

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 3, 2020

P. Pfister
E. Vyncke
Cisco
T. Pauly
Apple Inc.
D. Schinazi
Google LLC
W. Shao
Cisco
January 31, 2020

Discovering Provisioning Domain Names and Data
draft-ietf-intarea-provisioning-domains-11

Abstract

Provisioning Domains (PvDs) are defined as consistent sets of network configuration information. This allows hosts to manage connections to multiple networks and interfaces simultaneously, such as when a home router provides connectivity through both a broadband and cellular network provider.

This document defines a mechanism for explicitly identifying PvDs through a Router Advertisement (RA) option. This RA option announces a PvD identifier, which hosts can compare to differentiate between PvDs. The option can directly carry some information about a PvD and can optionally point to additional PvD information that can be retrieved using HTTP over TLS.

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 August 3, 2020.

Copyright Notice

Copyright (c) 2020 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Specification of Requirements	4
2. Terminology	4
3. Provisioning Domain Identification using Router Advertisements	5
3.1. PvD ID Option for Router Advertisements	5
3.2. Router Behavior	8
3.3. Non-PvD-aware Host Behavior	9
3.4. PvD-aware Host Behavior	9
3.4.1. DHCPv6 configuration association	10
3.4.2. DHCPv4 configuration association	11
3.4.3. Connection Sharing by the Host	11
3.4.4. Usage of DNS Servers	12
4. Provisioning Domain Additional Information	13
4.1. Retrieving the PvD Additional Information	13
4.2. Operational Consideration to Providing the PvD Additional Information	16
4.3. PvD Additional Information Format	16
4.3.1. Example	18
4.4. Detecting misconfiguration and misuse	18
5. Operational Considerations	19
5.1. Exposing Extra RA Options to PvD-Aware Hosts	19
5.2. Different RAs for PvD-Aware and Non-PvD-Aware Hosts	19
5.3. Enabling Multi-homing for PvD-Aware Hosts	21
5.4. Providing Additional Information to PvD-Aware Hosts	22
6. Security Considerations	23
7. Privacy Considerations	24
8. IANA Considerations	25
8.1. New entry in the Well-Known URIs Registry	26
8.2. Additional Information PvD Keys Registry	26
8.3. PvD Option Flags Registry	26

8.4. PvD JSON Media Type Registration	27
9. Acknowledgments	28
10. References	28
10.1. Normative References	28
10.2. Informative References	30
Authors' Addresses	32

1. Introduction

Provisioning Domains (PvDs) are defined in [RFC7556] as consistent sets of network configuration information. This information includes properties that are traditionally associated with a single networking interface, such as source addresses, DNS configuration, proxy configuration, and gateway addresses.

Clients that are aware of PvDs can take advantage of multiple network interfaces simultaneously. This enables using two PvDs in parallel for separate connections or for multi-path transports.

While most PvDs today are discovered implicitly (such as by receiving information via Router Advertisements from a router on a network that a client host directly connects to), [RFC7556] also defines the notion of Explicit PvDs. IPsec Virtual Private Networks are considered Explicit PvDs, but Explicit PvDs can also be discovered via the local network router. Discovering Explicit PvDs allows two key advancements in managing multiple PvDs:

1. The ability to discover and use multiple PvDs on a single interface, such as when a local router can provide connectivity to two different Internet Service Providers.
2. The ability to associate additional information about PvDs to describe the properties of the network.

While [RFC7556] defines the concept of Explicit PvDs, it does not define the mechanism for discovering multiple Explicit PvDs on a single network and their additional information.

This document specifies a way to identify PvDs with Fully Qualified Domain Names (FQDN), called PvD IDs. Those identifiers are advertised in a new Router Advertisement (RA) [RFC4861] option called the PvD ID Router Advertisement option which, when present, associates the PvD ID with all the information present in the Router Advertisement as well as any configuration object, such as addresses, derived from it. The PVD ID Router Advertisement option may also contain a set of other RA options, along with an optional inner Router Advertisement message header. These options and optional

inner header are only visible to 'PvD-aware' hosts, allowing such hosts to have a specialized view of the network configuration.

Since PvD IDs are used to identify different ways to access the internet, multiple PvDs (with different PvD IDs) can be provisioned on a single host interface. Similarly, the same PvD ID could be used on different interfaces of a host in order to inform that those PvDs ultimately provide equivalent services.

This document also introduces a mechanism for hosts to retrieve optional additional information related to a specific PvD by means of an HTTP over TLS query using a URI derived from the PvD ID. The retrieved JSON object contains additional information that would typically be considered too large to be directly included in the Router Advertisement, but might be considered useful to the applications, or even sometimes users, when choosing which PvD should be used.

For example, if Alice has both a cellular network provider and a broadband provider in her home, her PvD-aware devices and applications would be aware of both available uplinks. These applications could fail-over between these networks, or run connections over both (potentially using multi-path transports). Applications could also select specific uplinks based on the properties of the network; for example, if the cellular network provides free high-quality video streaming, a video-streaming application could select that network while most of the other traffic on Alice's device uses the broadband provider.

1.1. Specification of Requirements

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.

2. Terminology

This document uses the following terminology:

Provisioning Domain (PvD): A set of network configuration information; for more information, see [RFC7556].

PvD ID: A Fully Qualified Domain Name (FQDN) used to identify a PvD.

Explicit PvD: A PvD uniquely identified with a PvD ID. For more information, see [RFC7556].

Implicit PvD: A PvD that, in the absence of a PvD ID, is identified by the host interface to which it is attached and the address of the advertising router. See also [RFC7556].

PvD-aware host: A host that supports the association of network configuration information into PvDs and the use of these PvDs as described in this document. Also named PvD-aware node in [RFC7556].

3. Provisioning Domain Identification using Router Advertisements

Explicit PvDs are identified by a PvD ID. The PvD ID is a Fully Qualified Domain Name (FQDN) that identifies the network operator. Network operators MUST use names that they own or manage to avoid naming conflicts. The same PvD ID MAY be used in several access networks when they ultimately provide identical services (e.g., in all home networks subscribed to the same service); else, the PvD ID MUST be different to follow Section 2.4 of [RFC7556].

3.1. PvD ID Option for Router Advertisements

This document introduces a Router Advertisement (RA) option called the PvD Option. It is used to convey the FQDN identifying a given PvD (see Figure 1), bind the PvD ID with configuration information received over DHCPv4 (see Section 3.4.2), enable the use of HTTP over TLS to retrieve the PvD Additional Information JSON object (see Section 4), as well as contain any other RA options which would otherwise be valid in the RA.

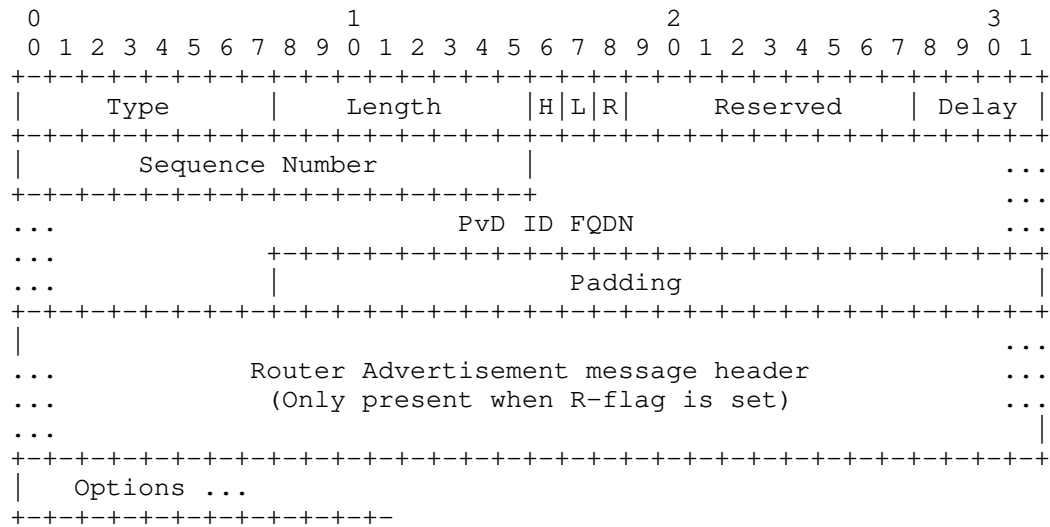


Figure 1: PvD ID Router Advertisements Option Format

Type: (8 bits) Set to 21.

Length: (8 bits) The length of the option in units of 8 octets, including the Type and Length fields, the Router Advertisement message header, if any, as well as the RA options that are included within the PvD Option.

H-flag: (1 bit) 'HTTP' flag stating whether some PvD Additional Information is made available through HTTP over TLS, as described in Section 4.

L-flag: (1 bit) 'Legacy' flag stating whether the PvD is associated with IPv4 information assigned using DHCPv4 (see Section 3.4.2).

R-flag: (1 bit) 'Router Advertisement' flag stating whether the PvD Option header is followed (right after padding to the next 64 bits boundary) by a Router Advertisement message header (see section 4.2 of [RFC4861]). The usage of the inner message header is described in Section 3.4.

Reserved: (13 bits) Reserved for later use. It MUST be set to zero by the sender and ignored by the receiver.

Delay: (4 bits) Unsigned integer used to delay HTTP GET queries from hosts by a randomized backoff (see Section 4.1). If the H-flag is not set, senders SHOULD set the delay to zero, and receivers SHOULD ignore the value.

Sequence Number: (16 bits) Sequence number for the PvD Additional Information, as described in Section 4. If the H-flag is not set, senders SHOULD set the Sequence Number to zero, and receivers SHOULD ignore the value.

PvD ID FQDN: The FQDN used as PvD ID encoded in DNS format, as described in Section 3.1 of [RFC1035]. Domain name compression described in Section 4.1.4 of [RFC1035] MUST NOT be used.

Padding: Zero or more padding octets to the next 8 octet boundary (see Section 4.6 of [RFC4861]). It MUST be set to zero by the sender, and ignored by the receiver.

RA message header: (16 octets) When the R-flag is set, a full Router Advertisement message header as specified in [RFC4861]. The sender MUST set the 'Type' to 134, the value for "Router Advertisement", and set the 'Code' to 0. Receivers MUST ignore both of these fields. The 'Checksum' MUST be set to 0 by the sender; non-zero checksums MUST be ignored by the receiver without causing the processing of the message to fail. All other fields are to be set and parsed as specified in [RFC4861] or any updating documents.

Options: Zero or more RA options that would otherwise be valid as part of the Router Advertisement main body, but are instead included in the PvD Option so as to be ignored by hosts that are not PvD-aware.

Figure 2 shows an example of a PvD Option with "example.org" as the PvD ID FQDN and including both a Recursive DNS Server (RDNSS) option and a prefix information option. It has a Sequence Number of 123, and indicates the presence of additional information that is expected to be fetched with a delay factor of 1.

0										1										2										3																			
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																		
Type: 21										Length: 12										1 0 0										Reserved										Delay:1									
Seq number: 123										7										e																													
x										a										m										p																			
l										e										3										o																			
r										g										0										0 (padding)																			
0 (padding)										0 (padding)										0 (padding)										0 (padding)																			
RDNSS option (RFC 8106) length: 5																														...																			
...																														...																			
...																																																	
Prefix Information Option (RFC 4861) length: 4																														...																			
...																																																	
...																																																	

Figure 2

3.2. Router Behavior

A router MAY send RAs containing one PvD Option, but MUST NOT include more than one PvD Option in each RA. The PvD Option MUST NOT contain further PvD Options.

The PvD Option MAY contain zero, one, or more RA options which would otherwise be valid as part of the same RA. Such options are processed by PvD-aware hosts, while ignored by other hosts as per section 4.2 of [RFC4861].

In order to provide multiple different PvDs, a router MUST send multiple RAs. RAs sent from different link-local source addresses establish distinct implicit PvDs, in the absence of a PvD Option. Explicit PvDs MAY share link-local source addresses with an Implicit PvD and any number of other Explicit PvDs.

In other words, different Explicit PvDs MAY be advertised with RAs using the same link-local source address; but different Implicit PvDs, advertised by different RAs, MUST use different link-local addresses because these Implicit PvDs are identified by the source addresses of the RAs. If a link-local address on the router is

changed, then any new RA will be interpreted as a different Implicit PvD by PvD-aware hosts.

As specified in [RFC4861] and [RFC6980], when the set of options causes the size of an advertisement to exceed the link MTU, multiple router advertisements MUST be sent to avoid fragmentation, each containing a subset of the options. In such cases, the PvD Option header (i.e., all fields except the 'Options' field) MUST be repeated in all the transmitted RAs. The options within the 'Options' field, MAY be transmitted only once, included in one of the transmitted PvD Options.

3.3. Non-PvD-aware Host Behavior

As the PvD Option has a new option code, non-PvD-aware hosts will simply ignore the PvD Option and all the options it contains (see section 4.2 of [RFC4861]. This ensures the backward compatibility required in Section 3.3 of [RFC7556]. This behavior allows for a mixed-mode network where a mix of PvD-aware and non-PvD-aware hosts coexist.

3.4. PvD-aware Host Behavior

Hosts MUST associate received RAs and included configuration information (e.g., Router Valid Lifetime, Prefix Information [RFC4861], Recursive DNS Server [RFC8106], Routing Information [RFC4191] options) with the Explicit PvD identified by the first PvD Option present in the received RA, if any, or with the Implicit PvD identified by the host interface and the source address of the received RA otherwise. If an RA message header is present both within the PvD Option and outside it, the header within the PvD Option takes precedence.

In case multiple PvD Options are found in a given RA, hosts MUST ignore all but the first PvD Option.

If a host receives PvD Options flags that it does not recognize (currently in the Reserved field), it MUST ignore these flags.

Similarly, hosts MUST associate all network configuration objects (e.g., default routers, addresses, more specific routes, DNS Recursive Resolvers) with the PvD associated with the RA that provisioned the object. For example, addresses that are generated using a received Prefix Information option (PIO) are associated with the PvD of the last received RA which included the given PIO.

PvD IDs MUST be compared in a case-insensitive manner as defined by [RFC4343]. For example, "pvd.example.com." or "PvD.Example.coM." would refer to the same PvD.

While performing PvD-specific operations such as resolving names, executing the default address selection algorithm [RFC6724] or executing the default router selection algorithm when forwarding packets ([RFC4861], [RFC4191] and [RFC8028]), hosts and applications MAY consider only the configuration associated with any non-empty subset of PvDs. For example, a host MAY associate a given process with a specific PvD, or a specific set of PvDs, while associating another process with another PvD. A PvD-aware application might also be able to select, on a per-connection basis, which PvDs should be used. In particular, constrained devices such as small battery operated devices (e.g., IoT), or devices with limited CPU or memory resources may purposefully use a single PvD while ignoring some received RAs containing different PvD IDs.

The way an application expresses its desire to use a given PvD, or a set of PvDs, or the way this selection is enforced, is out of the scope of this document. Useful insights about these considerations can be found in [I-D.kline-mif-mpvd-api-reqs].

3.4.1. DHCPv6 configuration association

When a host retrieves stateless configuration elements using DHCPv6 (e.g., DNS recursive resolvers or DNS domain search lists [RFC3646]), they MUST be associated with all the explicit and implicit PvDs received on the same interface and contained in a RA with the O-flag set [RFC4861].

When a host retrieves stateful assignments using DHCPv6, such assignments MUST be associated with the received PvD which was received with RAs with the M-flag set and including a matching PIO. A PIO is considered to match a DHCPv6 assignment when the IPv6 prefix from the PIO includes the assignment from DHCPv6. For example, if a PvD's associated PIO defines the prefix 2001:db8:cafe::/64, a DHCPv6 IA_NA message that assigns the address 2001:db8:cafe::1234:4567 would be considered to match.

In cases where an address would be assigned by DHCPv6 and no matching PvD could be found, hosts MAY associate the assigned address with any implicit PvD received on the same interface or to multiple implicit PvDs received on the same interface. This is intended to resolve backward compatibility issues with rare deployments choosing to assign addresses with DHCPv6 while not sending any matching PIO. Implementations are suggested to flag or log such scenarios as errors to help detect misconfigurations.

3.4.2. DHCPv4 configuration association

Associating DHCPv4 [RFC2131] configuration elements with Explicit PvDs allows hosts to treat a set of IPv4 and IPv6 configurations as a single PvD with shared properties. For example, consider a router that provides two different uplinks. One could be a broadband network that has data rate and streaming properties described in PvD additional information and that provides both IPv4 and IPv6 network access. The other could be a cellular network that provides only IPv6 network access, and uses NAT64 [RFC6146]. The broadband network can be represented by an Explicit PvD that points to the additional information, and also marks association with DHCPv4 information. The cellular network can be represented by a different Explicit PvD that is not associated with DHCPv4.

When a PvD-aware host retrieves configuration elements from DHCPv4, the information is associated either with a single Explicit PvD on that interface, or else with all Implicit PvDs on the same interface.

An Explicit PvD indicates its association with DHCPv4 information by setting the L-flag in the PvD RA Option. If there is exactly one Explicit PvD that sets this flag, hosts MUST associate the DHCPv4 information with that PvD. Multiple Explicit PvDs on the same interface marking this flag is a misconfiguration, and hosts SHOULD NOT associate the DHCPv4 information with any Explicit PvD in this case.

If no single Explicit PvD claims association with DHCPv4, the configuration elements coming from DHCPv4 MUST be associated with all Implicit PvDs identified by the interface on which the DHCPv4 transaction happened. This maintains existing host behavior.

3.4.3. Connection Sharing by the Host

The situation when a host shares connectivity from an upstream interface (e.g., cellular) to a downstream interface (e.g., Wi-Fi) is known as 'tethering'. Techniques such as ND-proxy [RFC4389], 64share [RFC7278] or prefix delegation (e.g., using DHCPv6-PD [RFC8415]) may be used for that purpose.

Whenever the RAs received from the upstream interface contain a PVD RA option, hosts that are sharing connectivity SHOULD include a PVD option within the RAs sent downstream with:

- o The same PVD-ID FQDN
- o The same H-flag, Delay and Sequence Number values

- o The L bit set whenever the host is sharing IPv4 connectivity received from the same upstream interface
- o The bits from the Reserved field set to 0

The values of the R-flag, Router Advertisement message header and Options field depend on whether the connectivity should be shared only with PvD-aware hosts or not (see Section 3.2). In particular, all options received within the upstream PvD Option and included in the downstream RA SHOULD be included in the downstream PvD Option.

3.4.4. Usage of DNS Servers

PvD-aware hosts can be provisioned with recursive DNS servers via RA options passed within an Explicit PvD, via RA options associated with an Implicit PvD, via DHCPv6 or DHCPv4, or from some other provisioning mechanism that creates an Explicit PvD (such as a VPN). In all of these cases, the recursive DNS server addresses SHOULD be associated with the corresponding PvD. Specifically, queries sent to a configured recursive DNS server SHOULD be sent from a local IP address that was provisioned for the PvD via RA or DHCP. Answers received from the DNS server SHOULD only be used on the same PvD.

PvD-aware applications will be able to select which PvD(s) to use for DNS resolution and connections, which allows them to effectively use multiple Explicit PvDs. In order to support non-PvD-aware applications, however, PvD-aware hosts SHOULD ensure that non-PvD-aware name resolution APIs like "getaddrinfo" only use resolvers from a single PvD for a given query. Handling DNS across PvDs is discussed in Section 5.2.1 of [RFC7556], and PvD APIs are discussed in Section 6 of [RFC7556].

Maintaining the correct usage of DNS within PvDs avoids various practical errors, such as:

- o A PvD associated with a VPN or otherwise private network may provide DNS answers that contain addresses inaccessible over another PvD. This includes the DNS queries to retrieve PvD additional information, which could otherwise send identifying information to the recursive DNS system (see Section 4.1).
- o A PvD that uses a NAT64 [RFC6146] and DNS64 [RFC6147] will synthesize IPv6 addresses in DNS answers that are not globally routable, and would be invalid on other PvDs. Conversely, an IPv4 address resolved via DNS on another PvD cannot be directly used on a NAT64 network.

4. Provisioning Domain Additional Information

Additional information about the network characteristics can be retrieved based on the PvD ID. This set of information is called PvD Additional Information, and is encoded as a JSON object [RFC8259]. This JSON object is restricted to the I-JSON profile, as defined in [RFC7493].

The purpose of this JSON object is to provide additional information to applications on a client host about the connectivity that is provided using a given interface and source address. It typically includes data that would be considered too large, or not critical enough, to be provided within an RA option. The information contained in this object MAY be used by the operating system, network libraries, applications, or users, in order to decide which set of PvDs should be used for which connection, as described in Section 3.4.

The additional information related to a PvD is specifically intended to be optional, and is targeted at optimizing or informing the behavior of user-facing hosts. This information can be extended to provide hints for host system behavior (such as captive portal or walled-garden PvD detection) or application behavior (describing application-specific services offered on a given PvD). This content may not be appropriate for light-weight Internet of Things (IoT) devices. IoT devices might need only a subset of the information, and would in some cases prefer a smaller representation like CBOR ([RFC7049]). Delivering a reduced version of the PvD Additional Information designed for such devices is not defined in this document.

4.1. Retrieving the PvD Additional Information

When the H-flag of the PvD Option is set, hosts MAY attempt to retrieve the PvD Additional Information associated with a given PvD by performing an HTTP over TLS [RFC2818] GET query to `https://<PvD-ID>/well-known/pvd`. Inversely, hosts MUST NOT do so whenever the H-flag is not set.

Recommendations for how to use TLS securely can be found in [RFC7525].

When a host retrieves the PvD Additional Information, it MUST verify that the TLS server certificate is valid for the performed request; specifically, that a DNS-ID [RFC6125] on the certificate is equal to the PvD ID expressed as an FQDN. This validation indicates that the owner of the FQDN authorizes its use with the prefix advertised by

the router. If this validation fails, hosts MUST close the connection and treat the PvD as if it has no Additional Information.

HTTP requests and responses for PvD additional information use the "application/pvd+json" media type (see Section 8). Clients SHOULD include this media type as an Accept header field in their GET requests, and servers MUST mark this media type as their Content-Type header field in responses.

Note that the DNS name resolution of the PvD ID, any connections made for certificate validation (such as OCSP [RFC6960]), and the HTTP request itself MUST be performed using the considered PvD. In other words, the name resolution, PKI checks, source address selection, as well as the next-hop router selection MUST be performed while using exclusively the set of configuration information attached with the PvD, as defined in Section 3.4. In some cases, it may therefore be necessary to wait for an address to be available for use (e.g., once the Duplicate Address Detection or DHCPv6 processes are complete) before initiating the HTTP over TLS query. In order to address privacy concerns around linkability of the PvD HTTP connection with future user-initiated connections, if the host has a temporary address per [RFC4941] in this PvD, then it SHOULD use a temporary address to fetch the PvD Additional Information and MAY deprecate the used temporary address and generate a new temporary address afterward.

If the HTTP status of the answer is greater than or equal to 400 the host MUST close its connection and consider that there is no additional PvD information. If the HTTP status of the answer is between 300 and 399, inclusive, it MUST follow the redirection(s). If the HTTP status of the answer is between 200 and 299, inclusive, the response is expected to be a single JSON object.

After retrieval of the PvD Additional Information, hosts MUST remember the last Sequence Number value received in an RA including the same PvD ID. Whenever a new RA for the same PvD is received with a different Sequence Number value, or whenever the expiry date for the additional information is reached, hosts MUST deprecate the additional information and stop using it.

Hosts retrieving a new PvD Additional Information object MUST check for the presence and validity of the mandatory fields specified in Section 4.3. A retrieved object including an expiration time that is already past or missing a mandatory element MUST be ignored.

In order to avoid synchronized queries toward the server hosting the PvD Additional Information when an object expires, object updates are delayed by a randomized backoff time.

- o When a host performs a JSON object update after it detected a change in the PvD Option Sequence Number, it MUST add a delay before sending the query. The target time for the delay is calculated as a random time between zero and $2^{**}(10 + \text{Delay})$ milliseconds, where 'Delay' corresponds to the 4-bit unsigned integer in the last received PvD Option.
- o When a host last retrieved a JSON object at time A that includes a expiry time B using the "expires" key, and the host is configured to keep the PvD information up to date, it MUST add some randomness into its calculation of the time to fetch the update. The target time for fetching the updated object is calculated as a uniformly random time in the interval $[(B-A)/2, B]$.

In the example Figure 2, the delay field value is 1, this means that the host calculates its delay by choosing a uniformly random time between 0 and $2^{**}(10 + 1)$ milliseconds, i.e., between 0 and 2048 milliseconds.

Since the 'Delay' value is directly within the PvD Option rather than the object itself, an operator may perform a push-based update by incrementing the Sequence Number value while changing the Delay value depending on the criticality of the update and its PvD Additional Information servers capacity.

In addition to adding a random delay when fetching Additional Information, hosts MUST enforce a minimum time between requesting Additional Information for a given PvD on the same network. This minimum time is RECOMMENDED to be 10 seconds, in order to avoid hosts causing a denial-of-service on the PvD server. Hosts also MUST limit the number of requests that are made to different PvD Additional Information servers on the same network within a short period of time. A RECOMMENDED value is to issue no more than five PvD Additional Information requests in total on a given network within 10 seconds. For more discussion, see Section 6.

The PvD Additional Information object includes a set of IPv6 prefixes (under the key "prefixes") which MUST be checked against all the Prefix Information Options advertised in the RA. If any of the prefixes included in any associated PIO is not covered by at least one of the listed prefixes, the associated PvD information MUST be considered to be a misconfiguration, and MUST NOT be used by the host. See Section 4.4 for more discussion on handling such misconfigurations.

If the request for PvD Additional Information fails due to a TLS certificate validation error, an HTTP error, or because the retrieved file does not contain valid PvD JSON, hosts MUST close any connection

used to fetch the PvD Additional Information, and MUST NOT request the information for that PvD ID again for the duration of the local network attachment. If a host detects 10 or more such failures to fetch PvD Additional Information, the local network is assumed to be misconfigured or under attack, and the host MUST NOT make any further requests for any PvD Additional Information, belonging to any PvD ID, for the duration of the local network attachment. For more discussion, see Section 6.

4.2. Operational Consideration to Providing the PvD Additional Information

Whenever the H-flag is set in the PvD Option, a valid PvD Additional Information object MUST be made available to all hosts receiving the RA by the network operator. In particular, when a captive portal is present, hosts MUST still be allowed to perform DNS, certificate validation, and HTTP over TLS operations related to the retrieval of the object, even before logging into the captive portal.

Routers SHOULD increment the PVD Option Sequence Number by one whenever a new PvD Additional Information object is available and should be retrieved by hosts. If the value exceeds what can be stored in the Sequence Number field, it MUST wrap back to zero.

The server providing the JSON files SHOULD also check whether the client address is contained by the prefixes listed in the additional information, and SHOULD return a 403 response code if there is no match.

4.3. PvD Additional Information Format

The PvD Additional Information is a JSON object.

The following table presents the mandatory keys which MUST be included in the object:

JSON key	Description	Type	Example
identifier	PvD ID FQDN	String	"pvd.example.com."
expires	Date after which this object is no longer valid	[RFC3339] Date	"2020-05-23T06:00:00Z"
prefixes	Array of IPv6 prefixes valid for this PvD	Array of strings	["2001:db8:1::/48", "2001:db8:4::/48"]

A retrieved object which does not include all three of these keys at the root of the JSON object MUST be ignored. All three keys need to be validated, otherwise the object MUST be ignored. The value stored for "identifier" MUST be matched against the PvD ID FQDN presented in the PvD RA option using the comparison mechanism described in Section 3.4. The value stored for "expires" MUST be a valid date in the future. If the PIO of the received RA is not covered by at least one of the "prefixes" key, the retrieved object SHOULD be ignored.

The following table presents some optional keys which MAY be included in the object.

JSON key	Description	Type	Example
dnsZones	DNS zones searchable and accessible	Array of strings	["example.com", "sub.example.com"]
noInternet	No Internet, set to "true" when the PvD is restricted.	Boolean	true

It is worth noting that the JSON format allows for extensions. Whenever an unknown key is encountered, it MUST be ignored along with its associated elements.

Private-use or experimental keys MAY be used in the JSON dictionary. In order to avoid such keys colliding with IANA registry keys, implementers or vendors defining private-use or experimental keys

MUST create sub-dictionaries. If a set of PvD Additional Information keys are defined by an organization that has a Formal URN Namespace [URN], the URN namespace SHOULD be used as the top-level JSON key for the sub-dictionary. For other private uses, the sub-dictionary key SHOULD follow the format of "vendor-*", where the "*" is replaced by the implementer's or vendor's identifier. For example, keys specific to the FooBar organization could use "vendor-foobar". If a host receives a sub-dictionary with an unknown key, the host MUST ignore the contents of the sub-dictionary.

4.3.1. Example

The following two examples show how the JSON keys defined in this document can be used:

```
{
  "identifier": "cafe.example.com.",
  "expires": "2020-05-23T06:00:00Z",
  "prefixes": ["2001:db8:1::/48", "2001:db8:4::/48"],
}

{
  "identifier": "company.foo.example.com.",
  "expires": "2020-05-23T06:00:00Z",
  "prefixes": ["2001:db8:1::/48", "2001:db8:4::/48"],
  "vendor-foo":
    {
      "private-key": "private-value",
    },
}
```

4.4. Detecting misconfiguration and misuse

Hosts MUST validate the TLS server certificate when retrieving PvD Additional Information, as detailed in Section 4.1.

Hosts MUST verify that all prefixes in all the RA PIOs are covered by a prefix from the PvD Additional Information. An adversarial router attempting to spoof the definition of an Explicit PvD, without the ability to modify the PvD Additional Information, would need to perform NAT66 in order to circumvent this check. Thus, this check cannot prevent all spoofing, but it can detect misconfiguration or mismatched routers that are not adding a NAT.

If NAT66 is being added in order to spoof PvD ownership, the HTTPS server for additional information can detect this misconfiguration. The HTTPS server SHOULD validate the source addresses of incoming connections (see Section 4.1). This check gives reasonable assurance

that neither NPTv6 [RFC6296] nor NAT66 were used and restricts the information to the valid network users. If the PvD does not provision IPv4 (it does not include the 'L' bit in the RA), the server cannot validate the source addresses of connections using IPv4. Thus, the PvD ID FQDN for such PvDs SHOULD NOT have a DNS A record.

5. Operational Considerations

This section describes some example use cases of PvDs. For the sake of simplicity, the RA messages will not be described in the usual ASCII art but rather in an indented list. Values in the PvD Option header that are not included in the example are assumed to be zero or false (such as the H-flag, Sequence Number, and Delay fields).

5.1. Exposing Extra RA Options to PvD-Aware Hosts

In this example, there is one RA message sent by the router. This message contains some options applicable to all hosts on the network, and also a PvD Option that also contains other options only visible to PvD-aware hosts.

- o RA Header: router lifetime = 6000
- o Prefix Information Option: length = 4, prefix = 2001:db8:cafe::/64
- o PvD Option header: length = 3 + 5 + 4, PvD ID FQDN = example.org., R-flag = 0 (actual length of the header with padding 24 bytes = 3 * 8 bytes)
 - * Recursive DNS Server: length = 5, addresses = [2001:db8:cafe::53, 2001:db8:f00d::53]
 - * Prefix Information Option: length = 4, prefix = 2001:db8:f00d::/64

Note that a PvD-aware host will receive two different prefixes, 2001:db8:cafe::/64 and 2001:db8:f00d::/64, both associated with the same PvD (identified by "example.org."). A non-PvD-aware host will only receive one prefix, 2001:db8:cafe::/64.

5.2. Different RAs for PvD-Aware and Non-PvD-Aware Hosts

It is expected that for some years, networks will have a mixed environment of PvD-aware hosts and non-PvD-aware hosts. If there is a need to give specific information to PvD-aware hosts only, then it is RECOMMENDED to send two RA messages, one for each class of hosts. This approach allows for two distinct sets of configuration

information to be sent in a way that will not disrupt non-PvD-aware hosts. It also lowers the risk that a single RA message will approach its MTU limit due to duplicated information.

If two RA messages are sent for this reason, they MUST be sent from two different link-local source addresses (Section 3.2). For example, here is the RA sent for non-PvD-aware hosts:

- o RA Header: router lifetime = 6000 (non-PvD-aware hosts will use this router as a default router)
- o Prefix Information Option: length = 4, prefix = 2001:db8:cafe::/64
- o Recursive DNS Server Option: length = 3, addresses = [2001:db8:cafe::53]
- o PvD Option header: length = 3 + 2, PvD ID FQDN = foo.example.org., R-flag = 1 (actual length of the header 24 bytes = 3 * 8 bytes)
- * RA Header: router lifetime = 0 (PvD-aware hosts will not use this router as a default router), implicit length = 2

And here is the RA sent for PvD-aware hosts:

- o RA Header: router lifetime = 0 (non-PvD-aware hosts will not use this router as a default router)
- o PvD Option header: length = 3 + 2 + 4 + 3, PvD ID FQDN = bar.example.org., R-flag = 1 (actual length of the header 24 bytes = 3 * 8 bytes)
- * RA Header: router lifetime = 1600 (PvD-aware hosts will use this router as a default router), implicit length = 2
- * Prefix Information Option: length = 4, prefix = 2001:db8:f00d::/64
- * Recursive DNS Server Option: length = 3, addresses = [2001:db8:f00d::53]

In the above example, non-PvD-aware hosts will only use the first listed RA sent by their default router and using the 2001:db8:cafe::/64 prefix. PvD-aware hosts will autonomously configure addresses from both PIOs, but will only use the source address in 2001:db8:f00d::/64 to communicate past the first hop router since only the router sending the second RA will be used as default router; similarly, they will use the DNS server 2001:db8:f00d::53 when communicating from this address.

5.3. Enabling Multi-homing for PvD-Aware Hosts

In this example, the goal is to have one prefix from one RA be usable by both non-PvD-aware and PvD-aware hosts; and to have another prefix usable only by PvD-aware hosts. This allows PvD-aware hosts to be able to effectively multi-home on the network.

The first RA is usable by all hosts. The only difference for PvD-aware hosts is that they can explicitly identify the PvD ID associated with the RA. PvD-aware hosts will also use this prefix to communicate with non-PvD-aware hosts on the same network.

- o RA Header: router lifetime = 6000 (non-PvD-aware hosts will use this router as a default router)
- o Prefix Information Option: length = 4, prefix = 2001:db8:cafe::/64
- o Recursive DNS Server Option: length = 3, addresses = [2001:db8:cafe::53]
- o PvD Option header: length = 3, PvD ID FQDN = foo.example.org., R-flag = 0 (actual length of the header 24 bytes = 3 * 8 bytes)

The second RA contains a prefix usable only by PvD-aware hosts. Non-PvD-aware hosts will ignore this RA; hence, the only PvD-aware hosts will be multi-homed.

- o RA Header: router lifetime = 0 (non-PvD-aware hosts will not use this router as a default router)
- o PvD Option header: length = 3 + 2 + 4 + 3, PvD ID FQDN = bar.example.org., R-flag = 1 (actual length of the header 24 bytes = 3 * 8 bytes)
 - * RA Header: router lifetime = 1600 (PvD-aware hosts will use this router as a default router), implicit length = 2
 - * Prefix Information Option: length = 4, prefix = 2001:db8:f00d::/64
 - * Recursive DNS Server Option: length = 3, addresses = [2001:db8:f00d::53]

Note: the above examples assume that the router has received its PvD IDs from upstream routers or via some other configuration mechanism. Another document could define ways for the router to generate its own PvD IDs to allow the above scenario in the absence of PvD ID provisioning.

5.4. Providing Additional Information to PvD-Aware Hosts

In this example, the router indicates that it provides additional information using the H-flag. The Sequence Number on the PvD Option is set to 7 in this example.

- o RA Header: router lifetime = 6000
- o Prefix Information Option: length = 4, prefix = 2001:db8:cafe::/64
- o Recursive DNS Server Option: length = 3, addresses=[2001:db8:cafe::53]
- o PvD Option header: length = 3, PvD ID FQDN = cafe.example.com., Sequence Number = 7, R-flag = 0, H-flag = 1 (actual length of the header with padding 24 bytes = 3 * 8 bytes)

A PvD-aware host will fetch `https://cafe.example.com/.well-known/pvd` to get the additional information. The following example shows a GET request that the host sends, in HTTP/2 syntax [RFC7540]:

```
:method = GET
:scheme = https
:authority = cafe.example.com
:path = /.well-known/pvd
accept = application/pvd+json
```

The HTTP server will respond with the JSON additional information:

```
:status = 200
content-type = application/pvd+json
content-length = 116

{
  "identifier": "cafe.example.com.",
  "expires": "2020-05-23T06:00:00Z",
  "prefixes": ["2001:db8:cafe::/48"],
}
```

At this point, the host has the additional information, and knows the expiry time. When either the expiry time passes, or a new Sequence Number is provided in an RA, the host will re-fetch the additional information.

For example, if the router sends a new RA with the Sequence Number set to 8, the host will re-fetch the additional information:

- o PvD Option header: length = 3 + 5 + 4 , PvD ID FQDN =
cafe.example.com., Sequence Number = 8, R-flag = 0, H-flag = 1
(actual length of the header with padding 24 bytes = 3 * 8 bytes)

However, if the router sends a new RA, but the Sequence Number has not changed, the host would not re-fetch the additional information (until and unless the expiry time of the additional information has passed).

6. Security Considerations

Since the PvD ID RA option can contain an RA header and other RA options, any security considerations that apply for specific RA options continue to apply when used within a PvD ID option.

Although some solutions such as IPsec or SeND [RFC3971] can be used in order to secure the IPv6 Neighbor Discovery Protocol, in practice actual deployments largely rely on link layer or physical layer security mechanisms (e.g., 802.1x [IEEE8021X]) in conjunction with RA Guard [RFC6105].

If multiple RAs are sent for a single PvD to avoid fragmentation, dropping packets can lead to processing only part of a PvD ID option, which could lead to hosts receiving only part of the contained options. As discussed in Section 3.2, routers MUST include the PvD ID option in all fragments generated.

This specification does not improve the Neighbor Discovery Protocol security model, but simply validates that the owner of the PvD FQDN authorizes its use with the prefix advertised by the router. In combination with implicit trust in the local router (if present), this gives the host some level of assurance that the PvD is authorized for use in this environment. However, when the local router cannot be trusted, no such guarantee is available.

It must be noted that Section 4.4 of this document only provides reasonable assurance against misconfiguration but does not prevent a hostile network access provider from advertising incorrect information that could lead applications or hosts to select a hostile PvD. However, a host that correctly implements the multiple PvD architecture ([RFC7556]) using the mechanism described in this document will be less susceptible to some attacks than a host that does not by being able to check for the various misconfigurations or inconsistencies described in this document.

Since expiration times provided in PvD Additional Information use absolute time, these values can be skewed for hosts without an

accurate time base, or due to clock skew. Such time values MUST NOT be used for security-sensitive functionality or decisions.

An attacker generating RAs on a local network can use the H-flag and the PvD ID to cause hosts on the network to make requests for PvD Additional Information from servers. This can become a denial-of-service attack, in which an attacker can amplify its attack by triggering TLS connections to arbitrary servers in response to sending UDP packets containing RA messages. To mitigate this attack, hosts MUST:

- o limit the rate at which they fetch a particular PvD's Additional Information;
- o limit the rate at which they fetch any PvD Additional Information on a given local network;
- o stop making requests for a PvD ID that does not respond with valid JSON;
- o stop making requests for all PvD IDs once a certain number of failures is reached on a particular network.

Details are provided in Section 4.1. This attack can be targeted at generic web servers, in which case the host behavior of stopping requesting for any server that doesn't behave like a PvD Additional Information server is critical. Limiting requests for a specific PvD ID might not be sufficient if the attacker changes the PvD ID values quickly, so hosts also need to stop requesting if they detect consistent failure when on a network that is under attack. For cases in which an attacker is pointing hosts at a valid PvD Additional Information server (but one that is not actually associated with the local network), the server SHOULD reject any requests that do not originate from the expected IPv6 prefix as described in Section 4.2.

7. Privacy Considerations

Retrieval of the PvD Additional Information over HTTPS requires early communications between the connecting host and a server which may be located further than the first hop router. Although this server is likely to be located within the same administrative domain as the default router, this property can't be ensured. To minimize the leakage of identity information while retrieving the PvD Additional Information, hosts SHOULD make use of an IPv6 temporary address and SHOULD NOT include any privacy-sensitive data, such as a User-Agent header field or an HTTP cookie.

Hosts might not always fetch PvD Additional Information, depending on whether or not they expect to use the information. However, if a host whitelisted only certain PvD IDs for which to fetch Additional Information, an attacker could send various PvD IDs in RAs to detect which PvD IDs are whitelisted by the client. To avoid this, hosts SHOULD either fetch Additional Information for all eligible PvD IDs on a given local network, or fetch the information for none of them.

From a user privacy perspective, retrieving the PvD Additional Information is not different from establishing a first connection to a remote server, or even performing a single DNS lookup. For example, most operating systems already perform early queries to static web sites, such as `http://captive.example.com/hotspot-detect.html`, in order to detect the presence of a captive portal.

The DNS queries associated with the PvD Additional Information MUST use the DNS servers indicated by the associated PvD, as described in Section 4.1. This ensures the name of the PvD Additional Information server is not unintentionally sent on another network, thus leaking identifying information about the networks with which the client is associated.

There may be some cases where hosts, for privacy reasons, should refrain from accessing servers that are located outside a certain network boundary. In practice, this could be implemented as a whitelist of 'trusted' FQDNs and/or IP prefixes that the host is allowed to communicate with. In such scenarios, the host SHOULD check that the provided PvD ID, as well as the IP address that it resolves into, are part of the allowed whitelist.

Network operators SHOULD restrict access to PvD Additional Information to only expose it to hosts that are connected to the local network, especially if the Additional Information would provide information about local network configuration to attackers. This can be implemented by whitelisting access from the addresses and prefixes that the router provides for the PvD, which will match the prefixes contained in the PvD Additional Information. This technique is described in Section 4.2.

8. IANA Considerations

Upon publication of this document, IANA is asked to remove the 'reclaimable' tag off the value 21 for the PvD Option (from the IPv6 Neighbor Discovery Option Formats registry).

8.1. New entry in the Well-Known URIs Registry

IANA is asked to add a new entry in the Well-Known URIs registry [RFC8615] with the following information:

URI suffix: 'pvd'

Change controller: IETF

Specification document: this document

Status: permanent

Related information: N/A

8.2. Additional Information PvD Keys Registry

IANA is asked to create and maintain a new registry called "Additional Information PvD Keys", which will reserve JSON keys for use in PvD additional information. The initial contents of this registry are given in Section 4.3, including both the table of mandatory keys and the table of optional keys.

The status of a key as mandatory or optional is intentionally not denoted in the table to allow for flexibility in future use cases. Any new assignments of keys will be considered as optional for the purpose of the mechanism described in this document.

New assignments for Additional Information PvD Keys Registry will be administered by IANA through Expert Review [RFC8126]. Experts are requested to ensure that defined keys do not overlap in names or semantics, and represent non-vendor-specific use cases. Vendor-specific keys SHOULD use sub-dictionaries, as described in Section 4.3.

IANA is asked to place this registry in a new page, entitled "Provisioning Domains (PvDs)".

8.3. PvD Option Flags Registry

IANA is also asked to create and maintain a new registry entitled "PvD Option Flags" reserving bit positions from 0 to 12 to be used in the PvD Option bitmask. Bit position 0, 1 and 2 are assigned by this document (as specified in Figure 1). Future assignments require Standards Action [RFC8126].

Since these flags apply to an IPv6 Router Advertisement Option, IANA is asked to place this registry under the existing "Internet Control

Message Protocol version 6 (ICMPv6) Parameters" page, as well as providing a link on the new "Provisioning Domains (PvDs)" page.

8.4. PvD JSON Media Type Registration

This document registers the media type for PvD JSON text, "application/pvd+json".

Type Name: application

Subtype Name: pvd+json

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: Encoding considerations are identical to those specified for the "application/json" media type.

Security considerations: See Section 6.

Interoperability considerations: This document specifies the format of conforming messages and the interpretation thereof.

Published specification: This document

Applications that use this media type: This media type is intended to be used by networks advertising additional Provisioning Domain information, and clients looking up such information.

Fragment identifier considerations: N/A

Additional information: N/A

Person and email address to contact for further information: See Authors' Addresses section

Intended usage: COMMON

Restrictions on usage: N/A

Author: IETF

Change controller: IETF

9. Acknowledgments

Many thanks to M. Stenberg and S. Barth for their earlier work: [I-D.stenberg-mif-mpvd-dns], as well as to Basile Bruneau who was author of an early version of this document.

Thanks also to Marcus Keane, Mikael Abrahamsson, Ray Bellis, Zhen Cao, Tim Chown, Lorenzo Colitti, Michael Di Bartolomeo, Ian Farrer, Phillip Hallam-Baker, Bob Hinden, Tatuya Jinmei, Erik Kline, Ted Lemon, Paul Hoffman, Dave Thaler, Suresh Krishnan, Gorrry Fairhurst, Jen Lenkova, Veronika McKillop, Mark Townsley and James Woodyatt for useful and interesting discussions and reviews.

Finally, special thanks to Thierry Danis for his valuable inputs and implementation efforts, Tom Jones for his integration effort into the NEAT project and Rigil Salim for his implementation work.

10. References

10.1. Normative References

- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [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>>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <<https://www.rfc-editor.org/info/rfc2818>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <<https://www.rfc-editor.org/info/rfc4191>>.
- [RFC4343] Eastlake 3rd, D., "Domain Name System (DNS) Case Insensitivity Clarification", RFC 4343, DOI 10.17487/RFC4343, January 2006, <<https://www.rfc-editor.org/info/rfc4343>>.

- [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>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [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>>.
- [RFC6980] Gont, F., "Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery", RFC 6980, DOI 10.17487/RFC6980, August 2013, <<https://www.rfc-editor.org/info/rfc6980>>.
- [RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, <<https://www.rfc-editor.org/info/rfc7493>>.
- [RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <<https://www.rfc-editor.org/info/rfc7525>>.
- [RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <<https://www.rfc-editor.org/info/rfc7556>>.
- [RFC8028] Baker, F. and B. Carpenter, "First-Hop Router Selection by Hosts in a Multi-Prefix Network", RFC 8028, DOI 10.17487/RFC8028, November 2016, <<https://www.rfc-editor.org/info/rfc8028>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [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>>.

- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <<https://www.rfc-editor.org/info/rfc8615>>.

10.2. Informative References

- [I-D.kline-mif-mpvd-api-reqs]
Kline, E., "Multiple Provisioning Domains API Requirements", draft-kline-mif-mpvd-api-reqs-00 (work in progress), November 2015.
- [I-D.stenberg-mif-mpvd-dns]
Stenberg, M. and S. Barth, "Multiple Provisioning Domains using Domain Name System", draft-stenberg-mif-mpvd-dns-00 (work in progress), October 2015.
- [IEEE8021X]
IEEE, "IEEE Standards for Local and Metropolitan Area Networks, Port-based Network Access Control, IEEE Std".
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <<https://www.rfc-editor.org/info/rfc2131>>.
- [RFC3646] Droms, R., Ed., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, DOI 10.17487/RFC3646, December 2003, <<https://www.rfc-editor.org/info/rfc3646>>.
- [RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "Secure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <<https://www.rfc-editor.org/info/rfc3971>>.
- [RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <<https://www.rfc-editor.org/info/rfc4389>>.
- [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>>.

- [RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <<https://www.rfc-editor.org/info/rfc6125>>.
- [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>>.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, DOI 10.17487/RFC6296, June 2011, <<https://www.rfc-editor.org/info/rfc6296>>.
- [RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <<https://www.rfc-editor.org/info/rfc6960>>.
- [RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <<https://www.rfc-editor.org/info/rfc7049>>.
- [RFC7278] Byrne, C., Drown, D., and A. Vizdal, "Extending an IPv6 /64 Prefix from a Third Generation Partnership Project (3GPP) Mobile Interface to a LAN Link", RFC 7278, DOI 10.17487/RFC7278, June 2014, <<https://www.rfc-editor.org/info/rfc7278>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.
- [RFC8106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 8106, DOI 10.17487/RFC8106, March 2017, <<https://www.rfc-editor.org/info/rfc8106>>.

- [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>>.
- [URN] IANA, "Uniform Resource Names (URN) Namespaces", <<https://www.iana.org/assignments/urn-namespaces/urn-namespaces.xhtml>>.

Authors' Addresses

Pierre Pfister
Cisco
11 Rue Camille Desmoulins
Issy-les-Moulineaux 92130
France

Email: ppfister@cisco.com

Eric Vyncke
Cisco
De Kleetlaan, 6
Diegem 1831
Belgium

Email: evyncke@cisco.com

Tommy Pauly
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: tpauly@apple.com

David Schinazi
Google LLC
1600 Amphitheatre Parkway
Mountain View, California 94043
United States of America

Email: dschinazi.ietf@gmail.com

Wenqin Shao
Cisco
11 Rue Camille Desmoulins
Issy-les-Moulineaux 92130
France

Email: wenshao@cisco.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: November 27, 2018

Z. Kahn, Ed.
LinkedIn
J. Brzozowski, Ed.
Comcast
R. White, Ed.
LinkedIn
May 26, 2018

Requirements for IPv6 Routers
draft-ietf-v6ops-ipv6rtr-reqs-04

Abstract

The Internet is not one network, but rather a collection of networks. The interconnected nature of these networks, and the nature of the interconnected systems that make up these networks, is often more fragile than it appears. Perhaps "robust but fragile" is an overstatement, but the actions of each vendor, implementor, and operator in such an interconnected environment can have a major impact on the stability of the overall Internet (as a system). The widespread adoption of IPv6 could, particularly, disrupt network operations, in a way that impacts the entire system.

This time of transition is an opportune time to take stock of lessons learned through the operation of large-scale networks on IPv4, and consider how to apply these lessons to IPv6. This document provides an overview of the design and architectural decisions that attend IPv6 deployment, and a set of IPv6 requirements for routers, switches, and middleboxes deployed in IPv6 networks. The hope of the editors and contributors is to provide the necessary background to guide equipment manufacturers, protocol implementors, and network operators in effective IPv6 deployment.

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 November 27, 2018.

Copyright Notice

Copyright (c) 2018 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Contributors	3
1.2. Acknowledgments	4
1.3. Use and Applicability	4
2. Review of the Internet Architecture	5
2.1. Robustness Principle	5
2.2. Complexity	7
2.2.1. Elegance	7
2.2.2. Trade-offs	8
2.3. Layered Structure	9
2.4. Routers	10
3. Requirements Related to Device Management and Security	12
3.1. Programmable Device Access	12
3.2. Human Readable Device Access	13
3.3. Supporting Zero Touch Provisioning for Connected Devices	13
3.4. Device Protection against Denial of Service Attacks	15
4. Requirements Related to Telemetry	15
4.1. Device State and Traceability	16
4.2. Topology State and Traceability	16
4.3. Flow State and Traceability	17
5. Requirements Related to IPv6 Forwarding and Addressing	17
5.1. The IPv6 Address is not a Host Identifier	17
5.2. Router IPv6 Addresses	18
5.3. The Maximum Transmission Unit	19
5.4. ICMP Considerations	20
5.5. Machine Access to the Forwarding Table	21
5.6. Processing IPv6 Extension Headers	22
5.7. IPv6 Operation by Default	22
5.8. IPv6 Only Operation	22

5.9. Prefix Length Handling in IPv6 Packet Forwarding	23
5.10. IPv6 Mobility Support	23
6. Security Considerations	23
6.1. Robustness and Security	23
6.2. Programmable Device Access and Security	24
6.3. Zero Touch Provisioning and Security	24
6.4. Defaulting to IPv6 Forwarding and Security	24
7. IANA Considerations	25
8. Conclusion	25
9. References	25
9.1. Normative References	25
9.2. Informative References	25
Authors' Addresses	32

1. Introduction

This memo defines and discusses requirements for devices that perform forwarding for Internet Protocol version 6 (IPv6). The "use and applicability" section below contains more information on the specific target of this draft, and the envisioned use of the draft.

Readers should recognize that while this memo applies to IPv6, routers and middleboxes IPv6 packets will often also process IPv4 packets, forward based on MPLS labels, and potentially process many other protocols. This memo will only discuss IPv4, MPLS, and other protocols as they impact the behavior of an IPv6 forwarding device; no attempt is made to specify requirements for protocols other than IPv6. The reader should, therefore, not count on this document as a "sole source of truth," but rather use this document as a guide.

For IPv4 router requirements, readers are referred to [RFC1812]. For simplicity, the term "devices" is used interchangeably with the phrase "routers and middleboxes" and the term "routers" throughout this document. These three terms represent stylistic differences, rather than substantive differences.

This document is broken into the following sections: a review of Internet architecture and principles, requirements relating to device management, requirements related to telemetry, requirements related to IPv6 forwarding and addressing, and future considerations. Following these sections, a short conclusion is provided for review.

1.1. Contributors

Shawn Zandi, Pete Lumbis, Fred Baker, James Woodyatt, Erik Muller, Lee Howard, and Joe Clarke contributed significant text and ideas to this draft.

1.2. Acknowledgments

The editors and contributors would like to thank Ron Bonica, Lorenzo Coitti, Brian E. Carpenter, Tim Chown, Peter Lothberg, and Mikael Abrahamsson for their comments, edits, and ideas on the text of this draft.

1.3. Use and Applicability

The conceived use of this draft is as a reference point. The first part of the draft is designed to help IPv6 implementors and network operators to understand Internet and Internetworking technologies, so they can better understand the context of IPv6. The second part of this draft outlines a common set of requirements for devices which are designed to forward IPv6 traffic. This can include (but is not limited to) the devices described below.

- o Devices which are primarily designed to forward traffic between more than two interfaces. These are normally referred to by the Internet community as routers or, in some cases, intermediate systems.
- o Devices which are designed to modify packets rather than "just" forwarding them. These are often referred to by the Internet community as middleboxes. See [RFC7663] for a fuller definition of middleboxes.

This draft is not designed to apply to consumer devices, such as smart devices (refrigerators, light bulbs, garage door openers, etc.), Internet of Things (IoT) devices, cell phones primarily used as an end user device (such as checking email, social media, games, and use as a voice device), and other devices of this class. It is up to each provider or equipment purchaser to determine how best to apply this document to their environment.

The intended use of this document is for operators to be able to point to a common set of functionality which should be available across all IPv6 implementations. Several members of the community have argued there is no common set of IPv6 features; rather each deployment of IPv6 calls for different feature sets. However, the authors of this draft believe outlining a common set of features expected of every IPv6 forwarding device is useful. Specifically:

- o If every IPv6 deployment situation is unique, and requires a different set of features, there will not be a solid definition of what an IPv6 forwarding device is, or performs. This fragments the concept of IPv6 forwarding devices in an unhelpful way, especially as IPv6 deployment is already seen as difficult.

- o It encourages developers and vendors to code a multitude of different IPv6 stacks, one for each possible set of features. This fragments the experience with these stacks, potentially preventing the development of a well designed, fully featured stacks the entire community can rely on.

Because this document is designed to be a reference point rather than a best common practice or a standard, this document does not use [RFC2119] upper case "must" and "should" throughout. Rather, it uses lower case "must" and "should" throughout, anticipating operators will find such guidance clear and useful.

2. Review of the Internet Architecture

The Internet relies on a number of basic concepts and considerations. These concepts are not explicitly called out in any specification, nor do they necessarily impact protocol design or packet forwarding directly. This section provides an overview of these concepts and considerations to help the reader understand the larger context of this document.

2.1. Robustness Principle

Every point where multiple protocols interact, is an interaction surface that can threaten the robustness of the overall system. While it may seem the global Internet has achieved a level of stability that makes it immune to such considerations, the reality is every network is a complex system, and is therefore subject to massive non repeatable unanticipated failures. Postel's Robustness Principle countered this problem with a simple statement, explicated in [RFC1122]: "Be conservative in what you do, and liberal in what you accept from others."

However, since this time, it has been noted that following this law allows errors in protocols to accumulate over time, with overall negative effects on the system as a whole. [RFC1918] describes several points in conjunction with this principle that bear updating based on further experience with large-scale protocol and network deployments within the Internet community, including:

- o Applications should deal with error states gracefully; an application should not degrade in a way that will cause the failure of adjacent systems when possible. For instance, when a routing protocol implementation fails, it should not do so in a way that will cause the spreading of or continued existence of false reachability information, nor should it fail in a way that overloads adjacent routers or interacting protocols and causing a cascading failure.

- o It is best to assume the network is filled with poor implementations and malevolent actors, both of which will find every possible failure mode over time.
- o It is best to assume every technology will be used to the limits of its technical capabilities, rather than assuming a particular protocol's scope of use will align (in any way) with the intent of the original designer(s). [RFC5218] defines a wildly successful protocol as one that "far exceeds its original goals, in terms of purpose (being used in scenarios far beyond the initial design), in terms of scale (being deployed on a scale much greater than originally envisaged), or both." Successful implementations attract more functionality, much like a few nodes in a scale free graph eventually become connectivity hubs.
- o Protocols and implementations change over time. A corollary of the assumption that protocols will be used until they reach their technical limits is that protocols will change over time as they gain new functionality. [RFC5218] points out several problems with "wild success" in a protocol: undesirable side effects, performance problems, and becoming a high value attack target. Protocol and implementation design should take into account use cases that have not yet been thought of by building flexibility into protocols. Protocols should also remained focused on a narrow range of use cases; it is often wise to invent a new protocol than to extend a single protocol into a broad set of use cases.
- o Protocols are sometimes replaced or updated to new versions in order to add new capabilities or features. Updating a protocol requires great care in providing for a transition mechanism between older and newer versions. [RFC8170] provides sound advice on protocol transition planning and mechanisms.
- o Obscure, but legal, protocol features are often ignored or left unimplemented. Protocols must handle receiving unexpected information gracefully so they do not fail because of incomplete or partial implementations. Protocols should avoid specifying contradictory states, or features that will cause interoperability issues if multiple implementations choose to implement different feature sets.
- o Monocultures are almost always bad. While multiple implementations can represent an interaction surface which increases complexity, particularly if a broad set of protocol capabilities and/or implementation features are used, using the same implementation at every point in a deployment results in a mono-culture. In a monoculture, a single event can trigger a

defect in every router, causing a network failure. Mono-cultures must be carefully balanced against interaction surfaces; often this is best accomplished by using multiple implementations and minimal, widely implemented, and well understood protocol features.

A summary of the points above might be this: It is important to work within the bounds of what is actually implemented in any given protocol, and to leave corner cases for another day. It is often easy to assume "virtual oceans" are easier to boil than physical ones, or for an ocean to appear much smaller because it is being implemented in software. This is often deceptive. It is never helpful to boil the ocean whether in a design, an implementation, or a protocol.

2.2. Complexity

Complexity, as articulated by Mike O'Dell (see [RFC3439]), is "the primary mechanism which impedes efficient scaling, and as a result is the primary driver of increases in both capital expenditures (CAPEX) and operational expenditures (OPEX)." At the same time, complexity cannot be "solved," but rather must be "managed." The simplest and most obvious solution to any problem is often easy to design, deploy, and manage. It's also often wrong and/or broken. As much as developers, designers, and operators might like to make things as simple as possible, hard problems require complex solutions. See Alderson and Doyle [COMPLEXHARD] for a discussion of the relationship between hard problems and complex solutions.

The following sections contain observations which apply to the management of complexity in both protocol and network design.

2.2.1. Elegance

Elegance should be the goal of protocol and network design. Rather than seeking out simple solutions because they are simple, seek out solutions that will solve the problem in the simplest way possible (and no simpler). Often this will require:

- o Ensuring the goal is actually the goal. Many times the goal is taken from the operational realm into the protocol design realm before enough thought has been applied to ensure the correct problem is being addressed.
- o Seeing the problem from different angles, trying to break the problem up in multiple ways; and trying, abandoning, and rebuilding ideas and implementations until a better way is found.

- o Sometimes the complexity of the solution will overwhelm the use case; sometimes it is better to leave the apparent problem unsolved, or allow the community time and space to find a simpler solution.

2.2.2. Trade-offs

There are always trade-offs. For any protocol, network, or operational design decision, there will always be a trade-off between at least two competing goals. If some problem appears to have a single solution without trade-offs, this doesn't mean the trade-offs don't exist. Rather, it means the trade-offs haven't been discovered yet. In the area of protocol and network design, these trade-offs often take the form of common "choose two of three" situations, such as "quick, cheap, high quality." In network and protocol design, the trade-offs are often:

- o The amount of state carried in the system and the speed at which it changes, or simply the state. The amount of state required to operate a system as it scales tends to be nonlinear. Some instances of this are described in [RFC3439] section 2.2.1, the Amplification Principle.
- o The number of interaction surfaces between the components that make up the complete system, and the depth of those interaction surfaces. Some examples of surfaces are described in [RFC3439]section 2.2.2, the Coupling Principle. Layering is essentially a form of abstraction; all abstractions are subject to the law of leaky abstractions, [LEAKYABS] which states: "all nontrivial abstractions leak."
- o The desired optimization, including efficient use of network resources, optimal support for business objectives, and optimal support for a specific set of applications.

These three make up a "triangle problem." For instance, to increase the optimization of traffic flow through a network generally requires adding more state to the control plane, leading to problems in complexity due to amplification. To reduce amplification, the control plane (or perhaps the various functions the control plane serves) can be broken up into subsystems, or modules. Breaking the control plane up into subsystems, however, introduces interaction surfaces between the components, which is another form of complexity. [RFC7980] provides a good overview of network complexity; in particular, section 3 of that document provides some examples of complexity trade-offs.

2.3. Layered Structure

The Internet data plane is organized around broad top and bottom layers, and much thinner middle layer. This is illustrated in the figure below.

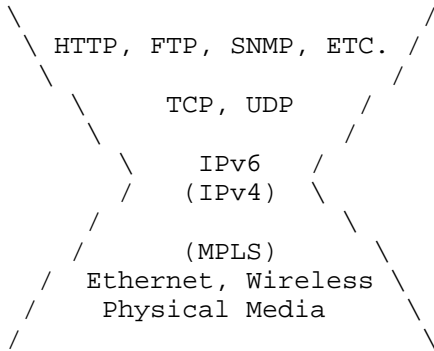


Figure 1

This layering emulates or mirrors many naturally occurring systems; it is a common strategy for managing complexity (see Meyer's presentation on complexity). [COMPLEXLAYER] The single protocol in the center, IPv6, serves to separate the complexity of the lower layers from the complexity of the upper layers. This center layer of the Internet ecosystem has traditionally been called the Network Layer, in reference to the Department of Defense (DoD) [DoD] and OSI models. [OSI] The Internet ecosystem includes two different protocols in this central location.

- o IPv4, an older network protocol that, it is anticipated, will be replaced over time as the Internet ecosystem standardizes on IPv6
- o IPv6, a newer network protocol that is being adopted

MPLS is often used as a "middle" sub-transport layer, and at other times as "middle" sub data link layer; hence MPLS is difficult to classify within the strictly hierarchical model depicted here. These protocols are often treated as if they exist in strict hierarchical layers with a well defined and followed Application Programming Interface (API), data models, Remote Procedure Calls (RPCs), sockets, etc. The reality, however, is there are often solid reasons for violating these layers, creating interaction surfaces that are often deeper than intended or understood without some experience. Beyond this, such layering mechanisms act as information abstractions. It is well known that all such abstractions leak (see above on the law of leaky abstractions). Because of these intentional and

unintentional leakages of information, the interactions between protocols is often subtle.

2.4. Routers

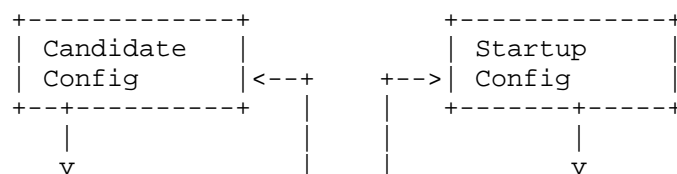
A router connects to two or more logical interfaces and at least one physical interface. A router processes packets by:

- o Receiving a packet through an interface
- o Stripping the data link, physical header, or tunnel encapsulation off the packet
- o Examining the packet for errors, and determining if this packet needs to be punted to another process on the router
- o Looking up the destination in a local forwarding table
- o Rewriting the data link and/or physical layer header
- o Transmitting the packet out an interface

When consulting the forwarding table, the router searches for a match based on:

- o The longest prefix containing the destination address (this is the most common matching element)
- o A label, such as a flow label or MPLS label
- o The source address or other header fields (not common)

The router then examines the information in the matching entry to determine the next hop, or rather the next logically connected device to forward the packet to. The next hop will either be another router, which will presumably carry the packet closer to the final destination, or it will be the destination host itself. The following figure provides a conceptual model of a router; not all routers actually have this set of tables and interactions, and some have many more moving parts. This model is simply used as a common reference to promote understanding.



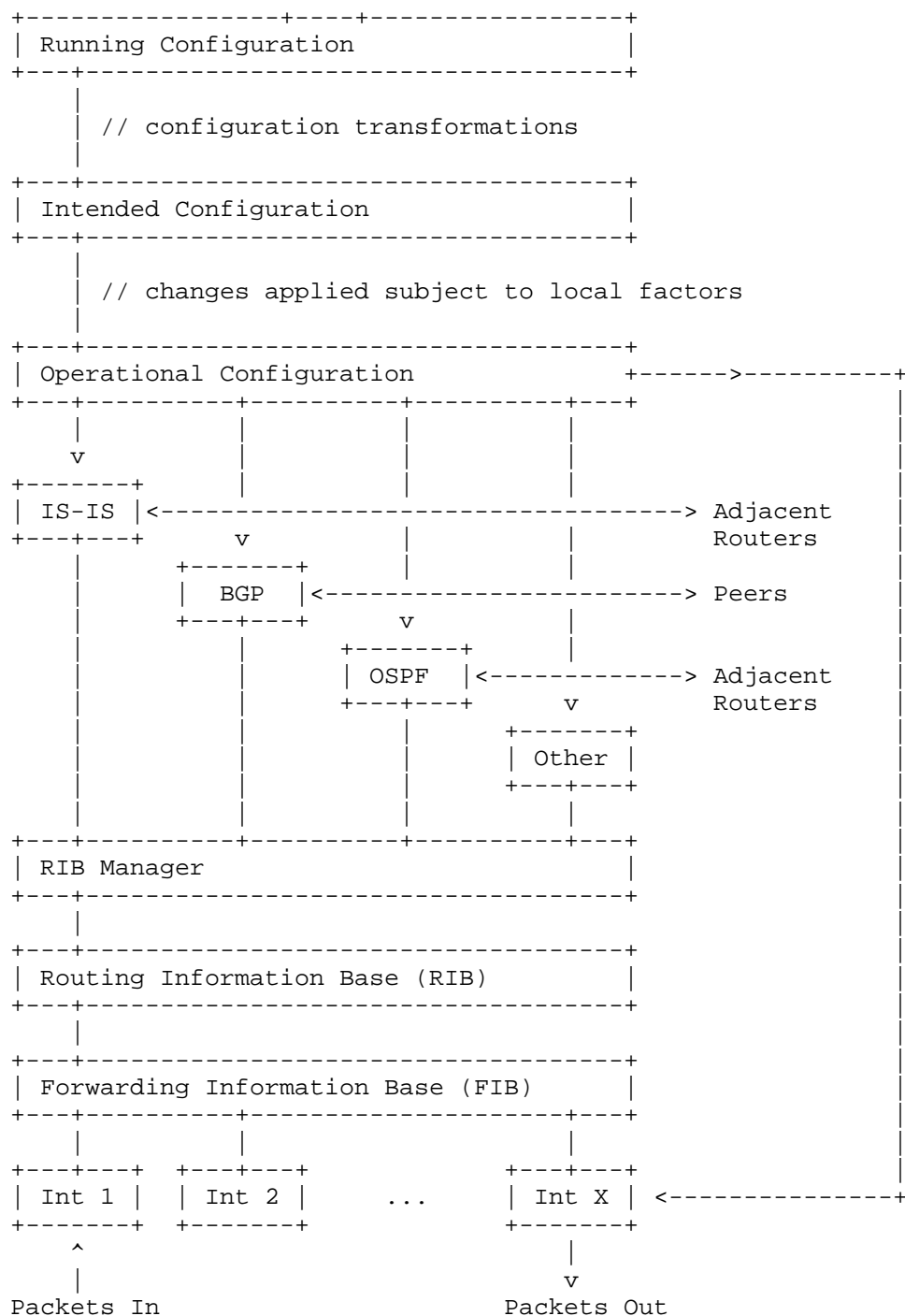


Figure 2

The configuration datastores in this figure follow [RFC8342].

3. Requirements Related to Device Management and Security

Network engineering began in the era of Command Line Interfaces (CLIs), and has generally stayed with these CLIs even as the Graphical User Interface (GUI) has become the standard way of interacting with almost every other computing device. Direct human interaction with routers and middleboxes in large-scale and complex environments, however, tends to result in an unacceptably low Mean Time Between Mistakes (MTBM), directly impacting the overall availability of the network. In reaction to this, operators have increased their reliance on automation, specifically targeting machine to machine interfaces, such as Remote Procedure Calls (RPCs) and Application Programming Interface (API) solutions, to manage and configure routers and middleboxes. This section considers the various components of device management.

Across all interface types, devices should provide and use complete, idempotent, stateless configurations. Further, default settings should be accessible in some way, even if they are hidden by default for configuration readability.

3.1. Programmable Device Access

Configuration primarily relates to the startup, candidate, running, intended, and operational configurations in the router model shown above. In order to deploy networks at scale, operators rely on automated management of router configuration. This effort has traditionally focused on screen scraping and other proprietary methods of "reading" and "writing" configuration information through a CLI. In the future, operators expect to move towards open source/open standards YANG models, regardless of how these are encoded and/or carried (or marshaled).

Vendors and implementors should implement machine readable interfaces with overlays to support human interaction, rather than human readable interfaces with overlays to support machine to machine interaction. Emphasis should be placed on machine to machine interaction for day to day operations, rather than on human readable interfaces, which are largely used in the process of troubleshooting. Within the realm of machine to machine interfaces, emphasis should be placed on marshaling information in YANG models.

To support automated router configuration, IPv6 routers and routers should support YANG configuration, including (but not limited to):

- o Openconfig models [OPENCONF] related to the protocols configured on the device, interface state, and device state
- o [RFC8343]: A YANG Data Model for Interface Management
- o [RFC7224]: IANA Interface Type YANG Module
- o [RFC8344]: A YANG Data Model for IP Management
- o [RFC7317]: A YANG Data Model for System Management
- o [RFC8349]: A YANG Data Model for Routing Management (NMDA Version)

3.2. Human Readable Device Access

To operate a network at scale, operators rely on the ability to access routers and middleboxes to troubleshoot and gather state manually through a number of different interfaces. These interfaces should provide current device configuration, current device state (such as interface state, packets drops, etc.), and current control plane contents (such as the RIB in the figure above). In other words, manual interfaces should provide information about the router (the whole device stack).

To support manual state gathering and troubleshooting, IPv6 routers and middleboxes should support:

- o TELNET ([RFC0854]): TELNET should be disabled by default, but should be available for operational purposes as required or as configured by the operator
- o SSH ([RFC4253]): SSH should be the default access for IPv6 capable routers
- o All network devices supporting IPv6 must support access through an Ethernet management port

3.3. Supporting Zero Touch Provisioning for Connected Devices

To operate a network at scale, operators rely on protocols and mechanisms that reduce provisioning time to a minimum. The preferred state is zero touch provisioning; plug a new router in and it just works without any manual configuration. The closer an operator can come to this ideal, the more MTBM and Operational Expenses (OPEX) can be reduced -- important goals in the real world. IPv6 routers should support several standards, including, but not limited to:

- o [I-D.ietf-dhc-rfc3315bis]: Dynamic Configuration Protocol for IPv6 must be supported.
- o [RFC4862]: IPv6 Stateless Address Autoconfiguration (SLAAC) must be supported, and must be enabled by default on all router interfaces. SLAAC must be able to be disabled by operators who prefer to use some other mechanism for address management and assignment (specifically for customer facing edge ports).
- o [RFC7217]: Semantically Opaque Interface Identifiers should be supported unless there's a need to embed MAC address.
- o [RFC7934]: Host Address Availability, the ability to assign multiple addresses to a host, should be supported.
- o [RFC7527]: Enhanced Duplicate Address Detection should be supported.
- o [RFC7527]: Enhanced Duplicate Address Detection may be disabled for manually configured interfaces.
- o [RFC8028]: First-Hop Router Selection by Hosts, specifically section 2.1, which says a router should be able to send a PIO with both the L and A bits cleared.
- o [RFC3810]: Routers supporting IPv6 must support Multicast Listener Discovery Version 2
- o [RFC7772]: Routers supporting IPv6 should support Reducing Energy Consumption of Router Advertisements
- o [RFC8273]: Routers supporting IPv6 should support Unique IPv6 Prefix per Host

The provisioning of Domain Name Systems (DNS) system information is a contentious topic, based on provider, operating system, interface, and other requirements. This document therefore addresses the mechanisms that must be included in IPv6 router implementations, but leaves the option of what to configure and deploy to the network operator. Routers supporting IPv6, and intended for user facing connections, must support:

- o [RFC3646]: DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6) if DHCPv6 is supported.
- o [RFC8106]: IPv6 Router Advertisement Options for DNS Configuration. This includes the ability to send Router Advertisements (RAs) with DNS information.

Whether these are enabled by default, or require extra configuration, is left as an exercise for providers and implementation developers to determine on a case by case basis.

3.4. Device Protection against Denial of Service Attacks

Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are unfortunately common in the Internet globally; these types of attacks cost network operators a great deal in opportunity and operational costs in prevention and responses. To provide for effective counters to DoS and DDoS attacks directly on routers:

- o Manufacturers and system integrators should test and clearly report the packet/traffic load handling capabilities of devices with and without various encryption methods enabled
- o Routers should be able to police traffic destined to the control plane based on the rate of traffic received, including the ability to police individual flows, targeted services, etc., at individual rates as described in [RFC6192]
- o Ideally, devices should be able to statefully filter traffic destined to the control plane

There are other useful techniques for dealing with DDoS attacks at the network level, including: transferring sessions to a new address and abandoning the address under attack, using BGP communities to spread the attack over multiple ingress ports and "consume" it, and requiring mutual authentication before allocating larger resource pools to a connection. These techniques are not "device level," and hence are not considered further here.

4. Requirements Related to Telemetry

Telemetry relates to information devices push to systems used to monitor and track the state of the network. This applies to individual devices as well as the network as a system. Two major challenges face operators in the area of telemetry:

- o Information that is laid out primarily for human, rather than machine, consumption. While human consumption of telemetry is important in some situations, this information should be supplied in a form that focuses on machine readability with an overlay or interpreter that allows human consumption.
- o Software systems that require information to be queried (or polled or even pushed) on a per-item basis. This form of organization can produce a lot of information, and a lot of individual packets,

very quickly, overwhelming monitoring systems and consuming a large amount of available network resources. Instead, telemetry should be focused on bulk collection.

There are three broad categories of telemetry: device state and traceability, topology state and traceability, and flow traceability. These three roughly correspond to the management plane, the control plane, and the forwarding plane of the network. Each of the sections below considers one of these three telemetry types.

4.1. Device State and Traceability

Ideally, the entire network could be monitored using a single modeling language to ease implementation of telemetry systems and increase the pace at which new software can be deployed in production environments. In real deployments, it is often impossible to reach this ideal; however, reducing the languages and methods used, while focusing on machine readability, can greatly ease the deployment and management of a large-scale network. Specifically, IPv6 routers should support:

- o [RFC6241] and [RFC8040]: NETCONF/RESTCONF transporting telemetry formatted according to YANG (see above)
- o [I-D.ietf-i2rs-yang-l2-network-topology]: An I2RS model for layer 2 topologies
- o [I-D.ietf-netconf-yang-push]: YANG Datastore Subscription
- o [RFC5424]: Syslog
- o gRPC based telemetry interfaces [GRPC]
- o Simple Network Management Protocol (SNMP) MIBs as appropriate

Syslog and SNMP access for telemetry should be considered "legacy," and should not be the focus of new telemetry access development efforts.

4.2. Topology State and Traceability

IPv6 routers are part of a system of devices that, combined, make up the entire network. Viewing the network as a system is often crucial for operational purposes. For instance, being able to understand changes in the topology and utilization of a network can lead to insights about traffic flow and network growth that lead to a greater understanding of how the network is operating, where problems are developing, and how to improve the network's performance. To support

systemic monitoring of the network topology, IPv6 devices should support at least:

- o [RFC5424]: North-Bound Distribution of Link-State and Traffic Engineering (TE) Information using BGP
- o [I-D.ietf-i2rs-yang-l2-network-topology]: An I2RS model for layer 2 topologies
- o [RFC8346]: An I2RS model for layer 3 topologies
- o [RFC8345]: A Data Model for Network Topologies

4.3. Flow State and Traceability

Network operators frequently need to observe and understand the types, sources, and destinations of traffic passing through devices. For example, information about traffic flows may be used to identify abuse (such as DDOS attacks) or to plan network expansions based on traffic patterns. To support insight and analysis of this traffic, IPv6 devices should support IPFIX as described in [RFC7011], PSAMP as described in [RFC5474], or some other flow state mechanism.

In-situ Operational and Management (iOAM) is a technology that being developed at the time of this writing; see [I-D.ietf-ippm-ioam-data]. Operators and vendors should consider the deployment of iOAM to provide deeper information about flow and topology information.

5. Requirements Related to IPv6 Forwarding and Addressing

There are a number of capabilities that a device should have to be deployed into an IPv6 network, and several forwarding plane considerations operators and vendors need to bear in mind. The sections below explain these considerations.

5.1. The IPv6 Address is not a Host Identifier

The IPv6 address is commonly treated as a host identifier; it is not. Rather, it is an interface identifier that describes the topological point where a particular host connects to the Internet. Specifically:

- o The IPv6 address will change when a device changes where it connects to the network.
- o A single host can have multiple addresses. For instance, a host may have one address per interface, or multiple addresses assigned

through different mechanisms, or through multiple connection points.

- o A single IPv6 address may represent many hosts, as in the case of a group of hosts reachable through a multicast address, or a set of services reachable through an anycast address.

Because the host address may change at any time, it is generally harmful to embed IPv6 addresses inside upper layer headers to identify a particular host.

5.2. Router IPv6 Addresses

Internet Routing Registries may allocate a network operator a wide range of prefix lengths (see [RFC6177] for further information). Within this allocation, network operators will often suballocate address space along nibble boundaries (/48, /52, /56, /60, and /64) for ease of configuration and management. Several common practices are:

- o Each multiaccess interface is allocated a /64
- o Point-to-point links are allocated a /64, but should be addressed with a longer prefix length to prevent certain kinds of denial of service attacks ([RFC6547] originally mandated 64 bit prefix lengths on point-to-point links; [RFC6164] explains possible security issues with assigning a 64 bit prefix length to a point-to-point, and recommends a /127 instead)
- o Although aggregation is typically only performed to the nibble boundaries noted above, variances are possible
- o Loopback addresses are assigned a /128

Given these common practices, routers designed to run IPv6 should support the following addressing conventions:

- o The default prefix length on any interface other than a loopback should be a /64
- o Configuring a prefix length longer than a /64 on any multi-access interface should require additional configuration steps to prevent manual configuration errors
- o Routers must not assume IPv6 prefix lengths only on nibble boundaries

- o Routers should support any prefix length shorter or greater than /64
- o Loopback interfaces should default to a /128 prefix length unless some additional configuration is undertaken to override this default setting
- o Routers must be able to generate link local addresses on all links and/or interfaces using stateless address autoconfiguration (see [RFC6434]).

5.3. The Maximum Transmission Unit

The long history of the Maximum Transmission Unit (MTU) in networks is not a happy one. Specific problems with MTU sizing include:

- o Many different default sizes on different media types, from very small (576 bytes on X.25) to very large (17914 bytes on 16Mbps Token Ring)
- o Many different ways to calculate the MTU on any given link; for instance a 9000 byte MTU can be calculated as 8184 bytes on one operating system, 8972 on another, and 9000 on a third
- o The increasing use of tunnel encapsulations in the network; for instance MPLS over GRE over IP over...
- o The wide variety of default MTUs across many different end hosts and operating systems
- o The general ineffectiveness of path MTU discovery to operate correctly in the face of packet filters and rate limiters (see the section on ICMP filtering below)
- o Lower speed links at the network edge which require a lot of time to serialize a packet with a large MTU
- o Increased jitter caused by the disparity between large and small packet size across a lower bandwidth links

The final point requires some further elucidation. The time required to serialize various packets at various speeds are:

- o 64 byte packet onto a 10Mb/s link: .5ms
- o 1500 byte packet onto a 10Mb/s link: 1.2ms
- o 9000 byte packet onto a 10Mb/s link: 7.2ms

- o 64 byte packet onto a 100Mb/s link: .05ms
- o 1500 byte packet onto a 100Mb/s link: .12ms
- o 9000 byte packet onto a 100Mb/s link: .72ms

A 64 byte packet trapped behind a single 1500 byte packet on a 10Mb/s link suffers 1.2ms of serialization delay. Each additional 1500 byte packet added to the queue in front of the 64 byte packet adds an additional 1.2ms of delay. In contrast, a 64 byte packet trapped behind a single 9000 byte packet on a 10Mb/s link suffers 7.7ms of serialization delay. Each additional 9000 byte packet added to the queue adds an additional 7.2ms of serialization delay. The practical result is that larger MTU sizes on lower speed links can add a significant amount of delay and jitter into a flow. On the other hand, increasing the MTU on higher speed links appears to add negligible additional delay and jitter.

The result is that it costs less in terms of overall systemic performance to use higher MTUs on higher speed links than on lower speed links. Based on this, increasing the MTU across any particular link may not increase overall end-to-end performance, but can greatly enhance the performance of local applications (such as a local BGP peering session, or a large/long standing elephant flow used to transfer data across a local fabric), while also providing room for tunnel encapsulations to be added with less impact on lower MTU end systems.

The general rule of thumb is to assume the largest size MTU should be used on higher speed transit only links in order to support a wide array of available link sizes, default MTUs, and tunnel encapsulations. Routers designed for a network core, data center core, or use on the global Internet should support at least 9000 byte MTUs on all interfaces. MTU detection mechanisms, such as IS-IS hello padding, described in [RFC3719], should be enabled to ensure correct point-to-point MTU configuration. Devices should also support:

- o [RFC8201]: Path MTU Discovery for IP version 6
- o [RFC4821]: Packetization Layer Path MTU Discovery

5.4. ICMP Considerations

Internet Control Message Protocol (ICMP) is described in [RFC0792] and [RFC4443]. ICMP is often used to perform a traceroute through a network (normally by using a TTL expired ICMP message), for Path MTU discovery, and, in IPv6, for autoconfiguration and neighbor

discovery. ICMP is often blocked by middleboxes of various kinds and/or ICMP filters configured on the ingress edge of a provider network, most often to prevent the discovery of reachable hosts and network topology. Routers implementing IPv6:

- o Should rate limit the generation of ICMP echo and echo responses by default (for instance, using a token bucket method as described in [RFC4443]). The device should support the configuration of not generating ICMP echo, echo response, and time exceeded packets to prevent topology discovery.
- o Should rate limit the generation of ICMP error messages with a token bucket method as described in [RFC4443]. Rate limits should be narrow enough to (a) protect the device's ability to generate packets and (b) reduce the usefulness of ICMP error packets as part of a distributed denial of service attack. Limits should be generous enough to allow successful path MTU discovery and traceroute. For example, in a small/mid-size device, the possible defaults could be bucket size=100, refill rate=100/s. Larger devices can afford more generous rate limits.
- o Should implement the filtering suggestions in [I-D.gont-opsec-icmp-ingress-filtering]
- o Should not filter Destination Unreachable or Packet Too Big ICMP error messages by default, as this has negative impacts on many aspects of IPv6 operation, particularly path MTU discovery.

There are implications for path MTU discovery and other useful mechanisms in filtering and rate limiting ICMP. The trade-off here is between allowing unlimited ICMP, which would allow path MTU detection to work, or limiting ICMP in a way that prevents negative side effects for individual devices, and hence the operational capabilities of the network as a whole. Operators rightly limit ICMP to reduce the attack surface against their network, as well as the opportunity for "perfect storm" events that inadvertently reduce the capability of routers and middleboxes. Hence ICMP can be treated as "quasi-reliable" in many situations; existence of an ICMP message can prove, for instance, that a particular host is unreachable. The non-existence of an ICMP message, however, does not prove a particular host exists or does not.

5.5. Machine Access to the Forwarding Table

In order to support treating the "network as a whole" as a single programmable system, it is important for each router have the ability to directly program forwarding information. This programmatic interface allows controllers, which are programmed to support

specific business logic and applications, to modify and filter traffic flows without interfering with the distributed control plane. While there are several programmatic interfaces available, this document suggests that the I2RS interface to the RIB be supported in all IPv6 routers. Specifically, these drafts should be supported to enable network programmability:

- o [I-D.ietf-i2rs-fb-rib-data-model]: Filter-Based RIB Data Model
- o [I-D.ietf-i2rs-fb-rib-info-model]: Filter-Based RIB Information Model
- o [I-D.ietf-i2rs-rib-data-model]: A YANG Data Model for Routing Information Base (RIB)
- o [RFC7922]: I2RS Traceability

5.6. Processing IPv6 Extension Headers

(To be added)

5.7. IPv6 Operation by Default

If a device forwards and/or originates IPv4 packets by default (without explicit configuration by the operator), it should forward and/or originate IPv6 packets by default. See the security considerations section below for reflections on the automatic configuration of IPv6 forwarding in parallel with IPv4.

5.8. IPv6 Only Operation

While the transition to IPv6 only networks may take years (or perhaps decades), a number of operators are moving to deploy IPv6 on internal networks supporting transport and data center fabric applications more quickly. Routers and middleboxes that support IPv6 should support IPv6 only operation, including:

- o Link Local addressing must be configurable and usable as the primary address on all interfaces on a device.
- o IPv4 and/or MPLS should not be required for proper device operation. For instance, an IPv4 address should not be required to determine the router ID for any protocol. See [RFC6540] section 2.
- o Any control plane protocol implementations must support the recommendations in [RFC7404] for operation using link local addresses only.

5.9. Prefix Length Handling in IPv6 Packet Forwarding

Routers must support IPv6 destination lookups in the forwarding process on a single bit prefix length increments, in accordance with [RFC7608].

5.10. IPv6 Mobility Support

Mobile IPv6 [RFC6275] and associated specifications, including [RFC3776] and [RFC4877] allow a node to change its point of attachment within the Internet, while maintaining (and using) a permanent address. All communication using the permanent address continues to proceed as expected even as the node moves around. At the present time, Mobile IP has seen only limited implementation. More usage and deployment experience is needed with mobility before any specific approach can be recommended for broad implementation in hosts and routers. Consequently, routers may support [RFC6275] and associated specifications (these specifications are not required for IPv6 routers).

6. Security Considerations

This document addresses several ways in which devices designed to support IPv6 forwarding. Some of the recommendations here are designed to increase device security; for instance, see the section on device access. Others may intersect with security, but are not specifically targeted at security, such as running IPv6 link local only on links. These are not discussed further here, as they improve the security stance of the network. Other areas discussed in this draft are more nuanced. This section gathers the intersection between operational concerns and security concerns into one place.

ICMP security is already considered in the section on ICMP; it will not be considered further here. Link local only addressing will increase security by removing transit only links within the network as a reachable destination.

6.1. Robustness and Security

Robustness, particularly in the area of error handling, largely improves security if designed and implemented correctly. Many attacks take advantage of mistakes in implementations and variations in protocols. In particular, any feature that is unevenly implemented among a number of implementations often offers an attack surface. Hence, reducing protocol complexity helps reduce the breadth of attack surfaces.

Another point to consider at the intersection of robustness and security is the issue of monocultures. Monocultures are in and of themselves a potential attack surface, in that finding a single failure mode can be exploited to take an entire network (or operator) down. On the other hand, reducing the number of implementations for any particular protocol will decrease the set of "random" features deployed in the network. These two goals will often be opposed to one another. Network designers and operators need to consider these two sides of this trade-off, and make an intelligent decision about how much diversity to implement versus how to control the attack surface represented by deploying a wide array of implementations.

6.2. Programmable Device Access and Security

Programmable interfaces, including programmable configuration, telemetry, and machine interface to the routing table, introduce a large attack surface; operators should be careful to ensure this attack surface is properly secured. Specifically:

- o Prevent external access to any administrative access points used for device programmability
- o Use AAA systems to ensure only valid devices and/or users access devices
- o Rate limit the change rate and protect management interfaces from DoS and DDoS attacks

Such interfaces should be treated no differently than SSH, SFTP, and other interfaces available to manage routers and middleboxes.

6.3. Zero Touch Provisioning and Security

Zero touch provisioning opens a new attack surface; insider attackers can simply install a new device, and assume it will be autoconfigured into the network. A "simple" solution would be to install door locks, but this will likely not be enough; defenses need to be layered to be effective. It is recommended that devices installed in the network need to contain a hardware or software identification system that allows the operator to identify devices that are installed in the network.

6.4. Defaulting to IPv6 Forwarding and Security

Operators should be aware that devices which forward IPv6 by default can introduce a new attack surface or new threats without explicit configuration. Operators should verify that IPv6 policies, including filtering, match or fulfill the same intent as any existing IPv4

policies when deploying devices capable of forwarding both IPv4 and IPv6.

7. IANA Considerations

This document has no actions for IANA.

8. Conclusion

The deployment of IPv6 throughout the Internet marks a point in time where it is good to review the overall Internet architecture, and assess the impact on operations of these changes. This document provides an overview of a lot of these changes and lessons learned, as well as providing pointers to many of the relevant documents to understand each topic more deeply.

9. References

9.1. 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>>.

9.2. Informative References

- [COMPLEXHARD]
Alderson, D. and J. Doyle, "Contrasting Views of Complexity and Their Implications For Network-Centric Infrastructures", 2010, <<http://ieeexplore.ieee.org/abstract/document/5477188/?reload=true>>.
- [COMPLEXLAYER]
Meyer, D., "Macro Trends, Architecture, and the Hidden Nature of Complexity", 2010, <<http://www.slideshare.net/dmm613/macro-trends-complexityandsdn-32951199>>.
- [DoD]
Wikipedia, "The Internet Protocol Suite", 2016, <https://en.wikipedia.org/wiki/Internet_protocol_suite>.
- [GRPC]
gRPC, "gRPC", 2016, <<http://www.grpc.io>>.

- [I-D.gont-opsec-icmp-ingress-filtering]
Gont, F., Hunter, R., Massar, J., and W. LIU, "Defeating Attacks which employ Forged ICMPv4/ICMPv6 Error Messages", draft-gont-opsec-icmp-ingress-filtering-03 (work in progress), July 2017.
- [I-D.ietf-dhc-rfc3315bis]
Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) bis", draft-ietf-dhc-rfc3315bis-13 (work in progress), April 2018.
- [I-D.ietf-i2rs-fb-rib-data-model]
Hares, S., Kini, S., Dunbar, L., Krishnan, R., Bogdanovic, D., and R. White, "Filter-Based RIB Data Model", draft-ietf-i2rs-fb-rib-data-model-01 (work in progress), March 2017.
- [I-D.ietf-i2rs-fb-rib-info-model]
Kini, S., Hares, S., Dunbar, L., Ghanwani, A., Krishnan, R., Bogdanovic, D., and R. White, "Filter-Based RIB Information Model", draft-ietf-i2rs-fb-rib-info-model-00 (work in progress), June 2016.
- [I-D.ietf-i2rs-rib-data-model]
Wang, L., Chen, M., Dass, A., Ananthakrishnan, H., Kini, S., and N. Bahadur, "A YANG Data Model for Routing Information Base (RIB)", draft-ietf-i2rs-rib-data-model-15 (work in progress), May 2018.
- [I-D.ietf-i2rs-yang-l2-network-topology]
Dong, J. and X. Wei, "A YANG Data Model for Layer-2 Network Topologies", draft-ietf-i2rs-yang-l2-network-topology-04 (work in progress), March 2018.
- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-02 (work in progress), March 2018.
- [I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", draft-ietf-netconf-yang-push-15 (work in progress), February 2018.

- [LEAKYABS] Spolsky, J., "The Law of Leaky Abstractions", 2002, <<https://www.joelonsoftware.com/2002/11/11/the-law-of-leaky-abstractions/>>.
- [OPENCONF] OpenConfig, "Openconfig release YANG models", 2016, <<https://github.com/openconfig/public/tree/master/release>>.
- [OSI] Wikipedia, "OSI Model", 2016, <https://en.wikipedia.org/wiki/OSI_model>.
- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <<https://www.rfc-editor.org/info/rfc792>>.
- [RFC0854] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, DOI 10.17487/RFC0854, May 1983, <<https://www.rfc-editor.org/info/rfc854>>.
- [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <<https://www.rfc-editor.org/info/rfc1122>>.
- [RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", RFC 1812, DOI 10.17487/RFC1812, June 1995, <<https://www.rfc-editor.org/info/rfc1812>>.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., 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>>.
- [RFC3439] Bush, R. and D. Meyer, "Some Internet Architectural Guidelines and Philosophy", RFC 3439, DOI 10.17487/RFC3439, December 2002, <<https://www.rfc-editor.org/info/rfc3439>>.
- [RFC3646] Droms, R., Ed., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, DOI 10.17487/RFC3646, December 2003, <<https://www.rfc-editor.org/info/rfc3646>>.

- [RFC3719] Parker, J., Ed., "Recommendations for Interoperable Networks using Intermediate System to Intermediate System (IS-IS)", RFC 3719, DOI 10.17487/RFC3719, February 2004, <<https://www.rfc-editor.org/info/rfc3719>>.
- [RFC3776] Arkko, J., Devarapalli, V., and F. Dupont, "Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents", RFC 3776, DOI 10.17487/RFC3776, June 2004, <<https://www.rfc-editor.org/info/rfc3776>>.
- [RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <<https://www.rfc-editor.org/info/rfc3810>>.
- [RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <<https://www.rfc-editor.org/info/rfc4253>>.
- [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>>.
- [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <<https://www.rfc-editor.org/info/rfc4821>>.
- [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>>.
- [RFC4877] Devarapalli, V. and F. Dupont, "Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture", RFC 4877, DOI 10.17487/RFC4877, April 2007, <<https://www.rfc-editor.org/info/rfc4877>>.
- [RFC5218] Thaler, D. and B. Aboba, "What Makes for a Successful Protocol?", RFC 5218, DOI 10.17487/RFC5218, July 2008, <<https://www.rfc-editor.org/info/rfc5218>>.
- [RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <<https://www.rfc-editor.org/info/rfc5424>>.

- [RFC5474] Duffield, N., Ed., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, DOI 10.17487/RFC5474, March 2009, <<https://www.rfc-editor.org/info/rfc5474>>.
- [RFC6164] Kohno, M., Nitzan, B., Bush, R., Matsuzaki, Y., Colitti, L., and T. Narten, "Using 127-Bit IPv6 Prefixes on Inter-Router Links", RFC 6164, DOI 10.17487/RFC6164, April 2011, <<https://www.rfc-editor.org/info/rfc6164>>.
- [RFC6177] Narten, T., Huston, G., and L. Roberts, "IPv6 Address Assignment to End Sites", BCP 157, RFC 6177, DOI 10.17487/RFC6177, March 2011, <<https://www.rfc-editor.org/info/rfc6177>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://www.rfc-editor.org/info/rfc6275>>.
- [RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, DOI 10.17487/RFC6434, December 2011, <<https://www.rfc-editor.org/info/rfc6434>>.
- [RFC6540] George, W., Donley, C., Liljenstolpe, C., and L. Howard, "IPv6 Support Required for All IP-Capable Nodes", BCP 177, RFC 6540, DOI 10.17487/RFC6540, April 2012, <<https://www.rfc-editor.org/info/rfc6540>>.
- [RFC6547] George, W., "RFC 3627 to Historic Status", RFC 6547, DOI 10.17487/RFC6547, February 2012, <<https://www.rfc-editor.org/info/rfc6547>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.

- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <<https://www.rfc-editor.org/info/rfc7217>>.
- [RFC7224] Bjorklund, M., "IANA Interface Type YANG Module", RFC 7224, DOI 10.17487/RFC7224, May 2014, <<https://www.rfc-editor.org/info/rfc7224>>.
- [RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <<https://www.rfc-editor.org/info/rfc7317>>.
- [RFC7404] Behringer, M. and E. Vyncke, "Using Only Link-Local Addressing inside an IPv6 Network", RFC 7404, DOI 10.17487/RFC7404, November 2014, <<https://www.rfc-editor.org/info/rfc7404>>.
- [RFC7527] Asati, R., Singh, H., Beebee, W., Pignataro, C., Dart, E., and W. George, "Enhanced Duplicate Address Detection", RFC 7527, DOI 10.17487/RFC7527, April 2015, <<https://www.rfc-editor.org/info/rfc7527>>.
- [RFC7608] Boucadair, M., Petrescu, A., and F. Baker, "IPv6 Prefix Length Recommendation for Forwarding", BCP 198, RFC 7608, DOI 10.17487/RFC7608, July 2015, <<https://www.rfc-editor.org/info/rfc7608>>.
- [RFC7663] Trammell, B., Ed. and M. Kuehlewind, Ed., "Report from the IAB Workshop on Stack Evolution in a Middlebox Internet (SEMI)", RFC 7663, DOI 10.17487/RFC7663, October 2015, <<https://www.rfc-editor.org/info/rfc7663>>.
- [RFC7772] Yourtchenko, A. and L. Colitti, "Reducing Energy Consumption of Router Advertisements", BCP 202, RFC 7772, DOI 10.17487/RFC7772, February 2016, <<https://www.rfc-editor.org/info/rfc7772>>.
- [RFC7922] Clarke, J., Salgueiro, G., and C. Pignataro, "Interface to the Routing System (I2RS) Traceability: Framework and Information Model", RFC 7922, DOI 10.17487/RFC7922, June 2016, <<https://www.rfc-editor.org/info/rfc7922>>.
- [RFC7934] Colitti, L., Cerf, V., Cheshire, S., and D. Schinazi, "Host Address Availability Recommendations", BCP 204, RFC 7934, DOI 10.17487/RFC7934, July 2016, <<https://www.rfc-editor.org/info/rfc7934>>.

- [RFC7980] Behringer, M., Retana, A., White, R., and G. Huston, "A Framework for Defining Network Complexity", RFC 7980, DOI 10.17487/RFC7980, October 2016, <<https://www.rfc-editor.org/info/rfc7980>>.
- [RFC7991] Hoffman, P., "The "xml2rfc" Version 3 Vocabulary", RFC 7991, DOI 10.17487/RFC7991, December 2016, <<https://www.rfc-editor.org/info/rfc7991>>.
- [RFC8028] Baker, F. and B. Carpenter, "First-Hop Router Selection by Hosts in a Multi-Prefix Network", RFC 8028, DOI 10.17487/RFC8028, November 2016, <<https://www.rfc-editor.org/info/rfc8028>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 8106, DOI 10.17487/RFC8106, March 2017, <<https://www.rfc-editor.org/info/rfc8106>>.
- [RFC8170] Thaler, D., Ed., "Planning for Protocol Adoption and Subsequent Transitions", RFC 8170, DOI 10.17487/RFC8170, May 2017, <<https://www.rfc-editor.org/info/rfc8170>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/info/rfc8201>>.
- [RFC8273] Brzozowski, J. and G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273, DOI 10.17487/RFC8273, December 2017, <<https://www.rfc-editor.org/info/rfc8273>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
- [RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <<https://www.rfc-editor.org/info/rfc8344>>.

- [RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <<https://www.rfc-editor.org/info/rfc8345>>.
- [RFC8346] Clemm, A., Medved, J., Varga, R., Liu, X., Ananthakrishnan, H., and N. Bahadur, "A YANG Data Model for Layer 3 Topologies", RFC 8346, DOI 10.17487/RFC8346, March 2018, <<https://www.rfc-editor.org/info/rfc8346>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.

Authors' Addresses

Zaid Ali Kahn (editor)
LinkedIn
CA
USA

Email: zaid@linkedin.com

John Brzozowski (editor)
Comcast
USA

Email: John_Brzozowski@comcast.com

Russ White (editor)
LinkedIn
Oak Island, NC 28465
USA

Email: russ@riw.us

v6ops
Internet-Draft
Intended status: Informational
Expires: January 12, 2020

J. Palet Martinez
The IPv6 Company
July 11, 2019

Additional NAT64/464XLAT Deployment Guidelines in Operator and
Enterprise Networks
draft-ietf-v6ops-nat64-deployment-08

Abstract

This document describes how NAT64 (including 464XLAT) can be deployed in an IPv6 network, whether cellular ISP, broadband ISP, or enterprise, and possible optimizations. The document also discusses issues to be considered when having IPv6-only connectivity, regarding: a) DNS64, b) applications or devices that use literal IPv4 addresses or non-IPv6 compliant APIs, and c) IPv4-only hosts or applications.

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 January 12, 2020.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Requirements Language	5
3. NAT64 Deployment Scenarios	5
3.1. Known to Work	6
3.1.1. Service Provider NAT64 with DNS64	6
3.1.2. Service Provider Offering 464XLAT, with DNS64	8
3.1.3. Service Provider Offering 464XLAT, without DNS64	12
3.2. Known to Work Under Special Conditions	14
3.2.1. Service Provider NAT64 without DNS64	14
3.2.2. Service Provider NAT64; DNS64 in the IPv6 hosts	16
3.2.3. Service Provider NAT64; DNS64 in the IPv4-only remote network	16
3.3. Comparing the Scenarios	17
4. Issues to be Considered	19
4.1. DNSSEC Considerations and Possible Approaches	19
4.1.1. Not using DNS64	20
4.1.2. DNSSEC validator aware of DNS64	21
4.1.3. Stub validator	22
4.1.4. CLAT with DNS proxy and validator	22
4.1.5. ACL of clients	22
4.1.6. Mapping-out IPv4 addresses	23
4.2. DNS64 and Reverse Mapping	23
4.3. Using 464XLAT with/without DNS64	23
4.4. Foreign DNS	24
4.4.1. Manual Configuration of DNS	25
4.4.2. DNS Privacy/Encryption Mechanisms	25
4.4.3. Split DNS and VPNs	26
4.5. Well-Known Prefix (WKP) vs Network-Specific Prefix (NSP)	26
4.6. IPv4 literals and non-IPv6 Compliant APIs	26
4.7. IPv4-only Hosts or Applications	27
4.8. CLAT Translation Considerations	27
4.9. EAM Considerations	28
4.10. Incoming Connections	28
5. Summary of Deployment Recommendations for NAT64/464XLAT	28
6. Deployment of 464XLAT/NAT64 in Enterprise Networks	31
7. Security Considerations	33
8. IANA Considerations	33
9. Acknowledgements	33
10. ANNEX A: Example of Broadband Deployment with 464XLAT	34
11. ANNEX B: CLAT Implementation	37
12. ANNEX C: Benchmarking	38
13. ANNEX D: Changes from -00 to -01/-02	38

14. ANNEX E: Changes from -02 to -03	38
15. ANNEX F: Changes from -03 to -04	39
16. ANNEX G: Changes from -04 to -05	39
17. ANNEX H: Changes from -05 to -06	39
18. ANNEX H: Changes from -06 to -07	39
19. References	39
19.1. Normative References	39
19.2. Informative References	42
Author's Address	45

1. Introduction

Stateful NAT64 ([RFC6146]) describes a stateful IPv6 to IPv4 translation mechanism, which allows IPv6-only hosts to communicate with IPv4-only servers using unicast UDP, TCP, or ICMP, by means of IPv4 public addresses sharing, among multiple IPv6-only hosts. Unless otherwise stated, references in the rest of this document to NAT64 (function) should be interpreted as to Stateful NAT64.

The translation of the packet headers is done using the IP/ICMP translation algorithm defined in [RFC7915] and algorithmically translating the IPv4 addresses to IPv6 addresses and vice versa, following [RFC6052].

DNS64 ([RFC6147]) is in charge of the synthesis of AAAA records from the A records, so only works for applications making use of DNS. It was designed to avoid changes in both, the IPv6-only hosts and the IPv4-only server, so they can use a NAT64 function. As discussed in Section 5.5 of [RFC6147], a security-aware and validating host has to perform the DNS64 function locally.

However, the use of NAT64 and/or DNS64 present three drawbacks:

- a. Because DNS64 ([RFC6147]) modifies DNS answers, and DNSSEC is designed to detect such modifications, DNS64 ([RFC6147]) may potentially break DNSSEC, depending on a number of factors, such as the location of the DNS64 function (at a DNS server or validator, at the end host, ...), how it has been configured, if the end-hosts is validating, etc.
- b. Because the need of using DNS64 ([RFC6147]) or an alternative "host/application built-in" mechanism for address synthesis, there may be an issue for NAT64 ([RFC6146]), as it doesn't work when IPv4 literal addresses or non-IPv6 compliant APIs are being used.
- c. NAT64 alone, was not designed to provide a solution for IPv4-only hosts or applications located within a network which are

connected to a service provider IPv6-only access, as it was designed for a very specific scenario ([RFC6144], Section 2.1).

Above drawbacks may be true if part of, an enterprise network, is connected to other parts of the same network or third-party networks by means of IPv6-only connectivity. This is just an example which may apply to many other similar cases. All them are deployment specific.

According to that, across this document, the use of "operator", "operator network", "service provider", and similar ones, are interchangeable with equivalent cases of enterprise networks (and similar ones). This may be also the case for "managed end-user networks".

Note that if all the hosts in a network were performing the address synthesis, as described in Section 7.2 of [RFC6147], some of the drawbacks may vanish. However, it is unrealistic today to expect that, considering the high number of devices and applications that aren't yet IPv6-enabled. So, in this document, this will be considered only for specific scenarios that can guarantee it.

An analysis of stateful IPv4/IPv6 mechanisms is provided in [RFC6889].

This document looks into different possible NAT64 ([RFC6146]) deployment scenarios, including IPv4-IPv6-IPv4 (464 for short) and similar ones, which were not documented in [RFC6144], such as 464XLAT ([RFC6877]), in operator (broadband and cellular) and enterprise networks, and provides guidelines to avoid operational issues.

Towards that, this document first looks into the possible NAT64 deployment scenarios (split in "known to work" and "known to work under special conditions"), providing a quick and generic comparison table among them. Then the document describes the issues that an operator need to understand on different matters that will allow to define what is the best approach/scenario for each specific network case. A summary provides some recommendations and decision points. A section with clarifications on the usage of this document for enterprise networks, is also provided. Finally, an annex provides an example of a broadband deployment using 464XLAT and another annex provides hints for a CLAT implementation.

[RFC7269] already provides information about NAT64 deployment options and experiences. Both, this document and [RFC7269] are complementary; they are looking into different deployment considerations and furthermore, this document is considering the updated deployment experience and newer standards.

The target deployment scenarios in this document may be covered as well by other IPv4-as-a-Service (IPv4aaS) transition mechanisms. Note that this is true only for the case of broadband networks, as in the case of cellular networks the only supported solution is the use of NAT64/464XLAT. So, it is out of scope of this document to provide a comparison among the different IPv4aaS transition mechanisms, which is being analyzed already in [I-D.lmhp-v6ops-transition-comparison].

Consequently, this document should not be understood as a guide for an operator or enterprise to decide which IPv4aaS is the best one for its own network. Instead it should be used as a tool for understanding all the implications, including relevant documents (or even specific parts of them), for the deployment of NAT64/464XLAT and facilitate the decision process regarding specific deployment details.

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. NAT64 Deployment Scenarios

Section 7 of DNS64 ([RFC6147]), provides three scenarios, depending on the location of the DNS64 function. However, since the publication of that document, other deployment scenarios and NAT64 use cases need to be considered in actual networks, despite some of them were specifically ruled out by the original NAT64/DNS64 work.

Consequently, the perspective in this document is to broaden those scenarios, including a few new ones. However, in order to be able to reduce the number of possible cases, we work under the assumption that typically, the service provider wants to make sure that all the customers have a service without failures. This means considering the following assumptions for the worst possible case:

- a. There are hosts that will be validating DNSSEC.
- b. IPv4 literal addresses and non-IPv6 compliant APIs are being used.
- c. There are IPv4-only hosts or applications beyond the IPv6-only link (e.g., tethering in cellular networks).

The document uses a common set of possible "participant entities":

1. An IPv6-only access network (IPv6).
2. An IPv4-only remote network/server/service (IPv4).
3. A NAT64 function (NAT64) in the service provider.
4. A DNS64 function (DNS64) in the service provider.
5. An external service provider offering the NAT64 function and/or the DNS64 function (extNAT64/extDNS64).
6. 464XLAT customer side translator (CLAT).

Note that the nomenclature used in parenthesis is the one that, for short, will be used in the figures. Note also that for simplicity, the boxes in the figures don't mean they are actually a single device; they just represent one or more functions as located in that part of the network (i.e. a single box with NAT64 and DNS64 functions can actually be several devices, not just one).

The possible scenarios are split in two general categories:

1. Known to work.
 2. Known to work under special conditions.
- 3.1. Known to Work

The scenarios in this category are known to work, as there are well-known existing deployments from different operators using them. Each one may have different pros and cons, and in some cases the trade-offs, maybe acceptable for some operators.

3.1.1. Service Provider NAT64 with DNS64

In this scenario (Figure 1), the service provider offers both, the NAT64 and the DNS64 functions.

This is the most common scenario as originally considered by the designers of NAT64 ([RFC6146]) and DNS64 ([RFC6147]), however also may have the implications related the DNSSEC.

This scenario also may fail to solve the issue of IPv4 literal addresses or non-IPv6 compliant APIs, as well as the issue of IPv4-only hosts or applications behind the IPv6-only access network.

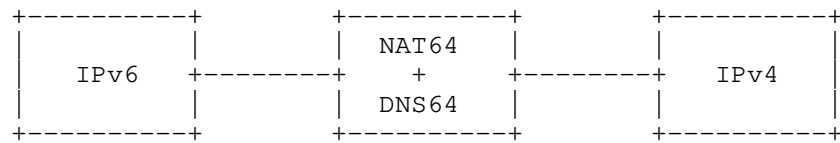


Figure 1: NAT64 with DNS64

A similar scenario (Figure 2) will be if the service provider offers only the DNS64 function, and the NAT64 function is provided by an outsourcing agreement with an external provider. All the considerations in the previous paragraphs of this section, are the same for this sub-case.

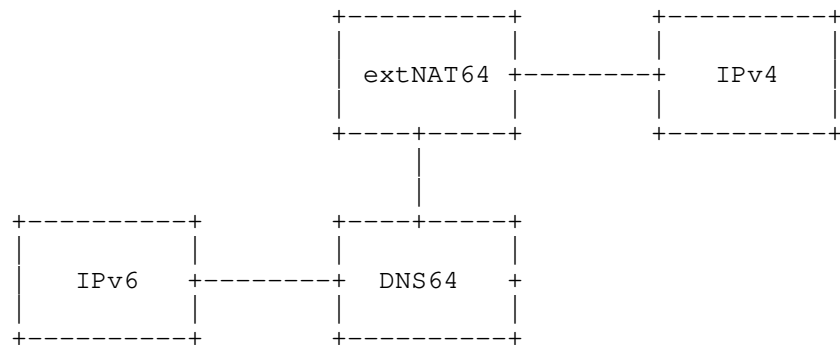


Figure 2: NAT64 in external service provider

This is equivalent to the scenario (Figure 3) where the outsourcing agreement with the external provider is to provide both the NAT64 and DNS64 functions. Once more, all the considerations in the previous paragraphs of this section are the same for this sub-case.

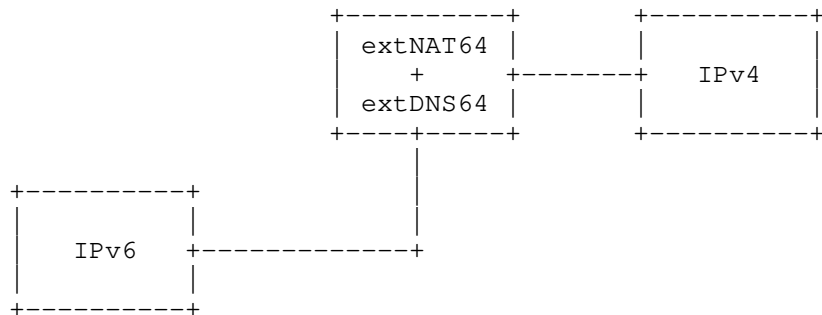


Figure 3: NAT64 and DNS64 in external provider

One additional equivalent scenario (Figure 4) will be if the service provider offers the NAT64 function only, and the DNS64 function is from an external provider with or without a specific agreement among them. This is a scenario already common today, as several "global" service providers provide free DNS/DNS64 services and users often configure manually their DNS. This will only work if both the NAT64 and the DNS64 functions are using the WKP (Well-Known Prefix) or the same NSP (Network-Specific Prefix). All the considerations in the previous paragraphs of this section, are the same for this sub-case.

Of course, if the external DNS64 function is agreed with the service provider, then we are in the same case as in the previous ones already depicted in this scenario.

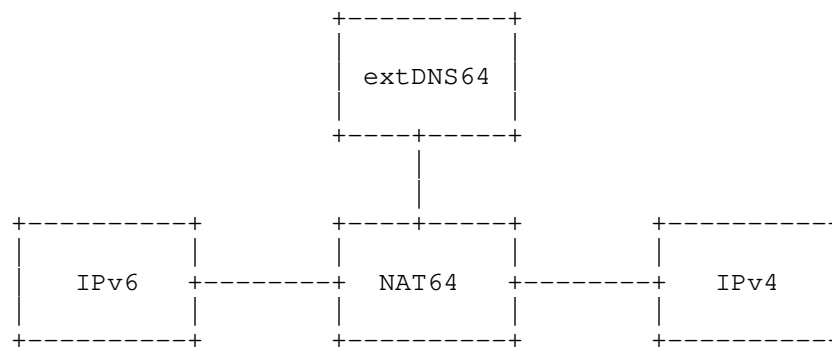


Figure 4: NAT64; DNS64 by external provider

3.1.2. Service Provider Offering 464XLAT, with DNS64

464XLAT ([RFC6877]) describes an architecture that provides IPv4 connectivity across a network, or part of it, when it is only natively transporting IPv6. [RFC7849] already suggest the need to support the CLAT function in order to ensure the IPv4 service continuity in IPv6-only cellular deployments.

In order to do that, 464XLAT ([RFC6877]) relies on the combination of existing protocols:

1. The customer-side translator (CLAT) is a stateless IPv4 to IPv6 translator (NAT46) ([RFC7915]) implemented in the end-user device or CE (Customer Edge Router), located at the "customer edge" of the network.
2. The provider-side translator (PLAT) is a stateful NAT64 ([RFC6146]), implemented typically at in the operator network.

3. Optionally, DNS64 ([RFC6147]), may allow an optimization: a single translation at the NAT64, instead of two translations (NAT46+NAT64), when the application at the end-user device supports IPv6 DNS (uses AAAA Resource Records).

Note that even if in the 464XLAT ([RFC6877]) terminology, the provider-side translator is referred as PLAT, for simplicity and uniformity, across this document is always referred as NAT64 (function).

In this scenario (Figure 5) the service provider deploys 464XLAT with a DNS64 function.

As a consequence, the DNSSEC issues remain, unless the host is doing the address synthesis.

464XLAT ([RFC6877]) is a very simple approach to cope with the major NAT64+DNS64 drawback: Not working with applications or devices that use literal IPv4 addresses or non-IPv6 compliant APIs.

464XLAT ([RFC6877]) has been used initially mainly in IPv6-only cellular networks. By supporting a CLAT function, the end-user device applications can access IPv4-only end-networks/applications, despite those applications or devices use literal IPv4 addresses or non-IPv6 compliant APIs.

In addition to that, in the same example of the cellular network above, if the User Equipment (UE) provides tethering, other devices behind it will be presented with a traditional NAT44, in addition to the native IPv6 support, so clearly it allows IPv4-only hosts behind the IPv6-only access network.

Furthermore, as discussed in [RFC6877], 464XLAT can be used in broadband IPv6 network architectures, by implementing the CLAT function at the CE.

The support of this scenario in a network, offers two additional advantages:

- o DNS load optimization: A CLAT should implement a DNS proxy (as per [RFC5625]), so that only IPv6 native queries and only for AAAA records are sent to the DNS64 server. Otherwise doubling the number of queries may impact the DNS infrastructure.
- o Connection establishment delay optimization: If the UE/CE implementation is detecting the presence of a DNS64 function, it may issue only the AAAA query, instead of both the AAAA and A queries.

In order to understand all the communication possibilities, let's assume the following representation of two dual-stack peers:

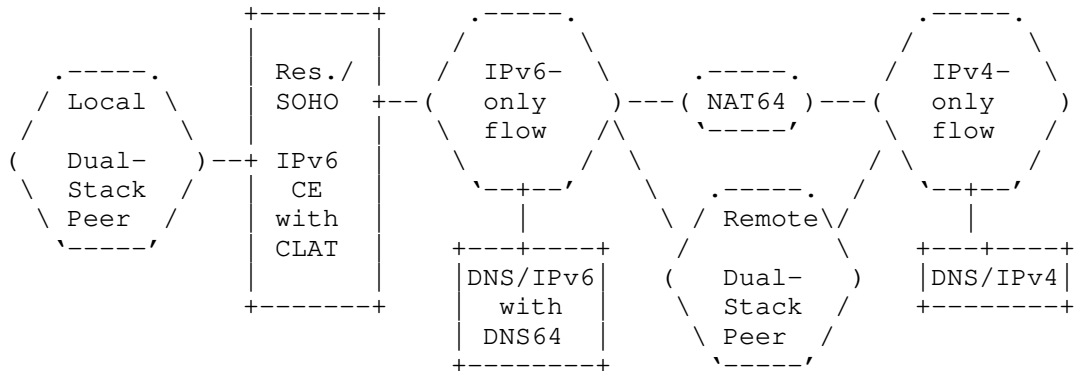


Figure A: Representation of 464XLAT among two peers with DNS64

The possible communication paths, among the IPv4/IPv6 stacks of both peers, in this case, are:

- Local-IPv6 to Remote-IPv6: Regular DNS and native IPv6 among peers.
- Local-IPv6 to Remote-IPv4: DNS64 and NAT64 translation.
- Local-IPv4 to Remote-IPv6: Not possible unless the CLAT implements EAM (Explicit Address Mappings) as indicated by Section 4.9. In principle, it is not expected that services are deployed in Internet using IPv6-only, unless there is certainty that peers will also be IPv6-capable.
- Local-IPv4 to Remote-IPv4: DNS64, CLAT and NAT64 translations.
- Local-IPv4 to Remote-dual-stack using EAM optimization: If the CLAT implements EAM as indicated by Section 4.9, instead of using the path d. above, NAT64 translation is avoided and the flow will use IPv6 from the CLAT to the destination.

The rest of the figures in this section show different choices for placing the different elements.

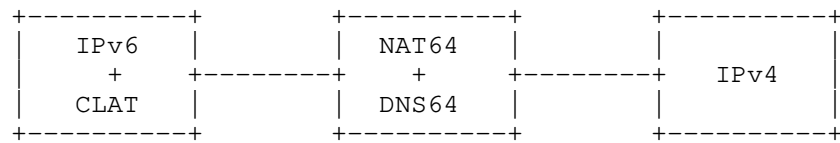


Figure 5: 464XLAT with DNS64

A similar scenario (Figure 6) will be if the service provider offers only the DNS64 function, and the NAT64 function is provided by an outsourcing agreement with an external provider. All the considerations in the previous paragraphs of this section are the same for this sub-case.

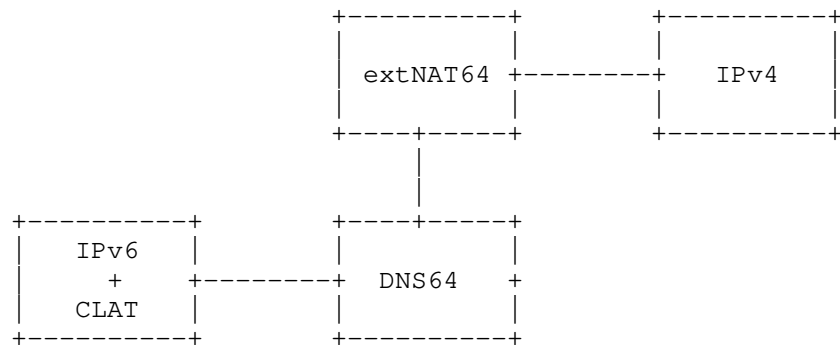


Figure 6: 464XLAT with DNS64; NAT64 in external provider

As well, is equivalent to the scenario (Figure 7) where the outsourcing agreement with the external provider is to provide both the NAT64 and DNS64 functions. Once more, all the considerations in the previous paragraphs of this section are the same for this sub-case.

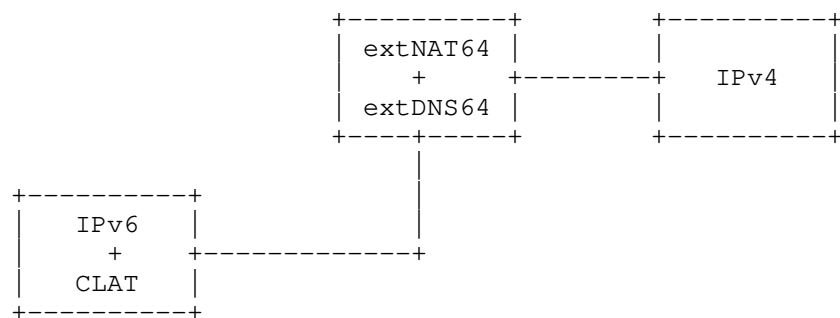


Figure 7: 464XLAT with DNS64; NAT64 and DNS64 in external provider

3.1.3. Service Provider Offering 464XLAT, without DNS64

The major advantage of this scenario (Figure 8), using 464XLAT without DNS64, is that the service provider ensures that DNSSEC is never broken, even in case the user modifies the DNS configuration. Nevertheless, some CLAT implementations or applications may impose an extra delay, which is induced by the dual A/AAAA queries (and wait for both responses), unless Happy Eyeballs v2 ([RFC8305]) is also present.

A possible variation of this scenario is the case when DNS64 is used only for the discovery of the NAT64 prefix. The rest of the document is not considering it as a different scenario, because once the prefix has been discovered, the DNS64 function is not used, so it behaves as if the DNS64 synthesis function is not present.

In this scenario, as in the previous one, there are no issues related to IPv4-only hosts (or IPv4-only applications) behind the IPv6-only access network, neither related to the usage of IPv4 literals or non-IPv6 compliant APIs.

The support of this scenario in a network, offers one advantage:

- o DNS load optimization: A CLAT should implement a DNS proxy (as per [RFC5625]), so that only IPv6 native queries are sent to the DNS64 server. Otherwise doubling the number of queries may impact the DNS infrastructure.

As indicated earlier, the connection establishment delay optimization is achieved only in the case of devices, Operating Systems, or applications that use Happy Eyeballs v2 ([RFC8305]), which is very common.

Let's assume the representation of two dual-stack peers as in the previous case:

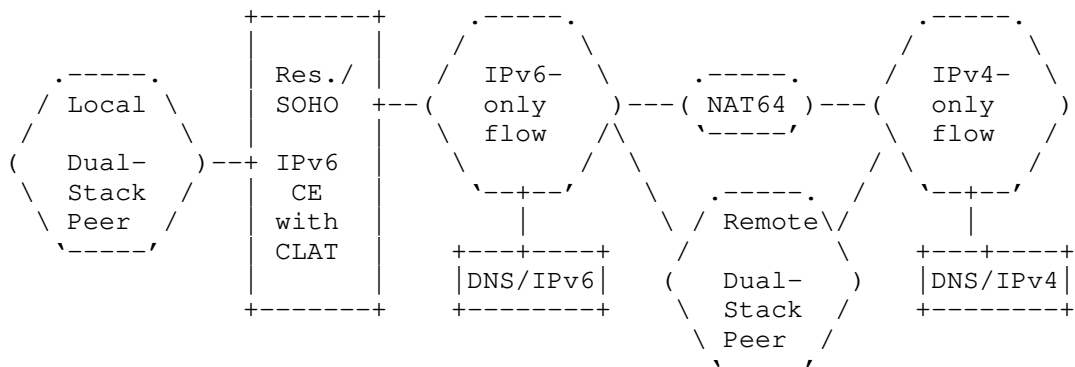


Figure B: Representation of 464XLAT among two peers without DNS64

The possible communication paths, among the IPv4/IPv6 stacks of both peers, in this case, are:

- a. Local-IPv6 to Remote-IPv6: Regular DNS and native IPv6 among peers.
- b. Local-IPv6 to Remote-IPv4: Regular DNS, CLAT and NAT64 translations.
- c. Local-IPv4 to Remote-IPv6: Not possible unless the CLAT implements EAM as indicated by Section 4.9. In principle, it is not expected that services are deployed in Internet using IPv6-only, unless there is certainty that peers will also be IPv6-capable.
- d. Local-IPv4 to Remote-IPv4: Regular DNS, CLAT and NAT64 translations.
- e. Local-IPv4 to Remote-dual-stack using EAM optimization: If the CLAT implements EAM as indicated by Section 4.9, instead of using the path d. above, NAT64 translation is avoided and the flow will use IPv6 from the CLAT to the destination.

It needs to be noticed that this scenario works while the local hosts/applications are dual-stack (which is the current situation), because the connectivity from a local-IPv6 to a remote-IPv4 is not possible without an AAAA synthesis. This aspect is important only when in the LANs behind the CLAT there are IPv6-only hosts and they need to communicate with remote IPv4-only hosts. However, it doesn't look a sensible approach from an Operating System or application vendor perspective, to provide IPv6-only support unless, similarly to case c above, there is certainty of peers supporting IPv6 as well. A

solution approach to this is also presented in [I-D.palet-v6ops-464xlat-opt-cdn-caches].

The following figures show different choices for placing the different elements.

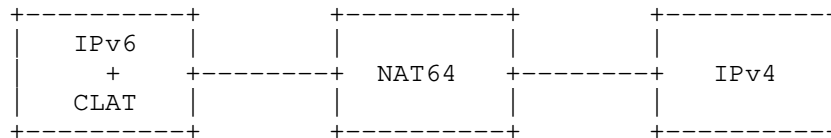


Figure 8: 464XLAT without DNS64

This is equivalent to the scenario (Figure 9) where there is an outsourcing agreement with an external provider for the NAT64 function. All the considerations in the previous paragraphs of this section are the same for this sub-case.

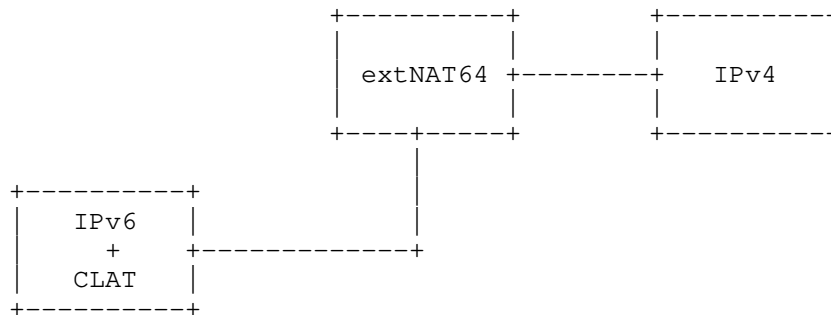


Figure 9: 464XLAT without DNS64; NAT64 in external provider

3.2. Known to Work Under Special Conditions

The scenarios in this category are known to not work unless significant effort is devoted to solve the issues, or are intended to solve problems across "closed" networks, instead of as a general Internet access usage. In addition to the different pros, cons and trade-offs, which may be acceptable for some operators, they have implementation difficulties, as they are beyond the original expectations of the NAT64/DNS64 original intent.

3.2.1. Service Provider NAT64 without DNS64

In this scenario (Figure 10), the service provider offers a NAT64 function, however there is no DNS64 function support at all.

As a consequence, an IPv6 host in the IPv6-only access network, will not be able to detect the presence of DNS64 by means of [RFC7050], neither to learn the IPv6 prefix to be used for the NAT64 function.

This can be sorted out as indicated in Section 4.1.1.

However, despite that, because the lack of the DNS64 function, the IPv6 host will not be able to obtain AAAA synthesized records, so the NAT64 function becomes useless.

An exception to this "useless" scenario will be manually configure mappings between the A records of each of the IPv4-only remote hosts and the corresponding AAAA records, with the WKP (Well-Known Prefix) or NSP (Network-Specific Prefix) used by the service provider NAT64 function, as if they were synthesized by a DNS64 function.

This mapping could be done by several means, typically at the authoritative DNS server, or at the service provider resolvers by means of DNS RPZ (Response Policy Zones, [I-D.vixie-dns-rpz]) or equivalent functionality. DNS RPZ, may have implications in DNSSEC, if the zone is signed. Also, if the service provider is using an NSP, having the mapping at the authoritative server, may create troubles to other parties trying to use different NSP or the WKP, unless multiple DNS "views" (split-DNS) is also being used at the authoritative servers.

Generally, the mappings alternative, will only make sense if a few set of IPv4-only remote hosts need to be accessed by a single network (or a small number of them), which support IPv6-only in the access. This will require some kind of mutual agreement for using this procedure, so it doesn't care if they become a trouble for other parties across Internet ("closed services").

In any case, this scenario doesn't solve the issue of IPv4 literal addresses or non-IPv6 compliant APIs, neither it solves the problem of IPv4-only hosts within that IPv6-only access network.

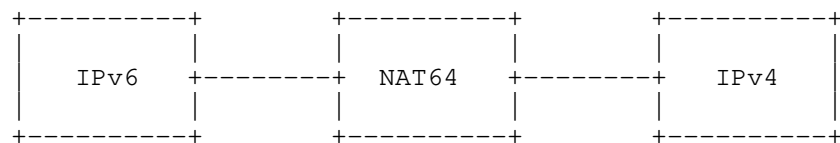


Figure 10: NAT64 without DNS64

3.2.2. Service Provider NAT64; DNS64 in the IPv6 hosts

In this scenario (Figure 11), the service provider offers the NAT64 function, but not the DNS64 function. However, the IPv6 hosts have a built-in DNS64 function.

This may become common if the DNS64 function is implemented in all the IPv6 hosts/stacks. However, commonly this is not the actual situation, even if it may happen in the medium-term. At this way, the DNSSEC validation is performed on the A record, and then the host can use the DNS64 function so to be able to use the NAT64 function, without any DNSSEC issues.

This scenario fails to solve the issue of IPv4 literal addresses or non-IPv6 compliant APIs, unless the IPv6 hosts also supports Happy Eyeballs v2 ([RFC8305], Section 7.1), which may solve that issue.

However, this scenario still fails to solve the problem of IPv4-only hosts or applications behind the IPv6-only access network.



Figure 11: NAT64; DNS64 in IPv6 hosts

3.2.3. Service Provider NAT64; DNS64 in the IPv4-only remote network

In this scenario (Figure 12), the service provider offers the NAT64 function only. The remote IPv4-only network offers the DNS64 function.

This is not common, and looks like doesn't make too much sense that a remote network, not deploying IPv6, is providing a DNS64 function. As in the case of the scenario depicted in Section 3.2.1, it will only work if both sides are using the WKP or the same NSP, so the same considerations apply. It can be also tuned to behave as in Section 3.1.1

This scenario still fails to solve the issue of IPv4 literal addresses or non-IPv6 compliant APIs.

This scenario also fails to solve the problem of IPv4-only hosts or applications behind the IPv6-only access network.



Figure 12: NAT64; DNS64 in the IPv4-only

3.3. Comparing the Scenarios

This section compares the different scenarios, including the possible variations (each one represented in the precedent sections by a different figure), looking at the following criteria:

- a. DNSSEC: Are there hosts validating DNSSEC?
- b. Literal/APIs: Are there applications using IPv4 literals or non-IPv6 compliant APIs?
- c. IPv4-only: Are there hosts or applications using IPv4-only?
- d. Foreign DNS: Is the scenario surviving if the user, Operating System, applications or devices change the DNS?
- e. DNS load opt. (DNS load optimization): Are there extra queries that may impact DNS infrastructure?
- f. Connect. opt. (Connection establishment delay optimization): Is the UE/CE issuing only the AAAA query or also an A query and waiting for both responses?

In the next table, the columns represent each of the scenarios from the previous sections, by the figure number. The possible values are:

- o "-" Scenario "bad" for that criteria.
- o "+" Scenario "good" for that criteria.
- o "*" Scenario "bad" for that criteria, however it is typically resolved, with the support of Happy Eyeballs v2 ([RFC8305]).

In some cases, "countermeasures", alternative or special configurations, may be available for the criteria designated as "bad". So, this comparison is considering a generic case, as a quick comparison guide. In some cases, a "bad" criterion is not necessarily a negative aspect, all it depends on the specific needs/ characteristics of the network where the deployment will take place.

For instance, in a network which has only IPv6-only hosts and apps using only DNS and IPv6-compliant APIs, there is no impact using only NAT64 and DNS64, but if the hosts may validate DNSSEC, that item is still relevant.

Item / Figure	1	2	3	4	5	6	7	8	9	10	11	12
DNSSEC	-	-	-	-	-	-	-	+	+	+	+	+
Literal/APIs	-	-	-	-	+	+	+	+	+	-	-	-
IPv4-only	-	-	-	-	+	+	+	+	+	-	-	-
Foreign DNS	-	-	-	-	+	+	+	+	+	-	+	-
DNS load opt.	+	+	+	+	+	+	+	+	+	+	+	+
Connect. opt.	+	+	+	+	+	+	+	*	*	+	+	+

Figure 13: Scenario Comparison

As a general conclusion, we should note that, if the network must support applications using any of the following:

- o IPv4 literals
- o non-IPv6-compliant APIs
- o IPv4-only hosts or applications

Then, only the scenarios with 464XLAT, a CLAT function, or equivalent built-in local address synthesis features, will provide a valid solution. Further to that, those scenarios will also keep working if the DNS configuration is modified. Clearly also, depending on if DNS64 is used or not, DNSSEC may be broken for those hosts doing DNSSEC validation.

All the scenarios are good in terms of DNS load optimization, and in the case of 464XLAT it may provide an extra degree of optimization. Finally, all them are also good in terms of connection establishment delay optimization. However, in the case of 464XLAT without DNS64, it requires the usage of Happy Eyeballs v2. This is not an issue, as commonly it is available in actual Operating Systems.

4. Issues to be Considered

This section reviews the different issues that an operator needs to consider towards a NAT64/464XLAT deployment, as they may bring to specific decision points about how to approach that deployment.

4.1. DNSSEC Considerations and Possible Approaches

As indicated in Section 8 of [RFC6147] (DNS64, Security Considerations), because DNS64 modifies DNS answers and DNSSEC is designed to detect such modifications, DNS64 may break DNSSEC.

If a device connected to an IPv6-only access network, queries for a domain name in a signed zone, by means of a recursive name server that supports DNS64, and the result is a synthesized AAAA record, and the recursive name server is configured to perform DNSSEC validation and has a valid chain of trust to the zone in question, it will cryptographically validate the negative response from the authoritative name server. This is the expected DNS64 behavior: The recursive name server actually "lies" to the client device. However, in most of the cases, the client will not notice it, because generally, they don't perform validation themselves and instead, rely on the recursive name servers.

A validating DNS64 resolver in fact, increase the confidence on the synthetic AAAA, as it has validated that a non-synthetic AAAA for sure, doesn't exist. However, if the client device is NAT64-oblivious (most common case) and performs DNSSEC validation on the AAAA record, it will fail as it is a synthesized record.

The best possible scenario from DNSSEC point of view, is when the client requests the DNS64 server to perform the DNSSEC validation (by setting the DO bit to 1 and the CD bit to 0). In this case, the DNS64 server validates the data, thus tampering may only happen inside the DNS64 server (which is considered as a trusted part, thus its likelihood is low) or between the DNS64 server and the client. All other parts of the system (including transmission and caching) are protected by DNSSEC ([Threat-DNS64]).

Similarly, if the client querying the recursive name server is another name server configured to use it as a forwarder, and is performing DNSSEC validation, it will also fail on any synthesized AAAA record.

All those considerations are extensively covered in Sections 3, 5.5 and 6.2 of [RFC6147].

A solution to avoid DNSSEC issues, will be that all the signed zones

also provide IPv6 connectivity, together with the corresponding AAAA records. However, this is out of the control of the operator needing to deploy a NAT64 function. This has been proposed already in [I-D.bp-v6ops-ipv6-ready-dns-dnssec].

An alternative solution, which was the one considered while developing [RFC6147], is that validators will be DNS64-aware, so could perform the necessary discovery and do their own synthesis. That was done under the expectation that it was sufficiently early in the validator-deployment curve that it would be ok to break certain DNSSEC assumptions for networks who were really stuck in a NAT64/DNS64-needing world.

As already indicated, the scenarios in the previous section, are in fact somehow simplified, looking at the worst possible case. Saying it in a different way: "trying to look for the most perfect approach". DNSSEC breach will not happen if the end-host is not doing validation.

Existing previous studies seems to indicate that the figures of DNSSEC actually broken by using DNS64 will be around 1.7% ([About-DNS64]) of the cases. However, we can't negate that this may increase, as DNSSEC deployment grows. Consequently, a decision point for the operator must depend on "do I really care for that percentage of cases and the impact in my helpdesk or can I provide alternative solutions for them?". Some possible solutions may be taken, as depicted in the next sections.

4.1.1. Not using DNS64

A solution will be to avoid using DNS64, but as already indicated this is not possible in all the scenarios.

The use of DNS64 is a key component for some networks, in order to comply with traffic performance metrics, monitored by some governmental bodies and other institutions ([FCC], [ARCEP]).

One drawback of not having a DNS64 at the network side, is that is not possible to heuristically discover the NAT64 ([RFC7050]). Consequently, an IPv6 host behind the IPv6-only access network, will not be able to detect the presence of the NAT64 function, neither to learn the IPv6 prefix to be used for it, unless it is configured by alternative means.

The discovery of the IPv6 prefix could be solved, as described in [RFC7050], by means of adding the relevant AAAA records to the ipv4only.arpa. zone, of the service provider recursive servers, i.e., if using the WKP (64:ff9b::/96):

```
ipv4only.arpa. SOA      . . 0 0 0 0 0
ipv4only.arpa. NS       .
ipv4only.arpa. AAAA     64:ff9b::192.0.0.170
ipv4only.arpa. AAAA     64:ff9b::192.0.0.171
ipv4only.arpa. A        192.0.0.170
ipv4only.arpa. A        192.0.0.171
```

An alternative option to the above, is the use of DNS RPZ ([I-D.vixie-dns-rpz]) or equivalent functionalities. Note that this may impact DNSSEC if the zone is signed.

One more alternative, only valid in environments with PCP support (for both the hosts or CEs and for the service provider network), is to follow [RFC7225] (Discovering NAT64 IPv6 Prefixes using PCP).

Other alternatives may be available in the future. All them are extensively discussed in [RFC7051], however the deployment evolution has evolved many considerations from that document. New options are being documented, such using Router Advertising ([I-D.ietf-6man-ra-pref64]) or DHCPv6 options ([I-D.li-intarea-nat64-prefix-dhcp-option]).

It may be convenient the simultaneous support of several of the possible approaches, in order to ensure that clients with different ways to configure the NAT64 prefix, successfully obtain it. This is also convenient even if DNS64 is being used.

Of special relevance to this section is also [I-D.cheshire-sudn-ipv4only-dot-arpa].

4.1.2. DNSSEC validator aware of DNS64

In general, by default, DNS servers with DNS64 function will not synthesize AAAA responses if the DNSSEC OK (DO) flag was set in the query.

In this case, as only an A record is available, if a CLAT function is present, it means that the CLAT will take the responsibility, as in the case of literal IPv4 addresses, to keep that traffic flow end-to-end as IPv4, so DNSSEC is not broken.

However, this will not work if a CLAT function is not present as the hosts will not be able to use IPv4 (which is the case for all the scenarios without 464XLAT).

4.1.3. Stub validator

If the DO flag is set and the client device performs DNSSEC validation, and the Checking Disabled (CD) flag is set for a query, the DNS64 recursive server will not synthesize AAAA responses. In this case, the client could perform the DNSSEC validation with the A record and then synthesize the AAAA ([RFC6052]). For that to be possible, the client must have learned beforehand the NAT64 prefix using any of the available methods ([RFC7050], [RFC7225], [I-D.ietf-6man-ra-pref64], [I-D.li-intarea-nat64-prefix-dhcp-option]). This allows the client device to avoid using the DNS64 function and still use NAT64 even with DNSSEC.

If the end-host is IPv4-only, this will not work if a CLAT function is not present (scenarios without 464XLAT).

Some devices or Operating Systems may implement, instead of a CLAT, an equivalent function by using Bump-in-the-Host ([RFC6535]), implemented as part of Happy Eyeballs v2 (Section 7.1 of [RFC8305]). In this case, the considerations in the above paragraphs are also applicable.

4.1.4. CLAT with DNS proxy and validator

If a CE includes CLAT support and also a DNS proxy, as indicated in Section 6.4 of [RFC6877], the CE could behave as a stub validator on behalf of the client devices. Then, following the same approach described in the Section 4.1.3, the DNS proxy actually will "lie" to the client devices, which in most of the cases will not notice it, unless they perform validation by themselves. Again, this allow the client devices to avoid using the DNS64 function and still use NAT64 with DNSSEC.

Once more, this will not work without a CLAT function (scenarios without 464XLAT).

4.1.5. ACL of clients

In cases of dual-stack clients, the AAAA queries typically take preference over A queries. If DNS64 is enabled for those clients, will never get A records, even for IPv4-only servers.

As a consequence, in cases where there are IPv4-only servers, and those are located in the path before the NAT64 function, the clients will not be able to reach them. If DNSSEC is being used for all those flows, specific addresses or prefixes can be left-out of the DNS64 synthesis by means of ACLs.

Once more, this will not work without a CLAT function (scenarios without 464XLAT).

4.1.6. Mapping-out IPv4 addresses

If there are well-known specific IPv4 addresses or prefixes using DNSSEC, they can be mapped-out of the DNS64 synthesis.

Even if this is not related to DNSSEC, this "mapping-out" feature is actually, quite commonly used to ensure that [RFC1918] addresses (for example used by LAN servers) are not synthesized to AAAA.

Once more, this will not work without a CLAT function (scenarios without 464XLAT).

4.2. DNS64 and Reverse Mapping

When a client device, using DNS64 tries to reverse-map a synthesized IPv6 address, the name server responds with a CNAME record pointing the domain name used to reverse-map the synthesized IPv6 address (the one under ip6.arpa), to the domain name corresponding to the embedded IPv4 address (under in-addr.arpa).

This is the expected behavior, so no issues need to be considered regarding DNS reverse mapping.

4.3. Using 464XLAT with/without DNS64

In the case the client device is IPv6-only (either because the stack or application is IPv6-only, or because it is connected via an IPv6-only LAN) and the remote server is IPv4-only (either because the stack is IPv4-only, or because it is connected via an IPv4-only LAN), only NAT64 combined with DNS64 will be able to provide access among both. Because DNS64 is then required, DNSSEC validation will be only possible if the recursive name server is validating the negative response from the authoritative name server and the client is not performing validation.

Note that is not expected at this stage of the transition, that applications, devices or Operating Systems are IPv6-only. It will not be a sensible decision for a developer to work on that direction, unless it is clear that the deployment scenario fully supports it.

On the other hand, an end-user or enterprise network may decide to run IPv6-only in the LANs. In case there is any chance for applications to be IPv6-only, the Operating System may be responsible either for doing a local address synthesis, or alternatively, setting up some kind of on-demand VPN (IPv4-in-IPv6), which need to be

supported by that network. This may become very common in enterprise networks, where "Unique IPv6 Prefix per Host" [RFC8273] is supported.

However, when the client device is dual-stack and/or connected in a dual-stack LAN by means of a CLAT function (or has a built-in CLAT function), DNS64 is an option.

1. With DNS64: If DNS64 is used, most of the IPv4 traffic (except if using literal IPv4 addresses or non-IPv6 compliant APIs) will not use the CLAT, so will use the IPv6 path and only one translation will be done at the NAT64. This may break DNSSEC, unless measures as described in the precedent sections are taken.
2. Without DNS64: If DNS64 is not used, all the IPv4 traffic will make use of the CLAT, so two translations are required (NAT46 at the CLAT and NAT64 at the PLAT), which adds some overhead in terms of the extra NAT46 translation. However, this avoids the AAAA synthesis and consequently will never break DNSSEC.

Note that the extra translation, when DNS64 is not used, takes place at the CLAT, which means no extra overhead for the operator. It however adds potential extra delays to establish the connections, and no perceptible impact for a CE in a broadband network, while it may have some impact in a battery powered device. This cost for a battery powered device, is possibly comparable to the cost when the device is doing a local address synthesis (see Section 7.1 of [RFC8305]).

4.4. Foreign DNS

Clients, devices or applications in a service provider network, may use DNS servers from other networks. This may be the case either if individual applications use their own DNS server, the Operating System itself or even the CE, or combinations of the above.

Those "foreign" DNS servers may not support DNS64, which as a consequence, will mean that those scenarios that require a DNS64 may not work. However, if a CLAT function is available, the considerations in Section 4.3 will apply.

In the case that the foreign DNS supports the DNS64 function, we may be in the situation of providing incorrect configurations parameters, for example, un-matching WKP or NSP, or a case such the one described in Section 3.2.3.

Having a CLAT function, even if using foreign DNS without a DNS64 function, ensures that everything will work, so the CLAT must be considered as an advantage even against user configuration errors.

The cost of this, is that all the traffic will use a double translation (NAT46 at the CLAT and NAT64 at the operator network), unless there is support for EAM (Section 4.9).

An exception to that is the case when there is a CLAT function at the CE, which is not able to obtain the correct configuration parameters (again, un-matching WKP or NSP).

However, it needs to be emphasized, that if there is not a CLAT function (scenarios without 464XLAT), an external DNS without DNS64 support, will disallow any access to IPv4-only destination networks, and will not guarantee the correct DNSSEC validation, so will behave as in the Section 3.2.1.

In summary, it can be said, that the consequences of the use of foreign DNS depend very much in each specific case. However, in general, if a CLAT function is present, most of the time, there will not be any. In the other cases, generally, the access to IPv6-enabled services is still guaranteed for IPv6-enabled hosts, but not for IPv4-only hosts, neither the access to IPv4-only services for any hosts in the network.

The causes of "foreign DNS" could be classified in three main categories, as depicted in the following sub-sections.

4.4.1. Manual Configuration of DNS

It is becoming increasingly common that end-users or even devices or applications configure alternative DNS in their Operating Systems, and sometimes in CEs.

4.4.2. DNS Privacy/Encryption Mechanisms

Clients or applications may use mechanisms for DNS privacy/encryption, such as DNS over TLS ([RFC7858]), DNS over DTLS ([RFC8094]), DNS queries over HTTPS ([RFC8484]) or DNS over QUIC ([I-D.huitema-quic-dnsquic]). Those are commonly cited as DoT, DoH and DoQ.

Those DNS privacy/encryption options, currently are typically provided by the applications, not the Operating System vendors. At the time of writing this document, at least DoT and DoH standards have declared DNS64 (and consequently NAT64) out of their scope, so an application using them may break NAT64, unless a correctly configured CLAT function is used.

4.4.3. Split DNS and VPNs

When networks or hosts use "split-DNS" (also called Split Horizon, DNS views or private DNS), the successful use of the DNS64 is not guaranteed. Section 4 of [RFC6950], analyses this case.

A similar situation may happen in case of VPNs that force all the DNS queries through the VPN, ignoring the operator DNS64 function.

4.5. Well-Known Prefix (WKP) vs Network-Specific Prefix (NSP)

Section 3 of [RFC6052] (IPv6 Addressing of IPv4/IPv6 Translators), discusses some considerations which are useful to decide if an operator should use the WKP or an NSP.

Taking in consideration that discussion and other issues, we can summarize the possible decision points as:

- a. The WKP MUST NOT be used to represent non-global IPv4 addresses. If this is required because the network to be translated use non-global addresses, then an NSP is required.
- b. The WKP MAY appear in inter-domain routing tables, if the operator provides a NAT64 function to peers. However, in this case, special considerations related to BGP filtering are required and IPv4-embedded IPv6 prefixes longer than the WKP MUST NOT be advertised (or accepted) in BGP. An NSP may be a more appropriate option in those cases.
- c. If several NAT64 use the same prefix, packets from the same flow may be routed to different NAT64 in case of routing changes. This can be avoided either by using different prefixes for each NAT64 function, or by ensuring that all the NAT64 coordinate their state. Using an NSP could simplify that.
- d. If DNS64 is required and users, devices, Operating Systems or applications may change their DNS configuration, and deliberately choose an alternative DNS64 function, most probably alternative DNS64 will use by default the WKP. In that case, if an NSP is used by the NAT64 function, clients will not be able to use the operator NAT64 function, which will break connectivity to IPv4-only destinations.

4.6. IPv4 literals and non-IPv6 Compliant APIs

A host or application using literal IPv4 addresses or older APIs, which aren't IPv6 compliant, behind a network with IPv6-only access, will not work unless any of the following alternatives is provided:

- o CLAT (or equivalent function).
- o Happy Eyeballs v2 (Section 7.1, [RFC8305]).
- o Bump-in-the-Host ([RFC6535]) with a DNS64 function.

Those alternatives will solve the problem for an end-host. However, if that end-hosts is providing "tethering" or an equivalent service to other hosts, that needs to be considered as well. In other words, in a case of a cellular network, it resolves the issue for the UE itself, but may be not the case for hosts behind it.

Otherwise, the support of 464XLAT is the only valid and complete approach to resolve this issue.

4.7. IPv4-only Hosts or Applications

An IPv4-only hosts or application behind a network with IPv6-only access, will not work unless a CLAT function is present.

464XLAT is the only valid approach to resolve this issue.

4.8. CLAT Translation Considerations

As described in Section 6.3 of [RFC6877] (IPv6 Prefix Handling), if the CLAT function can be configured with a dedicated /64 prefix for the NAT46 translation, then it will be possible to do a more efficient stateless translation.

Otherwise, if this dedicated prefix is not available, the CLAT function will need to do a stateful translation, for example performing stateful NAT44 for all the IPv4 LAN packets, so they appear as coming from a single IPv4 address, and then in turn, stateless translated to a single IPv6 address.

A possible setup, in order to maximize the CLAT performance, is to configure the dedicated translation prefix. This can be easily achieved automatically, if the broadband CE or end-user device is able to obtain a shorter prefix by means of DHCPv6-PD ([RFC8415]), or other alternatives. The CE can then use a specific /64 for the translation. This is also possible when broadband is provided by a cellular access.

The above recommendation is often not possible for cellular networks, when connecting smartphones (as UEs), as generally they don't use DHCPv6-PD ([RFC8415]). Instead, a single /64 is provided for each PDP context and prefix sharing ([RFC6877]) is used. So, in this case, the UEs typically have a build-in CLAT function which is

performing a stateful NAT44 translation before the stateless NAT46.

4.9. EAM Considerations

Explicit Address Mappings for Stateless IP/ICMP Translation ([RFC7757]) provide a way to configure explicit mappings between IPv4 and IPv6 prefixes of any length. When this is used, for example in a CLAT function, it may provide a simple mechanism in order to avoid traffic flows between IPv4-only nodes or applications and dual-stack destinations to be translated twice (NAT46 and NAT64), by creating mapping entries with the GUA of the IPv6-reachable destination. This optimization of the NAT64 usage is very useful in many scenarios, including CDNs and caches, as described in [I-D.palet-v6ops-464xlat-opt-cdn-caches].

In addition to that, it may provide as well a way for IPv4-only nodes or applications to communicate with IPv6-only destinations.

4.10. Incoming Connections

The use of NAT64, in principle, disallows IPv4 incoming connections, which may be still needed for IPv4-only peer-to-peer applications. However, there are several alternatives that resolve this issue:

- a. STUN ([RFC5389]), TURN ([RFC5766]) and ICE ([RFC8445]) are commonly used by peer-to-peer applications in order to allow incoming connections with IPv4 NAT. In the case of NAT64, they work as well. RFC editor note: If in time, replace STUN and TURN with [I-D.ietf-tram-stunbis] / [I-D.ietf-tram-turnbis].
- b. PCP ([RFC6887]) allows a host to control how incoming IPv4 and IPv6 packets are translated and forwarded. A NAT64 may implement PCP to allow this service.
- c. EAM ([RFC7757]) may also be used in order to configure explicit mappings for customers that require them. This is used for example by SIIT-DC ([RFC7755]) and SIIT-DC-DTM ([RFC7756]).

5. Summary of Deployment Recommendations for NAT64/464XLAT

NAT64/464XLAT has demonstrated to be a valid choice in several scenarios (IPv6-IPv4 and IPv4-IPv6-IPv4), being the predominant mechanism in the majority of the cellular networks, which account for hundreds of millions of users ([ISOC]). NAT64/464XLAT offer different choices of deployment, depending on each network case, needs and requirements. Despite that, this document is not an explicit recommendation for using this choice versus other IPv4aaS transition mechanisms. Instead, this document is a guide that

facilitates evaluating a possible implementation of NAT64/464XLAT and key decision points about specific design considerations for its deployment.

Depending on the specific requirements of each deployment case, DNS64 may be a required function, while in other cases the adverse effects may be counterproductive. Similarly, in some cases a NAT64 function, together with a DNS64 function, may be a valid solution, when there is a certainty that IPv4-only hosts or applications do not need to be supported (Section 4.6 and Section 4.7). However, in other cases (i.e. IPv4-only devices or applications need to be supported), the limitations of NAT64/DNS64, may suggest the operator to look into 464XLAT as a more complete solution.

In the case of broadband managed networks (where the CE is provided or suggested/supported by the operator), in order to fully support the actual user needs (IPv4-only devices and applications, usage of IPv4 literals and non-IPv6 compliant APIs), the 464XLAT scenario should be considered. In that case, it must support a CLAT function.

If the operator provides DNS services, in order to increase performance by reducing the double translation for all the IPv4 traffic, they may support a DNS64 function and avoid, as much as possible, breaking DNSSEC. In this case, if the DNS service is offering DNSSEC validation, then it must be in such way that it is aware of the DNS64. This is considered the simpler and safer approach, and may be combined as well with other recommendations described in this document:

- o DNS infrastructure MUST be aware of DNS64 (Section 4.1.2).
- o Devices running CLAT SHOULD follow the indications in Section 4.1.3 (Stub Validator). However, this may be out of the control of the operator.
- o CEs SHOULD include a DNS proxy and validator (Section 4.1.4).
- o Section 4.1.5 (ACL of clients) and Section 4.1.6 (Mapping-out IPv4 addresses) MAY be considered by operators, depending on their own infrastructure.

This "increased performance" approach has the disadvantage of potentially breaking DNSSEC for a small percentage of validating end-hosts versus the small impact of a double translation taking place in the CE. If CE performance is not an issue, which is the most frequent case, then a much safer approach is to not use DNS64 at all, and consequently, ensure that all the IPv4 traffic is translated at the CLAT (Section 4.3).

If DNS64 is not used, at least one of the alternatives described in Section 4.1.1, must be followed in order to learn the NAT64 prefix.

The operator needs to consider that if the DNS configuration can be modified (Section 4.4, Section 4.4.2, Section 4.4.3), which most probably is impossible to avoid, there are chances that instead of configuring a DNS64 a foreign non-DNS64 is used. In a scenario with only a NAT64 function IPv4-only remote host will no longer be accessible. Instead, it will continue to work in the case of 464XLAT.

Similar considerations need to be taken regarding the usage of a NAT64 WKP vs NSP (Section 4.5), as they must match with the configuration of the DNS64. In case of using foreign DNS, they may not match. If there is a CLAT and the configured foreign DNS is not a DNS64, the network will keep working only if other means of learning the NAT64 prefix are available.

As described in Section 4.8, for broadband networks, the CEs supporting a CLAT function, SHOULD support DHCPv6-PD ([RFC8415]), or alternative means for configuring a shorter prefix. The CE SHOULD internally reserve one /64 for the stateless NAT46 translation. The operator must ensure that the customers get allocated prefixes shorter than /64 in order to support this optimization. One way or the other, this is not impacting the performance of the operator network.

Operators may follow Section 7 of [RFC6877] (Deployment Considerations), for suggestions in order to take advantage of traffic engineering requirements.

In the case of cellular networks, the considerations regarding DNSSEC may appear as out-of-scope, because UEs Operating Systems, commonly don't support DNSSEC. However, applications running on them may do, or it may be an Operating System "built-in" support in the future. Moreover, if those devices offer tethering, other client devices behind the UE, may be doing the validation, hence the relevance of a proper DNSSEC support by the operator network.

Furthermore, cellular networks supporting 464XLAT ([RFC6877]) and "Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis" ([RFC7050]), allow a progressive IPv6 deployment, with a single APN supporting all types of PDP context (IPv4, IPv6, IPv4v6). This approach allows the network to automatically serve every possible combinations of UEs.

If the operator chooses to provide validation for the DNS64 prefix discovery, it must follow the advice from Section 3.1. of [RFC7050]

(Validation of Discovered Pref64::/n).

One last consideration, is that many networks may have a mix of different complex scenarios at the same time, for example, customers requiring 464XLAT, others not requiring it, customers requiring DNS64, others not, etc. In general, the different issues and the approaches described in this document can be implemented at the same time for different customers or parts of the network. That mix of approaches don't present any problem or incompatibility, as they work well together, being just a matter of appropriate and differentiated provisioning. In fact, the NAT64/464XLAT approach facilitates an operator offering both cellular and broadband services, to have a single IPv4aaS for both networks while differentiating the deployment key decisions to optimize each case. It even makes possible using hybrid CEs that have a main broadband access link and a backup via the cellular network.

In an ideal world we could safely use DNS64, if the approach proposed in [I-D.bp-v6ops-ipv6-ready-dns-dnssec] is followed, avoiding the cases where DNSSEC may be broken. However, this will not solve the issues related to DNS Privacy and Split DNS.

The only 100% safe solution, which also resolves all the issues, will be, in addition to having a CLAT function, not using a DNS64 but instead making sure that the hosts have a built-in address synthesis feature. Operators could manage to provide CEs with the CLAT function, however the built-in address synthesis feature is out of their control. If the synthesis is provided either by the Operating System (via its DNS resolver API) or by the application (via its own DNS resolver), in such way that the prefix used for the NAT64 function is reachable for the host, the problem goes away.

Whenever feasible, using EAM ([RFC7757]) as indicated in Section 4.9, provides a very relevant optimization, avoiding double-translations.

Applications that require incoming connections, typically already provide means for that. However, PCP and EAM, as indicated in Section 4.10, are valid alternatives, even for creating explicit mappings for customers that require them.

6. Deployment of 464XLAT/NAT64 in Enterprise Networks

The recommendations of this document can be used as well in enterprise networks, campus and other similar scenarios (including managed end-user networks).

This include scenarios where the NAT64 function (and DNS64 function, if available) are under the control of that network (or can be

configured manually according to that network specific requirements), and for whatever reasons, there is a need to provide "IPv6-only access" to any part of that network or it is IPv6-only connected to third party-networks.

An example of that is the IETF meetings network itself, where both NAT64 and DNS64 functions are provided, presenting in this case the same issues as per Section 3.1.1. If there is a CLAT function in the IETF network, then there is no need to use DNS64 and it falls under the considerations of Section 3.1.3. Both scenarios have been tested and verified already in the IETF network itself.

Next figures are only meant to represent a few of the possible scenarios, not pretending to be the only feasible ones.

Figure 14 provides an example of an IPv6-only enterprise network connected with dual-stack to Internet and using local NAT64 and DNS64 functions.

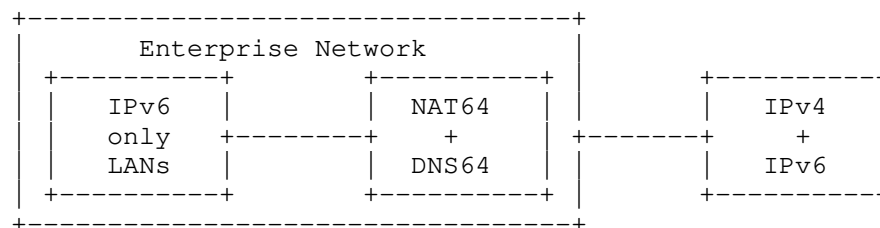


Figure 14: IPv6-only enterprise with NAT64 and DNS64

Figure 15 provides an example of a dual-stack (DS) enterprise network connected with dual-stack (DS) to Internet and using a CLAT function, without a DNS64 function.

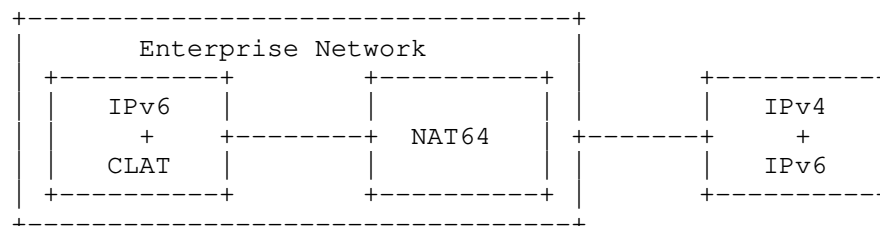


Figure 15: DS enterprise with CLAT, DS Internet, without DNS64

Finally, Figure 16 provides an example of an IPv6-only provider with a NAT64 function, and a dual-stack (DS) enterprise network by means of their own CLAT function, without a DNS64 function.

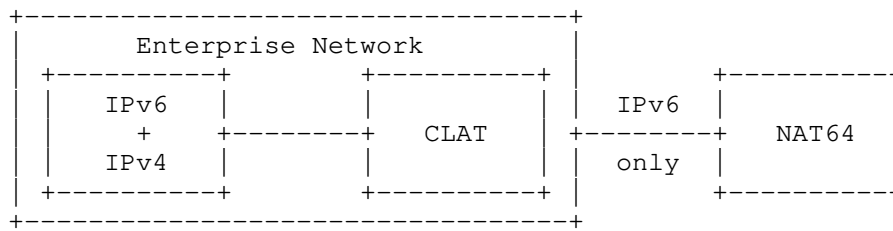


Figure 16: DS enterprise with CLAT, IPv6-only Access, without DNS64

7. Security Considerations

This document does not have new specific security considerations beyond those already reported by each of the documents cited. For example, DNS64 ([RFC6147]) already describes the DNSSEC issues.

Note that, as already described in Section 4.4, there may be undesirable interactions, specially if using VPNs or DNS privacy, which may impact in the correct performance of DNS64/NAT64.

It should be remarked that the use of a DNS64 function has equivalent privacy considerations as in the case of a regular DNS, either located in the service provider or an external one.

8. IANA Considerations

This document does not have any new specific IANA considerations.

Note: This section is assuming that <https://www.rfc-editor.org/errata/eid5152> is resolved, otherwise, this section may include the required text to resolve the issue.

Alternatively, this could be fixed also by [I-D.cheshire-sudn-ipv4only-dot-arpa].

9. Acknowledgements

The author would like to acknowledge the inputs of Gabor Lencse, Andrew Sullivan, Lee Howard, Barbara Stark, Fred Baker, Mohamed Boucadair, Alejandro D'Egidio, Dan Wing, Mikael Abrahamsson and Eric Vyncke.

Conversations with Marcelo Bagnulo, one of the co-authors of NAT64 and DNS64, as well as several emails in mailing lists from Mark Andrews, have been very useful for this work.

Christian Huitema inspired working in this document by suggesting

that DNS64 should never be used, during a discussion regarding the deployment of CLAT in the IETF network.

10. ANNEX A: Example of Broadband Deployment with 464XLAT

This section summarizes how an operator may deploy an IPv6-only network for residential/SOHO customers, supporting IPv6 inbound connections, and IPv4-as-a-Service (IPv4aaS) by using 464XLAT.

Note that an equivalent setup could also be provided for enterprise customers. In case they need to support IPv4 inbound connections, several mechanisms, depending on specific customer needs, allow that, for instance [RFC7757].

Conceptually, most part of the operator network could be IPv6-only (represented in the next pictures as "IPv6-only flow"), or even if this part of the network is actually dual-stack, only IPv6-access is available for some customers (i.e. residential customers). This part of the network connects the IPv6-only subscribers (by means of IPv6-only access links), to the IPv6 upstream providers, as well as to the IPv4-Internet by means of the NAT64 (PLAT in the 464XLAT terminology).

The traffic flow from and back to the CE to services available in the IPv6 Internet (or even dual-stack remote services, when IPv6 is being used), is purely native IPv6 traffic, so there are no special considerations about it.

Looking at the picture from the DNS perspective, there are remote networks with are IPv4-only, and typically will have only IPv4 DNS (DNS/IPv4), or at least will be seen as that from the CE perspective. At the operator side, the DNS, as seen from the CE, is only IPv6 (DNS/IPv6) and has also a DNS64 function.

In the customer LANs side, there is actually one network, which of course could be split in different segments. The most common setup will be those segments being dual-stack, using global IPv6 addresses and [RFC1918] for IPv4, as usual in any regular residential/SOHO IPv4 network. In the figure, it is represented as tree segments, just to show that the three possible setups are valid (IPv6-only, IPv4-only and dual-stack).

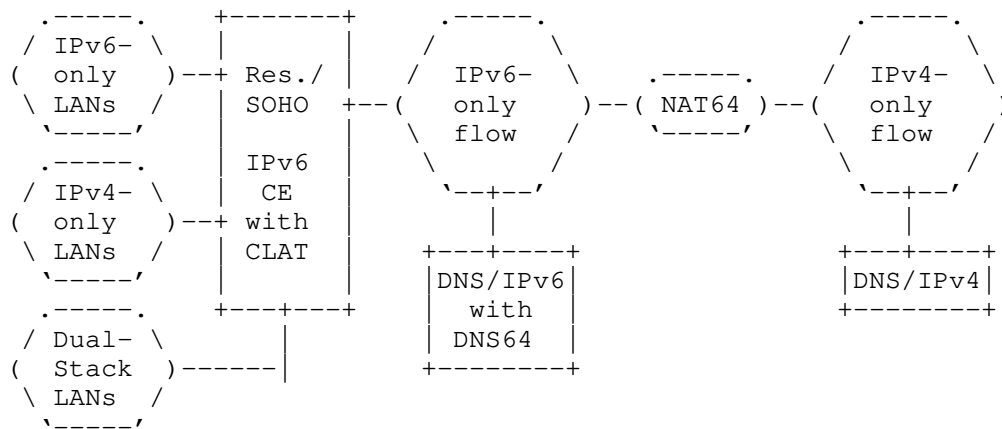


Figure 17: CE setup with built-in CLAT with DNS64

In addition to the regular CE setup, which will be typically access-technology dependent, the steps for the CLAT function configuration can be summarized as:

1. Discovery of the PLAT (NAT64) prefix: It may be done using [RFC7050], or in those networks where PCP is supported, by means of [RFC7225], or other alternatives that may be available in the future, such as Router Advertising ([I-D.ietf-6man-ra-pref64]) or DHCPv6 options ([I-D.li-intarea-nat64-prefix-dhcp-option]).
2. If the CLAT function allows stateless NAT46 translation, a /64 from the pool typically provided to the CE by means of DHCPv6-PD [RFC8415], need to be set aside for that translation. Otherwise, the CLAT is forced to perform an intermediate stateful NAT44 before the a stateless NAT46, as described in Section 4.8.

A more detailed configuration approach is described in [RFC8585].

The operator network needs to ensure that the correct responses are provided for the discovery of the PLAT prefix. It is highly recommended to follow [RIPE-690], in order to ensure that multiple /64s are available, including the one needed for the NAT46 stateless translation.

The operator needs to understand other issues, described across this document, in order to take the relevant decisions. For example, if several NAT64 functions are needed in the context of scalability/high-availability, an NSP should be considered (Section 4.5).

More complex scenarios are possible, for example, if a network offers

multiple NAT64 prefixes, destination-based NAT64 prefixes, etc.

If the operator decides not to provide a DNS64 function, then this setup turns into the one in the following Figure. This will be also the setup that "will be seen" from the perspective of the CE, if a foreign DNS is used and consequently is not the operator-provided DNS64 function.

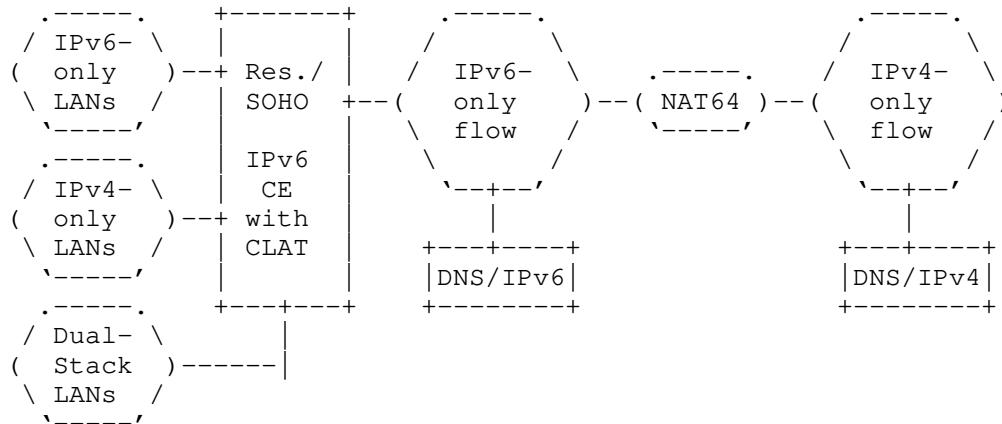


Figure 18: CE setup with built-in CLAT without DNS64

In this case, the discovery of the PLAT prefix needs to be arranged as indicated in Section 4.1.1.

In this case, the CE doesn't have a built-in CLAT function, or the customer can choose to setup the IPv6 operator-managed CE in bridge mode (and optionally use an external router), or for example, there is an access technology that requires some kind of media converter (ONT for FTTH, Cable-Modem for DOCSIS, etc.), the complete setup will look as in Figure 19. Obviously, there will be some intermediate configuration steps for the bridge, depending on the specific access technology/protocols, which should not modify the steps already described in the previous cases for the CLAT function configuration.

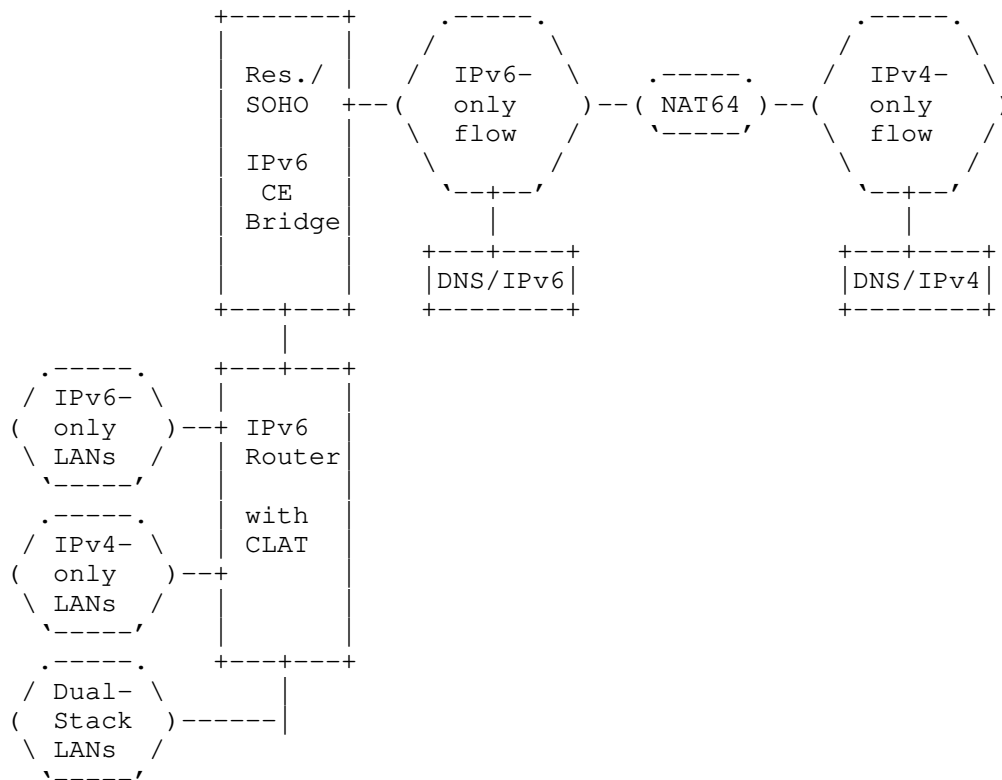


Figure 19: CE setup with bridged CLAT without DNS64

It should be avoided that several routers (i.e., the operator provided CE and a downstream user provided router) enable simultaneously routing and/or CLAT, in order to avoid multiple NAT44 and NAT46 levels, as well as ensuring the correct operation of multiple IPv6 subnets. In those cases, it is suggested the use of HNCP ([RFC8375]).

Note that the procedure described here for the CE setup, can be simplified if the CE follows [RFC8585].

11. ANNEX B: CLAT Implementation

In addition to the regular set of features for a CE, a CLAT CE implementation requires support of:

- o [RFC7915] for the NAT46 function.
- o [RFC7050] for the PLAT prefix discovery.

- o [RFC7225] for the PLAT prefix discovery if PCP is supported.
- o [I-D.ietf-6man-ra-pref64] for the PLAT prefix discovery by means of Router Advertising.
- o [I-D.li-intarea-nat64-prefix-dhcp-option] for the PLAT prefix discovery by means of DHCP.
- o If stateless NAT46 is supported, a mechanism to ensure that multiple /64 are available, such as DHCPv6-PD [RFC8415].

There are several OpenSource implementations of CLAT, such as:

- o Android: https://github.com/ddrown/android_external_android-clat.
- o Jool: <https://www.jool.mx>.
- o Linux: <https://github.com/toreanderson/clatd>.
- o OpenWRT: <https://github.com/openwrt-routing/packages/blob/master/nat46/files/464xlat.sh>.
- o VPP: <https://git.fd.io/vpp/tree/src/plugins/nat>.

12. ANNEX C: Benchmarking

[RFC8219] has defined a benchmarking methodology for IPv6 transition technologies. NAT64 and 464XLAT are addressed among the single and double translation technologies, respectively. DNS64 is addressed in Section 9, and the methodology is more elaborated in [DNS64-BM-Meth].

Several documents provide references to benchmarking results, for example in the case of DNS64, [DNS64-Benchm].

13. ANNEX D: Changes from -00 to -01/-02

Section to be removed after WGLC. Significant updates are:

1. Text changes across all the document.

14. ANNEX E: Changes from -02 to -03

Section to be removed after WGLC. Significant updates are:

1. Added references to new cited documents.
2. Reference to RFC8273 and on-demand IPv4-in-IPv6 VPN for IPv6-only LANs w/o DNS64.

3. Overall review and editorial changes.
15. ANNEX F: Changes from -03 to -04

Section to be removed after WGLC. Significant updates are:

 1. Added text related to EAM considerations.
16. ANNEX G: Changes from -04 to -05

Section to be removed after WGLC. Significant updates are:

 1. Added cross references to EAM section.
 2. Reworded "foreing DNS section".
 3. Overall editorial review of text, pictures and nits correction.
17. ANNEX H: Changes from -05 to -06

Section to be removed after WGLC. Significant updates are:

 1. Corrected EAMT to EAM.
 2. Typos and nits.
 3. New considerations regarding incoming connections.
18. ANNEX H: Changes from -06 to -07

Section to be removed after WGLC. Significant updates are:

 1. Inputs/clarifications from IESG review.
19. References
- 19.1. Normative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., 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>>.

[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>>.

- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <<https://www.rfc-editor.org/info/rfc5389>>.
- [RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP 152, RFC 5625, DOI 10.17487/RFC5625, August 2009, <<https://www.rfc-editor.org/info/rfc5625>>.
- [RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, DOI 10.17487/RFC5766, April 2010, <<https://www.rfc-editor.org/info/rfc5766>>.
- [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>>.
- [RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, DOI 10.17487/RFC6144, April 2011, <<https://www.rfc-editor.org/info/rfc6144>>.
- [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>>.
- [RFC6535] Huang, B., Deng, H., and T. Savolainen, "Dual-Stack Hosts Using "Bump-in-the-Host" (BIH)", RFC 6535, DOI 10.17487/RFC6535, February 2012, <<https://www.rfc-editor.org/info/rfc6535>>.
- [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>>.

- [RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <<https://www.rfc-editor.org/info/rfc6887>>.
- [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>>.
- [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>>.
- [RFC7757] Anderson, T. and A. Leiva Popper, "Explicit Address Mappings for Stateless IP/ICMP Translation", RFC 7757, DOI 10.17487/RFC7757, February 2016, <<https://www.rfc-editor.org/info/rfc7757>>.
- [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>>.
- [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>>.
- [RFC8273] Brzozowski, J. and G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273, DOI 10.17487/RFC8273, December 2017, <<https://www.rfc-editor.org/info/rfc8273>>.
- [RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <<https://www.rfc-editor.org/info/rfc8305>>.
- [RFC8375] Pfister, P. and T. Lemon, "Special-Use Domain 'home.arpa.'", RFC 8375, DOI 10.17487/RFC8375, May 2018, <<https://www.rfc-editor.org/info/rfc8375>>.
- [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>>.

- [RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <<https://www.rfc-editor.org/info/rfc8445>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.

19.2. Informative References

- [About-DNS64]
Linkova, J., "Let's talk about IPv6 DNS64 & DNSSEC", 2016, <<https://blog.apnic.net/2016/06/09/lets-talk-ipv6-dns64-dnssec/>>.
- [ARCEP] ARCEP, "Service client des operateurs : les mesures de qualite de service", 2018, <<https://www.arcep.fr/cartes-et-donnees/nos-publications-chiffrees/service-client-des-operateurs-mesures-de-la-qualite-de-service/service-client-des-operateurs-les-mesures-de-qualite-de-service.html>>.
- [DNS64-Benchm]
Lencse, G. and Y. Kadobayashi, "Benchmarking DNS64 Implementations: Theory and Practice", Computer Communications , vol. 127, no. 1, pp. 61-74, DOI 10.1016/j.comcom.2018.05.005, September 2018.
- [DNS64-BM-Meth]
Lencse, G., Georgescu, M., and Y. Kadobayashi, "Benchmarking Methodology for DNS64 Servers", Computer Communications , vol. 109, no. 1, pp. 162-175, DOI 10.1016/j.comcom.2017.06.004, September 2017.
- [FCC] FCC, "Measuring Broadband America Mobile 2013-2018 Coarsened Data", 2018, <<https://www.fcc.gov/reports-research/reports/measuring-broadband-america/measuring-broadband-america-mobile-2013-2018>>.
- [I-D.bp-v6ops-ipv6-ready-dns-dnssec]
Byrne, C. and J. Palet, "IPv6-Ready DNS/DNSSEC Infrastructure", draft-bp-v6ops-ipv6-ready-dns-dnssec-00 (work in progress), October 2018.

- [I-D.cheshire-sudn-ipv4only-dot-arpa]
Cheshire, S. and D. Schinazi, "Special Use Domain Name 'ipv4only.arpa'", draft-cheshire-sudn-ipv4only-dot-arpa-14 (work in progress), November 2018.
- [I-D.huitema-quic-dnsquic]
Huitema, C., Shore, M., Mankin, A., Dickinson, S., and J. Iyengar, "Specification of DNS over Dedicated QUIC Connections", draft-huitema-quic-dnsquic-06 (work in progress), March 2019.
- [I-D.ietf-6man-ra-pref64]
Colitti, L. and J. Linkova, "Discovering PREF64 in Router Advertisements", draft-ietf-6man-ra-pref64-01 (work in progress), June 2019.
- [I-D.ietf-tram-stunbis]
Petit-Huguenin, M., Salgueiro, G., Rosenberg, J., Wing, D., Mahy, R., and P. Matthews, "Session Traversal Utilities for NAT (STUN)", draft-ietf-tram-stunbis-21 (work in progress), March 2019.
- [I-D.ietf-tram-turnbis]
K, R., Johnston, A., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", draft-ietf-tram-turnbis-27 (work in progress), June 2019.
- [I-D.li-intarea-nat64-prefix-dhcp-option]
Li, L., Cui, Y., Liu, C., Wu, J., Baker, F., and J. Palet, "DHCPv6 Options for Discovery NAT64 Prefixes", draft-li-intarea-nat64-prefix-dhcp-option-02 (work in progress), April 2019.
- [I-D.lmhp-v6ops-transition-comparison]
Lencse, G., Palet, J., Howard, L., Patterson, R., and I. Farrer, "Pros and Cons of IPv6 Transition Technologies for IPv4aaS", draft-lmhp-v6ops-transition-comparison-03 (work in progress), July 2019.
- [I-D.palet-v6ops-464xlat-opt-cdn-caches]
Palet, J. and A. D'Egidio, "464XLAT Optimization", draft-palet-v6ops-464xlat-opt-cdn-caches-02 (work in progress), June 2019.

- [I-D.vixie-dns-rpz] Vixie, P. and V. Schryver, "DNS Response Policy Zones (RPZ)", draft-vixie-dns-rpz-04 (work in progress), December 2016.
- [ISOC] ISOC, "State of IPv6 Deployment 2018", 2018, <<https://www.internetsociety.org/resources/2018/state-of-ipv6-deployment-2018/>>.
- [RFC6889] Penno, R., Saxena, T., Boucadair, M., and S. Sivakumar, "Analysis of Stateful 64 Translation", RFC 6889, DOI 10.17487/RFC6889, April 2013, <<https://www.rfc-editor.org/info/rfc6889>>.
- [RFC6950] Peterson, J., Kolkman, O., Tschofenig, H., and B. Aboba, "Architectural Considerations on Application Features in the DNS", RFC 6950, DOI 10.17487/RFC6950, October 2013, <<https://www.rfc-editor.org/info/rfc6950>>.
- [RFC7051] Korhonen, J., Ed. and T. Savolainen, Ed., "Analysis of Solution Proposals for Hosts to Learn NAT64 Prefix", RFC 7051, DOI 10.17487/RFC7051, November 2013, <<https://www.rfc-editor.org/info/rfc7051>>.
- [RFC7269] Chen, G., Cao, Z., Xie, C., and D. Binet, "NAT64 Deployment Options and Experience", RFC 7269, DOI 10.17487/RFC7269, June 2014, <<https://www.rfc-editor.org/info/rfc7269>>.
- [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>>.
- [RFC7756] Anderson, T. and S. Steffann, "Stateless IP/ICMP Translation for IPv6 Internet Data Center Environments (SIIT-DC): Dual Translation Mode", RFC 7756, DOI 10.17487/RFC7756, February 2016, <<https://www.rfc-editor.org/info/rfc7756>>.
- [RFC7849] Binet, D., Boucadair, M., Vizdal, A., Chen, G., Heatley, N., Chandler, R., Michaud, D., Lopez, D., and W. Haeflner, "An IPv6 Profile for 3GPP Mobile Devices", RFC 7849, DOI 10.17487/RFC7849, May 2016, <<https://www.rfc-editor.org/info/rfc7849>>.

- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <<https://www.rfc-editor.org/info/rfc8094>>.
- [RFC8219] Georgescu, M., Pislaru, L., and G. Lencse, "Benchmarking Methodology for IPv6 Transition Technologies", RFC 8219, DOI 10.17487/RFC8219, August 2017, <<https://www.rfc-editor.org/info/rfc8219>>.
- [RFC8585] Palet Martinez, J., Liu, 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>>.
- [RIPE-690] RIPE, "Best Current Operational Practice for Operators: IPv6 prefix assignment for end-users - persistent vs non-persistent, and what size to choose", October 2017, <<https://www.ripe.net/publications/docs/ripe-690>>.
- [Threat-DNS64] Lencse, G. and Y. Kadobayashi, "Methodology for the identification of potential security issues of different IPv6 transition technologies: Threat analysis of DNS64 and stateful NAT64", Computers & Security, vol. 77, no. 1, pp. 397-411, DOI 10.1016/j.cose.2018.04.012, August 2018.

Author's Address

Jordi Palet Martinez
The IPv6 Company
Molino de la Navata, 75
La Navata - Galapagar, Madrid 28420
Spain

Email: jordi.palet@theipv6company.com
URI: <http://www.theipv6company.com/>

IPv6 Operations (v6ops)
Internet-Draft
Intended status: Informational
Expires: August 1, 2019

J. Palet Martinez
The IPv6 Company
H. M.-H. Liu
D-Link Systems, Inc.
M. Kawashima
NEC Platforms, Ltd.
January 28, 2019

Requirements for IPv6 Customer Edge Routers to Support IPv4 Connectivity
as-a-Service
draft-ietf-v6ops-transition-ipv4aaS-15

Abstract

This document specifies the IPv4 service continuity requirements for an IPv6 Customer Edge (CE) router, either provided by the service provider or by vendors who sell through the retail market.

Specifically, this document extends the "Basic Requirements for IPv6 Customer Edge Routers" (RFC7084) in order to allow the provisioning of IPv6 transition services for the support of "IPv4 as-a-Service" (IPv4aaS) by means of new transition mechanisms. The document only covers transition technologies for delivering IPv4 in IPv6-only access networks, commonly called "IPv4 as-a-Service" (IPv4aaS). This is necessary because there aren't sufficient IPv4 addresses available for every possible customer/device. However, devices or applications in the customer LANs (Local Area Networks) may be IPv4-only or IPv6-only and still need to communicate with IPv4-only services at the Internet.

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 August 1, 2019.

Copyright Notice

Copyright (c) 2019 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	5
2. Terminology	5
3. Requirements	6
3.1. LAN-Side Configuration	6
3.2. Transition Technologies Support for IPv4 Service Continuity (IPv4 as-a-Service - IPv4aaS)	6
3.2.1. 464XLAT	7
3.2.2. Dual-Stack Lite (DS-Lite)	9
3.2.3. Lightweight 4over6 (lw4o6)	10
3.2.4. MAP-E	10
3.2.5. MAP-T	11
4. IPv4 Multicast Support	11
5. UPnP Support	12
6. Comparison to RFC7084	12
7. Code Considerations	12
8. Security Considerations	13
9. IANA Considerations	13
10. Acknowledgements	13
11. Annex A: Usage Scenarios	14
12. Annex B: End-User Network Architecture	15
13. ANNEX C: Changes from -00	18
14. ANNEX D: Changes from -01	18
15. ANNEX E: Changes from -02	18
16. ANNEX F: Changes from -03	19
17. ANNEX G: Changes from -04	19
18. ANNEX H: Changes from -05	19
19. ANNEX I: Changes from -06	19
20. ANNEX J: Changes from -07	19
21. ANNEX K: Changes from -08, -09 and -10	20
22. ANNEX L: Changes from -11, -12, -13 and -14	20

23. References	20
23.1. Normative References	20
23.2. Informative References	23
Authors' Addresses	23

1. Introduction

This document defines IPv4 service continuity features over an IPv6-only network, for a residential or small-office router, referred to as an "IPv6 Transition CE Router", in order to establish an industry baseline for transition features to be implemented on such a router.

These routers rely upon "Basic Requirements for IPv6 Customer Edge Routers" [RFC7084]. The scope of this document is to ensure the IPv4 "service continuity" support, for devices in the LAN side. This ensures that remote IPv4-only services continue to be accessible, from an IPv6-only Internet Service Provider (ISP) access network from both, IPv4-only and IPv6-only applications and devices in the LAN side. These ISP access networks are typically referred to as Wide Area Networks (WANs), even if in some cases they may be metropolitan or regional. Figure 1 presents a simplified view of this architecture.

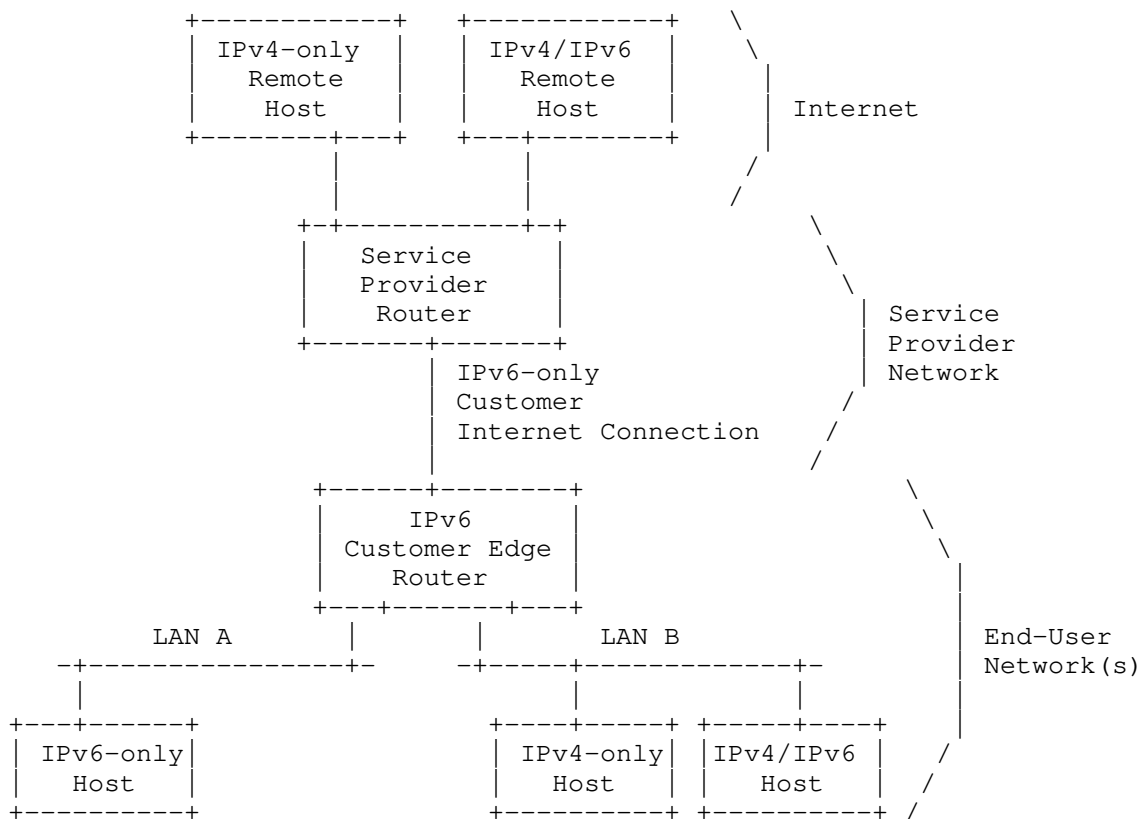


Figure 1: Simplified Typical IPv6-only Access Network

This document covers a set of IP transition techniques required when ISPs have, or want to have, an IPv6-only access network. This is a common situation when sufficient IPv4 addresses are no longer available for every possible customer and device, causing IPv4 addresses to become prohibitively expensive. This, in turn, may result in service providers provisioning IPv6-only WAN access. At the same time, they need to ensure that both IPv4-only and IPv6-only devices and applications in the customer networks can still reach IPv4-only devices and applications in the Internet.

This document specifies the IPv4 service continuity mechanisms to be supported by an IPv6 Transition CE Router, and relevant provisioning or configuration information differences from [RFC7084].

This document is not a recommendation for service providers to use any specific transition mechanism.

Automatic provisioning of more complex topology than a single router with multiple LAN interfaces may be handled by means of HNCP [RFC7788] (Home Networking Control Protocol), which is out of the scope of this document.

Since it is impossible to know prior to sale which transition mechanism a device will need over its lifetime, an IPv6 Transition CE Router intended for the retail market MUST support all the IPv4aaS transition mechanisms listed in this document. Service providers who specify feature sets for the IPv6 Transition CE Router, may define a different set of features than those included in this document, for example supporting only some of the transition mechanisms enumerated in this document.

A complete description of "Usage Scenarios" and "End-User Network Architecture" is provided in Annexes A and B, respectively, which together with [RFC7084], will facilitate the reader to have a clearer understanding of this document.

1.1. 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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Terminology

This document uses the same terms as in [RFC7084], with minor clarifications.

"IPv4aaS" stands for "IPv4 as-a-Service", meaning transition technologies for delivering IPv4 in IPv6-only connectivity.

The term "IPv6 transition Customer Edge Router with IPv4aaS" (shortened as "IPv6 Transition CE Router") is defined as an "IPv6 Customer Edge Router" that provides features for the delivery of IPv4 services over an IPv6-only WAN network, including IPv6-IPv4 communications.

The "WAN Interface" term used across this document, defines an IPv6 Transition CE Router attachment to an IPv6-only link used to provide connectivity to a service provider network, including link Internet-layer (or higher layers) "tunnels", such as IPv4-in-IPv6 tunnels.

3. Requirements

The IPv6 Transition CE Router MUST comply with [RFC7084] (Basic Requirements for IPv6 Customer Edge Routers) and this document adds new requirements, as described in the following sub-sections.

3.1. LAN-Side Configuration

A new LAN requirement is added, which in fact is common in regular IPv6 Transition CE Routers, and it is required by most of the transition mechanisms:

L-1: The IPv6 Transition CE Router MUST implement a DNS proxy as described in [RFC5625] (DNS Proxy Implementation Guidelines).

3.2. Transition Technologies Support for IPv4 Service Continuity (IPv4 as-a-Service - IPv4aaS)

The main target of this document is the support of IPv6-only WAN access. To enable legacy IPv4 functionality, this document also includes the support of IPv4-only devices and applications in the customers LANs, as well as IPv4-only services on the Internet. Thus, both IPv4-only and the IPv6-only devices in the customer-side LANs of the IPv6 Transition CE Router are able to reach the IPv4-only services.

Note that this document is only configuring the IPv4aaS in the IPv6 Transition CE Router itself, and not forwarding such information to devices attached to the LANs, so the WAN configuration, availability of native IPv4 or IPv4aaS, is transparent for them.

This document takes no position on simultaneous operation of one or several transition mechanisms and/or native IPv4.

In order to seamlessly provide IPv4 service continuity in the customer LANs, and allow automated IPv6 transition mechanism provisioning, the following general transition requirements are defined.

General transition requirements:

TRANS-1: The IPv6 Transition CE Router MUST support the DHCPv6 S46 priority options described in [RFC8026] (Unified IPv4-in-IPv6 Software Customer Premises Equipment (CPE): A DHCPv6-Based Prioritization Mechanism).

TRANS-2: The IPv6 Transition CE Router MUST have a GUI and either a CLI or API (or both) to manually enable/disable each of the

supported transition mechanisms.

TRANS-3: If an IPv6 Transition CE Router supports more than one LAN subnet, the IPv6 Transition CE Router MUST allow appropriate subnetting and configuration of the address space among the several interfaces. In some transition mechanisms, this may require differentiating mappings/translations on a per-interface basis.

In order to allow the service provider to disable all the transition mechanisms and/or choose the most convenient one, the IPv6 Transition CE Router MUST follow the following configuration steps:

- CONFIG-1: Request the relevant configuration options for each supported transition mechanisms, which MUST remain disabled at this step.
- CONFIG-2: Following Section 1.4 of [RFC8026], MUST check for a valid match in OPTION_S46_PRIORITY, which allows enabling/disabling a transition mechanism.
- CONFIG-3: Keep disabled all the transition mechanisms if no match is found between the priority list and the candidate list, unless a NAT64 prefix has been configured, in which case, 464XLAT MUST be enabled.

Because 464XLAT has not DHCPv6 configuration options, it can't be included, at the time being, in the OPTION_S46_PRIORITY. In the future, an update of [RFC8026] or a NAT64 DHCPv6 configuration option, may enable it. Meanwhile, if an operator provides 464XLAT, it needs to ensure that OPTION_S46_PRIORITY is not sent for any other transition mechanism to the relevant customers.

The following sections describe the requirements for supporting each one of the transition mechanisms. An IPv6 Transition CE Router intended for the retail market MUST support all of them.

3.2.1. 464XLAT

464XLAT [RFC6877] is a technique to provide IPv4 service over an IPv6-only access network without encapsulation. This architecture assumes a NAT64 [RFC6146] (Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers) function deployed at the service provider or a third-party network.

The IPv6 Transition CE Router MUST support CLAT functionality [RFC6877] if intended for the retail market. If 464XLAT is supported, it MUST be implemented according to [RFC6877]. The

following IPv6 Transition CE Router requirements also apply:

464XLAT requirements:

- 464XLAT-1: Unless a dedicated /64 prefix has been acquired, either using DHCPv6-PD [RFC8415] (IPv6 Prefix Options for DHCPv6) or by alternative means, the IPv6 Transition CE Router MUST perform IPv4 Network Address Translation (NAT) on IPv4 traffic translated using the CLAT.
- 464XLAT-2: The IPv6 Transition CE Router SHOULD support IGD-PCP IWF [RFC6970] (UPnP Internet Gateway Device - Port Control Protocol Interworking Function).
- 464XLAT-3: If PCP [RFC6887] is implemented, the IPv6 Transition CE Router MUST also implement [RFC7291] (DHCP Options for the PCP). Following [RFC6887], if no PCP server is configured, the IPv6 Transition CE Router MAY verify if the default gateway, or the NAT64 is the PCP server. The IPv6 Transition CE Router MUST use plain IPv6 mode (i.e., no IPv4-in-IPv6 encapsulation is used) to send PCP requests to the server.
- 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 PLAT-side translation IPv4 and IPv6 prefix(es)/suffix(es).
- 464XLAT-5: If PCP is implemented, the IPv6 Transition CE Router MUST follow [RFC7225] (Discovering NAT64 IPv6 Prefixes Using the PCP), in order to learn the PLAT-side translation IPv4 and IPv6 prefix(es)/suffix(es) used by an upstream PCP-controlled NAT64 device.
- 464XLAT-6: The priority for the NAT64 prefix, in case the network provides several choices, MUST be: 1) [RFC7225], 2) [RFC7050].

The NAT64 prefix could be discovered by means of [RFC7050] only in the case 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 [RFC7225]. This can also be used if the service provider uses DNS64 [RFC6147].

3.2.2. Dual-Stack Lite (DS-Lite)

Dual-Stack Lite [RFC6333] enables continued support for IPv4 services. Dual-Stack Lite enables a broadband service provider to share IPv4 addresses among customers by combining two well-known technologies: IP in IP (IPv4-in-IPv6) and Network Address Translation (NAT). It is expected that DS-Lite traffic is forwarded over the IPv6 Transition CE Router's native IPv6 WAN interface, and not encapsulated in another tunnel.

The IPv6 Transition CE Router MUST implement DS-Lite B4 functionality [RFC6333] if intended for the retail market. If DS-Lite is supported, it MUST be implemented according to [RFC6333]. The following IPv6 Transition CE Router requirements also apply:

DS-Lite requirements:

- DSLITE-1: The IPv6 Transition CE Router MUST support configuration of DS-Lite via the DS-Lite DHCPv6 option [RFC6334] (DHCPv6 Option for Dual-Stack Lite). The IPv6 Transition CE Router MAY use other mechanisms to configure DS-Lite parameters. Such mechanisms are outside the scope of this document.
- DSLITE-2: The IPv6 Transition CE Router SHOULD support IGD-PCP IWF [RFC6970] (UPnP Internet Gateway Device - Port Control Protocol Interworking Function).
- DSLITE-3: If PCP [RFC6887] is implemented, the IPv6 Transition CE Router SHOULD implement [RFC7291] (DHCP Options for the PCP). If PCP [RFC6887] is implemented and a PCP server is not configured, the IPv6 Transition CE Router MUST assume, by default, that the AFTR is the PCP server. The IPv6 Transition CE Router MUST use plain IPv6 mode (i.e., no IPv4-in-IPv6 encapsulation is used) to send PCP requests to the server. The term "default" above is to be interpreted as pertaining to a configuration as applied by a vendor, prior to the administrator changing it for its initial activation.
- DSLITE-4: The IPv6 Transition CE Router MUST NOT perform IPv4 Network Address Translation (NAT) on IPv4 traffic encapsulated using DS-Lite [RFC6333].

3.2.3. Lightweight 4over6 (lw4o6)

lw4o6 [RFC7596] specifies an extension to DS-Lite which moves the NAPT function from the DS-Lite tunnel concentrator to the tunnel client located in the IPv6 Transition CE Router, removing the requirement for a CGN (Carrier Grade NAT, AFTR - Address Family Transition Router) function in the tunnel concentrator and reducing the amount of centralized state.

The IPv6 Transition CE Router MUST implement lwB4 functionality [RFC7596] if intended for the retail market. If DS-Lite is implemented, lw4o6 SHOULD be implemented as well. If lw4o6 is supported, it MUST be implemented according to [RFC7596]. The following IPv6 Transition CE Router requirements also apply:

lw4o6 requirements:

- LW4O6-1: The IPv6 Transition CE Router MUST support configuration of lw4o6 via the lw4o6 DHCPv6 options [RFC7598] (DHCPv6 Options for Configuration of Software Address and Port-Mapped Clients). The IPv6 Transition CE Router MAY use other mechanisms to configure lw4o6 parameters. Such mechanisms are outside the scope of this document.
- LW4O6-2: The IPv6 Transition CE Router MUST support the DHCPv4-over-DHCPv6 (DHCP 4o6) transport described in [RFC7341] (DHCPv4-over-DHCPv6 Transport).
- LW4O6-3: The IPv6 Transition CE Router MAY support Dynamic Allocation of Shared IPv4 Addresses as described in [RFC7618] (Dynamic Allocation of Shared IPv4 Addresses).

3.2.4. MAP-E

MAP-E [RFC7597] is a mechanism for transporting IPv4 packets across an IPv6 network using IP encapsulation, including an algorithmic mechanism for mapping between IPv6 and IPv4 addresses.

The IPv6 Transition CE Router MUST support MAP-E CE functionality [RFC7597] if intended for the retail market. If MAP-E is supported, it MUST be implemented according to [RFC7597]. The following IPv6 Transition CE Router requirements also apply:

MAP-E requirements:

- MAPE-1: The IPv6 Transition CE Router MUST support configuration of MAP-E via the MAP-E DHCPv6 options [RFC7598] (DHCPv6 Options for Configuration of Software Address and Port-Mapped

Clients). The IPv6 Transition CE Router MAY use other mechanisms to configure MAP-E parameters. Such mechanisms are outside the scope of this document.

MAPE-2: The IPv6 Transition CE Router MAY support Dynamic Allocation of Shared IPv4 Addresses as described in [RFC7618] (Dynamic Allocation of Shared IPv4 Addresses).

3.2.5. MAP-T

MAP-T [RFC7599] is a mechanism similar to MAP-E, differing from it in that MAP-T uses IPv4-IPv6 translation, instead of encapsulation, as the form of IPv6 domain transport.

The IPv6 Transition CE Router MUST support MAP-T CE functionality [RFC7599] if intended for the retail market. If MAP-T is supported, it MUST be implemented according to [RFC7599]. The following IPv6 Transition CE Router requirements also apply:

MAP-T requirements:

MAPT-1: The IPv6 Transition CE Router MUST support configuration of MAP-T via the MAP-T DHCPv6 options [RFC7598] (DHCPv6 Options for Configuration of Software Address and Port-Mapped Clients). The IPv6 Transition CE Router MAY use other mechanisms to configure MAP-T parameters. Such mechanisms are outside the scope of this document.

MAPT-2: The IPv6 Transition CE Router MAY support Dynamic Allocation of Shared IPv4 Addresses as described in [RFC7618] (Dynamic Allocation of Shared IPv4 Addresses).

4. IPv4 Multicast Support

Existing IPv4 deployments support IPv4 multicast for services such as IPTV. In the transition phase, it is expected that multicast services will still be provided using IPv4 to the customer LANs.

If the IPv6 Transition CE Router supports delivery of IPv4 multicast services, then it MUST support [RFC8114] (Delivery of IPv4 Multicast Services to IPv4 Clients over an IPv6 Multicast Network) and [RFC8115] (DHCPv6 Option for IPv4-Embedded Multicast and Unicast IPv6 Prefixes).

5. UPnP Support

If the UPnP WANIPConnection:2 service [UPnP-WANIPC] is enabled on a CE router, but cannot be associated with an IPv4 interface established by an IPv4aaS mechanism or cannot determine which ports are available, an AddPortMapping() or AddAnyPortMapping() action MUST be rejected with error code 729 "ConflictWithOtherMechanisms". Port availability could be determined through PCP or access to a configured port set (if the IPv4aaS mechanism limits the available ports).

An AddPortMapping() request for a port that is not available MUST result in "ConflictInMappingEntry".

An AddAnyPortMapping() request for a port that is not available SHOULD result in a successful mapping with an alternative "NewReservedPort" value from within the configured port set range, or as assigned by PCP as per [RFC6970], Section 5.6.1.

Note that IGD:1 and its WANIPConnection:1 service have been deprecated by OCF (Open Connectivity Foundation).

6. Comparison to RFC7084

This document doesn't include support for 6rd [RFC5969], because it is an IPv6-in-IPv4 tunneling.

Regarding DS-LITE [RFC6333], this document includes slightly different requirements, related to the support of PCP [RFC6887], IGD-PCP IWF [RFC6970] and the prioritization of the transition mechanisms, including dual-stack.

7. Code Considerations

At the time of this writing, one of the apparent main issues for vendors to include new functionalities, such as support for new transition mechanisms, is the lack of space in the flash (or equivalent) memory. However, it has been confirmed from existing open source implementations (OpenWRT/LEDE, Linux, VPP, others), that adding the support for the new transition mechanisms, requires around 10-12 Kbytes, because most of the code base is shared among several transition mechanisms, which are already supported by [RFC7084]. A single data plane is common to all them, which typically means, in popular CEs already in the market [OpenWRT], the new required code is only about 0.15% of the total existing code size.

In general, the new requirements don't have extra cost in terms of RAM memory, nor other hardware requirements such as more powerful

CPUs, if compared to the cost of NAT44 code, so existing hardware should be able to support all them with minimal impact.

The other issue seems to be the cost of developing the code for those new functionalities. However, at the time of writing this document, it has been confirmed that there are several open source versions of the required code for supporting all the new transition mechanisms, and several vendors already have implementations and provide it to ISPs, so the development cost is negligible, and only integration and testing cost may become an issue.

Finally, in some cases, operators supporting several transition mechanisms may need to consider training costs for staff in all the techniques for their operation and management, even if this is not directly caused by supporting this document, but because the business decisions behind that.

8. Security Considerations

The IPv6 Transition CE Router must comply with the Security Considerations as stated in [RFC7084], as well as those stated by each transition mechanism implemented by the IPv6 Transition CE Router.

As described in [RFC8026] and [RFC8415] Security Consideration sections, there are generic DHCP security issues, which in the case of this document means that malicious nodes may alter the priority of the transition mechanisms.

Access network architecture for securing DHCP within the access network is out of scope of this document. Securing DHCP in the LAN is also not in scope. DHCP packets MUST NOT be forwarded between LAN and WAN interfaces of an IPv6 Transition CE router.

9. IANA Considerations

This document does not have any new specific IANA considerations.

10. Acknowledgements

Thanks to Mikael Abrahamsson, Fred Baker, Mohamed Boucadair, Brian Carpenter, Ian Farrer, Lee Howard, Richard Patterson, Barbara Stark, Ole Troan, James Woodyatt, Lorenzo Colitti and Alejandro D'Egidio, for their review and comments in this and/or previous versions of this document, as well as to the Last Call reviewers by the Ops-dir (Dan Romascanu), Sec-dir (Christian Huitema), Rtg-dir (Daniele Ceccarelli), Tsv-art (Martin Stiemerling), Gen-art (Matthew Miller) and IESG (Alissa Cooper, Benjamin Kaduk, Suresh Krishnan, Ben

Campbell, Spencer Dawkins, Mirja Kuhlewind, and Adam Roach).

11. Annex A: Usage Scenarios

The situation previously described, where there is ongoing IPv6 deployment and lack of IPv4 addresses, is not happening at the same pace in every country, and even within every country, for every ISP. For different technical, financial, commercial/marketing and socio-economic reasons, each network is transitioning at their own pace; the global transition timings cannot be reliably estimated.

Different studies (for example [IPv6Survey]) also show that the IPv6 deployment is a changing situation. In a single country, not all operators will necessarily provide IPv6 support. Consumers may also switch ISPs, and use the same IPv6 Transition CE Router with either an ISP that provides IPv4-only or an ISP that provides IPv6 with IPv4aaS.

So, to cover all those evolving situations, an IPv6 Transition CE Router is required, at least from the perspective of the transition support.

Moreover, because some services will remain IPv4-only for an undetermined time, and some service providers will remain IPv4-only for an undetermined period of time, IPv4 will be needed for an undetermined period of time. There will be a need for CEs with support "IPv4 as-a-Service" for an undetermined period of time.

This document, based on those premises, ensures that the IPv6 Transition CE Router allows the continued transition from networks that today may provide access with dual-stack or IPv6-in-IPv4, as described in [RFC7084], and as an "extension" to it, evolve to an IPv6-only access with IPv4-as-a-Service.

Considering that situation and different possible usage cases, the IPv6 Transition CE Router described in this document is expected to be used typically, in residential/household, Small Office/Home Office (SOHO) and Small/Medium Enterprise (SME). Common usage is any kind of Internet access (web, email, streaming, online gaming, etc.) and even more advanced requirements including inbound connections (IP cameras, web, DNS, email, VPN, etc.).

The above is not intended to be a comprehensive list of all the possible usage cases, just an overall view. In fact, combinations of the above usages are also possible, as well as situations where the same CE is used at different times in different scenarios or even different with services providers that may use a different transition mechanism.

The mechanisms for allowing inbound connections are "naturally" available in any IPv6 router when using GUA (IPv6 Global Unicast Addresses), unless they are blocked by firewall rules, which may require some manual configuration.

However, in the case of IPv4aaS, because the usage of private addresses and NAT and even depending on the specific transition mechanism, inbound connections typically require some degree of more complex manual configuration such as setting up a DMZ, virtual servers, or port/protocol forwarding. In general, IPv4 CE Routers already provide a GUI, CLI or API to manually configure them, or the possibility to setup the CE in bridge mode, so another CE behind it takes care of that. The requirements for that support are out of the scope of this document.

It is not relevant who provides the IPv6 Transition CE Router. In most of the cases it is the service provider, and in fact they are responsible, typically, of provisioning/managing at least the WAN side. Commonly, the user has access to configure the LAN interfaces, firewall, DMZ, and many other features. However, in many cases, the user must supply or may replace the IPv6 Transition CE Router. This underscores the importance of the IPv6 Transition CE Routers fulfilling the same requirements defined in this document.

The IPv6 Transition CE Router described in this document is not intended for usage in other scenarios such as large Enterprises, Data Centers, Content Providers, etc. So even if the documented requirements meet their needs, they may have additional requirements, which are out of the scope of this document.

12. Annex B: End-User Network Architecture

According to the descriptions in the preceding sections, an end-user network will likely support both IPv4 and IPv6. It is not expected that an end-user will change their existing network topology with the introduction of IPv6. There are some differences in how IPv6 works and is provisioned; these differences have implications for the network architecture.

A typical IPv4 end-user network consists of a "plug and play" router with NAT functionality and a single link upstream, connected to the service provider network.

From the perspective of an "IPv4 user" behind an IPv6 transition Customer Edge Router with IPv4aaS, this doesn't change.

However, while a typical IPv4 NAT deployment by default blocks all incoming connections and may allow opening of ports using a Universal

Plug and Play Internet Gateway Device (UPnP IGD) [UPnP-IGD] or some other firewall control protocol, in the case of an IPv6-only access and IPv4aaS, that may not be feasible depending on specific transition mechanism details. PCP (Port Control Protocol, [RFC6887]) may be an alternative solution.

Another consequence of using IPv4 private address space in the end-user network is that it provides stable addressing; that is, it doesn't change, even when you change service providers, and the addresses are always usable even when the WAN interface is down or the customer edge router has not yet been provisioned. In the case of an IPv6-only access, private IPv4 addresses are also available if the IPv4aaS transition mechanism keeps running the NAT interface towards the LAN side when the WAN interface is down.

More advanced routers support dynamic routing (which learns routes from other routers), and advanced end-users can build arbitrary, complex networks using manual configuration of address prefixes combined with a dynamic routing protocol. Once again, this is true for both IPv4 and IPv6.

In general, the end-user network architecture for IPv6 should provide equivalent or better capabilities and functionality than the current IPv4 architecture.

The end-user network is a stub network, in the sense that is not providing transit to other external networks. However, HNCP [RFC7788] allows support for automatic provisioning of downstream routers. Figure 2 illustrates the model topology for the end-user network.

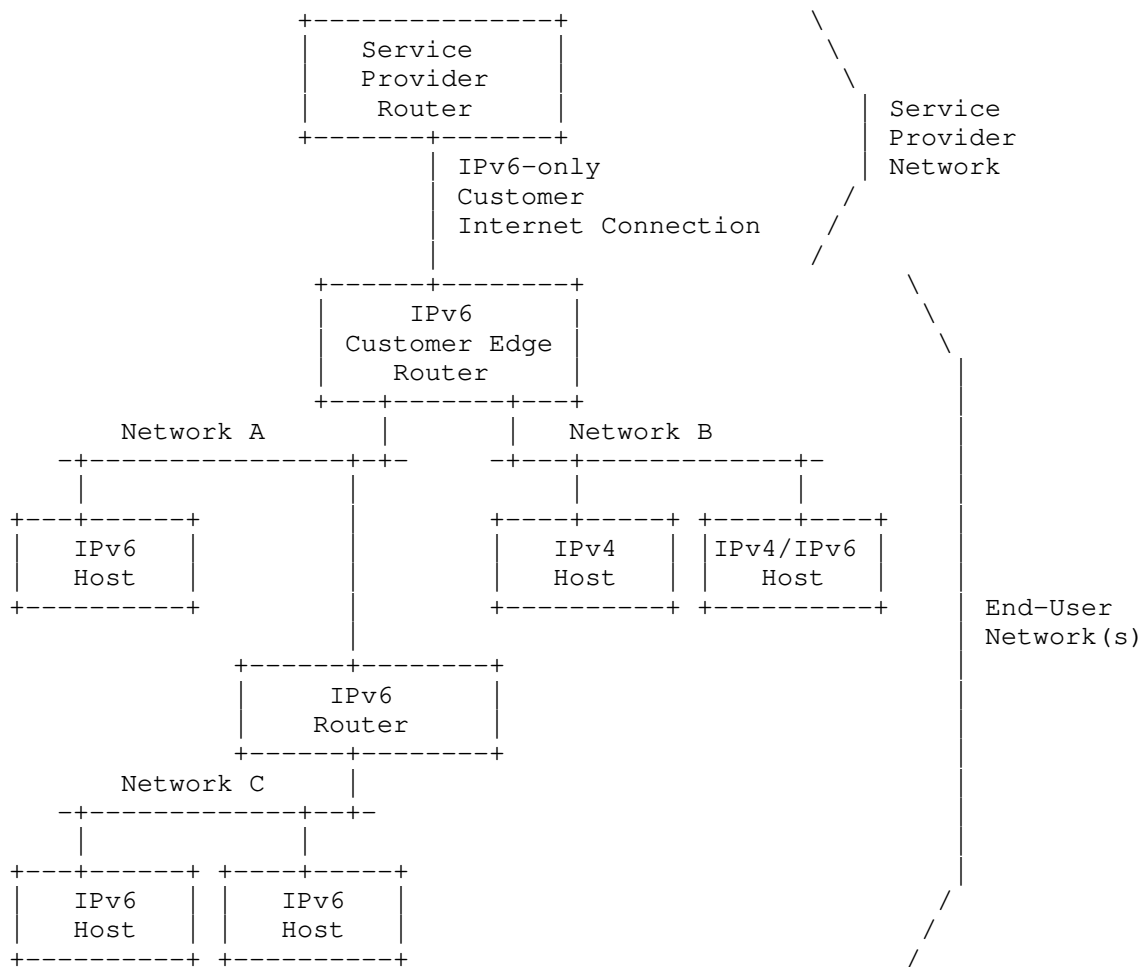


Figure 2: An Example of a Typical End-User Network

This architecture describes the:

- o Basic capabilities of the IPv6 Transition CE Router
- o Provisioning of the WAN interface connecting to the service provider
- o Provisioning of the LAN interfaces

The IPv6 Transition CE Router may be manually configured in an arbitrary topology with a dynamic routing protocol or using HNCP [RFC7788]. Automatic provisioning and configuration is described for

a single IPv6 Transition CE Router only.

13. ANNEX C: Changes from -00

Section to be removed by RFC Editor. Significant updates are:

1. ID-Nits: IANA section.
2. ID-Nits: RFC7084 reference removed from Abstract.
3. This document no longer updates RFC7084.
4. UPnP section reworded.
5. "CE Router" changed to "IPv6 Transition CE Router".
6. Reduced text in Annex A.

14. ANNEX D: Changes from -01

Section to be removed by RFC Editor. Significant updates are:

1. TRANS requirements reworked in order to increase operator control and allow gradual transitioning from dual-stack to IPv6-only on specific customers.
2. New TRANS requirement so all the supported transition mechanisms are disabled by default, in order to facilitate the operator management.
3. New TRANS requirement in order to allow turning on/off each transition mechanism by the user.
4. Clarification on how to obtain multiple /64 for 464XLAT.
5. S46 priority update to RFC8026 for including 464XLAT and related changes in several sections.

15. ANNEX E: Changes from -02

Section to be removed by RFC Editor. Significant updates are:

1. RFC8026 update removed, not needed with new approach.
2. TRANS and 464XLAT requirements reworded in order to match new approach to allow operator control on each/all the transition mechanisms.

3. Added text in 464XLAT to clarify the usage.

16. ANNEX F: Changes from -03

Section to be removed by RFC Editor. Significant updates are:

1. Several editorial changes across the document, specially TRANS requirements.
2. DNS proxy MUST instead of SHOULD.

17. ANNEX G: Changes from -04

Section to be removed by RFC Editor. Significant updates are:

1. Removed G-1.
2. Added support for draft-pref64folks-6man-ra-pref64.
3. General text clarifications.

18. ANNEX H: Changes from -05

Section to be removed by RFC Editor. Significant updates are:

1. Reworded and shorter UPnP section and new informative reference.
2. New general transition requirement in case multiple public IPv4 prefixes are provided, so to run multiple instances according to each specific transition mechanism.
3. General text clarifications.

19. ANNEX I: Changes from -06

Section to be removed by RFC Editor. Significant updates are:

1. Removed reference and text related to pref64folks-6man-ra-pref64.
2. General text clarifications.

20. ANNEX J: Changes from -07

Section to be removed by RFC Editor. Significant updates are:

1. Added text to UPnP section.

21. ANNEX K: Changes from -08, -09 and -10

Section to be removed by RFC Editor. Significant updates are:

1. Editorial edits.

22. ANNEX L: Changes from -11, -12, -13 and -14

Section to be removed by RFC Editor. Significant updates are:

1. Changes related to suggestions by Ops-dir, Sec-dir, Rtg-dir, Tsv-art and Gen-art, as well as comments from IESG review.
2. IANA section removed as a consequence of the removal of the inclusion of 464XLAT in the RFC8026 priority mechanism.

23. References

23.1. 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>>.
- [RFC5625] Bellis, R., "DNS Proxy Implementation Guidelines", BCP 152, RFC 5625, DOI 10.17487/RFC5625, August 2009, <<https://www.rfc-editor.org/info/rfc5625>>.
- [RFC5969] Townsley, W. and O. Troan, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) -- Protocol Specification", RFC 5969, DOI 10.17487/RFC5969, August 2010, <<https://www.rfc-editor.org/info/rfc5969>>.
- [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>>.
- [RFC6334] Hankins, D. and T. Mrugalski, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Option for Dual-Stack Lite", RFC 6334, DOI 10.17487/RFC6334, August 2011, <<https://www.rfc-editor.org/info/rfc6334>>.
- [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>>.
- [RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <<https://www.rfc-editor.org/info/rfc6887>>.
- [RFC6970] Boucadair, M., Penno, R., and D. Wing, "Universal Plug and Play (UPnP) Internet Gateway Device - Port Control Protocol Interworking Function (IGD-PCP IWF)", RFC 6970, DOI 10.17487/RFC6970, July 2013, <<https://www.rfc-editor.org/info/rfc6970>>.
- [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>>.
- [RFC7084] Singh, H., Beebee, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, DOI 10.17487/RFC7084, November 2013, <<https://www.rfc-editor.org/info/rfc7084>>.
- [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>>.
- [RFC7291] Boucadair, M., Penno, R., and D. Wing, "DHCP Options for the Port Control Protocol (PCP)", RFC 7291, DOI 10.17487/RFC7291, July 2014, <<https://www.rfc-editor.org/info/rfc7291>>.

- [RFC7341] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4-over-DHCPv6 (DHCP 4o6) Transport", RFC 7341, DOI 10.17487/RFC7341, August 2014, <<https://www.rfc-editor.org/info/rfc7341>>.
- [RFC7596] Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the Dual-Stack Lite Architecture", RFC 7596, DOI 10.17487/RFC7596, July 2015, <<https://www.rfc-editor.org/info/rfc7596>>.
- [RFC7597] Troan, O., Ed., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, Ed., "Mapping of Address and Port with Encapsulation (MAP-E)", RFC 7597, DOI 10.17487/RFC7597, July 2015, <<https://www.rfc-editor.org/info/rfc7597>>.
- [RFC7598] Mrugalski, T., Troan, O., Farrer, I., Perreault, S., Dec, W., Bao, C., Yeh, L., and X. Deng, "DHCPv6 Options for Configuration of Software Address and Port-Mapped Clients", RFC 7598, DOI 10.17487/RFC7598, July 2015, <<https://www.rfc-editor.org/info/rfc7598>>.
- [RFC7599] Li, X., Bao, C., Dec, W., Ed., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", RFC 7599, DOI 10.17487/RFC7599, July 2015, <<https://www.rfc-editor.org/info/rfc7599>>.
- [RFC7618] Cui, Y., Sun, Q., Farrer, I., Lee, Y., Sun, Q., and M. Boucadair, "Dynamic Allocation of Shared IPv4 Addresses", RFC 7618, DOI 10.17487/RFC7618, August 2015, <<https://www.rfc-editor.org/info/rfc7618>>.
- [RFC8026] Boucadair, M. and I. Farrer, "Unified IPv4-in-IPv6 Software Customer Premises Equipment (CPE): A DHCPv6-Based Prioritization Mechanism", RFC 8026, DOI 10.17487/RFC8026, November 2016, <<https://www.rfc-editor.org/info/rfc8026>>.
- [RFC8114] Boucadair, M., Qin, C., Jacquenet, C., Lee, Y., and Q. Wang, "Delivery of IPv4 Multicast Services to IPv4 Clients over an IPv6 Multicast Network", RFC 8114, DOI 10.17487/RFC8114, March 2017, <<https://www.rfc-editor.org/info/rfc8114>>.
- [RFC8115] Boucadair, M., Qin, J., Tsou, T., and X. Deng, "DHCPv6 Option for IPv4-Embedded Multicast and Unicast IPv6 Prefixes", RFC 8115, DOI 10.17487/RFC8115, March 2017, <<https://www.rfc-editor.org/info/rfc8115>>.

- [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>>.

23.2. Informative References

- [IPv6Survey] Palet Martinez, J., "IPv6 Deployment Survey", January 2018, <<https://indico.uknof.org.uk/event/41/contribution/5/material/slides/0.pdf>>.
- [OpenWRT] OpenWRT, "OpenWRT Packages", January 2018, <<https://openwrt.org/packages/start>>.
- [RFC7788] Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", RFC 7788, DOI 10.17487/RFC7788, April 2016, <<https://www.rfc-editor.org/info/rfc7788>>.
- [UPnP-IGD] UPnP Forum, "InternetGatewayDevice:2 Device Template Version 1.01", December 2010, <<http://upnp.org/specs/gw/igd2/>>.
- [UPnP-WANIPC] UPnP Forum, "WANIPConnection:2 Service", December 2010, <<http://upnp.org/specs/gw/UPnP-gw-WANIPConnection-v2-Service.pdf>>.

Authors' Addresses

Jordi Palet Martinez
The IPv6 Company
Molino de la Navata, 75
La Navata - Galapagar, Madrid 28420
Spain

Email: jordi.palet@theipv6company.com
URI: <http://www.theipv6company.com/>

Hans M.-H. Liu
D-Link Systems, Inc.
17595 Mount Herrmann St.
Fountain Valley, California 92708
US

Email: hans.liu@dlinkcorp.com
URI: <http://www.dlink.com/>

Masanobu Kawashima
NEC Platforms, Ltd.
800, Shimomata
Kakegawa-shi, Shizuoka 436-8501
Japan

Email: kawashimam@vx.jp.nec.com
URI: <https://www.necplatforms.co.jp/en/>

intarea
Internet-Draft
Intended status: Standards Track
Expires: April 4, 2019

R. Patterson
Sky UK
M. Abrahamsson
T-Systems
October 01, 2018

IP over Ethernet (IPoE) Session Health Checking
draft-patterson-intarea-ipoe-health-05

Abstract

PPP over Ethernet clients have the functionality to detect path unavailability by using PPP Keepalives. IP over Ethernet does not have this functionality, and it is not specified in the IETF when an IP over Ethernet client should consider its WAN connectivity down, unless there is a physical layer link down event.

This document describes a mechanism for IP over Ethernet clients to achieve connectivity validation, similar to that of PPP over Ethernet, by using BFD Echo, or an alternative health check mechanism.

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 April 4, 2019.

Copyright Notice

Copyright (c) 2018 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 (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
1.2. Terminology	3
2. Alternative Mitigations	4
3. IPoE Health Checks	4
3.1. Parameters	4
3.2. Startup	5
3.3. BFD Echo	5
3.4. IPoE Health Check Probe	5
4. IPoE Health Check DHCP Options	6
4.1. DHCPv6	6
4.2. DHCPv4	7
5. Recovery Behaviour	7
5.1. LAN Considerations	9
6. Multihomed Clients	9
7. Security Considerations	9
8. IANA Considerations	9
9. Acknowledgements	10
10. Appendix A. Changes from -00	10
11. Appendix B. Changes from -01	10
12. Appendix C. Changes from -02	10
13. Appendix D. Changes from -03	11
14. Appendix E. Changes from -04	11
15. References	11
15.1. Normative References	11
15.2. Informative References	12
15.3. URIs	13
Authors' Addresses	13

1. Introduction

PPP [RFC1661] makes use of regular LCP echos and replies to continually test the data link layer, if the peer fails to respond to a predetermined number of LCP echos, the PPP session is terminated and will return to the Link Dead phase, ready for reestablishing. IPoE currently lacks this functionality.

Physical link state change on an IPoE client can trigger the renewing of a DHCP lease, however any indirect upstream network changes are not visible to the IPoE client.

An outage or planned maintenance work on, for example, a Broadband Network Gateways (BNG) or intermediate DHCP Relay, can leave an IPoE client with a stale DHCP lease for up to the Valid Lifetime.

IPoE Session Health Check allows for an IPoE client to proactively or passively monitor the state of upstream connectivity, and defines several actions that may be taken to help the client recover.

[TR-146], Section 6.2 describes this problem, while [TR-124] identifies some requirements to solve the problem.

Several vendors have implemented subscriber connectivity checking on their BNG, using ARP and Neighbor Discovery as per Sections 6.2.4 and 6.2.5 of [TR-146]. This allows the BNG to detect loss of connectivity and to update local session state and DHCP lease bindings. Without reciprocal checking, this puts the CE at further risk of being in a stale state.

1.1. 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.

1.2. Terminology

This document makes use of the following terms:

- o BNG: Broadband Network Gateway. Often also running a DHCP server or relay.
- o CE: Customer Equipment. aka. Customer Premise Equipment (CPE), Residential Gateway (RG).
- o IPoE: IP over Ethernet.
- o IPoE Client: A network device, often a CE, running a DHCPv4 and/or DHCPv6 client.
- o IPoE Health Check: The name of the process described in this document.

2. Alternative Mitigations

- o Short DHCP lease times reduce the time a client may be left in a stale state, but scale poorly, putting extra load on the DHCP server.
- o Broadband Forum's [TR-146] and [TR-124] discuss this problem and recommend the use of BFD echo [RFC5880]. This document acknowledges TR-146 and recommends the use of BFD echo for health checks, but like the Broadband Forum, acknowledges that it is not widely available within consumer CEs. The renew action specified in TR-146 is often insufficient for a network to authenticate an IPoE Client, leaving the client in a stale state for up to the lease expiry, and no better off than without the check. This document introduces an alternative action to help expedite client recovery.
- o For planned maintenance, network engineers could include DHCPv4 Force Renew [RFC3203] or DHCPv6 Reconfigure [RFC3315]-bis in their maintenance plans, however neither of these have been widely adopted by CE or BNG vendors due to authentication complexity.

3. IPoE Health Checks

3.1. Parameters

IPoE Health Check uses the following parameters:

- o Interval (Integer): The frequency in seconds, which health checks are sent by the IPoE client. Default value: 120 seconds.
- o Retry Interval (Integer): The frequency in seconds, which health checks are sent by the IPoE client, after a failure. Default value: 10 seconds.
- o Limit (Integer): The number of failed consecutive checks before an action is taken. Default value: 3.
- o Release (Boolean): Instructs the client to send a DHCP RELEASE and to not attempt a renewal of the current lease. Default value: 0.

An IPoE client MAY be statically configured for IPoE health checks. Non-default static parameters SHOULD override any signalled via a dynamic means (e.g, DHCP or TR-69).

An IPoE client MAY use default parameters in lieu of manually configured, or dynamically signalled parameters (DHCP for example).

Statically configured or dynamically signalled parameters SHOULD override any default parameters.

3.2. Startup

An IPoE client supporting IPoE Health Check MUST begin sending health checks at the Retry Interval (Section 3.1) specified, upon successful binding of a lease that contains a valid IPoE Health Check DHCP Option (Section 4), or for which it has been statically configured.

After Limit IPoE health checks succeeds consecutively, the IPoE client MUST begin sending health checks at the regular Interval.

If Limit IPoE health checks fail consecutively, IPoE Health Check should be considered unusable and the IPoE client MUST cease the sending of health checks, ignoring the recovery behaviour (Section 5).

3.3. BFD Echo

If the IPoE Client supports BFD [RFC5880], it SHOULD use BFD Echo as the health check mechanism.

If the IPoE Client does not support BFD, or if it is unable to establish a BFD session with the upstream router, it MUST use IPoE Health Check Probe Section 3.4 as the health check mechanism.

3.4. IPoE Health Check Probe

Similar in format to a BFD Echo packet, the IPoE Health Check probe is a simple UDP/IP packet with a destination IP address from the lease being checked, and a destination MAC address of the upstream router. However, unlike BFD Echo, an IPoE Health Check probe does not require a BFD session to be established between peers; allowing for less state and configuration on a BNG and a simpler CPE implementation.

The destination IP MUST be an address locally bound on the IPoE client and MUST be from the lease triggering the IPoE Health Check.

The source IP SHOULD be the same as the destination address, but MAY be another address bound to the same interface the health check probe is being sent out. E.g. The link-local or a DHCPv6 NA assigned address.

The source MAC address MUST belong to the IPoE Client interface of the lease being checked.

The destination MAC address MUST be address of the upstream router.

Using an IP packet as the health check probe not only validates the layer 2 forwarding path, but also validates the DHCP lease state.

4. IPoE Health Check DHCP Options

This document defines a new option for both DHCPv4 and DHCPv6 servers to signal suggested health check parameters to clients. IPoE clients SHOULD use these values when no statically configured parameters have been defined.

The option data fields are common between DHCPv6 and DHCPv4.

4.1. DHCPv6

For DHCPv6, this option (Figure 1) MUST be within a specific Identity Association as an IPoE client MAY have multiple IAs with different health check parameters.

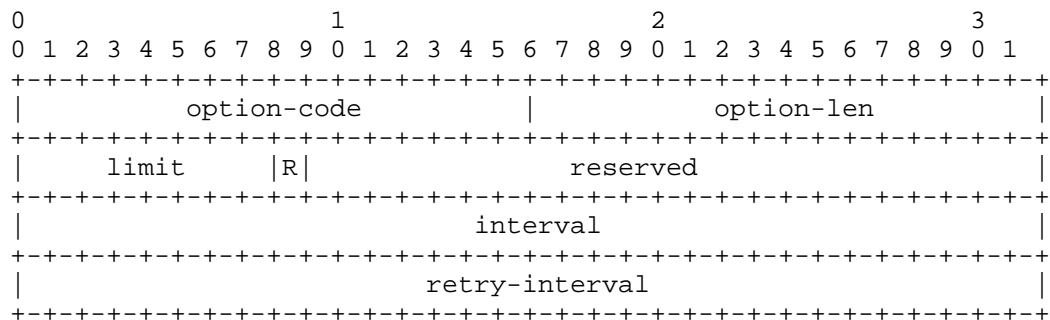


Figure 1: DHCPv6 IPoE Check Option Format

The description of the fields is as follows:

option-code: OPTION_IPOE_HEALTH (TBA1).
option-len: 12.
limit: Consecutive failed checks, before an action is taken.
R: Release flag. Instructs the client to send a DHCP RELEASE when a failure is detected, and to skip the renew step.
interval: Indicates how often a health check should be sent when no failure is encountered. Expressed in units of seconds.
retry-interval: Indicates how often a health check should be sent after a previous failure. Expressed in units of seconds.

4.2. DHCPv4

The DHCPv4 client can retrieve IPoE Health Check information by including `OPTION_IPOE_HEALTH` in a Parameter Request List option [RFC2132].

Figure 2 shows the DHCPv4 IPoE Health Check option.

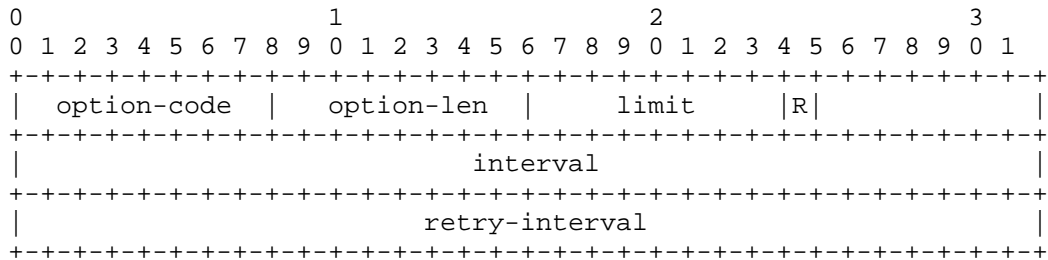


Figure 2: DHCPv4 IPoE Health Check Option Format

The description of the fields is as follows:

option-code: `OPTION_IPOE_HEALTH` (TBA2).
option-len: 10.
limit: Consecutive failed checks, before an action is taken.
R: Release flag. Instructs the client to send a DHCP RELEASE when a failure is detected, and to skip the renew step.
interval: Indicates how often a health check should be sent when no failure is encountered. Expressed in units of seconds.
retry-interval: Indicates how often a health check should be sent after a previous failure. Expressed in units of seconds.

5. Recovery Behaviour

IPoE Health Check defines the behaviour that an IPoE Client should take once the timeout threshold has been reached. This behaviour makes use of existing procedures outlined in [RFC2131], Section 4.4.5 for DHCPv4, and [RFC3315]-bis, Sections 18.2.4, 18.2.5 for DHCPv6.

When triggering a DISCOVER or SOLICIT, an IPoE client may also choose to use Rapid Commit [RFC4039], [RFC3315], Section 22.14 to help expedite the recovery process.

After Limit (Section 3.1) consecutive check failures, T1 and T2 MUST both be set to zero.

If the Release Flag (Section 3.1) is unset, the IPoE client SHOULD immediately attempt to renew the current lease from the original server. If connectivity to the original DHCP server has recovered, and the server can satisfy the request, the lease may be renewed and timers updated.

If the renew is unsuccessful, the IPoE client MUST move to the discovery or solicit phase.

If the Release Flag is unset, the IPoE client MUST keep the address or prefix in the preferred state until the preferred lifetime expires, and MUST keep the address or prefix until the valid lifetime expires.

If the Release Flag is set the IPoE Client MUST NOT send a DHCP RENEW, it MUST send a RELEASE as per [RFC2131], Section 3.1 for DHCPv4 and [RFC3315]-bis, Section 18.2.7.

If the IPoE client is already in the renew or rebind state when this behaviour is triggered, the client MUST cease renew or rebind attempts and wait for any outstanding messages to time out before sending a RELEASE.

If an outstanding renew or rebind attempt is successful, the IPoE client MUST update T1, T2 and lease lifetimes appropriately, and MUST NOT continue with this behaviour.

Once the RELEASE process has completed, the IPoE Client should move to the discovery or solicit phase.

The IPoE client MUST include the current address or prefix in the IA Address or IA_PD options within the DHCPv6 SOLICIT, or in the Requested IP Address Option of a DHCPv4 DISCOVER [RFC2131], Section 4.4.2.

The DHCPv6 SOLICIT DUID and IAID values MUST be the same as used in the current lease.

If the DHCP server assigns the current address or prefix, the IPoE client MUST update the current lease timers, and any differing parameters.

If the DHCP server assigns an alternate address or prefix, the IPoE client MUST deprecate the current lease and follow the actions outlined in requirement L-13 [RFC7084], Section 4.3

5.1. LAN Considerations

If all DHCPv6 leases have expired, either naturally or proactively with IPoE health checks, an IPoE client acting as a router, SHOULD withdraw itself as a default router on the LAN, following requirement G-5 of [RFC7084], Section 4.1.

6. Multihomed Clients

An IPoE client may have multiple leases from the same, or different DHCP servers. These leases may have different IPoE health check parameters, and health checks MUST be treated distinctly, tracking the particular lease that they belong to.

Local network administrators may choose to override DHCP-signalled parameters in order to facilitate appropriate IPoE Health Check operation in a multihomed environment.

7. Security Considerations

IPoE Health Check frequency would typically be controlled by the network using DHCP options, but overly aggressive, statically configured IPoE Health Checks, could have an adverse impact. For example, this may induce an overload on the IP access nodes. However, BFD Echo and the IPoE Health Check probe defined in this document, both use an IP packet destined for the IPoE client, the remote peer forwards the packet back to the IPoE client without any local processing.

The inclusion of the current lease address or prefix when sending a DISCOVER or SOLICIT, introduces a privacy risk, possibly leaking lease information if the IPoE client has been moved to a different network, e.g., from one fixed line provider to another.

8. IANA Considerations

IANA is requested to assign a new DHCPv6 Option Code in the registry maintained in <http://www.iana.org/assignments/dhcpv6-parameters> [1]:

Option Name	Value
OPTION_IPOE_HEALTH	TBA1

Also, IANA is requested to assign the following new DHCPv4 Option Code in the registry maintained in <http://www.iana.org/assignments/bootp-dhcp-parameters> [2]:

Option Name	Tag	Data Length	Meaning
OPTION_IPOE_HEALTH	TBA2	14	Provides a set of IPoE check configuration information.

9. Acknowledgements

The authors would like thank Ian Farrer, Dusan Mudric, Mohamed Boucadair, Jean-Yves Cloarec, Bernie Volz, Barbara Stark, Dave Freedman, and Job Snijders for their review and comments on this and previous versions of this document.

10. Appendix A. Changes from -00

This section should be removed by the RFC Editor.

- o Added reference to TR-146.
- o Added BFD Echo section, and wording to prefer it as the health check mechanism over ARP/ND, if available.

11. Appendix B. Changes from -01

This section should be removed by the RFC Editor.

- o Emphasised preference for use of BFD echo as the health check mechanism.
- o Removed lifetime expiration from Behaviour 2 and clarified usage.
- o Updated Behaviour 3 with instructions for whilst mid-renew/rebind.
- o Reworded multihoming section.
- o Added Acknowledgements.

12. Appendix C. Changes from -02

This section should be removed by the RFC Editor.

- o Added DHCP option flag to force ARP/ND for health checks.
- o Populated IANA Considerations.
- o Added Retry Interval distinct timer for between failed checks.
- o Added default parameter values.

13. Appendix D. Changes from -03

This section should be removed by the RFC Editor.

- o Reduced default Limit value.
- o Formatting and minor cosmetic changes.

14. Appendix E. Changes from -04

This section should be removed by the RFC Editor.

This revision is a major rewrite. Changes include:

- o Consolidated the multiple behaviours down to one.
- o Added a Release flag instead of a distinct behaviour.
- o Added custom IPoE Health Check Probe definition.
- o Removed ARP/ND and Passive checks.
- o Removed alternative target address parameter.
- o Removed IANA registry request for Behaviour values.

15. References

15.1. Normative References

- [RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <<https://www.rfc-editor.org/info/rfc826>>.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <<https://www.rfc-editor.org/info/rfc2131>>.
- [RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, DOI 10.17487/RFC2132, March 1997, <<https://www.rfc-editor.org/info/rfc2132>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <<https://www.rfc-editor.org/info/rfc3315>>.

- [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>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.

15.2. Informative References

- [RFC1661] Simpson, W., Ed., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, DOI 10.17487/RFC1661, July 1994, <<https://www.rfc-editor.org/info/rfc1661>>.
- [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>>.
- [RFC3203] T'Joens, Y., Hublet, C., and P. De Schrijver, "DHCP reconfigure extension", RFC 3203, DOI 10.17487/RFC3203, December 2001, <<https://www.rfc-editor.org/info/rfc3203>>.
- [RFC4039] Park, S., Kim, P., and B. Volz, "Rapid Commit Option for the Dynamic Host Configuration Protocol version 4 (DHCPv4)", RFC 4039, DOI 10.17487/RFC4039, March 2005, <<https://www.rfc-editor.org/info/rfc4039>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.
- [RFC7084] Singh, H., Beebee, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, DOI 10.17487/RFC7084, November 2013, <<https://www.rfc-editor.org/info/rfc7084>>.
- [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>>.
- [TR-124] "Functional Requirements for Broadband Residential Gateway Devices", 2006, <<https://www.broadband-forum.org/technical/download/TR-124.pdf>>.
- [TR-146] "Subscriber Sessions", 2013, <<https://www.broadband-forum.org/technical/download/TR-146.pdf>>.

15.3. URIs

[1] <http://www.iana.org/assignments/dhcpv6-parameters>

[2] <http://www.iana.org/assignments/bootp-dhcp-parameters>

Authors' Addresses

Richard Patterson
Sky UK

Email: richard.patterson@sky.uk

Mikael Abrahamsson
T-Systems

Email: mikael.abrahamsson@t-systems.se

IPv6 Maintenance
Internet-Draft
Intended status: Standards Track
Expires: January 20, 2019

L. Colitti
E. Kline
J. Linkova
Google
July 19, 2018

Discovering PREF64 in Router Advertisements
draft-pref64folks-6man-ra-pref64-00

Abstract

This document specifies a Router Advertisement option to configure the NAT64 prefix.

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 January 20, 2019.

Copyright Notice

Copyright (c) 2018 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	2
2. Why include the NAT64 prefix in Router Advertisements	2
3. Semantics	3
4. Option format	3
5. IANA Considerations	3
6. Security Considerations	4
7. References	4
7.1. Normative References	4
7.2. Informative References	4
7.3. URIs	5
Authors' Addresses	6

1. Introduction

NAT64 [RFC6146] with DNS64 [RFC6147] is a widely-deployed mechanism to provide IPv4 access on IPv6-only networks. In order to support functions such as local validation of DNSSEC [RFC4033] responses, 464xlat [RFC6877], and local IPv4 address synthesis [RFC8305], the host must be aware of the NAT64 prefix in use by the network. This document specifies a Router Advertisement [RFC4861] option to communicate the NAT64 prefix to hosts.

1.1. Requirements Language

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 RFC 2119 [RFC2119].

2. Why include the NAT64 prefix in Router Advertisements

Fate sharing. NAT64 requires a routing to be configured. IPv6 routing configuration requires receiving an IPv6 Router Advertisement [RFC4861]. Compared to currently-deployed NAT64 prefix discovery methods such as [RFC7050], including the NAT64 prefix in the Router Advertisement minimizes the number of packets required to configure a host. This speeds up the process of connecting to a network that supports NAT64/DNS64, and simplifies host implementation by removing the possibility that the a can have an incomplete layer 3 configuration (e.g., IPv6 addresses and prefixes, but no NAT64 prefix).

Deployability. All IPv6 hosts and networks are required to support [RFC4861]. Other options such as [RFC7225]b require implementing other protocols.

3. Semantics

This option specifies exactly one NAT64 prefix for all IPv4 addresses. Observation of current deployments suggest that they use RFC7050, which also only supports one prefix.

This option only supports the NAT64 prefix length of 96 bits. In the unlikely event of use cases for shorter prefixes ([RFC6052]) emerging another option could be created.

4. Option format

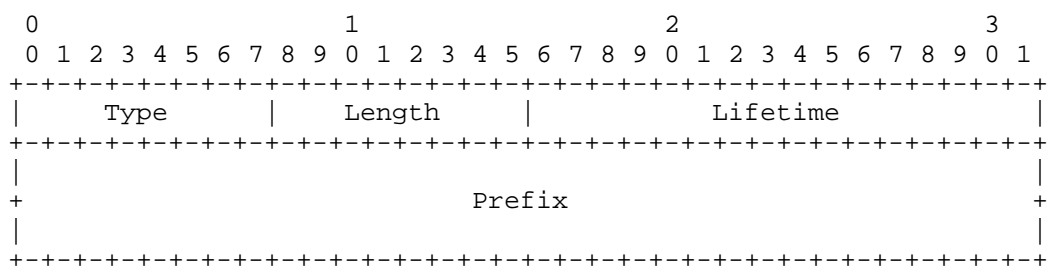


Figure 1: NAT64 Prefix Option Format

Fields:

- Type 8-bit identifier of the RDNSS option type as assigned by IANA: TBD
- Length 8-bit unsigned integer. The length of the option (including the Type and Length fields) is in units of 8 octets. The only valid value for this field is 2. A host MUST ignore the NAT64 prefix option if the length field value is not set to 2.
- Lifetime 16-bit unsigned integer. The maximum time in seconds over which this NAT64 prefix MAY be used. The value of Lifetime SHOULD by default be set to lesser of 3 x MaxRtrAdvInterval or 65535 seconds. A value of zero means that the prefix MUST no longer be used.
- Prefix The highest 96-bits of the NAT64 prefix.

5. IANA Considerations

The IANA is requested to assign a new IPv6 Neighbor Discovery Option type for the PREF64 option defined in this document.

Option Name	Type
PREF64 option	(TBD)

Table 1

The IANA registry for these options is:

<https://www.iana.org/assignments/icmpv6-parameters> [1]

6. Security Considerations

Because Router Advertisements are required in all IPv6 configuration scenarios, on IPv6-only networks, Router Advertisements must already be secured, e.g., by deploying RA guard [RFC6105]. Providing all configuration in Router Advertisements increases security by ensuring that no other protocols can be abused by malicious attackers to provide hosts with invalid configuration.

The security measures that must already be in place to ensure that Router Advertisements are only received from legitimate sources eliminate the problem of PREF64 validation described in section 3.1 of [RFC7050].

7. References

7.1. 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>>.
- [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>>.

7.2. Informative References

- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<https://www.rfc-editor.org/info/rfc4033>>.

- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <<https://www.rfc-editor.org/info/rfc8305>>.

7.3. URIs

- [1] <https://www.iana.org/assignments/icmpv6-parameters>

Authors' Addresses

Lorenzo Colitti
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: lorenzo@google.com

Erik Kline
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: ek@google.com

Jen Linkova
Google
1 Darling Island Rd
Pyrmont, NSW 2009
AU

Email: furry@google.com

IPv6 Maintenance
Internet-Draft
Intended status: Standards Track
Expires: April 4, 2019

L. Colitti
E. Kline
J. Linkova
Google
October 1, 2018

Discovering PREF64 in Router Advertisements
draft-pref64folks-6man-ra-pref64-02

Abstract

This document specifies a Router Advertisement option to communicate NAT64 prefixes to clients.

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 April 4, 2019.

Copyright Notice

Copyright (c) 2018 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	2
1.2. Terminology	2
2. Why include the NAT64 prefix in Router Advertisements	3
3. Semantics	3
4. Option format	4
5. Handling Multiple NAT64 Prefixes	4
6. Multihoming	5
7. IANA Considerations	6
8. Security Considerations	6
9. Acknowledgements	6
10. References	6
10.1. Normative References	6
10.2. Informative References	7
10.3. URIs	8
Authors' Addresses	8

1. Introduction

NAT64 [RFC6146] with DNS64 [RFC6147] is a widely-deployed mechanism to provide IPv4 access on IPv6-only networks. In order to support functions such as local validation of DNSSEC [RFC4033] responses, 464xlat [RFC6877], and local IPv4 address synthesis [RFC8305], the host must be aware of the NAT64 prefix in use by the network. This document specifies a Router Advertisement [RFC4861] option to communicate the NAT64 prefix to hosts.

1.1. Requirements Language

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 RFC 2119 [RFC2119].

1.2. Terminology

Pref64: an IPv6 prefix used for IPv6 address synthesis [RFC6146];

RA Router Advertisement, a message used by IPv6 routers to advertise their presence together with various link and Internet parameters ([RFC4861]);

PvD-aware host A host that supports the association of network configuration information into PvDs and the use of these PvDs. Also named PvD-aware node in [RFC7556].

2. Why include the NAT64 prefix in Router Advertisements

Fate sharing: NAT64 requires a routing to be configured. IPv6 routing configuration requires receiving an IPv6 Router Advertisement [RFC4861]. Compared to currently-deployed NAT64 prefix discovery methods such as [RFC7050], including the NAT64 prefix in the Router Advertisement minimizes the number of packets required to configure a host. This speeds up the process of connecting to a network that supports NAT64/DNS64, and simplifies host implementation by removing the possibility that the host can have an incomplete layer 3 configuration (e.g., IPv6 addresses and prefixes, but no NAT64 prefix).

Deployability: all IPv6 hosts and networks are required to support [RFC4861]. Other options such as [RFC7225] require implementing other protocols.

3. Semantics

For simplicity, this option only supports a NAT64 prefix length of 96 bits, as this is by the most common configuration used by hosts. Networks using one of the other prefix lengths supported in ([RFC6052]) can use other mechanisms such as [RFC7050] or [RFC7225]. If different prefix lengths become common, another RA option can be created to configure them.

This option may appear more than once in a Router Advertisement. Host behaviour with regards to synthesizing IPv6 addresses from IPv4 addresses SHOULD follow the recommendations given in Section 3 of [RFC7050], limited to the NAT64 prefixes that have non-zero lifetime.

This option specifies exactly one NAT64 prefix for all IPv4 destinations. If the network operator desires to route different parts of the IPv4 address space to different NAT64 devices, this can be accomplished by routing more specifics of the NAT64 prefix to those devices. For example, if the operator would like to route 10.0.0.0/8 through NAT64 device A and the rest of the IPv4 space through NAT64 device B, and the operator's NAT64 prefix is 2001:db8:a:b::/96, then the operator can route 2001:db8:a:b::a00:0/104 to NAT64 A and 2001:db8:a:b::/64 to NAT64 B.

In a network that provides both IPv4 and NAT64, it may be desirable for certain IPv4 addresses not to be translated. An example might be private address ranges that are local to the network and should not be reached through the NAT64. This type of configuration cannot be conveyed to hosts using this option, or through other NAT64 prefix provisioning mechanisms such as [RFC7050] or [RFC7225]. This problem does not apply in IPv6-only networks, because in such networks, the

host does not have an IPv4 address and cannot reach any IPv4 destinations without the NAT64.

4. Option format

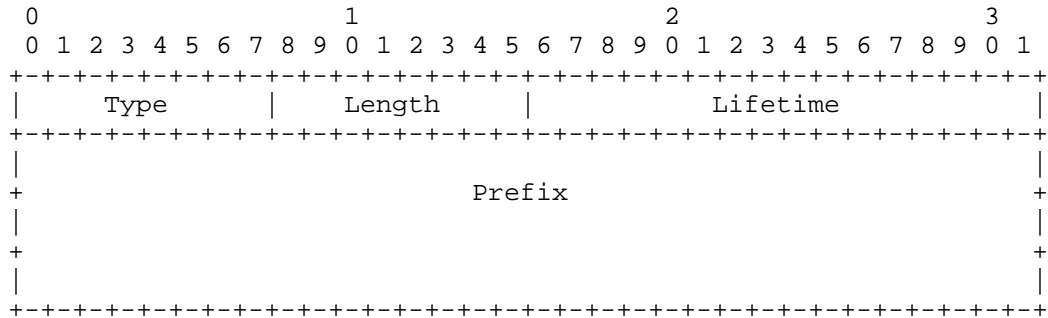


Figure 1: NAT64 Prefix Option Format

Fields:

- Type 8-bit identifier of the RDNSS option type as assigned by IANA: TBD
- Length 8-bit unsigned integer. The length of the option (including the Type and Length fields) is in units of 8 octets. The sender MUST set the Length to 2. A host MUST ignore the NAT64 prefix option if the length field value is 1. If the Length field value exceeds 2, the host MUST utilize the first 16 octets and ignore the rest of the option.
- Lifetime 16-bit unsigned integer. The maximum time in seconds over which this NAT64 prefix MAY be used. The value of Lifetime SHOULD by default be set to lesser of 3 x MaxRtrAdvInterval or 65535 seconds. A value of zero means that the prefix MUST no longer be used.
- Prefix The 96-bit NAT64 prefix.

5. Handling Multiple NAT64 Prefixes

In some cases a host may receive multiple NAT64 prefixes from different sources. Possible scenarios include (but are not limited to):

- o the host is using multiple mechanisms to discover Pref64 prefixes (e.g. by using PCP ([RFC7225]) and/or by resolving IPv4-only fully

qualified domain name ([RFC7050]) in addition to receiving the Pref64 RA option);

- o The pref64 option presents in a single RA more than once;
- o the host receives multiple RAs with different Pref64 prefixes on one or multiple interfaces.

When multiple Pref64 were discovered via RA Pref64 Option (the Option presents more than once in a single RA or multiple RAs were received), host behaviour with regards to synthesizing IPv6 addresses from IPv4 addresses SHOULD follow the recommendations given in Section 3 of [RFC7050], limited to the NAT64 prefixes that have non-zero lifetime..

When different Pref64 are discovered by using multiple mechanisms, hosts SHOULD select one source of information only. The RECOMMENDED order is:

- o PCP-discovered prefixes ([RFC7225]), if supported;
- o Pref64 discovered via RA Option;
- o Pref64 resolving IPv4-only fully qualified domain name ([RFC7050])

Note that if the network provides Pref64 both via this RA option and [RFC7225], hosts that receive the Pref64 via RA option may choose to use it immediately before waiting for PCP to complete, and therefore some traffic may not reflect any more detailed configuration provided by PCP.

6. Multihoming

Like most IPv6 configuration information, the Pref64 option is specific to the network on which it is received. For example, a Pref64 option received on a particular wireless network may not be usable unless the traffic is also sourced on that network. Similarly, a host connected to a cellular network that provides NAT64 generally cannot use that NAT64 for destinations reached through a VPN tunnel that terminates outside that network.

Thus, correct use of this option on a multihomed host generally requires the host to be PVD-aware.

This issue is not specific to the Pref64 RA option and, for example, is quite typical for DNS resolving on multihomed hosts (e.g. a host might resolve a destination name by using the corporate DNS server

via the VPN tunnel but then send the traffic via its Internet-facing interface).

7. IANA Considerations

The IANA is requested to assign a new IPv6 Neighbor Discovery Option type for the PREF64 option defined in this document.

Option Name	Type
PREF64 option	(TBD)

Table 1

The IANA registry for these options is:

<https://www.iana.org/assignments/icmpv6-parameters> [1]

8. Security Considerations

Because Router Advertisements are required in all IPv6 configuration scenarios, on IPv6-only networks, Router Advertisements must already be secured, e.g., by deploying RA guard [RFC6105]. Providing all configuration in Router Advertisements increases security by ensuring that no other protocols can be abused by malicious attackers to provide hosts with invalid configuration.

The security measures that must already be in place to ensure that Router Advertisements are only received from legitimate sources eliminate the problem of NAT64 prefix validation described in section 3.1 of [RFC7050].

9. Acknowledgements

Thanks to the following people (in alphabetical order) for their review and feedback: Brian E Carpenter, Nick Heatley, David Schinazi.

10. References

10.1. 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>>.

- [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>>.

10.2. Informative References

- [I-D.ietf-intarea-provisioning-domains] Pfister, P., Vyncke, E., Pauly, T., Schinazi, D., and W. Shao, "Discovering Provisioning Domain Names and Data", draft-ietf-intarea-provisioning-domains-02 (work in progress), June 2018.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<https://www.rfc-editor.org/info/rfc4033>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.

- [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>>.
- [RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <<https://www.rfc-editor.org/info/rfc7556>>.
- [RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <<https://www.rfc-editor.org/info/rfc8305>>.

10.3. URIs

- [1] <https://www.iana.org/assignments/icmpv6-parameters>

Authors' Addresses

Lorenzo Colitti
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: lorenzo@google.com

Erik Kline
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: ek@google.com

Jen Linkova
Google
1 Darling Island Rd
Pyrmont, NSW 2009
AU

Email: furry@google.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 5, 2021

F. Templin, Ed.
Boeing Research & Technology
January 1, 2021

IPv6 Prefix Delegation and Multi-Addressing Models
draft-templin-v6ops-pdhost-27

Abstract

Requesting nodes typically acquire IPv6 prefixes from a prefix delegation service for the network. The requesting node can provision the prefix according to whether it acts as a router on behalf of any downstream networks and/or as a host on behalf of its local applications. In the latter case, the requesting node can use portions of the delegated prefix for its own multi-addressing purposes. This document therefore considers prefix delegation models for both the classic routing and various multi-addressing use cases.

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 July 5, 2021.

Copyright Notice

Copyright (c) 2021 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 (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	6
3. Multi-Addressing Considerations	6
4. Multi-Addressing Alternatives for Delegated Prefixes	7
5. Address Autoconfiguration Considerations	8
6. MLD/DAD Implications	8
7. Dynamic Routing Protocol Implications	9
8. IPv6 Neighbor Discovery Implications	9
9. Prefix Delegation Services	9
10. IANA Considerations	10
11. Security Considerations	10
12. Acknowledgements	10
13. References	11
13.1. Normative References	11
13.2. Informative References	12
Appendix A. Change Log	13
Author's Address	14

1. Introduction

IPv6 Neighbor Discovery (ND) is the process by which nodes on the link discover each other's presence as well as advertise and receive configuration information. IPv6 Prefix Delegation (PD) entails 1) the communication of a prefix from a delegation service to a requesting node, 2) a representation of the prefix in the network's Routing Information Base (RIB) and the first-hop router's Forwarding Information Base (FIB), and 3) a control messaging service to maintain prefix lifetimes. Following delegation, the prefix is available for the requesting node's exclusive use and is not shared with any other nodes. This document considers prefix delegation models and multiaddressing considerations for requesting nodes that act as a router on behalf of any downstream networks and/or as a host on behalf of their local applications.

For nodes that connect downstream-attached networks (e.g., a cellphone that connects a "tethered" Internet of Things (IoT), a laptop computer with a complex internal network of virtual machines, etc.), the classic routing model applies as shown in Figure 1:

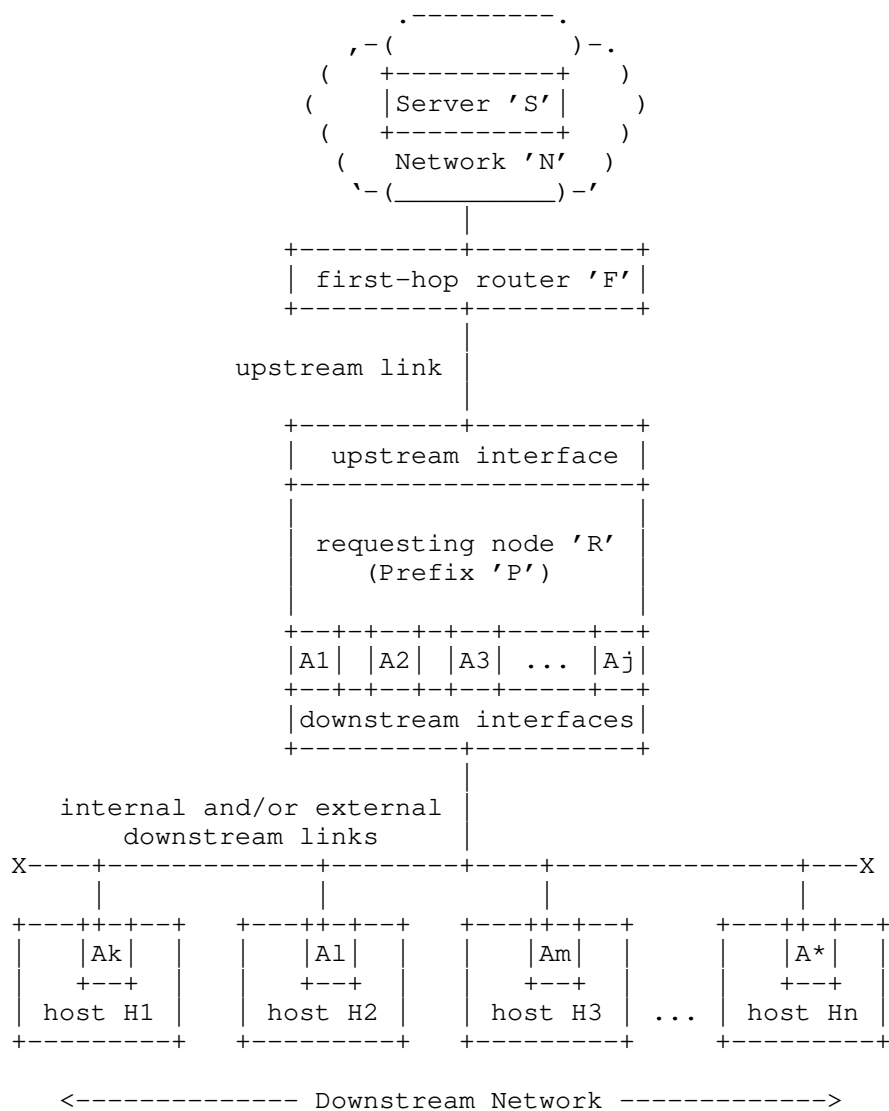


Figure 1: Classic Routing Model

In the classic routing model, requesting node 'R' has one or more upstream interfaces and connects zero or more internal and/or external downstream networks. When 'R' requests a prefix delegation, the following sequence of events transpires:

- o Server 'S' located in network 'N' delegates prefix 'P' to requesting node 'R'.

- o 'P' is injected into the RIB for 'N', and first hop router 'F' configures a FIB entry with 'R' as the next hop.
- o R' receives 'P' and assigns zero or more addresses 'A(*)' taken from 'P' to its downstream interfaces
- o 'R' advertises zero or more sub-prefixes taken from 'P' to hosts 'H(i)' on downstream networks.
- o 'R' delegates zero or more sub-prefixes taken from 'P' to requesting nodes in downstream networks.
- o 'R' acts as a router for hosts 'H(i)' on downstream networks and as a host on behalf of its local applications.

This document also considers the case when 'R' uses portions of 'P' for its own internal multi-addressing purposes. [RFC7934] provides Best Current Practice (BCP) motivations for the benefits of multi-addressing, while an operational means for providing nodes with multiple addresses is given in [RFC8273]. The following multi-addressing alternatives for delegated prefixes compliment this framework.

In a first alternative, when requesting node 'R' receives prefix 'P', it can assign addresses taken from 'P' to downstream virtual interfaces (e.g., a loopback) as shown in Figure 2:

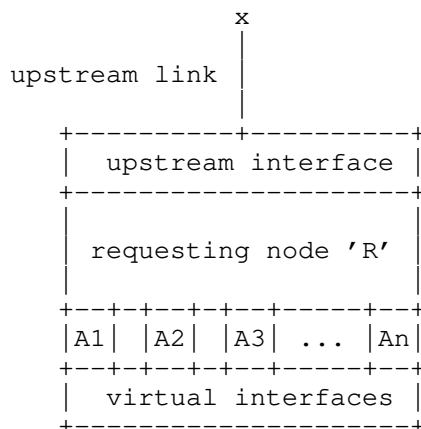


Figure 2: Address Assignment to Downstream Virtual Interfaces

In a second alternative, 'R' could assign IPv6 addresses taken from 'P' to the upstream interface over which the prefix was received as shown in Figure 3:

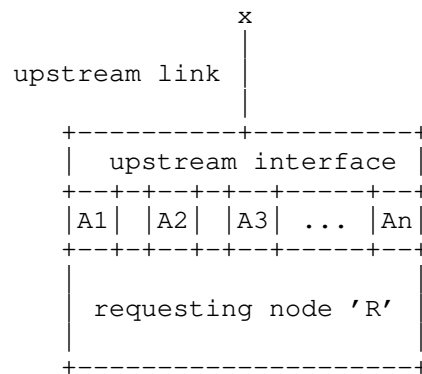


Figure 3: Upstream Interface Address Assignment

In a third alternative, 'R' could assign IPv6 addresses taken from 'P' to its local applications which appear as "psuedo" virtual interfaces as shown in Figure 4:

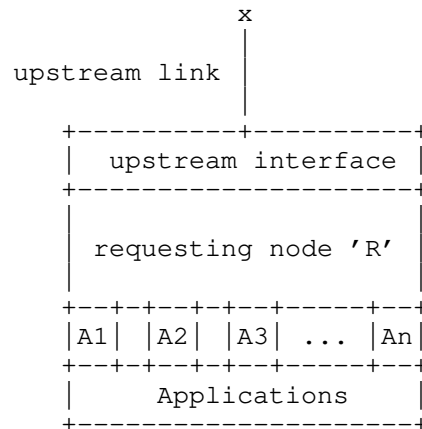


Figure 4: Application Addressing Model

With these IPv6 PD-based multi-addressing considerations, the node can configure an unlimited supply of addresses to make them available for local applications without requiring coordination with other nodes on upstream interfaces. The following sections present considerations for nodes that employ IPv6 PD mechanisms.

2. Terminology

The terms "node", "host" and "router" are the same as defined in [RFC8200]. The terms Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS), Neighbor Advertisement (NA), Redirect and Prefix Information Option (PIO) are the same as defined in [RFC4861]. All other terminology in the normative references applies, while the following terms are defined within the context of this document:

shared prefix

an IPv6 prefix that may be advertised to more than one node on the link. The router that advertises the prefix must consider the prefix as on-link so that the IPv6 ND address resolution function will identify the correct neighbor for each packet.

individual prefix

an IPv6 prefix that is advertised to exactly one node on the link, where the node may be unaware that the prefix is individual and may not participate in prefix maintenance procedures. The router that advertises the prefix can consider the prefix as on-link or not on-link. In the former case, the router performs address resolution and only forwards those packets that match one of the node's configured addresses so that the node will not receive unwanted packets. In the latter case, the router can simply forward all packets matching the prefix to the node which must then drop any packets that do not match one of its configured addresses. An example individual prefix service is documented in [RFC8273].

delegated prefix

an IPv6 prefix that is explicitly conveyed to a node for its own exclusive use, where the node is an active participant in prefix delegation and maintenance procedures. The first-hop router simply forwards all packets matching the prefix to the requesting node. The requesting node associates the prefix with downstream and/or internal virtual interfaces (i.e., and not the upstream interface).

3. Multi-Addressing Considerations

IPv6 allows nodes to assign multiple addresses to a single interface. [RFC7934] discusses options for multi-addressing as well as use cases where multi-addressing may be desirable. Address configuration options for multi-addressing include Stateless Address AutoConfiguration (SLAAC) [RFC4862], Dynamic Host Configuration Protocol for IPv6 (DHCPv6) address configuration [RFC8415], manual configuration, etc.

Nodes configure addresses from a shared or individual prefix and assign them to the upstream interface over which the prefix was received. When the node assigns the addresses, it is required to use Multicast Listener Discovery (MLD) [RFC3810] to join the appropriate solicited-node multicast group(s) and to use the Duplicate Address Detection (DAD) algorithm [RFC4862] to ensure that no other node configures a duplicate address.

In contrast, a node that configures addresses from a delegated prefix can assign them without invoking MLD/DAD on an upstream interface, since the prefix has been delegated to the node for its own exclusive use and is not shared with any other nodes.

4. Multi-Addressing Alternatives for Delegated Prefixes

When a node receives a delegated prefix, it has many alternatives for provisioning the prefix to its local interfaces and/or downstream networks. [RFC7278] discusses alternatives for provisioning a prefix obtained by a User Equipment (UE) device under the 3rd Generation Partnership Program (3GPP) service model. This document considers the more general case when the node receives a delegated prefix explicitly provided for its own exclusive use.

When the node receives the prefix, it can distribute the prefix to internal (virtual) or external (physical) downstream networks and optionally configure addresses for itself on downstream interfaces. The node then acts as a router on behalf of its downstream networks.

The node could instead (or in addition) use portions of the delegated prefix for its own multi-addressing purposes. In a first alternative, the node can assign as many addresses as it wants from the prefix to downstream virtual interfaces.

In a second alternative, the node can assign as many addresses as it wants from the prefix to the upstream interface over which the prefix was received, but in normal practice does not assign the prefix itself (or subnets from the prefix) to the upstream interface. If the node assigned the prefix to the upstream interface, any neighbors on the upstream link receiving an RA could configure addresses from the prefix and a default route with the node as the next hop. This could create a loop where upstream link neighbors send packets to the node which in turn forwards them to another upstream link neighbor. Still, there may be cases where the node provides services for dependent neighbors on the upstream link that have no other means of connecting to the network. ([RFC8415] chose to remain silent on this subject since it is operational rather than functional in nature.)

In a third alternative, the node can assign addresses taken from the delegated prefix to its local applications. The applications themselves then serve as virtual interfaces. (Note that, in the future, the practice of assigning unique non-link-local IPv6 addresses to applications could obviate the need for transport protocol port numbers.)

In these multi-addressing cases, the node normally assigns the prefix itself to a virtual interface such as a loopback so that unused portions of the prefix are correctly identified as unreachable. The node then acts as a host on behalf of its local applications even though neighbors on the upstream link consider it as a router.

5. Address Autoconfiguration Considerations

Nodes autoconfigure addresses according to Section 6 of IPv6 Node Requirements [RFC8504].

Nodes that connect to a network that spans more than just a single LAN configure at least one non-link-local address, i.e., for network management and error reporting purposes.

Nodes recognize the Subnet Router Anycast address [RFC4291] for each delegated prefix. Therefore, the node's use of the Subnet Router Anycast address must be indistinguishable from the behavior of an ordinary router when viewed from the outside world.

6. MLD/DAD Implications

When a node configures addresses for itself from a shared or individual prefix (and when the interface variable 'DupAddrDetectTransmits' is non-zero [RFC4862]), the node performs MLD/DAD by sending multicast messages over the upstream interface to test whether there is another node on the link that configures a duplicate address. When there are many such addresses and/or many such nodes, this could result in substantial multicast traffic that affects all nodes on the link.

When a node configures addresses for itself from a delegated prefix and assigns them on downstream interfaces, it can configure as many addresses as it wants without performing MLD/DAD for any of the addresses over the upstream interface.

When a node configures addresses for itself from a delegated prefix and assigns them on the upstream interface over which the prefix was received, the node honors MLD/DAD procedures according to the interface's 'DupAddrDetectTransmits' variable.

7. Dynamic Routing Protocol Implications

Nodes that receive delegated prefixes can be configured to either participate or not participate in a dynamic routing protocol over the upstream interface. When there are many nodes on the upstream link, dynamic routing protocol participation might be impractical due to scaling limitations, and may also be exacerbated by factors such as node mobility.

Unless it participates in a dynamic routing protocol, the node initially has only a default route pointing to a neighbor via an upstream interface. This means that packets sent by the node over an upstream interface will initially go through a default router even if there is a better first-hop node on the link. The node may subsequently receive Redirect messages from the default router that identify a better first-hop.

8. IPv6 Neighbor Discovery Implications

According to [RFC4861], when a node receives a shared or individual prefix with "L=1" and has a packet to send to an IPv6 destination within the prefix, it is required to use the IPv6 ND address resolution function to resolve the link-layer address of a neighbor on the link that configures the address.

Also according to [RFC4861], when a node receives a shared or individual prefix with "L=0" and has a packet to send to an IPv6 destination within the prefix, it sends the packet to a default router since "L=0" makes no statement about on-link or off-link properties of the prefix.

When a node requires a delegated prefix, it acts as a simple host by sending RS messages over the upstream interface in the manner described in Section 4.2 of [RFC7084] and invokes prefix delegation services as discussed in Section 9. The node considers the upstream interface as a non-advertising interface [RFC4861], i.e., it does not send RA messages over the upstream interface. The node further does not perform the IPv6 ND address resolution function over the upstream interface, since the delegated prefix is by definition not associated with the upstream interface.

9. Prefix Delegation Services

Selection of prefix delegation services must be considered according to specific use cases. An example service is that offered by standard DHCPv6 Prefix Delegation [RFC8415]. Alternative services based on IPv6 ND messaging have also been proposed [I-D.templin-6man-dhcpv6-ndopt][I-D.naveen-slaac-prefix-management].

Other, non-router, mechanisms may exist, such as proprietary IPAMs, [I-D.ietf-anima-prefix-management] and [I-D.li-opsawg-address-pool-management-arch]. Requirements for extending an IPv6 /64 Prefix from a Third Generation Partnership Project (3GPP) Mobile Interface to a LAN Link are discussed in [RFC7278].

10. IANA Considerations

This document introduces no IANA considerations.

11. Security Considerations

Security considerations for IPv6 Neighbor Discovery [RFC4861] and any applicable PD mechanisms apply to this document. Nodes that manage their delegated prefixes such that MLD/DAD procedures are not needed on the upstream interface can avoid introducing multicast messaging congestion on the upstream link. Also, routers that delegate prefixes keep only a single neighbor cache entry for each prefix delegation recipient, meaning that the router's neighbor cache cannot be subject to address resolution-based resource exhaustion attacks.

For shared and individual prefixes, if the advertising router considers the prefix as on-link the IPv6 ND address resolution function will prevent unwanted IPv6 packets from reaching the node. For delegated prefixes and individual prefixes that are not considered on-link, the router delivers all packets that match the prefix to the node. In that case, the node may receive unwanted IPv6 packets via an upstream interface for which it has no matching configured address. The node then drops the packets and observes the ICMPv6 "Destination Unreachable - Address/Port unreachable" procedures discussed in [RFC4443].

The node may also receive IPv6 packets via an upstream interface that do not match any of the node's delegated prefixes. In that case, the node drops the packets and observes the ICMPv6 "Destination Unreachable - No route to destination" procedures discussed in [RFC4443]. Dropping the packets is necessary to avoid a reflection attack that would cause the node to forward packets received from an upstream interface via the same or a different upstream interface.

12. Acknowledgements

This work was motivated by discussions on the v6ops list. Mark Smith, Ricardo Pelaez-Negro, Edwin Cordeiro, Fred Baker, Ron Bonica, Naveen Lakshman, Ole Troan, Bob Hinden, Brian Carpenter, Joel Halpern, Albert Manfredi, Dusan Mudric, Paul Marks, Joe Touch, Alex

Petrescu, Lorenzo Colitti, Tatuya Jinmei and Naveen Kottapalli provided useful comments that have greatly improved the document.

This work is aligned with the NASA Safe Autonomous Systems Operation (SASO) program under NASA contract number NNA16BD84C.

This work is aligned with the FAA as per the SE2025 contract number DTFAWA-15-D-00030.

This work is aligned with the Boeing Commercial Airplanes (BCA) Internet of Things (IoT) and autonomy programs.

This work is aligned with the Boeing Information Technology (BIT) MobileNet program.

13. References

13.1. Normative References

- [RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <<https://www.rfc-editor.org/info/rfc3810>>.
- [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>>.
- [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>>.
- [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>>.

- [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>>.

13.2. Informative References

- [I-D.ietf-anima-prefix-management]
Jiang, S., Du, Z., Carpenter, B., and Q. Sun, "Autonomic IPv6 Edge Prefix Management in Large-scale Networks", draft-ietf-anima-prefix-management-07 (work in progress), December 2017.
- [I-D.li-opsawg-address-pool-management-arch]
Li, C., Xie, C., Kumar, R., Fioccola, G., Xu, W., LIU, W., Ma, D., and J. Bi, "Coordinated Address Space Management architecture", draft-li-opsawg-address-pool-management-arch-01 (work in progress), July 2018.
- [I-D.naveen-slaac-prefix-management]
Kottapalli, N., "IPv6 Stateless Prefix Management", draft-naveen-slaac-prefix-management-00 (work in progress), November 2018.
- [I-D.templin-6man-dhcpv6-ndopt]
Templin, F., "A Unified Stateful/Stateless Configuration Service for IPv6", draft-templin-6man-dhcpv6-ndopt-10 (work in progress), June 2020.
- [I-D.templin-6man-rio-redirect]
Templin, F. and j. woodyatt, "Route Information Options in IPv6 Neighbor Discovery", draft-templin-6man-rio-redirect-08 (work in progress), June 2019.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC7084] Singh, H., Beebee, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, DOI 10.17487/RFC7084, November 2013, <<https://www.rfc-editor.org/info/rfc7084>>.

- [RFC7278] Byrne, C., Drown, D., and A. Vizdal, "Extending an IPv6 /64 Prefix from a Third Generation Partnership Project (3GPP) Mobile Interface to a LAN Link", RFC 7278, DOI 10.17487/RFC7278, June 2014, <<https://www.rfc-editor.org/info/rfc7278>>.
- [RFC7934] Colitti, L., Cerf, V., Cheshire, S., and D. Schinazi, "Host Address Availability Recommendations", BCP 204, RFC 7934, DOI 10.17487/RFC7934, July 2016, <<https://www.rfc-editor.org/info/rfc7934>>.
- [RFC8273] Brzozowski, J. and G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273, DOI 10.17487/RFC8273, December 2017, <<https://www.rfc-editor.org/info/rfc8273>>.
- [RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <<https://www.rfc-editor.org/info/rfc8504>>.

Appendix A. Change Log

<< RFC Editor - remove prior to publication >>

Changes from -25 to -26:

- o Version and reference update

Changes from -24 to -25:

- o Version and reference update

Changes from -23 to -24:

- o Version and reference update

Changes from -22 to -23:

- o Changed DHCPv6 references to RFC8415. Deprecate RFC3315 and RFC3633.
- o New text on assignment of addresses and prefixes on the upstream interface.

Changes from -21 to -22:

- o Changes to address list comments contributed by Lorenzo Colitti, Tatuya Jinmei, Brian Carpenter and Fred Baker.

- o Deleted section on ICMPv6 - now defer to normative reference [RFC4443].
- o Discuss 'DupAddrDetectTransmits' variable implications under MLD/DAD considerations.

Changes from -20 to -21:

- o Re-worked classic routing model section
- o Included multi-addressing case where addresses may be assigned to applications
- o Removed strong/weak end system discussions

Changes from -19 to -20:

- o figure 1 updates to show Server as being somewhere in the network
- o Introductory material to show relation to other RFCs on multi-addressing

Changes from -18 to -19:

- o added new section on Prefix Delegation Services

Changes from -17 to -18:

- o re-worked discussion on the prefix delegation service in Section 1
- o updated figures in Section 1

Changes from -16 to -17:

- o added supporting text in the introduction to discuss the Delegating Router's relationship with the Requesting Router and with supporting infrastructure in the operator's network
- o updated figures in introduction to include representation of operator's network
- o added new section on Address Autoconfiguration Considerations

Author's Address

Fred L. Templin (editor)
Boeing Research & Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org