

INTAREA
Internet-Draft
Updates: 6296 (if approved)
Intended status: Experimental
Expires: October 15, 2012

R. Bonica
Juniper Networks
F. Baker
Cisco Systems
M. Wasserman
Painless Security
G. Miller
Verizon
W. Kumari
Google, Inc.
April 13, 2012

Multihoming with IPv6-to-IPv6 Network Prefix Translation (NPTv6)
draft-bonica-v6-multihome-03

Abstract

RFC 6296 introduces IPv6-to-IPv6 Network Prefix Translation (NPTv6). By deploying NPTv6, a site can achieve addressing independence without contributing to excessive routing table growth. Section 2.4 of RFC 6296 proposes an NPTv6 architecture for sites that are homed to multiple upstream providers. By deploying the proposed architecture, a multihomed site can achieve access redundancy and load balancing, in addition to addressing independence.

This memo proposes an alternative NPTv6 architecture for hosts that are homed to multiple upstream providers. The architecture described herein provides transport-layer survivability, in addition to the benefits mentioned above. In order to provide transport-layer survivability, the architecture described herein requires a small amount of additional configuration.

This memo updates RFC 6296.

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 October 15, 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	4
1.1. Terminology	5
2. NPTv6 Deployment	5
2.1. Topology	6
2.2. Addressing	7
2.2.1. Upstream Provider Addressing	7
2.2.2. Site Addressing	7
2.3. Address Translation	8
2.4. Domain Name System (DNS)	8
2.5. Routing	9
2.6. Failure Detection and Recovery	10
2.7. Load Balancing	11
3. Discussion	12
4. IANA Considerations	12
5. Security Considerations	12
6. Acknowledgments	12
7. References	13
7.1. Normative References	13
7.2. Informative References	13
Authors' Addresses	14

1. Introduction

[RFC3582] establishes the following goals for IPv6 site multihoming:

Redundancy - A site's ability to remain connected to the Internet, even when connectivity through one or more of its upstream providers fails.

Transport-Layer Survivability - A site's ability to maintain transport-layer sessions across failover and restoration events. During a failover/restoration event, the transport-layer session may detect packet loss or reordering, but neither of these cause the transport-layer session to fail.

Load Sharing - The ability of a site to distribute both inbound and outbound traffic across its upstream providers.

[RFC3582] notes that a multihoming solution may require interactions with the routing subsystem. However, multihoming solutions must be simple and scalable. They must not require excessive operational effort and must not cause excessive routing table expansion.

[RFC6296] explains how a site can achieve address independence using IPv6-to-IPv6 Network Prefix Translation (NPTv6). In order to achieve address independence, the site assigns an inside address to each of its resources (e.g., hosts). Nodes outside of the site identify those same resources using a corresponding Provider Allocated (PA) address.

The site resolves this addressing dichotomy by deploying an NPTv6 translator between itself and its upstream provider. The NPTv6 translator maintains a static, one-to-one mapping between each inside address and its corresponding PA address. That mapping persists across flows and over time.

If the site disconnects from one upstream provider and connects to another, it may lose its PA assignment. However, the site will not need to renumber its resources. It will only need to reconfigure the mapping rules on its local NPTv6 translator.

Section 2.4 of [RFC6296] describes an NPTv6 architecture for sites that are homed to multiple upstream providers. While that architecture fulfills many of the goals identified by [RFC3582], it does not achieve transport-layer survivability. Transport-layer survivability is not achieved because in this architecture, a PA address is usable only when the multi-homed site is directly connected to the allocating provider.

This memo describes an alternative architecture for multihomed sites that require transport-layer survivability. It updates Section 2.4 of [RFC6296]. In this architecture, PA addresses remain usable, even when the multihomed site loses its direct connection to the allocating provider.

The architecture described in this document can be deployed in sites that are served by two or more upstream providers. For the purpose of example, this document demonstrates how the architecture can be deployed in a site that is served by two upstream providers.

1.1. Terminology

The following terms are used in this document:

inbound packet - A packet that is destined for the multi-homed site

outbound packet - A packet that originates at the multi-homed site and is destined for a point outside of the multi-homed site

NPTv6 inside interface - An interface that connects an NPTv6 translator to the site

NPTv6 outside interface- An interface that connects an NPTv6 translator to an upstream provider

2. NPTv6 Deployment

This section demonstrates how NPTv6 can be deployed in order to achieve the goals of [RFC3582].

2.1. Topology

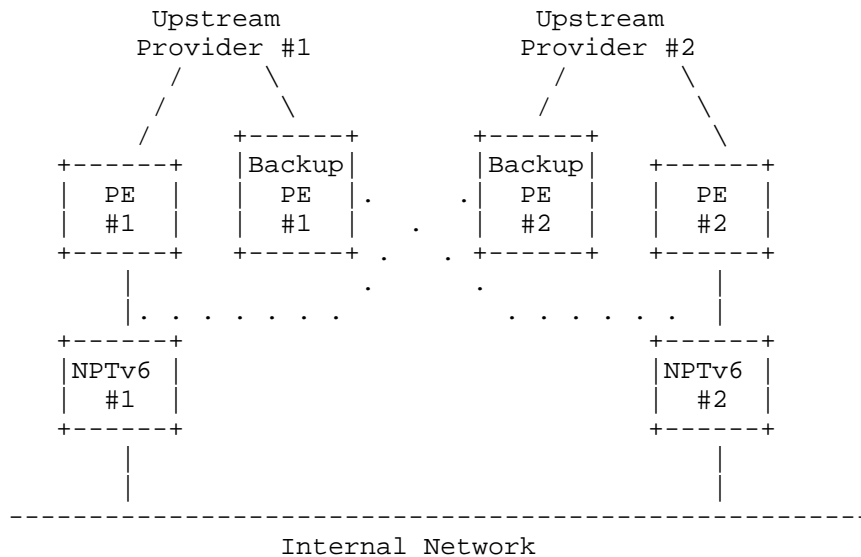


Figure 1: NPTv6 Multihomed Topology

In Figure 1, a site attaches all of its assets, including two NPTv6 translators, to an Internal Network. NPTv6 #1 is connected to Provider Edge (PE) Router #1, which is maintained by Upstream Provider #1. Likewise, NPTv6 #2 is connected to PE Router #2, which is maintained by Upstream Provider #2.

Each upstream provider also maintains a Backup PE Router. A forwarding tunnel connects the loopback interface of Backup PE Router #1 to the outside interface of NPTv6 #2. Another forwarding tunnel connects Backup PE Router #2 to NPTv6 #1. Network operators can select from many encapsulation techniques (e.g., GRE) to realize the forwarding tunnels. Tunnels are depicted as dotted lines in Figure 1.

In the figure, NPTv6 #1 and NPTv6 #2 are depicted as separate boxes. While vendors may produce a separate box to support the NPTv6 function, they may also integrate the NPTv6 function into a router.

During periods of normal operation, the Backup PE routers is very lightly loaded. Therefore, a single Backup PE router may back up multiple PE routers. Furthermore, the Backup PE router may be used for other purposes (e.g., primary PE router for another customer).

2.2. Addressing

2.2.1. Upstream Provider Addressing

A Regional Internet Registry (RIR) allocates Provider Address Block (PAB) #1 to Upstream Provider #1. From PAB #1, Upstream Provider #1 allocates two sub-blocks, using them as follows.

Upstream Provider #1 uses the first sub-block to number its internal interfaces. It also uses that sub-block to number the interfaces that connect it to its customers.

Upstream Provider #1 uses the second sub-block for customer address assignments. We refer to a particular assignment from this sub-block as a Customer Network Block (CNB). In this case, Upstream Provider #1 assigns CNB #1 to the multihomed site. For the purpose of example, assume that CNB #1 is a /59.

In a similar fashion, a Regional Internet Registry (RIR) allocates PAB #2 to Upstream Provider #2. Upstream Provider #2, in turn, assigns CNB #2 to the multihomed site. For the purpose of example, assume that CNB #2 is a /60.

The multihomed site does not number any of its interfaces from CNB #1 or CNB #2. Section 2.3 describes how the multihomed site uses CNB #1 and CNB #2.

2.2.2. Site Addressing

The site obtains a Site Address Block (SAB), either from Unique Local Address (ULA) [RFC4193] space, or by some other means. For the purpose of example, assume that the site obtains a /48 from ULA space.

The site then draws a /59 prefix and a /60 prefix from the SAB. In this document, we call those prefixes SAB #1 and SAB #2. Note that SAB #1 and CNB #1 are both /59 prefixes. Likewise, SAB #2 and CNB #2 are both /60 prefixes. In Section 2.3, the site will map SAB #1 to CNB #1 and SAB #2 to CNB #2. Mapped prefixes must be of identical size.

The site then numbers its resources from SAB #1 and SAB #2. SAB #1 and SAB #2 are the only usable portions of the SAB, because they are the only prefixes that will be mapped to CNB addresses.

During periods of normal operation, hosts that are numbered from SAB #1 receive inbound traffic from Upstream Provider #1. Hosts that are numbered from SAB #2 receive inbound traffic from Upstream Provider

#2. Selected hosts receive inbound traffic from both upstream providers, balancing the load between them. These hosts have multiple addresses, with at least one address being drawn from SAB #1 and at least one address being drawn from SAB #2.

Section 2.7 explains how load balancing is achieved.

2.3. Address Translation

Both NPTv6 translators are configured with the following rules:

For outbound packets, if the first 59 bits of the source address identify SAB #1, overwrite those 59 bits with the 59 bits that identify CNB #1

For outbound packets, if the first 60 bits of the source address identify SAB #2, overwrite those 60 bits with the 60 bits that identify CNB #2

For outbound packets, if none of the conditions above are met, silently discard the packet

For inbound packets, if the first 59 bits of the destination address identify CNB #1, overwrite those 59 bits with the 59 bits that identify SAB #1

For inbound packets, if the first 60 bits of the destination address identify CNB #2, overwrite those 60 bits with the 60 bits that identify SAB #2

For inbound packets, if none of the conditions above are met, silently discard the packet

Due to the nature of the rules described above, NPTv6 translation is stateless. Therefore, traffic flows do not need to be symmetric across NPTv6 translators. Furthermore, a traffic flow can shift from one NPTv6 translator to another without causing transport-layer session failure.

2.4. Domain Name System (DNS)

In order to make all site resources reachable by domain name [RFC1034], the site publishes AAAA records [RFC3596] associating each resource with all of its CNB addresses. While this DNS architecture is sufficient, it is suboptimal. Traffic that both originates and terminates within the site traverses NPTv6 translators needlessly. Several optimizations are available. These optimizations are well understood and have been applied to [RFC1918] networks for many

years.

2.5. Routing

Upstream Provider #1 uses an Interior Gateway Protocol to flood topology information throughout its domain. It also uses BGP [RFC4271] to distribute customer and peer reachability information.

PE #1 acquires a route to CNB #1 with NEXT-HOP equal to the outside interface of NPTv6 #1. PE #1 can either learn this route from a single-hop eBGP session with NPTv6 #1, or acquire it through static configuration. In either case, PE #1 overwrites the NEXT-HOP of this route with its own loopback address and distributes the route throughout Upstream Provider #1 using iBGP. The LOCAL PREF for this route is set high, so that the route will be preferred to alternative routes to CNB #1. Upstream Provider #1 does not distribute this route to CNB #1 outside of its own borders because it is part of the larger aggregate PAB #1, which is itself advertised.

NPTv6 #1 acquires a default route with NEXT-HOP equal to the directly connected interface on PE #1. NPTv6 #1 can either learn this route from a single-hop eBGP session with PE #1, or acquire it through static configuration.

Similarly, Backup PE #1 acquires a route to CNB #1 with NEXT-HOP equal to the outside interface of NPTv6 #2. Backup PE #1 can either learn this route from a multi-hop eBGP session with NPTv6 #2, or acquire it through static configuration. In either case, Backup PE #1 overwrites the NEXT-HOP of this route with its own loopback address and distributes the route throughout Upstream Provider #1 using iBGP. Distribution procedures are defined in [I-D.ietf-idr-best-external]. The LOCAL PREF for this route is set low, so that the route will not be preferred to alternative routes to CNB #1. Upstream Provider #1 does not distribute this route to CNB #1 outside of its own borders.

Even if Backup PE #1 maintains an eBGP session NPTv6 #2, it does not advertise the default route through that eBGP session. During failures, Backup PE #1 does not attract outbound traffic to itself.

PE #2 acquires a route to CNB #1 with NEXT-HOP equal to the outside interface of NPTv6 #2. PE #2 can either learn this route from a single-hop eBGP session with NPTv6 #2, or acquire it through static configuration. PE #2 uses this route to enforce source address filtering [RFC2827] on the interface through which it is connected to NPTv6 #2. PE #2 does not advertise this route to CNB #1 to any or its routing peers.

Finally, the Autonomous System Border Routers (ASBR) contained by Upstream Provider #1 maintain eBGP sessions with their peers. The ASBRs advertise only PAB #1 through those eBGP sessions. Upstream Provider #1 does not advertise any of the following to its eBGP peers:

- any prefix that is contained by PAB #1 (i.e., more specific)

- PAB #2 or any part thereof

- the SAB or any part thereof

Upstream Provider #2 is configured in a manner analogous to that described above.

Because both NPTv6 gateways are configured with identical translation rules, and because both PE routers maintain routes to CNB #1 and CNB #1, outbound packets can traverse either NPTv6 gateway. Outbound routing is controlled by the site and therefore, is beyond the scope of this document.

2.6. Failure Detection and Recovery

When PE #1 loses its route to CNB #1, it withdraws its iBGP advertisement for that prefix from Upstream Provider #1. The route advertised by Backup PE #1 remains and Backup PE #1 attracts traffic bound for CNB #1 to itself. Backup PE #1 forwards that traffic through the tunnel to NPTv6 #2. NPTv6 #2 performs translations and delivers the traffic to the Internal Network.

Likewise, when NPTv6 #1 loses its default route, it makes itself unavailable as a gateway for other hosts on the Internal Network. NPTv6 #2 attracts all outbound traffic to itself and forwards that traffic through Upstream Provider #2. Because PE #2 maintains routes to both CNB #1 and CNB #2, it does not discard any traffic from CNB #1 or CNB #2 as a result of source address filtering. The mechanism by which NPTv6 #1 makes itself unavailable as a gateway is beyond the scope of this document.

If PE #1 maintains a single-hop eBGP session with NPTv6 #1, the failure of that eBGP session will cause both routes mentioned above to be lost. Otherwise, another failure detection mechanism such as BFD [RFC5881] is required.

Regardless of the failure detection mechanism, inbound traffic traverses the tunnel only during failure periods and outbound traffic never traverses the tunnel. Furthermore, restoration is localized. As soon as the advertisement for CNB #1 is withdrawn throughout

Upstream Provider #1, restoration is complete.

Transport-layer connections survive Failure/Recovery events because both NPTv6 translators implement identical translation rules. When a traffic flow shifts from one translator to another, neither the source address nor the destination address changes.

2.7. Load Balancing

Outbound load balancing is controlled by the site and is beyond the scope of this document.

For inbound traffic, addressing determines load balancing. If a host is numbered from SAB #1, its address is mapped into CNB #1, which is announced only by Upstream Provider #1 (as part of PAB #1). Therefore, during periods of normal operation, all traffic bound for that host traverses Upstream Provider #1 and NPTv6 #1. Likewise, if a host is numbered from SAB #2, its address is mapped into CNB #2, which is announced only by Upstream Provider #2 (as part of PAB #2). Therefore, during periods of normal operation, all traffic bound for that host traverses Upstream Provider #2 and NPTv6 #2.

Selected hosts receive inbound traffic from both upstream providers, balancing load between them. These hosts have multiple addresses, with at least one address being drawn from SAB #1 and at least one address being drawn from SAB #2. The number of addresses drawn from each range determines how connections originating outside of the multihomed site distribute inbound load.

Recall that all CNB addresses associated with a host are published in DNS. When a remote host initiates a TCP connection, it selects from among these addresses in a relatively random manner. If it selects an address from CNB #1, inbound packets belonging to that connection will traverse Upstream Provider #1 and NPTv6 #1. If it selects an address from CNB #2, inbound packets belonging to that connection will traverse Upstream Provider #2 and NPTv6 #2.

When the multiply addressed host initiates a connection, it associates one of its own addresses with the connection. If the address that it chooses is from SAB #1, that address will be mapped to a CNB #1 address and return traffic will traverse Upstream Provider #1 and NPTv6 #1. Alternatively, if the host selects an address from SAB #2, that address will be mapped to a CNB #2 address and return traffic will traverse Upstream Provider #1 and NPTv6 #2.

3. Discussion

When compared to the multihoming architecture described in Section 2.4 of [RFC6296], the proposed architecture achieves transport-layer survivability at the cost of backup PE hardware and additional configuration. The cost of backup PE hardware is minimal, because backup PE routers are very lightly loaded during periods of normal operation. However, in the example provided above, Upstream Provider #1 must configure the following additional items:

- o an interface to the multihomed site on Backup PE #1
- o a forwarding tunnel connecting that interface to NPTv6 #2
- o either a multi-hop eBGP session between Backup PE #1 and NPTv6 #2, or a static route to CNB #1 on Backup PE #1

Furthermore, if PE #1 does not maintain an eBGP session with NPTv6 #1, Upstream Provider #1 must configure a static route to CNB #2 (as well as CNB #1) on PE #1. However, if PE #1 does maintain an eBGP session with NPTv6 #1, Upstream Provider #1 must configure policy on that session causing it to accept, but not readvertise a path to CNB #2.

4. IANA Considerations

This document requires no IANA actions.

5. Security Considerations

As with any architecture that modifies source and destination addresses, the operation of access control lists, firewalls and intrusion detection systems may be impacted. Also many users may confuse NPTv6 translation with a NAT. Two limitations of NAT are that a) it does not support incoming connections without special configuration and b) it requires symmetric routing across the NAT device. Many users understood these limitations to be security features. Because NPTv6 has neither of these limitations, it also offers neither of these features.

6. Acknowledgments

Thanks to Holger Zuleger, John Scudder and Yakov Rekhter for their helpful comments, encouragement and support. Special thanks to Johann Jonsson, James Piper, Ravinder Wali, Ashte Collins, Inga

Rollins and an anonymous donor, without whom this memo would not have been written.

7. References

7.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.
- [RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.
- [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.
- [RFC3582] Abley, J., Black, B., and V. Gill, "Goals for IPv6 Site-Multihoming Architectures", RFC 3582, August 2003.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.
- [RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.
- [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, June 2010.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.

7.2. Informative References

- [I-D.ietf-idr-best-external] Marques, P., Fernando, R., Chen, E., Mohapatra, P., and H. Gredler, "Advertisement of the best external route in BGP", draft-ietf-idr-best-external-05 (work in progress), January 2012.

Authors' Addresses

Ron Bonica
Juniper Networks
Sterling, Virginia 20164
USA

Email: rbonica@juniper.net

Fred Baker
Cisco Systems
Santa Barbara, California 93117
USA

Email: fred@cisco.com

Margaret Wasserman
Painless Security
356 Abbott Street
North Andover, Massachusetts 01845
USA

Phone: +1 781 405 7464
Email: mrw@painless-security.com
URI: <http://www.painless-security.com>

Gregory J. Miller
Verizon
Ashburn, Virginia 20147
USA

Email: gregory.j.miller@verizon.com

Warren Kumari
Google, Inc.
Mountain View, California 94043

Email: warren@kumari.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 14, 2012

B. Carpenter
Univ. of Auckland
S. Jiang
Huawei Technologies Co., Ltd
W. Tarreau
Exceliance
June 12, 2012

Using the IPv6 Flow Label for Server Load Balancing
draft-carpenter-flow-label-balancing-01

Abstract

This document describes how the IPv6 flow label as currently specified can be used to enhance 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 December 14, 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. Summary of Flow Label Specification	3
3. Summary of Load Balancing Techniques	4
4. Applying the Flow Label to L3/L4 Load Balancing	7
5. Security Considerations	9
6. IANA Considerations	10
7. Acknowledgements	10
8. Change log [RFC Editor: Please remove]	10
9. References	10
9.1. Normative References	10
9.2. Informative References	11
Authors' Addresses	11

1. Introduction

The IPv6 flow label has been redefined [RFC6437] and is now a recommended IPv6 node requirement [RFC6434]. 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 brief introductions to the flow label and to load balancing techniques, and then describes how the flow label can be used to enhance layer 3/4 load balancers in particular.

The motivation for this approach is to improve the performance of most types of layer 3/4 load balancers, especially for traffic including multiple IPv6 extension headers and in particular for fragmented packets. Fragmented packets, often the result of customers reaching the load balancer via a VPN with a limited MTU, are a common performance problem.

2. Summary of Flow Label Specification

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]. According to this definition, the flow label should be set to a constant value for a given traffic flow (such as an HTTP connection).

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. In fact, the logic of finding the transport header is always more complex for IPv6 than for IPv4, due to the absence of an Internet Header Length field in IPv6. 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 RFC 6437, source hosts should set the flow label, but if they do not (i.e. its value is zero), forwarding nodes (such as the first-hop router) may set it 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. By default, the flow label value should be calculated by a stateless algorithm. The resulting value should form part of a statistically uniform distribution.

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 substitute for the source port number, found at a fixed position in the layer 3 header.

The flow label is defined as an end-to-end component of the IPv6 header, but there are three qualifications to this:

1. Until the RFC 6437 standard is widely implemented as recommended by RFC 6434, the flow label will often be set to the default value of zero.
2. Because of the recommendation to use a stateless algorithm to calculate the label, there is a 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.
3. The flow label field is in an unprotected part of the IPv6 header, which means that intentional or unintentional changes to its value cannot be trivially detected by a receiver.

The first two points are addressed below in Section 4 and the third in Section 5.

3. Summary of Load Balancing Techniques

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, and the actual load balancing algorithm (the choice of which server to use for a new client session) is irrelevant to this discussion.

- o 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.
- o 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.

- o 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.
- o Finally the "front end" of many load balancing systems is a layer 3/4 load balancer. While it can sometimes be a dedicated hardware, it also happens to be a standard function of some network switches or routers (eg: using ECMP, [RFC2991]). 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. There are two common types of layer 3/4 load balancers, the totally stateless ones which only act on packets, generally involving a per-packet hashing of easy-to-find information such as the source address and/or port into a server number, and the stateful ones which take the routing decision on the very first packets of a session and maintain the same direction for all packets belonging to the same session. Clearly, both types of layer 3/4 balancers 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 could not be affected by use of the flow label to identify sessions:

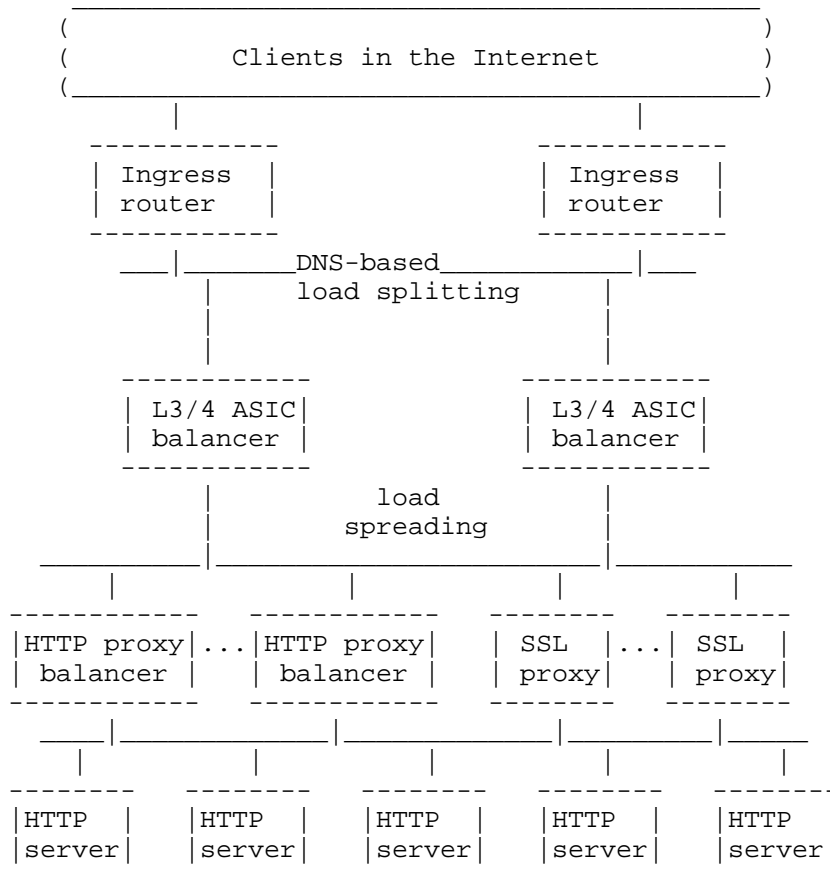
1. Balancers use various techniques to redirect traffic to a specific target server.
 - All servers are configured with the same IP address, they are all on the same LAN, and the load balancer sends directly to their individual MAC addresses.
 - Each server has its own IP address, and the balancer uses 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.

However, usage by the proxies seems unlikely to be cost-effective, so in this document we focus only on layer 3/4 balancers.

4. Applying the Flow Label to L3/L4 Load Balancing

The suggested model for using the flow label in a load balancing mechanism is as follows:

- o We are only concerned with IPv6 traffic in which the flow label value has been set at or near the source according to [RFC6437]. If the flow label of an incoming packet is zero, load balancers will continue to use the transport header in the traditional way. As the use of the flow label becomes more prevalent according to RFC 6434, load balancers, and therefore users, will reap a growing performance benefit.
- o If the flow label of an incoming packet is non-zero, 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. If any IPv6 extension headers, including fragment headers, are present, this will be significantly quicker than searching for the transport port numbers later in the packet. Moreover, the transport layer information such as the source port is not repeated in fragments, which generally prevents stateless load balancers from supporting fragmented traffic since they generally cannot reassemble fragments.

Note that balancers usually do not need to consider the destination address as it is always the same, i.e., the server address.

A stateless layer 3/4 load balancer would simply apply a hash algorithm to the 2-tuple {source address, flow label} on all packets, in order to select the same target server consistently for a given flow.

A stateful layer 3/4 load balancer would apply its usual load distribution algorithm to the first packet of a session, and store the {2-tuple, server} association in a table so that subsequent packets belonging to the same session are forwarded to the same server. Thus, 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 NAT are not expected to 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.

- o 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.
- o SSL and HTTP proxies, if present, should forward the flow label value towards the server. This has no performance benefit, but is consistent with the general RFC 6437 model for the flow label.

It should be noted that the performance benefit, if any, depends entirely on engineering trade-offs in the design of the L3/L4 balancer. An extra test is needed (is the label non-zero?), but all logic for handling extension headers can be omitted except for the first packet of a new flow. Since the only state to be stored is the 2-tuple and the server identifier, storage requirements will be reduced. Additionally, the method will work for fragmented traffic and for flows where the transport information is missing (unknown transport protocol) or obfuscated (e.g., IPsec). Traffic reaching the load balancer via a VPN is particularly prone to the fragmentation issue, due to MTU size issues. For some load balancer designs, these are very significant advantages.

In the unlikely event of two simultaneous flows from the same source address 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. Moreover, it is very likely that there will be many more flow label values than servers at most sites (1 million possible label values), so it is already expected that multiple flow label values will end up on the same server for a given IP address. In the case where many thousands of clients are hidden behind the same large-scale NAT with a single IP address, the assumption of low probability of conflicts might become incorrect unless flow label values are random enough to avoid following similar sequences for all clients. This is not expected to be a factor for IPv6 anyway, 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.

5. 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. This risk already exists today where the source address and port are used as hashing key in layer 3/4 load balancers, as well as where a persistence cookie is used in HTTP to designate a server. It even exists on layer 3 components which only rely on the source address to select a destination, making them more DDoS-prone. Nevertheless, all these methods are currently used because the benefits for load balancing and persistence hugely outweigh the risks.

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 site ingress router or the layer 3/4 balancer 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. This would deal with both intentional and accidental changes to the flow label.

RFC 6437 notes in its Security Considerations that if the covert channel risk is considered significant, a firewall might rewrite non-zero flow labels. As long as this is done as described in RFC 6437, it will not invalidate the mechanisms described above.

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. Still, it is important to understand that it is always trivial to force a load balancer to stick to the same server during an attack, so the security of the whole solution must not rely on the unpredictability of the flow label values alone, but should include defensive measures like most load balancers already have against abnormal use of source address or session cookies.

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 makes it harder for an attacker to target a specific server, because the association between a flow label value and a server is not known externally.

6. IANA Considerations

This document requests no action by IANA.

7. 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].

8. Change log [RFC Editor: Please remove]

draft-carpenter-flow-label-balancing-01: update following comments, 2012-06-12.

draft-carpenter-flow-label-balancing-00: restructured after IETF83, 2012-05-08.

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.

9. References

9.1. Normative References

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

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, December 2011.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme,

"IPv6 Flow Label Specification", RFC 6437, November 2011.

9.2. Informative References

- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.
- [RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, November 2000.
- [RFC4864] Van de Velde, G., Hain, T., Droms, R., Carpenter, B., and E. Klein, "Local Network Protection for IPv6", RFC 4864, May 2007.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.
- [RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, November 2011.
- [Tarreau] Tarreau, W., "Making applications scalable with load balancing", 2006, <http://lwt.eu/articles/2006_lb/>.

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

INTAREA WG
Internet-Draft
Intended status: Informational
Expires: October 20, 2012

M. Boucadair
France Telecom
J. Touch
USC/ISI
P. Levis
France Telecom
R. Penno
Cisco
April 18, 2012

Analysis of Solution Candidates to Reveal a Host Identifier (HOST_ID) in
Shared Address Deployments
draft-ietf-intarea-nat-reveal-analysis-02

Abstract

This document analyzes a set of solution candidates to mitigate some of the issues encountered when address sharing is used. In particular, this document focuses on means to reveal a host identifier (HOST_ID) when a Carrier Grade NAT (CGN) or application proxies are involved in the path. This host identifier must be unique to each host under the same shared IP address.

Requirements Language

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 RFC 2119 [RFC2119].

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 October 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	4
1.1. Context	4
1.2. Purpose and Scope	4
2. Problem to Be Solved	5
2.1. IPv6 May Also Be Concerned	6
3. Solutions Analysis	6
3.1. Requirements	6
3.2. Synthesis	6
3.3. Recommendation	9
4. HOST_ID and Privacy	9
5. IANA Considerations	11
6. Security Considerations	11
7. Acknowledgments	11
8. References	11
8.1. Normative References	11
8.2. Informative References	12
Appendix A. Detailed Solutions Analysis	14
A.1. Use the Identification Field of IP Header (IP-ID)	14
A.1.1. Description	14
A.1.2. Analysis	14
A.2. Define an IP Option	14
A.2.1. Description	14
A.2.2. Analysis	15
A.3. Assign Port Sets	15
A.3.1. Description	15
A.3.2. Analysis	15
A.4. Use ICMP	16
A.4.1. Description	16
A.4.2. Analysis	16
A.5. Define a TCP Option	17
A.5.1. Description	17
A.5.2. Analysis	17
A.6. PROXY Protocol	18
A.6.1. Description	18
A.6.2. Analysis	19
A.7. Host Identity Protocol (HIP)	19
A.7.1. Description	19
A.7.2. Analysis	19
A.8. Inject Application Headers	19
A.8.1. Description	19
A.8.2. Analysis	20
Authors' Addresses	20

1. Introduction

1.1. Context

As reported in [RFC6269], several issues are encountered when an IP address is shared among several subscribers. Examples of such issues are listed below:

- o Implicit identification (Section 13.2 of [RFC6269])
- o SPAM (Section 13.3 of [RFC6269])
- o Blacklisting a mis-behaving host (Section 13.1 of [RFC6269])
- o Redirect users with infected machines to a dedicated portal (Section 5.1 of [RFC6269])

The sole use of the IPv4 address is not sufficient to uniquely distinguish a host. As a mitigation, it is tempting to investigate means which would help in disclosing an information to be used by the remote server as a means to uniquely disambiguate packets of hosts using the same IPv4 address.

The risk of not mitigating these issues are: OPEX increase for IP connectivity service providers (costs induced by calls to a hotline), revenue loss for content providers (loss of users audience), customers unsatisfaction (low quality of experience, service segregation, etc.).

1.2. Purpose and Scope

The purpose of this document is to analyze a set of alternative channels to convey a host identifier and to assess to what extent they solve the problem described in Section 2. Below are listed the candidates analyzed in the document:

- o Use the Identification field of IP header (denoted as IP-ID, Appendix A.1).
- o Define a new IP option (Appendix A.2).
- o Assign port sets (Appendix A.3).
- o Use ICMP (Appendix A.3).
- o Define a new TCP Option (Appendix A.5).
- o Enable Proxy Protocol (Appendix A.6).
- o Activate HIP (Appendix A.7).
- o Inject application headers (Appendix A.8).

A synthesis is provided in Section 3 while the detailed analysis is elaborated in Appendix A.

Section 4 discusses privacy issues common to all HOST_ID solutions. It is out of scope of this document to elaborate on privacy issues specific to each solution.

2. Problem to Be Solved

Observation: Today, some servers use the source IPv4 address as an identifier to treat some incoming connections differently. Tomorrow, due to the introduction of CGNs (e.g., NAT44 [RFC3022], NAT64 [RFC6146]), that address will be shared. In particular, when a server receives packets from the same source address, because this address is shared, the server does not know which host is the sending host [RFC6269].

Objective: The server should be able to sort out the packets by sending host.

Requirement: The server must have extra information than the source IP address to differentiate the sending host. We call HOST_ID this information.

For all solutions analyzed, we provide answers to the following questions:

What is the HOST_ID? It must be unique to each host under the same IP address. It does not need to be globally unique. Of course, the combination of the (public) IP source address and the identifier (i.e., HOST_ID) ends up being relatively unique. As unique as today's 32-bit IPv4 addresses which, today, can change when a host re-connects.

Where is the HOST_ID? (which protocol, which field): If the HOST_ID is put at the IP level, all packets will have to bear the identifier. If it is put at a higher connection-oriented level, the identifier is only needed once in the session establishment phase (for instance TCP three-way-handshake), then, all packets received in this session will be attributed to the HOST_ID designated during the session opening.

Who puts the HOST_ID? For almost all the analyzed solutions, the address sharing function injects the HOST_ID. When there are several address sharing functions in the data path, we describe to what extent the proposed solution is efficient. Another option to avoid potential performance degradation is to let the host inject its HOST_ID but the address sharing function will check its content (just like an IP anti-spoofing function).

What are the security considerations? Security considerations are common to all analyzed solutions (see Section 6). Privacy-related aspects are discussed in Section 4.

2.1. IPv6 May Also Be Concerned

Some of the issues mentioned in Section 2 are independent of IPv4 vs. IPv6. Even in IPv6, address sharing can be used for a variety of reasons (e.g., to hide network topology, to defeat hosts from offering network services directly, etc.).

A solution to reveal HOST_ID is also needed in IPv6 deployment.

3. Solutions Analysis

3.1. Requirements

Whatever the channel used to convey the HOST_ID, the following requirements are to be met:

Uniqueness of identifiers in HOST_ID: It is RECOMMENDED that HOST_IDs be limited to providing local uniqueness rather than global uniqueness.

Refresh rate of HOST_ID: Address sharing function SHOULD NOT use permanent HOST_ID values.

Manipulate HOST_IDs: Address sharing function SHOULD be able to strip, re-write and add HOST_ID fields.

Interference between HOST_IDs: An address sharing function, able to inject HOST_IDs in several layers, SHOULD reveal subsets of the same information (e.g., full IP address, lower 16 bits of IP address, etc.).

3.2. Synthesis

The following Table 1 summarizes the approaches analyzed in this document.

- o "Success ratio" indicates the ratio of successful communications when the option is used. Provided figures are inspired from the results documented in [Options].
- o "Deployable today" indicates if the solution can be generalized without any constraint on current architectures and practices.
- o "Possible Perf Impact" indicates the level of expected performance degradation. The rationale behind the indicated potential performance degradation is whether the injection requires some treatment at the IP level or not.

- o "OS TCP/IP Modif" indicates whether a modification of the OS TCP/IP stack is required at the server side.

	IP Option	TCP Option	IP-ID	HTTP Header (XFF)	Proxy	Port Set	HIP	ICMP
UDP	Yes	No	Yes	No	No	Yes		Yes
TCP	Yes	Yes	Yes	No	Yes	Yes		Yes
HTTP	Yes	Yes	Yes	Yes	Yes	Yes		Yes
Encrypted Traffic	Yes	Yes	Yes	No	Yes	Yes		Yes
Success Ratio	30%	99%	100%	100%	Low	100%	Low	~100% (6)
Possible Perf Impact	High	Med to High	Low to Med	Med to High	High	No	N/A	High
OS TCP/IP Modif	Yes	Yes	Yes	No	No	No		Yes
Deployable Today	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Notes			(1)	(2)		(1) (3)	(4) (5)	(7)

Notes:

- (1) Requires mechanism to advertise NAT is participating in this scheme (e.g., DNS PTR record).
- (2) This solution is widely deployed.
- (3) When the port set is not advertised, the solution is less efficient for third-party services.
- (4) Requires the client and the server to be HIP-compliant and HIP infrastructure to be deployed.
- (5) If the client and the server are HIP-enabled, the address sharing function does not need to insert a host-hint. If the client is not HIP-enabled, designing the device that performs address sharing to act as a UDP/TCP-HIP relay is not viable.
- (6) Implementation-specific.
- (7) The solution is inefficient in various scenarios as discussed in Section 3.

Figure 1: Table 1: Summary of analyzed solutions.

According to the above table and the analysis elaborated in Appendix A:

- o IP Option, IP-ID and Proxy Protocol proposals are broken;
- o HIP is not largely deployed;
- o The use of Port Set may contradict the port randomization [RFC6056] requirement identified in [RFC6269]. This solution can be used by a service provider for the delivery of its own service offerings relying on implicit identification.
- o XFF is de facto standard deployed and supported in operational networks (e.g., HTTP Servers, Load-Balancers, etc.).
- o From an application standpoint, the TCP Option is superior to XFF/Forwarded-For since it is not restricted to HTTP. Nevertheless XFF/Forwarded-For is compatible with the presence of address sharing and load-balancers in the communication path. To provide a similar functionality, the TCP Option may be extended to allow conveying a list of IP addresses and port numbers to not lose the source IP address in the presence of load-balancers. Another alternative is to combine the usage of both the HOST_ID TCP Option and XFF/Forwarded-For. Extending TCP is still possible as analyzed in [ExtendTCP].

3.3. Recommendation

Taking into account the analysis above and [RFC6269] context, the following recommendation is made to mitigate the problem formulated in Section 2:

An address sharing function SHOULD support HOST_ID TCP Option (Appendix A.5).

4. HOST_ID and Privacy

IP address sharing is motivated by a number of different factors. For years, many network operators have conserved the use of public IPv4 addresses by making use of Customer Premises Equipment (CPE) that assigns a single public IPv4 address to all hosts within the customer's local area network and uses NAT [RFC3022] to translate between locally unique private IPv4 addresses and the CPE's public address. With the exhaustion of IPv4 address space, address sharing between customers on a much larger scale is likely to become much

more prevalent. While many individual users are unaware of and uninvolved in decisions about whether their unique IPv4 addresses get revealed when they send data via IP, some users realize privacy benefits associated with IP address sharing, and some may even take steps to ensure that NAT functionality sits between them and the public Internet. IP address sharing makes the actions of all users behind the NAT function unattributable to any single host, creating room for abuse but also providing some identity protection for non-abusive users who wish to transmit data with reduced risk of being uniquely identified.

The proposals considered in this document add a measure of uniqueness back to hosts that share a public IP address. The extent of that uniqueness depends on which information is included in the HOST_ID.

The volatility of the HOST_ID information is similar to the source IP address: a distinct HOST_ID may be used by the address sharing function when the host reboots or gets a new internal IP address. As with persistent IP addresses, persistent HOST_IDs facilitate user tracking over time.

As a general matter, the HOST_ID proposals do not seek to make hosts any more identifiable than they would be if they were using a public, non-shared IP address. However, depending on the solution proposal, the addition of HOST_ID information may allow a device to be fingerprinted more easily than it otherwise would be. Should multiple solutions be combined (e.g., TCP Option and XFF) that include different pieces of information in the HOST_ID, fingerprinting may become even easier.

The trust placed in the information conveyed in the HOST_ID is likely to be the same as for current practices with source IP addresses. In that sense, a HOST_ID can be spoofed as this is also the case for spoofing an IP address. Furthermore, users of network-based anonymity services (like Tor) may be capable of stripping HOST_ID information before it reaches its destination.

HOST_ID specification document(s) SHOULD explain the privacy impact of the solutions they specify, including the extent of HOST_ID uniqueness and persistence, assumptions made about the lifetime of the HOST_ID, whether and how the HOST_ID can be obfuscated or recycled, and the impact of the use of the HOST_ID on device or implementation fingerprinting. [I-D.iab-privacy-considerations] provides further guidance.

For more discussion about privacy, refer to [RFC6462].

5. IANA Considerations

This document does not require any action from IANA.

6. Security Considerations

The same security concerns apply for the injection of an IP option, TCP Option and application-related content (e.g., XFF) by the address sharing device. If the server trusts the content of the HOST_ID field, a third party user can be impacted by a misbehaving user to reveal a "faked" HOST_ID (e.g., original IP address).

HOST_ID may be used to leak information about the internal structure of a network behind an address sharing function. If this behavior is undesired for the network administrator, the address sharing function can be configured to strip any existing HOST_ID in received packets from internal hosts.

HOST_ID specification documents SHOULD elaborate further on threats inherent to each individual solution to convey the HOST_ID (e.g., use of the IP-ID field to count hosts behind a NAT [Count]).

7. Acknowledgments

Many thanks to D. Wing and C. Jacquenet for their review, comments and inputs.

Thanks also to P. McCann, T. Tsou, Z. Dong, B. Briscoe, T. Taylor, M. Blanchet, D. Wing and A. Yourtchenko for the discussions in Prague.

Some of the issues related to defining a new TCP Option have been raised by L. Eggert.

Privacy text is provided by A. Cooper.

8. References

8.1. Normative References

- [RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.
- [RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, January 2011.

8.2. Informative References

- [Count] "A technique for counting NATted hosts",
<<http://www.cs.columbia.edu/~smb/papers/fnat.pdf>>.
- [ExtendTCP]
Honda, M., Nishida, Y., Raiciu, C., Greenhalgh, A., Handley, M. and H. Tokuda,, "Is it still possible to extend TCP?", November 2011,
<<http://nrg.cs.ucl.ac.uk/mjh/tmp/mboxes.pdf>>.
- [I-D.abdo-hostid-tcpopt-implementation]
Abdo, E., Boucadair, M., and J. Queiroz, "HOST_ID TCP Options: Implementation & Preliminary Test Results", draft-abdo-hostid-tcpopt-implementation-02 (work in progress), January 2012.
- [I-D.chen-intarea-v4-uid-header-option]
Wu, Y., Ji, H., Chen, Q., and T. ZOU), "IPv4 Header Option For User Identification In CGN Scenario", draft-chen-intarea-v4-uid-header-option-00 (work in progress), March 2011.
- [I-D.iab-privacy-considerations]
Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., and J. Morris, "Privacy Considerations for Internet Protocols", draft-iab-privacy-considerations-02 (work in progress), March 2012.
- [I-D.ietf-appsawg-http-forwarded]
Petersson, A. and M. Nilsson, "Forwarded HTTP Extension", draft-ietf-appsawg-http-forwarded-01 (work in progress), March 2012.
- [I-D.ietf-intarea-ipv4-id-update]
Touch, J., "Updated Specification of the IPv4 ID Field", draft-ietf-intarea-ipv4-id-update-04 (work in progress), September 2011.
- [I-D.wing-nat-reveal-option]

Yourtchenko, A. and D. Wing, "Revealing hosts sharing an IP address using TCP option",
draft-wing-nat-reveal-option-03 (work in progress),
December 2011.

[I-D.yourtchenko-nat-reveal-ping]

Yourtchenko, A., "Revealing hosts sharing an IP address using ICMP Echo Request",
draft-yourtchenko-nat-reveal-ping-00 (work in progress),
March 2012.

[Not_An_Option]

R. Fonseca, G. Porter, R. Katz, S. Shenker, and I. Stoica,, "IP options are not an option", 2005, <<http://www.eecs.berkeley.edu/Pubs/TechRpts/2005/EECS-2005-24.html>>.

[Options]

Alberto Medina, Mark Allman, Sally Floyd, "Measuring Interactions Between Transport Protocols and Middleboxes", 2005, <<http://conferences.sigcomm.org/imc/2004/papers/p336-medina.pdf>>.

[Proxy]

Tarreau, W., "The PROXY protocol", November 2010, <<http://haproxy.1wt.eu/download/1.5/doc/proxy-protocol.txt>>.

[RFC2753]

Yavatkar, R., Pendarakis, D., and R. Guerin, "A Framework for Policy-based Admission Control", RFC 2753, January 2000.

[RFC5201]

Moskowitz, R., Nikander, P., Jokela, P., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

[RFC6146]

Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6269]

Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, June 2011.

[RFC6302]

Durand, A., Gashinsky, I., Lee, D., and S. Sheppard, "Logging Recommendations for Internet-Facing Servers", BCP 162, RFC 6302, June 2011.

[RFC6462]

Cooper, A., "Report from the Internet Privacy Workshop", RFC 6462, January 2012.

[Trusted_ISPs]

"Trusted XFF list", <http://meta.wikimedia.org/wiki/XFF_project#Trusted_XFF_list>.

Appendix A. Detailed Solutions Analysis

A.1. Use the Identification Field of IP Header (IP-ID)

A.1.1. Description

IP-ID (Identification field of IP header) can be used to insert an information which uniquely distinguishes a host among those sharing the same IPv4 address. An address sharing function can re-write the IP-ID field to insert a value unique to the host (16 bits are sufficient to uniquely disambiguate hosts sharing the same IP address). Note that this field is not altered by some NATs; hence some side effects such as counting hosts behind a NAT as reported in [Count].

A variant of this approach relies upon the format of certain packets, such as TCP SYN, where the IP-ID can be modified to contain a 16 bit HOST_ID. Address sharing devices performing this function would require to indicate they are performing this function out of band, possibly using a special DNS record.

A.1.2. Analysis

This usage is not compliant with what is recommended in [I-D.ietf-intarea-ipv4-id-update].

A.2. Define an IP Option

A.2.1. Description

A solution alternative to convey the HOST_ID is to define an IP option [RFC0791]. HOST_ID IP option can be inserted by the address sharing function to uniquely distinguish a host among those sharing the same IP address. An example of such option is documented in [I-D.chen-intarea-v4-uid-header-option]. This IP option allows to convey an IPv4 address, an IPv6 prefix, a GRE key, IPv6 Flow Label, etc.

Another way for using IP option has been described in Section 4.6 of [RFC3022].

A.2.2. Analysis

Unlike the solution presented in Appendix A.5, this proposal can apply for any transport protocol. Nevertheless, it is widely known that routers (and other middleboxes) filter IP options. IP packets with IP options can be dropped by some IP nodes. Previous studies demonstrated that "IP Options are not an option" (Refer to [Not_An_Option], [Options]).

As a conclusion, using an IP option to convey a host-hint is not viable.

A.3. Assign Port Sets

A.3.1. Description

This solution does not require any action from the address sharing function to disclose a host identifier. Instead of assuming all transport ports are associated with one single host, each host under the same external IP address is assigned a restricted port set. These port sets are then advertised to remote servers using off-line means. This announcement is not required for the delivery of internal services (i.e., offered by the service provider deploying the address sharing function) relying on implicit identification.

Port sets assigned to hosts may be static or dynamic.

Port set announcements to remote servers do not require to reveal the identity of individual hosts but only to advertise the enforced policy to generate non-overlapping port sets (e.g., the transport space associated with an IP address is fragmented to contiguous blocks of 2048 port numbers).

A.3.2. Analysis

The solution does not require defining new fields nor options; it is policy-based.

The solution may contradict the port randomization as identified in [RFC6269]. A mitigation would be to avoid assigning static port sets to individual hosts.

The method is convenient for the delivery of services offered by the service provider offering also the IP connectivity service.

A.4. Use ICMP

A.4.1. Description

Another alternative is to convey the HOST_ID using a separate notification channel than the packets issued to invoke the service.

An implementation example is defined in [I-D.yourtchenko-nat-reveal-ping]. This solution relies on a mechanism where the address sharing function encapsulates the necessary differentiating information into an ICMP Echo Request packet that it sends in parallel with the initial session creation (e.g., SYN). The information included in the ICMP Request Data portion describes the five-tuples as seen on both of the sides of the address sharing function.

A.4.2. Analysis

- o This ICMP proposal is valid for both UDP and TCP. Address sharing function may be configurable with the transport protocol which is allowed to trigger those ICMP messages.
- o A hint should be provided to the ultimate server (or intermediate nodes) an ICMP Echo Request conveys a HOST_ID. This may be implemented using magic numbers.
- o Even if ICMP packets are blocked in the communication path, the user connection does not have to be impacted.
- o Some implementations requiring to delay the establishment of a session until receiving the companion ICMP Echo Request, may lead to some user experience degradation.
- o Because of the presence of load-balancers in the path, the ultimate server receiving the SYN packet may not be the one which may receive the ICMP message conveying the HOST_ID.
- o Because of the presence of load-balancers in the path, the port number assigned by address sharing may be lost. Therefore the mapping information conveyed in the ICMP may not be sufficient to associate a SYN packet with a received ICMP.
- o The proposal is not compatible with the presence of cascaded NAT.
- o The ICMP proposal will add a traffic overhead for both the server and the address sharing device.
- o The ICMP proposal is similar to other mechanisms (e.g., syslog, netflow) for reporting dynamic mappings to a mediation platform (mainly for legal traceability purposes). Performance degradation are likely to be experienced by address sharing functions because ICMP messages are to be sent in particular for each new instantiated mapping (and also even if the mapping exists).
- o In some scenarios (e.g., Fixed-Mobile Convergence, Open WiFi, etc.), HOST_ID should be interpreted by intermediate devices which embed Policy Enforcement Points (PEP, [RFC2753]) responsible for

granting access to some services. These PEPs need to inspect all received packets in order to find the companion (traffic) messages to be correlated with ICMP messages conveying HOST_IDs. This induces more complexity to these intermediate devices.

A.5. Define a TCP Option

A.5.1. Description

HOST_ID may be conveyed in a dedicated TCP Option. An example is specified in [I-D.wing-nat-reveal-option] which defines a new TCP Option called USER_HINT. This option encloses the TCP client's identifier (e.g., the lower 16 bits of their IPv4 address, their VLAN ID, VRF ID, subscriber ID). The address sharing device inserts this TCP Option into the TCP SYN packet.

A.5.2. Analysis

Using a new TCP Option to convey the HOST_ID does not require any modification to the applications but it is applicable only for TCP-based applications. Applications relying on other transport protocols are therefore left unsolved.

[I-D.wing-nat-reveal-option] discusses the interference with other TCP Options.

The risk related to handling a new TCP Option is low as measured in [Options]. [I-D.abdo-hostid-tcpopt-implementation] provides a detailed implementation and experimentation report of HOST_ID TCP Option. [I-D.abdo-hostid-tcpopt-implementation] investigated in depth the impact of activation HOST_ID in host, address sharing function and the enforcement of policies at the server side. [I-D.abdo-hostid-tcpopt-implementation] reports a failure ratio of 0,103% among top 100000 websites.

Some downsides have been raised against defining a TCP Option to reveal a host identity:

- o Conveying an IP address in a TCP Option may be seen as a violation of OSI layers but since IP addresses are already used for the checksum computation, this is not seen as a blocking point. Moreover, updated version of [I-D.wing-nat-reveal-option] does not allow anymore to convey an IP address (the HOST_ID is encoded in 16bits).
- o TCP Option space is limited, and might be consumed by the TCP client. Earlier versions of [I-D.wing-nat-reveal-option] discuss two approaches to sending the HOST_ID: sending the HOST_ID in the

TCP SYN (which consumes more bytes in the TCP header of the TCP SYN) and sending the HOST_ID in a TCP ACK (which consumes only two bytes in the TCP SYN). Content providers may find it more desirable to receive the HOST_ID in the TCP SYN, as that more closely preserves the HOST_ID received in the source IP address as per current practices. It is more complicated to implement sending the HOST_ID in a TCP ACK, as it can introduce MTU issues if the ACK packet also contains TCP data, or a TCP segment is lost. The latest specification of the HOST_ID TCP Option, documented at [I-D.wing-nat-reveal-option], allows only to enclose the HOST_ID in the TCP SYN packet.

- o When there are several NATs in the path, the original HOST_ID may be lost. In such case, the procedure may not be efficient.
- o Interference with current usages such as X-Forwarded-For (see Appendix A.8) should be elaborated to specify the behavior of servers when both options are used; in particular specify which information to use: the content of the TCP Option or what is conveyed in the application headers.
- o When load-balancers or proxies are in the path, this option does not allow to preserve the original source IP address and source port. Preserving such information is required for logging purposes for instance (e.g., [RFC6302]). [I-D.abdo-hostid-tcpopt-implementation] defines a TCP Option which allows to reveal various combinations of source information (e.g., source port, source port and source IP address, source IPv6 prefix, etc.).

More discussion about issues raised when extending TCP can be found at [ExtendTCP].

A.6. PROXY Protocol

A.6.1. Description

The solution, referred to as Proxy Protocol [Proxy], does not require any application-specific knowledge. The rationale behind this solution is to prepend each connection with a line reporting the characteristics of the other side's connection as shown in the example depicted in Figure 2:

```
PROXY TCP4 192.0.2.1 192.0.2.15 56324 443\r\n
```

Figure 2: Example of PROXY connection report

Upon receipt of a message conveying this line, the server removes the

line. The line is parsed to retrieve the transported protocol. The content of this line is recorded in logs and used to enforce policies.

A.6.2. Analysis

This solution can be deployed in a controlled environment but it can not be deployed to all access services available in the Internet. If the remote server does not support the Proxy Protocol, the session will fail. Other complications will raise due to the presence of firewalls for instance.

As a consequence, this solution is broken and can not be recommended.

A.7. Host Identity Protocol (HIP)

A.7.1. Description

[RFC5201] specifies an architecture which introduces a new namespace to convey an identity information.

A.7.2. Analysis

This solution requires both the client and the server to support HIP [RFC5201]. Additional architectural considerations are to be taken into account such as the key exchanges, etc.

If the address sharing function is required to act as a UDP/TCP-HIP relay, this is not a viable option.

A.8. Inject Application Headers

A.8.1. Description

Another option is to not require any change at the transport nor the IP levels but to convey at the application payload the required information which will be used to disambiguate hosts. This format and the related semantics depend on its application (e.g., HTTP, SIP, SMTP, etc.).

For HTTP, the X-Forwarded-For (XFF) or Forwarded-For ([I-D.ietf-appsawg-http-forwarded]) headers can be used to display the original IP address when an address sharing device is involved. Service Providers operating address sharing devices can enable the feature of injecting the XFF header which will enclose the original IPv4 address or the IPv6 prefix part (see the example shown in Figure 3). The address sharing device has to strip all included XFF headers before injecting their own. Servers may rely on the contents

of this field to enforce some policies such as blacklisting misbehaving users. Note that XFF can also be logged by some servers (this is for instance supported by Apache).

```
Forwarded: for=192.0.2.1,for=[2001:db8::1]  
Forwarded: proto=https;by=192.0.2.15
```

Figure 3: Example of Forwarded-For

A.8.2. Analysis

Not all applications impacted by the address sharing can support the ability to disclose the original IP address. Only a subset of protocols (e.g., HTTP) can rely on this solution.

For the HTTP case, to prevent users injecting invalid HOST_IDs, an initiative has been launched to maintain a list of trusted ISPs using XFF: See for example the list available at: [Trusted_ISPs] of trusted ISPs as maintained by Wikipedia. If an address sharing device is on the trusted XFF ISPs list, users editing Wikipedia located behind the address sharing device will appear to be editing from their "original" IP address and not from the NATed IP address. If an offending activity is detected, individual hosts can be blacklisted instead of all hosts sharing the same IP address.

XFF header injection is a common practice of load balancers. When a load balancer is in the path, the original content of any included XFF header should not be stripped. Otherwise the information about the "origin" IP address will be lost.

When several address sharing devices are crossed, XFF header can convey the list of IP addresses (e.g., Figure 3). The origin HOST_ID can be exposed to the target server.

XFF also introduces some implementation complexity if the HTTP packet is at or close to the MTU size.

It has been reported that some "poor" implementation may encounter some parsing issues when injecting XFF header.

For encrypted HTTP traffic, injecting XFF header may be broken.

Authors' Addresses

Mohamed Boucadair
France Telecom
Rennes, 35000
France

Email: mohamed.boucadair@orange.com

Joe Touch
USC/ISI

Email: touch@isi.edu

Pierre Levis
France Telecom
Caen, 14000
France

Email: pierre.levis@orange.com

Reinaldo Penno
Cisco
USA

Email: repenno@cisco.com

INTAREA Working Group
Internet Draft
Intended status: Proposed Standard
Expires: December 2012

Youval Nachum
Marvell
Linda Dunbar
Huawei
Tal Mizrahi
Ilan Yerushalmi
Marvell
June 4, 2012

Scaling the Address Resolution Protocol for Large Data Centers
(SARP)
draft-nachum-sarp-02.txt

Abstract

This document provides a recommended architecture and network operation named SARP. SARP is based on fast proxies that significantly reduce broadcast domains and ARP/ND broadcast transmissions. SARP supports smooth and fast virtual machine (VM) mobility without any modification to the VM, while keeping the connection up and running efficiently. SARP is targeted for massive scaling data centers with a significant number of VMs using ARP and ND protocols.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on December 4, 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
1.1. SARP Motivation.....	3
1.2. SARP Overview	3
1.3. SARP Deployment Options	4
2. Terms and Abbreviations Used in this Document	5
3. SARP Description	6
3.1. Control Plane: ARP/ND	6
3.1.1. ARP/ND Request for a Local VM	6
3.1.2. ARP/ND Request for a Remote VM	6
3.2. Data Plane: Packet Transmission	7
3.2.1. Local Packet Transmission	7
3.2.2. Packet Transmission Between Sites	7
3.3. VM Migration	7
3.3.1. VM Local Migration	8
3.3.2. VM Migration from One Site to Another	8
3.3.2.1. Impact to ARP/ND Table of VMs being moved	9
3.4. Multicast and Broadcast	10
3.5. Non IP packet	10
3.6. ARP caching	10
3.7. High availability and load balancing	11
3.8. SARP Interaction with Overlay networks	12
4. Conclusions	12
5. Security Considerations	13
6. IANA Considerations	13
7. References	13
7.1. Normative References	13
7.2. Informative References	13
8. Acknowledgments	14

1. Introduction

1.1. SARP Motivation

SARP provides operational recommendations for network in data center(s) with a large number of virtual Machines which can migrate from one location to another without changing their IP/MAC addresses. [ARMD] has documented various impacts and scaling issues to data center networks, especially to L2/L3 boundary routers, when large number of VMs can move without changing their MAC/IP addresses.

1.2. SARP Overview

SARP is a type of proxies that constrain the ARP/ND broadcast/multicast messages to small segments regardless how wide their corresponding Layer 2 domain spread.

In order to enable VMs to be moved across greater number of servers while maintaining their MAC/IP addresses unchanged, the layer-2 network (e.g. VLAN) which interconnect those VMs may spread across different server racks, different rows of server racks, or even different data centers.

For ease of description, let's break the entire network which interconnects all those VMs into two segments: interconnecting segment and "access" segments. While the "Access" network is mostly likely Layer 2, the "interconnecting" segment might be not.

The SARP proxies are located at the boundaries where the "Access" segment connects to its "Interconnecting" segment. The boundary node could be a Hypervisor virtual switch, a Top of Rack switch, an Aggregation switch (or end of row switch), or a data center core switch. Figure 1 depicts an example of two remote data centers that are managed as a single flat Layer 2 domain. SARP proxies are implemented at the edge devices connecting the data center to the transport network. SARP significantly reduces the ARP/ND transmissions over the "interconnection" network. The ARP/ND broadcast/multicast messages are bounded by the SARP proxies.

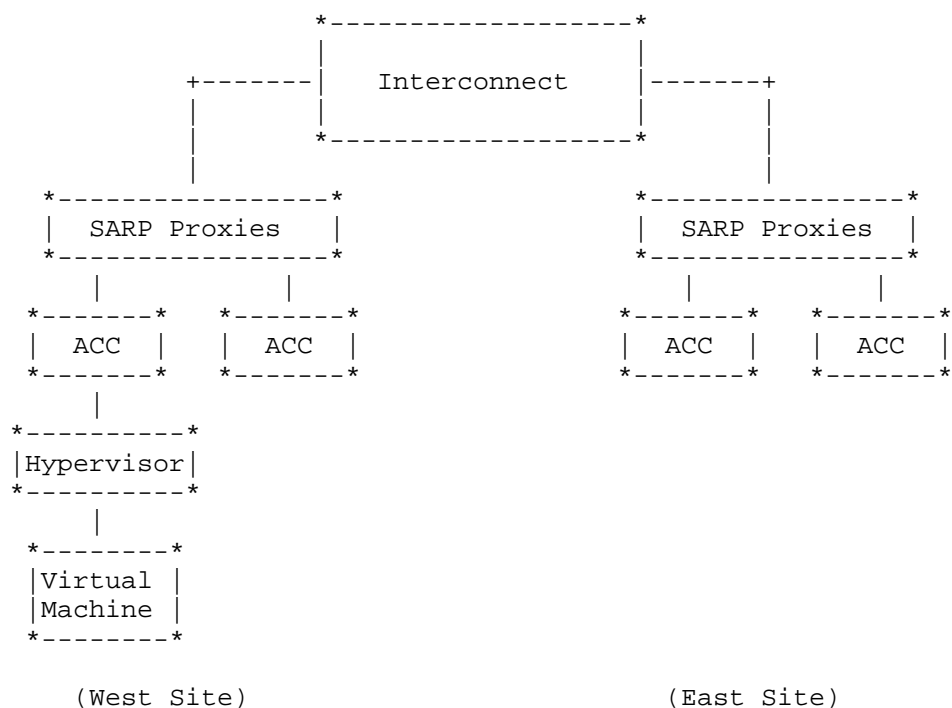


Figure 1 SARP Networking Architecture Example.

1.3. SARP Deployment Options

SARP deployment is tightly coupled with the data center architecture. SARP proxies are located at the point where the Layer 2 infrastructure connects to its Layer 2 cloud using overlay networks. SARP proxies can be located at the data center edge (as Figure 1 depicts), data center core, or data center aggregation. SARP can also be implemented by the hypervisor (as Figure 2 depicts).

To simplify the description, we will focus on data centers that are managed as a single flat Layer 2 network, where SARP proxies are located at the boundary where the data center connects to the transport network (as Figure 1 depicts).

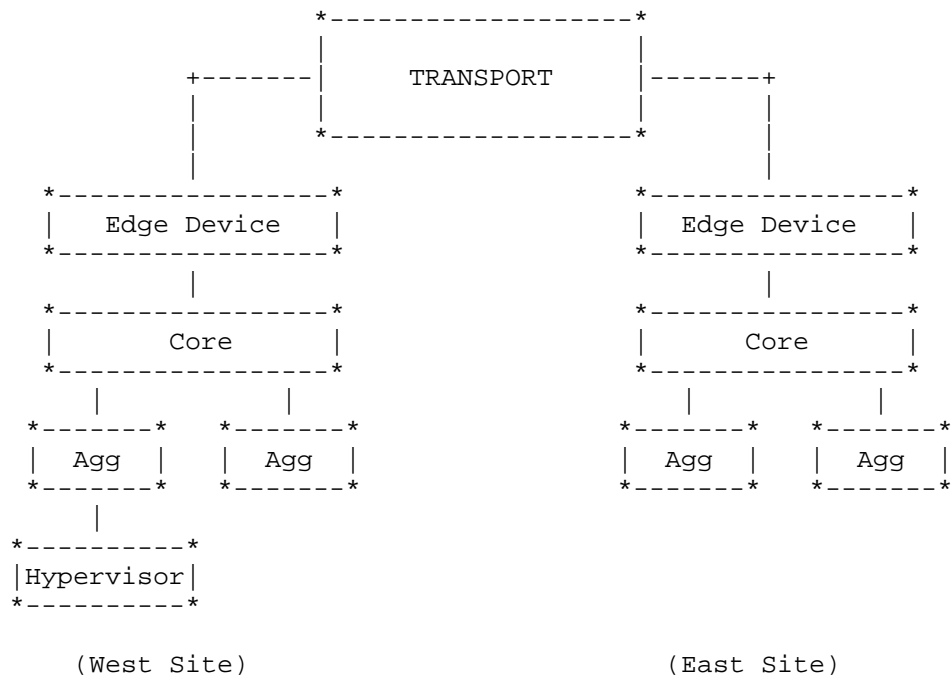


Figure 2 SARP deployment options.

2. Terms and Abbreviations Used in this Document

ARP: Address Resolution Protocol

FIB: Forwarding Information Base

IP-D: IP address of the destination virtual machine

IP-S: IP address of the source virtual machine

MAC-D: MAC address of the destination virtual machine

MAC-E: MAC address of the East Proxy SARP Device

MAC-S: MAC address of the source virtual machine

ND: Neighbor Discovery

SARP Proxy: The components that participates in the SARP protocol.

VM: Virtual Machine

3. SARP Description

3.1. Control Plane: ARP/ND

This section describes the ARP/ND procedure scenarios. In the first scenario, VMs share the same Access Segment. In the second scenario, the source VM is local Access Segment and the destination VM is located at the remote Access Segment.

In all scenarios, the VMs (source and destination) share the same L2 broadcast domain.

3.1.1. ARP/ND Request for a Local VM

When source and destination VMs are located at the same Access Segment, the Address Resolution process is as described in [ARP] and [ND]. When the VM sends an ARP request to learn the IP to MAC mapping of another local VM, it receives a reply from the other local VM with the IP-D to MAC-D mapping.

3.1.2. ARP/ND Request for a Remote VM

When the source and destination VMs are located at different Access Segments, the Address Resolution process is as follows.

In our example, the source VM is located at the west Access Segment and the destination VM is located at the east Access Segment.

When the source VM sends an ARP/ND request to find out the IP to MAC mapping of a remote VM, if the local SARP proxy doesn't have the ARP cache for the target IP address or the cache entry has expired, the ARP request is propagated to all Access Segments which might have VMs in the same virtual network as the originating VM, including the east Access Segment.

The destination VM responds to the ARP/ND request and transmits an ARP/ND reply having the IP-D to MAC-D mapping.

The east SARP proxy functions as the proxy ARP of its Local VMs. The east SARP proxy modifies the ARP reply message to be MAC-D to MAC-E and forwards the modified ARP reply message to all the SARP proxies.

The West SARP Proxy forwards the modified ARP reply message to the source VM.

The west SARP proxy can also function as an ARP cache of the Remote VMs. By doing so, it significantly reduces the volume of the ARP/ND transmission over the network.

3.2. Data Plane: Packet Transmission

3.2.1. Local Packet Transmission

When a VM transmits packets to a destination VM that is located at the same site, there is no change in the data plane. The packets are sent from (IP-S, MAC-S) to (IP-D, MAC-D).

3.2.2. Packet Transmission Between Sites

Packets that are sent between sites traverse the SARP proxy of both sites. In our example, all packets sent from the VM located at the west site to the destination VM located at the east site traverse the west SARP proxy and the east SARP proxy.

The source VM follows its ARP table and sends packets to (IP-D, MAC-E) destination addresses and with (IP-s, MAC-S) as the source addresses.

The west SARP proxy replaces the packet source address to its own source address (MAC-W), keeps the destination address to be (MAC-E), and forwards the packet to the east proxy SARP.

When the east proxy SARP receives the packet, it replaces the destination MAC address to be (MAC-D) based on the packet destination IP (i.e., IP-D), but it does not change the source MAC addresses. When the destination VM receives the packet, the Source Address field would be the MAC address of the west side SARP proxy,

Noted: it is common for data center network to have security policies to enforce some VMs can communicate with each other, and some VMs can't. Most likely, those policies are configured by VM's IP addresses. Even though the originating VM's MAC address is lost when the packet arrives at the destination VM, the originating VM's IP address is still present in the data packets for security policy to be enforced.

3.3. VM Migration

3.3.1. VM Local Migration

When a VM migrates locally within its Access segment, the SARP protocol is not required to perform any action. VM migration is resolved entirely by the Layer 2 mechanisms.

3.3.2. VM Migration from One Site to Another

In our example, the VM migrates from the west site to the east site while maintaining its MAC and IP addresses.

VM migration might affect networking elements based on their respective location:

- Origin site (west site)
- Destination site (east site)
- Other sites

Origin site:

The Origin site is the site where the VM is before migration. It is the west site in our example.

Before the VM (IP=IP-D, MAC=MAC-D) is moved, all VMs at the west site that have an ARP entry of IP-D in their ARP table have the (IP-D to MAC-D) mapping. VMs on any other "Access Segments" will have ARP entry of (IP-D to MAC-W) mapping where MAC-W is the MAC address of the SARP proxy on the West Access Segment.

After the VM (IP-D) in the West Site moves to East Site, if there is gratuitous ARP sent out by the destination hypervisor for the VM (IP-D), then the ARP mapping on VMs on all Access Segments will be updated by (IP-D to MAC-E) where MAC-E is the MAC address of the SARP proxy on the East Site. If there isn't any gratuitous ARP sent out by the destination hypervisor, the ARP mapping on the VMs in west site (and other sites) will eventually aged out.

Until ARP tables are updated, the source VMs from the west site continue sending packets to MAC-D. Switches at the west site are still configured with the old location of MAC-D. This can be resolved by VM manager sending out a fake gratuitous ARP on behalf of destination Hypervisor, MAC table aging or by redirecting the packets to the proxy SARP of the west site.

Destination Site:

The destination site is the site to which the VM migrated, the east site in our example.

Before any gratuitous ARP messages are sent out by the destination hypervisor, all VMs at the east site (and all other sites) that have an ARP entry of IP-D in their ARP table have the (IP-D to MAC-W) mapping. ARP mapping is updated by aging or by a gratuitous ARP message sent by the destination hypervisor. Until ARP tables are updated, the source VMs from the east site continue to send packets to MAC-W. This can be resolved by VM manager sending out a fake gratuitous ARP immediately after the VM migration, or redirecting the packets from the SARP proxy of the east site to the migrated VM by updating the destination MAC of the packets to MAC-D.

Other Sites:

All VMs at the other sites that have an ARP entry of IP-D in their ARP table have the (IP-D to MAC-W) mapping. ARP mapping is updated by aging or by a gratuitous ARP message sent by the destination hypervisor of the migrated VM and modified by the SARP proxy of the east site (IP-D to MAC-E) mapping. Until ARP tables are updated, the source VMs from the west site continue sending packets to MAC-W. This can be resolved by redirecting the packets from the SARP proxy of the west site to the SARP proxy of the east site by updating the destination MAC of the packets to MAC-E.

3.3.2.1. Impact to ARP/ND Table of VMs being moved

When a VM (IP-D) is moved from one site to another site, its ARP entries for VMs located at the other sites (i.e. neither east site nor west site) are unaffected by the migration.

The VM (IP-D)'s ARP entries (i.e. IP to MAC mapping) for VMs located at east site can be kept with no change until the ARP aging time since they are mapped to MAC-E. All traffic originated from the VM (IP-D) in its new location to VMs located at the east site traverses the SARP proxy of the east Site. This can be mitigated by ARP advertisement sent by the SARP proxy of the east site or by the hypervisor.

The VM (IP-D)'s ARP entries (i.e. IP to MAC mapping) for VMs located at west sites will not be changed either until the ARP entries age out or new data frames are received from the remote sites. Since all MAC addresses of the VMs located at the west site are unknown at the east site. All unknown traffic from the VM is intercepted by the SARP proxy of the east site and forwarded to the SARP proxy of the west site (just for ARP aging time). This can be resolved by the east SARP

proxy having mapping entries for VMs in the west side. Upon receiving unknown packets, it can update the migrating VM with the new IP to MAC mapping by sending a modified gratuitous ARP with (IP-D to MAC-W) mapping.

Note that overlay networks providing the Layer 2 network virtualization services configure their Edge Device MAC aging timers to be greater than the ARP request interval.

3.4. Multicast and Broadcast

To be added in a future version of this document

3.5. Non IP packet

To be added in a future version of this document

3.6. ARP caching

ARP/ND Requests for a VM located at a remote site require flooding messages over the interconnecting network to all sites which have enabled the virtual network on which the VM belongs to. This scenario is described in details at 3.1.2. In such cases, SARP caching can reduce the number of ARP/ND transmissions over interconnecting networks.

In the example presented at section 3.1.2. the source VM is located at the west site and the destination VM is located at the east site. When the source VM sends an ARP/ND request to discover the IP to MAC mapping of the remote VM, the ARP/ND request can be intercepted by the west SARP proxy.

The west SARP proxy learns or refreshes the source IP to source MAC mapping and looks up the IP to MAC translation of the destination IP. If the destination IP entry is found and is valid, the west SARP proxy replies with an ARP/ND reply without propagating the packet to other sites. Otherwise, the packet is propagated to all sites which have the virtual network enabled including the east site.

The propagated ARP/ND request is intercepted again by the east SARP proxy. It learns or refreshes the source IP to source MAC mapping and looks up the destination IP to MAC translation. If the destination IP entry is found and is valid the SARP proxy replies with an ARP/ND reply without propagating the ARP request to the east site. Otherwise, the ARP/ND request is broadcasted within the east site.

The destination VM responds to the ARP/ND request and transmits an ARP/ND reply having the IP-D to MAC-D mapping.

The east side SARP proxy intercepts the ARP/ND reply and learns or refreshes the Destination IP to Destination MAC mapping and propagates the ARP/ND reply to the west SARP proxy.

The West SARP Proxy intercepts the ARP/ND reply and learns or refreshes the Destination IP to Destination MAC mapping and propagates the ARP reply to the source VM.

The SARP proxies maintain an ARP caching table of IP to MAC mapping for all recent ARP/ND requests and replies. This table allows the SARP proxy to respond with low latency to the ARP/ND requests sent locally and avoid the broadcast transmissions of such requests over the transport network and all over the broadcast domains at the remote sites.

3.7. High availability and load balancing

The SARP proxy is located at the boundary where the local Layer 2 infrastructure connects to the interconnecting network. All traffic from the local site to the remote sites traverses the SARP proxy. The SARP proxy is subject to high availability and bandwidth requirements.

The SARP architecture supports multiple SARP proxies connecting a single site to the transport network. In SARP architecture all proxies can be active and can backup one another. The SARP architecture is robust and allows the network administrator to allocate proxies according to the bandwidth and high availability requirements.

Traffic is segregated between SARP proxies by using VLANs. An SARP proxy is the Master-SARP proxy of a set of VLANs and the Backup-SARP proxy of another set of VLANs.

For example the SARP proxies of the west site (as Figure 1 depicts) are SARP proxy-1 and SARP proxy-2. The west site supports VLAN-1 and VLAN-2 while SARP proxy-1 is the Master SARP proxy of VLAN-1 and the Backup proxy of VLAN-2 and SARP proxy-2 is the Master SARP proxy of VLAN-2 and the Backup SARP proxy of VLAN-1. Both proxies are members of VLAN-1 and VLAN-2.

The Master SARP proxy updates its Backup proxy with all the ARP reply messages. The Backup SARP proxy maintains a backup database to all the VLANs that it is the Backup SARP proxy.

The Master and the Backup SARP proxies maintain a keepalive mechanism. In case of a failure the Backup proxy becomes the Master SARP proxy. The failure decision is per VLAN. When the Master and the Backup proxies switchover, the backup SARP proxy can use the MAC address of the Master SARP proxy. The backup SARP proxy sends locally a gratuitous ARP message with the MAC address of the Master SARP proxy to update the forwarding tables on the local switches. The backup SARP proxy also updates the remote SARP proxies on the change.

3.8. SARP Interaction with Overlay networks

SARP interaction with overlay networks providing L2 network virtualization (such as IP, VPLS, Trill, OTV, NVGRE and VxLAN) is efficient and scalable.

The mapping of SARP to overlay networks is straightforward. The VM does the destination IP to SARP proxy MAC mapping. The mapping of the proxy MAC to its correct tunnel is done by the overlay networks. SARP significantly scales down the complexity of the overlay networks and transport networks by reducing the mapping tables to the number of SARP proxies.

4. Conclusions

SARP distributes the Layer 2 Forwarding Information Base (FIB) from the edge devices (functioning as SARP proxies) to the VMs. By doing so, it significantly reduces table sizes on the edge devices. The source VM maintains the mapping of its destination VMs to the destination site/cloud in the ARP table. The destination VM IP is translated to the destination MAC address of the SARP proxy at the destination site. The SARP proxies only maintain Layer 2 FIB of local VMs and remote edge devices.

SARP proxies can support FAST VM migration and provide minimum transition phase. When SARP proxy indicates or is informed of VM migration, it can update all its peers and trigger a fast update.

SARP seamlessly supports Layer 2 network virtualization services over the overlay network and significantly reduces their complexity in terms of table size and performance. The overlay networks are only required to map MAC addresses of the SARP proxies to the correct tunnel.

5. Security Considerations

The SARP proxies are located at the boundaries where the local Layer 2 infrastructure connects to its Layer 2 cloud. The SARP proxies interoperate with overlay network protocols that extend the Layer-2 subnet across data centers or between different systems within a datacenter.

SARP control plane and data plane are traversed by the overlay network hence SARP does not expose the network to additional security threats.

SARP proxies may be exposed to Denial of Service (DoS) attacks by means of ARP/ND message flooding. Thus, the SARP proxies must have sufficient resources to support the SARP control plane without making the network more vulnerable to DoS than without SARP proxies.

SARP adds security to the data plane by hiding all the local layer 2 MAC addresses from potential attacker located at the remote clouds. The only MAC addresses that are exposed at remote sites are the MAC addresses of the SARP proxies.

6. IANA Considerations

There are no IANA actions required by this document.

RFC Editor: please delete this section before publication.

7. References

7.1. Normative References

- [ARP] Plummer, D., "An Ethernet Address Resolution Protocol", RFC 826, November 1982.
- [ND] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

7.2. Informative References

- [ARMD] Narten, T., Karir, M., Foo, I., " Problem Statement for ARMD", draft-ietf-armd-problem-statement, February 2012.

8. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Youval Nachum
Marvell
6 Hamada St.
Yokneam, 20692 Israel
Email: youvaln@marvell.com

Linda Dunbar
Huawei Technologies
5430 Legacy Drive, Suite #175
Plano, TX 75024, USA
Phone: (469) 277 5840
Email: ldunbar@huawei.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam, 20692 Israel
Email: talmi@marvell.com

Ilan Yerushalmi
Marvell
6 Hamada St.
Yokneam, 20692 Israel
Email: yilan@marvell.com

