in the Flow Label's entry that the Flow Label has been refreshed).

Which Datagrams Should Carry (Non-Zero) Flow Labels?

Interestingly, this is the problem on which the least progress has been made.

There were some points of basic agreement. Small exchanges of data should have a zero Flow Label, because it is not worth creating a flow for a few datagrams. Real-time flows must obviously always have a Flow Label, since flows are a primary reason Flow Labels were created. The issue is what to do with peers sending large amounts of best effort traffic (e.g., TCP connections). Some people want all long-term TCP connections to use Flow Labels, others do not.

The argument in favor of using Flow Labels on individual TCP connections is that even if the source does not request special service, a network provider's routers may be able to recognize a large amount of traffic and use the Flow Label field to establish a special route that gives the TCP connection better service (e.g., lower delay or bigger bandwidth). Another argument is to assist in efficient demux at the receiver (i.e., IP and TCP demuxing could be

done once).

An argument against using Flow Labels in individual TCP connections is that it changes how we handling route caches in routers. Currently one can cache a route for a destination host, regardless of how many different sources are sending to that destination host. I.e., if five sources each have two TCP connections sending data to a server, one cache entry containing the route to the server handles all ten TCPs' traffic. Putting Flow Labels in each datagram changes the cache into a Flow Label cache, in which there is a cache entry for every TCP connection. So there's a potential for cache explosion. There are ways to alleviate this problem, such as managing the Flow Label cache as an LRU cache, in which infrequently used Flow Labels get discarded (and then recovered later). It is not clear, however, whether this will cause cache thrashing.

Observe that there is no easy compromise between these positions. One cannot, for instance, let the application decide whether to use a Flow Label. Those who want different Flow Labels for every TCP connection assume that they may optimize a route without the application's knowledge. And forcing all applications to use Flow Labels will force routing vendors to deal with the cache explosion issue, even if we later discover that we don't want to optimize individual TCP connections.

Note about the Priority Field

The original IPv6 specification combined the Priority and Flow Label fields and allowed flows to redefine the means of different values of the Priority field. During its discussions, the End-to-End group realized this meant that if a router forwarded a datagram with an unknown Flow Label it had to ignore the Priority field, because the priority values might have been redefined. (For instance, the priorities might have been inverted). The IPv6 community concluded this behavior was undesirable. Indeed, it seems likely that when the Flow Label are unknown, the router will be able to give much better service if it use the Priority field to make a more informed routing decision. So the Priority field is now a distinct field, unaffected by the Flow Label.

Acknowledgements

I would like to acknowledge the assistance of the members of the End-To-End Research Group, chaired by Bob Braden, whose discussions produced this memo. I would also like to particularly thank Deborah Estrin for her help in putting this memo together. Also thanks to Richard Fox, Noel Chiappa, and Tony Li for insightful comments on the draft.

Security Considerations

Security issues are not discussed in this memo.

Author's Address

Craig Partridge
BBN Systems and Technologies
10 Moulton St.
Cambridge, MA 02138

EMail: craig@aland.bbn.com