Learning the IPv6 Prefix of a Network’s IPv6/IPv4 Translator
draft-wing-behave-learn-prefix-04

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April 29, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

Some IPv6 applications obtain IPv4 address literals and want to communicate with those IPv4 hosts through an IPv6/IPv4 translator. The IPv6 application can send an IPv6 packet through the translator.
if it knows the IPv6 prefix of the IPv6/IPv4 translator. In many IPv6/IPv4 translation deployments, that IPv6 prefix is not fixed; rather, the prefix is chosen by the network operator. This specification provides three methods for a host to learn the IPv6 prefix of its IPv6/IPv4 translator. Unicast, any-source multicast (ASM), and source-specific multicast (SSM) are supported.

Table of Contents

1. Terminology .................................................. 3
2. Introduction .................................................. 3
3. Discussion on Mechanisms ................................. 4
4. Mechanisms to Learn the Translator’s IPv6 Prefix and Length 5
   4.1. Using DNS ............................................. 5
   4.2. Using DHCPv6 ......................................... 6
5. Authenticating the Learned Prefix .......................... 7
6. Security Considerations ........................................ 8
7. IANA Considerations ........................................... 8
8. Acknowledgements ............................................... 9
9. References ..................................................... 9
   9.1. Normative References ................................... 9
   9.2. Informative References ................................. 9
Appendix A. For future study ....................................... 11
   A.1. multi-homed hosts ....................................... 11
   A.2. Unicast and multicast translators ........................ 11
Appendix B. IPv4 Address Literals on the Internet ............ 11
Appendix C. Changes ................................................. 12
   C.1. Changes from -03 to -04 ................................ 12
   C.2. Changes from -02 to -03 ................................ 12
   C.3. Changes from -01 to -02 ................................ 12
   C.4. Changes from -00 to -01 ................................ 12
Author’s Address .................................................. 13
1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

AFT: Address Family Translator. A device that translates between IP address families.

DNS64: The function of synthesizing an AAAA record from an A record (also called "DNS rewriting" or "DNS-ALG"), described in [I-D.ietf-behave-dns64].

NSP (Network-Specific Prefix): A prefix assigned to an IPv6/IPv4 translator that uses a prefix belonging to the network operator.

2. Introduction

Certain applications, operating in certain translation scenarios, can benefit from knowing the IPv6 prefix of their IPv6/IPv4 translator. First, the host must be operating in an IPv6-initiated scenario with a local translator. The Framework document [I-D.ietf-behave-v6v4-framework] describes these as Scenario 1, "IPv6 network to IPv4 Internet", and Scenario 5, "An IPv6 network to an IPv4 network". Learning the prefix is useful for both stateful translation and stateless translation.

With those scenarios, the IPv6 host usually performs a DNS AAAA query which is processed by a DNS64 server. The DNS64 server generates a synthetic AAAA response, when necessary. This synthetic AAAA response contains the prefix of the IPv6/IPv4 translator. When the IPv6 host sends a packet to that address returned in the AAAA response, the packet is routed to the translator which translates it to IPv4. This functionality is transparent to the IPv6 host, for the most part.

However, an IPv6 application can also obtain an IPv4 address literal and wants to communicate with that IPv4 address. So far, several scenarios have been identified where this occurs:

- host-based DNSSEC validation (Section 6 of [I-D.ietf-behave-dns64])
- BitTorrent (Section 2.2 of [I-D.wing-behave-nat64-referrals])
- multicast translation ([I-D.venaas-behave-v4v6mc-framework] and Section 4 of [I-D.venaas-behave-mcast46])
o URI schemes with host IPv4 address literals rather than domain names (e.g., http://192.0.2.1, ftp://192.0.2.1, imap://192.0.2.1, ipp://192.0.2.1). See also [I-D.wing-behave-http-ip-address-literals] which describes a different workaround than the solution described in this document.

o update the host’s RFC3484 preference table to prefer translated prefixes below native prefixes.

o allow the host to perform its own DNS64 function. This allows the host to provide translation functions to IPv4 applications using, for example, BIS [I-D.huang-behave-rfc2767bis] or BIA [I-D.huang-behave-rfc3338bis].

When an IPv6/IPv4 translator is used with a Network-Specific Prefix (NSP), it is necessary for such applications to learn the IPv6 prefix (and length) of the translator so that the application can create an IPv6 packet that will be routed to the translator and be translated to IPv4.

Issue-1: Even when the Well-Known Prefix (WKP) is used, it may be useful for the host and/or the applications to know there is, in fact, a translator operating on the network. The mechanisms described in this draft could provide such an indication to the host and its applications. The need for learning the prefix with WKP is for future study.

3. Discussion on Mechanisms

Both DNS and DHCP are described in this document. It would be desirable to use DHCPv6, as it is intended to configure network settings such as the network’s IPv6/IPv4 translator. However, there is not ubiquitous support of DHCPv6.

DNS:

* available to all OSs and applications, without regard for OS support or network device support.

DHCPv6:

* requires DHCPv6 support in host operating system and network. Apple’s OSX does not support DHCPv6.

* requires OS provide API for application to query the new DHCP option described in this document. Microsoft’s Windows Vista provides such an API. Support in other OSs is unknown.
4. Mechanisms to Learn the Translator’s IPv6 Prefix and Length

Both the IPv6 prefix of the translator and the prefix length of the translator need to be learned. With that information, the application can generate an appropriate IPv6 address that will be routed to the translator for the translator to process.

The host can learn the necessary information using DNS or DHCP as described in the following sections.

Issue-3: If a conflict exists between DNS or DHCP which should take precedence?

4.1. Using DNS

Issue-4: Should we just use a TXT record, perhaps like "_TRANSLATE64", "_ASMTRANSLATE64", and "_SSMTRANSLATE64", instead of using NAPTR? A simple TXT record would ensure immediate ubiquitous support across all OSs and all DNS management systems.

This specification defines a new U-NAPTR [RFC4848] application to discover the translator’s IPv6 prefix and length. The input domain name is the exact same as would be used for a reverse DNS lookup, derived from the host’s IPv6 in the ".ip6.arpa." tree and follows the construction rules in Section 2.5 of [RFC3596]. This is shortened to 20 labels (representing a /64 network prefix) and, if DNS returns an error is shortened to 16 labels (representing a /48 network prefix).

If an IPv6/IPv4 translator is present on the network, the successful result of one of those queries will produce a NAPTR record with the desired service tag "TRANSLATE64:" which contains the unicast IPv6 prefix and prefix length of the translator, separated by a "/" (the same syntax as specified in Section 2.3 of [RFC4291]). The service tags "ASMTRANSLATE64:" and "SSMTRANSLATE:" are used for ASM and SSM.

For example, a host with the IP address 2001:db8:1:2:3:4:567:89ab would first send an NAPTR query for 3.0.0.0.2.0.0.0.1.0.0.0.8.b.d.0.1.0.0.2.IP6.ARPA (20 elements, representing a /64 network prefix). If that fails (returns NXDOMAIN), it would send an NAPTR query for 2.0.0.0.1.0.0.0.8.b.d.0.1.0.0.2.IP6.ARPA (16 elements, representing a /48 network prefix).

Note: Both /64 and /48 prefix lengths are shown in this version of the document for illustrative purposes. The number of elements
of this query will depend on the prefix length(s) defined by the
BEHAVE working group for a translator. If the BEHAVE working
group decides that all translators will have a certain prefix
length, then only one DNS query is sent.

If the host needs to authenticate the prefix it just learned (e.g.,
because the host is running a DNSSEC validator) the host performs the
additional authentication steps described in Section 5.

4.2. Using DHCPv6

A new DHCP option, OPTION_AFT_PREFIX_DHCP, is defined. It contains
the IPv6 unicast prefix, IPv6 ASM prefix, and IPv6 SSM prefix (and
their lengths) for the IPv6/IPv4 translator on this network.
5. Authenticating the Learned Prefix

If the host needs to authenticate the prefix it just learned (e.g., because the host is running a DNSSEC validator) the host performs the additional authentication steps described in Section 5.

Figure 1: DHCP option OPTION_AFT_PREFIX_DHCP

If the host needs to authenticate the prefix it just learned (e.g., because the host is running a DNSSEC validator) the host performs the additional authentication steps described in Section 5.
translator’s IPv6 prefix by performing the following steps:

a. The host sends a DNS PTR query for the IPv6 address of the translator (for "ipv6.arpa"), using 0 for the elements after the prefix length. This will return the fully-qualified hostname of that translator device.

b. Verify the full-qualified hostname is on the host’s configured list of authorized translators (e.g., seattle.translator.example.net).

c. Send a DNS AAAA query for that hostname.

d. Verify the AAAA response matches the IPv6 address obtained in step 1.

e. Perform DNSSEC validation of the AAAA response.

For example, if the translator’s IPv6 prefix length is /48, the host would send a PTR query for 2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.IP6.ARPA which would return a hostname, seattle.translator.example.net. The host verifies that seattle.translator.example.net is on its configured list of authorized translators, as maintained in a text file. The host sends an AAAA query for seattle.translator.example.net and verifies the AAAA response contains the same IPv6 address. The host then validates the DNSSEC signature for seattle.translator.example.net.

6. Security Considerations

After learning the IPv6 prefix of its translator by following the procedures in this specification, the IPv6 host will utilize this information for subsequent actions (e.g., sending a packet to it, or using that information to synthesize DNS records or to perform DNSSEC validation). If an attacker provides a fraudulent IPv6 to the IPv6 host, the attacker can become on-path for traffic to/from that IPv6 host and perform passive or active eavesdropping or traffic analysis. To protect against this attack, it is RECOMMENDED that IPv6 hosts be configured with the names of authorized translators and RECOMMENDED that IPv6 hosts uses DNSSEC to validate that name matches the IPv6 prefix learned via DNS or DHCPv6 as described in Section 5.

7. IANA Considerations

A new DHCPv6 option, OPTION_AFT_PREFIX_DHCP, needs to be assigned by IANA.
The new NAPTR Application Service tag "TRANSLATE64" is registered with IANA.

8. Acknowledgements

This draft was fostered by discussion on the 46translation mailing list and at the v4v6 Interim in Montreal. Special thanks to Iljitsch van Beijnum, Andrew Sullivan, Marcelo Bagnulo Braun, Fred Baker, and Xing Li for their comments and suggestions.

The mechanism to perform a shortened NAPTR query was described first by Martin Thomson [I-D.thomson-geopriv-res-gw-lis-discovery].

Thanks to Ralph Droms for his help with DHCPv6. Thanks to John Schnizlein for improving the DNS learning algorithm. Thanks to Keith Moore and Scott Brim for suggesting HTTP IPv4 address literals. Thanks to Stig Venaas for help with multicast. Thanks to Xuewei Wang and Xiaohu Xu for suggesting IPv6 Router Advertisements.

9. References

9.1. Normative References


9.2. Informative References


[I-D.ietf-behave-dns64] Bagnulo, M., Sullivan, A., Matthews, P., and I. Beijnum,
"DNS64: DNS extensions for Network Address Translation from IPv6 Clients to IPv4 Servers",
draft-ietf-behave-dns64-01 (work in progress),
October 2009.

[I-D.ietf-behave-v6v4-framework]
draft-ietf-behave-v6v4-framework-03 (work in progress),
October 2009.

[I-D.savolainen-mif-dns-server-selection]
Savolainen, T., "DNS Server Selection on Multi-Homed Hosts",
draft-savolainen-mif-dns-server-selection-01 (work in progress),
October 2009.

[I-D.thomson-geopriv-res-gw-lis-discovery]
Thomson, M. and R. Bellis, "Location Information Server (LIS) Discovery From Behind Residential Gateways",
draft-thomson-geopriv-res-gw-lis-discovery-02 (work in progress),
July 2009.

[I-D.van-beijnum-behave-ftp64]
Beijnum, I., "IPv6-to-IPv4 translation FTP considerations",
draft-van-beijnum-behave-ftp64-06 (work in progress),
October 2009.

[I-D.venaas-behave-mcast46]
draft-venaas-behave-mcast46-01 (work in progress),
July 2009.

[I-D.venaas-behave-v4v6mc-framework]
draft-venaas-behave-v4v6mc-framework-00 (work in progress),
July 2009.

[I-D.wing-behave-http-ip-address-literals]
Wing, D., "Coping with IP Address Literals in HTTP URIs with IPv6/IPv4 Translators",
draft-wing-behave-http-ip-address-literals-00 (work in progress),
October 2009.

[I-D.wing-behave-nat64-referrals]
Wing, D., "Referrals Across an IPv6/IPv4 Translator",
draft-wing-behave-nat64-referrals-01 (work in progress),
October 2009.
Appendix A. For future study

A.1. multi-homed hosts

A multi-homed host may have different translation devices available on each of its networks, and can learn those via DNS, DHCP.

When using DNS to learn the translator’s prefix (Section 4.1) or using DNS to authenticate the translator prefix (Section 5, it is possible a split horizon DNS exists. Such a split DNS requires the host to query the DNS server associated with that network prefix as described in [I-D.savolainen-mif-dns-server-selection].

A.2. Unicast and multicast translators

It may be necessary to use different prefixes for unicast, any source multicast (ASM), and source-specific multicast (SSM) (Section 2 of [I-D.venaas-behave-mcast46]).

Appendix B. IPv4 Address Literals on the Internet

There has been some doubt that IPv4 address literals occur on the Internet. An examination of the top 1 million domains at the end of August, 2009, showed 2.38% of the HTML in their home pages contained IPv4 address literals. This can be verified by examining the output of the following script:

```
wget http://s3.amazonaws.com/alexa-static/top-1m.csv.zip
unzip top-1m.csv.zip
cat top-1m.csv |
cut -d "," -f 2 |
xargs -I % -n 1 -t wget -nv % -O - --user-agent="Mozilla/5.0" |
grep -E "http://[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
Of the top 1 million websites at the end of August, 2009, 3455 of them are IPv4 address literals. This can be verified with the following script:

```bash
wget http://s3.amazonaws.com/alexa-static/top-1m.csv.zip
unzip top-1m.csv.zip
grep -E "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}" top-1m.csv | wc
```

Appendix C. Changes

C.1. Changes from -03 to -04

- Provided examples of IPv4 address literals with HTTP on the Internet (Appendix B).
- Removed Router Advertisements.

C.2. Changes from -02 to -03

- Removed FTP interworking, because [I-D.van-beijnum-behave-ftp64] proposes that FTP clients use the same IP address for the data connection as the control connection. This eliminates the need for the FTP client to learn the translator’s prefix.
- Added multicast to DHCP and RA messages.

C.3. Changes from -01 to -02

- Provided another method of using RA message for a host to learn its translator’s IPv6 prefix and length.
- Added IPv4 address literals in URIs and multicast as benefactors for learning the translator’s prefix.
- Added FTP interworking using PASV.
- Clarified which Scenarios this applies to, and that this is for stateful and stateless.

C.4. Changes from -00 to -01

- Made clearer this is for NAT64 prefix (changed title and some text).
- Changed from querying for ".aft_prefix" TXT record to querying ipv6.arpa NAPTR record.
BitTorrent is another application that benefits from knowing the NAT64 prefix; previously only DNSSEC was listed.

changed to standards track.

Author’s Address

Dan Wing
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134
USA

Email: dwing@cisco.com