IPv6 Operations
Internet-Draft
Updates: 6145 (if approved)
Intended status: Standards Track
Expires: April 22, 2016

T. Anderson
Redpill Linpro
A. Leiva Popper
NIC Mexico

October 20, 2015

Explicit Address Mappings for Stateless IP/ICMP Translation
draft-ietf-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 April 22, 2016.

Copyright Notice

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.

Translation using 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.

Inner (header or address)
: Refers to an IP header located inside the payload of an ICMP error packet, or to an IP address within that header. Compare "Outer".

Outer (header or address)
: Refers to the first IP header in a packet, or to an IP address within that header. In other words, an IP header or address that is NOT "Inner". If a reference is made to an IP header or address without the "Inner" or "Outer" qualifier, it should be considered as "Outer".

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

XLAT
: Short for "translation".

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 that is assigned to the SIIT function. IPv4 addresses are scarce, and as a result an operator might not have enough IPv4-translatable IPv6 addresses to number the 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 consequences:

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 not only have to 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. Finally, the Default Address Selection Policy Table [RFC6724] on the IPv6 nodes might need to be altered in order to ensure that outbound sessions towards the IPv4 Internet are sourced from an IPv4-translatable IPv6 address.

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, and formally updates Section 4.1 and Section 5.1 of [RFC6145]. Specifically, when the EAM algorithm is applied, it supplants [RFC6145]'s requirement that a translator operating in the stateless mode must translate the Source Address and Destination Address IP header fields according to Section 2.3 of [RFC6052].

3.1. Explicit Address Mapping Table

An SIIT implementation includes an EAMT, a conceptual table in which each row represents an EAM. Each EAM describes a mapping between IPv4 and IPv6 prefixes/addresses. An operator populates the EAMT to provide the mappings between the two address families.

The EAMT consists of the following columns:

o IPv4 Prefix

o 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.

Unless otherwise specified in Section 4, 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 IPv4 Prefix column of the EAMT is searched for the EAM entry that shares the longest common prefix with 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 IPv6 Prefix column of the EAMT is searched for the EAM entry that shares the longest common prefix with 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. Hairpinning of IPv6 Traffic

4.1. Problem Statement

Two IPv6 nodes that are both covered by EAMs might in certain circumstances attempt to communicate through a stateless translator, rather than using native IPv6 directly. This happens if one of the nodes initiate traffic towards the IPv4-converted IPv6 address whose embedded IPv4 address matches an EAM that covers the other node. Special consideration is required in order to make this communication pattern work in a bi-directional fashion. This is illustrated by the example below.

Assume that a stateless translator is configured with an [RFC6052] translation prefix of 64:ff9b::/96 and the EAMT shown in Figure 1. The IPv6 node 2001:db8:aaaa:: transmits an IPv6 packet towards 64:ff9b::192.0.2.2, which reaches the translator and is being translated into an IPv4 packet with source address 192.0.2.1 and destination address 192.0.2.2. This destination address is found in the EAMT, so the packet loops back into the translation function, and is translated back to an IPv6 packet with source address 2001:db8:aaaa:: and destination address 2001:db8:bbbb::b.

While this packet will reach its destination just fine, a problem will occur when 2001:db8:bbbb::b responds to it. The response packet will have a source address of 2001:db8:bbbb::b and a destination address of 2001:db8:aaaa::, and will be routed directly to its destination without being subjected to any form of translation. Because the source address of this response packet (2001:db8:bbbb::b) is not equal to the destination address of the initial outgoing packet (64:ff9b::192.0.2.2), the packet will most likely be discarded by 2001:db8:aaaa:: and bi-directional communication will most likely fail.

The above scenario could be made to work by ensuring that the stateless translator is hairpinning the traffic in both directions. Section 4.2 describes how this is accomplished. The resulting address translations are demonstrated step-by-step in Appendix B.1.

4.2. Recommendation

An SIIT implementation SHOULD include a feature that ensures that hairpinned IPv6 traffic is supported. The feature SHOULD be enabled by default. The following two subsections describe two alternate ways to implement this feature. An implementation MAY support both approaches.

4.2.1. Simple Hairpinning Support

When the simple hairpinning feature is enabled, the translator employs the following rules when translating from IPv4 to IPv6:

1. If the packet is not an ICMPv4 error: The EAM algorithm MUST NOT be used in order to translate the source address in the IPv4 header.

2. If the packet is an ICMPv4 error: The EAM algorithm MUST NOT be used when translating the destination address in the inner IPv4 header.

3. If the packet is an ICMPv4 error whose outer IPv4 source address is equal to its inner IPv4 destination address: The EAM algorithm MUST NOT be used in order to translate the source address in the outer IPv4 header.

Rule #2 and #3 are cumulative.

The addresses in question MUST instead be translated according to [RFC6145], as if they did not match any EAM.

4.2.2. Intrinsic Hairpinning Support

When the intrinsic hairpinning feature is enabled, the translator employs the following rules when receiving an IPv6 packet:

If all the conditions in either of the two sets below is true, the packet is to be hairpinned. The implementation MUST immediately (i.e., prior to forwarding it to the IPv4 network) translate the packet back to IPv6. During the second translation pass, the behaviour specified in Section 4.2.1 MUST be applied, and the Hop Limit field SHOULD NOT be decremented.

Condition set A:

A1. The packet is not an ICMPv4 error

A2. The destination address was translated using the [RFC6052] algorithm

A3. The destination address is found in the EAMT

Condition set B:

B1. The packet is an ICMPv4 error

B2. The inner source address was translated using the [RFC6052] algorithm

B3. The inner source address is found in the EAMT

5. Overlapping Explicit Address Mappings

The algorithm specified in Section 3 relies on making a lookup in the EAMT in order to find the EAM entry that shares the longest common prefix with the address being translated. Operators should note that configuring EAMs with overlapping or identical IPv4 or IPv6 Prefixes in the EAMT may create configurations where the IPv4-to-IPv6 and IPv6 -to-IPv4 address translations will not be symmetric. This may in some cases make bi-directional communication impossible.

The example EAMT in Figure 2 could be thought of as implementing IVI (Appendix A.2) (EAM #1), but additionally with a single exception in the style of SIIT-DC (Appendix A.3) (EAM #2). The IPv4 Prefixes of the two EAMs overlap, while the IPv6 Prefixes do not. This results in a situation where the IPv6 address 2001:db8:ffc6:3364:4000:: will be translated (according to EAM #1) to the IPv4 address 198.51.100.64. However, when this IPv4 address is translated back to IPv6, it will be translated (according to EAM #2) to the IPv6 address 2001:db8::abcd. Because the IPv4-to-IPv6 translation in this example does not mirror the corresponding IPv6-to-IPv4 translation, bi-directional communication involving the IPv6 address 2001:db8:ffc6:3364:4000:: might fail. In order to help avoid such situations, implementations MAY warn the operator when a new EAM that overlaps with a previously existing one is inserted into the EAMT.

EAMT containing overlapping IPv4 Prefixes

#	IPv4 Prefix	IPv6 Prefix
1	0.0.0.0/0	2001:db8:ff00::/40
2	198.51.100.64/32	2001:db8::abcd/128

Figure 2

In Figure 3, the IPv6 Prefixes of the two EAMs are identical. The behaviour of the stateless translator when translating an IPv6 packet that contains the address 2001:db8::1 to IPv4 is in this case unspecified. In order to prevent this situation from occurring, implementations MAY refuse to insert a new EAM, whose IPv4 or IPv6 Prefix value is identical to that of an already existing EAM, into the EAMT.

EAMT containing identical IPv6 prefixes

#	IPv4 Prefix	IPv6 Prefix
1	198.51.100.8/32	2001:db8::1/128
2	198.51.100.9/32	2001:db8::1/128

Figure 3

6. 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].

7. Security Considerations

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

8. IANA Considerations

This draft makes no request of the IANA.

9. 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 Fred Baker, Mohamed Boucadair, Cameron Byrne, Brian E Carpenter, Brian Haberman, Ray Hunter, Alvaro Retana, Michael Richardson, Dan Romascanu, Hemant Singh, and Andrew Yourtchenko.

10. References

10.1. Normative References

[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>.

10.2. Informative 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-03 (work in progress), October 2015.

[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>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <http://www.rfc-editor.org/info/rfc4862>.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, DOI 10.17487/RFC6144, April 2011, <http://www.rfc-editor.org/info/rfc6144>.

[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, DOI 10.17487/RFC6219, May 2011, <http://www.rfc-editor.org/info/rfc6219>.

[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>.

[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>.

[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. 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 4

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].

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 5

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.ietf-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	203.0.113.1/32	2001:db8:aaaa::1/128
2	203.0.113.2/32	2001:db8:bbbb::1/128

Figure 6

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].

Appendix B. Example IP Address Translations

Figure 7 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 7

B.1. Hairpinning Examples

The following examples show how hairpinned IPv6 packets between the IPv6 nodes 2001:db8:aaaa:: and 2001:db8:bbbb::b are translated according to Section 4. As in Appendix B, the EAMT in Figure 1 is used and the [RFC6052] translation prefix is 64:ff9b::/96. In addition, the [RFC6791] pool is assumed to contain only the single address 198.51.100.1.

Hairpinning of a normal IPv6 packet

XLAT Stage	Source Address	Destination Address
Initial	2001:db8:aaaa::	64:ff9b::192.0.2.2
Intermediate	192.0.2.1	192.0.2.2
Final	64:ff9b::192.0.2.1	2001:db8:bbbb::b

Figure 8

Figure 8 illustrates how a normal (i.e., not an ICMP error) IPv6 packet sent from 2001:db8:aaaa:: towards 64:ff9b::192.0.2.2 is is hairpinned. In this example, rule #1 in Section 4.2.1 was applied in order to disable the EAM algorithm when translating the intermediate IPv4 source address to IPv6.

Hairpinning of a router-originated ICMPv6 error

XLAT Stage	Loc.	Source Address	Destination Addr.
Initial	Outer	2001:db8::1234	64:ff9b::192.0.2.1
	Inner	64:ff9b::192.0.2.1	2001:db8:bbbb::b
Intermediate	Outer	198.51.100.1	192.0.2.1
	Inner	192.0.2.1	192.0.2.2
Final	Outer	64:ff9b::198.51.100.1	2001:db8:aaaa::
	Inner	2001:db8:aaaa::	64:ff9b::192.0.2.2

Figure 9

Figure 9 illustrates the hairpinning of an ICMPv6 error sent by an arbitrary IPv6 router (2001:db8::1234) in response to the packet Figure 8. In this example, rule #2 in Section 4.2.1 was applied in order to disable the EAM algorithm when translating the intermediate inner IPv4 destination address to IPv6.

Hairpinning of a host-originated ICMPv6 error

XLAT Stage	Loc.	Source Address	Destination Addr.
Initial	Outer	2001:db8:bbbb::b	64:ff9b::192.0.2.1
	Inner	64:ff9b::192.0.2.1	2001:db8:bbbb::b
Intermediate	Outer	192.0.2.2	192.0.2.1
	Inner	192.0.2.1	192.0.2.2
Final	Outer	64:ff9b::192.0.2.2	2001:db8:aaaa::
	Inner	2001:db8:aaaa::	64:ff9b::192.0.2.2

Figure 10

Figure 10 illustrates the hairpinning of an ICMPv6 error sent by the original destination host itself in response to the packet Figure 8. In this example, rules #2 and #3 in Section 4.2.1 were both applied in order to disable the EAM algorithm when translating the intermediate inner IPv4 destination address and the intermediate outer IPv4 destination address to IPv6.

Hairpinning of normal response packet

XLAT Stage	Source Address	Destination Address
Initial	2001:db8:bbbb::b	64:ff9b::192.0.2.1
Intermediate	192.0.2.2	192.0.2.1
Final	64:ff9b::192.0.2.2	2001:db8:aaaa::

Figure 11

Figure 11 illustrates how 2001:db8:bbbb::b's response to the packet in Figure 8 is hairpinned in the exact same fashion as the initial packet. Again, rule #1 in Section 4.2.1 was applied in order to disable the EAM algorithm when translating the intermediate IPv4

source address to IPv6. The example is included in order to illustrate how the addresses in the packet initially sent by 2001:db8:aaaa:: matches those in the translated response packet sent by 2001:db8:bbbb::b, thus facilitating bi-directional communication.

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

Alberto Leiva Popper
NIC Mexico
Av. Eugenio Garza Sada 427 L4-6
Monterrey, Nuevo Leon 64840
Mexico

Email: ydahhrk@gmail.com
URI: http://www.nicmexico.mx/