

Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 8, 2013

B. Carpenter
Univ. of Auckland
S. Jiang
Huawei Technologies Co., Ltd
W. Tarreau
Exceliance
December 5, 2012

Extending Use of the IPv6 Flow Label for Load Balancing Persistence
draft-tarreau-extend-flow-label-balancing-01

Abstract

This document describes how the IPv6 flow label could be used in an extended role to simplify persistence mechanisms for 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 June 8, 2013.

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

Internet-Draft Flow Label for Load Balancing Persistence December 2012

described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
2.	Server Load Balancing and the Persistence Problem	3
3.	Extended Role of Flow Label	4
4.	Security Considerations	6
5.	IANA Considerations	7
6.	Acknowledgements	7
7.	Change log [RFC Editor: Please remove]	7
8.	References	7
8.1.	Normative References	7
8.2.	Informative References	8
	Authors' Addresses	8

Internet-Draft Flow Label for Load Balancing Persistence December 2012

1. Introduction

The IPv6 flow label has been redefined [[RFC6437](#)] and is now a recommended IPv6 node requirement [[RFC6434](#)]. Its use for layer 3/4 server load balancing is described in [[I-D.carpenter-flow-label-balancing](#)], and the reader is assumed to be familiar with that document. In server load balancing, 'persistence' is defined as guaranteeing that a given session will run to completion on a single server. The present document describes extensions to the role of the flow label to simplify and enhance persistence mechanisms.

Note in draft: The authors recognize that this proposal is incomplete, that it needs considerable thought and discussion, and that other approaches might be possible. However, current approaches to persistence in server load balancing are complicated and pragmatic, and this new approach, even though it requires changes to client applications and proxies as well as to load balancers themselves, seems worth discussion.

2. Server Load Balancing and the Persistence Problem

The IPv6 flow label is a 20 bit field included in every IPv6 header [[RFC2460](#)]. It is recommended to be supported in all IPv6 nodes by [[RFC6434](#)] and it is defined in [[RFC6437](#)]. In [[I-D.carpenter-flow-label-balancing](#)], it is explained how the flow label value, set at or near the client of a server farm, may be used by a layer 3/4 load balancer as part of a 2-tuple {source address, flow label} to efficiently identify packets belonging to a given client's application data flow and to direct them to a specific server.

A layer 3/4 load balancer has to recognize incoming packets as belonging to new or existing client sessions, and choose a target server or proxy so as to ensure persistence. In a simple scenario,

the 2-tuple {source address, flow label} will be a sufficient label for a user data flow to guarantee persistence. However, there are various cases where this does not apply. Sometimes, 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, which most current load balancers have to deal with every day.

A common example is FTP. For a load balancer, passive mode FTP requires parsing the entire control stream on port 21, in order to find which incoming packet will initiate a data session on a port

chosen by the server. This expensive process is not always useful, due to the fact that sometimes clients fail to connect, or that the session is finally not used, e.g., because no transfer needs to be performed.

The same issue is even more prominent with HTTP/HTTPS. While it is costly but straightforward to insert a cookie in an HTTP stream to identify the server to which the user was assigned, 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 having specialised SSL proxies in load balancing scenarios.

An additional complication that arises frequently 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, or in mobile applications where several subsequent requests come from different network cells and thus different IP addresses (for instance, consulting a bank account in the train). 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.

In some application scenarios, such an inadvertent change in the client IP address may only have consequences for performance, such as reloading transaction context into a new server. In other cases it may be more serious and result in a transaction failure that seems

inexplicable to the user. A reliable and efficient solution to this problem is therefore needed.

3. Extended Role of Flow Label

We propose a new model in which the client application has control over the outgoing flow label, and assigns the same label to all transport connections related to a single application session. It would then be both possible and desirable to use the same flow label value for multiple correlated transport sessions from the same client. For this to work, it is also necessary for any proxies to be transparent to the flow label value. Thus, modifications to the network stack and the application layer code are needed in both hosts and proxies. In particular, the network API needs to provide a method for the application layer to set the flow label value for each new outgoing transport session, and to read it for each new incoming transport session. The former is needed for client applications, and both features are needed for HTTP proxies in particular.

Such a mechanism is not the recommended default host behaviour, but it is permitted by [[RFC6437](#)], which states that "a flow is not necessarily 1:1 mapped to a transport connection". The assignment of flow labels in this case is clearly no longer stateless, but the requirement that they be drawn from a uniform distribution should be retained.

In the case of FTP Passive mode, using a flow label, the client could generate an initial flow label value 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.

For a multiprotocol transaction, if a web client (browser) used the same flow label for any protocol targetting a given host (or domain), this could be used by load balancers to reach the same server for several protocols (such as HTTP and HTTPS), without having to inspect the stream payload at all nor to inspect anything beyond layer 3, which is unavoidable today.

In the case of an inadvertent change in the client IP address, most likely due to an intervening proxy, the use of the same flow label for an entire application session would allow a load balancer to reliably route all packets for the session to the same server without any additional packet inspection or cookie insertion. This would improve both efficiency and reliability for all parties concerned.

This proposal means that the layer 3/4 load balancer identifies a session by using the flow label value alone, rather than the 2-tuple {source address, flow label} as described in [\[I-D.carpenter-flow-label-balancing\]](#). However, it is of minor importance if two independent client sessions are directed to the same server because they happen to have the same flow label. They will in any case be treated separately by the server, and statistically there will be a negligible effect on load balancing, since there are a million different possible flow labels to spread traffic across the server farm.

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. In the case of two simultaneous application sessions using the same flow label value by chance, the two logs would be intermingled, so the analysis would

be more complex, but still quite feasible.

Such extensions to the role of the flow label in load balancing are theoretically very attractive, but would require a major refresh of client and proxy 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](#).

It is worth noting that what is important to save server-side resources is wide enough adoption. Most of today's load balanced traffic is HTTP originating from a handful of browsers which are regularly upgraded for security reasons. Thus, once a mechanism is adopted, it can quickly be deployed across popular browsers, and soon become the general case.

The difficulty of the upgrade path is then on the server side. The first step would consist in having layer 7 load balancers be able to consider the flow label, to avoid costly layer 7 analysis, each time it is possible. This means that if a non-null flow label is seen, then the load balancer would consider it, otherwise it would fall back to its default behaviour. The second step would consist in having front layer 3/4 load balancers bypass the layer 7 load balancer farms when the flow label is found. This point would greatly offload layer 7 load balancers.

Finally we observe that both clients and server farms would benefit from this approach, in terms of performance and reliability. Thus, although both parties (and proxy operators) would need to upgrade software, it is in their own interest to do so.

4. Security Considerations

The security considerations of [[RFC6437](#)] and [[I-D.carpenter-flow-label-balancing](#)] apply.

Using the flow label on its own as a session handle has limitations. It has no security properties and must not be used in any way as an identifier or authenticator; it does not have enough bits to be used as a nonce. Its value should never be used in the application layer nor where any form of resource sharing is not desired. For instance, it is not acceptable that an application would identify a user session by its flow label value, due to the inevitable collisions and the risk of forgery. The flow label value on its own should only be used where resource sharing is intended (for instance, load balancing) and by components explicitly designed for this task, taking into account all the risks exposed in the above security references, with solid protection against mis-use, and acceptable

fallbacks for remaining situations where the flow label values are unusable.

The setting of the flow label by an application is necessarily a stateful process, so that the application can store the label value for re-use in all transport sessions that are part of the same application transaction. Therefore, any firewall that chooses to rewrite the label, to avoid a perceived covert channel risk, must do

so in a stateful way such that a given incoming label value is always rewritten to the same outgoing value.

5. IANA Considerations

This document requests no action by IANA.

6. Acknowledgements

Valuable comments and contributions were made by...

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

7. Change log [RFC Editor: Please remove]

[draft-tarreau-extend-flow-label-balancing-01](#): small updates, 2012-12-05.

[draft-tarreau-extend-flow-label-balancing-00](#): extended role extracted after IETF83, 2012-06-12.

[draft-carpenter-v6ops-label-balance-02](#): clarified after WG discussions, 2012-03-06.

[draft-carpenter-v6ops-label-balance-01](#): updated with community comments, additional author, 2012-01-17.

[draft-carpenter-v6ops-label-balance-00](#): original version, 2011-10-13.

8. References

8.1. Normative References

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

Requirements", [RFC 6434](#), December 2011.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", [RFC 6437](#), November 2011.

8.2. Informative References

[I-D.carpenter-flow-label-balancing]
Carpenter, B., Jiang, S., and W. Tarreau, "Using the IPv6 Flow Label for Server Load Balancing", [draft-carpenter-flow-label-balancing-01](#) (work in progress), June 2012.

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

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

Willy Tarreau
Exceliance
R&D Produits reseau
3 rue du petit Robinson
78350 Jouy-en-Josas
France

Email: w@lwt.eu

