Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 23, 2008

M. Bagnulo
Huawei Labs at UC3M
F. Baker
Cisco Systems
February 20, 2008

IPv4/IPv6 Coexistence and Transition: Requirements for solutions
draft-bagnulo-v6ops-6man-nat64-pb-statement-01

Status of this Memo

Copyright Notice

Abstract

This note presents the problem statement, analysis and requirements for solutions to IPv4/IPv6 coexistence and eventual transition in a scenario in which dual stack operation is not the norm.

Comment [DT1]: I think this is not a sufficient scenario to match the title, which is more general. The other scenario is where you have dual stack operation, but the accepting-side (server-side) of communication has no public IPv4 address.

Table of Contents

1. Introduction

This note addresses requirements for solutions to IPv4/IPv6 coexistence and eventual transition in a scenario in which dual stack operation is not the norm.

2. Problem statement

Operationally, we now expect the transition to be less a matter of connecting ever-growing IPv6 islands in an IPv4 network, and more a matter of the network becoming a patchwork quilt of IPv4, IPv6, and dual domains.

- Hosts now generally support IPv6 and IPv4 natively.
- As described in [1], the IETF community had expected administrations to turn on IPv6 in their existing IPv4 networks, resulting in a simple coexistence scenario.
- Increasingly, we hear statements that people want to move directly to an IPv6-only or IPv6-dominant network.

In this context, "IPv6-only" refers to a network or system that only runs IPv6, and "IPv6-dominant" refers to a network or system that may use IPv4 internally or with other clients, but in the context only routes IPv6 datagrams. "IPv4-only" and "IPv4-dominant" are defined similarly. Since these are indistinguishable to the peer, the terms "IPv4-only" and "IPv6-only" will be used in this paper and considered to subsume the "dominant" issues.

2.1. Transition scenarios

There are six obvious transition scenarios:

- IPv4 system connecting to an IPv4 system across an IPv4 network,
- An IPv6 system connecting to an IPv6 system across an IPv6 network,
- an IPv4 system connecting to an IPv4 system across an IPv6 network,
- an IPv6 system connecting to an IPv6 system across an IPv4 network,
- an IPv4 system connecting to an IPv6 system, or
- an IPv6 system connecting to an IPv4 system.

2.1.1. Simple transition scenarios

The simplest coexistence cases are about an IPv4 system connecting to an IPv4 system across an IPv4 network, or an IPv6 system connecting to an IPv6 system across an IPv6 network. The dual stack case, in which both endpoints and the relevant applications support IPv4 and IPv6 and the network supports at least one of the protocols, falls

Comment [DT2]: Same comment as in abstract. (Also it's generally bad form to repeat the same text in both places.)

Comment [DT3]: The term "dual domain" isn't defined… Would you classify a "public IPv6 / private IPv4 NAT'ed" domain as IPv6 or dual or something else? From the terminology below, I think it would be "IPv6-dominant" right, but I cannot tell? Suggest rewording to not use "dual domain" before it's defined. Maybe insert a Terminology section before here, with the terms in the paragraph below defined there.

Comment [DT4]: Can you be more specific. Enterprises? ISPs? Home networks?

Comment [DT5]: If it NATs IPv4 datagrams (to public IPv4) does that mean it routes them or not?

Comment [DT6]: I disagree. The list below assumes the entire network is either IPv4, or it's IPv6. It's complexly missing the cases where the network contains IPv4-only/dominant and IPv6-only/dominant parts. For example, Figure 1 fits into none of these 6 scenarios cleanly.

Comment [DT7]: This term is confusing. Is this referring to an IPv4-only device? Or is it referring to an IPv4-capable device? Or is it really referring to an IPv4 endpoint (socket) on a device?

into this case as the applications can connect using whichever stack is consistent end--to--end.

The IETF strongly prefers and recommends this scenario, as the operational matters are the simplest. Until the Internet reaches IPv4 address exhaustion, an IPv4 and an IPv6 address can be assigned to every interface, and the applications are supported. When it becomes necessary to deploy only IPv6 addresses, since all other systems have both, IPv6-only systems cleanly interoperate with existing systems.

Comment [DT8]: This sentence is wrong for multiple reasons. First, it assumes that an IPv4 address == a PUBLIC IPv4 address. Second, the term "exhaustion" and "can be assigned" are ones many people quibble with on the grounds that it just gets more expensive, not that there's hard "exhaustion" necessarily. So "can be assigned" yes, if you want to pay enough money, but that doesn't mean you can support apps for people who don't have that kind of money.

2.1.2. Transition scenarios that do not require translation

[1] discusses the scenario in Figure 1, in which routers connect two dual domains via an IPv4-only domain. Obviously, this can be reversed: routers can connect two dual domains via an IPv6-only domain. Note that the connecting domain need not actually be IPv4-only or IPv6-only; to create this scenario, it need merely fail to offer IPv6 or IPv4 services to the neighboring domains.

Comment [DT9]: Is this synonymous with saying that it could be IPv4-dominant or IPv6-dominant?

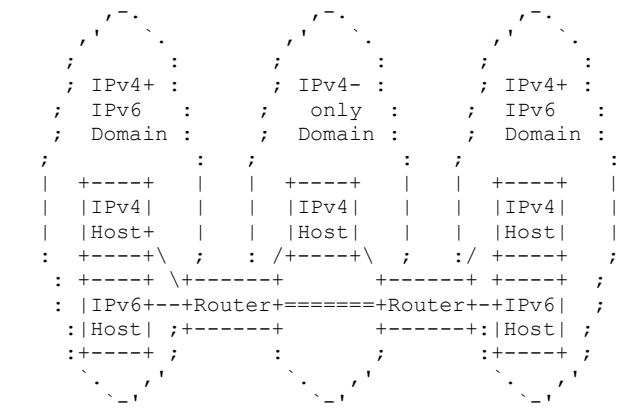

Figure 1: Disconnected continuity

In such a scenario, there are two obvious solutions: one can tunnel across the connecting domain, as shown, or one can translate between IP layers using something akin to traditional NAT technology. The tunnel approach offers some pros and some cons: it natively connects the dual domains, meaning that all applications should work, but they may have issues with the path MTU, and the tunnels require some form of configuration. The NAT approach similarly offers pros and cons: it offers something similar to standard routing, but it suffers from

Comment [DT10]: Does 6to4 require some form of configuration (it autoconfigures itself on Windows and on some home gateways, so if you count auto-configuration then sure but I'd consider it zero-configuration)? If you consider autoconfiguration as "some form of configuration" why is that a con?

the various ills of Network Address Translation on both sides, meaning that it may be difficult for the dual domains to offer services to each other.

In general, the IETF recommends the use of tunnels rather than a dual NAT.

There are at least three generic models that could be used to describe this kind of tunneling scenario:

- Static tunnels with interior dynamic routing
- Start-time negotiated tunnels to some central point with default routing (example in [9])
- Dynamic tunnels with specific routing to islands (examples might include ISATAP [5] or a tunnel broker of some description)

Static tunnels with routing through them are commonly deployed today, both in VPNs and in overlay networks. The positive side is that they provide simple service; the negative is that the generally require manual configuration and can result in suboptimal routing.

A "start-time" tunnel might be useful in an access network that serves homes or SOHO environments. In this model, the ISP informs the CPE of a cross-network peer that it can create a tunnel to, reducing the case to one similar to static tunneling but without manual configuration.

A dynamic tunneling environment is an overlay model in which systems create tunnels to various peers across the connecting domain as needed, based on a priori knowledge of the correlation between remote prefixes and next hop routers. This has not been adequately described at this point, and therefore involves complexities in implementation and deployment.

2.1.3. Transition scenarios that require translation

Translation, as found in Figure 2, is considered in NAT-PT [6], which has in turn been set aside via [8]. In essence, translation is required when an IPv4-only system connects to an IPv6-only system or an IPv6-only system connects to an IPv4-only system. These systems need not actually be IPv4-only or IPv6-only; if the connecting network is IPv4-only or IPv6-only and provides no tunnel, but only offers IPv4 service to one and only offers IPv6 service to the other, the situation is equivalent.

Comment [DT11]: Reference RFC 2993?

Comment [DT12]: For figure 1, 6to4 is a far better example than ISATAP (ISATAP is Intra-Site, which is what the IS stands for, and connects hosts not islands). 6to4 is meant for this scenario.

Comment [DT13]: Huh? 6to4, Teredo, and to some extent ISATAP (subject to the point I made above) all fit into this category. What is "not adequately described" about them?

Comment [DT14]: … for GENERAL PURPOSE use.

Comment [DT15]: In Figure 2, which network is "the connecting network"?

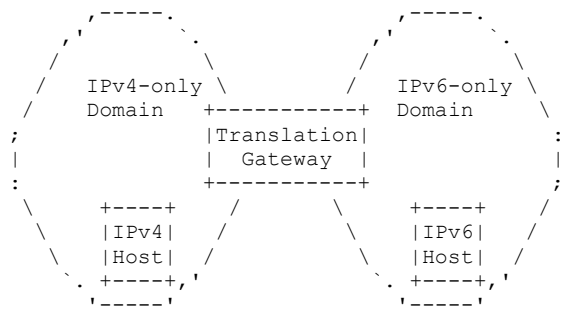

Figure 2: Translation

In such a scenario, it is necessary for the network to create a translation gateway, at which datagrams from one system are translated and then forwarded to the other. The situation is in many ways reflexive, since most Internet sessions are bidirectional - TCP between an IPv4 and an IPv6 system translate data messages in one direction and acknowledgments in the other.

They are not reflexive, however, in the distribution of domain names. If the application is client-server and the server is in one of the domains, the name of the server need only be propagated to the other. Reverse lookups, frequently used in spam verification would require the client's name to be propagated into the server's domain. But in this there are issues. The address of the client (the TCP peer) as seen by the server is not the remote system in the other domain; it is the translator. This is readily worked around for an IPv6 server, as the IPv4 address of the remote peer can be embedded in a "privacy" address [7], making the reverse lookup viable. This doesn't work on the IPv4 side, however.

Comment [DT16]: This doesn't make sense. A Privacy address has random bits, not an embedded v4 address. If it has a v4 address embedded, it's not "private".

2.2. Requirements for the overall transition strategy

Given the problem statement presented here, we see the following requirements for a complete transition strategy:

1. Any transition strategy must contemplate a period of coexistence, with ultimate transition (e.g., turning off IPv4) being a business decision.
2. Many are delaying turning on IPv6 (initiating coexistence in their networks) as long as possible.
3. Some are turning off IPv4 immediately, at least as a customer service.

4. Therefore, dual stack approaches, tunneled architectures, and translation architectures are all on the table.
5. Any solution that makes translation between semi-connected islands "normal" has failed the fundamental architecture of the Internet and can expect service complexity to be an issue. [3]
6. Translation architectures must provide for the advertisement of IPv4 names to IPv6 systems and vice versa. The address advertised in the "far" domain must be that of the translating gateway.
7. Tunneling architectures must provide a way to minimize and ideally eliminate configuration of the tunnel.

3. Preliminary analysis for translation mechanisms

3.1. Application behavior taxonomy

The general purpose of NAT64 ~~type of~~ mechanisms is to enable communication between a v4-only node and a v6-only node. However, there is a wide range of type of communications, when considering how they handle IP addresses. So, in order to properly characterize the problem, we need to do an analysis of ~~the~~ different application behavior in terms of ~~the~~ usage of ~~their~~ IP addresses. We will next present a taxonomy of the behavior of the application with respect ~~of~~to how they use the IP address. The support of the different types of behavior will impose a different set of constraints to the design of ~~a~~ NAT64 mechanisms. It is then important to decide which types of application behavior will be supported before starting to design a NAT64 mechanism. The proposed taxonomy is heavily based on the one presented in section 1.1 of draft-ietf-shim6-app-refer-00.txt.

The proposed application behavior taxonomy is the following:

Short-lived local handle. The IP address~~es~~ is never retained by the application. The only usage is for the application to pass it from the ~~DNS~~ name resolution APIs (e.g., getaddrinfo()) and the API to the protocol stack (e.g., connect() or sendto()). This type of communication can be either initiated by the v4-only node or by the v6-only node, resulting in two types of behaviors:~~,~~ v4-initiated short-lived local handle and v6-initiated short-lived local handle.

Long-lived application associations. The IP address is retained by the application for several instances of communication. However, it is always the same node that initiates the communication. This type of communication can be either initiated by the v4-only node or by the v6-only node, resulting in two type of behaviors:~~,~~ v4-initiated long-lived associations and v6-initiated long-lived associations.

Comment [DT17]: What does this mean?

Comment [DT18]: Eh? What's an "IPv4 name"? Do you mean "IPv4 addresses"?

Comment [DT19]: Undefined term. Or is this sentence supposed to be a definition? If so, reword as such.

Comment [DT20]: Getaddrinfo isn't limited to use for DNS. It also works with other protocols such as the Hosts file, etc.

Comment [DT21]: This taxonomy seems to be problematic in a sense. The class above in part, and this one, are one axis with respect to how long the association lasts. A separate axis is where the address comes from. Currently the taxonomy convolutes the two axes in a confusing way. For example the first class above implies a mechanism to learn (name resolution APIs), plus a lifetime (short), whereas this class only has a lifetime but no learning mechanism.

Callbacks. The application at one end retrieves the IP address of the peer and uses that to later communicate "back" to the peer. This type of communication can be either initiated by the v4-only node or by the v6-only node, resulting in two type of behaviors, v4-initiated callback, meaning that the initial communication is initiated by the v6-only node, and later the v4-only node initiates the callback, and v6-initiated callback, meaning that the initial communication is initiated by the v4-only node, and later the v6-only node initiates the callback. An additional disticntion can be made based on the time-frame of the call back operation. There can be short-lived call-backs, where the receiver immediately calls back to the initiator and long-lived call-backs where the receiver calls backs after a while.

Referrals. In an application with more than two parties, party B takes the IP address of party A and passes that to party C. After this party C uses the IP address to communicate with A. In this type of communication, the following 6 sub-cases are possible.

- o A and B are v6-only nodes and C is a v4-only node,
- o A and C are v6-only nodes and B is a v4-only node,
- o B and C are v6-only nodes and A is a v4-only node,
- o A and B are v4-only nodes and C is a v6-only node,
- o A and C are v4-only nodes and B is a v6-only node,
- o B and C are v4-only nodes and A is a v6-only node.

"Identity" comparison. Some applications might retain the IP address, not as a means to initiate communication as in the above cases, but as a means to compare whether a peer is the same as another peer. While this is insecure in general, it might be something which is used e.g., when TLS is used. This type of communication results in two sub-cases, when the v4-only node performs comparison of the v6-only node identity, and when the v6-only node performs comparison of the v4-only node identity.

Discussion: is there another type of application that embed IP addresses in the application data that doesn't fit in the previous cases?

3.2. Placement of the NAT64 mechanisms

Another aspect that is critical to design a NAT64 mechanism is the placement of the mechanisms involved. In other words, what elements can be modified/updated to support the NAT64 mechanisms. We assume that the NAT64 box supports a set of mechanisms that are the core part of the solution, but some approaches may require the modification of additional elements. In particular, we can identify the following additional elements that may require modification to support a NAT64 approach.

Comment [DT22]: From where? From the sockets API? (as noted earlier this doesn't work as it gives the address of the translator not the peer) From the peer using app-layer messages?

Comment [DT23]: Be consistent… sometimes the doc uses "callbacks", sometimes "call-backs".

Comment [DT24]: I'm not the expert on TLS, but I thought it just provided confidence in the name, not the IP address. A better example would be IPsec.

Comment [DT25]: Absolutely:
The case where you get the peer's address from the human using the app.

Modifications to v4-only nodes: one option is to require modifications to existent v4-only nodes in order to support the NAT64 mechanism. This option would impose high deployment costs, because the existent base of v4-only nodes is really ~~big~~ large and there ~~is~~ are no incentives for the v4-only nodes to install such mechanism, since it seems unlikely that v4-only nodes will have a strong need to communicate with v6-only nodes (at least at the initial stages of v6 deployment). However, it may be possible that this is the only viable solution for supporting some types of application behavior.

Modifications to v6-only nodes: Another option is to require modifications to v6-only nodes. This option seems much more acceptable, since the existent base of v6-nodes is relatively small and there would be a strong incentive for v6-only nodes to communicate with v4-only nodes, since most of the contents are available only in v4 today. However, imposing modifications to v6-only nodes does make deployment of the solution more difficult, since update of current v6-implementations is needed. In addition, there is an architectural consideration, that we would be imposing v6-only nodes to support "NAT hacks" in order to enable communication with the v4 world, and that those modifications may stay forever, even when the need for communication with the v4-Internet is not so pressing.

Modifications to both v4-only nodes and v6-only nodes. Another option is to require updates to both v4-only nodes and also to v6-only nodes. Needless to say that this would be the option with higher deployment costs.

No modifications. Another option is that the NAT64 mechanism~~s~~ does not require modifications to any host and that the mechanism is fully contained in the NAT64 box. This was the case of the previously defined NAT-PT approach. However, it may be challenging to design a solution with this constraint that does not suffer the limitations suffered by the NAT-PT mechanism that lead the IETF community to deprecate it.

Comment [DT26]: … for general use.

Another consideration related to the modifications imposed by a NAT64 approach is about what elements in the nodes need to be updated. In particular, it is important to determine if only the IP layer on the affected nodes needs to be modified or if other elements in the nodes need~~s~~ to be updated. In particular, it is critical to determine if applications need to be modified in order to support the NAT64 mechanism.

3.3. v4 addressing consideration

We assume that both the v6-only nodes and the v6 interface of the NAT64 boxes will have routable IPv6 addresses. However, on the v4 side, there are more options. ~~Either t~~The v4 interfaces of the NAT64 boxes and/or the v4-only nodes can have either v4 private addresses or v4 public addresses. Actually, it is possible that all the different combinations make sense. It seems clear that the case where public v4 addresses are used ~~i~~on both the v4 interface of the NAT64 box and the v4-only nodes is relevant. The case where the v4-only node has a private v4 address and the NAT64 box has a public v4 address seems also possible, but here it seems reasonable to assume that a NAT box will exist between the v4 only node and the NAT64 box. The case where both the v4 node and the NAT64 box have v4 private addresses could also make sense, since this could apply to a scenario where a site that has v4 private addresses and v6 addresses could try to use a NAT64 box internally. The last case, where the v4 node has public address and the NAT64 box has a private address seems harder to justify though.

Comment [DT27]: Not really. The IPv6-only system is on a leaf subnet, with the NAT64 box as the "router". Then that router is connected to the Internet behind a NAT. The v4 node is a typical Internet web server. This scenario seems entirely justifyable.

Another consideration related to v4 addressing of the NAT64 approach is the number of v4 addresses required by the NAT64 box. It is possible that some NAT64 approaches require a pool of v4 addresses instead of a single v4 address. Considering the status of the v4 address space consumption, it may not be feasible to use a NAT64 approach that requires a ~~big~~ large number of v4 public addresses.

3.4. Name-space considerations

One of the major choices ~~that are~~ faced when designing a NAT64 mechanism that enables communication initiated by the v4-only node towards a v6-only node. In this case, the v4-~~-~~only node needs to identify the v6-~~-~~only node and the problem is that there is no means to permanently map the v6 address space in the v4 address space. So in order to enable a v4-only node to identify a v6-only node a name space other than the IPv4 address space is needed. We will next discuss some options that could be considered to identify v6 nodes in the v4 world.

A first option is to use IPv4 addresses to identify IPv6 nodes. The problem is that the v6 address space is much bigger than the v4 address space, so it is not possible to do permanent mapping between these two. This basically implies that dynamic mapping between a given v4 address and different v6 addresses are established. While this works for some types of application behavior, it does not support others, such as communications initiated by a v4 node towards a v6 node in a general case (it is possible for a given subset of v6 nodes, but not as a general solution).

A second option is to use IPv6 addresses themselves. In this case, the IPv4 node is aware of the IPv6 address of the destination and it uses it to identify the target at the NAT64 box. This option would likely imply modifications in the v4 nodes.

A third option is to use FQDNs to identify nodes. In this case v4 nodes identify v6 nodes using FQDNs, which is already supported in the v4 world. The difficulties with such an approach is that DNS ALGs are likely to be required.

Comment [DT28]: Another difficulty is that apps may not be using DNS at all. E.g. in the callback mechanism, you might not have an FQDN at all (and indeed there may actually be no FQDN for one side at all).

A fourth option is to use a combination of IPv4 address, transport protocol and port for identification of a v6 node or a v6 flow.

3.5. Market timing considerations

We expect translation mechanisms to require deployment in the very near term, prior to IPv4 address depletion, and to be interoperable with end systems that have been deployed in that timeframe. Since address space depletion is expected to occur in the 2010-2012 timeframe and host software tends to be changed primarily when people buy new hardware (every 2-3 years on average), we expect that this needs to be compatible with currently-deployed Windows (XP and Vista), MacOSX (Tiger and Leopard), Linux, and Solaris operating systems. That argues for a solution that requires no changes to host software that cannot be reasonably expected to be deployed via patch update procedures - this is otherwise all solved in network devices.

4. Requirements for new generation of v4-v6 translation mechanisms

This list of requirements basically should contain all the aspects that should be considered when designing a new generation of translation mechanisms.

4.1. Basic Requirements that MUST be supported

These are the requirements for short term mechanism behaviour

R1: Changes in the hosts

The translation mechanism MUST NOT require changes in the v4-only nodes to support the Basic requirements described in this section. The translation mechanism MAY require changes to v6-only nodes.

Comment [DT29]: Is a node that is dual-stack-capable, but that cannot get an IPv4 address, considered to be a v6-only node? If so, then Vista (etc) on the IETF-v6only network is an example of this case. Why wouldn't the "MUST NOT" equally apply here? Such OS's already shipped as you point out above.

R2: Basic communication support

The ~~T~~translation mechanism must support v4-initiated and v6-initiated (?) short-lived local handle.

Comment [DT30]: MUST?

R3: Interaction with dual-stack hosts

The ~~T~~translation mechanism MUST allow using native connectivity when it is available. This means that if a v6-only node~~s~~ wants to communicate with a dual stack node, it must use native v6 connectivity and if a v4-only node~~s~~ wants to communicate with a dual stack node, it must use native v4 connectivity.(In this case, dual stack means a host with both IPv6 and IPv4 stacks, which are both active, i.e. they have v4 and v6 connectivity).

R4: Private Addressing.

The translation mechanism MUST support v4-initiated short-lived local handle type of communication when the v4-only node has a private v4 address. This covers both the case~~s~~ when there is a site with v4 private addresses and v6 addresses and the case where there is a site connected to the v4 Internet through a NAT.

Comment [DT31]: If I read this paragraph right, it means that v4-to-v4 NAT MUST be usable rather than requiring v6-to-v4 NAT, right?

R5: DNS semantics preservation

Any modifications to DNS responses associated with translation MUST NOT violate standard DNS semantics. This includes in particular that a DNS response should not be invalid if it ends up in the wrong context, i.e. traversing a non expected part of the topology.

Comment [DT32]: Does this include DNSSec or not?

R6: Routing

IPv6 routing should not be affected in any way, and there should be no risk of importing "entropy" from the IPv4 routing tables into IPv6.

Comment [DT33]: Meaning SHOULD NOT or MUST NOT?

Comment [DT34]: SHOULD or MUST?

R7: Protocols supported

The translation mechanism MUST support at least TCP, UDP, ICMP, TLS.

R8: Behave-type requirements

Comment [DT35]: Currently this item is really a TODO, rather than a set of requirements…

We could include a set of requirements similar to the ones defined by the BEHAVE WG related to Mapping timeout (5min), Address mapping behaviour (Endpoint independent, Address Dependent, Address and Port dependent), Port Assignment(Port preservation, no port preservation, port overloading), Filtering behaviour (Endpoint independent, Address Dependent, Address and Port dependent). However, this may be assuming some form of solution, so maybe this should be defined later, once the solution space has been explored.

R9: Fragmented packets

The translation mechanism MUST support fragmented packets when the fragments arrive in an ordered fashion.

R10: Security

The adoption of the translation mechanism MUST ~~not~~ NOT introduce new vulnerabilities in the Internet

Comment [DT36]: This requirement is pretty much impossible to meet in practice, as any new mechanism will in general have some new threats associated that are specific to it.

4.2. Important things that SHOULD be supported

I1: DNSSec support

DNSSec support SHOULD NOT be prevented. If the translation mechanism is used jointly with DNSSec, then DNSSec requirements take precedence over the translation requirements. Morevoer DNSSec must not be weakened in any way

Comment [DT37]: "MUST NOT"?

I2: Operational flexibility

It should be possible to locate the translation device at an arbitrary point in the network (i.e., not at fixed points such as a site exit), so that there is full operational flexibility.

Comment [DT38]: SHOULD?

Comment [DT39]: I think this word is meant in the "deploy" sense, not in the "find" (by an end node) sense, correct? If so, suggest "deploy".

I3: Central Management

Any configuration need for an IPv6 host to make use of the mechanism should be possible centrally, e.g., a DHCP option.

Comment [DT40]: SHOULD?

I4: Fragmented packets bis

The translation mechanism SHOULD support fragmented packets when the fragments arrive in an out of order fashion.

I5: Richer application behaviour support

The translation mechanism SHOULD support the other types of application behaviours, including Long-lived application associations, callbacks and referrals. In order to support this, ~~,~~ the translation mechanism MAY require changes to v4-only nodes too.

Comment [DT41]: Since this is in the SHOULD section, suggest "SHOULD NOT".

I6: MIPv6 support

The translation mechanism SHOULD ~~not~~ NOT prevent MIPv6 Route Optimization when the CN is a v4-only node.

I7: SCTP support

The translation mechanism SHOULD ~~not~~ NOT prevent an SCTP communication

between a v6-only node and a v4-only node.

I8: DCCP support

The translation mechanism SHOULD ~~not~~ NOT prevent a DCCP communication between a v6-only node and a v4-only node.

I9: Multicast support

The translation mechanism SHOULD ~~not~~ NOT prevent multicast traffic between the v4-only nodes and the v6-only nodes.

4.3. Non-goals

It would be important that the translation mechanism could support IPSec using AH and ESP both in tunnel and transport modes. However, IPSec and translation approaches seem hardly compatible, so it is a non-goal trying to support IPSec through the translation mechanism.

Comment [DT42]: But what about IPsec *TO* the translator, i.e. where the translator is an IPsec Gateway?

5. Contributors

This draft contains contributions from Iljitsch van Beijnum, Brian Carpenter and Elwyn Davies (this doesn't mean that they agree on the draft, just that we have used text provided by them).

6. Security considerations

TBD

7. Acknowledgments

Marcelo Bagnulo is partly funded by Trilogy, a research project supported by the European Commission under its Seventh Framework Program.

8. References

8.1. Normative References

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

[2] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/ Co-existence Security Considerations", RFC 4942, September 2007.

8.2. Informative References

[3] Bush, R. and D. Meyer, "Some Internet Architectural Guidelines and Philosophy", RFC 3439, December 2002.

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

[5] Templin, F., Gleeson, T., Talwar, M., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", RFC 4214, October 2005.

[6] Tsirtsis, G. and P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February 2000.

[7] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[8] Aoun, C. and E. Davies, "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status", RFC 4966, July 2007.

[9] Stenberg, M. and O. Troan, "IPv6 Prefix Delegation routing state maintenance approaches", draft-stenberg-v6ops-pd-route-maintenance-00 (work in progress), December 2007.

Authors' Addresses

Marcelo Bagnulo
Huawei Labs at Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
Email: marcelo@it.uc3m.es
URI: http://www.it.uc3m.es

Fred Baker
Cisco Systems
Santa Barbara, California 93117
USA

Phone: +1-408-526-4257
Fax: +1-413-473-2403
Email: fred@cisco.com

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).