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

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:

   o 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

```
  [IPv4 Internet]  [IPv6 Internet]
        |             |
  +-----|-----+       |
  | (BR/XLAT) |       |
  +-----|-----+       |
        |             |      +-----<IPv6-only node/server>----------+
  [IPv6-only IDC network]    |                      +---------------+|
      |                      |  /--(ER/XLAT)--AF_INET  Dual-stack  ||
      \------------------------+                     |    Application ||
                             |  \------------AF_INET6  Software    ||
                             |                      +---------------+|
                             +-------------------------------------+
```
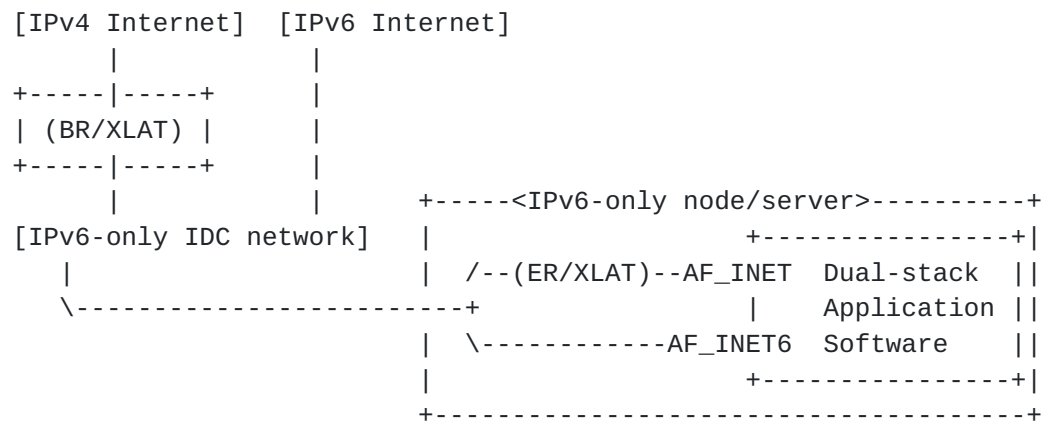
                              Figure 1

   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

          [IPv4 Internet]  [IPv6 Internet]
                |             |
          +-----|-----+       |
          | (BR/XLAT) |       |
          +-----|-----+       |
                |             |
           [IPv6-only IDC network]   +--<IPv4-only node/server>--+
                |               |          +----------------+|
          +-----|-----+   [v4-only] |          |    IPv4-only   ||
          | (ER/XLAT)-----[network]--------AF_INET  Application ||
          +-----------+   [segment] |          |    Software    ||
                                    |          +----------------+|
                                    +--------------------------+
```
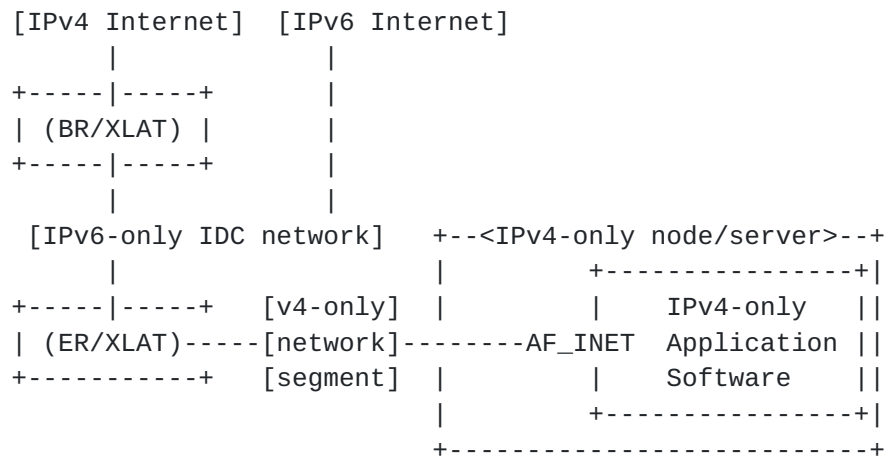
                              Figure 2

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"

     [IPv4 Internet]  [IPv6 Internet]
           |             |
   +-----|-----+        |
   | (BR/XLAT) |        |
   +-----|-----+        |
           |             |
    [IPv6-only IDC network]
       |
       |   +-------------+
       |   |  _IPv6_      |
       |   | /      \     |
      +====  (ER/XLAT) |
       |   | \_    _/    |
       |   |   IPv4      |            +--<Dual stack node/server>--+
       |   +-------------+            |         +----------------+|
       |                             |  /---AF_INET  Dual-stack  ||
      [Dual-stack network segment]----<         |     Application ||
                                     |  \--AF_INET6  Software     ||
                                     |         +----------------+|
                                     +---------------------------+
```
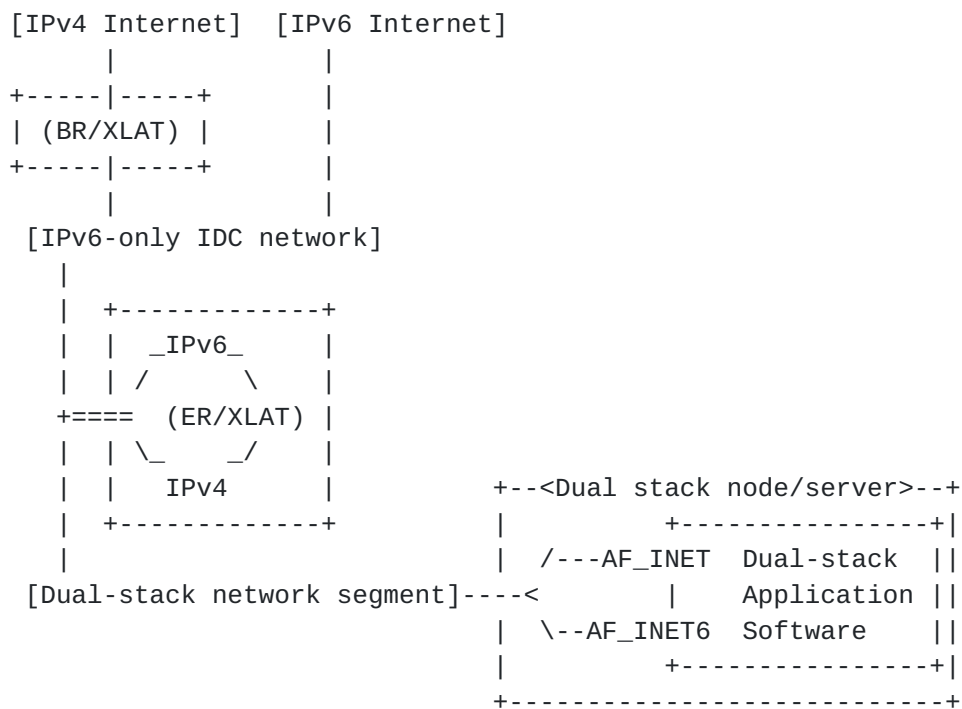
                              Figure 3

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

       [IPv4 Internet] [IPv6 Internet]
             | |
      +-----|-----+ |
      | (BR/XLAT) | |
      +-----|-----+ |
            | |
       [IPv6-only IDC network]
                 |
      +----------|-------------+
      | ____/ _IPv6_ |
      | /   \ |
      | (IPv6 Bridge) (ER/XLAT) |
      | ____   _   _/ |
      | \ / IPv4 | +--<Dual stack node/server>--+
      +----------|-------------+ | +----------------+|
                 | | /---AF_INET Dual-stack ||
         [Dual-stack network segment]----< | Application ||
                                     | \--AF_INET6 Software ||
                                     | +----------------+|
                                     +---------------------------+

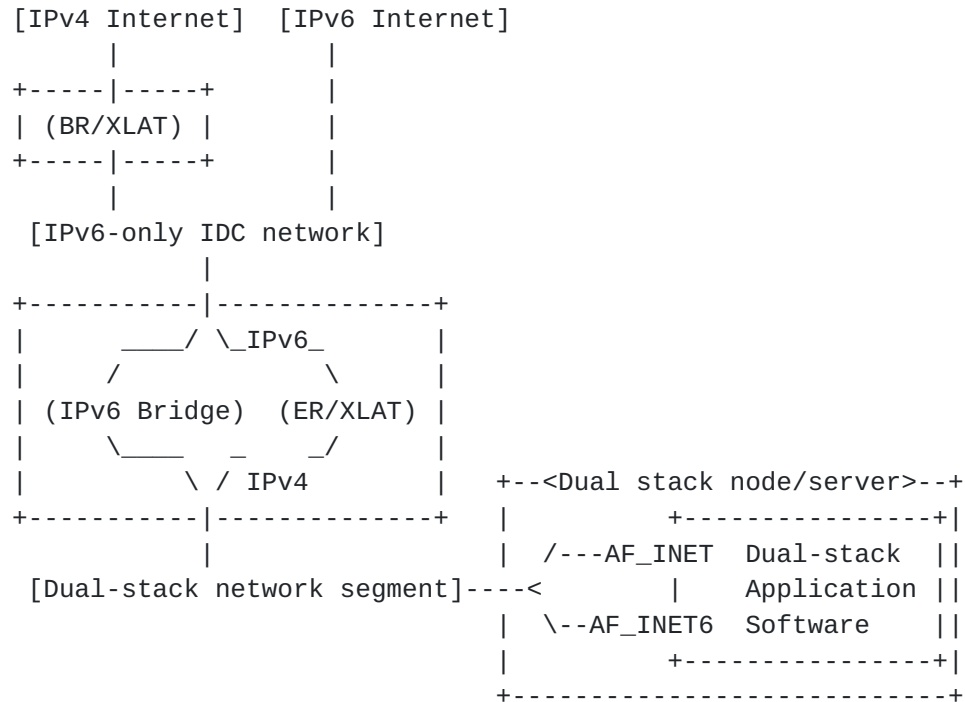   Figure 4

   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

```
+-[Pkt#1: IPv4]-+              +--[Pkt#2: IPv6]-------------+
| SRC 192.0.2.1 |  (XLAT#1)    | SRC 2001:db8:a::           |
| DST 192.0.2.2 |--(@ ER A)-->| DST 2001:db8:46::192.0.2.2 |---\
+---------------+              +----------------------------+   |
                                                             (XLAT#2)
+-[Pkt#4: IPv4]-+              +--[Pkt#3: IPv6]-------------+ ( @ BR )
| SRC 192.0.2.1 |   (XLAT#3)   | SRC 2001:db8:46::192.0.2.1 |   |
| DST 192.0.2.2 |<--(@ ER B)--| DST 2001:db8:b::           |<--/
+---------------+              +----------------------------+
```

                              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

        +-[Pkt#1: IPv4]-+   +--[Pkt#2: IPv6]---+
        | SRC 192.0.2.1 | (XLAT#1) | SRC 2001:db8:a:: |
        | DST 192.0.2.2 |--(@ ER A)-->| DST 2001:db8:b:: |
        +---------------+   +------------------+
                                                |
        +-[Pkt#3: IPv4]-+ |
        | SRC 192.0.2.1 | (XLAT#2) |
        | DST 192.0.2.2 |<-------(@ ER B)------/
        +---------------+

                                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