IPv6 Operations
Internet-Draft
Updates: 6145 (if approved)
Intended status: Standards Track
Expires: July 12, 2015

T. Anderson
Redpill Linpro

January 08, 2015

Explicit Address Mappings for Stateless IP/ICMP Translation
draft-anderson-v6ops-siit-eam-03

Abstract

This document extends the Stateless IP/ICMP Translation Algorithm (SIIT) with an Explicit Address Mapping (EAM) algorithm, and formally updates RFC 6145. The EAM algorithm facilitates stateless IP/ICMP translation between arbitrary (non-IPv4-translatable) IPv6 endpoints and IPv4.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 12, 2015.

Table of Contents

1. Introduction

The Stateless IP/ICMP Translation Algorithm (SIIT) [RFC6145] specifies that when translating IPv4 addresses to IPv6 and vice versa, all addresses must be translated using the algorithm specified in [RFC6052]. This document specifies an alternative to the [RFC6052] algorithm, where IP addresses are translated according to a table of Explicit Address Mappings configured on the stateless translator. This removes the previous constraint that IPv6 nodes that communicate with IPv4 nodes through SIIT must be configured with IPv4-translatable IPv6 addresses.

The Explicit Address Mapping Table does not replace [RFC6052]. For most use cases, it is expected that both algorithms are used in concert. The Explicit Address Mapping algorithm is used only when a mapping matching the address to be translated exists. If no matching mapping exists, the [RFC6052] algorithm will be used instead. Thus, when translating an individual IP packet, an SIIT implementation might translate one of the two IP address fields according to an EAM, while the other IP address field is translated according to [RFC6052].

1.1. Terminology

This document makes use of the following terms:

EAM
: An Explicit Address Mapping, as specified in Section 3.2.

EAMT
: The Explicit Address Mapping Table, as specified in Section 3.1.

SIIT
: The Stateless IP/ICMP Translation algorithm, as specified in [RFC6145].

IPv4-converted IPv6 addresses
: As defined in Section 1.3 of [RFC6052].

IPv4-translatable IPv6 addresses
: As defined in Section 1.3 of [RFC6052].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Problem Statement

Section 3.2.1 of [RFC6144] notes that "stateless translation mechanisms typically put constraints on what IPv6 addresses can be assigned to IPv6 nodes that want to communicate with IPv4 destinations using an algorithmic mapping". In practice, this means that the IPv6 nodes must be configured with IPv4-translatable IPv6 addresses. For the reasons discussed below, some environments may find that the use of IPv4-translatable IPv6 addresses is not desired or even possible.

Limited availability:
: The number of IPv4-translatable IPv6 addresses available to an operator is equal to the number of IPv4 addresses he assigns to

the SIIT function. IPv4 addresses are scarce, and as a result an operator might not have enough IPv4-translatable IPv6 addresses to number his entire IPv6 infrastructure.

Restricted format:
IPv4-translatable IPv6 addresses must conform to the format specified in Section 2.2 of [RFC6052]. This format is not compatible with other common IPv6 address formats, such as the EUI-64 based IPv6 address format used by IPv6 Stateless Address Autoconfiguration [RFC4862].

An operator could overcome the above two problems by building an IPv6 network using regular (non-IPv4-translatable) IPv6 addresses, and assign IPv4-translatable IPv6 addresses as secondary addresses on the nodes that want to communicate with IPv4 nodes through SIIT only. However, doing so may result in a new set of undesired properties:

Routing complexity:
The IPv4-translatable IPv6 addresses must be routed throughout the IPv6 network separately from the primary (non-IPv4-translatable) IPv6 addresses used by the nodes. It might be impossible to aggregate these routes, as two adjacent IPv4-translatable IPv6 addresses might not be assigned to two adjacent IPv6 nodes. As a result, in order to support SIIT, the IPv6 network might need to carry a large number of extraneous routes. These routes must be separately injected into the IPv6 routing topology somehow. Any intermediate devices in the IPv6 network such as a firewall might require special configuration in order to treat the IPv4-translatable IPv6 address the same as the primary IPv6 address, for example by requiring that any ACL entries involving the primary IPv6 address of a node must be duplicated.

Operational complexity:

The IPv4-translatable IPv6 addresses must not only be assigned to the IPv6 nodes participating in SIIT; all applications and services on those nodes must also be configured to use them. For example, if the IPv6 node is a load balancer, it might require a separate Virtual Server definition using the IPv4-translatable IPv6 address in addition to one using the service's primary IPv6 address. A web server might require specific configuration to listen for connections on both the IPv4-translatable and the primary IPv6 address. A High-Availability cluster service must be set up to fail over both addresses between cluster nodes, and depending on how the IPv6 network learns the location of the IPv4-translatable IPv6 address, the fail-over mechanism used for the two addresses might be completely different. Service monitoring must be done for both the IPv4-translatable and the primary IPv6 address, and any trouble-shooting procedures must be extended to involve both addresses.

In short, the use of IPv4-translatable IPv6 addresses in parallel with regular IPv6 addresses is in many ways analogous to the use of Dual Stack [RFC4213]. While no actual IPv4 packets are used, the IPv4-translatable IPv6 addresses creates a secondary "stack" in the infrastructure that must be treated and operated separately from the primary one. This increases the complexity of the overall infrastructure, in turn increasing operational overhead, and reducing reliability. An operator who for such reasons finds the use Dual Stack unappealing, might feel the same way about using SIIT with IPv4-translatable IPv6 addresses.

3. Explicit Address Mapping Algorithm

This normative section defines the EAM algorithm. SIIT implementations are REQUIRED to support the specifications herein.

3.1. Explicit Address Mapping Table

An SIIT implementation MUST include an Explicit Address Mapping Table (EAMT). By default, the EAMT SHOULD be empty. The operator MUST be able to populate the EAMT using the implementation's normal configuration interfaces. The implementation MAY additionally support other ways of populating the EAMT.

The EAMT consists of the following columns:

IPv4 Prefix

IPv6 Prefix

SIIT implementations MAY include other columns in order to support proprietary extensions to the EAM algorithm.

Throughout this document, figures representing the EAMT contain an Index column using the pound sign as the header. This column is not a required part of this specification; it is included only as a convenience to the reader.

3.2. Explicit Address Mapping Specification

An EAM consists of an IPv4 Prefix and an IPv6 Prefix. The prefix length MAY be omitted, in which case the implementation MUST assume it to be 32 for IPv4 and 128 for IPv6. Figure 1 illustrates an EAMT containing examples of valid EAMs.

Example EAMT

#	IPv4 Prefix	IPv6 Prefix
1	192.0.2.1	2001:db8:aaaa::
2	192.0.2.2/32	2001:db8:bbbb::b/128
3	192.0.2.16/28	2001:db8:cccc::/124
4	192.0.2.128/26	2001:db8:dddd::/64
5	192.0.2.192/31	64:ff9b::/127

Figure 1

An EAM's IPv4 Prefix value MUST have an identical or smaller number of suffix bits than its corresponding IPv6 Prefix value.

Overlapping EAMs SHOULD be considered an error, and attempts to insert them into the EAMT SHOULD be blocked. The behaviour of an SIIT implementation when overlapping EAMs are present in the EAMT is left undefined.

When translating a packet between IPv4 and IPv6, an SIIT implementation MUST individually translate each IP address it encounters in the packet's IP headers (including any IP headers contained within ICMP errors) according to Section 3.3.

3.3. IP Address Translation Procedure

This section describes step-by-step how an SIIT implementation translates addresses between IPv4 and IPv6. Only the outcome of the algorithm described should be considered normative, that is, an SIIT implementation MAY implement the exact procedure differently than

what is described here, but the outcome of the algorithm MUST be the same.

For concrete examples of IP addresses translations, refer to Appendix B.

3.3.1. Address Translation Steps: IPv4 to IPv6

1. The EAMT is searched for an EAM entry containing an IPv4 Prefix identical to that of the IPv4 address being translated. The IPv4 Prefix and IPv6 Prefix values of the EAM entry found is from now on referred to as EAM4 and EAM6, respectively.

2. If no matching EAM entry is found, the EAM algorithm is aborted. The SIIT implementation MUST proceed to translate the address in accordance with [RFC6145] (and its updates).

3. The prefix bits of EAM4 are removed from IPv4 address being translated. The remaining suffix bits from the IPv4 address being translated are stored in a temporary buffer.

4. The prefix bits of EAM6 are prepended to the temporary buffer.

5. If the temporary buffer at this point does not contain a 128-bit value, it is padded with trailing zeroes so that it reaches a length of 128 bits.

6. The contents of the temporary buffer is the translated IPv6 address.

3.3.2. Address Translation Steps: IPv6 to IPv4

1. The EAMT is searched for an EAM entry containing an IPv6 Prefix identical to that of the IPv6 address being translated. The IPv4 Prefix and IPv6 Prefix values of the EAM entry found is from now on referred to as EAM4 and EAM6, respectively.

2. If no matching EAM entry is found, the EAM algorithm is aborted. The SIIT implementation MUST proceed to translate the address in accordance with [RFC6145] (and its updates).

3. The prefix bits of EAM6 are removed from IPv6 address being translated. The remaining suffix bits from the IPv6 address being translated are stored in a temporary buffer.

4. The prefix bits of EAM4 are prepended to the temporary buffer.

5. If the temporary buffer at this point does not contain a 32-bit value, any trailing bits are discarded so that the buffer is reduced to a length of 32 bits.

6. The contents of the temporary buffer is the translated IPv4 address.

4. Lack of Checksum Neutrality

When one or both of the address fields in an IP/ICMP packet are translated according to EAM, the translation can not be relied upon to be checksum neutral, even if the well-known prefix 64:ff9b::/96 is used. This consideration is discussed in more detail in Section 4.1 of [RFC6052].

5. Security Considerations

The EAM algorithm does not introduce any new security issues beyond those that are already discussed in Section 7 of [RFC6145].

6. IANA Considerations

This draft makes no request of the IANA. The RFC Editor may remove this section prior to publication.

7. Acknowledgements

This document was conceived due to comments made by Dave Thaler in the v6ops session at IETF 91 as well as e-mail discussions between Fred Baker and the author.

Valuable reviews, suggestions, and other feedback was given by Cameron Byrne, Brian E Carpenter, Alberto Leiva, and Andrew Yourtchenko.

8. References

8.1. Normative References

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

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

8.2. Informative References

[I-D.anderson-v6ops-siit-dc]
tore, t., "SIIT-DC: Stateless IP/ICMP Translation for IPv6 Data Centre Environments", draft-anderson-v6ops-siit-dc-01 (work in progress), October 2014.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

[RFC6219] Li, X., Bao, C., Chen, M., Zhang, H., and J. Wu, "The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition", RFC 6219, May 2011.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, April 2013.

[RFC7335] Byrne, C., "IPv4 Service Continuity Prefix", RFC 7335, August 2014.

Appendix A. Use Cases

The following subsections lists some use cases that at the time of writing leverage SIIT with the EAM algorithm.

A.1. 464XLAT

When the CLAT component in the 464XLAT [RFC6877] architecture does not have a dedicated IPv6 prefix assigned, it may instead use "one interface IPv6 address that is claimed by the CLAT". This IPv6 address might not be IPv4-translatable. If this is the case, the CLAT essentially implements the EAM algorithm using an EAMT as follows (assuming the CLAT’s IPv4 address is picked from the IPv4 Service Continuity Prefix [RFC7335]):

Example EAMT for an 464XLAT CLAT

#	IPv4 Prefix	IPv6 Prefix
1	192.0.0.1/32	CLAT_claimed_IPv6_address/128

Figure 2

In this particular use case, the EAM algorithm is used to translate IPv6 destination addresses to IPv4, and conversely, IPv4 source addresses to IPv6. Other addresses are translated using [RFC6052]. Note that this is the exact opposite of the SIIT-DC use case (Appendix A.3).

A.2. IVI

IVI [RFC6219] describes a stateless translation model that embeds IPv4 addresses in a 40-bit translation prefix where bits 33-40 are required to be 1. The embedded IPv4 address is located in bits 41-72 of the IPv6 address. Bits 73-128 are required to be 0.

The location of the eight least significant IPv4 address bits makes the IVI address mapping differ from [RFC6052].

Example EAMT for IVI

#	IPv4 Prefix	IPv6 Prefix
1	0.0.0.0/0	2001:db8:ff00::/40

Figure 3

In this particular use case, all addresses are translated according to the EAM algorithm. In other words, [RFC6052] mapping is not used at all.

A.3. SIIT-DC

SIIT-DC [I-D.anderson-v6ops-siit-dc] describes the use of SIIT to facilitate connectivity from the IPv4 Internet to services hosted in an IPv6-only data centre. In order to avoid the constraints relating to the use of IPv4-translatable IPv6 addresses discussed in Section 2 the stateless IPv4/IPv6 translators are provisioned with an EAMT containing one entry per IPv6-only service that are to be made available from the IPv4 Internet, for example (assuming 2001:db8:aaaa::1 and 2001:db8:bbbb::1 are assigned to load balancers or servers that provides the IPv6-only services in question):

Example EAMT for SIIT-DC

#	IPv4 Prefix	IPv6 Prefix
1	192.0.2.1/32	2001:db8:aaaa::1/128
2	192.0.2.2/32	2001:db8:bbbb::1/128

Figure 4

In this particular use case, the EAM algorithm is used to translate IPv4 destination addresses to IPv6, and conversely, IPv6 source addresses to IPv4. Other addresses are translated using [RFC6052]. Note that this is the exact opposite of the 464XLAT use case (Appendix A.1).

Appendix B. Example IP Address Translations

Figure 5 demonstrates how a set of example IP addresses are translated given the example EAMT in Figure 1. Implementors may use the examples given to develop test cases to validate correct operation. Note that the address translations are bidirectional, so a single row in the table describes two address translations: IPv4 to IPv6, and IPv6 to IPv4.

It is also assumed that the [RFC6052] translation prefix is configured to be 64:ff9b::/96.

Example IP Address Translations

IPv4 Address	IPv6 Address	Comment
192.0.2.1	2001:db8:aaaa::	According to EAM #1
192.0.2.2	2001:db8:bbbb::b	According to EAM #2
192.0.2.16	2001:db8:cccc::	According to EAM #3
192.0.2.24	2001:db8:cccc::8	According to EAM #3
192.0.2.31	2001:db8:cccc::f	According to EAM #3
192.0.2.128	2001:db8:dddd::	According to EAM #4
192.0.2.152	2001:db8:dddd:0:6000::	According to EAM #4
192.0.2.183	2001:db8:dddd:0:dc00::	According to EAM #4
192.0.2.191	2001:db8:dddd:0:fc00::	According to EAM #4
192.0.2.193	64:ff9b::1	According to EAM #5
192.0.2.200	64:ff9b::c000:2c8	According to RFC 6052

Figure 5

Author’s Address

Tore Anderson
Redpill Linpro
Vitaminveien 1A
0485 Oslo
Norway

Phone: +47 959 31 212
Email: tore@redpill-linpro.com
URI: http://www.redpill-linpro.com

IPv6 Operations Working Group (v6ops)
Internet-Draft
Intended status: Informational
Expires: September 9, 2015

F. Gont
SI6 Networks / UTN-FRH
J. Linkova
Google
T. Chown
University of Southampton
W. Liu
Huawei Technologies

March 8, 2015

Observations on IPv6 EH Filtering in the Real World
draft-gont-v6ops-ipv6-ehs-in-real-world-02

Abstract

This document presents real-world data regarding the extent to which packets with IPv6 extension headers are filtered in the Internet (as measured in August 2014), and where in the network such filtering occurs. The aforementioned results serve as a problem statement that is expected to trigger operational advice on the filtering of IPv6 packets carrying IPv6 Extension Headers, so that the situation improves over time. This document also explains how the aforementioned results were obtained, such that the corresponding measurements can be reproduced by other members of the community.

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 September 9, 2015.

Table of Contents

1. Introduction

IPv6 Extension Headers (EHs) allow for the extension of the IPv6 protocol, and provide support for core functionality such as IPv6 fragmentation. While packets employing IPv6 Extension Headers have been suspected to be dropped in some IPv6 deployments, there was not much concrete data on the topic. Some preliminary measurements have been presented in [PMTUD-Blackholes], [Gont-IEPG88] and [Gont-Chown-IEPG89], whereas [Linkova-Gont-IEPG90] presents more comprehensive results on which this document is based.

This document presents real-world data regarding the extent to which IPv6 Extension Headers are filtered in the Internet, as measured in August 2014 (pending operational advice in this area).

2. Support of IPv6 Extension Headers in the Internet

This section summarizes the results obtained when measuring the support of IPv6 Extension Headers on the path towards different types of public IPv6 servers. Two sources were employed for the list of public IPv6 servers: the "World IPv6 Launch Day" site (http://www.worldipv6launch.org/) and Alexa's top 1M web sites (http://www.alexa.com). For each list of domain names, the following datasets were obtained:

- Web servers (AAAA records of the aforementioned list)
- Mail servers (MX -> AAAA of such list)
- Name servers (NS -> AAAA of such list)

IPv6 addresses other than global unicast addresses and duplicate addresses were eliminated from each of those lists prior to obtaining the results included in this document. Additionally, addresses that were found to be unreachable were discarded from the dataset (please see Appendix B for further details).

For each of the aforementioned address sets, three different types of probes were performed:

- IPv6 packets with a Destination Options header of 8 bytes
- IPv6 packets resulting in two IPv6 fragments of 512 bytes each (approximately)
- IPv6 packets with a Hop-by-Hop Options header of 8 bytes

In the case of packets with Destination Options Header and Hop-by-Hop Options header, the desired EH size was achieved by means of PadN options [RFC2460]. The upper-layer protocol of the probe packets was, in all cases, TCP [RFC0793] segments with the Destination Port set to the service port [IANA-PORT-NUMBERS] of the corresponding dataset. For example, the probe packets for all the measurements involving web servers were TCP segments with the destination port set to 80.

Besides obtaining the packet drop rate when employing the aforementioned IPv6 extension headers, we tried to identify whether the Autonomous System (AS) dropping the packets was the same as the Autonomous System of the destination/target address. This is of particular interest since it essentially reveals whether the packet drops are under the control of the intended destination of the packets. Packets dropped by the destination AS are less of a

concern, since the device dropping the packets is under the control of the same organization as that to which the packets are destined (hence, it is probably easier to update the filtering policy if deemed necessary). On the other hand, packets dropped by transit ASes are more of a concern, since they affect the deployability and usability of IPv6 extension headers (including IPv6 fragmentation) by a third-party (the destination AS). In any case, we note that it is impossible to tell whether, in those cases where IPv6 packets with extension headers get dropped, the packet drops are the result of an explicit and intended policy, or the result of improper device configuration defaults, buggy devices, etc. Thus, packet drops that occur at the destination AS might still prove to be problematic.

Since there is some ambiguity when identifying the autonomous system to which a specific router belongs, our measurements result in a percentage *range* (see Appendix B.2). In the following tables, the values shown within parentheses represent the estimated range of possibility that when a packet is dropped, the packet drop occurs in an AS other than the destination AS.

Dataset	DO8	HBH8	FH512
Webservers	11.88% (17.60%-20.80%)	40.70% (31.43%-40.00%)	30.51% (5.08%-6.78%)
Mailservers	17.07% (6.35%-26.98%)	48.86% (40.50%-65.42%)	39.17% (2.91%-12.73%)
Nameservers	15.37% (14.29%-33.46%)	43.25% (42.49%-72.07%)	38.55% (3.90%-13.96%)

Table 1: WIPv6LD dataset: Packet drop rate for different destination types, and estimated percentage of dropped packets that were deemed to be dropped in a different AS (lower, in parentheses)

NOTE: As an example, we note that the cell describing the support of IPv6 packets with DO8 for webservers (containing the value "11.88% (17.60%-20.80%)") should be read as: "when sending IPv6 packets with DO8 to public webservers, 11.88% of such packets get dropped. Among those packets that get dropped, between 17.60%-20.80% of them get dropped at an AS other than the destination AS".

EH Type	Webservers	Mailservers	Nameservers
DO8	11.88% (17.60%-20.80%)	17.07% (6.35%-26.98%)	15.37% (14.29%-33.46%)
HBH8	40.70% (31.43%-40.00%)	48.86% (40.50%-65.42%)	43.25% (42.49%-72.07%)
FH512	30.51% (5.08%-6.78%)	39.17% (2.91%-12.73%)	38.55% (3.90%-13.96%)

Table 2: WIPv6LD dataset: Packet drop rate for different EH types, and estimated percentage of dropped packets that were deemed to be dropped in a different AS (lower, in parentheses)

NOTE: This table contains the same information as Table 1, but makes it easier to obtain the drop rates for each EH type. Each cell should be read in exactly the same way as each cell in Table 1.

Dataset	DO8	HBH8	FH512
Webservers	10.91% (46.52%-53.23%)	39.03% (36.90%-46.35%)	28.26% (53.64%-61.43%)
Mailservers	11.54% (2.41%-21.08%)	45.45% (41.27%-61.13%)	35.68% (3.15%-10.92%)
Nameservers	21.33% (10.27%-56.80%)	54.12% (50.64%-81.00%)	55.23% (5.66%-32.23%)

Table 3: Alexa’s top 1M sites dataset: Packet drop rate for different destination types, and estimated percentage of dropped packets that were deemed to be dropped in a different AS (lower, in parentheses)

EH Type	Webservers	Mailservers	Nameservers
DO8	10.91% (46.52%-53.23%)	11.54% (2.41%-21.08%)	21.33% (10.27%-56.80%)
HBH8	39.03% (36.90%-46.35%)	45.45% (41.27%-61.13%)	54.12% (50.64%-81.00%)
FH512	28.26% (53.64%-61.43%)	35.68% (3.15%-10.92%)	55.23% (5.66%-32.23%)

Table 4: Alexa's top 1M sites dataset: Packet drop rate for different EH types, and estimated percentage of dropped packets that were deemed to be dropped in a different AS (lower, in parentheses)

NOTE: This table contains the same information as Table 3, but makes it easier to obtain the drop rates for each EH type. Each cell should be read in exactly the same way as each cell in Table 3.

There are a number of observations to be made based on the results presented above. Firstly, while it has been generally assumed that it is IPv6 fragments that are dropped by operators, our results indicate that it is IPv6 extension headers in general that result in packet drops. Secondly, our results indicate that a significant percentage of such packet drops occur in transit Autonomous Systems; that is, the packet drops are not under the control of the same organization as the final destination.

3. IANA Considerations

There are no IANA registries within this document. The RFC-Editor can remove this section before publication of this document as an RFC.

4. Security Considerations

This document presents real-world data regarding the extent to which IPv6 packets employing extension headers are filtered in the Internet. As such, this document does not introduce any new security issues.

5. Acknowledgements

The authors would like to thank (in alphabetical order) Mark Andrews, Fred Baker, Brian Carpenter and Tatuya Jinmei for providing valuable comments on earlier versions of this document. Additionally, the authors would like to thank participants of the v6ops and opsec working groups for their valuable input on the topics discussed in this document.

The authors would like to thank Fred Baker for his guidance in improving this document.

Fernando Gont would like to thank Jan Zorz and Go6 Lab <http://go6lab.si/> for providing access to systems and networks that were employed to produce some of the measurement results presented in this document. Additionally, he would like to thank SixXS <https://www.sixxs.net> for providing IPv6 connectivity.

6. References

6.1. Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

[RFC6946] Gont, F., "Processing of IPv6 "Atomic" Fragments", RFC 6946, May 2013.

6.2. Informative References

[Gont-Chown-IEPG89]
Gont, F. and T. Chown, "A Small Update on the Use of IPv6 Extension Headers", IEPG 89. London, UK. March 2, 2014, <http://www.iepg.org/2014-03-02-ietf89/fgont-iepg-ietf89-eh-update.pdf>.

[Gont-IEPG88]
Gont, F., "Fragmentation and Extension header Support in the IPv6 Internet", IEPG 88. Vancouver, BC, Canada. November 13, 2013, <http://www.iepg.org/2013-11-ietf88/fgont-iepg-ietf88-ipv6-frag-and-eh.pdf>.

[IANA-PORT-NUMBERS]
IANA, "Service Name and Transport Protocol Port Number Registry", <http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt>.

[IPv6-Toolkit]
"SI6 Networks' IPv6 Toolkit", <http://www.si6networks.com/tools/ipv6toolkit>.

[Linkova-Gont-IEPG90]
Linkova, J. and F. Gont, "IPv6 Extension Headers in the Real World v2.0", IEPG 90. Toronto, ON, Canada. July 20, 2014, <http://www.iepg.org/2014-07-20-ietf90/iepg-ietf90-ipv6-ehs-in-the-real-world-v2.0.pdf>.

[PMTUD-Blackholes]
De Boer, M. and J. Bosma, "Discovering Path MTU black holes on the Internet using RIPE Atlas", July 2012, <http://www.nlnetlabs.nl/downloads/publications/pmtu-black-holes-msc-thesis.pdf>.

[RFC5927] Gont, F., "ICMP Attacks against TCP", RFC 5927, July 2010.

[RFC6980] Gont, F., "Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery", RFC 6980, August 2013.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, December 2013.

[RFC7113] Gont, F., "Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)", RFC 7113, February 2014.

[RFC7123] Gont, F. and W. Liu, "Security Implications of IPv6 on IPv4 Networks", RFC 7123, February 2014.

[blackhole6] blackhole6, , "blackhole6 tool manual page", <http://www.si6networks.com/tools/ipv6toolkit>, 2014.

[path6] path6, , "path6 tool manual page", <http://www.si6networks.com/tools/ipv6toolkit>, 2014.

Appendix A. Reproducing Our Experiment

This section describes, step by step, how to reproduce the experiment with which we obtained the results presented in this document. Each subsection represents one step in the experiment. The tools employed for the experiment are traditional UNIX-like tools (such as gunzip), and the SI6 Networks' IPv6 Toolkit [IPv6-Toolkit].

A.1. Obtaining the List of Domain Names

The primary data source employed was Alexa's Top 1M web sites, available at: <http://s3.amazonaws.com/alexa-static/top-1m.csv.zip>. The file is a zipped file containing the list of the most popular web sites, in CSV format. The aforementioned file an be extracted with "gunzip < top-1m.csv.zip > top-1m.csv".

A list of domain names (i.e., other data stripped) can be obtained with the following command of [IPv6-Toolkit]: "cat top-1m.csv | script6 get-alexa-domains > top-1m.txt". This command will create a "top-1m.txt" file, containing one domain name per line.

> NOTE: The domain names corresponding to the WIPv6LD dataset is available at: <http://www.si6networks.com/datasets/wipv6day-domains.txt>. Since the corresponding file is a text file containing one domain name per line, the steps produced in this subsection need not be performed. The WIPv6LD data set should be processed in the same way as the Alexa Dataset, starting from Appendix A.2.

A.2. Obtaining AAAA Resource Records

The file obtained in the previous subsection contains a list of domain names that correspond to web sites. The AAAA records for such domains can be obtained with:

```
$ cat top-1m.txt | script6 get-aaaa > top-1m-web-aaaa.txt
```

The AAAA records corresponding to the mailservers of each of the aforementioned domain names can be obtained with:

```
$ cat top-1m.txt | script6 get-mx | script6 get-aaaa > top-1m-mail-aaaa.txt
```

The AAAA records corresponding to the nameservers of each of the aforementioned domain names can be obtained with:

```
$ cat top-1m.txt | script6 get-ns | script6 get-aaaa > top-1m-dns-aaaa.txt
```

A.3. Filtering the IPv6 Address Datasets

The lists of IPv6 addresses obtained in the previous step could possibly contain undesired addresses (i.e., non-global unicast addresses) and/or duplicate addresses. In order to remove both undesired and duplicate addresses each of the three files from the previous section should be filtered accordingly:

```
$ cat top-1m-web-aaaa.txt | addr6 -i -q -B multicast -B unspec -k global > top-1m-web-aaaa-unique.txt

$ cat top-1m-mail-aaaa.txt | addr6 -i -q -B multicast -B unspec -k global > top-1m-mail-aaaa-unique.txt

$ cat top-1m-dns-aaaa.txt | addr6 -i -q -B multicast -B unspec -k global > top-1m-dns-aaaa-unique.txt
```

A.4. Performing Measurements with Each IPv6 Address Dataset

A.4.1. Measurements with web servers

In order to measure DO8 with the list of webservers:

```
# cat top-1m-web-aaaa-unique.txt | script6 trace6 do8 tcp 80 > > top-1m-web-aaaa-do8-m.txt
```

In order to measure HBH8 with the list of webservers:

```
# cat top-1m-web-aaaa-unique.txt | script6 trace6 hbh8 tcp 80 > > top-1m-web-aaaa-hbh8--m.txt
```

In order to measure FH512 with the list of webservers:

```
# cat top-1m-web-aaaa-unique.txt | script6 trace6 fh512 tcp 80 > > top-1m-web-aaaa-fh512-m.txt
```

A.4.2. Measurements with mail servers

In order to measure DO8 with the list of mailservers:

```
# cat top-1m-mail-aaaa-unique.txt | script6 trace6 do8 tcp 25 > top-
1m-mail-aaaa-do8-m.txt
```

In order to measure HBH8 with the list of webservers:

```
# cat top-1m-mail-aaaa-unique.txt | script6 trace6 hbh8 tcp 25 > top-
1m-mail-aaaa-hbh8-m.txt
```

In order to measure FH512 with the list of webservers:

```
# cat top-1m-mail-aaaa-unique.txt | script6 trace6 fh512 tcp 25 >
top-1m-mail-aaaa-fh512-m.txt
```

A.4.3. Measurements with DNS servers

In order to measure DO8 with the list of mameservers:

```
# cat top-1m-dns-aaaa-unique.txt | script6 trace6 do8 tcp 53 > top-
1m-dns-aaaa-do8-m.txt
```

In order to measure HBH8 with the list of webservers:

```
# cat top-1m-dns-aaaa-unique.txt | script6 trace6 hbh8 tcp 53 > top-
1m-dns-aaaa-hbh8-m.txt
```

In order to measure FH512 with the list of webservers:

```
# cat top-1m-dns-aaaa-unique.txt | script6 trace6 fH512 tcp 53 > top-
1m-dns-aaaa-fh512-m.txt
```

A.5. Obtaining Statistics from our Measurements

A.5.1. Statistics for Web Servers

In order to compute the statistics corresponding to our measurements of DO8 with the list of webservers:

```
$ cat top-1m-web-aaaa-do8-m.txt | script6 get-trace6-stats > top-1m-
web-aaaa-do8-stats.txt
```

In order to compute the statistics corresponding to our measurements of HBH8 with the list of webservers:

```
$ cat top-1m-web-aaaa-hbh8-m.txt | script6 get-trace6-stats > top-1m-
web-aaaa-hbh8-stats.txt
```

In order to compute the statistics corresponding to our measurements of FH512 with the list of webservers:

```
$ cat top-1m-web-aaaa-fh512-m.txt | script6 get-trace6-stats > top-
1m-web-aaaa-fh512-stats.txt
```

A.5.2. Statistics for Mail Servers

In order to compute the statistics corresponding to our measurements of DO8 with the list of mailservers:

```
$ cat top-1m-mail-aaaa-do8-m.txt | script6 get-trace6-stats > top-1m-
mail-aaaa-do8-stats.txt
```

In order to compute the statistics corresponding to our measurements of HBH8 with the list of mailservers:

```
$ cat top-1m-mail-aaaa-hbh8-m.txt | script6 get-trace6-stats > top-
1m-mail-aaaa-hbh8-stats.txt
```

In order to compute the statistics corresponding to our measurements of FH512 with the list of mailservers:

```
$ cat top-1m-mail-aaaa-fh512-m.txt | script6 get-trace6-stats > top-
1m-mail-aaaa-fh512-stats.txt
```

A.5.3. Statistics for Name Servers

In order to compute the statistics corresponding to our measurements of DO8 with the list of nameservers:

```
$ cat top-1m-dns-aaaa-do8-m.txt | script6 get-trace6-stats > top-1m-
dns-aaaa-do8-stats.txt
```

In order to compute the statistics corresponding to our measurements of HBH8 with the list of mailservers:

```
$ cat top-1m-dns-aaaa-hbh8-m.txt | script6 get-trace6-stats > top-1m-
dns-aaaa-hbh8-stats.txt
```

In order to compute the statistics corresponding to our measurements of FH512 with the list of mailservers:

```
$ cat top-1m-dns-aaaa-fh512-m.txt | script6 get-trace6-stats > top-
1m-dns-aaaa-fh512-stats.txt
```

Appendix B. Measurements Caveats

A number of issues have needed some consideration when producing the results presented in this document. These same issues should be considered when troubleshooting connectivity problems resulting from the use of IPv6 Extension headers.

B.1. Isolating the Dropping Node

Let us assume that we find that IPv6 packets with EHs are being dropped on their way to the destination system 2001:db8:d::1, and that the output of running traceroute towards such destination is:

1. 2001:db8:1:1000::1
2. 2001:db8:2:4000::1
3. 2001:db8:3:4000::1
4. 2001:db8:3:1000::1
5. 2001:db8:4:4000::1
6. 2001:db8:4:1000::1
7. 2001:db8:5:5000::1
8. 2001:db8:5:6000::1
9. 2001:db8:d::1

Additionally, let us assume that the output of EH-enabled traceroute to the same destination is:

1. 2001:db8:1:1000::1
2. 2001:db8:2:4000::1
3. 2001:db8:3:4000::1
4. 2001:db8:3:1000::1
5. 2001:db8:4:4000::1

For the sake of brevity, let us refer to the last-responding node in the EH-enabled traceroute ("2001:db8:4:4000::1" in this case) as "M". Assuming both packets in both traceroutes employ the same path, we'll refer to "the node following the last responding node in the EH-enabled traceroute" ("2001:db8:4:1000::1" in our case), as "M+1", etc.

Based on traceroute information above, which node is the one actually dropping the EH-enabled packets will depend on whether the dropping node filters packets before making the forwarding decision, or after making the forwarding decision. If the former, the dropping node will be M+1. If the latter, the dropping node will be "M".

Throughout this document (and our measurements), we assume that those nodes filtering packets that carry IPv6 EHs apply their filtering policy, and only then, if necessary, forward the packets. Thus, in

our example above the last responding node to the EH-enabled traceroute ("M") is "2001:db8:4:4000::1", and therefore we assume the dropping node to be "2001:db8:4:1000::1" ("M+1").

Additionally, we note that when isolating the dropping node we assume that both the EH-enabled and the EH-free traceroutes result in the same paths. However, this might not be the case.

B.2. Obtaining the Responsible Organization for the Packet Drops

In order to identify the organization operating the dropping node, one would be tempted to lookup the ASN corresponding to the dropping node. However, assuming that M and M+1 are two peering routers, any of these two organizations could be providing the address space employed for such peering. Or, in the case of an Internet eXchange Point (IXP), the address space could correspond to the IXP AS, rather than to any of the participating ASes. Thus, the organization operating the dropping node (M+1) could be the AS for M+1, but it might as well be the AS for M+2. Only when the ASN for M+1 is the same as the ASN for M+2 we have certainty about who the responsible organization for the packet drops is (see slides 21-23 of [Linkova-Gont-IEPG90]).

In the measurement results presented in Section 2, the aforementioned ambiguity results in "percentage ranges" (rather than a specific ratio): the lowest percentage value means that, when in doubt, we assume the packet drops occur in the same AS as the destination; on the other hand, the highest percentage value means that, when in doubt, we assume the packet drops occur at different AS than the destination AS.

We note that the aforementioned ambiguity should also be considered when troubleshooting and reporting IPv6 packet drops, since identifying the organization responsible for the packet drops might probe to be a non-trivial task.

Finally, we note that a specific organization might be operating more than one Autonomous System. However, our measurements assume that different Autonomous System Numbers imply different organizations.

Appendix C. Troubleshooting Packet Drops due to IPv6 Extension Headers

Isolating IPv6 blackholes essentially involves performing IPv6 traceroute for a destination system with and without IPv6 extension headers. The (EH-free) traceroute would provide the full working path towards a destination, while the EH-enabled traceroute would provide the address of the last-responding node for EH-enabled packets (say, "M"). In principle, one could isolate the dropping

node by looking-up "M" in the EH-free traceroute, with the dropping node being "M+1" (see Appendix B.1 for caveats).

At the time of this writing, most traceroute implementations do not support IPv6 extension headers. However, the path6 tool [path6] of [IPv6-Toolkit] provides such support. Additionally, the blackhole6 tool [blackhole6] automates the troubleshooting process and can readily provide information such as: dropping node's IPv6 address, dropping node's Autonomous System, etc.

Authors' Addresses

Fernando Gont
SI6 Networks / UTN-FRH
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fgont@si6networks.com
URI: http://www.si6networks.com

J. Linkova
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
USA

Email: furry@google.com

Tim Chown
University of Southampton
Highfield
Southampton , Hampshire SO17 1BJ
United Kingdom

Email: tjc@ecs.soton.ac.uk

Will(Shucheng) Liu
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

V6OPS Working Group
Internet-Draft
Intended status: Informational
Expires: May 17, 2017

P. Matthews
Nokia
V. Kuarsingh
Cisco

November 13, 2016

Routing-Related Design Choices for IPv6 Networks
draft-ietf-v6ops-design-choices-12

Abstract

This document presents advice on certain routing-related design choices that arise when designing IPv6 networks (both dual-stack and IPv6-only). The intended audience is someone designing an IPv6 network who is knowledgeable about best current practices around IPv4 network design, and wishes to learn the corresponding practices for IPv6.

Table of Contents

1. Introduction

This document discusses routing-related design choices that arise when designing an IPv6-only or dual-stack network. The focus is on choices that do not come up when designing an IPv4-only network. The document presents each choice and the alternatives, and then discusses the pros and cons of the alternatives in detail. Where consensus currently exists around the best practice, this is documented; otherwise the document simply summarizes the current state of the discussion. Thus this document serves to both document the reasoning behind best current practices for IPv6, and to allow a designer to make an informed choice where no such consensus exists.

The design choices presented apply to both Service Provider and Enterprise network environments. Where choices have selection criteria which differ between the Service Provider and the Enterprise

environment, this is noted. The designer is encouraged to ensure that they familiarize themselves with any of the discussed technologies to ensure the best selection is made for their environment.

This document does not present advice on strategies for adding IPv6 to a network, nor does it discuss transition in these areas, see [RFC6180] for general advice,[RFC6782] for wireline service providers, [RFC6342]for mobile network providers, [RFC5963] for exchange point operators, [RFC6883] for content providers, and both [RFC4852] and [RFC7381] for enterprises. Nor does this document discuss the particulars of creating an IPv6 addressing plan; for advice in this area, see [RFC5375] or [v6-addressing-plan]. The document focuses on unicast routing design only and does not cover multicast or the issues involved in running MPLS over IPv6 transport

Section 2 presents and discusses a number of design choices. Section 3 discusses some general themes that run through these choices.

2. Design Choices

Each subsection below presents a design choice and discusses the pros and cons of the various options. If there is consensus in the industry for a particular option, then the consensus position is noted.

2.1. Addresses

This section discusses the choice of addresses for router loopbacks and links between routers. It does not cover the choice of addresses for end hosts.

In IPv6, an interface is always assigned a Link-Local Address (LLA) [RFC4291]. The link-local address can only be used for communicating with devices that are on-link, so often one or more additional addresses are assigned which are able to communicate off-link. This additional address or addresses can be one of three types:

- Provider-Independent Global Unicast Address (PI GUA): IPv6 address allocated by a regional address registry [RFC4291]
- Provider-Aggregatable Global Unicast Address (PA GUA): IPv6 Address allocated by your upstream service provider
- Unique Local Address (ULA): IPv6 address locally assigned [RFC4193]

This document uses the term "multi-hop address" to collectively refer to these three types of addresses.

PI GUAs are, for many situations, the most flexible of these choices. Their main disadvantages are that a regional address registry will only allocate them to organizations that meet certain qualifications, and one must pay an annual fee. These disadvantages mean that many smaller organization may not qualify or be willing to pay for these addresses.

PA GUAs have the advantage that they are usually provided at no extra charge when you contract with an upstream provider. However, they have the disadvantage that, when switching upstream providers, one must give back the old addresses and get new addresses from the new provider ("renumbering"). Though IPv6 has mechanisms to make renumbering easier than IPv4, these techniques are not generally applicable to routers and renumbering is still fairly hard [RFC5887] [RFC6879] [RFC7010] . PA GUAs also have the disadvantage that it is not easy to have multiple upstream providers ("multi-homing") if they are used (see "Ingress Filtering Problem" in [RFC5220]).

ULAs have the advantage that they are extremely easy to obtain and cost nothing. However, they have the disadvantage that they cannot be routed on the Internet, so must be used only within a limited scope. In many situations, this is not a problem, but in certain situations this can be problematic. Though there is currently no document that describes these situations, many of them are similar to those described in [RFC6752]. See also [I-D.ietf-v6ops-ula-usage-recommendations].

Not discussed in this document is the possibility of using the technology described in [RFC6296] to work around some of the limitations of PA GUAs and ULAs.

2.1.1. Where to Use Addresses

As mentioned above, all interfaces in IPv6 always have a link-local address. This section addresses the question of when and where to assign multi-hop addresses in addition to the LLA. We consider four options:

a. Use only link-local addresses on all router interfaces.

b. Assign multi-hop addresses to all link interfaces on each router, and use only a link-local address on the loopback interfaces.

c. Assign multi-hop addresses to the loopback interface on each router, and use only a link-local address on all link interfaces.

d. Assign multi-hop addresses to both link and loopback interfaces on each router.

Option (a) means that the router cannot be reached (ping, management, etc.) from farther than one-hop away. The authors are not aware of anyone using this option.

Option (b) means that the loopback interfaces are effectively useless, since link-local addresses cannot be used for the purposes that loopback interfaces are usually used for. So option (b) degenerates into option (d).

Thus the real choice comes down to option (c) vs. option (d).

Option (c) has two advantages over option (d). The first advantage is ease of configuration. In a network with a large number of links, the operator can just assign one multi-hop address to each router and then enable the IGP, without going through the tedious process of assigning and tracking the addresses on each link. The second advantage is security. Since packets with link-local addresses cannot be should not be routed, it is very difficult to attack the associated nodes from an off-link device. This implies less effort around maintaining security ACLs.

Countering these advantages are various disadvantages to option (c) compared with option (d):

- It is not possible to ping a link-local-only interface from a device that is not directly attached to the link. Thus, to troubleshoot, one must typically log into a device that is directly attached to the device in question, and execute the ping from there.

- A traceroute passing over the link-local-only interface will return the loopback address of the router, rather than the address of the interface itself.

- In cases of parallel point to point links it is difficult to determine which of the parallel links was taken when attempting to troubleshoot unless one sends packets directly between the two attached link-locals on the specific interfaces. Since many network problems behave differently for traffic to/from a router than for traffic through the router(s) in question, this can pose a significant hurdle to some troubleshooting scenarios.

- On some routers, by default the link-layer address of the interface is derived from the MAC address assigned to interface. When this is done, swapping out the interface hardware (e.g.

interface card) will cause the link-layer address to change. In some cases (peering config, ACLs, etc) this may require additional changes. However, many devices allow the link-layer address of an interface to be explicitly configured, which avoids this issue. This problem should fade away over time as more and more routers select interface identifiers according to the rules in [RFC7217].

- o The practice of naming router interfaces using DNS names is difficult and not recommended when using link-locals only. More generally, it is not recommended to put link-local addresses into DNS; see [RFC4472].

- o It is often not possible to identify the interface or link (in a database, email, etc) by giving just its address without also specifying the link in some manner.

It should be noted that it is quite possible for the same link-local address to be assigned to multiple interfaces. This can happen because the MAC address is duplicated (due to manufacturing process defaults or the use of virtualization), because a device deliberately re-uses automatically-assigned link-local addresses on different links, or because an operator manually assigns the same easy-to-type link-local address to multiple interfaces. All these are allowed in IPv6 as long as the addresses are used on different links.

For more discussion on the pros and cons, see [RFC7404]. See also [RFC5375] for IPv6 unicast address assignment considerations.

Today, most operators use option (d).

2.1.2. Which Addresses to Use

Having considered above whether or not to use a "multi-hop address", we now consider which of the addresses to use.

When selecting between these three "multi-hop address" types, one needs to consider exactly how they will be used. An important consideration is how Internet traffic is carried across the core of the network. There are two main options: (1) the classic approach where Internet traffic is carried as unlabeled traffic hop-by-hop across the network, and (2) the more recent approach where Internet traffic is carried inside an MPLS LSP (typically as part of a L3 VPN).

Under the classic approach:

- o PI GUAs are a very reasonable choice, if they are available.

- PA GUAs suffer from the "must renumber" and "difficult to multi-home" problems mentioned above.

- ULAs suffer from the "may be problematic" issues described above.

Under the MPLS approach:

- PA GUAs are a reasonable choice, if they are available.

- PA GUAs suffer from the "must renumber" problem, but the "difficult to multi-home" problem does not apply.

- ULAs are a reasonable choice, since (unlike in the classic approach) these addresses are not visible to the Internet, so the problematic cases do not occur.

2.2. Interfaces

2.2.1. Mix IPv4 and IPv6 on the Same Layer-3 Interface?

If a network is going to carry both IPv4 and IPv6 traffic, as many networks do today, then a question arises: Should an operator mix IPv4 and IPv6 traffic or keep them separated? More specifically, should the design:

a. Mix IPv4 and IPv6 traffic on the same layer-3 interface, OR

b. Separate IPv4 and IPv6 by using separate interfaces (e.g., two physical links or two VLANs on the same link)?

Option (a) implies a single layer-3 interface at each end of the connection with both IPv4 and IPv6 addresses; while option (b) implies two layer-3 interfaces at each end, one for IPv4 addresses and one with IPv6 addresses.

The advantages of option (a) include:

- Requires only half as many layer 3 interfaces as option (b), thus providing better scaling;

- May require fewer physical ports, thus saving money and simplifying operations;

- Can make the QoS implementation much easier (for example, rate-limiting the combined IPv4 and IPv6 traffic to or from a customer);

- Works well in practice, as any increase in IPv6 traffic is usually counter-balanced by a corresponding decrease in IPv4 traffic to or from the same host (ignoring the common pattern of an overall increase in Internet usage);

- And is generally conceptually simpler.

For these reasons, there is a relatively strong consensus in the operator community that option (a) is the preferred way to go. Most networks today use option (a) wherever possible.

However, there can be times when option (b) is the pragmatic choice. Most commonly, option (b) is used to work around limitations in network equipment. One big example is the generally poor level of support today for individual statistics on IPv4 traffic vs IPv6 traffic when option (a) is used. Other, device-specific, limitations exist as well. It is expected that these limitations will go away as support for IPv6 matures, making option (b) less and less attractive until the day that IPv4 is finally turned off.

2.3. Static Routes

2.3.1. Link-Local Next-Hop in a Static Route?

For the most part, the use of static routes in IPv6 parallels their use in IPv4. There is, however, one exception, which revolves around the choice of next-hop address in the static route. Specifically, should an operator:

a. Use the far-end’s link-local address as the next-hop address, OR

b. Use the far-end’s GUA/ULA address as the next-hop address?

Recall that the IPv6 specs for OSPF [RFC5340] and ISIS [RFC5308] dictate that they always use link-locals for next-hop addresses. For static routes, [RFC4861] section 8 says:

> A router MUST be able to determine the link-local address for each of its neighboring routers in order to ensure that the target address in a Redirect message identifies the neighbor router by its link-local address. For static routing, this requirement implies that the next-hop router’s address should be specified using the link-local address of the router.

This implies that using a GUA or ULA as the next hop will prevent a router from sending Redirect messages for packets that "hit" this static route. All this argues for using a link-local as the next-hop address in a static route.

However, there are two cases where using a link-local address as the next-hop clearly does not work. One is when the static route is an indirect (or multi-hop) static route. The second is when the static route is redistributed into another routing protocol. In these cases, the above text from RFC 4861 notwithstanding, either a GUA or ULA must be used.

Furthermore, many network operators are concerned about the dependency of the default link-local address on an underlying MAC address, as described in the previous section.

Today most operators use GUAs as next-hop addresses.

2.4. IGPs

2.4.1. IGP Choice

One of the main decisions for a network operator looking to deploy IPv6 is the choice of IGP (Interior Gateway Protocol) within the network. The main options are OSPF, IS-IS and EIGRP. RIPng is another option, but very few networks run RIP in the core these days, so it is covered in a separate section below.

OSPF [RFC2328] [RFC5340] and IS-IS [RFC5120][RFC5120] are both standardized link-state protocols. Both protocols are widely supported by vendors, and both are widely deployed. By contrast, EIGRP [RFC7868] is a Cisco proprietary distance-vector protocol. EIGRP is rarely deployed in service-provider networks, but is quite common in enterprise networks, which is why it is discussed here.

It is out of scope for this document to describe all the differences between the three protocols; the interested reader can find books and websites that go into the differences in quite a bit of detail. Rather, this document simply highlights a few differences that can be important to consider when designing IPv6 or dual-stack networks.

Versions: There are two versions of OSPF: OSPFv2 and OSPFv3. The two versions share many concepts, are configured in a similar manner and seem very similar to most casual users, but have very different packet formats and other "under the hood" differences. The most important difference is that OSPFv2 will only route IPv4, while OSPFv3 will route both IPv4 and IPv6 (see [RFC5838]). OSPFv2 was by far the most widely deployed version of OSPF when this document was published. By contrast, both IS-IS and EIGRP have just a single version, which can route both IPv4 and IPv6.

Transport. IS-IS runs over layer 2 (e.g. Ethernet). This means that the functioning of IS-IS has no dependencies on the IP layer: if

there is a problem at the IP layer (e.g. bad addresses), two routers can still exchange IS-IS packets. By contrast, OSPF and EIGRP both run over the IP layer. This means that the IP layer must be configured and working OSPF or EIGRP packets to be exchanged between routers. For EIGRP, the dependency on the IP layer is simple: EIGRP for IPv4 runs over IPv4, while EIGRP for IPv6 runs over IPv6. For OSPF, the story is more complex: OSPFv2 runs over IPv4, but OSPFv3 can run over either IPv4 or IPv6. Thus it is possible to route both IPv4 and IPv6 with OSPFv3 running over IPv6 or with OSPFv3 running over IPv4. This means that there are number of choices for how to run OSPF in a dual-stack network:

- Use OSPFv2 for routing IPv4 , and OSPFv3 running over IPv6 for routing IPv6, OR
- Use OSPFv3 running over IPv6 for routing both IPv4 and IPv6, OR
- Use OSPFv3 running over IPv4 for routing both IPv4 and IPv6.

Summarization and MPLS: For most casual users, the three protocols are fairly similar in what they can do, with two glaring exceptions: summarization and MPLS. For summarization, both OSPF and IS-IS have the concept of summarization between areas, but the two area concepts are quite different, and an area design that works for one protocol will usually not work for the other. EIGRP has no area concept, but has the ability to summarize at any router. Thus a large network will typically have a very different OSPF, IS-IS and EIGRP designs, which is important to keep in mind if you are planning on using one protocol to route IPv4 and a different protocol for IPv6. The other difference is that OSPF and IS-IS both support RSVP-TE, a widely-used MPLS signaling protocol, while EIGRP does not: this is due to OSPF and IS-IS both being link-state protocols while EIGRP is a distance-vector protocol.

The table below sets out possible combinations of protocols to route both IPv4 and IPv6, and makes some observations on each combination. Here "EIGRP-v4" means "EIGRP for IPv4" and similarly for "EIGRP-v6". For OSPFv3, it is possible to run it over either IPv4 or IPv6; this is not indicated in the table.

IGP for IPv4	IGP for IPv6	Protocol separation	Similar configuration possible	Multiple Known Deployments
OSPFv2	OSPFv3	YES	YES	YES (8)
OSPFv2	IS-IS	YES	-	YES (3)
OSPFv2	EIGRP-v6	YES	-	-
OSPFv3	OSPFv3	NO	YES	-
OSPFv3	IS-IS	YES	-	-
OSPFv3	EIGRP-v6	YES	-	-
IS-IS	OSPFv3	YES	-	YES (2)
IS-IS	IS-IS	-	YES	YES (12)
IS-IS	EIGRP-v6	YES	-	-
EIGRP-v4	OSPFv3	YES	-	? (1)
EIGRP-v4	IS-IS	YES	-	-
EIGRP-v4	EIGRP-v6	-	YES	? (2)

In the column "Multiple Known Deployments", a YES indicates that a significant number of production networks run this combination, with the number of such networks indicated in parentheses following, while a "?" indicates that the authors are only aware of one or two small networks that run this combination. Data for this column was gathered from an informal poll of operators on a number of mailing lists. This poll was not intended to be a thorough scientific study of IGP choices, but to provide a snapshot of known operator choices at the time of writing (Mid-2015) for successful production dual stack network deployments. There were twenty six (26) network implementations represented by 17 respondents. Some respondents provided information on more then one network or network deployment. Due to privacy considerations, the networks' represented and respondents are not listed in this document.

A number of combinations are marked as offering "Protocol separation". These options use a different IGP protocol for IPv4 vs IPv6. With these options, a problem with routing IPv6 is unlikely to affect IPv4 or visa-versa. Some operator may consider this as a benefit when first introducing dual stack capabilities or for ongoing technical reasons.

Three combinations are marked "Similar configuration possible". This means it is possible (but not required) to use very similar IGP configuration for IPv4 and IPv6: for example, the same area boundaries, area numbering, link costing, etc. If you are happy with your IPv4 IGP design, then this will likely be a consideration. By contrast, the options that use, for example, IS-IS for one IP version and OSPF for the other version will require considerably different configuration, and will also require the operations staff to become familiar with the difference between the two protocols.

It should be noted that a number of ISPs have run OSPF as their IPv4 IGP for quite a few years, but have selected IS-IS as their IPv6 IGP. However, there are very few (none?) that have made the reverse choice. This is, in part, because routers generally support more nodes in an IS-IS area than in the corresponding OSPF area, and because IS-IS is seen as more secure because it runs at layer 2.

2.4.2. IS-IS Topology Mode

When IS-IS is used to route both IPv4 and IPv6, then there is an additional choice of whether to run IS-IS in single-topology or multi-topology mode.

With single-topology mode (also known as Native mode) [RFC5308]:

- IS-IS keeps a single link-state database for both IPv4 and IPv6.
- There is a single set of link costs which apply to both IPv4 and IPv6.
- All links in the network must support both IPv4 and IPv6, as the calculation of routes does not take this into account. If some links do not support IPv6 (or IPv4), then packets may get routed across links where support is lacking and get dropped. This can cause problems if some network devices do not support IPv6 (or IPv4).
- It is also important to keep the previous point in mind when adding or removing support for either IPv4 or IPv6.

With multi-topology mode [RFC5120]:

- IS-IS keeps two link-state databases, one for IPv4 and one for IPv6.

- IPv4 and IPv6 can have separate link metrics. Note that most implementations today require separate link metrics: a number of operators have rudely discovered that they have forgotten to configure the IPv6 metric until sometime after deploying IPv6 in multi-topology mode!

- Some links can be IPv4-only, some IPv6-only, and some dual-stack. Routes to IPv4 and IPv6 addresses are computed separately and may take different paths even if the addresses are located on the same remote device.

- The previous point may help when adding or removing support for either IPv4 or IPv6.

In the informal poll of operators, out of 12 production networks that ran IS-IS for both IPv4 and IPv6, 6 used single topology mode, 4 used multi-topology mode, and 2 did not specify. One motivation often cited by then operators for using Single Topology mode was because some device did not support multi-topology mode.

When asked, many people feel multi-topology mode is superior to single-topology mode because it provides greater flexibility at minimal extra cost. Never-the-less, as shown by the poll results, a number of operators have used single-topology mode successfully.

Note that this issue does not come up with OSPF, since there is nothing that corresponds to IS-IS single-topology mode with OSPF.

2.4.3. RIP / RIPng

A protocol option not described in the table above is RIP for IPv4 and RIPng for IPv6 [RFC2080]. These are distance vector protocols that are almost universally considered to be inferior to OSPF, IS-IS, or EIGRP for general use.

However, there is one specialized use where RIP/RIPng is still considered to be appropriate: in star topology networks where a single core device has lots and lots of links to edge devices and each edge device has only a single path back to the core. In such networks, the single path means that the limitations of RIP/RIPng are mostly not relevant and the very light-weight nature of RIP/RIPng gives it an advantage over the other protocols mentioned above. One concrete example of this scenario is the use of RIP/RIPng between cable modems and the CMTS.

2.5. BGP

2.5.1. Which Transport for Which Routes?

BGP these days is multi-protocol. It can carry routes of many different types, or more precisely, many different AFI/SAFI combinations. It can also carry routes when the BGP session, or more accurately the underlying TCP connection, runs over either IPv4 or IPv6 (here referred to as either "IPv4 transport" or "IPv6 transport"). Given this flexibility, one of the biggest questions when deploying BGP in a dual-stack network is the question of which route types should be carried over sessions using IPv4 transport and which should be carried over sessions using IPv6 transport.

This section discusses this question for the three most-commonly-used SAFI values: unlabeled (SAFI 1), labeled (SAFI 4) and VPN (SAFI 128). Though we do not explicitly discuss other SAFI values, many of the comments here can be applied to the other values.

Consider the following table:

Route Family	Transport	Comments
Unlabeled IPv4	IPv4	Works well
Unlabeled IPv4	IPv6	Next-hop
Unlabeled IPv6	IPv4	Next-hop
Unlabeled IPv6	IPv6	Works well
Labeled IPv4	IPv4	Works well
Labeled IPv4	IPv6	Next-hop
Labeled IPv6	IPv4	(6PE) Works well
Labeled IPv6	IPv6	Next-hop or MPLS over IPv6
VPN IPv4	IPv4	Works well
VPN IPv4	IPv6	Next-hop
VPN IPv6	IPv4	(6VPE) Works well
VPN IPv6	IPv6	Next-hop or MPLS over IPv6

The first column in this table lists various route families, where "unlabeled" means SAFI 1, "labeled" means the routes carry an MPLS label (SAFI 4, see [RFC3107]), and "VPN" means the routes are normally associated with a layer-3 VPN (SAFI 128, see [RFC4364]). The second column lists the protocol used to transport the BGP session, frequently specified by giving either an IPv4 or IPv6 address in the "neighbor" statement.

The third column comments on the combination in the first two columns:

- For combinations marked "Works well", these combinations are standardized, widely supported and widely deployed.

- For combinations marked "Next-hop", these combinations are not standardized and are less-widely supported. These combinations all have the "next-hop mismatch" problem: the transported route needs a next-hop address from the other address family than the transport address (for example, an IPv4 route needs an IPv4 next-hop, even when transported over IPv6). Some vendors have implemented ways to solve this problem for specific combinations, but for combinations marked "next-hop", these solutions have not been standardized (cf. 6PE and 6VPE, where the solution has been standardized).

- For combinations marked as "Next-hop or MPLS over IPv6", these combinations either require a non-standard solution to the next-hop problem, or require MPLS over IPv6. At the time of writing, MPLS over IPv6 is not widely supported or deployed.

Also, it is important to note that changing the set of address families being carried over a BGP session requires the BGP session to be reset (unless something like [I-D.ietf-idr-dynamic-cap] or [I-D.ietf-idr-bgp-multisession] is in use). This is generally more of an issue with eBGP sessions than iBGP sessions: for iBGP sessions it is common practice for a router to have two iBGP sessions, one to each member of a route reflector pair, so one can change the set of address families on first one of the sessions and then the other.

The following subsections discuss specific combinations in more detail.

2.5.1.1. BGP Sessions for Unlabeled Routes

Unlabeled routes are commonly carried on eBGP sessions, as well as on iBGP sessions in networks where Internet traffic is carried unlabeled across the network.

In these scenarios, there are three reasonable choices:

a. Carry unlabeled IPv4 and IPv6 routes over IPv4, OR

b. Carry unlabeled IPv4 and IPv6 routes over IPv6, OR

c. Carry unlabeled IPv4 routes over IPv4, and unlabeled IPv6 routes over IPv6

Options (a) and (b) have the advantage that one one BGP session is required between pairs of routers. However, option (c) is widely considered to be the best choice. There are several reasons for this :

- o It gives a clean separation between IPv4 and IPv6. This can be especially useful when first deploying IPv6 and troubleshooting resulting problems.

- o This avoids the next-hop problem described above.

- o The status of the routes follows the status of the underlying transport. If, for example, the IPv6 data path between the two BGP speakers fails, then the IPv6 session between the two speakers will fail and the IPv6 routes will be withdrawn, which will allow the traffic to be re-routed elsewhere. By contrast, if the IPv6 routes were transported over IPv4, then the failure of the IPv6 data path might leave a working IPv4 data path, so the BGP session would remain up and the IPv6 routes would not be withdrawn, and thus the IPv6 traffic would be sent into a black hole.

- o It avoids resetting the BGP session when adding IPv6 to an existing session, or when removing IPv4 from an existing session.

Rarely, there are situations where option (c) is not practical. In those cases today, most operators use option (a), carrying both route types over a single BGP session.

2.5.1.2. BGP sessions for Labeled or VPN Routes

When carrying labeled or VPN routes, the only widely-supported solution at time of writing is to carry both route types over IPv4. This may change in as MPLS over IPv6 becomes more widely implemented.

There are two options when carrying both over IPv4:

a. Carry all routes over a single BGP session, OR

b. Carry the routes over multiple BGP sessions (e.g. one for VPN IPv4 routes and one for VPN IPv6 routes)

Using a single session is usually simplest for an iBGP session going to a route reflector handling both route families. Using a single session here usually means that the BGP session will reset when changing the set of address families, but as noted above, this is usually not a problem when redundant route reflectors are involved.

In eBGP situations, two sessions are usually more appropriate. [JUSTIFICATION?]

2.5.2. eBGP Endpoints: Global or Link-Local Addresses?

When running eBGP over IPv6, there are two options for the addresses to use at each end of the eBGP session (or more properly, the underlying TCP session):

a. Use link-local addresses for the eBGP session, OR

b. Use global addresses for the eBGP session.

Note that the choice here is the addresses to use for the eBGP sessions, and not whether the link itself has global (or unique-local) addresses. In particular, it is quite possible for the eBGP session to use link-local addresses even when the link has global addresses.

The big attraction for option (a) is security: an eBGP session using link-local addresses is extremely difficult to attack from a device that is off-link. This provides very strong protection against TCP RST and similar attacks. Though there are other ways to get an equivalent level of security (e.g. GTSM [RFC5082], MD5 [RFC5925], or ACLs), these other ways require additional configuration which can be forgotten or potentially mis-configured.

However, there are a number of small disadvantages to using link-local addresses:

- Using link-local addresses only works for single-hop eBGP sessions; it does not work for multi-hop sessions.
- One must use "next-hop self" at both endpoints, otherwise re-advertising routes learned via eBGP into iBGP will not work. (Some products enable "next-hop self" in this situation automatically).
- Operators and their tools are used to referring to eBGP sessions by address only, something that is not possible with link-local addresses.
- If one is configuring parallel eBGP sessions for IPv4 and IPv6 routes, then using link-local addresses for the IPv6 session introduces extra operational differences between the two sessions which could otherwise be avoided.
- On some products, an eBGP session using a link-local address is more complex to configure than a session that uses a global address.

- If hardware or other issues cause one to move the cable to a different local interface, then reconfiguration is required at both ends: at the local end because the interface has changed (and with link-local addresses, the interface must always be specified along with the address), and at the remote end because the link-local address has likely changed. (Contrast this with using global addresses, where less re-configuration is required at the local end, and no reconfiguration is required at the remote end).

- Finally, a strict application of [RFC2545] forbids running eBGP between link-local addresses, as [RFC2545] requires the BGP next-hop field to contain at least a global address.

For these reasons, most operators today choose to have their eBGP sessions use global addresses.

3. General Observations

There are two themes that run though many of the design choices in this document. This section presents some general discussion on these two themes.

3.1. Use of Link-Local Addresses

The proper use of link-local addresses is a common theme in the IPv6 network design choices. Link-layer addresses are, of course, always present in an IPv6 network, but current network design practice mostly ignores them, despite efforts such as [RFC7404].

There are three main reasons for this current practice:

- Network operators are concerned about the volatility of link-local addresses based on MAC addresses, despite the fact that this concern can be overcome by manually-configuring link-local addresses;

- It is very difficult to impossible to ping a link-local address from a device that is not on the same subnet. This is a troubleshooting disadvantage, though it can also be viewed as a security advantage.

- Most operators are currently running networks that carry both IPv4 and IPv6 traffic, and wish to harmonize their IPv4 and IPv6 design and operational practices where possible.

3.2. Separation of IPv4 and IPv6

Currently, most operators are running or planning to run networks that carry both IPv4 and IPv6 traffic. Hence the question: To what degree should IPv4 and IPv6 be kept separate? As can be seen above, this breaks into two sub-questions: To what degree should IPv4 and IPv6 traffic be kept separate, and to what degree should IPv4 and IPv6 routing information be kept separate?

The general consensus around the first question is that IPv4 and IPv6 traffic should generally be mixed together. This recommendation is driven by the operational simplicity of mixing the traffic, plus the general observation that the service being offered to the end user is Internet connectivity and most users do not know or care about the differences between IPv4 and IPv6. Thus it is very desirable to mix IPv4 and IPv6 on the same link to the end user. On other links, separation is possible but more operationally complex, though it does occasionally allow the operator to work around limitations on network devices. The situation here is roughly comparable to IP and MPLS traffic: many networks mix the two traffic types on the same links without issues.

By contrast, there is more of an argument for carrying IPv6 routing information over IPv6 transport, while leaving IPv4 routing information on IPv4 transport. By doing this, one gets fate-sharing between the control and data plane for each IP protocol version: if the data plane fails for some reason, then often the control plane will too.

4. IANA Considerations

This document makes no requests of IANA.

5. Security Considerations

This document introduces no new security considerations that are not already documented elsewhere.

The following is a brief list of pointers to documents related to the topics covered above that the reader may wish to review for security considerations.

For general IPv6 security, [RFC4942] provides guidance on security considerations around IPv6 transition and coexistence.

For OSPFv3, the base protocol specification [RFC5340] has a short security considerations section which notes that the fundamental

mechanism for protecting OSPFv3 from attacks is the mechanism described in [RFC4552].

For IS-IS, [RFC5308] notes that ISIS for IPv6 raises no new security considerations over ISIS for IPv4 over those documented in [ISO10589] and [RFC5304].

For BGP, [RFC2545] notes that BGP for IPv6 raises no new security considerations over those present in BGP for IPv4. However, there has been much discussion of BGP security recently, and the interested reader is referred to the documents of the IETF's SIDR working group.

6. Acknowledgements

Many, many people in the V6OPS working group provided comments and suggestions that made their way into this document. A partial list includes: Rajiv Asati, Fred Baker, Michael Behringer, Marc Blanchet, Ron Bonica, Randy Bush, Cameron Byrne, Brian Carpenter, KK Chittimaneni, Tim Chown, Lorenzo Colitti, Gert Doering, Francis Dupont, Bill Fenner, Kedar K Gaonkar, Chris Grundemann, Steinar Haug, Ray Hunter, Joel Jaeggli, Victor Kuarsingh, Jen Linkova, Ivan Pepelnjak, Alexandru Petrescu, Rob Shakir, Mark Smith, Jean-Francois Tremblay, Dave Thaler, Tina Tsou, Eric Vyncke, Dan York, and Xuxiaohu.

The authors would also like to thank Pradeep Jain and Alastair Johnson for helpful comments on a very preliminary version of this document.

7. Informative References

[I-D.ietf-idr-bgp-multisession]
Scudder, J., Appanna, C., and I. Varlashkin, "Multisession BGP", draft-ietf-idr-bgp-multisession-07 (work in progress), September 2012.

[I-D.ietf-idr-dynamic-cap]
Ramachandra, S. and E. Chen, "Dynamic Capability for BGP-4", draft-ietf-idr-dynamic-cap-14 (work in progress), December 2011.

[I-D.ietf-v6ops-ula-usage-recommendations]
Liu, B. and S. Jiang, "Considerations For Using Unique Local Addresses", draft-ietf-v6ops-ula-usage-recommendations-05 (work in progress), May 2015.

[ISO10589]
International Standards Organization, "Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473)", International Standard 10589:2002, Nov 2002.

[RFC2080] Malkin, G. and R. Minnear, "RIPng for IPv6", RFC 2080, DOI 10.17487/RFC2080, January 1997, <http://www.rfc-editor.org/info/rfc2080>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <http://www.rfc-editor.org/info/rfc2328>.

[RFC2545] Marques, P. and F. Dupont, "Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing", RFC 2545, DOI 10.17487/RFC2545, March 1999, <http://www.rfc-editor.org/info/rfc2545>.

[RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001, <http://www.rfc-editor.org/info/rfc3107>.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <http://www.rfc-editor.org/info/rfc4193>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <http://www.rfc-editor.org/info/rfc4291>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <http://www.rfc-editor.org/info/rfc4364>.

[RFC4472] Durand, A., Ihren, J., and P. Savola, "Operational Considerations and Issues with IPv6 DNS", RFC 4472, DOI 10.17487/RFC4472, April 2006, <http://www.rfc-editor.org/info/rfc4472>.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <http://www.rfc-editor.org/info/rfc4552>.

[RFC4852] Bound, J., Pouffary, Y., Klynsma, S., Chown, T., and D. Green, "IPv6 Enterprise Network Analysis - IP Layer 3 Focus", RFC 4852, DOI 10.17487/RFC4852, April 2007, <http://www.rfc-editor.org/info/rfc4852>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/ Co-existence Security Considerations", RFC 4942, DOI 10.17487/RFC4942, September 2007, <http://www.rfc-editor.org/info/rfc4942>.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <http://www.rfc-editor.org/info/rfc5082>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <http://www.rfc-editor.org/info/rfc5120>.

[RFC5220] Matsumoto, A., Fujisaki, T., Hiromi, R., and K. Kanayama, "Problem Statement for Default Address Selection in Multi-Prefix Environments: Operational Issues of RFC 3484 Default Rules", RFC 5220, DOI 10.17487/RFC5220, July 2008, <http://www.rfc-editor.org/info/rfc5220>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <http://www.rfc-editor.org/info/rfc5304>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <http://www.rfc-editor.org/info/rfc5308>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <http://www.rfc-editor.org/info/rfc5340>.

[RFC5375] Van de Velde, G., Popoviciu, C., Chown, T., Bonness, O., and C. Hahn, "IPv6 Unicast Address Assignment Considerations", RFC 5375, DOI 10.17487/RFC5375, December 2008, <http://www.rfc-editor.org/info/rfc5375>.

[RFC5838] Lindem, A., Ed., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, DOI 10.17487/RFC5838, April 2010, <http://www.rfc-editor.org/info/rfc5838>.

[RFC5887] Carpenter, B., Atkinson, R., and H. Flinck, "Renumbering Still Needs Work", RFC 5887, DOI 10.17487/RFC5887, May 2010, <http://www.rfc-editor.org/info/rfc5887>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <http://www.rfc-editor.org/info/rfc5925>.

[RFC5963] Gagliano, R., "IPv6 Deployment in Internet Exchange Points (IXPs)", RFC 5963, DOI 10.17487/RFC5963, August 2010, <http://www.rfc-editor.org/info/rfc5963>.

[RFC6180] Arkko, J. and F. Baker, "Guidelines for Using IPv6 Transition Mechanisms during IPv6 Deployment", RFC 6180, DOI 10.17487/RFC6180, May 2011, <http://www.rfc-editor.org/info/rfc6180>.

[RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, DOI 10.17487/RFC6296, June 2011, <http://www.rfc-editor.org/info/rfc6296>.

[RFC6342] Koodli, R., "Mobile Networks Considerations for IPv6 Deployment", RFC 6342, DOI 10.17487/RFC6342, August 2011, <http://www.rfc-editor.org/info/rfc6342>.

[RFC6752] Kirkham, A., "Issues with Private IP Addressing in the Internet", RFC 6752, DOI 10.17487/RFC6752, September 2012, <http://www.rfc-editor.org/info/rfc6752>.

[RFC6782] Kuarsingh, V., Ed. and L. Howard, "Wireline Incremental IPv6", RFC 6782, DOI 10.17487/RFC6782, November 2012, <http://www.rfc-editor.org/info/rfc6782>.

[RFC6879] Jiang, S., Liu, B., and B. Carpenter, "IPv6 Enterprise Network Renumbering Scenarios, Considerations, and Methods", RFC 6879, DOI 10.17487/RFC6879, February 2013, <http://www.rfc-editor.org/info/rfc6879>.

[RFC6883] Carpenter, B. and S. Jiang, "IPv6 Guidance for Internet Content Providers and Application Service Providers", RFC 6883, DOI 10.17487/RFC6883, March 2013, <http://www.rfc-editor.org/info/rfc6883>.

[RFC7010] Liu, B., Jiang, S., Carpenter, B., Venaas, S., and W. George, "IPv6 Site Renumbering Gap Analysis", RFC 7010, DOI 10.17487/RFC7010, September 2013, <http://www.rfc-editor.org/info/rfc7010>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <http://www.rfc-editor.org/info/rfc7217>.

[RFC7381] Chittimaneni, K., Chown, T., Howard, L., Kuarsingh, V., Pouffary, Y., and E. Vyncke, "Enterprise IPv6 Deployment Guidelines", RFC 7381, DOI 10.17487/RFC7381, October 2014, <http://www.rfc-editor.org/info/rfc7381>.

[RFC7404] Behringer, M. and E. Vyncke, "Using Only Link-Local Addressing inside an IPv6 Network", RFC 7404, DOI 10.17487/RFC7404, November 2014, <http://www.rfc-editor.org/info/rfc7404>.

[RFC7868] Savage, D., Ng, J., Moore, S., Slice, D., Paluch, P., and R. White, "Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP)", RFC 7868, DOI 10.17487/RFC7868, May 2016, <http://www.rfc-editor.org/info/rfc7868>.

[v6-addressing-plan] SurfNet, "Preparing an IPv6 Address Plan", 2013, <http://www.ripe.net/lir-services/training/material/IPv6-for-LIRs-Training-Course/Preparing-an-IPv6-Addressing-Plan.pdf>.

Authors' Addresses

Philip Matthews
Nokia
600 March Road
Ottawa, Ontario K2K 2E6
Canada

Phone: +1 613-784-3139
Email: philip_matthews@magma.ca

Victor Kuarsingh
Cisco
88 Queens Quay
Toronto, ON M5J0B8
Canada

Email: victor@jvknet.com

v6ops
Internet-Draft
Intended status: Informational
Expires: April 20, 2016

M. Byerly
Fastly
M. Hite
Evernote
J. Jaeggli
Fastly

October 18, 2015

Close encounters of the ICMP type 2 kind (near misses with ICMPv6 PTB)
draft-ietf-v6ops-pmtud-ecmp-problem-06

Abstract

This document calls attention to the problem of delivering ICMPv6 type 2 "Packet Too Big" (PTB) messages to the intended destination (typically the server) in ECMP load balanced or anycast network architectures. It discusses operational mitigations that can be employed to address this class of failures.

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 April 20, 2016.

Copyright Notice

Table of Contents

1. Introduction

Operators of popular Internet services face complex challenges associated with scaling their infrastructure. One scaling approach is to utilize equal-cost multi-path (ECMP) routing to perform stateless distribution of incoming TCP or UDP sessions to multiple servers or to middle boxes such as load balancers. Distribution of traffic in this manner presents a problem when dealing with ICMP signaling. Specifically, an ICMP error is not guaranteed to hash via ECMP to the same destination as its corresponding TCP or UDP session. A case where this is particularly problematic operationally is path MTU discovery [RFC1981].

2. Problem

A common application for stateless load balancing of TCP or UDP flows is to perform an initial subdivision of flows in front of a stateful load balancer tier or multiple servers so that the workload becomes divided into manageable fractions of the total number of flows. The flow division is performed using ECMP forwarding and a stateless but sticky algorithm for hashing across the available paths (see [RFC2991] for background on ECMP routing). This nexthop selection for the purposes of flow distribution is a constrained form of anycast topology, where all anycast destinations are equidistant from the upstream router responsible for making the last next-hop forwarding decision before the flow arrives on the destination device. In this approach, the hash is performed across some set of available protocol headers. Typically, these headers may include all or a subset of (IPv6) Flow-Label, IP-source, IP-destination,

protocol, source-port, destination-port and potentially others such as ingress interface.

A problem common to this approach of distribution through hashing is impact on path MTU discovery. An ICMPv6 type 2 PTB message generated on an intermediate device for a packet sent from a server that is part of an ECMP load balanced service to a client will have the load balanced anycast address as the destination and hence will be statelessly load balanced to one of the servers. While the ICMPv6 PTB message contains as much of the packet that could not be forwarded as possible, the payload headers are not considered in the forwarding decision and are ignored. Because the PTB message is not identifiable as part of the original flow by the IP or upper layer packet headers, the results of the ICMPv6 ECMP hash calculation are unlikely to be hashed to the same nexthop as packets matching the TCP or UDP ECMP hash of the flow.

An example packet flow and topology follow. The packet for which the PTB message was generated was intended for the client.

ptb -> router ecmp -> nexthop L4/L7 load balancer -> destination

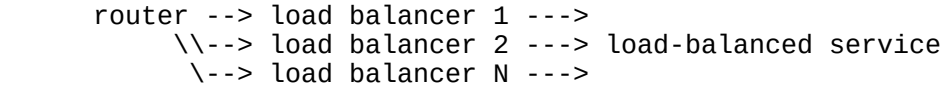

Figure 1

The router ECMP decision is used because it is part of the forwarding architecture, can be performed at line rate, and does not depend on shared state or coordination across a distributed forwarding system which may include multiple linecards or routers. The ECMP routing decision is deterministic with respect to packets having the same computed hash.

A typical case where ICMPv6 PTB messages are received at the load balancer is a case where the path MTU from the client to the load balancer is limited by a tunnel in which the client itself is not aware of.

Direct experience says that the frequency of PTB messages is small compared to total flows. One possible conclusion being that tunneled IPv6 deployments that cannot carry 1500 MTU packets are relatively rare. Techniques employed by clients such as happy-eyeballs may actually contribute some amelioration to the IPv6 client experience by preferring IPv4 in cases that might be identified as failures.

Still, the expectation of operators is that PMTUD should work and that unnecessary breakage of client traffic should be avoided.

A final observation regarding server tuning is that it is not always possible even if it is potentially desirable to be able to independently set the TCP MSS for different address families on some end-systems. On Linux platforms, advmss may be set on a per route basis for selected destinations in cases where discrimination by route is possible.

The problem as described does also impact IPv4; however implementation of RFC 4821 [RFC4821] TCP MTU probing, the ability to fragment on wire at tunnel ingress points and the relative rarity of sub-1500 byte MTUs that are not coupled to changes in client behavior (for example, endpoint VPN clients set the tunnel interface MTU accordingly to avoid fragmentation for performance reasons) makes the problem sufficiently rare that some existing deployments have choosen to ignore it.

3. Mitigation

Mitigation of the potential for PTB messages to be mis-delivered involves ensuring that an ICMPv6 error message is distributed to the same anycast server responsible for the flow for which the error is generated. With apppropiate hardware support, mitigation could be done by the mechanism hosts use to identify the flow; by looking into the payload of the ICMPv6 message (to determine which TCP flow it was associated with) before making a forwarding decision. Because the encapsulated IP header occurs at a fixed offset in the ICMP message it is not outside the realm of possibility that routers with sufficient header processing capability could parse that far into the payload. Employing a mediation device that handles the parsing and distribution of PTB messages after policy routing or on each load-balancer/server is a possibility.

Another mitigation approach is predicated upon distributing the PTB message to all anycast servers under the assumption that the one for which the message was intended will be able to match it to the flow and update the route cache with the new MTU and that devices not able to match the flow will discard these packets. Such distribution has potentially significant implications for resource consumption and for self-inflicted denial-of-service if not carefully employed. Fortunately, in real-world deployments we have observed that the number of flows for which this problem occurs is relatively small (example, 10 or fewer pps on 1Gb/s or more worth of https traffic in a real world deployment); sensible ingress rate limiters which will discard excessive message volume can be applied to protect even very

large anycast server tiers with the potential for fallout limited to circumstances of deliberate duress.

3.1. Alternative Mitigations

As an alternative, it may be appropriate to lower the TCP MSS to 1220 in order to accommodate 1280 byte MTU. We consider this undesirable as hosts may not be able to independently set TCP MSS by address-family thereby impacting IPv4, or alternatively that middle-boxes need to be employed to clamp the MSS independently from the end-systems. Potentially, extension headers might further alter the lower bound that the MSS would have to be set to, making clamping still more undesirable.

3.2. Implementation

1. Filter-based-forwarding matches next-header ICMPv6 type-2 and matches a next-hop on a particular subnet directly attached to 1 or more routers. The filter is policed to reasonable limits (we chose 1000pps, more conservative rates might be required in other implementations).

2. Filter is applied on input side of all external (internet or customer facing) interfaces.

3. A proxy located at the next-hop forwards ICMPv6 type-2 packets received at the next-hop to an Ethernet broadcast address (example ff:ff:ff:ff:ff:ff) on all specified subnets. This was necessitated by router inability (in IPv6) to forward the same packet to multiple unicast next-hops.

4. Anycasted servers receive the PTB error and process packet as needed.

A simple Python scapy script that can perform the ICMPv6 proxy reflection is included.

```
#!/usr/bin/python

from scapy.all import *

IFACE_OUT = ["p2p1", "p2p2"]

def icmp6_callback(pkt):
    if pkt.haslayer(IPv6) and (ICMPv6PacketTooBig in pkt) \
    and pkt[Ether].dst != 'ff:ff:ff:ff:ff:ff':
        del(pkt[Ether].src)
        pkt[Ether].dst = 'ff:ff:ff:ff:ff:ff'
        pkt.show()
        for iface in IFACE_OUT:
            sendp(pkt, iface=iface)

def main():
    sniff(prn=icmp6_callback, filter="icmp6 \
    and (ip6[40+0] == 2)", store=0)

if __name__ == '__main__':
    main()
```

This example script listens on all interfaces for IPv6 PTB errors being forwarded using filter-based-forwarding. It removes the existing Ethernet source and rewrites a new Ethernet destination of the Ethernet broadcast address. It then sends the resulting frame out the p2p1 and p2p2 interfaces which attached to vlans where our anycast servers reside.

3.2.1. Alternative Implementation

Alternatively, network designs in which a common layer 2 network exists on the ECMP hop could distribute the proxy onto the end systems, eliminating the need for policy routing. They could then rewrite the destination -- for example, using iptables before forwarding the packet back to the network containing all of the server or load balancer interfaces. This implmentation can be done entirely within the Linux iptables firewall. Because of the distributed nature of the filter, more conservative rate limits are required than when a global rate limit can be employed.

An example ip6tables / nftables rule to match icmp6 traffic, not match broadcast traffic, impose a rate limit of 10 pps, and pass to a target destination would resemble:

```
ip6tables -I INPUT -i lo -p icmpv6 -m icmpv6 --icmpv6-type 2/0 \
-m pkttype ! --pkt-type broadcast -m limit --limit 10/second \
-j TEE 2001:DB8::1
```

As with the scapy example, once the destination has been rewritten from a hardcoded ND entry to an Ethernet broadcast address -- in this case to an IPv6 documentation address -- the traffic will be reflected to all the hosts on the subnet.

4. Improvements

There are several ways that improvements could be made to the problem how to ECMP load balance of ICMPv6 PTB messages. little in the way of Internet protocol specification change is required, rather we forsee practical implemention change which insofar as we are aware does not exist in current router switch or layer3/4 load balancers. alternatively improved behavior on the part of client/server detection of path mtu in band could render the behavior of devices in the path irrelevant.

1. Routers with sufficient capacity within the lookup process could parse all the way through the L3 or L4 header in the ICMPv6 payload beginning at bit offset 32 of the ICMP header. By reordering the elements of the hash to match the inward direction of the flow, the PTB error could be directed to the same next-hop as the incoming packets in the flow.

2. The FIB (Forwarding Information Base) on the router could be programmed with a multicast distribution tree that included all of the necessary next-hops, and unicast ICMPv6 packets could be policy routed to these destinations.

3. Ubiquitous implementation of RFC 4821 [RFC4821] Packetization Layer Path MTU Discovery would probably go a long way towards reducing dependence on ICMPv6 PTB by end systems.

5. Acknowledgements

The authors would like to thank Marak Majkowsiki for contributing text, examples, and a very close review. The authors would like to thank Mark Andrews, Brian Carpenter, Nick Hilliard and Ray Hunter, for review.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

The employed mitigation has the potential to greatly amplify the impact of a deliberately malicious sending of ICMPv6 PTB messages. Sensible ingress rate limiting can reduce the potential for impact;

however, legitimate PMTUD messages may be lost once the rate limit is reached; the scenario is analogous to other cases where DOS traffic can crowd out legitimate traffic, however with a limited subset of overall traffic.

The proxy replication results in devices on the subnet not associated with the flow that generated the PTB, being recipients of the ICMPv6 PTB message; which contains a large fragment of the packet that exceeded the allowable MTU. This replication of the packet fragment could arguably result in information disclosure. Recipient machines should be in a common administrative domain.

8. Informative References

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, DOI 10.17487/RFC1981, August 1996, <http://www.rfc-editor.org/info/rfc1981>.

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <http://www.rfc-editor.org/info/rfc2991>.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <http://www.rfc-editor.org/info/rfc4821>.

Authors' Addresses

Matt Byerly
Fastly
Kapolei, HI
US

Email: suckawha@gmail.com

Matt Hite
Evernote
Redwood City, CA
US

Email: mhite@hotmail.com

Joel Jaeggli
Fastly
Mountain View, CA
US

Email: joelja@gmail.com

IPv6 Operations
Internet-Draft
Intended status: Informational
Expires: April 14, 2016

T. Anderson
Redpill Linpro

October 12, 2015

SIIT-DC: Stateless IP/ICMP Translation for IPv6 Data Centre Environments
draft-ietf-v6ops-siit-dc-03

Abstract

This document describes the use of the Stateless IP/ICMP Translation (SIIT) algorithm in an IPv6 Internet Data Centre (IDC). In this deployment model, traffic from legacy IPv4-only clients on the Internet is translated to IPv6 upon reaching the IDC operator's network infrastructure. From that point on, it may be treated the same as traffic from native IPv6 end users. The IPv6 endpoints may be numbered using arbitrary (non-IPv4-translatable) IPv6 addresses. This facilitates a single-stack IPv6-only network infrastructure, as well as efficient utilisation of public IPv4 addresses.

The primary audience is IDC operators who are deploying IPv6, running out of available IPv4 addresses, and/or feel that dual stack causes undesirable operational complexity.

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 April 14, 2016.

Table of Contents

1. Introduction

Historically, dual stack [RFC4213] [RFC6883] has been the recommended way to transition from a legacy IPv4-only environment to one capable of serving IPv6 users. However, for IDC operators, dual stack operation has a number of disadvantages compared to single stack operation. In particular, running two protocols rather than one results in increased complexity and operational overhead, with little return on investment for as long as large parts of the public Internet remains predominantly IPv4-only. Furthermore, the dual stack approach does not in any way help with the depletion of the IPv4 address space, which at the time of writing is a pressing concern in most parts of the world.

Therefore, some IDC operators may instead prefer an approach in which they only need to operate one protocol in the data centre as they prepare for the future. SIIT-DC is one such approach. Its design goals include:

- Promote the deployment of native IPv6 services (cf. [RFC6540]).
- Provide IPv4 service availability for legacy users with no loss of performance or functionality.
- To ensure that that the legacy users' IPv4 addresses remain visible to the nodes and applications located in the IPv6 network.
- To conserve and maximise the utilisation of the operator's public IPv4 addresses.
- To avoid introducing more complexity than absolutely necessary, especially on the nodes and applications.
- To be easy to scale and deploy in a fault-tolerant manner.

The following subsections elaborates on how SIIT-DC meets these goals.

1.1. Single Stack IPv6 Operation

SIIT-DC allows IDC operators to build their infrastructure and applications on an IPv6-only foundation. IPv4 end-user connectivity becomes a service provided by the network, which systems administration and application development staff do not need to concern themselves with. This promotes universal IPv6 deployment for the IDC operator's services and applications.

SIIT-DC requires no special support or change from the underlying IPv6 infrastructure, it is compatible with all standard IPv6 networks. Traffic between IPv6-enabled end users and IPv6-enabled

services will always be transported native end-to-end; SIIT-DC does not intercept or handle native IPv6 traffic at all.

When the day comes to discontinue all support for IPv4, no change needs to be made to the overall architecture - it's only a matter of shutting off the SIIT-DC Border Relays (BRs). Operators who deploy native IPv6 along with SIIT-DC will thus avoid requiring any future migration or deployment projects relating to IPv6 deployment and/or IPv4 sun-setting.

1.2. Stateless Operation

Unlike other solutions that provide either dual stack availability to single-stack services (e.g., Stateful NAT64 [RFC6146] and Layer-4/7 proxies), or that provide conservation of IPv4 addresses (e.g., NAPT44 [RFC3022]), SIIT-DC does not maintain any state associated with individual connections or flows. In this sense it operates exactly like a regular IP router, and has similar scaling properties - the limiting factors are packets per second and bandwidth. The number of concurrent flows and flow initiation rates are irrelevant for performance.

This not only allows individual BRs to easily attain "line rate" performance, it also allows for per-packet load balancing between multiple BRs using Equal-Cost Multipath Routing [RFC2991]. Asymmetric routing is also acceptable, which makes it easy to avoid sub-optimal traffic patterns; the prefixes involved may be anycasted from all the BRs in the provider's network, thus ensuring that the most optimal path through the network is used, even where the optimal path in one direction differs from the optimal path in the opposite direction.

Finally, stateless operation means that high availability is easily achieved. If a BR should fail, its traffic can be re-routed onto another BR using a standard IP routing protocol. This does not impact existing flows any more than what any other IP re-routing event would.

1.3. IPv4 Address Conservation

In most parts of the world, it is difficult or even impossible to obtain generously sized IPv4 delegation from the Internet Numbers Registry System [RFC7020]. The resulting scarcity in turn impacts individual end users and operators, which might be forced to purchase IPv4 addresses from other operators in order to cover their needs. This process can be risky to business continuity, in the case no suitable block for sale can be located, and/or turn out to be prohibitively expensive. In spite of this, an IDC operator will find

that providing IPv4 service remains essential, as a large share of the Internet end users still do not have IPv6 connectivity.

A key goal of SIIT-DC is to help reduce a data centre operator's IPv4 address requirement to the absolute minimum, by allowing the operator to remove them entirely from nodes and applications that do not need to communicate with endpoints in the IPv4 Internet. One example would be servers that are operating in a supporting/back-end role and only communicates with other servers (database servers, file servers, and so on). Another example would be the network infrastructure itself (router-to-router links, loopback addresses, and so on). Furthermore, as LAN prefix sizes must always be rounded up to the nearest power of two (or larger, if one reserves space for future growth), even more IPv4 addresses will often end up being wasted without even being used.

With SIIT-DC, the operator can remove these valuable IPv4 addresses from his back-end servers and network infrastructure, and reassign them to the SIIT-DC service as IPv4 Service Addresses. There exists no requirement that IPv4 Service Addresses are assigned in an aggregated manner, so there is nothing lost due to infrastructure overhead; every single IPv4 address assigned to SIIT-DC can be used an IPv4 Service Address.

1.4. Clients' IPv4 Source Addresses Visible to Applications

SIIT-DC uses the [RFC6052] algorithm to map the entire end-user's IPv4 source address into an predefined IPv6 Translation Prefix. This ensures that there is no loss of information; the end-user's IPv4 source address remains available to the application located in the IPv6 network, allowing it to perform tasks like Geo-Location, logging, abuse handling, and so forth.

1.5. Compatible with Standard IPv4 and IPv6 Stacks

Except for the introduction of the BRs themselves, no change to the network, nodes, applications, or anything else is required in order to support SIIT-DC. SIIT-DC is practically invisible from the point of view of the IPv4 clients, the IPv6 nodes, the IPv6 data centre network, and the IPv4 Internet. SIIT-DC interoperates with all standards-compliant IPv4 or IPv6 stacks.

2. Terminology

This document makes use of the following terms:

SIIT-DC Border Relay (BR)

A device or a logical function that performs stateless protocol translation between IPv4 and IPv6. It MUST do so in accordance with [RFC6145] and [I-D.ietf-v6ops-siit-eam].

SIIT-DC Edge Relay (ER)
A device or logical function that provides "native" IPv4 connectivity to IPv4-only devices or application software. It is very similar in function to a BR, but is typically located close to the IPv4-only component(s) it is supporting rather than on the IDC's outer network border. The ER is an optional component of SIIT-DC. It is discussed in more detail in [I-D.ietf-v6ops-siit-dc-2xlat].

IPv4 Service Address
An IPv4 address representing a node or service located in an IPv6 network. It is coupled with an IPv6 Service Address using an EAM. Packets sent to this address is translated to IPv6 by the BR, and possibly back to IPv4 by an ER, before reaching the node or service.

IPv4 Service Address Pool
One or more IPv4 prefixes routed to the BR's IPv4 interface. IPv4 Service Addresses are allocated from this pool. That this does not necessarily have to be a "pool" per se, as it could also be one or more host routes (whose prefix length is equal to /32). The purpose of using a pool rather than host routes is to facilitate IPv4 route aggregation and ease provisioning of new IPv4 Service Addresses.

IPv6 Service Address
An IPv6 address assigned to an application, node, or service; either directly or indirectly (through an ER). It is coupled with an IPv4 Service Address using an EAM. IPv4-only clients communicates with the IPv6 Service Address through SIIT-DC.

Explicit Address Mapping (EAM)
A bi-directional coupling between an IPv4 Service Address and an IPv6 Service Address configured in a BR or ER. When translating between IPv4 and IPv6, the BR/ER changes the address fields in the translated packet's IP header according to any matching EAM. The EAM algorithm is specified in [I-D.ietf-v6ops-siit-eam].

Translation Prefix

An IPv6 prefix into which the entire IPv4 address space is mapped, according to the algorithm in [RFC6052]. The Translation Prefix is routed to the BR's IPv6 interface. When translating between IPv4 and IPv6, an BR/ER will insert/remove the Translation Prefix into/from the address fields in the translated packet's IP header, unless an EAM exists for the IP address that is being translated.

IPv4-translatable IPv6 addresses
As defined in Section 1.3 of [RFC6052].

IDC
Short for "Internet Data Centre"; a data centre whose main purpose is to deliver services to the public Internet, the use case SIIT-DC is primarily targeted at. IDCs are typically operated by Internet Content Providers or Managed Services Providers.

SIIT
The Stateless IP/ICMP Translation algorithm, as specified in [RFC6145].

XLAT
Short for "Translation". Used in figures to indicate where a BR/ER uses SIIT [RFC6145] to translate IPv4 packets to IPv6 and vice versa.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Architectural Overview

This section describes the basic SIIT-DC architecture.

SIIT-DC Architecture

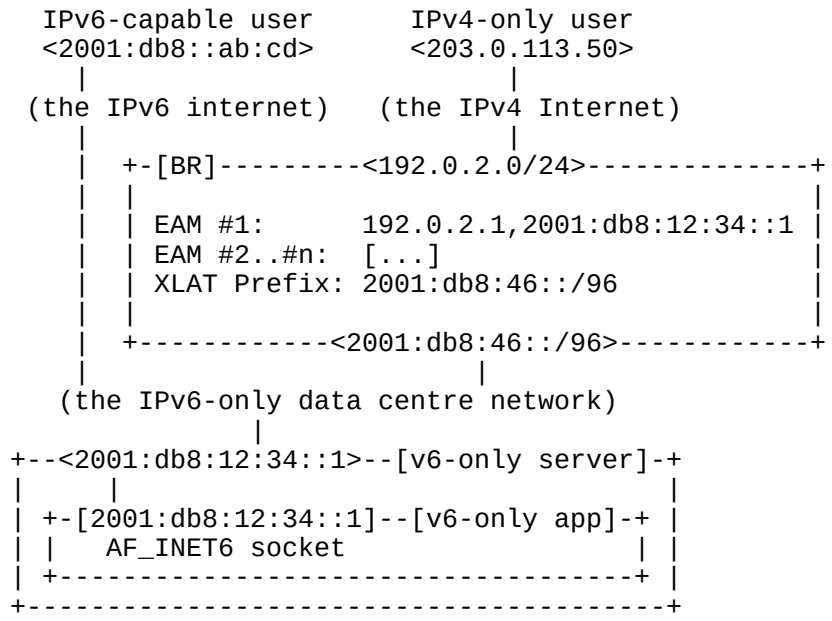

Figure 1

In Figure 1, 192.0.2.0/24 is the IPv4 Service Address Pool. Individual IPv4 Service Addresses are assigned from this prefix, and traffic destined for it is routed to the BR's IPv4-facing network interface. There are no restrictions on how many IPv4 Service Address Pools are used or their prefix length, as long as they are all routed to the BR's IPv4-facing network interface.

When translating packets between IPv4 and IPv6, the BR uses the EAM to replace any occurrence of the IPv4 Service Address (192.0.2.1) with its corresponding IPv6 Service Address (2001:db8:12:34::1). Addresses that do not match any EAM configured in the BR are translated by inserting or removing the Translation Prefix (2001:db8:46::/96), cf. Section 2.2 of [RFC6052].

The BR can be deployed as a separate device or as a logical function in another multi-purpose device, such as an IP router. Any number of BRs may exist simultaneously in the IDC's network infrastructure, as long as they all configured with the same Translation Prefix and an identical EAM Table.

The IPv6 Service Address of should be registered in DNS using an "IN AAAA" record, while its corresponding IPv4 Service Address should be registered using an "IN A" record. This ensures that IPv6-capable clients access the application/service directly using its native IPv6 end-to-end, while IP4-only clients will access it through SIIT-DC.

3.1. Packet Flow

In this example, the "IPv4-only user" from Figure 1 initiates a connection to the application running on the IPv6-only server. After first having looked up the "IN A" record in DNS, the user starts by transmitting an TCP SYN packet to the IPv4 Service Address. This IPv4 packet is routed to the BR, and is there translated to IPv6 as follows:

IPv4 to IPv6 translation

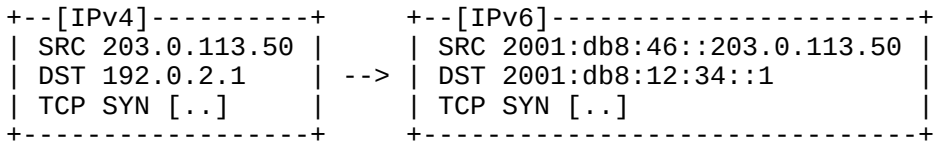

Figure 2

The resulting IPv6 packet is routed to the IPv6-only server, which processes and responds to it as if it had been a native IPv6 packet all along. The server's IPv6 response packet is then routed back to the BR, where it is translated back to IPv4 as follows:

IPv6 to IPv4 translation

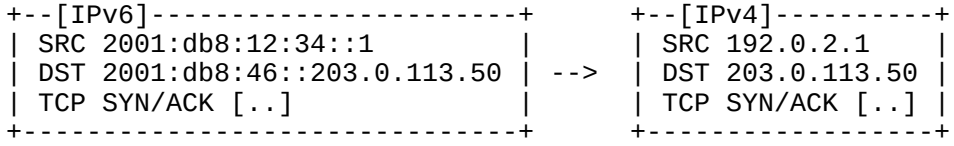

Figure 3

It is important to note that neither the IPv4 client nor the IPv6 server/application need any special support to participate in SIIT-DC. However, the application may optionally be taught to extract the embedded IPv4 source address from incoming IPv6 packets with source addresses within the Translation Prefix. This will allow it to perform IPv4-specific tasks such as Geo-Location, logging, abuse handling, and so on.

4. Deployment Considerations and Guidelines

4.1. Application/Device Support for IPv6

SIIT-DC as described in this document requires that the application (and/or the node the application is located on) supports IPv6 networking, and that it has no dependency on local IPv4 network connectivity.

SIIT-DC can however support legacy IPv4-dependent applications and nodes through the introduction of an ER. The ER provides the legacy application or node with seemingly native IPv4 Internet connectivity, so that it may operate correctly in an otherwise IPv6-only network environment. This approach is described in more detail in [I-D.ietf-v6ops-siit-dc-2xlat].

4.2. Application Support for NAT

The operator should carefully examine whether or not the application protocols he would like to use SIIT-DC with are able to operate in a network environment where rewriting of IP addresses occur. In general, if an application layer protocol works correctly through standard NAT44 (see [RFC3235]), it will most likely work correctly through SIIT-DC as well.

Higher-level protocols that embed IP addresses as part of their payload are particularly problematic [RFC2663] [RFC2993] [RFC3022]. One well-known example of such a protocol is FTP [RFC0959]. Such protocols can be made to work with SIIT-DC through the introduction of an ER, which provides end-to-end IPv4 address transparency by reversing the translations performed by the BR before passing the packets to the NAT-incompatible application. This approach is described in more detail in [I-D.ietf-v6ops-siit-dc-2xlat].

4.3. Application Communication Pattern

SIIT-DC is best suited for traditional client/server applications where IPv4-only clients on the Internet initiate traffic towards an IPv6-only service, which in turn is passively listening for inbound traffic and responding as necessary. In this case, an IPv4 client looks exactly like an native IPv6 client from the IPv6 service's point of view, and thus does not require any special treatment. One particularly common application protocol that follows this client/server communication pattern, and thus is ideally suited for use with SIIT-DC, is HTTP [RFC7230].

It is also possible to combine SIIT-DC with DNS64 [RFC6147] in order to allow an IPv6-only application to initiate communication with

IPv4-only nodes through SIIT-DC. However, in this case, care must be taken so that all outgoing communication is sourced from an IPv6 Service Address that is found in an EAM configured in the BR. If another address is used, the BR will most likely be unable to translate it to IPv4, causing the packet to be discarded. This could be prevented by altering the Default Address Selection Policy Table [RFC6724] on the IPv6 node.

An alternative approach to the above would be to place an ER in front of the application in question, as described [I-D.ietf-v6ops-siit-dc-2xlat]. This provides the application with seemingly native IPv4 connectivity, which it may use freely for bi-directional communication with the IPv4 Internet. An application or node located behind an ER does not need to worry about selecting a specific source address, as it will only have valid options available.

4.4. Choice of Translation Prefix

Either a Network-Specific Prefix (NSP) from the provider's own IPv6 address space or the IANA-allocated Well-Known Prefix 64:ff9b::/96 (WKP) may be used. From a technical point of view, both work equally well. However, only a single WKP exists, so if a provider would like to deploy more than one instance of SIIT-DC in his network, or another translation technology such as Stateful NAT64 [RFC6146], the operator will be forced to use an NSP for all but one of those deployments.

Another consideration is that the WKP cannot be used in inter-domain routing. By using an NSP instead, SIIT-DC will support a deployment where the BR and the IPv6 Service Address are located in different Autonomous Systems.

The Translation Prefix may use any of the lengths described in Section 2.2 of [RFC6052], but /96 has two distinct advantages over the others. First, converting it to IPv4 can be done in a single operation by simply stripping off the first 96 bits; second, it allows for IPv4 addresses to be embedded directly into the text representation of an IPv6 address using the familiar dotted quad notation, e.g., "2001:db8::198.51.100.10" (cf. Section 2.4 of [RFC6052])), instead of being converted to hexadecimal notation. This makes it easier to write IPv6 ACLs and similar that match translated endpoints in the IPv4 Internet.

For the reasons discussed above, this document recommends that an NSP with a prefix length of /96 is used. Section 3.3 of [RFC6052] discusses the choice of translation prefix in more detail.

4.5. Routing Considerations

The prefixes that constitute the IPv4 Service Address Pool and the IPv6 Translation Prefix may be routed to the BRs as any other IPv4 or IPv6 route in the provider's network. If more than one BR is being deployed, it is recommended that a routing protocol (IGP) used to advertise the routes within the provider's network. This will ensure that the traffic that is to be translated will reach the closest BR, reducing or eliminating sub-optimal traffic patterns, as well as providing high availability: Should one BR fail, the IGP will automatically redirect the traffic to the closest alternate BR.

4.6. Location of the SIIT-DC Border Relays

The goal of SIIT-DC is to facilitate a true IPv6-only application and network architecture, with the sole exception being the IPv4 interfaces of the BRs and the network infrastructure required to connect the BRs to the IPv4 Internet. Therefore, the BRs must be located somewhere between the IPv4 Internet and the application delivery stack - which includes all servers, load balancers, firewalls, intrusion detection systems, and similar devices that are processing traffic to a greater extent than merely forwarding it.

It is optimal to place the BRs as close as possible to the direct path between the location of the IPv6 Service Address and the end users. If the closest BR was located a long way from the direct path, all packets in both directions must make a detour in order to traverse the BR. This would increase the RTT between the service and the end user by by two times the extra latency incurred by the detour, as well as cause unnecessary load on the network links on the detour path.

Where possible, it is beneficial to implement the BRs as a logical function within the routers would have handled the traffic anyway, had the topology been dual stacked. This way, a SIIT-DC deployment does not require separate networks ports (which might become saturated and impact the service quality), nor will it require extra rack space and energy. Some particularly good choices of the location could be within an IDC's access routers, or within the Autonomous System's border routers.

Finally, another possibility is that the IDC operator outsources the SIIT-DC service to another entity, for example his upstream ISP. Doing so allows the IDC operator to build a true IPv6-only infrastructure.

4.7. Migration from Dual Stack

While this document mainly discusses the use of IPv6-only nodes and applications, it is important to note that SIIT-DC is fully compatible with dual stack infrastructures, including dual stack nodes and applications.

Thus, migrating a dual-stacked service to an IPv6-only one where SIIT-DC provides the IPv4 Internet connectivity is easy. The operator would start out by designating the service's current native IPv6 address as the IPv6 Service Address, and assign it a corresponding IPv4 Service Address. At this point, the service will respond on both its old (native) IPv4 address, and the SIIT-DC IPv4 Service Address. The operator may now move traffic from the former to the latter by changing the service's "IN A" DNS record. Once all IPv4 traffic has been successfully moved to SIIT-DC, the old IPv4 address may be reclaimed.

4.8. Translation of ICMPv6 Errors to IPv4

In response to an IPv4 packet subsequently translated to IPv6 by the BR, an IPv6 router in the IDC network may need to transmit an ICMPv6 error back to the origin IPv4 node. By default, such an ICMPv6 error will most likely be discarded by the BR, unless the source address of the ICMPv6 error happens to be a IPv4-translatable IPv6 address or covered by an EAM.

To facilitate reliable delivery of such ICMPv6 errors, an SIIT-DC operator SHOULD implement the recommendations in [RFC6791] in the BRs.

4.9. MTU and Fragmentation

There are some key differences between IPv4 and IPv6 relating to packet sizes and fragmentation that one MUST consider when deploying SIIT-DC. They result in a few problematic corner cases, which can be dealt with in a few different ways. The following subsections will discuss these in detail, and provide operational guidance.

In particular, the operator may find that relying on fragmentation in the IPv6 domain is undesired or even operationally impossible [I-D.taylor-v6ops-fragdrop]. For this reason, the recommendations in this section seeks to minimise the use of IPv6 fragmentation.

Unless otherwise stated, the following subsections assume that the MTU in both the IPv4 and IPv6 domains is 1500 bytes.

4.9.1. IPv4/IPv6 Header Size Difference

The IPv6 header is up to 20 bytes larger than the IPv4 header. This means that a full-size 1500 bytes large IPv4 packet cannot be translated to IPv6 without being fragmented, otherwise it would likely have resulted in a 1520 bytes large IPv6 packet.

If the transport protocol used is TCP, this is generally not a problem, the IPv6 node will advertise a TCP MSS of 1440 bytes during the initial TCP handshake. This causes the IPv4 clients to never send larger packets than what can be translated to a single full-size IPv6 packet, eliminating any need for fragmentation.

For other transport protocols, full-size IPv4 packets with the DF flag cleared will need to be fragmented by the BR. This may be avoided by increasing the Path MTU between the BR and the IPv6 nodes to 1520 bytes or greater. If this is done, the MTU on the IPv6 nodes themselves SHOULD NOT be increased accordingly, as doing so would cause them to undergo Path MTU Discovery for all destinations on the IPv6 Internet. The nodes MUST however be able to accept and process incoming packets larger than their own MTU. If the nodes' IPv6 implementation allows the initial Path MTU to be set differently for specific destinations, it MAY be increased to 1520 for destinations within the Translation Prefix specifically.

4.9.2. IPv6 Atomic Fragments

In keeping with the fifth paragraph of Section 4 of [RFC6145], a stateless translator like a BR will by default add an IPv6 Fragmentation header to the resulting IPv6 packet when translating an IPv4 packet with the Don't Fragment flag set to 0. This happens even though the resulting IPv6 packet isn't actually fragmented into several pieces, resulting in an IPv6 Atomic Fragment [RFC6946]. These Atomic Fragments are generally not useful in an IDC environment, and it is therefore recommended that this behaviour is disabled in the BRs. To this end, Section 4 of [RFC6145] notes that the "translator MAY provide a configuration function that allows the translator not to include the Fragment Header for the non-fragmented IPv6 packets".

Note that IPv6 Atomic Fragments are currently being deprecated by RFC6145bis [I-D.bao-v6ops-rfc6145bis]. As a result, a BR that conforms to the updated standard is required to behave as recommended above.

In IPv6, the Identification value is located inside the Fragmentation header. That means that if the generation of IPv6 Atomic Fragments is disabled, the IPv4 Identification value will be lost during translation to IPv6. This could potentially confuse some diagnostic tools.

4.9.3. Minimum Path MTU Difference Between IPv4 and IPv6

Section 5 of [RFC2460] specifies that the minimum IPv6 link MTU is 1280 bytes. Therefore, an IPv6 node can reasonably assume that if it transmits an IPv6 packet that is 1280 bytes or smaller, it is guaranteed to reach its destination without requiring fragmentation or invoking the Path MTU Discovery algorithm [RFC1981]. However, this assumption might prove false if the destination is an IPv4 node reached through a protocol translator such as a BR, as the minimum IPv4 link MTU is 68 bytes. See Section 3.2 of [RFC0791].

Section 5.1 of [RFC6145] specifies that a stateless translator should set the IPv4 Don't Fragment flag to 1 when it translates a non-fragmented IPv6 packet to IPv4. This means that when the path to the destination IPv4 node contains an IPv4 link with an MTU smaller than 1260 bytes (which corresponds to an IPv6 MTU smaller than 1280 bytes, cf. Section 4.9.1), the Path MTU Discovery algorithm will be invoked, even if the original IPv6 packet was only 1280 bytes large. This happens as a result of the IPv4 router connecting to the IPv4 link with the small MTU returning an ICMPv4 Need To Fragment error with an MTU value smaller than 1260, which in turns is translated by the BR to an ICMPv6 Packet Too Big error with an MTU value smaller than 1280 which is then transmitted to the origin IPv6 node.

When an IPv6 node receives an ICMPv6 Packet Too Big error indicating an MTU value smaller than 1280, the last paragraph of Section 5 of [RFC2460] gives it two choices on how to proceed:

- It may reduce its Path MTU value to the value indicated in the Packet Too Big, i.e., limit the size of subsequent packets transmitted to that destination to the indicated value. This approach causes no problems for the SIIT-DC function, as it simply allows Path MTU Discovery to work transparently across the BR.

- It may reduce its Path MTU value to exactly 1280, and in addition include a Fragmentation header in subsequent packets sent to that destination. In other words, the IPv6 node will start emitting Atomic Fragments. The Fragmentation header signals to the the BR that the Don't Fragment flag should be set to 0 in the resulting IPv4 packet, and it also provides the Identification value.

If the use of the IPv6 Fragmentation header is problematic, and the operator has IPv6 nodes that implement the second option above, the operator should consider enabling the functionality described as the "second approach" in Section 6 of [RFC6145]. This functionality changes the BR's behaviour as follows:

- When translating ICMPv4 Need To Fragment to ICMPv6 Packet Too Big, the resulting packet will never contain an MTU value lower than 1280. This prevents the IPv6 nodes from generating Atomic Fragments.

- When translating IPv6 packets smaller than or equal to 1280 bytes, the Don't Fragment flag in the resulting IPv4 packet will be set to 0. This ensures that in the eventuality that the path contains an IPv4 link with an MTU smaller than 1260, the IPv4 router connected to that link will have the responsibility to fragment the packet before forwarding it towards its destination.

In summary, this approach could be seen as prompting the IPv4 protocol itself to provide the "link-specific fragmentation and reassembly at a layer below IPv6" required for links that "cannot convey a 1280-octet packet in one piece", to paraphrase Section 5 of [RFC2460].

Note that IPv6 Atomic Fragments are currently being deprecated by RFC6145bis [I-D.bao-v6ops-rfc6145bis]. As a result, a BR that conforms to the updated standard is required to behave as suggested above.

4.10. IPv4-translatable IPv6 Service Addresses

SIIT-DC is designed so that the IPv6 Service Addresses are not required to be IPv4-translatable IPv6 addresses. Section 2 of [I-D.ietf-v6ops-siit-eam] discusses why it is desirable to avoid requiring the use of IPv4-translatable IPv6 addresses.

It is however quite possible to deploy SIIT-DC in combination with IPv4-translatable IPv6 Service Addresses. The primary benefits in doing so are:

- The operator is not required to provision EAMs for IPv4-translatable IPv6 Service Addresses onto the BR/ERs.

- [RFC6145] translation can be performed in a checksum-neutral manner, cf. Section 4.1 of [RFC6052].

The trade-off is that the IPv4-translatable IPv6 Service Addresses must be configured on the IPv6 nodes, and the applications must be set up to use them - likely in addition to their primary (non-IPv4-translatable) IPv6 addresses. The IPv4-translatable IPv6 Service Addresses must also be routed from the BR through the IDC's IPv6 network infrastructure to the nodes on which they are assigned. This essentially requires the entire IPv6 infrastructure to be made aware of and handle translated IPv4 traffic as a special case, which

significantly increases complexity. As previously described in Section 1.1, avoiding such drawbacks is a design goal of SIIT-DC. The use of IPv4-translatable IPv6 Service Addresses is therefore discouraged.

5. Acknowledgements

The author would like to thank the following individuals for their contributions, suggestions, corrections, and criticisms: Fred Baker, Cameron Byrne, Brian E Carpenter, Ross Chandler, Tobias Gondrom, Christer Holmberg, Dagfinn Ilmari Mannsaaker, Lars Olafsen, Stig Sandbeck Mathisen, Knut A. Syed, Qin Wu, Andrew Yourtchenko.

6. IANA Considerations

This draft makes no request of the IANA.

7. Security Considerations

7.1. Mistaking the Translation Prefix for a Trusted Network

If a Network-Specific Prefix from the provider’s own address space is chosen for the translation prefix, as recommended in Section 4.4, care MUST be taken if the translation service is used in front of services that have application-level ACLs that distinguish between the operator’s own networks and the Internet at large, as traffic from translated IPv4 end users on the Internet might appear to be originating from the provider’s own network. It is therefore important that the translation prefix is treated the same as the Internet at large, rather than as a trusted network.

In order to alleviate this problem, the operator may opt to use a Translation Prefix that is distinct from and not a subset of the IPv6 prefixes used elsewhere in the network infrastructure.

8. References

8.1. Normative References

[I-D.ietf-v6ops-siit-eam]
Anderson, T. and A. Leiva, "Explicit Address Mappings for Stateless IP/ICMP Translation", draft-ietf-v6ops-siit-eam-01 (work in progress), June 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, DOI 10.17487/RFC6052, October 2010, <http://www.rfc-editor.org/info/rfc6052>.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, DOI 10.17487/RFC6145, April 2011, <http://www.rfc-editor.org/info/rfc6145>.

[RFC6791] Li, X., Bao, C., Wing, D., Vaithianathan, R., and G. Huston, "Stateless Source Address Mapping for ICMPv6 Packets", RFC 6791, DOI 10.17487/RFC6791, November 2012, <http://www.rfc-editor.org/info/rfc6791>.

8.2. Informative References

[I-D.bao-v6ops-rfc6145bis]
Bao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm (rfc6145bis)", draft-bao-v6ops-rfc6145bis-02 (work in progress), October 2015.

[I-D.ietf-v6ops-siit-dc-2xlat]
Anderson, T. and S. Steffann, "SIIT-DC: Dual Translation Mode", draft-ietf-v6ops-siit-dc-2xlat-01 (work in progress), June 2015.

[I-D.taylor-v6ops-fragdrop]
Jaeggli, J., Colitti, L., Kumari, W., Vyncke, E., Kaeo, M., and T. Taylor, "Why Operators Filter Fragments and What It Implies", draft-taylor-v6ops-fragdrop-02 (work in progress), December 2013.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[RFC0959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, DOI 10.17487/RFC0959, October 1985, <http://www.rfc-editor.org/info/rfc959>.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, DOI 10.17487/RFC1981, August 1996, <http://www.rfc-editor.org/info/rfc1981>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <http://www.rfc-editor.org/info/rfc2663>.

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <http://www.rfc-editor.org/info/rfc2991>.

[RFC2993] Hain, T., "Architectural Implications of NAT", RFC 2993, DOI 10.17487/RFC2993, November 2000, <http://www.rfc-editor.org/info/rfc2993>.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <http://www.rfc-editor.org/info/rfc3022>.

[RFC3235] Senie, D., "Network Address Translator (NAT)-Friendly Application Design Guidelines", RFC 3235, DOI 10.17487/RFC3235, January 2002, <http://www.rfc-editor.org/info/rfc3235>.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, DOI 10.17487/RFC4213, October 2005, <http://www.rfc-editor.org/info/rfc4213>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <http://www.rfc-editor.org/info/rfc6146>.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <http://www.rfc-editor.org/info/rfc6147>.

[RFC6540] George, W., Donley, C., Liljenstolpe, C., and L. Howard, "IPv6 Support Required for All IP-Capable Nodes", BCP 177, RFC 6540, DOI 10.17487/RFC6540, April 2012, <http://www.rfc-editor.org/info/rfc6540>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC6883] Carpenter, B. and S. Jiang, "IPv6 Guidance for Internet Content Providers and Application Service Providers", RFC 6883, DOI 10.17487/RFC6883, March 2013, <http://www.rfc-editor.org/info/rfc6883>.

[RFC6946] Gont, F., "Processing of IPv6 "Atomic" Fragments", RFC 6946, DOI 10.17487/RFC6946, May 2013, <http://www.rfc-editor.org/info/rfc6946>.

[RFC7020] Housley, R., Curran, J., Huston, G., and D. Conrad, "The Internet Numbers Registry System", RFC 7020, DOI 10.17487/RFC7020, August 2013, <http://www.rfc-editor.org/info/rfc7020>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

Appendix A. Complete SIIT-DC IDC topology example

Figure 4 attempts to "tie it all together" and show a more complete SIIT-DC topology, in order to better demonstrate its advantageous properties discussed in Section 1. These are discussed in more detail below.

Example SIIT-DC IDC topology

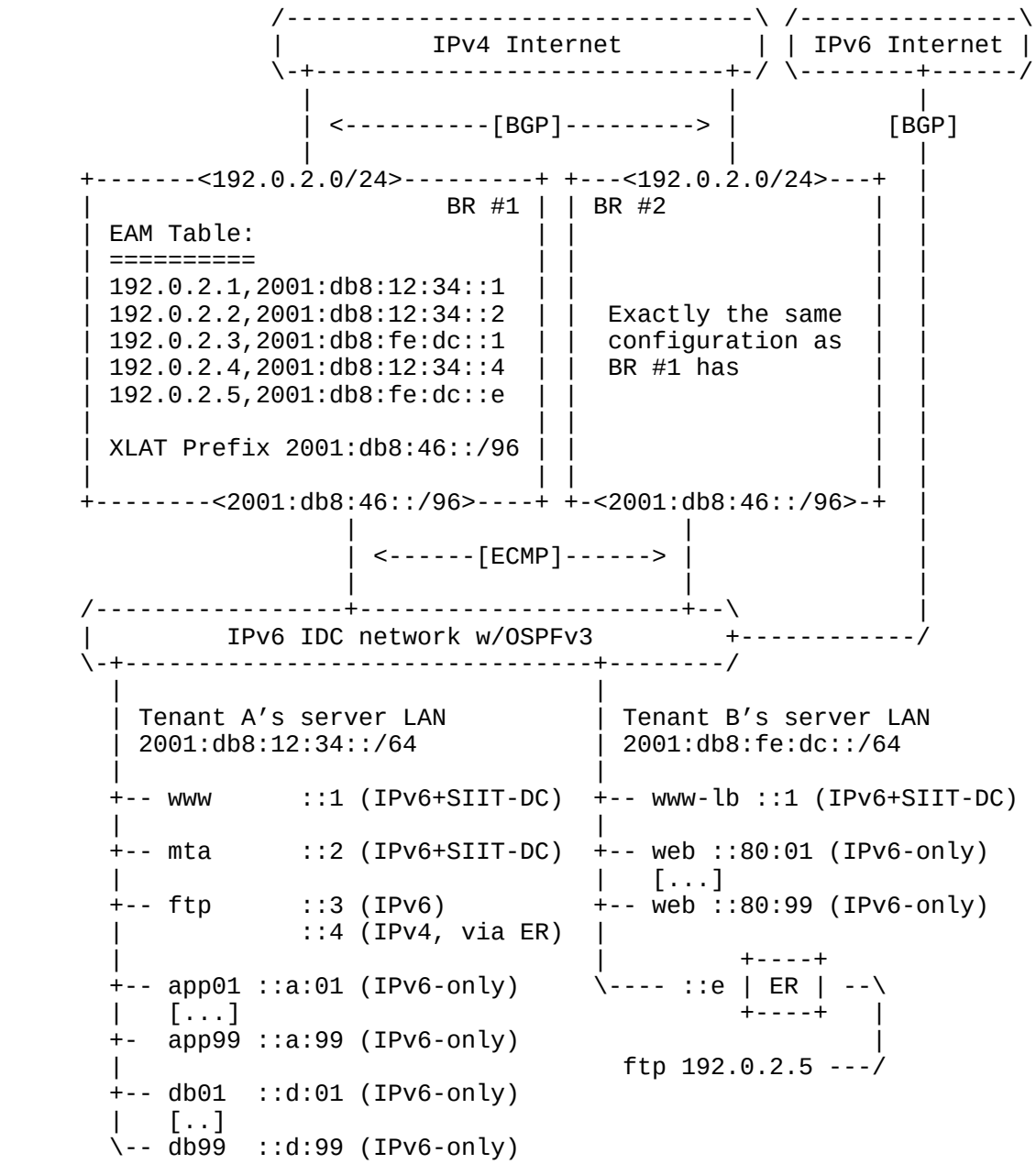

Figure 4

Single Stack IPv6 Operation

As discussed in Section 1.1, SIIT-DC facilitates an IPv6-only IDC network infrastructure. The only places where IPv4 is absolutely required is between the BRs and the IPv4 Internet, and between any ERs and the IPv4-only applications or devices they are serving (illustrated here as the two tenants' FTP servers). The figure also illustrates how SIIT-DC does not interfere with native IPv6; when there is no longer a need to support IPv4 clients, the BRs may be decommissioned without causing any impact to native IPv6 traffic.

Stateless Operation
As discussed in Section 1.2, SIIT-DC operates in a stateless fashion. In the illustration, both BRs are simultaneously advertising (i.e., anycasting) the IPv4 Service Address Pool and the IPv6 Translation Prefix, so incoming traffic from the IPv4 Internet may arrive at either of the BRs, while outgoing IPv6 traffic destined for IPv4 endpoints are load balanced between them using Equal-Cost Multipath Routing. No continuous state synchronisation between the two BRs occurs. Should one of the BRs fail, the BGP and OSPF protocols will ensure that traffic converges on the remaining BR. Existing sessions will not be disrupted, beyond any disruption caused by the BGP/OSPF convergence process itself.

IPv4 Address Conservation
As discussed in Section 1.3, SIIT-DC conserves the IDC operator's IPv4 address space. Even though the two customers in the example above have several hundred servers, the majority of them are not used to run services made available directly from the Internet, and therefore do not need to consume IPv4 addresses. The IDC network infrastructure consumes no IPv4 addresses, either. Finally, the IPv4 addresses that are assigned to the SIIT-DC function as IPv4 Service Address Pools may assigned with 100% efficiency, one address at a time; there is no requirement to assign multiple addresses to a single customer in a contiguous block.

Application support
As discussed in Section 1.5, as long as the application protocol is translation-friendly (illustrated here with HTTP and SMTP), it will work with SIIT-DC without requiring any special adaptation. Furthermore, translation-unfriendly applications (illustrated here with FTP) will also work when located behind an ER [I-D.ietf-v6ops-siit-dc-2xlat]. Tenant A's FTP server illustrates how an ER may be located in the networking stack of a node, while Tenant B's FTP server illustrates how the ER may be deployed as a network service. The latter approach enables SIIT-DC to support IPv4-only nodes/devices.

Author's Address

Tore Anderson
Redpill Linpro
Vitaminveien 1A
0485 Oslo
Norway

Phone: +47 959 31 212
Email: tore@redpill-linpro.com
URI: http://www.redpill-linpro.com

IPv6 Operations
Internet-Draft
Intended status: Informational
Expires: April 14, 2016

T. Anderson
Redpill Linpro
S. Steffann
S.J.M. Steffann Consultancy

October 12, 2015

SIIT-DC: Dual Translation Mode
draft-ietf-v6ops-siit-dc-2xlat-02

Abstract

This document describes an extension of the Stateless IP/ICMP Translation for IPv6 Internet Data Centre Environments architecture (SIIT-DC), which allows applications, protocols, or nodes that are incompatible with IPv6, and/or Network Address Translation to operate correctly in an SIIT-DC environment. This is accomplished by introducing a new component called an SIIT-DC Edge Relay, which reverses the translations made by an SIIT-DC Border Relay. The application and/or node is thus provided with seemingly native IPv4 connectivity that provides end-to-end address transparency.

The reader is expected to be familiar with the SIIT-DC architecture described in I-D.ietf-v6ops-siit-dc.

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 April 14, 2016.

Copyright Notice

Table of Contents

1. Introduction

SIIT-DC [I-D.ietf-v6ops-siit-dc] describes an architecture where IPv4-only users can access IPv6-only services through a stateless translator called an SIIT-DC Border Relay (BR). This approach has certain limitations, however. In particular, the following cases will work poorly or not at all:

- Application protocols that do not support NAT (i.e., the lack of end-to-end transparency of IP addresses).

- o Nodes that cannot connect to IPv6 networks at all, or that can only connect such networks if they also provide IPv4 connectivity (i.e., dual-stacked networks).

- o Application software which makes use of legacy IPv4-only APIs, or otherwise makes assumptions that IPv4 connectivity is available.

By extending the SIIT-DC architecture with a new component called an Edge Relay (ER), all of the above can be made to work correctly in an otherwise IPv6-only network environment using SIIT-DC.

The purpose of the ER is to reverse the IPv4-to-IPv6 packet translations previously done by the BR for traffic arriving from IPv4 clients and forward this as "native" IPv4 to the node or application. In the reverse direction, IPv4 packets transmitted by the node or application are intercepted by the ER, which translates them to IPv6 before they are forwarded to the BR, which in turn will reverse the translations and forward them to the IPv4 client. The node or application is thus provided with "virtual" IPv4 Internet connectivity that retains end-to-end transparency for the IPv4 addresses.

2. Terminology

This document makes use of the following terms:

SIIT-DC Border Relay (BR)
: A device or a logical function that performs stateless protocol translation between IPv4 and IPv6. It MUST do so in accordance with [RFC6145] and [I-D.ietf-v6ops-siit-eam].

SIIT-DC Edge Relay (ER)
: A device or logical function that provides "native" IPv4 connectivity to IPv4-only devices or application software. It is very similar in function to a BR, but is typically located close to the IPv4-only component(s) it is supporting rather than on the IDC’s outer network border. An ER may be either Node-Based (Section 3.1) or Network-Based (Section 3.2).

IPv4 Service Address
: An IPv4 address representing a node or service located in an IPv6 network. It is coupled with an IPv6 Service Address using an EAM. Packets sent to this address is translated to IPv6 by the BR, and possibly back to IPv4 by an ER, before reaching the node or service.

IPv6 Service Address

An IPv6 address assigned to an application, node, or service; either directly or indirectly (through an ER). It is coupled with an IPv4 Service Address using an EAM. IPv4-only clients communicates with the IPv6 Service Address through SIIT-DC.

Explicit Address Mapping (EAM)
A bi-directional coupling between an IPv4 Service Address and an IPv6 Service Address configured in a BR or ER. When translating between IPv4 and IPv6, the BR/ER changes the address fields in the translated packet's IP header according to any matching EAM. The EAM algorithm is specified in [I-D.ietf-v6ops-siit-eam].

Translation Prefix
An IPv6 prefix into which the entire IPv4 address space is mapped, according to the algorithm in [RFC6052]. The Translation Prefix is routed to the BR's IPv6 interface. When translating between IPv4 and IPv6, an BR/ER will insert/remove the Translation Prefix into/from the address fields in the translated packet's IP header, unless an EAM exists for the IP address that is being translated.

IPv4-converted IPv6 addresses
As defined in Section 1.3 of [RFC6052].

IDC
Short for "Internet Data Centre"; a data centre whose main purpose is to deliver services to the public Internet, the use case SIIT-DC is primarily targeted at. IDCs are typically operated by Internet Content Providers or Managed Services Providers.

SIIT
The Stateless IP/ICMP Translation algorithm, as specified in [RFC6145].

XLAT
Short for "Translation". Used in figures to indicate where a BR/ER uses SIIT [RFC6145] to translate IPv4 packets to IPv6 and vice versa.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Edge Relay Description

An Edge Relay (ER) is at its core an implementation of the Stateless IP/ICMP Translation algorithm [RFC6145] that supports Explicit Address Mappings [I-D.ietf-v6ops-siit-eam]. It provides virtual IPv4 connectivity for nodes or applications which require this to operate correctly in an SIIT-DC environment.

Packets from the IPv4 Internet destined for an IPv4 Service Address is first translated to IPv6 by a BR. The resulting IPv6 packets are subsequently forwarded to the ER that owns the IPv6 Service Address the translated packets are addressed to. The ER then translates them back to IPv4 before forwarding them to the IPv4 application or node. In the other direction, the exact same translations happen, only in reverse. This process provides end-to-end transparency of IPv4 addresses.

An ER may handle an arbitrary number of IPv4/IPv6 Service Addresses. All the EAMs configured in the BR that involve the IPv4/IPv6 Service Addresses handled by an ER MUST also be present in the ER’s configuration.

An ER may be implemented in two distinct ways; as a software-based service residing inside an otherwise IPv6-only node, or as a network-based service that provides an isolated IPv4 network segment to which nodes that require IPv4 can connect. In both cases native IPv6 connectivity may be provided simultaneously with the virtual IPv4 connectivity. Thus, dual-stack connectivity is facilitated in case the node or application support it.

The choice between a node- or network-based ER is made on a per-service or per-node basis. An arbitrary number of each type of ER may co-exist in an SIIT-DC architecture.

This section describes the different approaches and discusses which approach fits best for the various use cases.

3.1. Node-Based Edge Relay

A Node-based Edge Relay

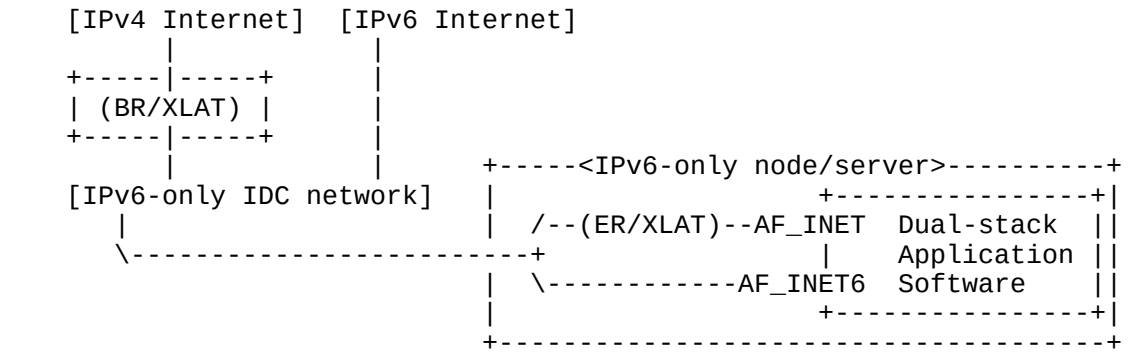

Figure 1

A node-based ER is typically implemented as a logical software function that runs inside the operating system of an IPv6 node. It provides applications running on the same node with IPv4 connectivity. Its IPv4 Service Address SHOULD be considered a regular local address that allows application running on the same node to use it with IPv4-only API calls, e.g., to create AF_INET sockets that listen for and accept incoming connections to its IPv4 Service Address. An ER may accomplish this by creating a virtual network adapter to which it assigns the IPv4 Service Address and points a default IPv4 route. This approach is similar to the "Bump-in-the-Stack" approach discussed in [RFC6535], however it does not include an Extension Name Resolver.

As shown in Figure 1, if the application supports dual-stack operation, IPv6 clients will be able to communicate with it directly using native IPv6. Neither the BR nor the ER will intercept this communication. Support for IPv6 in the application is however not a requirement; the application may opt not to establish any IPv6 sockets. Foregoing IPv6 in this manner will simply preclude connectivity to the service from IPv6-only clients; connectivity to the service from IPv4 clients (through the BR) will continue work in the same way.

The ER requires a dedicated IPv6 Service Address for each IPv4 Service Address it has configured. The IPv6 network MUST forward traffic to these IPv6 Service Addresses to the node, whose operating system MUST in turn forward them to the ER. This document does not attempt to fully explore the multitude of ways this could be accomplished, however considering that the IPv6 protocol is designed for having multiple addresses assigned to a single node, one particularly straight-forward way would be to assign the ER's IPv6 Service Addresses as secondary IPv6 addresses on the node itself so that it the upstream router learns of their location using the IPv6 Neighbor Discovery Protocol [RFC4861].

3.2. Network-Based Edge Relay

A Basic Network-based Edge Relay

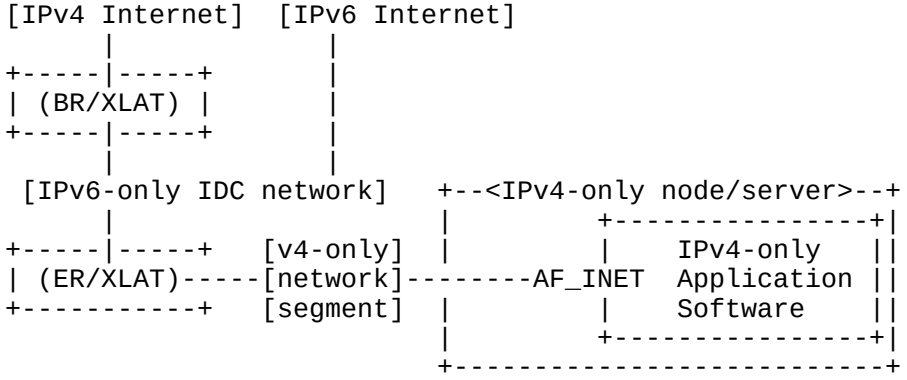

Figure 2

A network-based ER performs the exact same as a node-based ER does, only that instead of assigning the IPv4 Service Addresses to an internal-only virtual network adapter, traffic destined for them are forwarded onto a network segment to which nodes that require IPv4 connectivity connect to. The ER also functions as the default IPv4 router for the nodes on this network segment.

Each node on the IPv4 network segment MUST acquire and assign an IPv4 Service Address to a local network interface. While this document does not attempt to explore all the various methods by which this could be accomplished, some examples are provided in Appendix A.

The basic ER illustrated in Figure 2 establishes an IPv4-only network segment between itself and the IPv4-only nodes it serves. This is fine if the nodes it provides IPv4 access have no support for IPv6 whatsoever; however if they are dual-stack capable, it is would not

be ideal to take away their IPv6 connectivity in this manner. While it is RECOMMENDED to use a node-based ER in this case, appropriate implementations of a node-based ER might not be available for every node. If the application protocol in question does not work correctly in a NAT environment, standard SIIT-DC cannot be used either, which leaves a network-based ER is the only remaining solution. The following subsections contains examples on how the ER could be implemented in a way that provides IPv6 connectivity for dual-stack capable nodes.

3.2.1. Edge Router "On A Stick"

A Network-based Edge Relay "On A Stick"

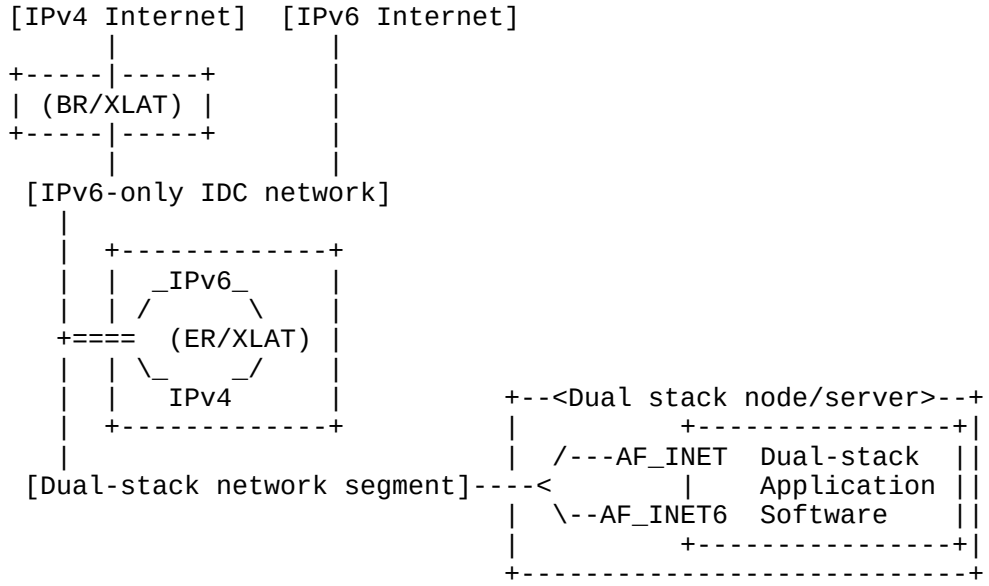

Figure 3

The ER "On A Stick" approach illustrated in Figure 3 ensures that the dual-stack capable node retains native IPv6 connectivity by connecting the ER’s IPv4 and IPv6 interfaces to the same network segment, alternatively by using a single dual-stacked interface. Native IPv6 traffic between the IDC network and the node bypasses the ER entirely, while IPv4 traffic from the node will be routed directly to the ER (because it acts as its default IPv4 router), where it is translated to IPv6 before being transmitted to the upstream default IPv6 router. The ER could attract inbound traffic to the IPv6 Service Addresses by responding to the upstream router’s IPv6 Neighbor Discovery [RFC4861] messages for them.

3.2.2. Edge Router that Bridges IPv6 Packets

A Network-based Edge Relay containing an IPv6 Bridge

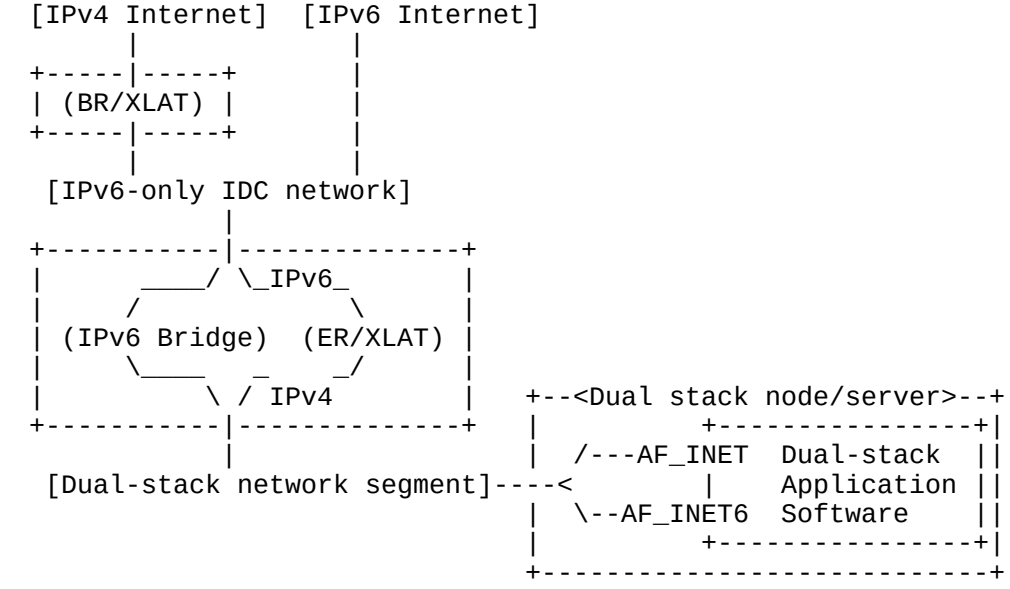

Figure 4

The ER illustrated in Figure 4 will transparently bridge IPv6 frames between its upstream and downstream interfaces. IPv6 packets addressed the ER's own IPv6 Service Addresses from the upstream IDC network are intercepted (e.g., by responding to IPv6 Neighbor Discovery [RFC4861] messages for them) and routed through the translation function before being forwarded out its downstream interface as IPv4 packets. The downstream network segment thus becomes dual-stacked.

4. Deployment Considerations

4.1. IPv6 Path MTU

The IPv6 Path MTU between the ER and the BR will typically be larger than the default value defined in Section 4 of [RFC6145] (1280 bytes), as it will typically contained within a single administrative domain. Therefore, it is RECOMMENDED that the IPv6 Path MTU configured in the ER is raised accordingly. It is RECOMMENDED that the ER and the BR use identical configured IPv6 Path MTU values.

4.2. IPv4 MTU

In order to avoid IPv6 fragmentation, an ER SHOULD ensure that the IPv4 MTU used by applications or nodes is equal to the configured IPv6 Path MTU - 20, so that an maximum-sized IPv4 packet can fit in an unfragmented IPv6 packet. This ensures that the application may do its part in avoiding IP-level fragmentation from occurring, e.g., by segmenting/fragmenting outbound packets at the application layer, and advertising the maximum size its peer may use for inbound packets (e.g., through the use of the TCP MSS option).

A node-based ER could accomplish this by configuring this MTU value on the virtual network adapter, while a network-based ER could do so by advertising the MTU to its downstream nodes using the DHCPv4 Interface MTU Option [RFC2132].

4.3. IPv4 Identification Header

If the generation of IPv6 Atomic Fragments is disabled, the value of the IPv4 Identification header will be lost during the translation. Conversely, enabling the generation of IPv6 Atomic Fragments will ensure that the IPv4 Identification Header will carried end-to-end. Note that for this to work bi-directionally, IPv6 Atomic Fragment generation MUST be enabled on both the BR and the ER.

Apart from certain diagnostic tools, there are few (if any) application protocols that make use of the IPv4 Identification header. Therefore, the loss of the IPv4 Identification value will therefore generally not cause any problems.

IPv6 Atomic Fragments and their impact on the IPv4 Identification header is further discussed in Section 4.9.2 of [I-D.ietf-v6ops-siit-dc].

5. Intra-IDC IPv4 Communication

Although SIIT-DC is primarily intended to facilitate communication between IPv4-only nodes on the Internet and services located in an IPv6-only IDC network, an IPv4-only node or application located behind an ER might need to communicate with other nodes or services in the IDC. The IPv4-only node or application will need to so through the ER, as it will typically be incapable to contact IPv6 destinations directly. The following subsections discusses various methods on how to facilitate such communication.

5.1. Hairpinning by the SIIT-DC Border Relay

If the BR supports hairpinning as described in Section 4.2 of [I-D.ietf-v6ops-siit-eam], the easiest solution is to make the target service available through SIIT-DC in the normal way, that is, by provisioning an EAM to the BR that assigns an IPv4 Service Address with the target service's IPv6 Service Address.

This allows the IPv4-only node or application to transmit packets destined for the target service's IPv4 Service Address, which the ER will then translate to a corresponding IPv4-converted IPv6 address by inserting the Translation Prefix [RFC6052]. When this IPv6 packet reaches the BR, it will be hairpinned and transmitted back to the target service's IPv6 Service Address (where it could possibly pass through another ER before reaching the target service). Return traffic from the target service will be hairpinned in the same fashion.

Hairpinned IPv4-IPv4 packet flow

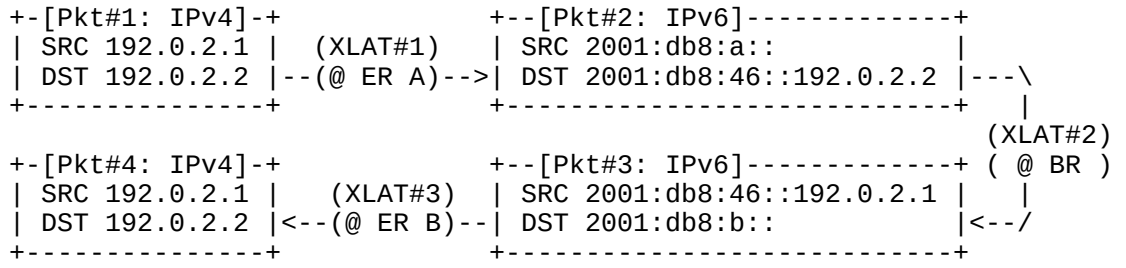

Figure 5

Figure 5 illustrates the flow of a hairpinned packet sent from the IPv4-only node/app behind ER A towards an IPv6-only node/app behind ER B. ER A is configured with the EAM {192.0.2.1,2001:db8:a::}, ER B with {192.0.2.2,2001:db8:b::}. The BR is configured with both EAMs, and supports hairpinning. Note that if the target service had not been located behind an ER, the third and final translation (XLAT#3) would not have happened, i.e., the target service/node would have received and responded to packet #3 directly.

If the IPv4-only nodes/services do not need connectivity with the public IPv4 Internet, private IPv4 addresses [RFC1918] could be used as their IPv4 Service Addresses in order to conserve the IDC operator's pool of public IPv4 addresses.

5.2. Additional EAMs Configured in Edge Relay

If the BR does not support hairpinning, or if the hairpinning solution is not desired for some other reason, intra-IDC IPv4 traffic

may be facilitated by configuring additional EAMs on the ER for each service the IPv4-only node or application needs to communicate with. This makes the IPv6 traffic between the ER and the target service's IPv6 Service Address follow the direct path through the IPv6 network. The traffic does not pass the BR, which means that this solution might yield better latency than the hairpinning approach.

The additional EAM configured in the ER consists of the target's IPv6 Service Address and an IPv4 Service Address. The IPv4-only node or application will contact the target's assigned IPv4 Service Address using its own IPv4 Service Address as the source. The ER will then proceed to translate this to an IPv6 packet with the local application/node's own IPv6 Service Address as source and the target service's IPv6 Service Address as the destination, and forward this to the IPv6 network. Replies from the target service will undergo these translations in reverse.

If the target service is also located behind another ER, that other ER MUST also be provisioned with an additional EAM that contains the origin IPv4-only application/node's IPv4 and IPv6 Service Addresses. Otherwise, the target service's ER will be unable to translate the source address of the incoming packets.

Non-hairpinned IPv4-IPv4 packet flow

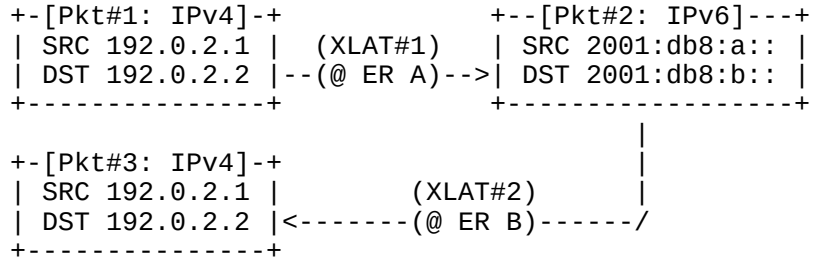

Figure 6

Figure 6 illustrates the flow of a packet carrying intra-IDC IPv4 traffic between two IPv4-only nodes/applications that are both located behind ERs. Both ER A and ER B are configured with two EAMs: {192.0.2.1,2001:db8:a::} and {192.0.2.2,2001:db8:b::}. The packet will follow the regular routing path through the IPv6 IDC network; the BR is not involved and the packet will not be hairpinned.

The above approach is not mutually exclusive with the hairpinning approach described in Section 5.1: If both EAMs above are also configured on the BR, both 192.0.2.1 and 192.0.2.2 would be reachable from other IPv4-only services/nodes using the hairpinning approach. They would also be reachable from the IPv4 Internet.

Note that if the target service in this example was not located behind an ER, but instead was a native IPv6 service listening on 2001:db8:b::, the second translation step in Figure 6 would not occur; the target service would receive and respond to packet #2 directly.

As with the hairpinning approach, if the IPv4-only nodes/services do not need connectivity to/from the public IPv4 Internet, private IPv4 addresses [RFC1918] could be used as their IPv4 Service Addresses. Alternatively, in the case where the target service is on native IPv6, the target's assigned IPv4 Service Address has only local significance behind the ER. It could therefore be assigned from the IPv4 Service Continuity Prefix [RFC7335].

6. Acknowledgements

The author would like to especially thank the authors of 464XLAT [RFC6877]: Masataka Mawatari, Masanobu Kawashima, and Cameron Byrne. The architecture described by this document is merely an adaptation of their work to a data centre environment, and could not have happened without them.

The author would like also to thank the following individuals for their contributions, suggestions, corrections, and criticisms: Fred Baker, Tobias Brox, Olafur Gudmundsson, Christer Holmberg, Ray Hunter, Shucheng LIU (Will), Andrew Yourtchenko.

7. IANA Considerations

This draft makes no request of the IANA.

8. Security Considerations

This section discusses security considerations specific to the use of an ER. See the Security Considerations section in [I-D.ietf-v6ops-siit-dc] for security considerations applicable to the SIIT-DC architecture in general.

If the ER receives an IPv4 packet from the application/node from a source address it does not have an EAM for, both the source and destination addresses will be rewritten according to [RFC6052]. After undergoing the reverse translation in the BR, the resulting

IPv4 packet routed to the IPv4 network will have a spoofed IPv4 source address. The ER SHOULD therefore ensure that ingress filtering [RFC2827] is used on the ER's IPv4 interface, so that such packets are immediately discarded.

If the ER receives an IPv6 packet with both the source and destination address equal to one of its local IPv6 Service Addresses, the resulting packet would appear to the IPv4-only application/node as locally generated, as both the source address and the destination address will be the same address. This could trick the application into believing the packet came from a trusted source (itself). To prevent this, the ER SHOULD discard any received IPv6 packets that have a source address that is either 1) equal to any of its local IPv6 Service Addresses, or 2) after translation from IPv6 to IPv4, equal to any of its local IPv4 Service Addresses.

9. References

9.1. Normative References

[I-D.ietf-v6ops-siit-dc]
Anderson, T., "SIIT-DC: Stateless IP/ICMP Translation for IPv6 Data Centre Environments", draft-ietf-v6ops-siit-dc-02 (work in progress), August 2015.

[I-D.ietf-v6ops-siit-eam]
Anderson, T. and A. Leiva, "Explicit Address Mappings for Stateless IP/ICMP Translation", draft-ietf-v6ops-siit-eam-01 (work in progress), June 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <http://www.rfc-editor.org/info/rfc826>.

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <http://www.rfc-editor.org/info/rfc1918>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <http://www.rfc-editor.org/info/rfc2131>.

[RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, DOI 10.17487/RFC2132, March 1997, <http://www.rfc-editor.org/info/rfc2132>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <http://www.rfc-editor.org/info/rfc2827>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, DOI 10.17487/RFC6052, October 2010, <http://www.rfc-editor.org/info/rfc6052>.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, DOI 10.17487/RFC6145, April 2011, <http://www.rfc-editor.org/info/rfc6145>.

[RFC6535] Huang, B., Deng, H., and T. Savolainen, "Dual-Stack Hosts Using "Bump-in-the-Host" (BIH)", RFC 6535, DOI 10.17487/RFC6535, February 2012, <http://www.rfc-editor.org/info/rfc6535>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <http://www.rfc-editor.org/info/rfc6877>.

[RFC7335] Byrne, C., "IPv4 Service Continuity Prefix", RFC 7335, DOI 10.17487/RFC7335, August 2014, <http://www.rfc-editor.org/info/rfc7335>.

Appendix A. Examples: Network-Based IPv4 Connectivity

A.1. Subnet with IPv4 Service Addresses

One relatively straight-forward way to provide IPv4 connectivity between the ER and the IPv4 node(s) it serves is to ensure the IPv4 Service Address(es) can be enclosed within a larger IPv4 prefix. The ER may then claim one address in this prefix for itself, and use it to provide an IPv4 default router address. The ER may then proceed to assign the IPv4 Service Address(es) to its downstream node(s) using DHCPv4 [RFC2131]. For example, if the IPv4 Service Addresses are 192.0.2.26 and 192.0.2.27, the ER would configure the address 192.0.2.25/29 on its IPv4-facing interface and would add the two IPv4 Service Addresses to its DHCPv4 pool.

One disadvantage of this method is that IPv4 communication between the IPv4 node(s) behind the ER and other services made available through SIIT-DC becomes impossible, if those other services are assigned IPv4 Service Addresses that also are covered by the same IPv4 prefix (e.g., 192.0.2.28). This happens because the IPv4 nodes will mistakenly believe they have an on-link route to the entire prefix, and attempt to resolve the addresses using ARP [RFC0826], instead of sending them to the ER for translation to IPv6. This problem could however be overcome by avoiding assigning IPv4 Service Addresses which overlaps with an IPv4 prefix handled by an ER (at the expense of wasting some potential IPv4 Service Addresses), or by ensuring that the overlapping IPv6 Service Addresses are only assigned to services which do not need to communicate with the IPv4 node(s) behind the ER. A third way to avoid this problem is discussed in Appendix A.2.

A.2. Subnet with Unrouted IPv4 Addresses

In order to avoid the problem discussed in Appendix A.1, a private unrouted IPv4 network that does not encompass the IPv4 Service Address(es) could be used to provide connectivity between the ER and the IPv4-only node(s) it serves. An IPv4-only node must then assign its IPv4 Service Address as secondary local address, while the ER routes each of the IPv4 Service Addresses to its assigned node using that node’s private on-link IPv4 address as the next-hop. This approach would ensure there are no overlaps with IPv4 Service addresses elsewhere in the infrastructure, but on the other hand it would preclude the use of DHCPv4 [RFC2131] for assigning the IPv4 Service Addresses.

This approach creates a need to ensure that the IPv4 application is selecting the IPv4 Service Address (as opposed to its private on-link IPv4 address) as its source address when initiating outbound connections. This could be accomplished by altering the Default Address Selection Policy Table [RFC6724] on the IPv4 node.

Authors' Addresses

Tore Anderson
Redpill Linpro
Vitaminveien 1A
0485 Oslo
Norway

Phone: +47 959 31 212
Email: tore@redpill-linpro.com
URI: http://www.redpill-linpro.com

Sander Steffann
S.J.M. Steffann Consultancy
Tienwoningenweg 46
Apeldoorn, Gelderland 7312 DN
The Netherlands

Email: sander@steffann.nl

Independent Submission
Internet-Draft
Expires: September 1, 2015

E. Lewis
ICANN
Date: March 1, 2015

Loopback Prefix for IPv6
draft-ipversion6-loopback-prefix-00

Abstract

The IPv6 address range of 0::/64 is reserved for loopback addresses. This expands from the single loophack address already defined for IPv6, ::1, to allow for a set of addresses to be used when packets are intended to stay within a host system. Multiple loopback addresses allow for simultaneous varied uses of the loopback addresses as has proven, albeit in limited ways, in IPv4. And exception is made to accomodate the ::0/128, already defined as The Unspecified Address.

Table of Contents

0. NOTE TO RFC EDITOR AND REVIEWERS

This section should be removed prior to publication.

1. Introduction

The "IP Version 6 Addressing Architecture" [RFC 4291] defines a single IPv6 loopback address as ::1/128. In "Special-Purpose IP Address Registries" [RFC6890], 127.0.0.0/8 is assigned for loophack addresses, with usually just 127.0.0.1/32 implemented by default.

Ordinarily, just one address (whether IPv4 or IPv6) is sufficient for loopback addressing on a node but there have been a few use cases showing that it is desireable to have more than 1 (but less than the over 16 million that are in an IPv4 /8).

One use case is testing or prototyping, desiring to mimic a small network of processes on one node. To demonstrate a particular protocol's server running on a well-known port, having multiple addresses where packets can "travel" within the host is useful.

Another use case has arisen from ICANN's Controlled Interruption approach [need reference] which directs errant traffic to a loopback address with two distinct goals in mind. One is to prevent the leakage of packets that are known to be erroneously sent and two is to leave "bread crumbs" in log files for operators to use to help track why the erroneous packets are being sent.

The use of ::0/64 is (proposed) to represent an address range (or block) encompassing The Unspecified Address and loopback addresses.

2. Use of ::0/64 Addresses

The Unspecified Address, or ::0/128, remains as defined in RFC 4291's section 2.5.2. That definition is included by reference here so as to prevent any unintentional changes to the original text.

For all other addresses within ::0/64, the rules for using are the same as the rules in RFC 4291's section 2.5.3, again included by reference so as not to introduce any unintentional changes.

3. IANA Considerations

Registration in the IANA IPv6 Special-Purpose Address Registry

The IANA is directed to add ::0/64 to the "IANA IPv6 Special-Purpose Address Registry" specified in [RFC6890] as follows:

Address Block: ::0/64

Name: Loopback and Unspecified Addresses

RFC: [THIS DOCUMENT]

Allocation Date: [APPROVAL DATE]

Termination Date: N/A

Source: True [1]

Destination: False

Forwardable: False

Global: False

Reserved-by-Protocol: True

[1] True for ::0/128, False for all other addresses in ::0/64

The IANA is directed to remove Table 17 and Table 18 a
defined in RFC 6890, section 2.2.3.

4. Security Considerations

Security is not (yet) a consideration

5. Acknowledgements

We all this all to David Conrad.

6. References

6.1. Normative References

[RFC 4291] "IP Version 6 Addressing Architecture", Hinden & Deering, Feb 2006
[RFC 6890] "Special-Purpose IP Address Registries:, Cotton, Vegoda, Bonica & Haberman, Apr 2013

Authors' Addresses

Edward Lewis
edward.lewis@icann.org
801 17th Street NW
Suite 400
Washington, DC, 20006
US

V6OPS
Internet-Draft
Intended status: Informational
Expires: September 26, 2015

B. Liu
S. Jiang
Y. Bo
Huawei Technologies

March 25, 2015

Multiple IPv6 Prefixes: Background and Considerations
draft-liu-v6ops-running-multiple-prefixes-03

Abstract

This document describes several typical multiple prefixes use cases, and discusses that running multiple IPv6 prefixes/addresses in one network/host should be common proactice that administrators need to adapt.

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 September 26, 2015.

Table of Contents

1. Introduction

In IPv6 networks, there are deployment scenarios in which multiple prefixes coexists simultaneously in one network. Several typical use cases are:

- Multiple Prefixes with Different Scopes (described in Section 2.1)

- IPv6 multihoming based on multiple PA prefixes (described in Section 2.2)

- Make-before-break renumbeirng (described in Section 2.3)

- An IPv6 network with multiple services, each of which has a distinct prefix (described in Section 2.4) .

To support the multiple prefixes running mode, there have been some technologies developed. This document discusses these technologies of different aspects, which could allow and smoothen the multiple prefiex operation.

Note that, although MIF (Multiple InterFaces) [RFC6418] architecture also involves mutliple IPv6 prefixes, it mainly targets different interfaces which attach to different networks respectively. This document discusses the multiple IPv6 prefixes running in the same network.

2. Multiple Prefixes Use cases

2.1. Multiple Prefixes with Different Scopes

IPv6 contains link-local addresses, global addresses and unique local addresses, which by definition are global but normally are site-scope by practice.

As specified in [RFC4291], all interfaces are required to have at least one Link-Local unicast address. This is the basic case of running multiple prefixes. However, this does not require operations from the network administrators since it is automatically processed.

Besides Link-Local addresses, the Unique Local Addresses (ULAs, [RFC4193]) might also be used for the interal communication within a site network. In many deployment, the ULA is used along with PA (Provider Aggregated) addresses, which connect to the public network. The benefit of such combination is to provide seperate local communication from the globally communication so that the local communication would not be impacted when ISP uplink fail or prefix(es) be renumbered. It is especially benificial for the home network and private OAM plane or internal-only nodes in an enterprise.

2.2. Multihoming based on Multiple PA Prefixes

When a network is multihomed, the multiple upstream network providers would assign prefixes respectively. If a network does not acquires a PI (Provider Independent) address spaceu, multihoming will result coexistent multiple PA prefixes. In such network, a single host have multiple PA IPv6 addresses that assoicated with different prefixes.

This scenario rarely exists in IPv4 networks, since IPv4 only allows single address per interface. But it is quite practical in IPv6. This new feature of IPv6 allows the SMEs (Small/Medium Enterprises) to multihome without the burden of running PI address space or running IPv6 NAT. Furthermore, multiple PA spaces do not have the potential global routing system scalable issue as the PI does [RFC4894].

However, multihoming with multiple PA prefixes has some operational issues which mainly include address selection, next-hop selection, and exit-router selection. For detailed discussion, please refer to [RFC7157]. [Editor’s note: more discussion to be filled.]

2.3. Multiple Prefix Co-existing during Network Renumbering

[RFC4192] describes a procedure that can be used to renumber a network from one prefix to another smoothly through a "make-before-break" transition. In the transition period, both the old and new prefixes are available; the usage of multiple prefixes provides the smooth transition and avoids the session outage issue in most of renumbering operations.

2.4. Service Prefixes

An IPv6 network may simultaneously provide multiple services, such as IPTV, Internet access, VPN, etc. Each of these services should have a distinct prefix. The network may apply different policy based on the distinguished prefixes. This deployment would simplify the management and processing on network devices, such as forwarding routers, access authentication devices, account devices, bourder filter, etc. The ISPs would provide one subscriber multiple addresses/prefixes to access different services. This deployment would paricularly benefit for traffic recognision and management.

3. Operational Availability and Considerations

This section discusses some technologies of different aspects, which could allow and smooth the multiple prefiex operation.

3.1. Multiple prefix provisioning

- Multiple Prefixes from Different Provisioning Domains

  In [I-D.ietf-mif-mpvd-arch], provisioning domain is defined as consistent set of network configuration information. Classically, the entire set available on a single interface is provided by a single source, such as network administrator, and can therefore be treated as a single provisioning domain.

  But in modern IPv6 networks, multihoming or service prefixes may result in provisioning information from more than one provisioning domains being presented on a single link. In these scenarios, current technologies lack support of distinguishing information from multiple provisioning domains, thus the host would not be able to associate configuration information with provisioning domains.

  However, there are several techniques under developing in MIF WG to solve the problems, we could expect them to be standardized in the near future.

- Co-existing DHCPv6/SLAAC

  Both SLAAC [RFC4862] and DHCPv6-PD [RFC3633] could assign IPv6 prefixes. DHCPv6-PD is normally run between routers and routers or routers and DHCPv6 [RFC3315] servers; while SLAAC is normally run between routers and downstream hosts. The two protocols could collabarate sufficiently to cover the whole network's prefix provisioning.

  If operate properly, SLAAC and DHCPv6 could also co-exist for IPv6 addresses provisioning based on different prefixes. They need to carefully deal with the interaction between the two protocols. It is mostly regarding to the M flag in Neighbor Discovery [RFC4861] messages.

3.2. Address Selection

In order to support multiple addresses well, IPv6 introduced address selection mechanism which utilize a address selection policy table to calculate a proper source address for a given destination address. Of cource, destination adresses selection is also defined. [RFC6724] described the rationale and algorithms in detail, and also defines a default address selection policy table for operating systems.

Note that, the [RFC6724] is a replacement of the old [RFC3484] specification to improve some behaviors (e.g. to prefer IPv4 over ULA for outside connectivity). Currently, so far there haven't been many operating systems supporting the new standard, but we could expect that the new standard would be available in all new released operating systems and becomes the mainstream in the near future.

3.3. Exit-router selection

In multiple PA multihoming networks, if the ISPs enable ingress filtering at the edge (BCP38, [RFC2827]), then there comes the exit router selection issues that outgoing packets are routed to the appropriate border router and ISP link. Normally, a packet sourced from an address assigned by ISP X should not be sent via ISP Y, otherwise it would be filtered by ISP Y.

In the past, the administrators have to either communicate with the ISP for not filtering the prefixes or manually configure routing policies within the network to make sure the traffics are forwarded to the right upstream link, based on source prefixes. Now, there are some source-based routing technologies under development and standardization. We could expect these solutions available soon.

4. Security Considerations

This document does not introduce any new mechanisms or protocols technologies and as such does not introduce any new security threads.

Nevertheless, relevant important security considerations are worth to be iterated here:

- [RFC7157] gives the security considerations for multi-prefix based multihoming.
- Address selection relevant security considerations are described in [RFC6724].
- ND cache exhausion caused by multiple addresses per host in a big L2 network is described in Section 3.2. It is possibility that malicious users intentionally configure massive addresses on host to make the gateway ND cache exhausted. So administrators always need to consider mitigation operations for potential ND cache DoS attack which is documented as [RFC6583].

5. IANA Considerations

This draft does not request any IANA action.

6. Acknowledgements

Valuable inputs of the texts/ideas were from Ole Troan.

Useful comments were received from Brian Carpenter, Victor Kuarsingh, Lorenzo Colliti, Mikael Abrahamsson, Fred Baker, Lee Howard and Roberta Maglione.

This document was produced using the xml2rfc tool [RFC2629]. (initiallly prepared using 2-Word-v2.0.template.dot.)

7. References

7.1. Normative References

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

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

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

7.2. Informative References

[I-D.ietf-mif-mpvd-arch] Anipko, D., "Multiple Provisioning Domain Architecture", draft-ietf-mif-mpvd-arch-11 (work in progress), March 2015.

[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.

[RFC3484] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, February 2003.

[RFC4192] Baker, F., Lear, E., and R. Droms, "Procedures for Renumbering an IPv6 Network without a Flag Day", RFC 4192, September 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4894] Hoffman, P., "Use of Hash Algorithms in Internet Key Exchange (IKE) and IPsec", RFC 4894, May 2007.

[RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", RFC 6418, November 2011.

[RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, March 2012.

[RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

[RFC6879] Jiang, S., Liu, B., and B. Carpenter, "IPv6 Enterprise Network Renumbering Scenarios, Considerations, and Methods", RFC 6879, February 2013.

[RFC7157] Troan, O., Miles, D., Matsushima, S., Okimoto, T., and D. Wing, "IPv6 Multihoming without Network Address Translation", RFC 7157, March 2014.

Authors' Addresses

Bing Liu
Huawei Technologies
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: leo.liubing@huawei.com

Sheng Jiang
Huawei Technologies
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com

Bo Yang
Huawei Technologies
Q21, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: boyang.bo@huawei.com

IPv6 Operations
Internet-Draft
Intended status: Informational
Expires: September 7, 2015

E. Vyncke
Cisco

March 6, 2015

HTTP State Management Mechanisms with Multiple Addresses User Agents
draft-vyncke-v6ops-happy-eyeballs-cookie-01

Abstract

HTTP servers usually save session states in their persistent storage indexed by session cookies generated by the HTTP servers. It is up to the HTTP user-agent to send this session cookie on each HTTP request. Some HTTP servers check whether the cookie is associated with the HTTP user-agent by the means of the user-agent IP address. Everything linking a state to an IP address (such as OAuth access code) to an IP address has the same issue.

If the Happy Eyeball mechanism is used to select between IPv6 and IPv4, it may happen that while using the same HTTP server, some HTTP requests are done over IPv6 and the others over IPv4, which leads to two different sets of session states in the HTTP server. This has the consequence of inconsistencies at the HTTP server.

The only purpose of this document is to document this issue in more details than in section 8.2 of RFC 6883 including security considerations and mitigations.

A similar problem arises with the use of non RFC 6888 compliant Carrier-Grade NAT (CGN) devices used to access an IPv4-only HTTP server or HTTP user-agent using multi-homing.

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 September 7, 2015.

Table of Contents

1. HTTP Session Management with HTTP Cookie

HTTP requests are basically stateless, therefore if a HTTP server requires to have some states associated to a HTTP user-agent (such as user name, login state, history, shopping basket, ...), there is a need to conserve those states. This is usually done by using a HTTP cookie (see also RFC6265 [RFC6265]) identifying the session; also called "session state cookie".

This session state cookie is generated by the HTTP server at the very first HTTP request from a HTTP user-agent. The cookie is usually opaque (often a random number) and has no semantic except as being an index within the persistent storage of the HTTP server. This index is used to access the complete state of the user-agent. This mechanism is secure if the cookie is transferred with confidentiality

between the server and the user-agent. If the cookie transfer and storage are not secured, then any hostile user-agent can reuse this cookie to access the full original session states (including shopping basket, payment details, ...); this attack is called 'session cookie stealing'. This attack can happen if the HTTP traffic is intercepted by a man-in-the-middle attack but a good use of Transport Level Security RFC5246 [RFC5246] can prevent it. The attack can also happen with some hostile scripting or other pieces of malware running on the user agent, that could copy and send the session cookie to the hostile user-agent; hence, it is not enough to use TLS to secure the session cookies.

Some HTTP applications link the user-agent IP address (whether IPv6 or IPv4) to the session state, probably for additional security checks in order to prevent session cookie stealing. This link leads to some issues in a dual-stack world which are described in this document.

The author knows about at least two large web sites having this problem. It was so severe that those sites which were dual-stack had to move back to being IPv4-only... until the application and its security is updated.

1.1. Other Use of Session Cookies

Beside the use of session cookies by the HTTP server to keep states on the server, the very same cookie is also sometimes used by Server Load Balancing (SLB) mechanism to ensure that all HTTP requests from the same user-agent (even if behind a NAT) are always sent to the same physical HTTP server. This is required if the server persistent storage is local to the server and is not shared by all the physical servers behind the SLB.

1.2. new section

Actually the problem is more generic than the session cookie, everything linking a state to an IP address has the same issue. This includes OAuth [RFC6749] access tokens, bearer tokens, ... but also other mechanisms such as rate limiting per IP address or access control per IP address (for instance a captive portal for a guest net).

2. Issues

Similar issues can be caused by Happy Eyeball RFC6555 [RFC6555], Carrier-Grade NAT (CGN) and having multiple interface or being multi-homed.

2.1. Happy Eyeballs Issue

When a HTTP user-agent uses the Happy Eyeball mechanism to access a HTTP server, then, part of the HTTP requests can happen over IPv6 and another part over IPv4 if the latency between IPv4 and IPv6 varies quickly over time. If there is a link between the session cookie and the user-agent IP address, then upon the first change of IP protocol version, the states associated to the cookie will be invalidated and will be deleted. Here is an example:

1. User-agent with IPv4 address, ADDR4, connect to the server by using IPv4 because IPv6 is slower; the first request does not have any HTTP cookie;

2. Server generates a new cookie C4 and stores in its persistent storage that C4 is associated with address ADDR4;

3. User-agent continues his/her session using IPv4, on each new request the HTTP server receives the cookie C4 and checks that the user-agent address is indeed ADDR4;

4. Latency of IPv6 changes and becomes now faster than IPv4;

5. User-agent now uses its IPv6 address, ADDR6, to connect to the same server and continues to use the same cookie C4 as the server name is unchanged;

6. The server receives the HTTP request with the C4 cookie and checks whether C4 is associated with ADDR6 which is not the case... All session states are deleted and a new cookie, C6, is generated and associated to the IPV6 address ADDR6;

7. The end-user becomes frustrated because he/she has to restart his/her complete session from the beginning.

This cookie invalidation may have some security benefit but it actually prevents a host using Happy Eyeballs to have a persistent session with a dual-stack HTTP server; with painful consequences for the user-experience: disconnection, loss of shopping basket, ...

2.2. Carrier-Grade NAT Issue

RFC6888 [RFC6888] describes the CGN requirements but not all CGN implement them. Some CGN in the real world have a pool of IPv4 addresses and do not always use the same public IPv4 address for all requests from a CGN client. This obviously leads to the same problem as in section Section 2.1. This will happen for IPv4-only HTTP servers.

Whether the CGN is used by IPv4 clients or by IPv6 clients (via NAT64 RFC6146 [RFC6146])does not make any difference to the problem. The use of the address family translation by MAP-T MAP-T [I-D.ietf-softwire-map-t] does not suffer from this issue for IPv4-only HTTP servers since one subscriber is restricted to several layer-4 ports from a single IPv4 address.

2.3. Multiple Interfaces Issue

When the HTTP user-agent has multiple interfaces, for example 3GPP and Wi-Fi, the preferred IP address depends on the WiFi or 3GPP availability. In this case, a similar issue to Section 2.1 also happens as the session cookie can be linked first to the Wi-Fi IP address then when the user-agent looses its Wi-Fi connectivity the session cookie will be overwritten by a new session cookie linked to the 3GPP address.

Whether the user-agent uses IPv4-only, IPv6-only or dual-stack has no impact on the issue.

3. Mitigations

The obvious mitigation for this issue is NOT to link any HTTP state management (including cookies) to any IP address of the HTTP user-agent at the risk of increasing the risk of "session cookie stealing".

The author also believes that:

Multipath TCP RFC6824 [RFC6824] hides completely the set of addresses of the client to the application. Only the first subflow’s IP addresses are exposed to the application, even if a later subflow uses a different address family; so, any session cookie will be permanently linked to the first IP address used by the HTTP user-agent;

HTTP/2 [I-D.ietf-httpbis-http2] multiplexes multiple HTTP sessions over a single TCP connection, therefore, Happy Eyeball (or bad CGN) sees only one TCP connection and a change of IP address will never occur during the lifetime of this TCP connection.

4. IANA Considerations

This document contains no IANA considerations.

5. Security Considerations

The association of the session cookie with the user-agent IP address has some security value as it can help prevent "session cookie stealing" in some limited situations; this benefit should be balanced with the lack of persistent session and the remaining vulnerability if the HTTP session can be intercepted by a man-in-the-middle attack. Moreover with more and more CGN being deployed, linked a session cookie to an IP address shared by hundreds of subscribers is less effective as the cookie could be reused by any subscribers using the same shared public IP address.

6. Acknowledgements

The author would like to thank Brian Carpenter, Ray Hunter, Jeroen Massar, Dan Metzler, Erik Nygren, Mark ZZZ Smith, Joe Touch, Dan Wing and Andrew Yourtchenko for some discussions on this topic. Of course, RFC6883 [RFC6883] has already mentionned this issue without many details.

7. Informative References

[I-D.ietf-httpbis-http2]
Belshe, M., Peon, R., and M. Thomson, "Hypertext Transfer Protocol version 2", draft-ietf-httpbis-http2-17 (work in progress), February 2015.

[I-D.ietf-softwire-map-t]
Li, X., Bao, C., Dec, W., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", draft-ietf-softwire-map-t-08 (work in progress), December 2014.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 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.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011.

[RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, April 2012.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, January 2013.

[RFC6883] Carpenter, B. and S. Jiang, "IPv6 Guidance for Internet Content Providers and Application Service Providers", RFC 6883, March 2013.

[RFC6888] Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.

Author's Address

Eric Vyncke
Cisco
De Kleetlaan 6a
Diegem 1831
Belgium

Phone: +32 2 778 4677
Email: evyncke@cisco.com