Using the IPv6 Flow Label for Server Load Balancing

draft-carpenter-v6ops-label-balance-01

Abstract

This document describes how the IPv6 flow label can be used in support of layer 3/4 load distribution and balancing for large server farms.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on July 20, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1. Introduction ........................................... 3
2. Role of the Flow Label ................................. 5
3. Possible extended role .................................. 7
4. Security Considerations ................................. 9
5. IANA Considerations .................................... 10
6. Acknowledgements ....................................... 10
7. Change log [RFC Editor: Please remove] ............... 10
8. References .............................................. 10
   8.1. Normative References ............................... 10
   8.2. Informative References ............................. 11
Authors’ Addresses .......................................... 11
1. Introduction

The IPv6 flow label has been redefined [RFC6437] and its use for load sharing in multipath routing has been specified [RFC6438]. Another scenario in which the flow label could be used is in load distribution for large server farms. Load distribution is a slightly more general term than load balancing, but the latter is more commonly used. This document starts with a brief introduction to load balancing techniques and then describes how the flow label might be used to enhance layer 3/4 flow balancers in particular.

Load balancing for server farms is achieved by a variety of methods, often used in combination [Tarreau]. The flow label is not relevant to all of them. The actual load balancing algorithm (the choice of server for a new client session) is irrelevant to this discussion.

- The simplest method is simply using the DNS to return different server addresses for a single name such as www.example.com to different users. Typically this is done by rotating the order in which different addresses are listed by the relevant authoritative DNS server, assuming that the client will pick the first one. Routing may be configured such that the different addresses are handled by different ingress routers. The flow label can have no impact on this method and it is not discussed further.

- Another method, for HTTP servers, is to operate a layer 7 reverse proxy in front of the server farm. The reverse proxy will present a single IP address to the world, communicated to clients by a single AAAA record. For each new client session (an incoming TCP connection and HTTP request), it will pick a particular server and proxy the session to it. Hopefully the act of proxying will be cheap compared to the act of serving the required content. The proxy must retain TCP state and proxy state for the duration of the session. This TCP state could, potentially, include the incoming flow label value.

- A component of some load balancing systems is an SSL reverse proxy farm. The individual SSL proxies handle all cryptographic aspects and exchange raw HTTP with the actual servers. Thus, from the load balancing point of view, this really looks just like a server farm, except that it’s specialised for HTTPS. Each proxy will retain SSL and TCP and maybe HTTP state for the duration of the session, and the TCP state could potentially include the flow label.

- Finally the "front end" of many load balancing systems is a layer 3/4 load balancer. In this case, it is the layer 3/4 load balancer whose IP address is published as the primary AAAA record for the service. All client sessions will pass through this device. According to the precise scenario, it will spread new sessions across the actual application servers, across an SSL
proxy farm, or across a set of layer 7 proxies. In all cases, the layer 3/4 load balancer has to recognize incoming packets as belonging to new or existing client sessions, and choose the target server or proxy so as to ensure persistence. ‘Persistence’ is defined as guaranteeing that a given session will run to completion on a single server. The layer 3/4 load balancer therefore needs to inspect each incoming packet to identify the session. Typically, this depends on the source address and the protocol and port numbers in the packet. Clearly, such a balancer could inspect and make use of the flow label value.

Our focus is on how the balancer identifies a particular flow. For clarity, note that two aspects of layer 3/4 load balancers are not affected at all by use of the flow label to identify sessions.

1. Balancers use various techniques to redirect traffic to a specific target server.
   - All servers are configured with the same IP address, they are all on the same LAN, and the load balancer sends directly to their individual MAC addresses.
   - Each server has its own IP address, and the balancer uses an IP-in-IP tunnel to reach it.
   - Each server has its own IP address, and the balancer performs NAPT (network address and port translation) to deliver the client’s packets to that address.

   The choice between these methods is not affected by use of the flow label.

2. A layer 3/4 balancer must correctly handle Path MTU Discovery by forwarding relevant ICMPv6 packets in both directions. This too is not affected by use of the flow label.

   The following diagram, inspired by [Tarreau], shows a maximum layout.
From the previous paragraphs, we can identify several points in this diagram where the flow label might be relevant:

1. Layer 3/4 load balancers.
2. SSL proxies.
3. HTTP proxies.

2. Role of the Flow Label

The IPv6 flow label is a 20 bit field included in every IPv6 header [RFC2460] and it is defined in [RFC6437]. According to this definition, it should be set to a constant value for a given traffic flow (such as an HTTP connection), but until the standard is widely implemented it will often be set to the default value of zero. Any device that has access to the IPv6 header has access to the flow.
label, and it is at a fixed position in every IPv6 packet. In contrast, transport layer information, such as the port numbers, is not always in a fixed position, since it follows any IPv6 extension headers that may be present. Therefore, within the lifetime of a given transport layer connection, the flow label can be a more convenient "handle" than the port number for identifying that particular connection.

According to [RFC6437], source hosts should set the flow label, but if they do not (i.e. its value is zero), forwarding nodes may do so instead. In both cases, the flow label value must be constant for a given transport session, normally identified by the IPv6 and Transport header 5-tuple. The flow label should be calculated by a stateless algorithm. The value should form part of a statistically uniform distribution, making it suitable as part of a hash function used for load distribution. Because of using a stateless algorithm to calculate the label, there is a very low (but non-zero) probability that two simultaneous flows from the same source to the same destination have the same flow label value despite having different transport protocol port numbers.

A careful reading of RFC 6437 shows that for a given source accessing a well-known TCP port at a given destination, the flow label is in effect a proxy for the source port number, found at a fixed position in the layer 3 header. Thus, the suggested model for using the flow label in a load balancing mechanism is as follows:

- It is clearly better if the original source, e.g. an HTTP client, sets the flow label. However, if the flow label of an incoming packet is zero, there are two possibilities:
  1. The ingress router at the server site could implement the stateless mechanism in Section 3 of [RFC6437] to set the flow label value to an appropriate value. This relieves the subsequent load balancers of the need to fully analyse the IPv6 and Transport header 5-tuple to identify the packets belonging to the same flow.
  2. Load balancers will use the flow label value as described below if it is set, but use the transport header in the traditional way otherwise.

- In either case, the idea is that as the use of the flow label becomes more prevalent, load balancers will reap a growing performance benefit.

- The layer 3/4 load balancers can use the 2-tuple {source address, flow label} as the session key for whatever load distribution algorithm they support, instead of searching for the transport port number later in the header. Note that they do not need to consider the destination address as it is always the same, i.e., the server address.

Such a load balancer may nevertheless need to inspect the payload of the first packet of a new session, in order to apply its load distribution algorithm. However, for all subsequent packets of the session, it can ignore all IPv6 extension headers, which should lead to a performance benefit. Whether this benefit is valuable will depend on engineering details of the specific load balancer.

Layer 3/4 balancers that redirect the incoming packets by NAPT cannot obtain any saving of time by using the flow label, because they must in any case follow the extension header chain in order to locate and modify the port number and transport checksum. The same would apply to balancers that perform TCP state tracking for any reason.

- Note that correct handling of ICMPv6 for Path MTU Discovery requires the layer 3/4 balancer to keep state for the client source address, independently of either the port numbers or the flow label.
- The SSL proxies may also use the 2-tuple as a session key. However, since they have to process the transport layer in any case, this might not lead to any performance benefit.

An SSL proxy should forward the flow identifier between the ciphered side and the clear side. Being able to forward data used for persistence is very important, as it’s the only way to stack multiple layers of network components without losing information.

- The HTTP proxies may do the same. However, since they have to process the transport and application layers in any case, this might not lead to any performance benefit.

Note that in the unlikely event of two simultaneous flows from the same source having the same flow label value, the two flows would end up assigned to the same server, where they would be distinguished as normal by their port numbers. Since this would be a statistically rare event, it would not damage the overall load balancing effect. In the case where many thousands of clients are hidden behind the same large-scale NAT with a single IP address, this might be an incorrect assumption, but this is not expected to be a factor for IPv6, since there is no valid reason to implement NAT [RFC4864]. The statistical assumption is valid for sites that implement network prefix translation [RFC6296], since this technique provides a different address for each client.

3. Possible extended role

A particular aspect of the session persistence issue is when multiple independent transport connections from the same client need to be
handled by the same server instance. This can be an extremely difficult task which often requires ugly tricks such as pattern matching within a buffered stream, cookie insertion, etc. If the client application has control over the outgoing flow label, then it can itself assign the same label to all transport connections related to a single application session.

A common example is FTP. For a load balancer, passive-mode FTP requires parsing the entire control stream (port 21), in order to find which incoming packet will initiate a data session on a port chosen by the server. This does not always work well due to the fact that sometimes clients don’t connect, or that the session is finally not used (e.g., because no transfer needs to be performed).

Using a flow label, the client could generate an initial unpredictable flow identifier when a file transfer is expected, and assign the same flow label to all data connections related to the same control connection. A flow label based load balancer would then by definition send the data traffic to the same server as the control traffic, and would thus guarantee that the sessions are properly associated. Such a mechanism is permitted by [RFC6437], although it is not the recommended default.

The same need is even more prominent with HTTP/HTTPS: while it is not difficult to insert a cookie in an HTTP stream to identify the server the user was assigned to, it is very difficult to do that for HTTPS, because the stream must be deciphered first. Deciphering the stream requires a huge amount of centralized power, since the load balancer needs to see the clear stream; this is in fact the main reason for SSL proxies in load balancing scenarios. If a web client (browser) used the same flow label for any protocol targeting a given host (or domain), this could be used by load balancers to reach the same server for both HTTP and HTTPS, without having to open the stream payload at all nor to inspect anything beyond layer 3, which clearly is not possible today.

An additional complication that can arise is when a single client inadvertently generates sessions that appear to originate from different IP addresses. This can arise, for example, if an enterprise uses a proxy farm for outgoing traffic. When two consecutive client requests pass through two distinct proxies, a different IP source address may be presented to the server load balancer, which then cannot rely on address-based persistence. It would be possible in principle to use the same flow label value for correlated sessions from the same client, if the proxies were transparent to the flow label value.

In some application scenarios, an inadvertent change in the client IP
address may have only minor consequences, such as reloading transaction context into a new server. In other cases it may be more serious and result in a transaction failure. For this reason, a reliable solution in which the flow label value on its own acts as a load balancer session handle would be advantageous.

Using the flow label in this way would also greatly simplify the logging of user sessions. A very common task is to match logs from various equipments to follow a user’s activity and decide whether it indicates a bug, user error or attack. Logging a flow label would of course help because it’s easier to find the beginning and end of a session and decide whether it’s legitimate or not.

Such extensions to the role of the flow label in load balancing are theoretically very attractive, but would require a major refresh of client software as well as of load balancers themselves. It amounts to considering an entire application session, in a broad sense, as a single flow for the purposes of RFC 6437.

4. Security Considerations

Security aspects of the flow label are discussed in [RFC6437]. As noted there, a malicious source or man-in-the-middle could disturb load balancing by manipulating flow labels.

Specifically, [RFC6437] states that "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." The former point is answered by also using the source address. The latter point is more complex. If the risk is considered serious, the ingress router mentioned above should verify incoming flows with non-zero flow label values. If a flow from a given source address and port number does not have a constant flow label value, it is suspect and should be dropped.

The suggestion in Section 3 of using the flow label on its own as a session handle is somewhat problematic. It should only be used for applications that are otherwise protected (for example by SSL or TLS) and load balancers should be designed to detect and ignore traffic where the same flow label value is used by many different source addresses.

The flow label may be of use in protecting against distributed denial of service (DDOS) attacks against servers. As noted in RFC 6437, a source should generate flow label values that are hard to predict, most likely by including a secret nonce in the hash used to generate each label. The attacker does not know the nonce and therefore has
no way to invent flow labels which will all target the same server, even with knowledge of both the hash algorithm and the load balancing algorithm.

New flows are assigned to a server according to any of the usual algorithms available on the load balancer (e.g., least connections, round robin, etc.). The association between the flow label value and the server is stored in a table (often called stick table) so that future connections using the same flow label can be sent to the same server. This method is more robust against a loss of server and also prevents an attacker from targeting a single server, because the association between a flow label value and a server is not known externally.

5. IANA Considerations

This document requests no action by IANA.

6. Acknowledgements

Valuable comments and contributions were made by Fred Baker, Lorenzo Colitti, Joel Jaeggli, Gurudeep Kamat, Julius Volz, and others.

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

7. Change log [RFC Editor: Please remove]


8. References

8.1. Normative References


8.2. Informative References


Authors’ Addresses

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  
Q14, Huawei Campus  
No.156 Beiqing Road  
Hai-Dian District, Beijing 100095  
P.R. China  
Email: jiangsheng@huawei.com