behave	F. Baker, Ed.	
Internet-Draft	Cisco Systems	
Intended status: Standards Track	X. Li, Ed.	
Expires: August 28, 2009	C. Bao, Ed.	
	CERNET Center/Tsinghua University	
	February 24, 2009	

TOC

Framework for IPv4/IPv6 Translation
draft-baker-behave-v4v6-framework-02

Status of this Memo

Copyright Notice

Abstract

This note describes a framework for IPv4/IPv6 translation. This is in the context of replacing NAT-PT, which was deprecated by RFC 4966, and to enable networks to have IPv4 and IPv6 coexist in a somewhat rational manner while transitioning to an IPv6 network.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.) [RFC2119].

Table of Contents

---

1. Introduction

TOC

This note describes a framework for IPv4/IPv6 translation. This is in the context of replacing NAT-PT (Tsirtsis, G. and P. Srisuresh, “Network Address Translation - Protocol Translation (NAT-PT),” February 2000.) [RFC2766], which was deprecated by [RFC4966] (Aoun, C. and E. Davies, “Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status,” July 2007.), and to enable networks to have IPv4 and IPv6 coexist in a somewhat rational manner while transitioning to an IPv6-only network.
Deprecation of NAT-PT wasn't intended to say that NAT-PT was "bad", nor did the IETF think that deprecating the technology would stop people from using it. As with the 1993 deprecation of the RIP routing protocol at the time the Internet was converting to CIDR, the point was to inform the community that NAT-PT had operational issues and was not considered a viable medium or long term strategy for either coexistence or transition. The point was to encourage network operators to actually move in the direction of transition.
[RFC4213] (Nordmark, E. and R. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers,” October 2005.) describes the IETF's view of the most sensible transition model. The IETF recommends, in short, that network operators (transit providers, service providers, enterprise networks, small and medium business, SOHO and residential customers, and any other kind of network that may currently be using IPv4) obtain an IPv6 prefix, turn on IPv6 routing within their networks and between themselves and any peer, upstream, or downstream neighbors, enable it on their computers, and use it in normal processing. This should be done while leaving IPv4 stable, until a point is reached that

any communication that can be carried out could use either protocol equally well. At that point, the economic justification for running both becomes debatable, and network operators can justifiably turn IPv4 off. This process is comparable to that of [RFC4192] (Baker, F., Lear, E., and R. Droms, "Procedures for Renumbering an IPv6 Network without a Flag Day," September 2005.), which describes how to renumber a network using the same address family without a flag day. While running stably with the older system, deploy the new. Use the coexistence period to work out such kinks as arise. When the new is also running stably, shift production to it. When network and economic conditions warrant, remove the old, which is now no longer necessary.
The question arises: what if that is infeasible due to the time available to deploy or other considerations? What if the process of moving a network and its components or customers is starting too late for contract cycles to effect IPv6 turn-up on important parts at a point where it becomes uneconomical to deploy global IPv4 addresses in new services? How does one continue to deploy new services without balkanizing the network?
This set of documents describes translation as one of the tools networks might use to facilitate coexistence and ultimate transition.

1.1. Why translation?

TOC

Besides dual stack deployment, there are two fundamental approaches one could take to interworking between IPv4 and IPv6: tunneling and translation. One could - and in the 6NET we did - build an overlay network using the new protocol inside tunnels. Various proposals take that model, including 6to4 (Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds," February 2001.) [RFC3056], Teredo (Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)," February 2006.) [RFC4380], ISATAP (Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)," March 2008.) [RFC5214],and DS-Lite (Durand, A., Droms, R., Haberman, B., and J. Woodyatt, "Dual-stack lite broadband deployments post IPv4 exhaustion," November 2008.)
[I-D.durand-softwire-dual-stack-lite]. The advantage of doing so is that the new is enabled to work without disturbing the old protocol, providing connectivity between users of the new protocol. There are two disadvantages to tunneling:

* operators of those networks are unable to offer services to users of the new architecture, and those users are unable to use the services of the underlying infrastructure - it is just bandwidth, and

*it doesn't enable new protocol users to communicate with old protocol users.

As noted, in this work, we look at Internet Protocol translation as a transition strategy. [RFC4864] (Van de Velde, G., Hain, T., Droms, R., Carpenter, B., and E. Klein, "Local Network Protection for IPv6," May 2007.) forcefully makes the point that many of the reasons people use Network Address Translators are met as well by routing or protocol mechanisms that preserve the end to end addressability of the Internet. What it did not consider is the case in which there is an ongoing requirement to communicate with IPv4 systems, but configuring IPv4 routing is not in the network operator's view the most desirable strategy, or is infeasible due to a shortage of global address space. Translation enables the client of a network, whether a transit network, an access network, or an edge network, to access the services of the network and communicate with other network users regardless of their protocol usage - within limits. Like NAT-PT, IPv4/IPv6 translation under this rubric is not a long term support strategy, but it is a medium term coexistence strategy that can be used to facilitate a long term program of transition.

1.2. Terminology

TOC

The following terminology is used in this document and other documents related to it.

CPE: The acronym expands to "Customer Premises Equipment"; in the context of this document set, it refers to the router in front of a host, whether in an ISP or other network environment, that performs some relevant function such as advertising a specific prefix.

Dual Stack implementation: A Dual Stack implementation, in this context, comprises an enabled end system stack plus routing in the network. It implies that two application instances are capable of communicating using either IPv4 or IPv6 - they have stacks, they have addresses, and they have any necessary network support including routing.

IPv4 capable node: A node which has an IPv4 protocol stack. Apart from the local LAN, where link-local (169.254.0.0/16) addresses might be used without operational intent, an interface on the node must be assigned one or more IPv4 addresses for the stack to be usable.

IPv4 enabled node: A node which has an IPv4 protocol stack and is assigned one or more IPv4 addresses that are not link-local

(169.254.0.0/16). Both IPv4-only and IPv6/IPv4 nodes are IPv4 enabled.

IPv6 capable node: A node which has an IPv6 protocol stack. Apart from the local LAN, where link-local (FE80::/64) addresses might be used without operational intent, an interface on the node must be assigned or must autoconfigure one or more IPv6 addresses for the stack to be usable.

IPv6 enabled node: A node which has an IPv6 protocol stack and one or more IPv6 addresses that are not link-local (FE80::/64). Both IPv6-only and IPv6/IPv4 nodes are IPv6 enabled.

IPv4-embedded IPv6 addresses: They are the IPv6 addresses which have unique relationship to specific IPv4 addresses. This relationship is self described by embedding IPv4 address in the IPv6 address. The IPv4-embedded IPv6 addresses are used for both the statesless and the stateful modes.

IPv4-related IPv6 addresses: They are the IPv6 addresses which have unique relationship to specific IPv4 addresses. This relationship is maintained as the states (mapping table between IPv4 address/transport_port and IPv6 address/transport_port) in the IP/ICMP translator. The states are session initiated. The IPv4-related IPv6 addresses are used fo the stateful mode only.

IPv4-only: An IPv4-only implementation, in this context, comprises an enabled end system stack plus routing in the network. It implies that two application instances are capable of communicating using IPv4, but not IPv6 - they have an IPv4 stack, addresses, and network support including IPv4 routing and potentially IPv4/IPv4 translation, but some element is missing that prevents communication using IPv6.

IPv6-only: An IPv6-only implementation, in this context, comprises an enabled end system stack plus routing in the network. It implies that two application instances are capable of communicating using IPv6, but not IPv4 - they have an IPv6 stack, addresses, and network support including routing in IPv6, but some element is missing that prevents communication using IPv4.

LIR Prefix: The IPv6 prefix assigned by the network operator for embedding IPv4 addresses into IPv6 addresses. In this case, each network running a translator will create a representation of the whole IPv4 address space in the IPv6 address space.

LIR: See Local Internet Registry.

Local Internet Registry: A Local Internet Registry (LIR) is an organization which has received an IP address allocation from a

Regional Internet Registry (RIR), and which may assign parts of this allocation to its own internal network or those of its customers. An LIR is thus typically an Internet service provider or an enterprise network.

Physical IPv4 address pool: Zero or more IPv4 addresses used by the translator in stateful translation.

State: "State" refers to dynamic information that is stored in a network element. For example, if two systems are connected by a TCP connection, each stores information about the connection, which is called "connection state". In this context, the term refers to dynamic correlations between IP addresses on either side of a translator, or {IP Address, Transport type, transport port number} tuples on either side of the translator. Of stateful algorithms, there are at least two major flavors depending on the kind of state they maintain:

> **Hidden state:** the existence of this state is unknown outside the network element that contains it.
>
> **Known state:** the existence of this state is known by other network elements.

Stateful Translation: A translation algorithm may be said to "require state in a network element" or be "stateful" if the transmission or reception of a packet creates or modifies a data structure in the relevant network element.

Stateless Translation: A translation algorithm that is not "stateful" is "stateless". It may require configuration of a static translation table, or may derive its needed information algorithmically from the messages it is translating.

Well-Known Prefix: The IPv6 prefix assigned by IANA for embedding IPv4 addresses into IPv6 addresses. In this case, there would be a single representation of a public IPv4 address in the IPv6 address space.

1.3. Translation objectives

TOC

In any translation model, there is a question of objectives. Ideally, one would like to make any system and any application running on it able to "talk with" - exchange datagrams supporting applications - with any other system running the same application regardless of whether they have an IPv4 stack and connectivity or IPv6 stack and

connectivity. That was the model NAT-PT, and the things it necessitated led to scaling and operational difficulties.
So the question comes back to what different kinds of connectivity can be easily supported and what are harder, and what technologies are needed to at least pick the low-hanging fruit. We observe that applications today fall into three main categories:

> **Client/Server Application:** Per whatis.com, "'Client/server' describes the relationship between two computer programs in which one program, the client, makes a service request from another program, the server, which fulfills the request." In networking, the behavior of the applications is that connections are initiated from client software and systems to server software and systems. Examples include mail handling between an end user and his mail system (POP3, IMAP, and MUA->MTA SMTP), FTP, the web, and DNS name translation.
>
> **Peer to Peer Application:** Peer to peer applications are those that transfer information directly, rather than through the use of an intermediate repository such as a bulletin board or database. In networking, any system (peer) might initiate a session with any other system (peer) at any time. These in turn fall broadly into two categories:
>
> > **Peer to peer infrastructure applications:** Examples of "infrastructure applications" include SMTP between MTAs, Network News, and SIP. Any MTA might open an SMTP session with any other at any time; any SIP Proxy might similarly connect with any other SIP Proxy. An important characteristic of these applications is that they use ephemeral sessions - they open sessions when they are needed and close them when they are done.
> >
> > **Peer to peer file exchange applications:** Examples of these include Limewire, BitTorrent, and UTorrent. These are applications that open some sessions between systems and leave them open for long periods of time, and where ephemeral sessions are important, are able to learn about the reliability of peers from history or by reputation. They use the long term sessions to map content availability. Short term sessions are used to exchange content. They tend to prefer to ask for content from servers that they find reliable and available.

NAT-PT is an example of a facility with known state - at least two software components (the data plane translator and the DNS Application Layer Gateway, which may be implemented in the same or different systems) share and must coordinate translation state. A typical IPv4/IPv4 NAT implements an algorithm with hidden state. Obviously,

stateless translation requires less computational overhead than stateful translation, and less memory to maintain the state, because the translation tables and their associated methods and processes exist in a stateful algorithm and don't exist in a stateless one.
If the key questions are the ability to open connections between systems, then one must ask who opens connections.

- We need a technology that will enable systems that act as clients to be able to open sessions with other systems that act as servers, whether in the IPv6->IPv4 direction or the IPv4->IPv6 direction. Ideally, this is stateless; especially in a carrier infrastructure, the preponderance of accesses will be to servers, and this optimizes access to them. However, a stateful algorithm is acceptable if the complexity is minimized and a stateless algorithm cannot be constructed.

- We also need a technology that will allow peers to connect with each other, whether in the IPv6->IPv4 direction or the IPv4->IPv6 direction. Again, it would be ideal if this was stateless, but a stateful algorithm is acceptable if the complexity is minimized and a stateless algorithm cannot be constructed. In the case of infrastructure applications, which know nothing of choosing among peers by reputation, the IPv4->IPv6 direction is a stronger requirement. Peer to peer file exchange applications, however, may be more forgiving - it may well be adequate to make a subset of IPv4->IPv6 connections work instead of all. (EDITOR'S NOTE: I would be very interested in comments on this assertion)

- We do not need an algorithm that enables clients to connect to clients, because they don't connect.

The complexity arguments bring us in the direction of hidden state: if state must be shared between the application and the translator or between translation components, complexity and deployment issues are greatly magnified. We would very much prefer that any software changes be confined to the translator.

1.4. Transition Plan

TOC

While the design of IPv4 made it impossible for IPv6 to be compatible on the wire, the designers intended that it would coexist with IPv4 during a period of transition. The primary mode of coexistence was dual-stack operation - routers would be dual-stacked so that the network could carry both address families, and IPv6-capable hosts could be dual-stack to maintain access to IPv4-only partners. The goal was that the preponderance of hosts and routers in the Internet would be IPv6-capable long before IPv4 address space allocation was completed.

At this time, it appears the exhaustion of IPv4 address space will occur before significant IPv6 adoption.
Curran's "A Transition Plan for IPv6" (Curran, J., “An Internet Transition Plan,” July 2008.) [RFC5211] proposes a three-phase progression:

> **Preparation Phase (current):** characterized by pilot use of IPv6, primarily through transition mechanisms defined in RFC 4213, and planning activities.
>
> **Transition Phase (2010 through 2011):** characterized by general availability of IPv6 in provider networks which SHOULD be native IPv6; organizations SHOULD provide IPv6 connectivity for their Internet-facing servers, but SHOULD still provide IPv4-based services via a separate service name.
>
> **Post-Transition Phase (2012 and beyond):** characterized by a preponderance of IPv6-based services and diminishing support for IPv4-based services.

In each of these phases, the coexistence problem and solution space has a different focus:

> **Preparation Phase:** Coexistence tools are needed to facilitate early adopters by removing impediments to IPv6 deployment, and to assure that nothing is lost by adopting IPv6, in particular that the IPv6 adopter has unfettered access to the global IPv4 Internet regardless of whether they have a global IPv4 address (or any IPv4 address or stack at all.) While it might appear reasonable for the cost and operational burden to be borne by the early adopter, the shared goal of promoting IPv6 adoption would argue against that model. Additionally, current IPv4 users should not be forced to retire or upgrade their equipment and the burden remains on service providers to carry and route native IPv4.
>
> **Transition Phase:** While IPv6 adoption can be expected to accelerate, there will still be a significant portion of the Internet operating in IPv4-only or preferring IPv4. During this phase the norm shifts from IPv4 to IPv6, and coexistence tools evolve to ensure interoperability between domains that may be restricted to IPv4 or IPv6.
>
> **Post-Transition Phase:** In this phase, IPv6 is ubiquitous and the burden of maintaining interoperability shifts to those who choose to maintain IPv4-only systems. While these systems should be allowed to live out their economic life cycles, the IPv4-only legacy users at the edges should bear the cost of coexistence tools, and at some point service provider networks should not be expected to carry and route native IPv4 traffic.

The choice between the terms "transition" versus "coexistence" has engendered long philosophical debate. "Transition" carries the sense that we are going somewhere, while "coexistence" seems more like we are sitting somewhere. Historically with IETF, "transition" has been the term of choice [RFC4213] (Nordmark, E. and R. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers,” October 2005.) [RFC5211] (Curran, J., “An Internet Transition Plan,” July 2008.), and the tools for interoperability have been called "transition mechanisms". There is some perception or conventional wisdom that adoption of IPv6 is being impeded by the deficiency of tools to facilitate interoperability of nodes or networks that are constrained (in some way, fully or partially) from full operation in one of the address families. In addition, it is apparent that transition will involve a period of coexistence; the only real question is how long that will last.
Thus, coexistence is an integral part of the transition plan, not in conflict with it, but there will be a balancing act. It starts out being a way for early adopters to easily exploit the bigger IPv4 Internet, and ends up being a way for late/never adopters to hang on with IPv4 (at their own expense, with minimal impact or visibility to the Internet). One way to look at solutions is that cost incentives (both monetary cost and the operational overhead for the end user) should encourage IPv6 and discourage IPv4. That way natural market forces will keep the transition moving - especially as the legacy IPv4-only stuff ages out of use. There will come a time to set a date after which no one is obligated to carry native IPv4 but it would be premature to attempt to do so yet. The end goal should not be to eliminate IPv4 by fiat, but rather render it redundant through ubiquitous IPv6 deployment. IPv4 may never go away completely, but rational plans should move the costs of maintaining IPv4 to those who insist on using it after wide adoption of IPv6.

1.5. Scenarios of the IPv4/IPv6 translation

TOC

There are four types of IPv4/IPv6 translation scenarios, including
(1) Connecting between the IPv4 Internet and the IPv6 Internet
(2) Connecting an IPv6 network to the IPv4 Internet
(3) Connecting an IPv4 network to the IPv6 Internet
(4) Connecting between an IPv4 network and an IPv6 network
Each one in the above can be divided into two subscenarios, including the IPv6 initiated communication and the IPv4 initiated communication. So there are eight subscenarios.
Note that in order to perform the required function, the translator needs to represent the IPv4 addresses in the IPv6 Internet and the IPv6 addresses in the IPv4 Internet. We will evaluate the four types of scenarios and their requirements of the address space.

1.5.1. Connecting between the IPv4 Internet and the IPv6 Internet

TOC

This is the ideal translation case. However, due to the hugh difference between the address spaces of IPv4 and IPv6, it is impossible to represent the global IPv6 address space in IPv4, so a general solution for this case does not exist.

1.5.2. Connecting an IPv6 network to the IPv4 Internet

TOC

Due to the lack of the public routable IPv4 addresses or under other technical or economical constrains, the ISP's or enterprise's network is IPv6-only, but the hosts in the network require to communicate with the global IPv4 Internet. This is a finite state problem, since the number of IPv6 hosts in the ISP's or enterprise's network is limited and the global IPv4 addresses can easily be embedded in the ISP's or enterprise's IPv6 address space.
In this case, the initiation-direction of the communication makes things interesting. The IPv6 initiated communication is relatively easy, since the global IPv4 addresses can be embedded in the ISP's or enterprise's IPv6 block (usually a /48). However, the IPv4 initiated communication is hard, since it is no one-to-one mapping between the IPv6 /48 (or even /64) and the IPv4 address pool.
In order to provide the solution for this case, there are three techniques.
(1) Using tightly coupled SIIT [RFC2765] (Nordmark, E., “Stateless IP/ICMP Translation Algorithm (SIIT),” February 2000.) and DNS-ALG (DNS Application Layer gateway) presented in NAT-PT [RFC2766] (Tsirtsis, G. and P. Srisuresh, “Network Address Translation - Protocol Translation (NAT-PT),” February 2000.). This is a stateful translation scheme. However, due to the scalability and other problems, this technique is deprecated by IETF [RFC4966] (Aoun, C. and E. Davies, “Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status,” July 2007.).
(2) Only support IPv6 initiated communication as presented in NAT66 [I-D.bagnulo-behave-nat64] (Bagnulo, M., Matthews, P., and I. Beijnum, “NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers,” March 2009.), it is also a stateful translation scheme, but without tightly-decoupled DNS-ALG [I-D.bagnulo-behave-dns64] (Bagnulo, M., Sullivan, A., Matthews, P., Beijnum, I., and M. Endo, “DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers,” March 2009.). However, it cannot make IPv6-only servers accessible by IPv4-only hosts.

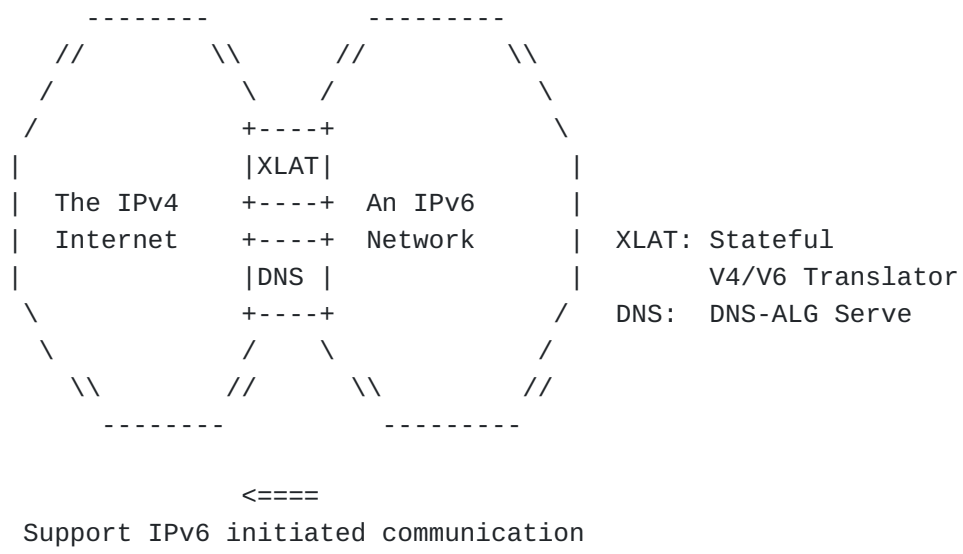

Figure 1: NAT64

(3) Select a subset of the IPv6 addresses in the ISP's or enterprise's network by embedding the IPv4 addresses in them as presented in IVI [I-D.baker-behave-ivi] (Li, X., Bao, C., Baker, F., and K. Yin, "IVI Update to SIIT and NAT-PT," September 2008.). These IPv6 addresses (called IVI addresses) can support both IPv4 initiated communication and IPv6 initiated communications without tightly-decoupled DNS-ALG. In addition, this kind of translation is stateless and has good features such as better scalability, supporting multiple translators.

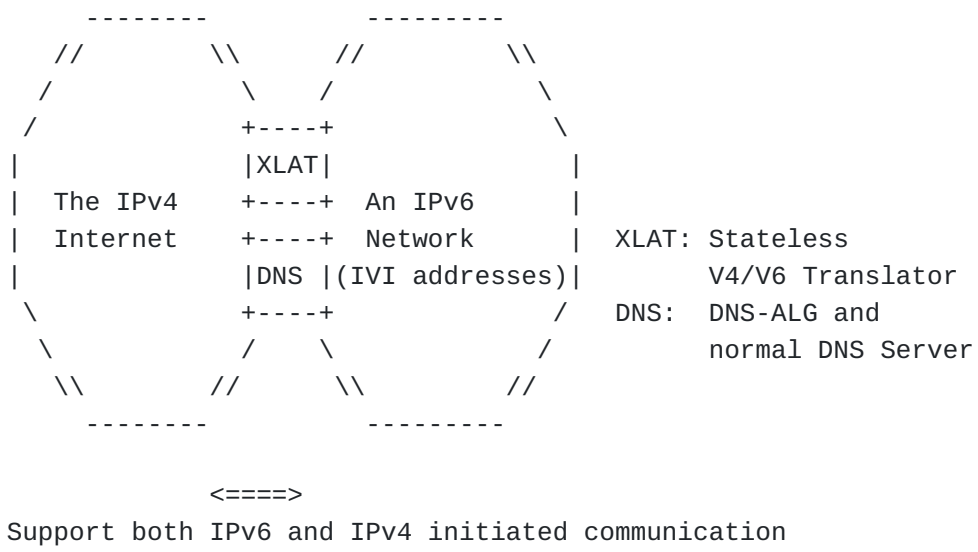

Figure 2: IVI

In order to save the public IPv4 addresses, the transport-layer port multiplexing techniques can be used in this case.

1.5.3. Connecting an IPv4 network to the IPv6 Internet

TOC

Due to the technical or economical constrains, the ISP's or enterprise's network is IPv4-only, but the IPv4-only hosts require the communicate with the global IPv6 Internet. This is not a finite state problem, since there is no way to represent the global IPv6 address space using the IPv4 addresses. When the size of the IPv6 Internet reaches a certain value, it is not practical to provide the translation service for a big ISP or enterprise, therefore the dual stack solution should be used. This is to say that one SHOULD do IVI in parts of the network being built from scratch, while IPv4 parts are becoming dual stack.
However, there is a requirement for the legacy IPv4 hosts to provide services to the IPv6 hosts. The key issue for this case is to use a pool of public IPv4 addresses or [RFC1918] (Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, “Address Allocation for Private Internets,” February 1996.) address to represent IPv6 in IPv4. Since the number of concurrent sessions for a IPv4 server or a pool of server is limited, it is possble to do translation in this case.
Based on the above discussion, the IPv6 initiated communication can be achieved without DNS-ALG.

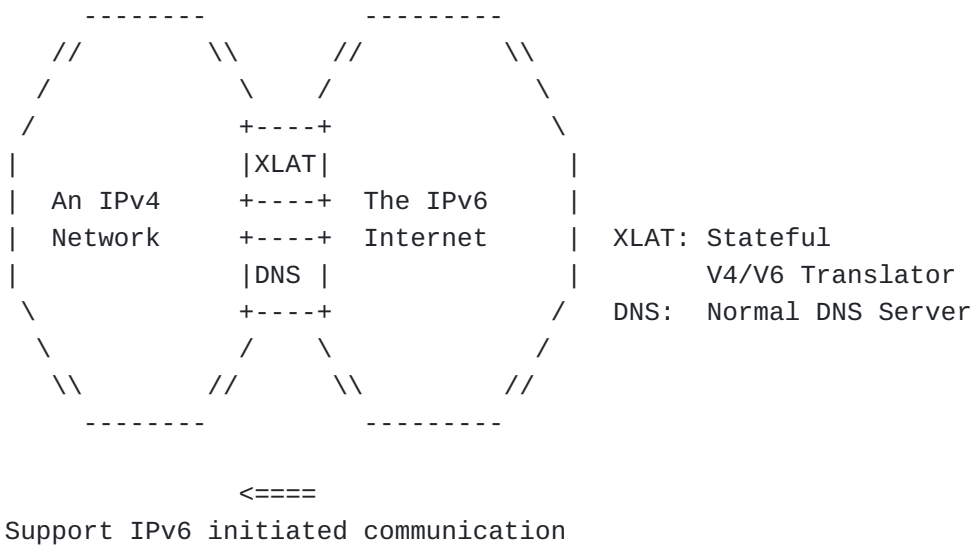

Figure 3: NAT64 type 2

In order to save the public or [RFC1918] (Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, “Address Allocation for Private Internets,” February 1996.) IPv4 addresses, the transport layer port multiplexing techniques can be used in this case.

1.5.4. Connecting between an IPv4 network and an IPv6 network

TOC

This is the case when both IPv6-to-IPv4 and IPv4-to-IPv6 translation are within the same organization. Therefore, it is a finite state problem, since the number of IPv4 hosts and IPv6 hosts are limited. In this case, the initiation-direction of the communication does not play an important role due to the finite state nature. The IPv4 addresses used are either public IPv4 addresses or [RFC1918] (Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, “Address Allocation for Private Internets,” February 1996.) addresses. The IPv6 addresses used are either public IPv6 addresses or ULA (Unique Local Address) (Hinden, R. and B. Haberman, “Unique Local IPv6 Unicast Addresses,” October 2005.) [RFC4193]. Both stateless and stateful translation scheme can be used for this case.

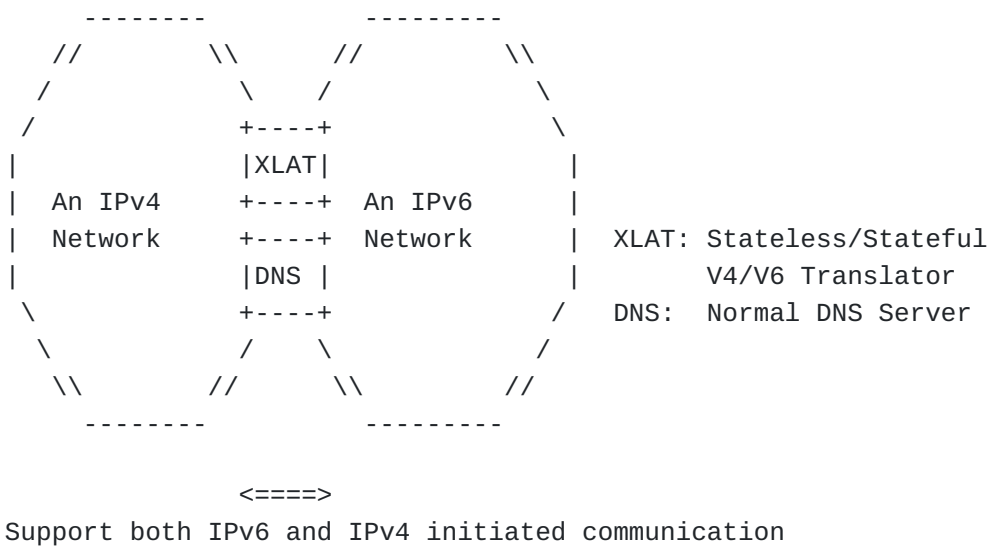

Figure 4: Same Organization

1.6. Expected uses of translation

TOC

There are several potential uses of translation. They are all easily described in terms of "interoperation between a set of systems that only communicate using IPv4 and a set of systems that only communicate using IPv6", but the differences at a detail level make them interesting. At minimum, these include:

- Connection of IPv4-only islands to an IPv6-only network, which might include
  - Connecting a small pool of legacy equipment with a view to eventual obsolescence
  - Connecting a legacy network with a view to eventual transition.
- Connection of IPv6-only islands to an IPv4-only network
- Connecting IPv4-only devices with IPv6-only devices regardless of network type

*Connections between IPv4-only networks and IPv6-only networks, especially as a service within a large network such as an enterprise or ISP network or between peer networks.

1.6.1. Connection of IPv4-only islands to an IPv6-only network

TOC

While the basic issue is the same, there are at least two interesting special cases of this: connecting a small pool of legacy equipment with a view to eventual obsolescence, and connecting a legacy network with a view to eventual transition.

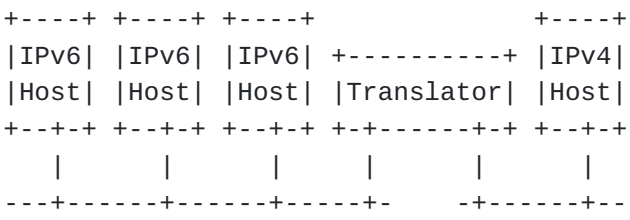

Figure 5: Printer pool or other legacy equipment

In the first case, Figure 5 (Printer pool or other legacy equipment), one might have a pool of equipment (printers, perhaps) that is IPv4-capable, but either the network it serves or some equipment in that network is IPv6-only. One pools the IPv4-only devices behind a translator, which enables IPv6-only systems to connect to the IPv4-only equipment. If the network is dual stack and only some of the equipment is IPv6-only, the translator should be a function of a router, and the router should provide normal IPv4 routing services as well as IPv6->IPv4 translation.

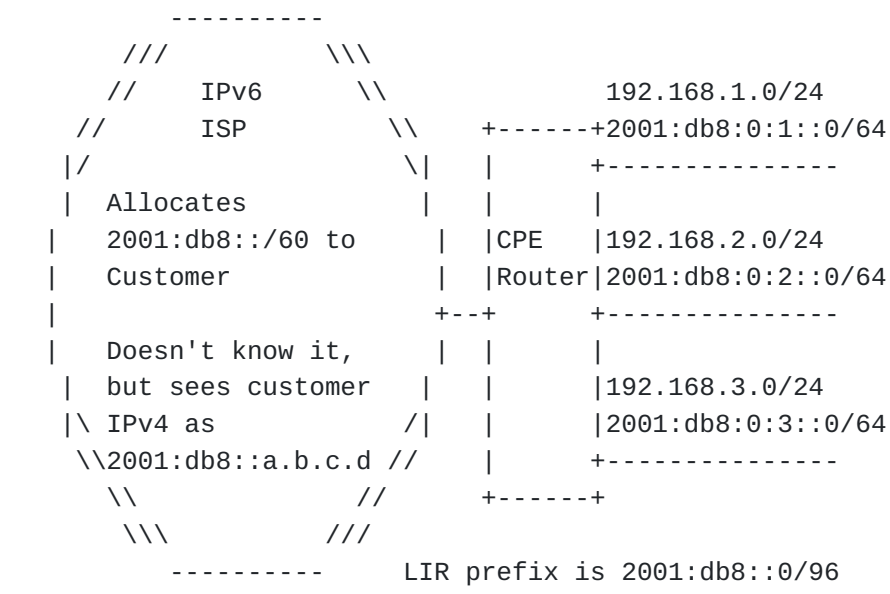

Figure 6: Customer dual stack network

Figure 6 (Customer dual stack network) creates transition options to a customer network connected to an IPv6-only ISP, or some equivalent relationship. The customer might internally be using traditional IPv4 with NAT services, and the ISP might change its connection to an IPv6-only network and encourage it to transition. If the ISP assigns a /60 prefix to a SOHO, for example, the CPE router in the SOHO could distribute several dual stack subnets internally, one for wireless and one for each of several fixed LANs (the entertainment system, his office, her office, etc). One of the /64 prefixes would be dedicated to representing the SOHO's IPv4 addresses in the ISP or the IPv4 network beyond it, and the other prefixes for the various internal subnets. Internally, the subnets might carry prefix pairs 192.168.n.0/24 and 2001:db8:0.n::/64 for n in 1..15 (1..0xF), and externally might appear as 2001:db8:0:n::/64 for the IPv6 subnets and 2001:db8::192.168.n.0/120 for the IPv4 devices. Note that to connect to an IPv4-only network beyond, RFC 1918 addresses would have to be statefully mapped using traditional IPv4 mechanisms somewhere; if this is done by the ISP, collusion on address mapping is required, and the case in Section 1.6.4 (ISP-supported connections between IPv4-only networks and IPv6-only networks) is probably a better choice.
In this environment, the key issue is that one wants a prefix that enables the entire [RFC1918] (Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, “Address Allocation for Private Internets,” February 1996.) address space to be embedded in a single /64 prefix, with the assumption that any routing structure behind the translator is managed by IPv4 routing.

1.6.2. Connection of IPv6-only islands to an IPv4-only network

TOC

To be completed

1.6.3. Connecting IPv4-only devices with IPv6-only devices

TOC

To be completed

1.6.4. ISP-supported connections between IPv4-only networks and IPv6-only networks

TOC

In this case (see Figure 7 (Service provider translation with multiple interchange points)) we presume that a service provider or equivalent is offering a service in a network in which IPv4 routing is not supported, but customers are allocated relatively large pools of general IPv6 addresses, suitable for clients of IPv4 or IPv6 hosts, and relatively small pools of addresses mapped to global IPv4 addresses that are intended to be accessible to IPv4 peers and clients through translation. Presumably, there are a number of such customers, and the administration wishes to use normal routing to manage the issues.
As a carrier offering, there is also a need for stateless translation, to accomplish two things: the ability to use multiple translators in parallel without having to maintain state among them, and to minimize the software overhead on the translator for systems that communicate regularly. There may also be stateful translation, the purpose of which is temporary connections between systems that do so only occasionally.

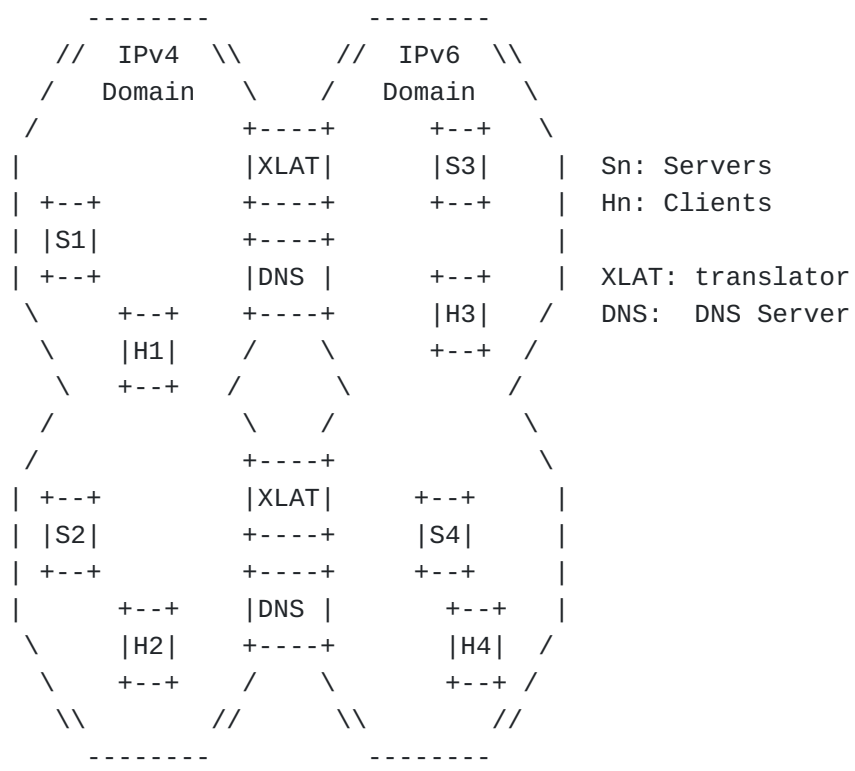

Figure 7: Service provider translation with multiple interchange points

---

Since [RFC4291] (Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” February 2006.) specifies that routable IPv6 prefixes are 64 bits or shorter apart from host routes, one wishes to allocate each customer a /64 mapped to a few IPv4 addresses and a shorter prefix for his general use. The customer's CPE advertises the two prefixes into the IPv6 routing domain to attract relevant traffic. The translator advertises the mapped equivalent of an IPv4 default route into the IPv6 domain to attract all other traffic to it, for translation into the IPv4 routing domain. It also advertises an appropriate IPv4 prefix aggregating the mapped prefixes into the IPv4 domain to attract traffic intended for these customers.
In this case, the LIR prefix MUST be within /32../63; a /64 puts the entire IPv4 address space into the host part, which is equivalent to the case in Section 1.6.1 (Connection of IPv4-only islands to an IPv6-only network), and a prefix shorter than /32 wastes space with no redeeming argument. In general, the LIR prefix should be 64 bits less the length of IPv4 prefixes it allocates to its IPv4-mapped customers. For example, if it is allocating a mapped IPv4 /24 to each customer, the LIR prefix used for mapping between IPv4 and IPv6 addresses should be a /40, and the least significant bits in the IPv4 address form the host part of the address.

Note that the significant difference between doing this between specific networks and between "the IPv4 Internet" and "the IPv6 Internet" is primarily in the Advertised IPv4 Prefix and the LIR Prefix. Between specific networks, or between a specific IPv6 network and the general IPv4 network, the translators and the DNS server are operated by the same operator, and as a result the IPv6 network is likely to use the same Advertised IPv4 Prefix and the same LIR prefix. Between general networks, they may have different operators or the same operator may have differing requirements. As result, they will use different Advertised IPv4 Prefixes and LIR prefixes. The algorithm, however, is the same.

---

2. Framework

TOC

Having laid out the preferred transition model and the options for implementing it (Section 1.1 (Why translation?)), defined terms (Section 1.2 (Terminology)), considered the requirements (Section 1.3 (Translation objectives)), considered the transition model (Section 1.4 (Transition Plan)), and considered the kinds of networks the facility would support (Section 1.6 (Expected uses of translation)), we now turn to a framework for IPv4/IPv6 translation. This framework has three main parts:

- *The recommended address format
- *The functional components of a translation solution, which include
  - -A DNS Translator,
  - -An optional stateless translator, or/and
  - -An optional stateful translator.
- *The operational characteristics of the solution.

2.1. Translation Scenario and Operation Mode

TOC

In this document, we only discuss the scenario:
Connecting an IPv6 network to the IPv4 Internet, including
(1) An IPv6 network to the IPv4 Internet.
(2) The IPv4 Internet to an IPv6 network.

2.1.1. Translation Model

TOC

The translation model is shown in the following figure.

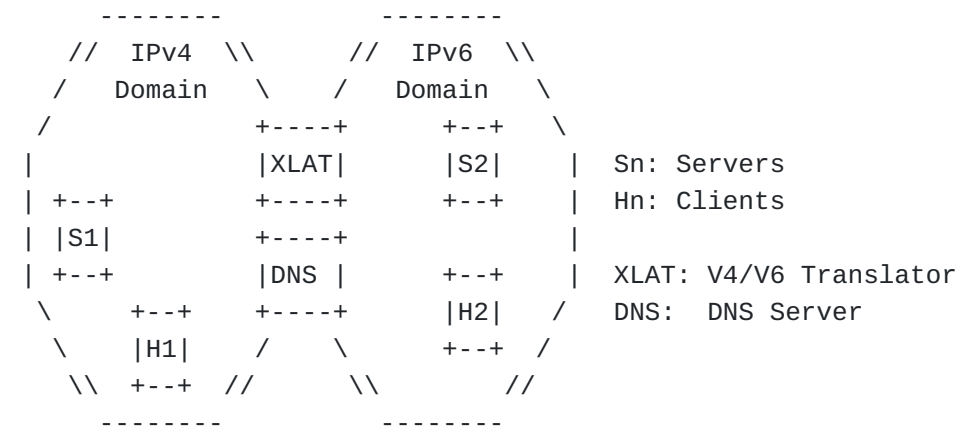

Figure 8: Translation Model

2.1.2. Operation Mode of the Translator

TOC

There are two translation modes: stateless translation and stateful translation. For the stateless translation, the translation information is carried in the address itself, permitting both IPv4->IPv6 and IPv6->IPv4 sessions establishment. For the 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.

In order to perform the required function, the translator needs to represent the IPv4 addresses in the IPv6 Internet and the IPv6 addresses in the IPv4 Internet.

For the representation of the IPv4 addresses in the IPv6 Internet, both stateless and stateful translation schemes use the same method, i.e. embedding the original IPv4 address in the IPv6 address.

For the representation of the IPv6 addresses in the IPv4 Internet, the stateless and stateful translation schemes use different methods.

(1) For the stateless translation, a subset of IPv6 address can be defined by embedding the original IPv4 address in the IPv6 address. The original IPv4 address will serve as the IPv6 representation in the IPv4 land.
(2) For the stateful translation, representing arbitrary IPv6 addresses in the IPv4 Internet requires some form of translation state that will define the mapping between the original IPv6 address and its representation in the IPv4 land.

2.2. Embedded Address Format

TOC

Embedding IPv4 address in IPv6 address (defined as IPv4-embedded IPv6 address) will be formed by concatenating a prefix to the IPv4 address and optionally a suffix. The prefix is called the PREFIX and the suffix is called SUFFIX. The resulting IPv6 representation is depicted in the figure below.

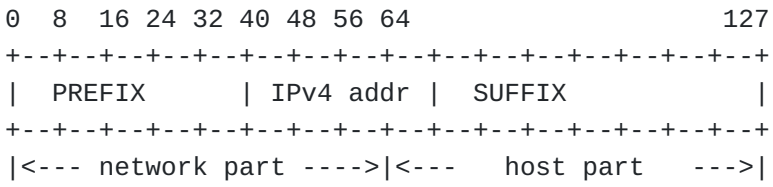

Figure 9: Embedded Address Format

For the representation of the IPv4 addresses in the IPv6 Internet in both stateless and stateful modes, the PREFIX is advertised in the IPv6 network by the translator, and packets addressed to this PRFIX will be routed to the translator. This PREFIX is configured for each translator, and DNS ALG.
For the representation of the IPv6 addresses in the IPv4 Internet in the stateless mode, more specifics (defined as IVI6) inside this PREFIX are advertised in the ISP's IPv6 network by the CPE routers, and packets addressed to the more specifcs inside this PREFIX will be routed to the IPv6 end systems. This PREFIX is not used for the representation of the IPv6 addresses in the IPv4 Internet in the stateless mode.
As shown in Figure 9 (Embedded Address Format), the embedded address format has three components:

bits 0..n-1 (PREFIX):
An LIR-specified prefix, either 32..63 bits long or 96 bits long,

bits n..n+31 An embedded IPv4 address. Except in the case of a 96 bit prefix, this address intentionally straddles the boundary between [RFC4291] (Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” February 2006.)'s 64 bit "subnet" locator and its 64 bit host identifier. The intention is that the /64 be used in routing and the bits in the host part be used for host identification as described in the address architecture.

bits n+32..127 (SUFFIX): Entirely zero; note that if n=96, this is null.

The selection of the PREFIX, the prefix length and SUFFIX is discussed in the following sections.

2.2.1. LIR prefix versus Well-Known prefix

TOC

2.2.1.1. Stateless mode

TOC

2.2.1.1.1. IPv6 Routing system scalability

TOC

In the stateless mode, the more specifics inside the IPv4-embedded IPv6 address block are used to represent the IPv6 end systems, therefore the LIR prefix should be used. The reason is that the LIR prefix can be aggregated in the ISP's border routers and will not affect the global IPv6 routing system. On the other hand, if the Well-Known prefix is used, the global IPv4 routing table will be inserted into the global IPv6 routing system, which is known to be a very bad idea.
In the stateless mode, it is possible to use LIR prefix to represent the IPv6 addresses in the IPv4 Internet and use Well-Known prefix to represent the IPv4 addresses in the IPv6 Internet. However, this is also a bad idea. The reason is that there will be two possible IPv6 addresses to represent a single IPv4 host, i.e. if the IPv4 address is used by a host in the IPv4 Internet, a Well-Known prefix should be used; if the IPv4 address (as IVI6) is used by a host in the IPv6 Internet, a LIR prefix should be used. The IVI6 hosts must know which one to use in order to communicate with that host. However, if the LIR

prefix is used in both representations, this problem is solved by the "more specific win" routing principle.
The potential leakage of the IPv6 more specifics introduced by using LIR prefix could be controlled by ISP's general routing practice, since this specific is the same compared with other more specifics inside ISP's autonomous system.

2.2.1.1.2. Referral support

TOC

For the referral support in the stateless mode, only the IVI6 hosts use LIR prefix to represent IPv4 addresses in IPv6 and the IVI6 hosts know the PREFIX, therefore, the IVI6 hosts could pass the original IPv4 addresses to the other hosts rather than mapped form and the referral support is the same as in the dual-stack case.

2.2.1.1.3. Native connectivity preference in communications involving dual stack nodes

TOC

In the stateless mode, the IVI6 hosts are IPv6 single-stack host, therefore, the native connectivity preference can be achieve automatically.

2.2.1.1.4. DNS ALG configuration

TOC

For the DNS ALG configuration in the stateless mode, the IVI6 hosts know the PREFIX, therefore, DNS ALG can be implemented in the end-system without additional information.

2.2.1.1.5. Support for multiple translators

TOC

Support for multiple translators in the stateless mode, either LIR or suffix can be used to identify different translators.

2.2.1.2. Stateful mode

TOC

2.2.1.2.1. IPv6 Routing system scalability

TOC

In the stateful mode, it is possible to either use LIR prefix or Well-Known prefix to represent the IPv4 addresses in the IPv6 Internet. If the LIR prefix is used, the protential leakge of the IPv6 more specifics may happen. This can be filtered at the ISP's border routers via manual configuration. If the Well-Known prefix is used, the configuration could be simplier since it is the unique Well-Known prefix.

2.2.1.2.2. Referral support

TOC

This section analyzes the impact of the prefix type selected for representing the IPv4 addresses in the IPv6 Internet in the referral operations.
A referral operation is when a host A passes the IP address of a Host B to a third Host C as application data. The host Host C will then initiate a communication towards the Host B using the IP address received. This is not a rare operation in some type of applications, such as VoIP or peer-to-peer applications.
All the scenarios where Host A and Host C are in different IP version, they require a specific ALG, since the IP address information contained as application data must be translated, in order to be meaningful a the receiver.
A general observation about these scenarios is that in the case a Well-Known prefix is used, it would be possible for the ALG to identify the IPv6 addresses containing an embedded IPv4 address and translate it, cause they could identify the Well-Known prefix and know that are not general use IPv6 addresses. If the PREFIX is a LIR prefix, it may be possible for the ALG to translate the address in the referral, as long as the translator is configured to know that this specific prefix is unused to map IPv4 addresses. So, a Well-Known prefix is more likely to work with referral in the case that ALG is needed than the LIR prefix.

2.2.1.2.3. Native connectivity preference in communications involving dual stack nodes

TOC

When dual stack nodes are involved in the communication, the potential issue is that they prefer translated connectivity over the native connectivity. There are multiple ways to try to deal with this issue.
Communication initiated from an IPv6-only node towards a dual stack node: In this case, the IPv6 only node will query for the FQDN of the

dual stack node. The DNS ALG function will try first to get the AAAA RR. Since there is one available, it will return it and no AAAA RR will be synthesized from the A RR of the dual stack node. However, it should be noted that the DNS64 must first try to get the real AAAA RR before starting the synthesis, if not, it may result in the aforementioned problem.
Communication initiated from a dual stack node toward an IPv4 only node: Nodes that have both IPv6 and IPv4 connectivity and are configured with an address for a DNS ALG as their resolving nameserver may receive responses containing synthetic AAAA resource records. If the node prefers IPv6 over IPv4, using the addresses in the synthetic AAAA RRs means that the node will attempt to communicate through the translator mechanism first, and only fall back to native IPv4 connectivity if connecting through translator fails (if the application tries the full set of destination addresses). In order to the node prefers native connectivity, we can configure the PREFIX in the RFC3484 policy table. If a Well-Known prefix is used, it can be configured in the default policy table. If we use a LIR prefix, we need a mean to properly configure the policy table, which is not currently available (only manual configuration is currently defined) (see [I-D.ietf-6man-addr-select-sol] for more on this topic).

2.2.1.2.4. DNS ALG configuration

TOC

The DNS ALG function can be placed either in the DNS server or in the end host. In order to synthesize AAAA RR, the DNS ALG function needs to know the PREFIX. In the case that a Well-Known prefix is used, the PREFIX information can be hardcoded in the DNS ALG code, requiring no additional tools for learning it. In the case that a LIR prefix is used, the DNS ALG needs to discover the PREFIX information. In the case that the DNS ALG is located in the servers, it may be a viable option to manually configure the PREFIX in the DNS ALG for a few servers. However, in the case the the DNS ALG is located in the hosts, the manual option seems inconvenient and alternative automatic means need to be provisioned. Moreover, since this information is used for DNSSEC operations, the mechanism to configure the PREFIX need to be secure. The result is that the LIR prefix option requires more tools than the Well-Known prefix.

2.2.1.2.5. Support for multiple translators

TOC

This issue is somehow orthogonal on whether the prefix is Well-Known or LIR. In both cases, it is possible to use a single prefix for multiple translators or different prefixes for different translators. In any

case, this would be achieved by inserting (or not) some subnet bits between the prefix and the embedded IPv4 address that would be used to identify the translator box. This issue does have implications on some of the different issues considered before. In particular, if a per translator prefix is used, then there is the need to configure the prefix in the DNS ALG, so the non configuration feature of the Well-Known prefix is no longer achieved.

2.2.2. Prefix length

TOC

One issue that is worth considering is the one related to IPv6 address consumption. In particular, depending on the selected prefix length, IPv6 address consumption can become an issue. If we consider the case of the Well-Know prefix, the prefix would be allocated by IANA for this particular purpose. As such, it seems reasonable that a short prefix can be obtained for this. Requesting for a /24 or even a few bits shorter seems feasible. The potential benefit of this is that IPv4 prefixes can be represented as IPv6 prefixes that are shorter than 64 bits. This would result in routing based on the upper 64 bits, which is compatible with current IPv6 practices. For instance, if we use a /24 for the Well-Know prefix, an IPv4 /24 would result in an IPv6 /48, which seems somehow equivalent from the routing perspective.
On the other hand, if we go for the LIR prefix option, then the prefix must come out of the IPv6 allocation for the site running the translator. If the site running the translator is an ISP, then probably the allocation of the ISP is a /32 or shorter, so, it may be possible for the ISP to allocate a somehow short prefix for this, maybe a /40. However, if the translator is run by an end site, which normal allocation is a /48, then the LIR prefix for the translator should be much longer than that, possibly a /56. So, in the case the site needs to route based on the IPv4 prefix embedded in the IPv6 address (e.g. in order to access to different parts of the IPv4 space through different routes), then it is likely that it will need to route on the lower 64 bits of the IPv6 address.
According to current specifications, routers must handle routes containing prefixes of any valid length, from 0 to 128. However, some users have reported that routers exhibit worse performance when routing using long prefixes, in particular when using prefixes longer than 80 bits. This implies that using prefixes shorter than that would result in better performance in some cases.

TOC

2.2.3. Suffix

In the current implementation of the stateless mode, the suffix is entirely zero. For the stateful mode when using Well-Known prefix, the suffix can be used to represent different NAT boxes.

2.2.4. Recommdations

TOC

For the PREFIX selection, we recommand to use LIR prefix. For the stateful translator, the Well-Known prefix can be used.
For the prefix length selection, there are some obvious values that might be popular, including /40, /44, and /96, but there is no requirement than any of them be used; this is left to the operator's discretion.
For the SUFFIX selection, it is is entirely zero at this time. However, it could be used for the future extension of the translation functions.

2.3. Translation components

TOC

As noted in Section 1.6 (Expected uses of translation),translation involves several components. An IPv4 client or peer must be able to determine the address of its server by obtaining an A record from DNS even if the server is IPv6-only - only has an IPv6 stack, or is in an IPv6-only network. Similarly, an IPv6 client or peer must be able to determine the address of its server by obtaining an AAAA record from DNS even if the server is IPv4-only - only has an IPv4 stack, or is in an IPv4-only network. Given the address, the client/peer must be able to initiate a connection to the server/peer, and the server/peer must be able to reply. It would be very nice if this scaled to the size of regional networks with straightforward operational practice.
To that end, we describe four subsystems:

*A Domain Name System Translator

*A stateless IPv4/IPv6 translator

*A stateful IPv4/IPv6 translator

*Translators for some applications

TOC

2.3.1. DNS Translator

[I-D.bagnulo-behave-dns64] (Bagnulo, M., Sullivan, A., Matthews, P., Beijnum, I., and M. Endo, “DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers,” March 2009.) describes the mechanisms by which a DNS Translator is intended to operate. It is designed to operate on the basis of known but fixed state: the resource records, and therefore the names and addresses, that it translates are known to network elements outside of the data plane translator, but the process of serving them to applications does not interact with the data plane translator in any way.
There are at least three possible implementations of a DNS Translator:

Static records: One could literally program DNS with corresponding A and AAAA records. This is most appropriate for stub services such as access to a legacy printer pool.

Dynamic Translation of static records: In more general operation, the expected behavior is for the application to request both A and AAAA records, and for an A record to be (retrieved and) translated by the DNS translator if and only if no reachable AAAA record exists. This has ephemeral issues with cached translations, which can be dealt with by caching only the source record and forcing it to be translated whenever accessed.

Static or Dynamic Translation of Dynamic DNS records: In Dynamic DNS usage, a system could potentially report the translation of a name using a Mapped IPv4 Address, or using both a Mapped IPv4 Address and some other address. The DNS translator has several options; it could store a AAAA record for the Mapped IPv4 Address and depend on translation of that for A records inline, it could store both an A and a AAAA record, or (when there is another IPv6 address as well which is stored as the AAAA record) it could store only the A record.

2.3.2. Stateless Translation - IPv4-embedded IPv6 addresses

TOC

[I-D.baker-behave-v4v6-translation] (Baker, F., “IP/ICMP Translation Algorithm,” February 2009.) describes and defines the behavior of a stateless translator. This is an optional facility; one could implement or deploy only the stateful mode described in Section 2.3.3 (Stateful translation - IPv4-related IPv6 address), at the cost of being able to have systems with IPv6 addresses that are not embedded to IPv4 addresses access IPv4 servers and peers. Stateless translation enables IPv4-only clients and peers to initiate connections to IPv6-only servers or peers equipped with IPv4-embedded IPv6 addresses, as

described in Figure 7 (Service provider translation with multiple interchange points). It also enables scalable coordination of IPv4-only stub networks or ISP IPv6-only networks as described in Figure 6 (Customer dual stack network).
In addition, since [RFC3484] (Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)," February 2003.)address selection would select a IPv4-embedded IPv6 address when it is available, stateless translation enables IPv6 clients and peers with Mapped IPv4 Addresses to open connections with IPv4 servers and peers in a scalable fashion, supporting asynchronous routes.

2.3.3. Stateful translation - IPv4-related IPv6 address

TOC

[I-D.baker-behave-v4v6-translation] (Baker, F., "IP/ICMP Translation Algorithm," February 2009.) also describes and defines the behavior of the data plane component of a stateful translator.
[I-D.bagnulo-behave-nat64] (Bagnulo, M., Matthews, P., and I. Beijnum, "NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers," March 2009.) describes the management of the state tables necessitated by stateful translation. Like stateful translation, this is an optional facility; one could implement or deploy only the stateful mode described in Section 2.3.2 (Stateless Translation - IPv4-embedded IPv6 addresses), at the cost of IPv4 access to IPv6-only servers and peers, the ability to use multiple translators interchangeably, and some level of scalability. Stateful translation is defined to enable IPv6 clients and peers without Mapped IPv4 Addresses to connect to IPv4-only servers and peers.
Stateful translation could be defined to enable IPv4 clients and peers to connect to IPv6-only servers and peers without Mapped IPv4 Addresses. This is far more complex, however, and is out of scope in the present work.

2.3.4. Translation gateway technologies

TOC

In addition, some applications require special support. An example is FTP. FTP's active mode doesn't work well across NATs without extra support such as SOCKS. Across NATs, it generally uses passive mode. However, the designers of FTP inexplicably wrote different and incompatible passive mode implementations for IPv4 and IPv6 networks. Hence, either they need to fix FTP, or a translator must be written for the application. Other applications may be similarly broken.
As a general rule, a simple operational recommendation will work around many application issues, which is that there should be a server in each domain or an instance of the server should have an interface in each

domain. For example, an SMTP MTA may be confused by finding an IPv6 address in its EHLO when it is connected to using IPv4 (or vice versa), but would perfectly well if it had an interface in both the IPv4 and IPv6 domains and was used as an application layer bridge between them.

2.4. Translation in operation

TOC

Operationally, there are two ways that translation could be used - as a permanent solution making transition "the other guy's problem", and as a temporary solution for a new part of one's network while bringing up IPv6 services in the remaining parts of one's network. The delay could, for example, be caused by contract cycles that prevent IPv6 deployment during the life of the contract. We obviously recommend the latter. For the IPv4 parts of the network, [RFC4213] (Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers," October 2005.)'s recommendation holds: bringing IPv6 up in those domains, moving production to it, and then taking down the now-unnecessary IPv4 service when economics warrant remains the least risk approach to transition.

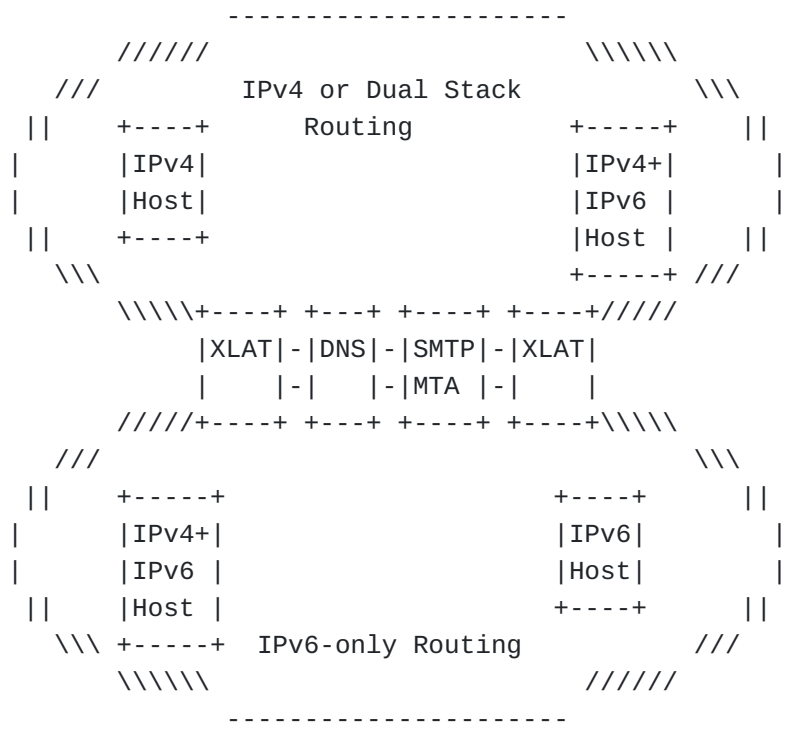

Figure 10: Translation Operational Model

During the coexistence phase, as shown in Figure 10 (Translation Operational Model), one expects a combination of hosts - IPv6-only gaming devices and handsets, older computer operating systems that are IPv4-only, and modern mainline operating systems that support both. One also expects a combination of networks - dual stack devices operating in single stack networks are effectively single stack, whether that stack is IPv4 or IPv6, as the other isn't providing communications services.

2.4.1. Impact Outside the Network Layer

TOC

The potential existence of IP/ICMP translators is already taken care of from a protocol perspective in [RFC2460] (Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification,” December 1998.). However, an IPv6 node that wants to be able to use translators needs some additional logic in the network layer.
The network layer in an IPv6-only node, when presented by the application with either an IPv4 destination address or an IPv4-mapped IPv6 destination address, is likely to drop the packet and return some error message to the application. In order to take advantage of translators such a node should instead send an IPv6 packet where the destination address is the IPv4-mapped address and the source address is the node's temporarily assigned IPv4-translated address. If the node does not have a temporarily assigned IPv4-translated address it should acquire one using mechanisms that are not discussed in this document. Note that the above also applies to a dual IPv4/IPv6 implementation node which is not configured with any IPv4 address.
There are no extra changes needed to applications to operate through a translator beyond what applications already need to do to operate on a dual node. The applications that have been modified to work on a dual node already have the mechanisms to determine whether they are communicating with an IPv4 or an IPv6 peer. Thus if the applications need to modify their behavior depending on the type of the peer, such as ftp determining whether to fall back to using the PORT/PASV command when EPRT/EPSV fails (as specified in [RFC2428] (Allman, M., Ostermann, S., and C. Metz, “FTP Extensions for IPv6 and NATs,” September 1998.)), they already need to do that when running on dual nodes and the presence of translators does not add anything. For example, when using the socket API [RFC3493] (Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, “Basic Socket Interface Extensions for IPv6,” February 2003.) the applications know that the peer is IPv6 if they get an AF_INET6 address from the name service and the address is not an IPv4-mapped address (i.e., IN6_IS_ADDR_V4MAPPED returns false). If this

is not the case, i.e., the address is AF_INET or an IPv4-mapped IPv6 address, the peer is IPv4.
One way of viewing the translator, which might help clarify why applications do not need to know that a translator is used, is to look at the information that is passed from the transport layer to the network layer. If the transport passes down an IPv4 address (whether or not is in the IPv4-mapped encoding) this means that at some point there will be IPv4 packets generated. In a dual node the generation of the IPv4 packets takes place in the sending node. In an IPv6-only node conceptually the only difference is that the IPv4 packet is generated by the translator - all the information that the transport layer passed to the network layer will be conveyed to the translator in some form. That form just "happens" to be in the form of an IPv6 header.

2.5. Unsolved problems

TOC

Just say "multicast"; this framework could support multicast, but at this point does not. This is a place for future work.
As noted, IPv4 client/peer access to IPv6 servers and peers lacking Mapped IPv4 Addresses is not solved.
Interoperation between IPv4-only clients and IPv6-only clients is not supported, and is not believed to be needed.

---

3. IANA Considerations

TOC

This memo requires no parameter assignment by the IANA.
Note to RFC Editor: This section will have served its purpose if it correctly tells IANA that no new assignments or registries are required, or if those assignments or registries are created during the RFC publication process. From the author's perspective, it may therefore be removed upon publication as an RFC at the RFC Editor's discretion.

---

4. Security Considerations

TOC

One "security" issue has been raised, with an address format that was considered and rejected for that reason. At this point, the editor knows of no other security issues raised by the address format that are not already applicable to the addressing architecture in general.

---

5. Acknowledgements

TOC

This is under development by a large group of people. Those who have posted to the list during the discussion include Andrew Sullivan, Andrew Yourtchenko, Brian Carpenter, Dan Wing, Ed Jankiewicz, Fred Baker, Hiroshi Miyata, Iljitsch van Beijnum, John Schnizlein, Kevin Yin, Magnus Westerlund, Marcelo Bagnulo Braun, Margaret Wasserman, Masahito Endo, Phil Roberts, Philip Matthews, Remi Denis-Courmont, Remi Despres, and Xing Li.
The appendix is largely derived from Hiroshi Miyata's analysis, which is in turn based on documents by many of those just named.
Ed Jankiewicz described the transition plan.
The definition of a "Local Internet Registry" came from the Wikipedia, and was slightly expanded to cover the present case. (EDITOR'S QUESTION: Would it be better to describe this as an "operator-defined prefix"?)

---

6. References

TOC

6.1. Normative References

TOC

[I-D.bagnulo-behave-dns64]	Bagnulo, M., Sullivan, A., Matthews, P., Beijnum, I., and M. Endo, “DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers,” draft-bagnulo-behave-dns64-02 (work in progress), March 2009 (TXT).
[I-D.bagnulo-behave-nat64]	Bagnulo, M., Matthews, P., and I. Beijnum, “NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers,” draft-bagnulo-behave-nat64-03 (work in progress), March 2009 (TXT).
[I-D.baker-behave-v4v6-translation]	Baker, F., “IP/ICMP Translation Algorithm,” draft-baker-behave-v4v6-translation-02 (work in progress), February 2009 (TXT).
[RFC2119]	Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2460]	Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification,” RFC 2460, December 1998 (TXT, HTML, XML).
[RFC4291]	Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” RFC 4291, February 2006 (TXT).

6.2. Informative References

TOC

[I-D.baker-behave-ivi]	Li, X., Bao, C., Baker, F., and K. Yin, "IVI Update to SIIT and NAT-PT," draft-baker-behave-ivi-01 (work in progress), September 2008 (TXT).
[I-D.durand-softwire-dual-stack-lite]	Durand, A., Droms, R., Haberman, B., and J. Woodyatt, "Dual-stack lite broadband deployments post IPv4 exhaustion," draft-durand-softwire-dual-stack-lite-01 (work in progress), November 2008 (TXT).
[I-D.ietf-v6ops-addcon]	Velde, G., Popoviciu, C., Chown, T., Bonness, O., and C. Hahn, "IPv6 Unicast Address Assignment Considerations," draft-ietf-v6ops-addcon-10 (work in progress), September 2008 (TXT).
[I-D.miyata-v6ops-snatpt]	Miyata, H. and M. Endo, "sNAT-PT: Simplified Network Address Translation - Protocol Translation," draft-miyata-v6ops-snatpt-02 (work in progress), September 2008 (TXT).
[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," draft-xli-behave-ivi-07 (work in progress), January 2010 (TXT).
[RFC1918]	Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets," BCP 5, RFC 1918, February 1996 (TXT).
[RFC2428]	Allman, M., Ostermann, S., and C. Metz, "FTP Extensions for IPv6 and NATs," RFC 2428, September 1998 (TXT, HTML, XML).
[RFC2765]	Nordmark, E., "Stateless IP/ICMP Translation Algorithm (SIIT)," RFC 2765, February 2000 (TXT).
[RFC2766]	Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)," RFC 2766, February 2000 (TXT).
[RFC3056]	Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds," RFC 3056, February 2001 (TXT).
[RFC3142]	Hagino, J. and K. Yamamoto, "An IPv6-to-IPv4 Transport Relay Translator," RFC 3142, June 2001 (TXT).
[RFC3484]	Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)," RFC 3484, February 2003 (TXT).
[RFC3493]	

	Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, “Basic Socket Interface Extensions for IPv6,” RFC 3493, February 2003 (TXT).
[RFC3879]	Huitema, C. and B. Carpenter, “Deprecating Site Local Addresses,” RFC 3879, September 2004 (TXT).
[RFC4192]	Baker, F., Lear, E., and R. Droms, “Procedures for Renumbering an IPv6 Network without a Flag Day,” RFC 4192, September 2005 (TXT).
[RFC4193]	Hinden, R. and B. Haberman, “Unique Local IPv6 Unicast Addresses,” RFC 4193, October 2005 (TXT).
[RFC4213]	Nordmark, E. and R. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers,” RFC 4213, October 2005 (TXT).
[RFC4380]	Huitema, C., “Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs),” RFC 4380, February 2006 (TXT).
[RFC4862]	Thomson, S., Narten, T., and T. Jinmei, “IPv6 Stateless Address Autoconfiguration,” RFC 4862, September 2007 (TXT).
[RFC4864]	Van de Velde, G., Hain, T., Droms, R., Carpenter, B., and E. Klein, “Local Network Protection for IPv6,” RFC 4864, May 2007 (TXT).
[RFC4941]	Narten, T., Draves, R., and S. Krishnan, “Privacy Extensions for Stateless Address Autoconfiguration in IPv6,” RFC 4941, September 2007 (TXT).
[RFC4966]	Aoun, C. and E. Davies, “Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status,” RFC 4966, July 2007 (TXT).
[RFC5211]	Curran, J., “An Internet Transition Plan,” RFC 5211, July 2008 (TXT).
[RFC5214]	Templin, F., Gleeson, T., and D. Thaler, “Intra-Site Automatic Tunnel Addressing Protocol (ISATAP),” RFC 5214, March 2008 (TXT).

Authors' Addresses

TOC

	Fred Baker (editor)
	Cisco Systems
	Santa Barbara, California 93117
	USA
Phone:	+1-408-526-4257
Fax:	+1-413-473-2403
Email:	fred@cisco.com
	Xing Li (editor)
	CERNET Center/Tsinghua University

	Room 225, Main Building, Tsinghua University
	Beijing, 100084
	China
Phone:	+86 62785983
Email:	xing@cernet.edu.cn
	Congxiao Bao (editor)
	CERNET Center/Tsinghua University
	Room 225, Main Building, Tsinghua University
	Beijing, 100084
	China
Phone:	+86 62785983
Email:	congxiao@cernet.edu.cn