IPv6 operations
Internet-Draft
Updates: 6877, 8585 (if approved)
Intended status: Standards Track
Expires: 21 April 2026

L. Colitti
J. Linkova
Google
T. Jensen
Cloudflare

18 October 2025

464XLAT Customer-side Translator (CLAT): Node Behavior and Recommendations
draft-ietf-v6ops-claton-10

Abstract

464XLAT defines an architecture for providing IPv4 connectivity across an IPv6-only network. The solution involves two functional elements: provider-side translator (PLAT) and customer-side translator (CLAT). This document updates the 464XLAT specification (RFC6877) by further defining CLAT node behavior and IPv6 Customer Edge Routers to Support IPv4-as-a-Service by providing recommendations for the node developers on enabling and disabling CLAT.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on 21 April 2026.

Copyright Notice

Table of Contents

1. Introduction

464XLAT is widely deployed in 3GPP networks (as described in Section 4.2 of [RFC6877]) where User Equipment (UE) such as mobile phones and customer equipment (CE) routers perform the customer-side translation (CLAT) function, providing an IPv4 address and default route for applications and tethered devices. Enabling 464XLAT allowed mobile operators to transition UE to IPv6-only mode, where UE cellular interfaces have only IPv6 addresses, and no forwardable IPv4 addresses.

IPv6-only hosts used to be rather uncommon outside of mobile networks and datacenters. Even if a network offers provider-side translator (PLAT) functionality in the form of NAT64 [RFC6146], hosts (desktops, laptops, etc.) still needed the network to provide IPv4 connectivity,

as otherwise applications which require IPv4 would fail. However, as more and more operating systems outside of the 3GPP world support CLAT, it becomes possible to migrate those devices to IPv6-only mode, while still providing IPv4-as-a-Service via 464XLAT. Networks such as public Wi-Fi, enterprise networks, or even home networks can deploy 464XLAT as described in Section 4.2 of [RFC6877]:

* PLAT functions are performed by NAT64.

* CLAT is performed by the host itself.

In another variation of the 464XLAT deployment (Section 4.1 of [RFC6877]) a CE router is connected to an IPv6-only network and provides CLAT functions for IPv4-enabled downstream devices. [RFC8585] specifies 464XLAT support requirements for such devices.

While Section 6 of [RFC6877] discusses implementation considerations for the 464XLAT architecture, there is a need for more detailed guidance for CLAT implementations. The increase in IPv6-only deployments has provided valuable operational experience, which has revealed gaps in existing CLAT requirements. This document addresses these gaps by specifying normative behavior and providing recommendations for implementing CLAT functions. For example, it provides guidance on how CLAT nodes (such as a host or a CE router) should enable CLAT when connecting to an IPv6-only network and how they should react to network changes to minimize negative impact on user traffic. This document compliments and updates [RFC6877] and [RFC8585].

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

This document reuses most of the terminology from Section 2 of [RFC6877] and additionally defines the following terms:

* CLAT Node: a node (a host or a router) which performs CLAT functions by running one or multiple CLAT instances (e.g., one CLAT instance per interface).

* IPv4-only application: An application which requires the presence of an IPv4 address and/or IPv4 default route to operate. Examples include, but are not limited to, applications using IPv4 address literals or opening IPv4-only sockets.

* IPv6-only network: A network that does not assign IPv4 addresses to hosts and facilitates connectivity to IPv4-only destinations using NAT64 [RFC6146]. In this document, the term "IPv6-only network" specifically refers to networks that provide NAT64 as it is required by the 464XLAT architecture [RFC6877].

* Native IPv4 (such as in ’native IPv4 connectivity’ or ’native IPv4 default gateway’): IPv4 connectivity provided by the network without using any form of IPv4-as-a-Service or IP address family translation mechanisms (such as 464XLAT).

* ULA: Unique Local Addresses [RFC4193].

4. Enabling CLAT

For performance and security reasons CLAT SHOULD NOT be enabled (unless explicitly configured otherwise) for an interface if the node has native IPv4 connectivity over that interface (see Section 5 for more details). Therefore, recommendations provided in this section are only applicable to the IPv6-only interfaces of a given node (the node has no native IPv4 connectivity via that interface).

To enable CLAT, an IPv6-only node needs to discover the PLAT-side translation IPv6 prefix, also known as the NAT64 prefix (see Section 6.3 of [RFC6877]). The PREF64 Router Advertisement (RA) option [RFC8781] provides that information and can be used as a strong indication that the network supports PLAT (NAT64) functionality. Therefore an IPv6-only node SHOULD enable CLAT as soon as a Router Advertisement containing a PREF64 option is received.

Discussion: this document does not require that a node MUST enable CLAT upon receiving an RA because enabling CLAT by default may be undesirable in certain scenarios. First, in a controlled deployment environment, an administrator might explicitly prefer that the node not have CLAT enabled. Second, CLAT is an IPv6 transition technology whose operational utility will diminish as IPv4 dependencies become less common. Even where CLAT is supported, default enablement may eventually become unnecessary.

A node may have multiple IPv6-only interfaces (for example, a mobile phone can be connected to an IPv6-only Wi-Fi network and to an IPv6-only mobile network). In that case the node SHOULD run an

independent, dedicated CLAT instance on each interface connected to a network equipped with PLAT. Consequently, each CLAT instance SHOULD install a separate default IPv4 route on each CLAT-enabled interface.

Per Section 3.1 of [RFC9872], if RAs received by the node do not contain a PREF64 option, the node is allowed use other mechanisms to detect the PLAT presence and obtain the NAT64 prefix (such as [RFC7050]). When discovering the NAT64 prefix using the mechanism defined in [RFC7050], the node MUST follow recommendations provided in [RFC8880]. Specifically, the node MUST send the query to the DNS servers for the specific network interface per Section 7.1 of [RFC8880]. In particular, queries for AAAA resource records of "ipv4only.arpa." MUST be sent to the DNS resolvers provided through the specific network interface, not to any resolvers configured manually or otherwise.

Any delay in enabling CLAT on an IPv6-only node would be impactful for IPv4-only applications, as such applications cannot benefit from 464XLAT until CLAT is operational. Therefore it is desirable that the node enables CLAT as soon as network support for PLAT is detected while native IPv4 connectivity is not yet detected. The node SHOULD enable CLAT after discovering the NAT64 prefix, unless by that time the node has already obtained a non-link-local IPv4 address. The node SHOULD NOT wait for an explicit (DHCP Option 108) or an implicit (DHCP timeouts) indication that native IPv4 connectivity is not available. However, to mitigate attacks described in Section 7 of [RFC7050] the node MAY delay enabling CLAT if the NAT64 prefix was discovered via DNS [RFC7050] only. Such a delay minimizes the chance of temporary control of traffic by an attacker at the expense of delayed IPv4 connectivity through CLAT. The length of this optional delay is implementation specific. If native IPv4 connectivity becomes available later for a given interface, the node SHOULD disable CLAT for that interface (unless explicitly configured to keep it running) as discussed in the following section.

If the node supports multiple IPv4 continuity solutions, the node MUST follow recommendations from Section 4 of [RFC7335] to avoid IPv4 address space conflicts.

5. Disabling CLAT

It is possible that after a CLAT instance has started, native IPv4 connectivity becomes available (e.g., an IPv4 address received via DHCP). Unless explicitly configured otherwise, the node SHOULD disable CLAT immediately upon obtaining a native IPv4 default gateway.

While disabling CLAT is impactful for all applications and traffic flows already utilizing CLAT, disabling CLAT is recommended not only from a performance perspective, but also from a security point of view. Section 11 discusses this aspect in more details.

Native IPv4 connectivity, which subsequently causes CLAT to be disabled, might not be intended by the administrator. It could instead be the result of a network misconfiguration (such as accidentally enabling IPv4) or an attack (such as a rogue DHCPv4 server). In such cases it might be useful for an administrator to receive signals when CLAT turns on or off as well as changes to network-received configuration. Therefore the node SHOULD support logging the reason for disabling CLAT and any other changes to CLAT configuration or network signals CLAT is acting on to support administrator debugging and auditing. As logging is mostly beneficial in managed environments, the logging behaviour SHOULD be configurable. The logging SHOULD be disabled by default to avoid performance impacts when the likelihood of anyone consuming the logs is low.

There are some corner cases when an administrator might prefer the node to use CLAT even if native IPv4 connectivity is available (e.g., for performance reasons, if IPv4-as-a-Service performs better than native IPv4). This behaviour might be desirable for devices which do not move between networks, such as servers or workstations, where the administrator might want to have CLAT enabled unconditionally. However for the reasons described above such behaviour MUST be explicitly enabled by the administrator via configuration and MUST NOT be a default behaviour, especially for unmanaged nodes.

6. CLAT Addresses Considerations

6.1. CLAT IPv4 Addresses

A CLAT instance provides an IPv4 address and the default IPv4 route to the local node’s network stack. However, the node can also extend the network downstream and provides IPv4 network connectivity to other connected systems (e.g. tethering). Those systems can be physical (e.g., various clients connected to a CE router), or logical (e.g., virtual systems running on a node, while the host system acts as a router and performs CLAT). In all those cases, systems behind a CLAT node usually use [RFC1918] addresses. A CLAT instance can either translate these IPv4 addresses statelessly into IPv6 addresses using a dedicated IPv6 prefix per [RFC6052], or perform a stateful NAT44 between these IPv4 addresses and a dedicated CLAT IPv4 address, which is then statelessly translated to a single dedicated IPv6 address. In both cases, even with stateful NAT44, translation between IPv4 and IPv6 is stateless.

This section only applies to a single dedicated IPv4 address which a CLAT instance uses for providing network connectivity only to local applications or using stateful NAT44 when extending network connectivity downstream.

A CLAT instance needs a single IPv4 CLAT address and a single CLAT-only IPv6 address (which is distinct from the one or more IPv6 addresses used by the node running CLAT for its own native IPv6 connectivity, see Section 6.2.1). A node providing CLAT functions to local applications SHOULD use IPv4 addresses from the dedicated 192.0.0.0/29 range [RFC7335], reserved for IPv4 continuity solutions including but not limited to 464XLAT. If the node runs multiple CLAT instances, the node SHOULD use different local IPv4 addresses for each CLAT instance. This approach limits the number of CLAT instances per node to 8, which seems to be more than sufficient for typical configurations at the time this document was published. If in the future some deployment scenarios require more than 8 CLAT instances per node, a new IPv4 range will be requested from IANA.

The node MUST NOT send packets on wire from the local CLAT addresses.

The node SHOULD use 255.255.255.255 as a netmask for the CLAT address. That allows all 8 addresses from 192.0.0.0/29 to be used, if needed, since this means that each address is treated as being its own subnet, rather than being part of a subnet delineated by the prefix.

It should be noted that 192.0.0.0/29 is shared between multiple IPv4 continuity solutions such as 464XLAT and DS-Lite (see [RFC7335]). For example, Section 10 of [RFC6333] reserves 192.0.0.1 for the Dual-Stack Lite default router. However, as per Section 4 of [RFC7335], the node "MUST NOT enable two active IPv4 continuity solutions simultaneously in a way that would cause a node to have overlapping 192.0.0.0/29 address space" [RFC7335]. Therefore, as long as the node is not using DS-Lite, it MAY use 192.0.0.0/29 for CLAT.

6.2. CLAT IPv6 Addresses

6.2.1. Obtaining CLAT IPv6 Addresses

Section 6.3 of [RFC6877] recommends that a CLAT instance acquires a dedicated /64 for translating between IPv4 and IPv6, and only uses a single interface IPv6 address if a dedicated prefix is not available via DHCPv6-PD [RFC8415]. However, deployments where each node can obtain a dedicated /64 just for CLAT are rather uncommon, especially in environments such as enterprise networks and Wi-Fi hotspots. Quite often the CLAT instance uses a single IPv6 address as a source for all IPv4 traffic translated by CLAT. In particular, if the CLAT

only provides the IPv4 connectivity to applications local to the node (or if the node chooses to perform stateful NAT44) the CLAT instance only needs a single CLAT IPv6 address, so obtaining a /64 is wasteful. For instance, a home network that gets a /60 from its ISP can only connect up to 15 CLAT devices before it runs out of available prefixes. Even if the node extends IPv4 connectivity downstream, the CLAT instance can first perform stateful NAT44 to translate all IPv4 addresses used by downstream devices to a single IPv4 address, and then perform stateless CLAT.

This document updates [RFC6877] by relaxing the requirement to acquire a dedicated /64 prefix for the purpose of sending and receiving statelessly translated packets. The following recommendations are made instead:

* If the node is extending IPv4 connectivity downstream, the node MUST either:

  - Obtain a dedicated prefix that satisfies the requirements in section 2.2 of [RFC6052] (e.g., via DHCPv6-PD), and statelessly translate downstream IPv4 addresses to IPv6 addresses in this prefix.

  - Perform stateful NAT44 from the downstream IPv4 addresses to a single IPv4 address and then perform stateless translation of that single IPv4 address to a single dedicated IPv6 address.

* If the CLAT instance only uses a single IPv4 address (including scenarios where the node is performing stateful NAT44 to that address), the instance SHOULD NOT obtain a dedicated prefix for the purpose of sending and receiving statelessly translated packets.

If the CLAT instance does not obtain a dedicated IPv6 prefix, the instance MUST obtain a dedicated IPv6 address used exclusively for CLAT functions. A dedicated address is needed because applications running on a CLAT node can use an IPv4 socket and an IPv6 socket to produce an IPv4 packet and an IPv6 packet that after translation are identical, and the CLAT has no way to know whether to translate those replies and pass them back to the IPv4 socket or pass them back to the IPv6 socket as is. For example, these two packets sent by the node will be identical after translation:

* An IPv4 UDP packet from 192.0.0.4:12345 to 203.0.113.8:53

* An IPv6 UDP packet from the CLAT IPv6 address, port 12345 to [64:ff9b::203.0.113.8]:53

Similarly, responses (such as ICMP errors) to IPv4 and IPv6 packets may be identical when they arrive at the CLAT for translation back to IPv4. For example:

* An ICMPv6 Echo Reply packet from 64:ff9b::203.0.113.1 can be a response to either an IPv6 ICMP Echo Request to 64:ff9b::203.0.113.1, or an IPv4 ping to 203.0.113.1, translated by a CLAT instance.

* An ICMPv6 error packet from a global IPv6 address (not belonging to the NAT64 prefix) might be a response to either native IPv6 traffic from the host, or CLAT traffic (see [I-D.ietf-v6ops-icmpext-xlat-v6only-source] for more details).

Therefore, in the absence of dedicated IPv6 addresses, the architecture would need additional stateful elements on the client side which are not part of 464XLAT as defined in [RFC6877]. Discussion of such architectures is beyond the scope of this document.

If the dedicated CLAT address is obtained via Stateless Address Autoconfiguration (SLAAC, [RFC4862]), the CLAT instance SHOULD ensure that the address is checksum-neutral. This means the CLAT IPv6 address has the same complement checksum as the local CLAT IPv4 address. See Section 4.1 of [RFC6052]. This means that the local IPv4 address needs to be assigned/known before the IPv6 address is configured. Using a checksum-neutral CLAT address provides the following benefits:

* Better performance as CLAT doesn’t need to recalculate the checksum.

* If a protocol uses the standard IP checksum, CLAT doesn’t need to recalculate the checksum. That improves the chances of the protocol working via CLAT even if CLAT is not aware of the protocol’s semantics.

To protect user privacy and prevent user tracking through CLAT addresses, the node SHOULD generate a different interface identifier for the CLAT address when connecting to different networks, even if the NAT64 prefix and the local IPv4 CLAT address do not change. In particular, the node SHOULD generate a random CLAT address every time the network attachment changes to another network.

6.2.2. CLAT vs non-CLAT IPv6 Addresses Behaviour

Reports from the field indicate that some CLAT implementations exhibit different behavior for their CLAT IPv6 addresses compared to native IPv6 addresses. While this approach may simplify implementation, it often leads to a degraded user experience, as described below: Therefore the node MUST treat its CLAT IPv6 addresses as any other IPv6 address and comply with [RFC4861] and [RFC4862]. In particular:

* The node MUST perform Duplicate Address Detection (DAD) for each dedicated CLAT address (Section 5.4 of [RFC4862]).
  - Justification: performing DAD minimizes loss of connectivity in the unlikely event of address collision. Additionally, real world deployment experience shows that network infrastructure devices mandate a DAD packet from the client before enabling network access.
* The node MUST process received unicast Neighbor Solicitations (NSes) as well as multicast ones sent to the solicited-node multicast address [RFC4861] for the node CLAT addresses.
  - Justification: If a node doesn’t respond to unicast NSes, anytime the first-hop router gets a packet for the CLAT address and its Neighbor Cache entry is ’STALE’ (Section 7.3.2 of [RFC4861]), the Neighbor Unreachability Detection process (Section 7.3.3 of [RFC4861]) will delete that CLAT address’s cache entry. This forces the address resolution process to restart from scratch. Until resolution finishes, traffic for the CLAT address might drop, leading to a degraded user experience, especially for applications sensitive to jitter and packet loss.
* If the node supports Gratuitous Neighbor Advertisements [RFC9131], the node SHOULD send them for the CLAT addresses.
  - Justification: not following this recommendation leads to user-visible packet loss when the CLAT instance starts receiving traffic after period of inactivity, or when connected to the network for the first time. The problem is discussed in Section 2 of [RFC9131].
* The node which has the address registration using DHCPv6 [RFC9686] enabled MUST register the CLAT addresses if the network supports the registration.

- Justification: not registering CLAT addresses reduces traffic visibility for network operators, which complicates troubleshooting and forensics, as discussed in [RFC9686].

7. CLAT and Multiple Prefixes per Interface

IPv6 multihoming, particularly when multiple routers on the same link advertise different prefixes in Prefix Information Options (PIOs), presents a complex and not yet fully resolved challenge.

When routers on a given link are managed independently (e.g., by different ISPs), the resulting set of configuration parameters received by a host can be difficult to utilize without creating a complex and fragile state machine. For example, if router_A advertises a PIO with prefix_A and PREF64_A, while router_B advertises a PIO with prefix_B and a PREF64_B, it is crucial that the CLAT bundles the information received from each router. A CLAT instance must use PREF64_A and generate a CLAT address from prefix_A, sending translated packets to router_A. Alternatively, it must use PREF64_B, generate an address from prefix_B, and send translated packets to router_B. Mixing configuration information from different routers (e.g., generating a CLAT address from prefix_A but using PREF64_B for translation) can lead to packet loss. For example, if packets with source addresses from prefix_A are sent to router_B, that router (or the uplink network) might drop the packets according to BCP 38 [RFC2827]. Similarly, if the CLAT instance uses PREF64_A, advertised by router_A, but those packets are sent to router_B, that router might not be configured to translate packets for that prefix.

This document does not aim to define CLAT behavior for every possible multi-router/multi-prefix scenario. Instead, this section provides recommendations for common scenarios, leaving numerous corner cases out of scope.

This section assumes that a router is identified by its link-local address, used as a source address for RAs. For example, "detecting multiple routers" means that the node received RAs from multiple link-local addresses.

A node discovering multiple routers on the same interface advertising the _same PIOs and NAT64 prefix_, SHOULD only create one CLAT instance using one of the PIOs to form a CLAT address.

A node discovering multiple routers on the same interface signalling the _different PIOs and NAT64 prefixes_, MAY create one CLAT instance for each tuple of PIOs and NAT64 prefix (both PIO and NAT64 prefix in a given tuple MUST be advertised by the same router), or only a single CLAT instance using the NAT64 prefix discovered through the selected IPv6 default router and the address formed from a PIO advertised by that router.

When a node creates a single CLAT instance and must choose between multiple PIOs, the node SHOULD select a single PIO using the same algorithm as for choosing the source address for a destination within the selected NAT64 prefix ([RFC6724], updated by [I-D.ietf-6man-rfc6724-update]).

Discussion: This approach, leveraging the default source address selection algorithm (Section 5 of [RFC6724]), typically results in the policy table (rule 6) and longest prefix match (rule 8) being used for prefix selection. This ensures CLAT address selection aligns with default source address selection for native IPv6 flows, offering the following advantages:

* When using the well-known NAT64 prefix (64:ff9b::/96), non-ULA prefixes are preferred over ULA prefixes by default. This is beneficial as ULA source packets may not reach PLAT devices.

* For network-specific NAT64 prefixes within the known-local ULA range ([I-D.ietf-6man-rfc6724-update]), the ULA prefix is preferred. This can be advantageous in home and enterprise environments where administrators intend to perform NAT64 for specific source prefixes only.

* For network-specific NAT64 prefixes within the operator’s global non-ULA range, the longest prefix match selects the PIO, ensuring CLAT uses the operator’s source address for traffic to the operator’s PLAT in multi-prefix environments.

* In managed environments, operators can customize CLAT behavior by modifying the policy table if the default prefix selection is unsuitable.

Creating a single CLAT instance significantly simplifies the CLAT state machine. However, this approach may concentrate all traffic from that instance onto the same first-hop router and NAT64 device in some multihomed topologies. As traffic shifts from CLAT to native IPv6, this drawback becomes less significant and does not justify the added complexity of multiple instances.

7.1. Link Renumbering

As discussed above, a single CLAT instance per interface, using a single PIO, is typically sufficient, even if the link has multiple assigned subnets. However, PIO selection can significantly impact user experience during link renumbering.

[RFC8978] discusses various examples of "flash renumbering," where the IPv6 prefix assigned to the link changes without explicit host notification. [I-D.ietf-6man-slaac-renum] and [I-D.link-6man-gulla] discuss methods to mitigate the impact of flash renumbering. These methods generally rely on hosts with addresses from both old and new prefixes ceasing use of the old prefix and adopting the new prefix. For nodes running CLAT instances, this requires disabling instances using addresses from the old prefix and creating an instance using an address from the new prefix.

The CLAT node SHOULD use at least the following signals to detect link renumbering events:

* A prefix used to form the CLAT address becomes deprecated or invalid ([RFC4862]).

* The router (or routers) advertising the PIO used to form the CLAT address

  - has changed it state from being reachable or probably reachable to being unknown or suspect (i.e., it’s neighbor cache entry moved to the ’INCOMPLETE’ state or ceased to exist, see Section 6.3.6 of [RFC4861]).

  - ceased to be a router (see Section 7.3.3 of [RFC4861]).

Upon receiving a signal indicating a possible renumbering event, the node SHOULD disable the CLAT instance(s) affected by the renumbering, and create new instance(s). In case of implicit signals (provided by the Neighbor Unreachability Detection, [RFC4861], rather than by a Router Advertisement deprecating or invalidating a prefix), the node MAY send Router Solicitations to obtain the most up-to-date network configuration information. When sending Router Solicitations the node MUST follow recommendations specified in Section 6.3.7 of [RFC4861]. The node MAY react to a potential renumbering event in a "make-before-break" manner, when old instances are still running until all required information to enable new ones becomes available.

8. MTU Considerations

The IPv4 header is 20 bytes long (or longer if IP options are present), while the IPv6 header is 40 bytes. This means that when CLAT translates an IPv4 packet to IPv6, it usually adds 20 bytes to the packet size. However, when CLAT translates a fragmented IPv4 packet, then Fragment Header needs to be added to the resulting IPv6 packet (Section 4.1 of [RFC7915]). The length of IPv6 Fragment Extension header is 8 bytes (Section 4.5 of [RFC8200]). Therefore, to minimize undesirable IP fragmentation ([RFC8900]), the CLAT instance SHOULD present IPv4-only applications with an IPv4 MTU which is 28 bytes smaller than the IPv6 MTU of the interface the instance is running on.

9. Updates to RFC8585

This document makes the following changes to Section 3.2.1 of [RFC8585]:

OLD TEXT:

===

464XLAT requirements:

===

NEW TEXT:

===

464XLAT requirements:

464XLAT-0: The IPv6 Transition CE Router SHOULD follow recommendations provided in THIS DOCUMENT (note to the RFC Editor to add a reference to this document’s RFC number).

===

OLD TEXT:

===

464XLAT-4: The IPv6 Transition CE Router MUST implement [RFC7050] ("Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis") in order to discover the provider-side translator (PLAT) translation IPv4 and IPv6 prefix(es)/suffix(es).

===

NEW TEXT:

===

464XLAT-4: The IPv6 Transition CE Router MUST implement [RFC8781] ("Discovering PREF64 in Router Advertisements") and SHOULD implement [RFC7050] ("Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis") in order to discover the provider-side translator (PLAT) translation IPv4 and IPv6 prefix(es)/suffix(es).

===

OLD TEXT:

===

464XLAT-6: If the network provides several choices for the discovery/ learning of the NAT64 prefix, the priority to use one or the other MUST follow this order: 1) [RFC7225] and 2) [RFC7050].

The NAT64 prefix could be discovered by means of the method defined in [RFC7050] only if the service provider uses DNS64 [RFC6147]. It may be the case that the service provider does not use or does not trust DNS64 [RFC6147] because the DNS configuration at the CE (or hosts behind the CE) can be modified by the customer. In that case, the service provider may opt to configure the NAT64 prefix by means of the option defined in [RFC7225]. This can also be used if the service provider uses DNS64 [RFC6147].

===

NEW TEXT

===

464XLAT-6: If the network provides several choices for the discovery/ learning of the NAT64 prefix, the priority to use one or the other MUST follow this order: 1)[RFC7225] 2)[RFC8781] and 3)[RFC7050].

464XLAT-7: If the IPv6 Transition CE Router performs CLAT functions it SHOULD also include the PREF64 option containing the PLAT prefix in Router Advertisements ([RFC8781]) sent via the LAN interfaces. If the IPv6 Transition CE Router acts as a DHCP server it SHOULD enable DHCP Option 108 ([RFC8925]) processing. The router SHOULD have a configuration knob to disable DHCP Option 108 processing.

[RFC8781] allows the service provider to signal NAT64 prefix independently from DNS64 presence. At the same time the NAT64 prefix could be discovered by means of the method defined in [RFC7050] only if the service provider uses DNS64 [RFC6147]. It may be the case that the service provider does not use or does not trust DNS64 [RFC6147] because the DNS configuration at the CE (or hosts behind the CE) can be modified by the customer. In that case, the service provider may opt to configure the NAT64 prefix by means of the option defined in [RFC7225]. This can also be used if the service provider uses DNS64 [RFC6147].

===

10. Operational Considerations

There are no new operations or manageability requirements for mobile networks introduced by this document. For non-mobile envinronments, such as datacenters, public Wi-Fi, or enterprise networks the operational considerations are documented in [I-D.ietf-v6ops-6mops] .

11. Security Considerations

If a malicious actor spoofs PLAT presence signals (such as an RA with PREF64 option) or DNS responses for DNS-based NAT64 prefix detection [RFC7050], traffic of IPv4-only applications using CLAT can be affected:

* If there is no PLAT (NAT64) devices, traffic to NAT64 destinations would be dropped.

* If the attacker intercepts traffic for the NAT64 prefix (e.g., by providing the victim with a bogus NAT64 prefix and steering traffic for those destinations towards themselves), the attacker might be able to perform man-in-the-middle attacks (active attack on insecure traffic or hoarding secure traffic for "Harvest Now, Decrypt Later" attacks for non-PQC secure traffic, see Section 8 of [I-D.ietf-pquip-pqc-engineers]).

Rogue RAs can also disturb traffic to destinations that support both IPv4 and IPv6 by causing IPv6 through an attacker’s PLAT to be used instead of the legitimate network owner’s IPv4 path.

Using the PREF64 RA option to detect PLAT presence and the NAT64 prefix is less prone to such attacks than DNS-based detection ([RFC7050]), as the attacker needs to be on-link and be able to bypass layer-2 security features such as RA Guard (see Section 4 of [RFC9872] for more details). Therefore Section 4 recommends the PREF64 RA option as a preferred way to detect PLAT presence, in accordance with [RFC9872].

The attack vector described above is not specific to 464XLAT deployments: security implications of rogue RAs have been discussed and documented before (see [RFC6104]). To prevent such an attack, IPv6-enabled networks need to secure RAs, e.g., by deploying RA-Guard [RFC6105]. However, networks without explicit (intentional) IPv6 deployment are inherently IPv6-ignorant, and consequently might lack IPv6 security features. In such networks IPv6-enabled endpoints may be inadvertently exposed to link-local IPv6 connectivity. This unintended exposure can facilitate PLAT presence signal falsification, as described above. This document mitigates this risk by recommending that endpoints disable CLAT when the network provides non-link-local IPv4 connectivity, as outlined in Section 5.

However, the recommended behaviour (disabling CLAT in the presence of native IPv4 connectivity) introduces another attack vector: a rogue DHCPv4 server. An attacker can provide the CLAT node with an IPv4 address and default gateway, causing the node to disable CLAT. Similarly to the spoofed PLAT presence case discussed above, a rogue DHCP server allows the attacker to implement:

* A denial-of-service attack (as the victim’s IPv4 traffic will be dropped by the network).

* A man-in-the-middle attack by acting as an IPv4 default gateway for the victim node.

It should be noted that such attacks are not specific to the CLAT scenario, and can occur in IPv4-only or dual-stack networks as well. Various well-known Layer 2 security techniques (such as DHCP snooping) are available and considered a best practice in IPv4-enabled deployments. To mitigate rogue DHCPv4 server attacks on CLAT nodes, network administrators can deploy DHCPv4-related security features even if the network is expected to operate in IPv6-only mode.

As discussed, disabling CLAT when native IPv4 connectivity is present helps mitigate RA-based attacks but enables DHCPv4-based attack vectors if the network lacks Layer 2 security features. The choice between disabling CLAT when native IPv4 connectivity is present and keeping it enabled is dictated by the threat model and risk

assessment. At the time of this document's publication, it is much more likely that an IPv4-only network lacks IPv6 security than an IPv6-only network not having IPv4 Layer 2 security features deployed. Therefore, this document prescribes that the node SHOULD disable CLAT if native IPv4 connectivity is present. The choice of SHOULD, rather than MUST, is determined by considerations of future compatibility: nodes operating in environments where rogue RAs are no more likely than rogue DHCP servers might choose to keep CLAT enabled, while still complying with this specification.

12. Privacy Considerations

This document does not introduce any new privacy considerations, but there are some existing privacy considerations not documented in [RFC6877]. In particular, if a CLAT instance utilizes the same CLAT IPv6 address for an extensive period of time or, much worse, uses the same CLAT address when connecting to different networks, eavesdroppers and information collectors could potentially correlate various network activity to the same node. To mitigate that risk and make address-based network-activity correlation more difficult, Section 6.2.1 recommends that the node SHOULD generate a different interface identifier for the CLAT address when connecting to different networks.

It should be noted that the node's CLAT IPv6 address is only used (and visible to observers) when the traffic is carried from the CLAT node to the PLAT. In the vast majority of the cases it means that address is never visible outside of the network boundary, so to perform address-based network-activity correlation the observer needs to be located in the same network as the CLAT node, or the PLAT needs to reside outside of the administrative domain, such as a public NAT64 service.

13. IANA Considerations

This memo does not introduce any requests to IANA.

14. References

14.1. Normative References

[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>.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <https://www.rfc-editor.org/info/rfc4193>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[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>.

[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>.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <https://www.rfc-editor.org/info/rfc6147>.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, DOI 10.17487/RFC6333, August 2011, <https://www.rfc-editor.org/info/rfc6333>.

[RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <https://www.rfc-editor.org/info/rfc6724>.

[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>.

[RFC7050] Savolainen, T., Korhonen, J., and D. Wing, "Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis", RFC 7050, DOI 10.17487/RFC7050, November 2013, <https://www.rfc-editor.org/info/rfc7050>.

[RFC7335] Byrne, C., "IPv4 Service Continuity Prefix", RFC 7335, DOI 10.17487/RFC7335, August 2014, <https://www.rfc-editor.org/info/rfc7335>.

[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>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8880] Cheshire, S. and D. Schinazi, "Special Use Domain Name 'ipv4only.arpa'", RFC 8880, DOI 10.17487/RFC8880, August 2020, <https://www.rfc-editor.org/info/rfc8880>.

[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>.

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

[RFC8585] Palet Martinez, J., Liu, H. M.-H., and M. Kawashima, "Requirements for IPv6 Customer Edge Routers to Support IPv4-as-a-Service", RFC 8585, DOI 10.17487/RFC8585, May 2019, <https://www.rfc-editor.org/info/rfc8585>.

[RFC8781] Colitti, L. and J. Linkova, "Discovering PREF64 in Router Advertisements", RFC 8781, DOI 10.17487/RFC8781, April 2020, <https://www.rfc-editor.org/info/rfc8781>.

[RFC8925] Colitti, L., Linkova, J., Richardson, M., and T. Mrugalski, "IPv6-Only Preferred Option for DHCPv4", RFC 8925, DOI 10.17487/RFC8925, October 2020, <https://www.rfc-editor.org/info/rfc8925>.

[RFC9131] Linkova, J., "Gratuitous Neighbor Discovery: Creating Neighbor Cache Entries on First-Hop Routers", RFC 9131, DOI 10.17487/RFC9131, October 2021, <https://www.rfc-editor.org/info/rfc9131>.

[RFC9686] Kumari, W., Krishnan, S., Asati, R., Colitti, L., Linkova, J., and S. Jiang, "Registering Self-Generated IPv6 Addresses Using DHCPv6", RFC 9686, DOI 10.17487/RFC9686, December 2024, <https://www.rfc-editor.org/info/rfc9686>.

[RFC9872] Buraglio, N., Jensen, T., and J. Linkova, "Recommendations for Discovering IPv6 Prefix Used for IPv6 Address Synthesis", RFC 9872, DOI 10.17487/RFC9872, September 2025, <https://www.rfc-editor.org/info/rfc9872>.

[I-D.ietf-6man-rfc6724-update] Buraglio, N., Chown, T., and J. Duncan, "Prioritizing known-local IPv6 ULAs through address selection policy", Work in Progress, Internet-Draft, draft-ietf-6man-rfc6724-update-25, 11 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-6man-rfc6724-update-25>.

14.2. Informative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. J., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <https://www.rfc-editor.org/info/rfc1918>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <https://www.rfc-editor.org/info/rfc2827>.

[RFC6104] Chown, T. and S. Venaas, "Rogue IPv6 Router Advertisement Problem Statement", RFC 6104, DOI 10.17487/RFC6104, February 2011, <https://www.rfc-editor.org/info/rfc6104>.

[RFC6105] Levy-Abegnoli, E., Van de Velde, G., Popoviciu, C., and J. Mohacsi, "IPv6 Router Advertisement Guard", RFC 6105, DOI 10.17487/RFC6105, February 2011, <https://www.rfc-editor.org/info/rfc6105>.

[RFC7225] Boucadair, M., "Discovering NAT64 IPv6 Prefixes Using the Port Control Protocol (PCP)", RFC 7225, DOI 10.17487/RFC7225, May 2014, <https://www.rfc-editor.org/info/rfc7225>.

[RFC8900] Bonica, R., Baker, F., Huston, G., Hinden, R., Troan, O., and F. Gont, "IP Fragmentation Considered Fragile", BCP 230, RFC 8900, DOI 10.17487/RFC8900, September 2020, <https://www.rfc-editor.org/info/rfc8900>.

[RFC8978] Gont, F., 貼or転, J., and R. Patterson, "Reaction of IPv6 Stateless Address Autoconfiguration (SLAAC) to Flash-Renumbering Events", RFC 8978, DOI 10.17487/RFC8978, March 2021, <https://www.rfc-editor.org/info/rfc8978>.

[I-D.ietf-pquip-pqc-engineers] Banerjee, A., Reddy.K, T., Schoinianakis, D., Hollebeek, T., and M. Ounsworth, "Post-Quantum Cryptography for Engineers", Work in Progress, Internet-Draft, draft-ietf-pquip-pqc-engineers-14, 25 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-pquip-pqc-engineers-14>.

[I-D.ietf-6man-slaac-renum] Gont, F., Zorz, J., Patterson, R., and J. Linkova, "Improving the Robustness of Stateless Address Autoconfiguration (SLAAC) to Flash Renumbering Events", Work in Progress, Internet-Draft, draft-ietf-6man-slaac-renum-10, 3 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-6man-slaac-renum-10>.

[I-D.ietf-v6ops-icmpext-xlat-v6only-source] Lamparter, D. and J. Linkova, "Using Dummy IPv4 Address and Node Identification Extensions for IP/ICMP translators (XLATs)", Work in Progress, Internet-Draft, draft-ietf-v6ops-icmpext-xlat-v6only-source-00, 4 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-v6ops-icmpext-xlat-v6only-source-00>.

[I-D.ietf-v6ops-6mops] Buraglio, N., Caletka, O., and J. Linkova, "IPv6-Mostly Networks: Deployment and Operations Considerations", Work in Progress, Internet-Draft, draft-ietf-v6ops-6mops-02, 26 August 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-v6ops-6mops-02>.

[I-D.link-6man-gulla] Linkova, J., "Using Prefix-Specific Link-Local Addresses to Improve SLAAC Robustness", Work in Progress, Internet-Draft, draft-link-6man-gulla-01, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-link-6man-gulla-01>.

Appendix A. Enabling and Disabling CLAT: Flowchart

For illustrative purposes, the following diagram provides a high-level overview of the state machine for enabling and disabling CLAT on a node. It is not exhaustive of all corner cases or custom environment needs and does not override this document's normative requirements.

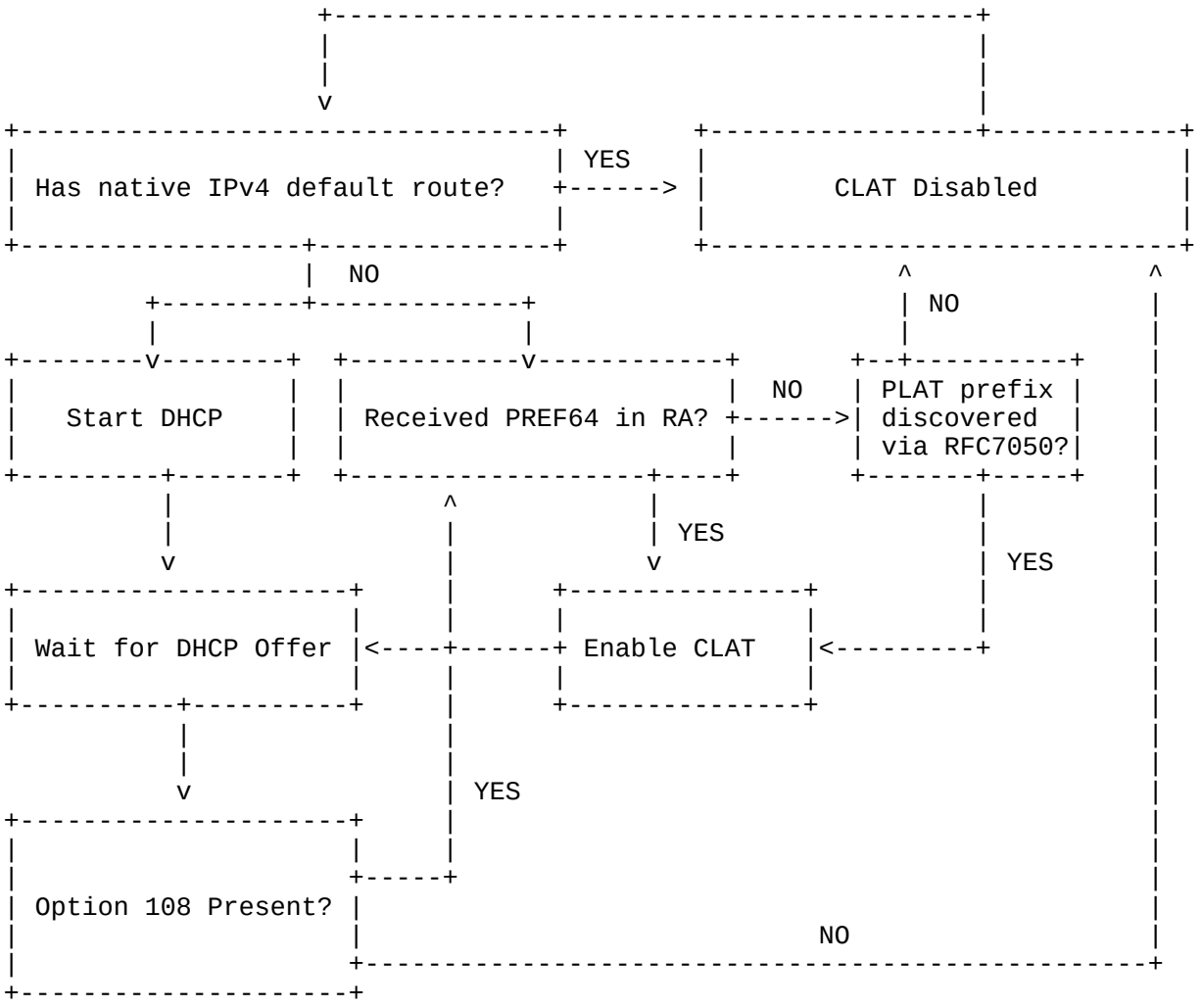

Figure 1: Enabling and Disabling CLAT Instance

Acknowledgements

Thanks to Mohamed Boucadair, Ondrej Caletka, Stuart Cheshire, Jeremy Duncan, Goetz Goerisch, Jason Healy, Ed Horley, KAWASHIMA Masanobu, Ted Lemon, George Michaelson, Jordi Palet, Nathan Sherrard, Dieter Siegmund, Philipp S. Tiesel, Eric Vyncke for the discussions, the input, and all contribution.

Authors' Addresses

Lorenzo Colitti
Google
Shibuya 3-21-3,
Japan
Email: lorenzo@google.com

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

Tommy Jensen
Cloudflare
Email: tojens.ietf@gmail.com