IPv6 operations
Internet-Draft
Obsoletes: 6791 (if approved)
Updates: 7915 (if approved)
Intended status: Standards Track
Expires: 10 May 2026

D. Lamparter
NetDEF, Inc.
J. Linkova
Google

6 November 2025

Using Dummy IPv4 Address and Node Identification Extensions for IP/ICMP translators (XLATs)
draft-ietf-v6ops-icmpext-xlat-v6only-source-01

Abstract

This document suggests that when a source IPv6 address of an ICMPv6 message can not be translated to an IPv4 address, the protocol translators use the dummy IPv4 address (192.0.0.8) to translate the IPv6 source address, and utilize the ICMP extension for Node Identification (draft-ietf-intarea-extended-icmp-nodeid) to carry the original IPv6 source address of ICMPv6 messages. This document obsoletes RFC6791, Stateless Source Address Mapping for ICMPv6 Packets and updates IP/ICMP Translation Algorithm (RFC7915).

About This Document

This note is to be removed before publishing as an RFC.

Source, version control history, and issue tracker for this draft can be found at https://github.com/eqvinox/icmpext-clat-source.

(Note the draft was renamed (from "clat" to "xlat") prior to submission but changing the repository name on github breaks too many things to be worth the effort.)

Status of This Memo

This Internet-Draft will expire on 10 May 2026.

Table of Contents

1. Introduction

To allow communication between IPv6-only and IPv4-only devices, IPv4/IPv6 translators translate IPv6 and IPv4 packet headers according to the IP/ICMP Translation Algorithm defined in [RFC7915]. For example, 464XLAT ([RFC6877]) defines an architecture for providing IPv4 connectivity across an IPv6-only network. The solution contains two key elements: provider-side translator (PLAT, usually in a form of stateful NAT64, [RFC6146])) and customer-side translator (CLAT). CLAT implementations translate private IPv4 addresses to global IPv6 addresses, and vice versa, as defined in [RFC7915].

To map IPv4 addresses to IPv6 ones the translators use the translation prefix (either a well-known or a network-specific one, see [RFC6052]). The resulting IPv6 addresses can be statelessly translated back to IPv4. However, this is not the case for arbitrary global IPv6 addresses; those addresses can only be translated to IPv4 by stateful translators and only if a corresponding translation rule exists.

One scenario where this is necessary is translating ICMPv6 error messages sent by intermediate nodes to the CLAT address in a 464XLAT environment. The most typical example is a diagnostic tool like traceroute sending packets to an IPv4 destination from an IPv6-only host. Received ICMPv6 Time Exceeded messages are translated to ICMP Time Exceeded messages. If those packets were originated from an IPv4 address and translated to ICMPv6 by the PLAT (NAT64) device, then the source address of such a packet can be mapped back to IPv4 by removing the translation prefix. However ICMPv6 error messages sent by devices located between the IPv6-only host and the NAT64 device have "native" IPv6 source addresses, which cannot be mapped back to IPv4. Those packets are usually dropped and tools like traceroute can not represent IPv6 intermediate hops in any meaningful way. Such behaviour complicates troubleshooting. It might also be confusing for users and increases operational costs, as users report packet loss in the network based on traceroute output.

It should be noted that a similar issue occurs in IPv6 Data Center Environments when an ICMPv6 error message is sent to an IPv4-only client. As per Section 4.8 of [RFC7755], ICMPv6 error packets are usually dropped by the translator.

[I-D.ietf-intarea-extended-icmp-nodeid] defines the Node Identification Object which can carry an IP Address Sub-Object, containing an IP address. This document proposes that when an ICMPv6 error message is translated to an ICMP one, and the IPv6 source address cannot be mapped or translated to an IPv4 one, the translator uses the dummy IPv4 address as the IPv4 source and appends a Node Identification Object with an IP Address Sub-Object, containing the IPv6 address of the original ICMPv6 error message.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

Translator: a device performing translation between IPv6 and IPv4 packet headers according to the IP/ICMP Translation Algorithm defined in [RFC7915]. Translators can be stateless ([RFC7915]) or stateful ([RFC6146]).

Translatable IPv6 address: an IPv6 address which matches the NAT64 prefix known to the translator, or for which the translator has a stateful entry, mapping that IPv6 address to an IPv4 one.

Untranslatable IPv6 address: an IPv6 address which does not match the NAT64 prefix(es) configured on the translator, and for which the translator does not have a stateful entry, mapping this IPv6 address to an IPv4 one.

4. Translation Behavior

4.1. Overview

Whenever a translator translates an ICMPv6 Destination Unreachable, ICMPv6 Time Exceeded or ICMPv6 Packet Too Big ([RFC4443]) to the corresponding ICMPv4 ([RFC0792]) message, and the IPv6 source address in the outermost IPv6 header is untranslatable, the translator SHOULD use the dummy IPv4 address (192.0.0.8) as the IPv4 source address for the translated packet.

To preserve the original IPv6 source address of the packet, the translator SHOULD append a Node Identification Object ([I-D.ietf-intarea-extended-icmp-nodeid]) with an IP Address Sub-Object containing the IPv6 source address of the original ICMPv6 packet.

The translator MUST NOT use 192.0.0.8/32 to translate the source IPv6 address and MUST NOT add the extension if the packet's IPv6 source address is translatable.

4.2. Adding Node Identification Extension Object

A Node Identification Extension Object SHOULD be added when translating:

* ICMPv6 Destination Unreachable to ICMPv4 Destination Unreachable

* ICMPv6 Time Exceeded to ICMPv4 Time Exceeded.

* ICMPv6 Packet Too Big to ICMPv4 Destination Unreachable.

and the IPv6 source address in the outermost IPv6 header is untranslatable.

When adding the Node Identification Extension Object, the translator MUST include the IP Address Sub-Object containing the original IPv6 source address of the packet.

This document doesn't prescribe the exact procedure for adding the Node Identification Extension Object when translating ICMPv6 messages to ICMPv4. Section 11.1 describes one possible approach, but the choice is left to implementors, as long as the following requirements are met:

* The resulting ICMPv4 message contains the Node Identification Extension Object with the IP Address Sub-Object.

* The checksum field of the Extension Header (Section 7 of [RFC4884]) is updated accordingly.

4.2.1. Order of Operations and Translating ICMPv6 Packet Too Big

This specification does not prescribe whether the Node Identification Extension Object is added before or after translating an ICMPv6 message to ICMPv4. This choice is up to the implementation. However ICMP Extensions can not be added to ICMPv6 Packet Too Big messages (see Section 4.6 of [RFC4884]). Therefore in order to be able to add the Node Identification Extension Object and preserve the original

untranslatable IPv6 address, the translator needs to add the object to the resulting ICMPv4 packet after it's been translated from ICMPv6. The translator MAY choose to not append the Node Identification Extension Object when translating an ICMPv6 Packet Too Big to an ICMPv4 Destination Unreachable. Such implementations SHOULD still translate ICMPv6 Packet Too Big from untranslatable sources using 192.0.0.8 as an IPv4 source address and SHOULD NOT drop those packets.

5. Previous Work

[RFC6791] proposes using the Interface Information Object and Interface IP Address Sub-Object defined in [RFC5837] to preserve information about untranslatable IPv6 addresses. However it should be noted that Section 4.2 of [RFC5837] suggests that an IPv4 packet MUST only contain an IP Interface Sub-Object representing an IPv4 address. Therefore using the mechanism described in [RFC6791] requires updating [RFC5837].

More importantly, Section 3.2 of [RFC6791] recommends using a single (or small pool of) public IPv4 address as the source address of the translated ICMP message. Such an approach assumes that the translator is configured with at least one public IPv4 address, which is often not feasible for CLAT instances running on endpoints like mobile phones, laptops etc.

The solution proposed in this document has a number of benefits:

* It does not require public IPv4 addresses configured on the translator.

* No changes to processing of the Interface Information Object are required.

* Using a dedicated IPV4 dummy address 192.0.0.8 indicates to the user that it's not an actual IPv4 address of the intermediate node.

Therefore this document deprecates [RFC6791].

6. Updates to RFC7915

This document makes the following changes to Section 5.1 ("Translating IPv6 Headers into IPv4 Headers") of [RFC7915]:

OLD TEXT

| Source Address: Mapped to an IPv4 address based on the algorithms
| presented in Section 6.

NEW TEXT

| Source Address: Mapped to an IPv4 address based on the algorithms
| presented in Section 6. When translating ICMPv6 error messages to
| ICMPv4 error messages and the valid IPv6 source address in the
| outermost IPv6 header can not be mapped to an IPv4 address (the
| address does not match the prefix used in algorithmic mapping and
| there are no static or stateful entries for that address), the
| translator SHOULD follow the recommendations in draft-ietf-v6ops-
| icmpext-xlat-v6only-source.

This document also updates the very last paragraph of Section 5.2 of [RFC7915] ("Error payload:") as follows:

OLD TEXT:

| For extensions not defined in [RFC4884], the translator passes the
| extensions as opaque bit strings and any IPv6 address literals
| contained therein will not be translated to IPv4 address literals;
| this may cause problems with processing of those ICMP extensions.

NEW TEXT:

| For extensions not defined in [RFC4884], the translator passes the
| extensions as opaque bit strings and any IPv6 address literals
| contained therein will not be translated to IPv4 address literals;
| this may cause problems with processing of those ICMP extensions.
| If the valid IPv6 source address in the outermost IPv6 header of
| the ICMPv6 messages cannot be mapped to an IPv4 address (the
| address does not match the prefix used in algorithmic mapping and
| there are no static or stateful entries for that address), the
| translator SHOULD follow the recommendations in draft-equinox-
| intarea-icmpext-xlat-source.

7. Applicability Considerations

The mechanism described in this document necessitates that the translator distinguishes between ICMPv6 packets originating from untranslatable addresses requiring translation (triggered by an IPv4 packet translated to IPv6) and native IPv6 traffic that does not. When the translator employs dedicated IPv6 address(es) for IPv4 translation (e.g., a CLAT instance acquiring dedicated address(es) or a dedicated /64), this differentiation is straightforward.

However, if the same IPv6 address is used for both IPv4 translation and native IPv6 traffic, the translator may require more complex techniques to differentiate. These techniques could include maintaining state and/or analyzing the invoking packet header within the ICMPv6 message body to determine if the invoking packet was translated

8. Security Considerations

This document does not introduce new security considerations.

9. Privacy Considerations

This document does not introduce any privacy considerations.

10. IANA Considerations

This memo includes no request to IANA.

11. Appendix

11.1. Adding a Node Identification Extension Object: Suggested Algorithm

11.1.1. Adding a New ICMP Extension Structure

If the original ICMPv6 message does not contain an ICMP Extension Structure (as defined in Section 7 of [RFC4884]), the translator SHOULD append a new ICMP Extension Structure to the ICMP message. When adding the new Extension Structure, the translator MUST:

* Create a new ICMP Extension Structure, containing one Extension Header and one Node Identification Extension object. The Node Identification Extension object MUST contain an IP Address Sub-Object, carrying the IPv6 source address of the ICMPv6 message being translated.

* Append that Extension Structure to the ICMP message.

* If the resulting packet size exceeds the minimum IPv6 MTU: truncate the embedded invoking packet by removing the trailing 28 octets (to accommodate for 4 octets of the extension header and 24 octets of the extension object).

* Set the length field of the ICMP message to the length of the padded "original datagram" field, measured in 32-bit words.

11.1.2. Adding a Node Identification Extension Object to an Existing ICMP Extension Structure

If the original ICMPv6 message already contains an ICMP Extension Structure, the translator SHOULD append a Node Identification Extension object containing the IP Address Sub-Object to that structure. When appending the object, the translator MUST:

* Create a Node Identification Extension object containing an IP Address Sub-Object. The IP Address Sub-Object MUST contain the original source IPv6 address of the ICMPv6 message being translated.

* Append a Node Identification Extension object to the Extension Structure.

* Update the checksum field of the Extension Header accordingly.

* If the resulting packet size exceeds the minimum IPv6 MTU: truncate the embedded invoking packet by removing the trailing 24 octets (to accommodate for 24 octets of the extension object) and update the length field of the ICMP message

12. References

12.1. Normative References

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <https://www.rfc-editor.org/info/rfc4884>.

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, DOI 10.17487/RFC6052, October 2010, <https://www.rfc-editor.org/info/rfc6052>.

[RFC7915] Bao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm", RFC 7915, DOI 10.17487/RFC7915, June 2016, <https://www.rfc-editor.org/info/rfc7915>.

[RFC6791] Li, X., Bao, C., Wing, D., Vaithianathan, R., and G. Huston, "Stateless Source Address Mapping for ICMPv6 Packets", RFC 6791, DOI 10.17487/RFC6791, November 2012, <https://www.rfc-editor.org/info/rfc6791>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[I-D.ietf-intarea-extended-icmp-nodeid] Fenner, B. and R. Thomas, "Adding Extensions to ICMP Errors for Originating Node Identification", Work in Progress, Internet-Draft, draft-ietf-intarea-extended-icmp-nodeid-04, 19 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-intarea-extended-icmp-nodeid-04>.

12.2. Informative References

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <https://www.rfc-editor.org/info/rfc6146>.

[RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <https://www.rfc-editor.org/info/rfc5837>.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <https://www.rfc-editor.org/info/rfc6877>.

[RFC7755] Anderson, T., "SIIT-DC: Stateless IP/ICMP Translation for IPv6 Data Center Environments", RFC 7755, DOI 10.17487/RFC7755, February 2016, <https://www.rfc-editor.org/info/rfc7755>.

Acknowledgements

This document is the result of discussions with Thomas Jensen. The authors would like to thank Ondナ册j Caletka, Lorenzo Colitti, Darren Dukes, Bill Fenner, Tobias Fiebig, and Jordi Palet Martテュnez for their feedback, comments and guidance. The authors would like to particularly thank Tore Anderson for pointing out the existence and relevance of [RFC6791].

Authors' Addresses

David 'equinox' Lamparter
NetDEF, Inc.
04229 Leipzig
Germany
Email: equinox@diac24.net, equinox@opensourcerouting.org

Jen Linkova
Google
1 Darling Island Rd
Pyrmont NSW 2009
Australia
Email: furry13@gmail.com, furry@google.com