Network Working Group	S. Venaas
Internet-Draft	cisco Systems
Intended status: Informational	X. Li
Expires: December 23, 2011	C.X. Bao
	CERNET Center/Tsinghua University
	June 21, 2011

Framework for IPv4/IPv6 Multicast Translation
draft-venaas-behave-v4v6mc-framework-03.txt

Abstract

This draft describes how IPv4/IPv6 multicast translation may be used in various scenarios and attempts to be a framework for possible solutions. This can be seen as a companion document to the document "Framework for IPv4/IPv6 Translation" by Baker et al. When considering scenarios and solutions for unicast translation, one should also see how they may be extended to provide multicast translation.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.
Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet- Drafts is at http://datatracker.ietf.org/drafts/current/.
Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."
This Internet-Draft will expire on December 23, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.
This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

There will be a long period of time where IPv4 and IPv6 systems and networks need to coexist. There are various solutions for how this can be done for unicast, some of which are based on translation. The document [RFC6144] discusses the needs and provides a framework for unicast translation for various scenarios. Here we discuss the need for multicast translation for those scenarios.
For unicast the problem is basically how two hosts can communicate when they are not able to use the same IP protocol. For multicast we can restrict ourselves to looking at how a single source can efficiently send to multiple receivers. When using a single IP protocol one builds a multicast distribution tree from the source to the receivers, and independent of the number of receivers, one sends each data packet only once on each link. We wish to maintain the same characteristics when there are different IP protocols used. That is, when the nodes of the tree (source, receivers and routers) cannot all use the same IP protocol. In general there may be multiple sources sending to a multicast group, but that can be thought of as separate trees, one per source. We will focus on the case where the source and the receivers cannot all use the same IP protocol. If the issue is the network in between, encapsulation may be a better alternative. Note that if the source supports both IPv4 and IPv6, then one alternative could be for the source to send two streams. This need not be the same host. There could be two different hosts, and in different locations/networks, sending the same content.

2. Translation scenarios

We will consider six different translation scenarios. For each of the scenarios we will look at how host in one network can receive multicast from a source in another network. For unicast one might consider the following six scenarios as described in [RFC6144]:
Scenario 1: An IPv6 network to the IPv4 Internet
Scenario 2: The IPv4 Internet to an IPv6 network
Scenario 3: The IPv6 Internet to an IPv4 network
Scenario 4: An IPv4 network to the IPv6 Internet
Scneario 5: An IPv6 network to an IPv4 network
Scenario 6: An IPv4 network to an IPv6 network
We have intentionally left out how one might connect the entire IPv4 Internet with the entire IPv6 Internet. In these scenarios one would

look at how a host in one network initiates a uni- or bi-directional flow to another network. The initiator needs to somehow know which address to send the initial packet to, and the initial packet gets translated before reaching its destination.

For unicast translation it is quite natural to talk about networks and the Internet. For multicast this is not so clear, since there is limited use of multicast on the Internet. Certain parts of the Internet, e.g. academic and research networks and the links connecting them do carry multicast though. Also, the challenges and ideas described here regarding the Internet, also applies in other cases where there are multiple connected networks exchanging multicast.

For multicast one generally need a receiver to signal the group (and sometimes also the source) it wants to receive from. The signalling generally goes hop-by-hop towards the source to build multicast forwarding state that later is used to forward multicast in the reverse direction. This means that for the receiving host to receive multicast, it must first somehow know which group (and possibly source) it should signal that it wants to receive. These signals would then probably go hop-by-hop to a translator, and then the translated signalling would go hop-by-hop from the translator to the source. Note that this description is correct for SSM (source-specific multicast), but is in reality more complex for ASM (any-source multicast). An anology to unicast might perhaps be TCP streaming where a SYN is sent from the host that wants to receive the stream to the source of the stream. Then the application data flows in the reverse direction of the initial signal. Hence we argue that the above unicast scenarios correspond to the following multicast scenarios, respectively:

Scenario 1: An IPv6 network receiving multicast from the IPv4 Internet
Scenario 2: The IPv4 Internet receiving multicast from an IPv6 network
Scneario 3: The IPv6 Internet receiving multicast from an IPv4 network
Scenario 4: An IPv4 network receiving multicast from the IPv6 Internet
Scenario 5: An IPv6 network receiving multicast from an IPv4 network
Scenario 6: An IPv4 network receiving multicast from an IPv6 network

2.1. Scenario 1: An IPv6 network receiving multicast from the IPv4 Internet

Here we have a network, say ISP or enterprise, that for some reason is IPv6-only, but the hosts in the IPv6-only network should be able to receive multicast from sources in the IPv4 internet. The unicast equivalent is "IPv6 network to the IPv4 Internet".

This is simple because the global IPv4 address space can be embedded into IPv6 [RFC6052]. Unicast addresses according to the unicast translation in use. For multicast one may embed all IPv4 multicast addresses inside a single IPv6 multicast prefix. Or one may have multiple embeddings to allow for appropriate mapping of scopes and ASM versus SSM. Using this embedding, the IPv6 host (or an application running on the host) can send IPv6 MLD reports for IPv6 groups (and if SSM, also sources) that specify which IPv4 source and groups that it wants to receive. The usual IPv6 state (including MLD and possibly PIM) needs to be created. If PIM is involved we need to use RPF to set up the tree and accept data, so the source addresses must be routed towards some translation device. This is likely to be the same device that would do the unicast translation. The translation device can in this case be completely stateless. There is some multicast state, but that is similar to the state in a multicast router when translation is not performed. Basically if the translator receives MLD or PIM messages asking for a specific group (or source and group), it uses these mappings to find out which IPv4 group (or source and group) it needs to send IGMP or PIM messages for. This is no different than multicast in general, except for the translation. Whenever the translator receives data from the IPv4 source, it checks if it has anyone interested in the respective IPv6 group (or source and group), and if so, translates and forwards the data packets.

IPv6 applications need to somehow learn which IPv6 group (or source and group) to join. This is further discussed in Section 3.4.

2.2. Scenario 2: The IPv4 Internet receiving multicast from an IPv6 network

Here we will consider an IPv6 network connected to the IPv4 internet, and how any IPv4 host may receive multicast from a source in the IPv6 network. The unicast equivalent is "the IPv4 Internet to an IPv6 network".

This is difficult since the IPv6 multicast address space cannot be embedded into IPv4. Indeed this case has many similarities with how IPv4 networks can receive from the IPv6 Internet. See scenario (4), Section 2.4. However, in this case, all IPv4 hosts on the Internet should use the same mapping, and it might make sense to have additional requirements on the IPv6 network, rather than to add requirements for the IPv4 Internet.

One solution here might be for the IPv6 source application to somehow register with the translator to set up a mapping and receive an IPv4 address. The application could then possibly send SDP that includes both its IPv6 source and group, and the IPv4 source and group it got from the translator. Of course the signalling could also be done by manually adding a static mapping to the translator and specifying that address to the application. If instead we were to do signalling on the IPv4 side, then an IPv4 receiver would probably need a mechanism for finding an IPv4 address of the translator for a given IPv6 group. The IPv4 address could perhaps be embedded in the IPv6 group address? Or with say SDP there could be a way of specifying the IPv4 translator address. The IPv4 host could then communicate with the translator to establish a mapping (unless one exists) and learn which IPv4 group to join.

The best alternative might be to restrict the IPv6 multicast groups that should be accessible on the IPv4 internet to a certain IPv6 prefix. This may allow stateless translation. This could also be used in the reverse direction, for an IPv6 host to receive from an IPv4 source. Or in other words, the same mapping can be used in both directions. This has similarities with IVI [I-D.xli-behave-ivi], [RFC6145], [RFC6052] and also [I-D.venaas-behave-mcast46]. By using IVI source addresses (IPv4-translatable addresses) and a similar technique for multicast addresses, the correct IPv4 source and group addresses can be derived from those. This method has many benefits, the main issue is that it cannot work for arbitrary IPv6 multicast addresses.

2.3. Scenario 3: The IPv6 Internet receiving multicast from an IPv4 network

We here consider the case where the Internet is IPv6, but there is some network of perhaps legacy IPv4 hosts that is IPv4-only. We want any IPv6 host on the Internet to be able to receive multicast from an IPv4 source. The unicast equivalent is "the IPv6 Internet to an IPv4 network".
This scenario can be solved using the same techniques as in Scenario 1, Section 2.1. There may however be differences regarding exactly which mappings are used and how applications may become aware of them. To obtain full benefit of multicast, all IPv6 hosts need to use the same mappings.

2.4. Scenario 4: An IPv4 network receiving multicast from the IPv6 Internet

Here we consider how an IPv4-only host in an IPv4 network may receive from an IPv6 multicast sender on the Internet. The unicast equivalent is "an IPv4 network to the IPv6 Internet".
For dual-stack hosts in an IPv4 network one should consider tunneling. This is difficult since we cannot embed the entire IPv6 space into IPv4. One might consider some of the techniques from scenario (2), Section 2.2. That scenario is however much easier since one may restrict which IPv6 groups are used and there is a limited number of sources.
For unicast one might use a DNS-ALG for this, where the ALG would instantiate translator mappings as needed. This is the technique used in NAT-PT *[RFC2766]*, which was deprecated by [RFC4966].
However, for multicast one generally does not use DNS. One could consider doing the same with an ALG for some other protocol. E.g. translate addresses in SDP files when they pass the translator, or in any other protocol that might transfer multicast addresses. This would be very complicated and not recommended.
Rather than using an ALG that translates addresses in application protocol payload, one could consider new signalling mechanisms for more explicit signalling. The additional signalling could be either on the IPv6 or the IPv4 side. It may however not be a good idea to require additional behavior by host and applications on the IPv6 Internet to accomodate legacy IPv4 networks. Also, since one may not be able to

provide unique IPv4 multicast addresses for all the IPv6 multicast groups that are in use, it makes more sense that the mappings are done locally in each of the IPv4 networks, where IPv4 multicast addresses might be assigned on-demand. An IPv4 receiver might somehow request an IPv4 mapping for an IPv6 group (and possibly source). This creates a mapping in the translator so that when the IPv4 receiver joins the IPv4 group, the translator knows which IPv6 group (and possibly source) to translate it into. Of course the signalling could also be done manually by adding a static mapping to the translator and somehow specifying the right IPv4 address to the application.

2.5. Scenario 5: An IPv6 network receiving multicast from an IPv4 network

In this scenario we consider IPv4 and IPv6 networks belonging to the same organization. The unicast equivalent is "an IPv6 network to an IPv4 network".
We would like any IPv6 host to receive from any IPv4 sources. Here one can use the same techniques as for an IPv6 network receiving from the IPv4 internet. It is really a special case of scenario (1), Section 2.1.
The fact that the number of hosts are limited and that there is common management might simplify things. Due to the limited scale, one could perhaps just manually configure all the static mappings needed in the translator and manually create the necessary announcements or in some cases have the applications create the necessary announcements. But it might be better to use a stateless approach where IPv4 unicast and multicast addresses are embedded into IPv6. Like IVI [I-D.xli-behave-ivi], or [I-D.venaas-behave-mcast46].

2.6. Scenario 6: An IPv4 network receiving multicast from an IPv6 network

In this scenario we consider IPv4 and IPv6 networks belonging to the same organization. The unicast equialent is "an IPv4 network to an IPv6 network".
We would like any IPv4 host to receive from any IPv6 source. This can be seen as special cases of either scenario (2), Section 2.2 or scenario (4), Section 2.4, where any of those techniques might apply. However, as discussed in scenario (5) Section 2.5 where we looked at how to do multicast in the reverse direction; the limited number of hosts and common managment might allow us to just use static mappings or a stateless approach by restricting which IPv6 addresses are used. By using these techniques one may be able to create mappings that can be used for multicast in both directions, combining this scenario with scenario (5).

3. Framework

Having considered some possible scenarios for where and how we may use multicast translation, we will now consider some general issues and the different components of such solutions.

3.1. Addressing

When doing stateless translation, one need to somehow encode IPv4 addresses inside IPv6 addresses so that there is a well defined way for the translator to transform an IPv6 address into IPv4. This can be done with techniques like IVI [I-D.xli-behave-ivi] and [I-D.venaas-behave-mcast46].
There are two types of addressing schemes related to the IPv4/IPv6 multicast translation. The source addressing and the group addressing.

3.1.1. Source addressing

Source addressing issues is the same as in the unicast IPv4/IPv6 translation defined in [RFC6052]. The IPv4-mapped address is used for representing IPv4 in IPv6 and the IPv4-translatable address is used for representing IPv6 in IPv4 when the stateless translator is used. The multicast RPF relies on the source address to build the distribution tree. Therefore, depending on the operation mode of the IPv4/IPv6 translator and receiving directions, the IPv4-mapped or the IPv4-translatable addresses will be used.

3.1.2. Group addressing

Group addressing issue is unique to the IPv4/IPv6 multicast translation. The entire IPv4 group addresses can be uniquely represented by the IPv6 group addresses, while the entire IPv6 group addresses cannot be uniquely represented by the IPv6 group addresses. Therefore, special group address mapping rule between IPv4 group addresses and IPv6 group addresses should be defined for the IPv4/IPv6 multicast translation.

3.2. Routing

The actual translation of multicast packets may not be very complicated, in particular if it can be stateless. For the multicast to actually go through the translator we need to have routes for the multicast source addresses involved, so that multicast packets both on their way to and from the translator satisfy RPF checks. These routes are also needed for protocols like PIM-SM to establish a multicast tree, since RPF is used to determine where to send join messages. To go into more detail we need to look at different scenarios like SSM (Source-Specific Multicast) and ASM (Any-Source Multicast), and PIM versus IGMP/MLD.

3.2.1. Translation with PIM and SSM

When doing SSM, a receiver specifies both source and group addresses. If the receiver is to receive translated packets, it must do an IGMP/MLD join for the source and group address that the data packets will have after translation. We will later look at how it may learn those addresses. For the source address it joins, the unicast routing (or it may be an alternate topology specific to multicast), must point towards the translator. With this in place, PIM should build a tree hop-by-hop from the last-hop router to the translator. The translator then maps the source and group addresses in the PIM join to the source and group the data packets have before translation. The translator then does a PIM join for that source and group. Provided the routing is correct, this will then build a tree all the way to the source. Finally when these joins reach the source, any data sent by the source will follow this path to the translator, get translated, and then continue to the receiver.

3.2.2. Translation with PIM and ASM

Let us first consider PIM Sparse Mode. In this case a receiver just joins a group. If this group is to be received via the translator we need to send joins towards the translator, but initially PIM will send joins towards the RP (Rendezvous-Point) for the group. The most efficient solution is probably to make sure that the translator is configured as an RP for all groups that one may receive through it. That is, for the groups it translates to. E.g. if IPv4 groups are embedded into an IPv6 multicast prefix, then the translator could be an RP for that specific prefix. The translator may then translate the group and join towards the group address that is used before translation. Note that if the translator also is an RP for the addresses used before translation, it should know which sources exist and join each of these. If it is not an RP, it needs to join towards the RP. If the translator did not know the sources, it may join each of the sources as soon as it receives from them (that is, switching to Shortest Path Trees). When the translator receives data, it translates it and then sends the translated data. This then follows the joins for the translated groups to the receivers. When the last-hop routers start receiving, they will probably (this is usually the default behavior) switch to SPTs (Shortest Path Trees). These trees also need to go to the translator and would probably follow the same path as the previously built shared tree. One might argue here that switching to the SPT has no benefit if it is the same path anyway. Also with shared trees, RPF is not an issue, so the translated source addresses don't need to be routed towards the translator.
At the end of the previous paragraph we pointed out that there is no benefit in switching to shortest path trees if they have to go via the translator anyway. A possibility here could be to use Bidirectional PIM where there is no source specific state and data always go through the

RP. It is possible to use Bidir just for those groups that are translated, and then make the translator the RP.

3.2.3. Translation with IGMP/MLD

For translation taking place close to the edge, e.g. a home gateway, one may consider just using IGMP and MLD, and no PIM. In that case the translator should for any received MLD reports for IPv6 groups that correspond to translated IPv4 groups, map those into IGMP reports that it sends out on the IPv4 side. And vice versa for data in the other direction. Note that a translator implementation could also choose to do this in just one direction. For SSM it would also need to translate the source addresses.

3.3. Translation in operation

Currently, the proposed solutions for IPv6/IPv4 translation are classified into stateless translation and stateful translation.

3.3.1. Stateless Translation

For stateless translation, the translation information is carried in the address itself, permitting both IPv4->IPv6 and IPv6-<IPv4 sessions establishment. The stateless translation supports end-to- end address transparency and has better scalability compared with the stateful translation. See [RFC6145] and [I-D.xli-behave-ivi].
Stateless translation can be used for Scenarios 1, 2, 5 and 6, i.e. it supports "An IPv6 network receiving multicast from the IPv4 Internet", "the IPv4 Internet receiving multicast from an IPv6 network", "An IPv6 network receiving multicast from an IPv4 network" and "An IPv4 network receiving multicast from an IPv6 network".
In the stateless translation, an IPv6 network uses the IPv4-translatable addresses, while the IPv4 Internet or an IPv4 network can be represented by IPv4-mapped addresses.

```
      --------
   //        \\        -----------
  /            \     //          \\
 /             +----+              \
|              |XLAT|               |
|  The IPv4    +----+  An IPv6      |
|  Internet    +----+  Network      |  XLAT: Stateless v4/v6
|              |DNS |  (address     |        Translator
 \             +----+   subset)    /   DNS:  DNS64/DNS46
  \            /     \\          //
   \\        //        ----------
     --------
        sending ----> receiving
      receiving <---- sending
```

```
     --------          ---------
  //        \\      //          \\
 /             +----+              \
|              |XLAT|               |
|  An IPv4     +----+  An IPv6      |
|  Network     +----+  Network      |  XLAT: v4/v6
|              |DNS |               |        Translator
 \             +----+              /   DNS:  DNS64/DNS46
   \\        //      \\          //
     --------          ---------
        sending ----> receiving
      receiving <---- sending
```

3.3.2. **Stateful Translation**

For stateful translation, the translation state is maintained between IPv4 address/port pairs and IPv6 address/port pairs, enabling IPv6 systems to open sessions with IPv4 systems. See [RFC6145] and [RFC6146].
Stateful translator can be used for Scenarios 1, 3 and 5, i.e. it supports "An IPv6 network receiving multicast from the IPv4 Internet", "The IPv6 Internet receiving multicast from an IPv4 network" and "An IPv6 network receiving multicast from an IPv4 network".
In the stateful translation, an IPv6 network or the IPv6 Internet use any IPv6 addresses, while the IPv4 Internet or an IPv4 network can be represented by IPv4-mapped addresses.

```
     --------
   //        \\       -----------
  /            \     //          \\
 /             +----+              \
|              |XLAT|               |
|  The IPv4    +----+  An IPv6      |
|  Internet    +----+  Network      |  XLAT: Stateful v4/v6
|              |DNS |               |        Translator
 \             +----+              /   DNS:  DNS64
  \            /     \\          //
   \\        //       -----------
     --------
        sending ----> receiving
```

```
                      -----------
    ----------       //          \\
   //          \\    /             \
 /             +----+              \
|              |XLAT|               |
|  An IPv4     +----+  The IPv6     |
|  Network     +----+  Internet     |  XLAT: v4/v6
|              |DNS |               |        Translator
 \             +----+               /  DNS:  DNS64
  \\        //      \             /
    ---------         \\         //
                      -----------
        sending ----> receiving
```

```
     --------          ---------
  //        \\      //          \\
 /             +----+              \
|              |XLAT|               |
|  An IPv4     +----+  An IPv6      |
|  Network     +----+  Network      |  XLAT: v4/v6
|              |DNS |               |        Translator
 \             +----+              /   DNS:  DNS64
   \\        //      \\          //
     --------          ---------
        sending ----> receiving
```

3.4. Application layer issues

The main application layer issue is perhaps how the applications learn
what groups (or sources and groups) to join. For unicast, applications

may often obtain addresses via DNS and a DNS-ALG. For multicast, DNS is usually not used, and there are a wide range of different ways applications learn addresses. It can be through configuration or user input, it can be URLs on a web page, it can be SDP files (via SAP or from web page or mail etc), or also via protocols like RTSP/SIP. It is no easy task to handle all of these possible methods using ALGs.

SDP is maybe the most common way for applications to learn which multicast addresses (and other parameters) to use in order to receive a multicast session. Inside the SDP files it is common to use literal IP addresses, but it is also possible to specify domain names. Applications would then query the DNS for the addresses, and a DNS-ALG could perform the necessary translation. There is however a problem with this.

Here is a typical SDP taken from RFC 2327:

```
         v=0
         o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
         s=SDP Seminar
         i=A Seminar on the session description protocol
         u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
         e=mjh@isi.edu (Mark Handley)
         c=IN IP4 224.2.17.12/127
         t=2873397496 2873404696
         a=recvonly
         m=audio 49170 RTP/AVP 0
         m=video 51372 RTP/AVP 31
         m=application 32416 udp wb
         a=orient:portrait
```

The line of interest here is "c=IN IP4 224.2.17.12/127". It is legal to use a domain name, this line would then become e.g. "c=IN IP4 mcast.example.com/127". The problem here is that the application is told to use IPv4. It will expect the name to resolve to an IPv4 address, and may ignore any IPv6 replies. One could argue that it would be incorrect to use IPv6, since IPv4 is specified. For DNS to solve our problem, we would need a new IP neutral SDP syntax, and applications would need to be updated to support it.

An alternative to rewriting addresses in the network is to make the applications aware of the translation and mappings in use. One approach could be for the source to create say SDP that includes both the original and the translated addresses. This may require use of techniques like CCAP [I-D.boucadair-mmusic-ccap] for specifying both IPv4 and IPv6 multicast addresses, allowing the receiver to choose which one to use. The other alternative would be for the receiving application to be aware of the translation and the mapping in use. This means that the receiving application can receive the original SDP, but then apply the mapping to those addresses.

As we just discussed, it may be useful for applications to perform the mappings. The next question is how they may learn those mappings. The

easiest would be if there was a standard way used for all mappings, e.g. a well-known IPv6 prefix for embedding IPv4 addresses. But that does not work in all scenarios. There could be a way for applications to learn which prefix to use, see [I-D.wing-behave-learn-prefix]. But note that there may be different multicast prefixes depending on whether we are doing SSM or ASM and scope. In addition we need the unicast prefix for the multicast source addresses. Alternatively one could imagine applications requesting mappings for specific addresses on demand from the translator. The translator could have static mappings, or install mappings as requested by applications.
An alternative to making applications aware of the translation and rewriting addresses as needed, could be to do translation in the API or stack, so that e.g. an application joins an IPv4 group, the API or stack rewrites that into IPv6 and sends the necessary MLD reports. When IPv6 packets arrive, the API/stack can rewrite those packets back to IPv4. This could allow legacy IPv4 applications to run on a dual-stack node (or IPv6-only with translation in the API) to receive IPv4 packets through an IPv6-only network. But in this case it might be better to just use tunneling.

3.5. Further Work

There are some special cases and scenarios that should be added to this document. One is addressing. Are there certain types of IPv6 multicast addresses that could make translation easier? What happens if there are multiple translators? And also more details on translation in the host, e.g. bump-in-the-stack or bump-in-the-API.
The document layout of the IPv4/IPv6 multicast translation should be presented in this document.

4. IANA Considerations

This document requires no IANA assignments.

5. Security Considerations

This requires more thought, but the author is not aware of any obvious security issues specific to multicast translation.

6. Acknowledgements

Dan Wing provided early feedback that helped shape this document. Dave Thaler also provided good feedback that unfortunately still has not been addressed in this document. See Section 3.5.

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.
[RFC2766]	Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February 2000.
[RFC4966]	Aoun, C. and E. Davies, "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status", RFC 4966, July 2007.
[RFC5245]	Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.
[RFC6052]	Bao, C., Huitema, C., Bagnulo, M., Boucadair, M. and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.
[RFC6144]	Baker, F., Li, X., Bao, C. and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.
[RFC6145]	Li, X., Bao, C. and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.
[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.

7.2. Informative References

[I-D.xli-behave-ivi]	Li, X, Bao, C, Chen, M, Zhang, H and J Wu, "The CERNET IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition", Internet-Draft draft-xli-behave-ivi-07, January 2010.
[I-D.venaas-behave-mcast46]	Venaas, S, Asaeda, H, SUZUKI, S and T Fujisaki, "An IPv4 - IPv6 multicast translator", Internet-Draft draft-venaas-behave-mcast46-02, December 2010.
[I-D.wing-behave-learn-prefix]	Wing, D, "Learning the IPv6 Prefix of a Network's IPv6/IPv4 Translator", Internet-Draft draft-wing-behave-learn-prefix-04, October 2009.
[I-D.boucadair-mmusic-ccap]	Boucadair, M and H Kaplan, "Session Description Protocol (SDP) Connectivity Capability (CCAP) Attribute", Internet-Draft draft-boucadair-mmusic-ccap-00, July 2009.

Authors' Addresses

Stig Venaas Venaas cisco Systems
Tasman Drive San Jose, CA 95134 USA EMail: stig@cisco.com

Xing Li Li CERNET Center/Tsinghua University Room 225, Main Building, Tsinghua University Beijing, 100084 CN Phone: +86 10-62785983 EMail: xing@cernet.edu.cn

Congxiao Bao Bao CERNET Center/Tsinghua University Room 225, Main Building, Tsinghua University Beijing, 100084 CN Phone: +86 10-62785983 EMail: congxiao@cernet.edu.cn