6MAN
Internet-Draft
Obsoletes: 3697 (if approved)
Updates: 2205, 2460 (if approved)
Intended status: Standards Track
Expires: November 12, 2011

S. Amante
Level 3
B. Carpenter
Univ. of Auckland
S. Jiang
Huawei Technologies Co., Ltd
J. Rajahalme
Nokia Siemens Networks

May 11, 2011

IPv6 Flow Label Specification
draft-ietf-6man-flow-3697bis-04

Abstract

This document specifies the IPv6 Flow Label field and the minimum requirements for IPv6 nodes labeling flows, IPv6 nodes forwarding labeled packets, and flow state establishment methods. Even when mentioned as examples of possible uses of the flow labeling, more detailed requirements for specific use cases are out of scope for this document.

The usage of the Flow Label field enables efficient IPv6 flow classification based only on IPv6 main header fields in fixed positions.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

From the viewpoint of the network layer, a flow is a sequence of packets sent from a particular source to a particular unicast, anycast, or multicast destination that a node desires to label as a flow. From an upper layer viewpoint, a flow could consist of all packets in a specific transport connection or a media stream. However, a flow is not necessarily 1:1 mapped to a transport connection.

Traditionally, flow classifiers have been based on the 5-tuple of the source and destination addresses, ports, and the transport protocol type. However, some of these fields may be unavailable due to either fragmentation or encryption, or locating them past a chain of IPv6 extension headers may be inefficient. Additionally, if classifiers depend only on IP layer headers, later introduction of alternative transport layer protocols will be easier.

The usage of the 3-tuple of the Flow Label and the Source and Destination Address fields enables efficient IPv6 flow classification, where only IPv6 main header fields in fixed positions are used.

The flow label could be used in both stateless and stateful scenarios. A stateless scenario is one where any node that processes the flow label in any way does not need to store any information about a flow before or after a packet has been processed. A stateful scenario is one where a node that processes the flow label value needs to store information about the flow, including the flow label value. A stateful scenario might also require a signaling mechanism to establish flow state in the network.

The flow label can be used most simply in stateless scenarios. This specification concentrates on the stateless model and how it can be used as a default mechanism. Details of stateful models, signaling, specific flow state establishment methods and their related service models are out of scope for this specification. The basic requirement for stateful models is set forth in Section 4.

The minimum level of IPv6 flow support consists of labeling the flows. A specific goal is to enable and encourage the use of the flow label for various forms of stateless load distribution, especially across Equal Cost Multi-Path (EMCP) and/or Link Aggregation Group (LAG) paths. ECMP and LAG are methods to bond together multiple physical links used to procure the required capacity necessary to carry an offered load greater than the bandwidth of an individual physical link. IPv6 source nodes SHOULD be able to label known flows (e.g., TCP connections, application

streams), even if the node itself does not require any flow-specific treatment. Node requirements for stateless flow labeling are given in Section 3.

This document replaces [RFC3697] and Section 6 and Appendix A of [RFC2460]. A rationale for the changes made is documented in [I-D.ietf-6man-flow-update]. The present document also includes a correction to [RFC2205] concerning the flow label.

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 [RFC2119].

2. IPv6 Flow Label Specification

The 20-bit Flow Label field in the IPv6 header [RFC2460] is used by a node to label packets of a flow. A Flow Label of zero is used to indicate packets that have not been labeled. Packet classifiers can use the triplet of Flow Label, Source Address, and Destination Address fields to identify which flow a particular packet belongs to. Packets are processed in a flow-specific manner by nodes that are able to do so in a stateless manner, or that have been set up with flow-specific state. The nature of the specific treatment and the methods for flow state establishment are out of scope for this specification.

Flow label values should be chosen such that their bits exhibit a high degree of variability, making them suitable for use as part of the input to a hash function used in a load distribution scheme. At the same time, third parties should be unlikely to be able to guess the next value that a source of flow labels will choose.

In statistics, a discrete uniform distribution is defined as a probability distribution in which each value in a given range of equally spaced values (such as a sequence of integers) is equally likely to be chosen as the next value. The values in such a distribution exhibit both variability and unguessability. Thus, as specified below in Section 3, an approximation to a discrete uniform distribution is preferable as the source of flow label values. Intentionally, there are no precise mathematical requirements placed on the distribution or the method used to achieve such a distribution.

Once set to a non-zero value, the Flow Label MUST be delivered unchanged to the destination node(s). That is, a forwarding node MUST NOT change the flow label value in an arriving packet if it is non-zero. A possible exception to this rule is if a security gateway

for operational security reasons changes a non-zero Flow Label value to a different non-zero value compliant with this RFC; see Section 6.1 for details.

There is no way to verify whether a flow label has been modified en route or whether it belongs to a uniform distribution. Therefore, no Internet-wide mechanism can depend mathematically on unmodified and uniformly distributed flow labels; they have a "best effort" quality. Implementers should be aware that the flow label is an unprotected field that could have been accidentally or intentionally changed en route (see Section 6). This leads to the following formal rule:

- Forwarding nodes such as routers and load distributors MUST NOT depend only on Flow Label values being uniformly distributed. In any usage such as a hash key for load distribution, the Flow Label bits MUST be combined at least with bits from other sources within the packet, so as to produce a constant hash value for each flow and a suitable distribution of hash values across flows. Typically the other fields used will be some or all components of the usual 5-tuple. In this way, load distribution will still occur even if the Flow Label values are poorly distributed.

Although uniformly distributed flow label values are recommended below, and will always be helpful for load distribution, it is unsafe to assume their presence in the general case, and the use case needs to work even if the flow label value is zero.

As a general practice, packet flows should not be reordered, and the use of the Flow Label field does not affect this. In particular, a Flow label value of zero does not imply that reordering is acceptable.

3. Flow Labeling Requirements in the Stateless Scenario

This section defines the minimum requirements for methods of setting the flow label value within the stateless scenario of flow label usage.

To enable Flow Label based classification, source nodes SHOULD assign each unrelated transport connection and application data stream to a new flow. A typical definition of a flow for this purpose is any set of packets carrying the same 5-tuple {dest addr, source addr, protocol, dest port, source port}.

It is desirable that flow label values should be uniformly distributed to assist load distribution. It is therefore RECOMMENDED that source hosts support the flow label by setting the flow label field for all packets of a given flow to the same value chosen from

an approximation to a discrete uniform distribution. Both stateful and stateless methods of assigning a value could be used, but it is outside the scope of this specification to mandate an algorithm. The algorithm SHOULD ensure that the resulting flow label values are unique with high probability. However, if two simultaneous flows are by chance assigned the same flow label value, and have the same source and destination addresses, it simply means that they will receive the same treatment throughout the network. As long as this is a low probability event, it will not significantly affect load distribution.

A possible stateless algorithm is to use a suitable 20 bit hash of values from the IP packet's 5-tuple. A simple hash function is described in Appendix A.

An alternative approach is to to use a pseudo-random number generator to assign a flow label value for a given transport session; such a method will require minimal local state to be kept at the source node, by recording the flow label associated with each transport socket.

Viewed externally, either of these approaches will produce values that appear to be uniformly distributed and pseudo-random.

An implementation in which flow labels are assigned sequentially is NOT RECOMMENDED, as it would then be simple for on-path observers to guess the next value.

A source node which does not otherwise set the flow label MUST set its value to zero.

A node that forwards a flow whose flow label value in arriving packets is zero MAY change the flow label value. In that case, it is RECOMMENDED that the forwarding node sets the flow label field for a flow to a uniformly distributed value as just described for source nodes.

o The same considerations apply as to source hosts setting the flow label; in particular, the normal case is that a flow is defined by the 5-tuple.

o This option, if implemented, would presumably be used by first-hop or ingress routers. It might place a considerable per-packet processing load on them, even if they adopted a stateless method of flow identification and label assignment. This is why the principal recommendation is that the source host should set the label.

The preceding rules taken together allow a given network to include routers that set flow labels on behalf of hosts that do not do so.

They also recommend that flow labels exported to the Internet are always either zero or uniformly distributed.

4. Flow State Establishment Requirements

A node that sets the flow label MAY also take part in a flow state establishment method that results in assigning specific treatments to specific flows, possibly including signaling. Any such method MUST NOT disturb nodes taking part in the stateless scenario just described. Thus, any node that sets flow label values according to a stateful scheme MUST choose labels that conform to Section 3 of the present specification. Further details are not discussed in this document.

5. Essential correction to RFC 2205

[RFC2460] reduced the size of the flow label field from 24 to 20 bits. The references to a 24 bit flow label field on pages 87 and 88 of [RFC2205] are updated accordingly.

6. Security Considerations

This section considers security issues raised by the use of the Flow Label, including the potential for denial-of-service attacks, and the related potential for theft of service by unauthorized traffic (Section 6.2). Section 6.3 addresses the use of the Flow Label in the presence of IPsec including its interaction with IPsec tunnel mode and other tunneling protocols. We also note that inspection of unencrypted Flow Labels may allow some forms of traffic analysis by revealing some structure of the underlying communications. Even if the flow label were encrypted, its presence as a constant value in a fixed position might assist traffic analysis and cryptoanalysis.

The flow label is not protected in any way, even if IPsec authentication [RFC4302] is in use, so it can be forged by an on-path attacker. Implementers are advised that any en-route change to the flow label value is undetectable. On the other hand, a uniformly distributed pseudo-random flow label cannot be readily guessed by an attacker; see [I-D.gont-6man-flowlabel-security] for further discussion.

6.1. Covert Channel Risk

The flow label could be used as a covert data channel, since apparently pseudo-random flow label values could in fact consist of

covert data. This could for example be achieved using a series of otherwise innocuous UDP packets whose flow label values constitute a covert message, or by co-opting a TCP session to carry a covert message in the flow labels of successive packets. Both of these could be recognised as suspicious - the first because isolated UDP packets would not normally be expected to have non-zero flow labels, and the second because the flow label values in a given TCP session should all be equal. However, other methods, such as co-opting the flow labels of occasional packets, might be rather hard to detect.

In situations where the covert channel risk is considered significant, the only certain defense is for a firewall to rewrite non-zero flow labels in a stateless manner, like a first-hop router (see Section 3). This would be an exceptional violation of the rule that the flow label, once set to a non-zero value, must not be changed. To preserve load distribution capability, such a firewall MUST NOT set non-zero flow labels to zero.

6.2. Theft and Denial of Service

Since the mapping of network traffic to flow-specific treatment is triggered by the IP addresses and Flow Label value of the IPv6 header, an adversary may be able to obtain unintended service by modifying the IPv6 header or by injecting packets with false addresses and/or labels. Theft of service is not further discussed in this document, since it can only be analysed for specific stateful methods of using the flow label. However, a denial of service attack becomes possible in the stateless model when the modified or injected traffic depletes the resources available to forward it and other traffic streams. If a DoS attack were undertaken against a given Flow Label (or set of Flow Labels), then traffic containing an affected Flow Label might well experience worse-than-best-effort network performance.

Note that since the treatment of IP headers by nodes is typically unverified, there is no guarantee that flow labels sent by a node are set according to the recommendations in this document. A man-in-the-middle or injected-traffic denial of service attack specifically directed at flow label handling would involve setting unusual flow labels. For example, an attacker could set all flow labels reaching a given router to the same arbitrary non-zero value, or could perform rapid cycling of flow label values such that the packets of a given flow will each have a different value. Either of these attacks would cause a stateless load distribution algorithm to perform badly and would cause a stateful classifier to behave incorrectly. For this reason, stateless classifiers should not use the flow label alone to control load distribution, and stateful classifiers should include explicit methods to detect and ignore suspect flow label values.

Since flows are identified by the 3-tuple of the Flow Label and the Source and Destination Address, the risk of denial of service introduced by the Flow Label is closely related to the risk of denial of service by address spoofing. An adversary who is in a position to forge an address is also likely to be able to forge a label, and vice versa.

There are two issues with different properties: Spoofing of the Flow Label only, and spoofing of the whole 3-tuple, including Source and Destination Address.

The former can be done inside a node which is using or transmitting the correct source address. The ability to spoof a Flow Label typically implies being in a position to also forge an address, but in many cases, spoofing an address may not be interesting to the spoofer, especially if the spoofer's goal is theft of service, rather than denial of service.

The latter can be done by a host which is not subject to ingress filtering [RFC2827] or by an intermediate router. Due to its properties, this is typically useful only for denial of service. In the absence of ingress filtering, almost any third party could instigate such an attack.

In the presence of ingress filtering, forging a non-zero Flow Label on packets that originated with a zero label, or modifying or clearing a label, could only occur if an intermediate system such as a router was compromised, or through some other form of man-in-the-middle attack.

6.3. IPsec and Tunneling Interactions

The IPsec protocol, as defined in [RFC4301], [RFC4302], [RFC4303] does not include the IPv6 header's Flow Label in any of its cryptographic calculations (in the case of tunnel mode, it is the outer IPv6 header's Flow Label that is not included). Hence modification of the Flow Label by a network node has no effect on IPsec end-to-end security, because it cannot cause any IPsec integrity check to fail. As a consequence, IPsec does not provide any defense against an adversary's modification of the Flow Label (i.e., a man-in-the-middle attack).

IPsec tunnel mode provides security for the encapsulated IP header's Flow Label. A tunnel mode IPsec packet contains two IP headers: an outer header supplied by the tunnel ingress node and an encapsulated inner header supplied by the original source of the packet. When an IPsec tunnel is passing through nodes performing flow classification, the intermediate network nodes operate on the Flow Label in the outer

header. At the tunnel egress node, IPsec processing includes removing the outer header and forwarding the packet (if required) using the inner header. The IPsec protocol requires that the inner header's Flow Label not be changed by this decapsulation processing to ensure that modifications to label cannot be used to launch theft- or denial-of-service attacks across an IPsec tunnel endpoint. This document makes no change to that requirement; indeed it forbids changes to the Flow Label.

When IPsec tunnel egress decapsulation processing includes a sufficiently strong cryptographic integrity check of the encapsulated packet (where sufficiency is determined by local security policy), the tunnel egress node can safely assume that the Flow Label in the inner header has the same value as it had at the tunnel ingress node.

This analysis and its implications apply to any tunneling protocol that performs integrity checks. Of course, any Flow Label set in an encapsulating IPv6 header is subject to the risks described in the previous section.

6.4. Security Filtering Interactions

The Flow Label does nothing to eliminate the need for packet filtering based on headers past the IP header, if such filtering is deemed necessary for security reasons on nodes such as firewalls or filtering routers.

7. Differences from RFC 3697

The main differences between this specification and its predecessor are as follows:

- This specification encourages non-zero flow label values to be used, and clearly defines how to set a non-zero value.
- It encourages a stateless model with uniformly distributed flow label values.
- It does not specify any details of a stateful model.
- It retains the rule that the flow label must not be changed en route, but allows routers to set the label on behalf of hosts that do not do so.
- It discusses the covert channel risk and its consequences for firewalls.

For further details see [I-D.ietf-6man-flow-update].

8. IANA Considerations

This document requests no action by IANA.

9. Acknowledgements

Valuable comments and contributions were made by Ran Atkinson, Fred Baker, Steve Blake, Remi Despres, Alan Ford, Fernando Gont, Brian Haberman, Tony Hain, Joel Halpern, Qinwen Hu, Chris Morrow, Thomas Narten, Mark Smith, Pascal Thubert, Iljitsch van Beijnum, and other participants in the 6man working group.

Cristian Calude suggested the von Neumann algorithm in Appendix A.

Steve Deering and Alex Conta were co-authors of RFC 3697, on which this document is based.

Contributors to the original development of RFC 3697 included Ran Atkinson, Steve Blake, Jim Bound, Francis Dupont, Robert Elz, Tony Hain, Robert Hancock, Bob Hinden, Christian Huitema, Frank Kastenholz, Thomas Narten, Charles Perkins, Pekka Savola, Hesham Soliman, Michael Thomas, Margaret Wasserman, and Alex Zinin.

This document was produced using the xml2rfc tool [RFC2629].

10. Change log [RFC Editor: Please remove]

draft-ietf-6man-flow-3697bis-04: update to resolve further WG comments, 2011-05-11:

- Suggested a specific hash algorithm to generate a flow label.
- Removed reference to stateful domain.
- Added text about covert channel and tuned text about firewall behavior; removed the confusing word "immutable".
- Added that Section 6 of RFC 2460 is replaced.
- Editorial fixes.

draft-ietf-6man-flow-3697bis-03: update to resolve WGLC comments, 2011-05-02:

- Clarified that the network layer view of flows is agnostic about transport sessions.
- Honed the definition of stateless v stateful models.
- Honed the text about using a pseudo-random function.
- Moved material about violation of immutability to Security section, and rephrased accordingly.

- Dropped material about setting the flow label at a domain exit router: doesn't belong here now that we have dropped almost all the stateful text.
- Removed normative reference to draft-gont-6man-flowlabel-security.
- Removed the statement that a node that does not set or use the flow label must ignore it: this statement appears to be a no-op.
- Added a summary of changes from RFC 3697.
- Miscellaneous editorial fixes.

draft-ietf-6man-flow-3697bis-02: update to remove most text about stateful methods, 2011-03-13

draft-ietf-6man-flow-3697bis-01: update after resolving 11 initial issues, 2011-02-26

draft-ietf-6man-flow-3697bis-00: original version, built from RFC3697 and draft-ietf-6man-flow-update-01, 2011-01-31

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

11.2. Informative References

[I-D.gont-6man-flowlabel-security] Gont, F., "Security Assessment of the IPv6 Flow Label", draft-gont-6man-flowlabel-security-01 (work in progress), November 2010.

[I-D.ietf-6man-flow-update] Amante, S., Carpenter, B., and S. Jiang, "Rationale for update to the IPv6 flow label specification", draft-ietf-6man-flow-update-05 (work in progress), May 2011.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC3697] Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification", RFC 3697, March 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[vonNeumann] von Neumann, J., "Various techniques used in connection with random digits", National Bureau of Standards Applied Math Series 12, 36-38, 1951.

Appendix A. Simple 20-bit Hash Function

As mentioned in Section 3, a stateless hash function may be used to generate a flow label value from an IPv6 packet's 5-tuple. An example function, based on an algorithm by von Neumann known to produce an approximately uniform distribution [vonNeumann], is as follows:

1. Split the destination and source addresses into two 64 bit values each, thus transforming the 5-tuple into a 7-tuple.
2. Add the seven components together using unsigned 64 bit arithmetic, discarding any carry bits.
3. Apply the von Neumann algorithm to the resulting string of 64 bits:
    1. Starting at the least significant end, select two bits.
    2. If the two bits are 00 or 11, discard them.
    3. If the two bits are 01, output a 0 bit.
    4. If the two bits are 10, output a 1 bit.
    5. Repeat with the next two bits in the input 64 bit string.
    6. Stop when 20 bits have been output (or when the 64 bit string is exhausted).
4. In the highly unlikely event that the result is exactly zero, set the flow label arbitrarily to the value 1.

Authors' Addresses

Shane Amante
Level 3 Communications, LLC
1025 Eldorado Blvd
Broomfield, CO 80021
USA

Email: shane@level3.net

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

Email: brian.e.carpenter@gmail.com

Sheng Jiang
Huawei Technologies Co., Ltd
Huawei Building, No.3 Xinxi Rd.,
Shang-Di Information Industry Base, Hai-Dian District, Beijing
P.R. China

Email: jiangsheng@huawei.com

Jarno Rajahalme
Nokia Siemens Networks
Linnoitustie 6
02600 Espoo
Finland

Email: jarno.rajahalme@nsn.com