IPv6 Flow Label Update

Shane Amante  
Level 3 Communications, Inc.

Brian Carpenter  
University of Auckland

Sheng Jiang  
Huawei

Jarno Rajahalme  
Nokia-Siemens Networks

April 11, 2012  
2012 North American IPv6 Summit
Overview

- RFC 6294: Survey of proposed use cases for the IPv6 flow label
  - Surveys variety of QoS, label switching & other forms of information passing proposed for the IPv6 flow-label over the last several years

- RFC 6438: Flow Label for Load Balancing Tunneled Traffic over ECMP & LAG’s

- RFC 6437: Obsoletes “old” flow label RFC 3697

- RFC 6436
  - Contains background and rationale for changes in RFC 6437.

- Other load-balancing work in the IETF
Flow Label History

- Flow Label *was* still an *experimental* field
- Predecessor to MPLS label switching, when speed of (full) IP FIB lookups was in doubt
- Likely would have used stateful method (RSVP) to establish a path and set-up flow-labels used through the network
(My) Assertion

• Deep Packet Inspection (DPI) is dumb ...
• ... especially in the Core for fine-grained load-balancing over LAG and/or ECMP paths
• Must avoid brittle “architecture” for IPv6
  • Can’t create new applications, because core will not support them ...
RFC 6438: Flow Label for Load Balancing Tunneled Traffic over ECMP & LAG’s
Origin of RFC 6438

- LISP & AMT need fast forwarding of tunneled packets, but **DO NOT** want checksums – more “HW friendly”
- LISP also needed load-balancing over LAG/ECMP
- In IPv6, UDP checksum over entire packet is mandatory, because there is **NO** IPv6 packet header checksum
- UDP-lite [RFC 3828] allows partial checksum\(^1\) ... but, it’s not [widely] implemented
- Confusion in last flow-label spec [RFC 3697], theoretically didn’t allow IPv6 flow-label to be set by routers, for tunneled packets

\(^1\)At a minimum over UDP-lite + IPv6 packet pseudo-header
• Tunnel end-points, (e.g.: TEP A & TEP B), encapsulate traffic as IPv[4|6]/IPv6 and forward to R1 or R2

• R1 (& R2) can **ONLY** use outermost IP header 2-tuple, \{src_ip, dst_ip\}, as input-keys for LAG and/or ECMP hash algorithm

• **Result:** All tunnel traffic from TEP A ⟷ TEP B is placed on a single (bottom) link, at R1 (& R2), resulting in out-of-balance LAG or ECMP bundle
RFC 6438

Problem Desc. (2/2)

- R1 & R2 only use `{src_ip, dst_ip}` as input-keys for LAG/ECMP hash algorithm

---

**IPv6 Header**

<table>
<thead>
<tr>
<th>Vers</th>
<th>Traffic Class</th>
<th>Flow Label = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>17, (UDP)</td>
</tr>
</tbody>
</table>

**Source Address**

**Destination Address**

**Payload Length**

**Hop Limit**

---

**UDP**

<table>
<thead>
<tr>
<th>Vers</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>41, (IPv6)</td>
</tr>
</tbody>
</table>

**Inner IPv6 Header**

<table>
<thead>
<tr>
<th>Vers</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>17, (UDP)</td>
</tr>
</tbody>
</table>

**Source Port**

**Destination Port**

**Length**

**Checksum**

---

**Outer IPv6 Header**

<table>
<thead>
<tr>
<th>Vers</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Source Port**

**Destination Port**

**Length**

**Checksum**

---

**TEP A**

**Input Interface**

<table>
<thead>
<tr>
<th>Vers</th>
<th>Traffic Class</th>
<th>Flow Label = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

**Source Address**

**Destination Address**

**Payload Length**

**Hop Limit**

---

**UDP**
• Tunnel end-points, (e.g.: TEP A & TEP B), encapsulate traffic as IPv[4|6]/IPv6

• During encapsulation phase, TEP’s use the 5-tuple of the incoming IPvN packet to create a stateless IPv6 flow-label that is placed in outermost IPv6 header

• **Result:** All tunnel traffic from TEP A ➞ TEP B should be well balanced across the LAG or ECMP bundle between R1 & R2
Tunnel end-points use the 5-tuple of incoming IPvN packet to create a stateless IPv6 flow-label that is placed in outermost IPv6 header.
RFC 6438
Solution (3/3)

- Intermediate Routers/Switches (R1, R2) use outer IPv6 header 3-tuple \{src_ip, dst_ip + flow_label\} as input-keys for LAG/ECMP hash algorithm – result should be more even load-balancing on LAG/ECMP’s R1, R2
RFC 6438 Summary

- TEP’s act as “hosts” encoding a stateless IPv6 flow-label to be used by intermediate switch/routers for stateless LAG/ECMP load-balancing
- Load-balancing of non-tunneled (native) IPv6 packets specified in RFC 6437
  - SHOULD still use IPv6 header 5 or 6-tuple
- RFC 6438 backwards compatible with RFC 3697
- RFC 6438 was largely non-controversial change
Origins of RFC 6437

• RFC 3697 was considered very confusing, thus not implemented on hosts

• Strict immutability of flow-label was impractical for a variety of reasons

• Unclear if flow-label was supposed to be used (at all) as part of input-keys for LAG/ECMP calculations
RFC 6437 Goals

- Recognize the original, *stateful* use of IPv6 flow-label never came to fruition
- Clarify its use, once-and-for-all, given the plethora of proposals\(^1\) that have attempted to claim it over the years – the last 20-bits in the IPv6 header!
- (Slightly) relax strict immutability to support ‘incremental deployment’ at routers, etc.
- Promote use of IPv6 flow-label that would increase longevity, (long-term flexibility), of IPv6

\(^1\)RFC 6294
1) Flow-labels ARE NOT immutable, because they are not protected by either an IPv6 pseudo-header checksum or IPSec AH

2) All packets belonging to the same “flow” MUST have the same flow-label value

a) flow = \{ src_ip, dst_ip, protocol, src_port, dst_port \}
3) **Source hosts** SHOULD set a unique, “uniformly distributed” flow-label value\(^1\) to each unrelated transport connection

4) Only if flow-label = 0, a router MAY set a (non-unique, stateless) uniformly distributed flow-label value\(^2\)

a) Typically, (only) a 1st-hop router would set the flow-label to promote incremental deployment, (until host Operating Systems catch up).

---

\(^1\) No algorithm is specified; however, one example is provided in Appendix A.

\(^2\) Would only apply to flows containing whole (non-fragmented) packets.
5) Once set to a non-zero value, flow label values should not be changed, except:

a) Middleboxes (e.g.: firewalls) MAY change the flow-label value, but it is RECOMMENDED that they also use a new uniformly distributed value, just like source hosts

b) Allows for the case where security admins want to prevent the flow-label from being used as (another) covert channel in the IPv6 header
6) Routers MUST NOT depend solely on flow-label for an input-key to LAG/ECMP hash algorithm

a) Routers MUST combine the flow-label with other IP header fields as input-keys for LAG/ECMP hash calculations, e.g.:

- (Long-term) Minimum input-keys = \{src_ip, dst_ip, flow_label\}; or,

- (Short-term) Maximum input-keys = \{src_ip, dst_ip, flow_label, protocol, src_port, dst_port\}
RFC 6437 Summary

- Eventually, core routers/switches could just use 3-tuple of \{src_ip, dst_ip + flow-label\}, at fixed offsets in IPv6 header, as input-keys for LAG/ECMP load distribution.

- Future Transport-layer protocols could be developed without the need to adapt intermediate routers or switches to perform DPI to find adequate input-keys for LAG/ECMP load balancing.
Other IETF work to improve load-balancing over LAG/ECMP
Other (MPLS) Load-Balancing Drafts

- RFC 6391: Flow Aware Transport PW’s (FAT PW’s)
  - Fine-grained load-balancing of p2p PW’s [RFC 4447] over MPLS

- draft-ietf-mpls-entropy-label-01
  - Adds support for MPLS tunnel protocols (RSVP, LDP, BGP), ideally without regard to the applications riding on top
  - Goal is to support IPVPN, VPLS, 6PE, etc.
Summary
Summary

• Finally, a real use for the IPv6 flow-label!
• Ask your HW & SW vendors for support
• Tell your Security folks to NOT set/reset the flow-label at middleboxes