A Review and Qualitative Analysis of IPv6 and IPv4 Interoperability Technologies

Antti Maula
Helsinki University of Technology
antti.maula@tkk.fi

Abstract

Deployment of IPv6 has been delayed even though the standard has existed for over ten years and the number of free IPv4 addresses is decreasing fast. Some obstacles in deployment are economic but also technical issues remain. The Internet cannot be converted to IPv6 overnight, thus a transition period is required during which both protocols co-exist and work together seamlessly. There is a vast amount of interoperability technologies available and this paper presents proposed solutions to operate IPv6-only network segments in cooperation with IPv4-only network segments.

KEYWORDS: IPv6, IPv4, interoperability, technology review, qualitative analysis

1 Introduction

IP protocol version 4 (IPv4), which is currently used in most parts of the internet, is becoming outdated. For a start, the address space of IPv4 is too limited compared to the current demand rate for new addresses. A viable solution for IPv4 address exhaustion is to replace the old IPv4 protocol with the newer IPv6 protocol, which was standardized over ten years ago (in 1998)[7]. Despite that, the IPv6 technology remains narrowly deployed [10]. Reasons are mainly financial as network operators will not directly benefit from IPv6 deployment, but also some technical issues still exist.

One of the key issues is how to get existing IPv4 networks and new IPv6 networks to co-exist before all networks are fully IPv6 compatible. The internet is massive with its multimillion hosts and it is clear that not all existing IPv4 networks can be converted to IPv6 overnight. This means that during the transition period, IPv4 and IPv6 networks need to interoperate until all subnetworks have fully adapted to IPv6.

Several methods for interoperability exists. For example, it may be achieved by creating a routing system which transparently translates traffic between the IPv4 and IPv6 networks. This kind of translation of traffic between different network types is often called NAT. However, this term does not refer to the IPv4 NAT which is used to translate IPv4 address to another IPv4 address, but instead translates IPv4 address to IPv6 and vice versa [24]. Many different technical solutions have been proposed to solve this traffic translation problem, but no comprehensive solution seems to exist.

This paper is a qualitative analysis of the existing methods to support IPv4 and IPv6 interoperability and the paper is structured as follows: section 2 contains descriptions of existing solutions and their main technical features. Section 3 presents some discussion about the state of the technologies and section 4 presents the required future work and the final conclusions are drawn in section 5.

2 Related technologies

This section presents the existing IPv6 and IPv4 interoperability technologies and their technical details. Interoperability technologies include techniques which allow use of isolated IPv6 subnets in mostly IPv4 based Internet and all of these technologies are intended to be used only during the transition period and they should automatically stop working when underlying network infrastructure has implemented full IPv6 support. Some of the methods presented here are implemented only by software while others require additional hardware to function. These two models are namely “End-host” and “Middlebox” architectures, in which the “End-host” is software-only based and requires implementation on the end nodes only, whereas the “Middlebox” model needs some additional hardware, which is usually placed on the edge of IPv6 and IPv4 networks to act as a translation gateway.

This section is structured so that “End-host”-only based methods are presented in subsection 1 while “Middlebox” architectures are presented in subsection 2. Some implementations take advantage of both techniques, allowing the technique to work with either one alone or together. These technologies are presented in subsection 3 - “Hybrid technologies”.

2.1 End-host technologies

End-host technologies are comprised of implementations which do not require additional translation devices for operation. This method has the advantage that it can be utilized without hardware investments.

2.1.1 Dual Stack

Dual stack (also known as Dual IP layer) is more like an implementation technique rather than a transition protocol in a sense that is does not define a new communication scheme but, instead, states that for simple IPv6 and IPv4 interoperation support, nodes should support both protocols. Nodes

may be configured to use either one or both protocol implementations at the same time depending on the deployment environment. Modern implementations provide a hybrid of these both, and this allows programmers to use the same programming API for both protocol types. Dual stack technique is described in detail in RFC4213 [20]. Dual stacking can be combined with Configured Host-to-Host tunneling of "6in4" introduced in section 2.3.1.

2.1.2 6over4 - Transmission of IPv6 over IPv4 Domains without Explicit Tunnels

"6over4" [5] method utilizes IPv4 multicast to form an "virtual Ethernet" from a group of isolated IPv6 host, what is altogether the sole purpose of this method. This method also defines contents for the IPv6 local network discovery packets, enabling IPv6 neighbour discovery to work as defined in the IPv6 specification for neighbour discovery [18]. Absense of explicit tunnels makes it easier to connect separated IPv6-only hosts together, but this also poses a limitation as the underlying IPv4 network needs to support Multicast in order to take advantage of this method. "6over4" method is classified as "End-host"-only, because all configuration may be done on the hosts participating to the virtualized IPv6-network.

2.1.3 BIS - Dual Stack Hosts using the "Bump-In-the-Stack" Technique

"BIS" is a method for enabling applications written for the IPv4 protocol stack to work with IPv6-only services. "BIS" is implemented as a module intercepting data flow between TCP/IP (version 4) implementation and the network interface card driver module. This intercepted data is then transparently translated between IPv4 and IPv6 protocols. Authors of the specification claim that application support for IPv6 is poor [25]. However, at the time of writing, many important network applications, such as web servers and browsers, have already been ported to use the IPv6 protocol stack [13]. This leads to the inference that usability of this technique is limited unless some old but still important software cannot be ported to use the new IPv6 stack. In that case, this technique may prove to be very useful.

2.1.4 AYIYA - Anything In Anything

"AYIYA"[14] method is an firewall/NAT penetration protocol intended as an replacement for the "6in4" (See section 2.3.1) routing method. because the "6in4" implementation cannot pass NAT (Network Address Translation) unless the NAT is explicitly configured to forward it. "AYIYA" works on higher protocol level such as TCP or UDP, therefore bypassing firewalls and NATs easily. However, using a high level protocol causes high overhead for the tunneling, and reduces the usability of this design. Specification for this method is only available as Internet-Draft (Work in progress) which has expired on January 5, 2005. Old expiration date of the method suggests that the technique is no longer begin developed or supported.

2.1.5 ISATAP - Intra-Site Automatic Tunnel Addressing Protocol

"ISATAP"[23] is an mechanism for connecting Dual-stack nodes over IPv4 network. The method provides automatic tunneling, hence, effectively forming an virtual LAN.

It is almost identical compared to the older "6over4" method except that it doesn't require multicast support from the IPv4 layer. Multicast is replaced with NBMA (Non-broadcast multiple-access network) backend, what can be used to connect a group of hosts in link layer. Hosts can be grouped, for example; by creating a virtual circuit, which the "ISATAP" implementation can use as a link level layer.

2.2 Middlebox technologies

Middlebox technologies include techniques which require some external hardware, such as new routers with proper IPv6/IPv4 interoperability support. As its name suggests, middlebox devices are usually located at the border of a network segment and transparent to the nodes operating inside the segment.

2.2.1 SIIT

SIIT[19] stands for "Stateless IP/ICMP Translation Algorithm" and is a technique which translates headers of IP and ICMP packets between IPv6 to IPv4 protocols. SIIT allows IPv6-only nodes to access IPv4-only nodes, but does not require that participating nodes have an permanent unique IPv4 address set, which in turn, saves IPv4 addresses. Despite of this, SIIT requires temporary IPv4 addresses which will only be required during communication with an IPv4-only host. However, SIIT doesn't have any support for acquiring such address but instead leaves that as an deployment specific problem. Some hints are given, such as using an DHCP server with short IPv4 leases, but there is no concrete mechanism or process defined for this.

2.2.2 NAT-PT

NAT-PT[24] is an abbreviation for "Network Address Translation - Protocol Translation". NAT-PT aims to provide transparent routing between nodes in IPv6 networks that attempt to communicate with nodes in IPv4 networks and the other way round. It combines two methods: Network Address Translation (NAT) and Protocol Translation (PT). The method is designed so that it does not require Dual Stack support (See section 2.1.1) from the interoperating nodes.

NAT-PT uses protocol translation method from "SIIT" and combines it with NAT (Network Address Translation) as well as suitable ALGs (Application Level Gateway) for special protocol translation. Authors of this method state that NAT-PT allows most common application to work over IPv6-only and IPv4-only network segment boundaries. However, NAT-PT is nowadays considered to be deprecated due to numerous problems in its design [1]. NAT-PT was originally introduced in RFC2766 but was obsoleted by RFC4966, which changed its status to "historic" (deprecated).

2.2.3 6to4 - Connection of IPv6 Domains via IPv4 Clouds

"6to4"[6] method is a technique for connecting separated IPv6 network segments together over an existing IPv4 network infrastructure. It differs from the "6in4" method in that it does not require the same kind of explicit tunnel setup. The technique also includes support for IPv4 hosts to access nodes inside IPv6 networks by using additional relay routers. Support also exists for assigning a unique IPv6 prefix to any network with at least one public IPv4 address. This method also scales up well, as the number of relay routers can be incremented on demand.

2.2.4 SOCKS-based IPv6/IPv4 Gateway Mechanism

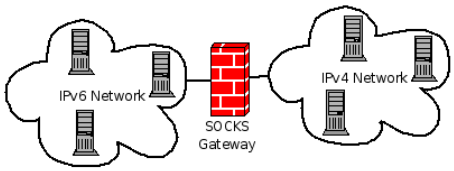

Figure 1: SOCKS operational diagram

As described in RFS3089 [15], this is a mechanism designed to provide IPv6/IPv4 interoperability by utilizing SOCKS (version 5) protocol. SOCKSv5 protocol is a framework intended to support firewall traversal in transparent and secure matter [16]. The SOCKS-based IPv6/IPv4 gateway mechanism is based solely on the existing SOCKSv5 protocol, and it does not define any new protocols.

The mechanism itself operates by using a standard SOCKS server as a middleman. The SOCKS protocol is intended to be a firewall traversal protocol, so it is located on the edge of different networks, which in turn is useful as it allows the network segments to be of different type. SOCKS server accepts connections from both sides and acts as an application level gateway for the connections, effectively allowing IPv6 and IPv4 packets to traverse between different network types. To make this transparent for applications, this method requires some modification for the socket handling library of the accessing hosts to take care of the extra steps for constucting a new connection to outside network. Diagram 1 shows the basic use case of SOCKS gateway based IPv6/IPv4 translation.

2.2.5 Stateful NAT64 & DNS64 - Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers

Internet-drafts for NAT64 and DNS64 are the newest drafts proposed by the Behave working group to support IPv6/IPv4 interoperability. These new technique together are intended to replace the old and deprecated "NAT-PT" method, as it has solutions for problems introduced by the old "NAT-PT" method. (Assuming these current drafts are accepted for the standards track)

"Stateful NAT64"[3] defines a method for translating IPv6 packets to IPv4 packets and vice-versa, while DNS64[4] is a method for synthesizing IPv6 DNS records (AAAA) from IPv4 DNS records (A). Together these two methods provide means of communication between an IPv6-only client and an IPv4-only server. The combined mechanism also enables direct communication between an IPv4 and an IPv6 node when the connection can be negotiated using existing NAT-bypassing or P2P techniques. "Stateful NAT64" also supports IPv4-initiated connections to a statically configured set of IPv6-hosts.

The NAT64 part of the method is implemented with "IP/ICMP Translation Algorithm", also known as "xlate"[17]. in which the headers of packets are modified according to connection information. If this Internet-Draft is accepted as a standard, this method will obsolete the currently existing packet header transformation scheme SIIT (See 2.2.1)

In addition to processing transmitted packets with NAT64, special measures are required for DNS processing as well. Hosts running IPv4 only usually have a DNS record only for IPv4 address, what is called as "A" record. IPv6 hosts cannot communicate with IPv4 hosts which don't have IPv6 addresses even with the NAT64 method described above. In addition to this method, the DNS64 implementation adds support for dynamically creating synthesized IPv6 ("AAAA") DNS-records from existing IPv4 "A" DNS-records. A synthesized record is created by adding a certain prefix to the IPv4 "A" record.

The protocol has the following properties:

- NAT64 is compatible with protocols TCP[9], UDP[2] and ICMP[21] in a sense that it allows them to pass through uninterrupted.
- The NAT64 is capable of sharing a single IPv4 addresses while forming IPv6 subnets. One IPv4 address may be shared among multiple nodes which aids at slowing down the IPv4 exhaustion rate.
- The NAT64 supports both stateless and stateful operation, but when the NAT64 middlebox is operating in stateless mode (no state has been learned yet), only connections originating from IPv6 nodes are supported.

2.2.6 6rd - IPv6 rapid deployment

"6rd"[8] method is an special case of the "6to4" (see section 2.2.3) technology, and it was implemented by a French Internet service provider, namely "Free of the Iliad group", to increase the popularity of IPv6 among their customers. Technically it differs from the "6to4" in a sense that it uses different IPv6 prefix per service provider, instead of the fixed IPv6 prefix of the "6to4" specification. Service provider added support for the "6rd" method to their "Freebox home-gateway"-device, and advertised to their clients that they could become native IPv6 users just by upgrading the software or their home gateways. The main purpose for doing this was to break the circle in which Internet Service Providers wait for customer demand and customers do not demand as they have been told that the applications they are using will work with existing networks. Authors of "6rd" added some value for the IPv6 deployment, and as a result, many of their users started to use IPv6 providing middleboxes. The specification of the "6rd" is informal and not

intended to be a standard. In fact, the internet-draft for the "6rd" method has expired on October 9, 2009 and no standard or replacing new draft have been filed in to the Internet Engineering Task Force.

2.3 Hybrid technologies

This subsection includes technical approaches which do not strictly belong to End-host or Middlebox categories. Technologies presented here may require support from both layers or they can work in either side.

2.3.1 6in4 - IPv6 inside IPv4

"6in4" is a technique which encapsulates an IPv6 frame inside an IPv4 frame to transmit it over an IPv4-only network. The transmitted IP frame has its IP protocol number set to 41, which is registered as "IPv6-in-IPv4"-protocol by IANA (Internet Assigned Numbers Authority)[12]. Receiver node decapsulates the IPv6 packet from the IPv4 packet and continues to process it as IPv6[20]. This method may be utilized by single hosts but might also be implemented in as a middlebox. This mechanism is intended for connecting scattered IPv6 networks together over IPv4 medium, but can also be used to connect IPv6 to IPv4 network by using middlebox approach.

"6in4" technique is simple to implement and, for example, works well together with the dual-stack method. However, this method is somewhat limited, as it requires a lot of configuration. Configuration is required for setting up tunnels to other to set up system "6in4" connections are also easily limited by firewalls. Firewalls usually only accept TCP and UDP protocols per default, hence effectively stopping "6in4" traffic. "6in4" doesn't provide any security features, so it is possible to hijack a "6in4" connection and start acting as a middleman, so if security is required, some additional protection methods must be used.

2.3.2 Teredo - Tunneling IPv6 over UDP through Network Address Translations NATs

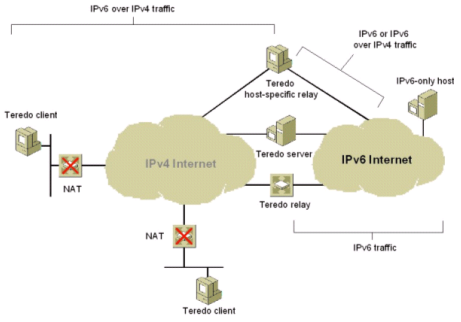

Figure 2: Teredo infrastructure (Image source Microsoft TechNet[22])

"Teredo"[11] is a tunneling protocol to provide access for nodes behing IPv6-incompatible NAT devices. The method defines means to encapsulate IPv6 traffic inside IPv4-UDP frames to be routeable in an IPv4 network.

Basic infrastructure of a Teredo system is shown in image 2. Operation of Teredo is quite complex, as it has several operating steps:

1. **Diagnosis.** In this phase, Teredo analyzes the connectivity of the current home network and attempts to determine the type of possible NAT device in the way.
2. **Addressing.** This phase assigns IPv6 routeable address for each participating IPv4 host.
3. **Transmit.** Encapsulate IPv6 packets inside IPv4-UDP frames and transmit to Teredo proxy or directly to native IPv6 hosts if possible.

Teredo consists of three different node types:

- **Client.** Node behing a NAT and connected to IPv4-only network. Requires Teredo tunneling in order to reach IPv6 network.
- **Server.** Preconfigured hosts for the initial connection analysis. No payload is transferred through the Teredo server, so its load on the network is small.
- **Relay.** Node responsible for transferring data through tunnels to nodes connected to IPv4-only networks. Requires a lot of bandwidth as it relays traffic.

Teredo is a very well supported protocol and implementations exists at least for Windows and Linux(compatible) operating systems. When Teredo cannot find a direct route between nodes, it will fall back to use centralized relays.

3 Discussion

Interoperability is somewhat a technical problem, but in my opinion, it is mostly economical. IPv6 standard has been around for a decade, but only a small percentage of internet service providers have deployed it. Even though the technical issues with interoperability would be solved, there still remains the very old question: Who is willing to pay for IPv6 deployment as long as the old infrastructure still works. Statistics clearly show that IPv4 addressess will be exhausted in two years, it has not motivated service providers to adapt it. In my opinion, the only possible method for effective IPv6 deployment would be by rewarding those who use it and fining those who do not.

4 Future work

This paper presents all existing major interoperability technologies. Description of technologies is solely based on their documentation and no empirical testing has been performed. The research showed that selection of interoperability techniques is wide-ranging. However, most of the reviewed techniques solve a set of problems but also many times also introduce some new ones. Often the problems in technology implementations are realized after the technology is already implemented and taken into use. So to be able to evaluate some specific implementation in detail, some empricial research is required. This empirical research should evaluate

the practical quality of these methods, providing data about the real software and hardware implementations. However, this kind of empirical testing is outside of this papers scope. Nevertheless, this kind of testing should be conducted before making conclusive recommendation about the best choise for interoperability technology.

5 Conclusions

IPv6 and IPv4 interoperability has been an issue since the IPv6 was designed. Some simple methods for interoperability were designed along with the IPv6 protocol, but future studies have shown that these measures are quite insufficient. Pressure for finding an suitable solution is getting high, as IPv4 addresses are running low. Currently, there is no standardized solution to solve the problem, but some promising technologies have been proposed.

The most recent work on the interoperability conducted by the Behave group seems promising, as they have taken into account most of the problems discovered from the previous interoperability technology implementations. The latest drafts from the Behave group are the NAT64+DNS64 protocols, which address all the issues discovered in its predecessor "NAT-PT". While the NAT64/DNS64 protocal stack seems promising, only time will show its real value. For example, the technology has not yet reached standard status, and if it ever will, it will face evaluation by a much larger audience. Larger testing group may find new issues which may have not shown in the technology evaluation earlier, therefore endering another promising interoperability technology obsolete.

References

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

[2] F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. RFC 4787 (Best Current Practice), Jan. 2007.

[3] M. Bagnulo, P. Matthews, and I. van Beijnum. Stateful nat64: Network address and protocol translation from ipv6 clients to ipv4 servers (work in progress). Internet-Draft, Work in Progress, 2010.

[4] M. Bagnulo, A. Sullivan, P. Matthews, I. van Beijnum, and M. Endo. Dns64: Dns extensions for network address translation from ipv6 clients to ipv4 servers (work in progress). Internet-Draft, Work in Progress, 2010.

[5] B. Carpenter and C. Jung. Transmission of IPv6 over IPv4 Domains without Explicit Tunnels. RFC 2529 (Proposed Standard), Mar. 1999.

[6] B. Carpenter and K. Moore. Connection of IPv6 Domains via IPv4 Clouds. RFC 3056 (Proposed Standard), Feb. 2001.

[7] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), Dec. 1998. Updated by RFCs 5095, 5722.

[8] R. Despres. IPv6 Rapid Deployment on IPv4 Infrastructures (6rd). RFC 5569 (Informational), Jan. 2010.

[9] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. RFC 5382 (Best Current Practice), Oct. 2008.

[10] S. Gunderson. Global IPv6 statistics-measuring the current state of IPv6 for ordinary users, 2008.

[11] C. Huitema. Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs). RFC 4380 (Proposed Standard), Feb. 2006.

[12] IANA. Assigned internet protocol numbers.

[13] IPv6.org. Ipv6 enabled applications.

[14] U. . S. Jeroen Massar. Anything in anything (ayiya). Internet-Draft.

[15] H. Kitamura. A SOCKS-based IPv6/IPv4 Gateway Mechanism. RFC 3089 (Informational), Apr. 2001.

[16] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. SOCKS Protocol Version 5. RFC 1928 (Proposed Standard), Mar. 1996.

[17] X. Li, C. Bao, and F. Baker. Ip/icmp translation algorithm (work in progress). Internet-Draft, Work in Progress, 2010.

[18] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). RFC 2461 (Draft Standard), Dec. 1998. Obsoleted by RFC 4861, updated by RFC 4311.

[19] E. Nordmark. Stateless IP/ICMP Translation Algorithm (SIIT). RFC 2765 (Proposed Standard), Feb. 2000.

[20] E. Nordmark and R. Gilligan". Basic Transition Mechanisms for IPv6 Hosts and Routers. RFC 4213 (Proposed Standard), Oct. 2005.

[21] P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. RFC 5508 (Best Current Practice), Apr. 2009.

[22] M. TechNet. Teredo overview. `http://technet.microsoft.com/en-us/library/bb457011.aspx`. Checked 15.04.2010.

[23] F. Templin, T. Gleeson, and D. Thaler. Intra-Site Automatic Tunnel Addressing Protocol (ISATAP). RFC 5214 (Informational), Mar. 2008.

[24] G. Tsirtsis and P. Srisuresh. Network Address Translation - Protocol Translation (NAT-PT). RFC 2766 (Historic), Feb. 2000. Obsoleted by RFC 4966, updated by RFC 3152.

[25] K. Tsuchiya, H. Higuchi, and Y. Atarashi. Dual Stack Hosts using the "Bump-In-the-Stack" Technique (BIS). RFC 2767 (Informational), Feb. 2000.