Thread Border Router Best Practices

March 2018

This Thread Technical white paper is provided for reference purposes only. The full technical specification is available publicly. To join gain access, please follow this link: https://www.threadgroup.org/ThreadSpec.

If there are questions or comments on these technical papers, please send them to help@threadgroup.org.

This document and the information contained herein is provided on an “AS IS” basis and THE THREAD GROUP DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO (A) ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OF THIRD PARTIES (INCLUDING WITHOUT LIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING PATENT, COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NONINFRINGEMENT.

IN NO EVENT WILL THE THREAD GROUP BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OF BUSINESS, OR FOR ANY OTHER DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR IN TORT, IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

Table of Contents

Terminology

TERM	DEFINITION
CPE	Customer Premises Equipment (Typically an ISP-provided Modem or Modem + Router).
DNS64	A technology and technique described in [RFC6147] whereby IPv6-only clients are provided, when used in conjunction with NAT64, with synthesized DNS AAAA records by translating actual IPv4-only A records.
ISP	Internet Service Provider
IPv6 Transition	Services needed to migrate the Internet from IPv4 to IPv6.
IPv6 Translation and Tunneling	Mechanisms for interconnecting IPv6 networks via IPv4 networks.
NAT	Network Address Translation
NAT64	An IPv6 Transition technology that enables communications between IPv6-only nodes and IPv4-only nodes by using a form of network address translation (NAT) among a single IPv4 address and IPv4-embedded IPv6 addresses. See [RFC6146].
ND	Neighbor Discovery

Introduction

In a Thread network, the Border Router is an important piece of infrastructure. It facilitates communication of Internet Protocol (IP) packets into and out of the Thread network. In most Thread networks the Border Router will be responsible for providing full Internet communication to Thread devices.

It is highly desirable that Thread device vendors can create products that can be easily commissioned onto a Thread network, after which those devices can rely on standard networking functions for IP communications – be it with other networks in the home, or servers hosted on the Internet. All of this should be done in a way that is broadly interoperable and common for Thread devices and Border Routers from various vendors.

Thread Group aims for a functionality and end-user experience similar to that of Wi-Fi access points and home routers, whereby Wi-Fi devices can be relied upon to automatically obtain their IP networking configuration and are automatically provided a route to the wider Internet, as well as IP connectivity to other types of networks in the home such as Ethernet.

End-to-end IPv6 routability is not yet available in every home (although Ipv6 availability has grown significantly in the last years and continues to grow). This white paper will go into some detail on the selection and implementation of suitable IPv6 Translation and Tunneling technologies, such as NAT64 and DNS64, to achieve an optimal user experience even when an ISP offers IPv4 only in the user's home network.

This document is not a replacement for any part of the Thread specifications; it is designed to act as a guide suggesting best practices and optimal technology selection for vendors that are looking to create a (generic) Thread Border Router. Any requirements language (capitalized "MUST", "SHOULD", etc.) only holds for a device that claims full compliance to these best practices.

Use Cases and Requirements

Goals and Non-Goals of This Document

The goals of this white paper include:

- Detailing how to design a generic Thread Border Router that facilitates
  - Site connectivity: Thread Routers and End-devices achieve end-to-end IP communication with other networks in the home (Wi-Fi, Ethernet)
  - Global connectivity: Thread Routers and End-devices achieve end-to-end IP communication with other networks outside the home (WAN, via home Internet connections such as DSL, DOCSIS, GPON or 3G/4G/5G cellular)
- Referencing the Thread specification and RFCs defined by the IETF, whenever possible to provide guidance for implementers
- Focusing on the most common residential networking scenarios and providing solutions for those
- Listing requirements and services to enable a generic Thread Border Router to support multiple vendor ecosystems and multiple IP-based application-layer technologies.

Non-goals of this document include:

- Defining which application-layer technologies a Thread Border Router should implement
- Proposing changes to (the behavior of) any non-Thread equipment such as upstream CPE routers, Wi-Fi access points, or cable modems.
- Provide details for gateway functions to translate communications to or from non-IP networks.

Requirements for In-Scope Features

The following requirements were used as the basis to draft this best-practices white paper. The term “recommendations” below refers to the recommended best practices as listed in this paper.

1. **Standards Based**
   1.1. Recommendations SHOULD be based on existing standards
   1.2. Recommendations SHOULD reference Thread specification and IETF RFCs when possible
   1.3. Recommendations SHOULD assume CPE will only implement protocols that are in wide deployment (>80% of homes) as of 2016
2. **Global Connectivity**

2.1. Recommendations MUST allow Thread hosts to connect to global IPv4 networks

2.2. Recommendations MUST allow Thread hosts to receive responses from global IPv4 networks within a realistic time window

2.3. Recommendations MUST allow Thread hosts to connect to global IPv6 networks

2.4. Recommendations MUST allow Thread hosts to receive responses from global IPv6 networks within a realistic time window

2.5. Recommendation SHOULD allow Thread hosts to receive unsolicited IPv6 messages from nodes in global IPv6 networks

2.6. Thread hosts MAY be required to register with the Border Router in order to receive unsolicited messages from off-mesh sources

3. Site Connectivity

3.1. Recommendations MUST allow Thread hosts to connect to on-site IPv4 hosts

3.2. Recommendations MUST allow Thread hosts to connect to on-site IPv6 hosts

3.3. Recommendations SHOULD allow on-site IPv6 hosts, such as laptops or phones, to connect to Thread hosts (and run discovery protocols to discover Thread hosts)

4. Security and Firewall

4.1. Recommendations MUST ensure that only ingress traffic enters a Thread network that Thread hosts have expressed interest in

Proposed Solution Summary

A summary of the proposed solution, in terms of the recommended best practices and protocols for Thread Border Routers, is presented below. More details can be found in the next section “Best Practices” on page 14.

Protocols Summary

Requirement	Solution (Protocols)	RECOMMENDED	OPTIONAL	Not Recommended
IPv4 Global, IPv4 Site connectivity	NAT64, DNS64, stateful firewall, Self-election via Thread Network Data	✓		
IPv6 Global, IPv6 Site connectivity	DHCPv6-PD, stateful firewall, Self-election via Thread Network Data	✓ Except when only /64 prefix available		
Thread Commissioning / Discovery	MeshCoP Border Agent, DNS-SD (for Commissioning)	✓		
Security for Site Address Referral	PCP	✓		
Private Networking (Global)	VPN		✓	
IPv6 Global via Cellular Hotspot	ND Proxy			✓ Allowed when only /64 prefix available
IPv6 Site connectivity	ND Proxy			✓ High overhead for Thread 1.1 hosts
Shared IPv6 Prefixes across Site: ULA	HOMENET: HNCP + Babel		✓	
Service Discovery	CoRE Resource Directory		✓	

Assumed Customer Environment Summary

Common services the customer premise equipment (CPE) and Internet Service Provider (ISP) provide via standard protocols such as DHCP, DHCPv6-PD, and PPPoE are:

- IPv4: single global IPv4 address for which the CPE will run NAT (NAT44) to a private local IPv4 address range of administratively configured size.
- IPv6: global prefix delegation of varying prefix sizes;
  - Most commonly, a prefix size which allows subnetting (mostly /48, /56, or /60); this allows a /64 global subnet to be assigned exclusively to a Thread network by a Thread Border Router.
  - More rarely, only a /64 size prefix is delegated to the CPE which does not allow further subnetting within the home; for such cases ND-Proxy may be used although this can create excessive traffic on the Thread network. In this case either inbound traffic flows are restricted or a stateful firewall is needed.

Device and Service Discovery Summary

Thread devices arriving into home networks may perform discovery as follows:

- Local devices with services that match theirs to provide application level functions (a thermostat discovering other temperature sensors for example).
  - This discovery may be on the local Thread network only
  - This discovery may include a full home network of different subnets (Thread, Wi-Fi, Ethernet, or others)
- Cloud services provided by the device owner’s cloud vendor of choice. A service may be provisioned into the device upon installation (e.g. during Thread commissioning), or a service may be discovered by the device using DNS.
- Device-specific maintenance or management services. A device may have a manufacturer’s service for software updates, maintenance reporting or check-in so e.g. a device automatically performs a lookup for *acmeservices.com* using DNS. The Border Router should therefore support DNS service for Thread devices.

Border Router Device Categories

A key difference between Thread Border Routers and devices such as CPE Routers or Wi-Fi Access Points is that Thread Border Router services are both more limited and provided more autonomously to devices on the Thread network. Furthermore, when multiple Border Routers are part of the same Thread network, this allows supplemental redundancy paths for packet forwarding to and from the Thread network.

This allows Thread Border Router functions to be enabled also by devices not purely dedicated to network infrastructure: such as an IoT appliance that has both a Thread interface as well as one or more supplemental interfaces external to the Thread network like Wi-Fi or Ethernet. Such appliances may be implemented by more constrained devices, such as ones running an RTOS instead of a full-featured operating systems optimized for network infrastructure (e.g. Linux, OpenWRT) which is common in CPEs and Wi-Fi Access Points.

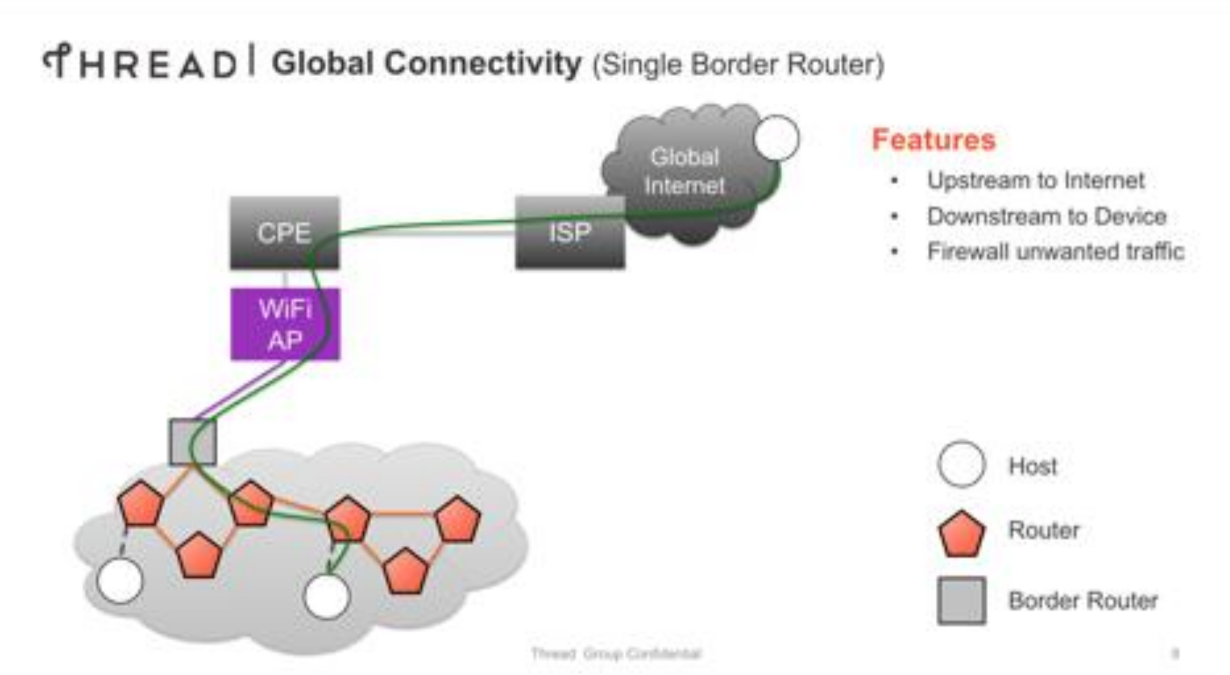
Connectivity Summary

This section outlines the communication flows that this document recommends a Thread Border Router to support. Recommended features are:

- Global Connectivity (IPv6 uplink, IPv4 uplink, single Border Router, multiple Border Routers)
- Site Connectivity (IPv6 uplink, IPv4 uplink, connectivity between Thread Hosts and Wi-Fi/Ethernet hosts)

Global Connectivity

Global connectivity is a core use case a Thread Border Router is expected to solve. Given a premises where an Internet connection, Local Area Network (e.g. Wi-Fi or Ethernet), and a Thread Network with at least one Border Router are deployed, the user expects their Thread hosts to be able to connect to global cloud services. This expectation is true both when the Customer Premise Equipment (CPE) supports IPv6 and IPv4, or IPv4 only. The use case is depicted below for a single Border Router case.



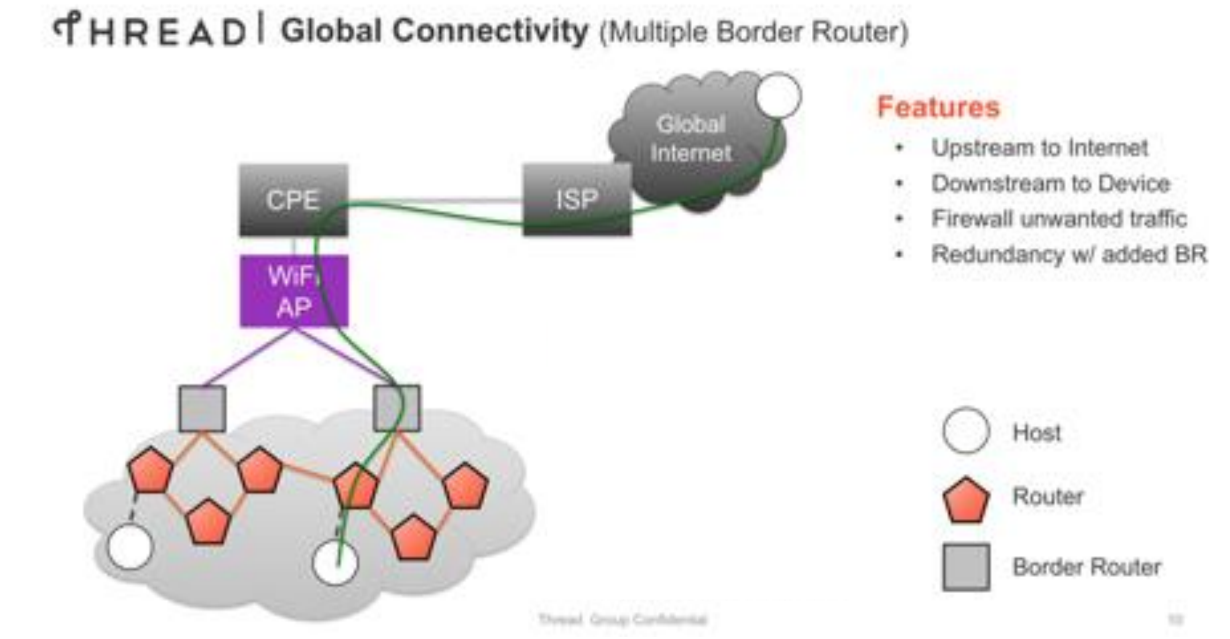
THREAD | Global Connectivity (Single Border Router)

Features

- Upstream to Internet
- Downstream to Device
- Firewall unwanted traffic

Thread Group Confidential

Thread networks aim to have no single point of failure. As such, a Thread network with multiple Border Routers must be able to meet the same user expectation for global connectivity as the single Border Router case. Additionally, a Thread network with multiple Border Routers should handle dynamic fail-over when one of the Border Routers fails or goes offline. The diagrams below depict multiple Border Routers use cases.

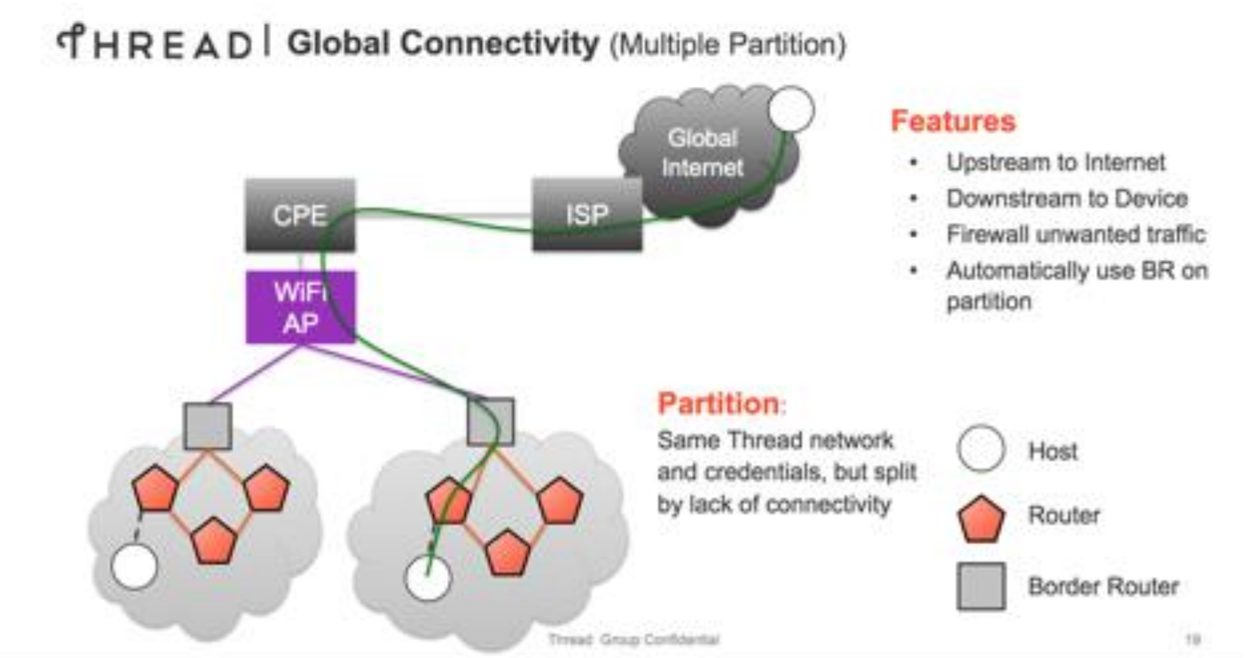


THREAD | Global Connectivity (Multiple Border Router)

Features

- Upstream to Internet
- Downstream to Device
- Firewall unwanted traffic
- Redundancy w/ added BR

Thread Group Confidential



THREAD | **Global Connectivity** (Multiple Partition)

Features

- Upstream to Internet
- Downstream to Device
- Firewall unwanted traffic
- Automatically use BR on partition

Partition:
Same Thread network and credentials, but split by lack of connectivity

Thread Group Confidential 19

When there are multiple partitions that each have a Border Router, connectivity to and from the Internet continues to operate through those Border Routers. However, device to device connectivity *between* the partitions does not work because mesh-local IPv6 addresses or global IPv6 addresses assigned to the mesh will not be routed between the Border Routers.

When there are multiple partitions and one of them does not have a Border Router, the devices in that Partition will no longer have IP connectivity outside the Thread network until either a Border Router is introduced or the Partition merges with another Partition that does have a Border Router.

Site Connectivity

The discussion on Global Connectivity above also applies to site connectivity. To communicate with IPv6 servers on the local site network, e.g. a home Wi-Fi link, a client on a Thread network requires its Border Router to acquire from the site's router/CPE, by standard methods, a delegation for an IPv6 subnet prefix for Global address or Unique Local Address (ULA), and to forward IPv6 packets between Thread devices and the site IPv6 network accordingly.

Security and Firewalling

Thread Networks have limited bandwidth. As such, a Thread Border Router must protect the Thread Network from excessive messages from the LAN using some basic firewall rules and rate limiting. Also, Border Routers should contribute an extra layer of defense against DoS attacks from malicious hosts on the Internet and even malicious hosts on-site. The latter may occur when IP-enabled devices on-site become compromised via malicious code.

Protocol Applicability Summary

Below table lists the protocols that are recommended to be supported by the Border Router, to provide support for the respective use cases listed in the columns.

Use case → **Solution** ↓	Global IPv6	Global IPv4	Site IPv6	Site IPv4	Firewall
NAT64 / DNS64 / stateful-firewall		✓		✓	✓
NAT66 / stateful-firewall	✓		✓ No address referral		
ND Proxy (LAN only, not Thread)	✓		✓		
DHCPv6-PD	✓ Except when only /64 available		✓ Except when only /64 available		
PCP					✓
HNCP + Babel	✓ Only when CPE upgraded to run babel		✓		

Best Practices

The best practices for a Thread Border Router are described in more detail in this section, looking at connectivity to IPv4 destination, connectivity to IPv4 destinations, security for Thread Management Framework (TMF) communications, and IP multicast support.

Connectivity to IPv4 Destinations

Typical home network infrastructure, e.g. Wi-Fi optionally bridged to Ethernet, provides a sub-IP layer capable of carrying both IPv4 and IPv6. The full transition to an IPv6-only public Internet is still many years away. Over the foreseeable future, the need for Thread Border Routers to provide connectivity for Thread hosts to IPv4-only destinations on the public Internet can be expected to persist. Accordingly, this section discusses the requirements for Thread Border Routers to support IPv4 connectivity.

Border Router as IPv4 LAN Host and DHCPv4 Client

Thread Border Routers MUST fulfil the role of a fully functional IPv4 host on non-Thread home local-area network links that they are connected to. In particular, they use the *Dynamic Host Configuration Protocol for IPv4 (DHCP)* [RFC2131] to acquire the following host configuration parameters from any suitable offerer:

- IPv4 interface address
- IPv4 default router address
- IPv4 subnet mask
- DNS server addresses
- DNS domain search list

Usage of these parameters by Thread Border Routers is described in the following subsections. Care is recommended when designing how the DHCP client selects suitable offers when multiple DHCP servers are available, but no specific method is described here.

NAT64: Thread Client Connectivity to IPv4 Servers

Thread Border Routers MUST use *Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers* [RFC6146] to provide connectivity to Thread hosts for IPv4-only destinations.

The NAT64 function MUST use an IPv4 address acquired via DHCP in its IPv4 address pool. When the DHCP lease for any IPv4 address in its pool expires without successful renewal or reconfiguration, the NAT64 states associated with that IPv4 address MUST be flushed.

Thread Border Routers MUST advertise the IPv6 prefix for their NAT64 function in Thread network data as a Thread External Route.

The private IPv6 prefix of the NAT64 function MUST be the well-known NAT64 prefix defined in *IPv6 Addressing of IPv4/IPv6 Translators* [RFC6052].

DNS64: Thread Client Locating IPv4 Servers

Thread Border Routers MAY advertise support for *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* [RFC3315] to Thread hosts in the Thread network data, which is an optional featured in the Thread specification [Thread 1.1.1].

If a Thread Border Router operates as a DHCPv6 agent, then it SHOULD also provide *DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers* [RFC6147] to translate queries and answers for AAAA records from Thread hosts to and from synthesized addresses that map the A record in the public DNS to the corresponding IPv6 address in the NAT64 prefix. This function also translates queries and answers for PTR records in the IP6.ARPA domain to queries for the corresponding IN-ADDR.ARPA domain for IPv4 addresses.

If provided, the DNS64 service MUST only be available at a mesh-local IPv6 address, and this address – along with any relevant DNS domain search list parameters obtained via DHCPv4 –SHOULD be offered to Thread hosts via DHCPv6.

Thread hosts may also assign global addresses via SLAAC. In this case the hosts would acquire DNS service information from the network data and the Border Router implements NAT64 and DNS64 for the devices.

To provide adequate support to Thread hosts for browsing and locating public IPv4 services using *DNS-Based Service Discovery* [RFC6763], the DNS64 function SHOULD support DNS Long-Lived Queries [I-D.sekar-dns-llq].

Thread Servers and IPv4 Clients

To provide adequate support to Thread hosts for *Unilateral Self-Address Fixing (UNSAF) Across Network Address Translation* [RFC3424], Thread Border Routers MUST provide a *Port Control Protocol (PCP)* [RFC6887] service.

If the Thread Border Router has a DHCPv6 agent, then the mesh-local address of the Thread interface on the Border Router MUST be offered to clients with DHCPv6 using *DHCP Options for the Port Control Protocol (PCP)* [RFC7291]. Otherwise the PCP service MUST be available at the *Port Control Protocol (PCP) IPv6 Anycast Address* [RFC7723], i.e. 2001:1::1.

The PCP service MUST have support for *Port Control Protocol (PCP) Proxy Function* [RFC7648], and it SHOULD be capable of mapping ports in the NAT64 in response to MAP requests.

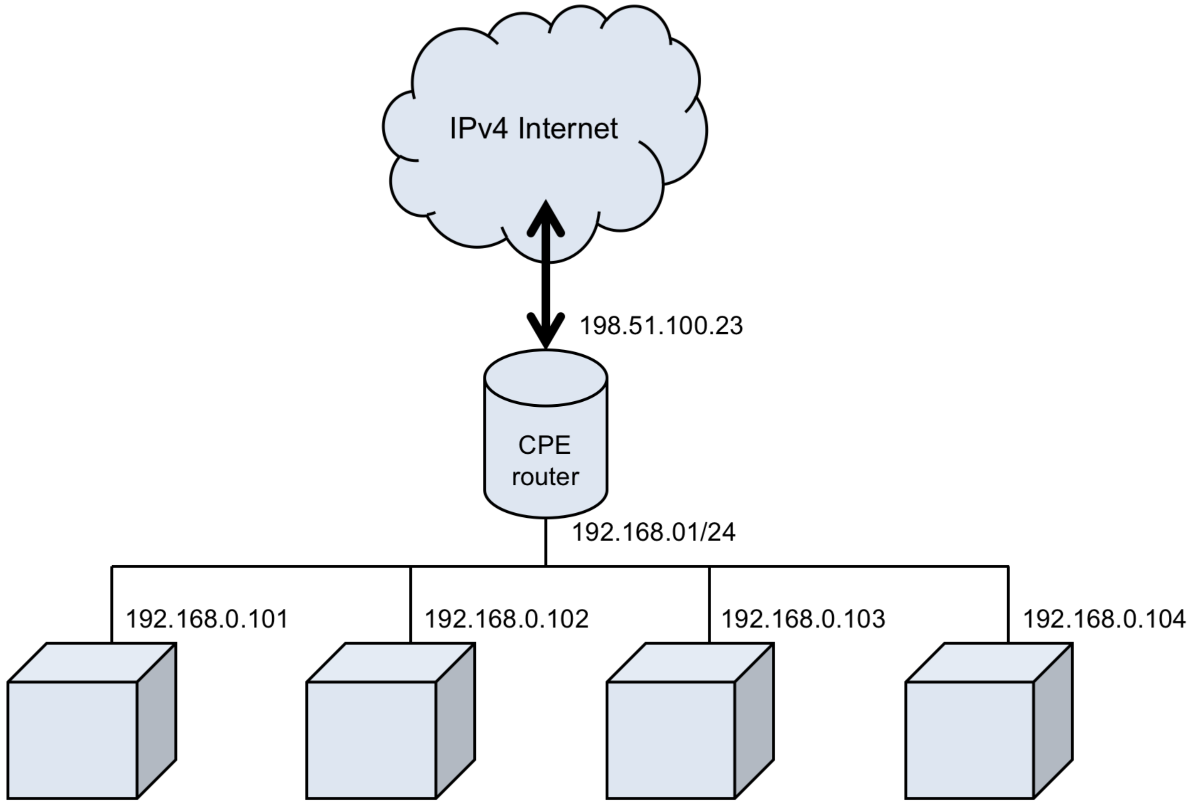
Any support provided to Thread server hosts for registering their services in any local or global service discovery systems is OPTIONAL.

Connectivity to IPv6 Destinations

Thread is a mesh network capable of providing IPv6 service to applications in nodes at both A) publicly routed global IPv6 addresses, and B) unique local addresses (ULAs) constrained to private routing domains. This section describes the considerations for providing IPv6 communication services between Thread hosts and public IPv6 Internet hosts, and between Thread hosts and private non-Thread network hosts.

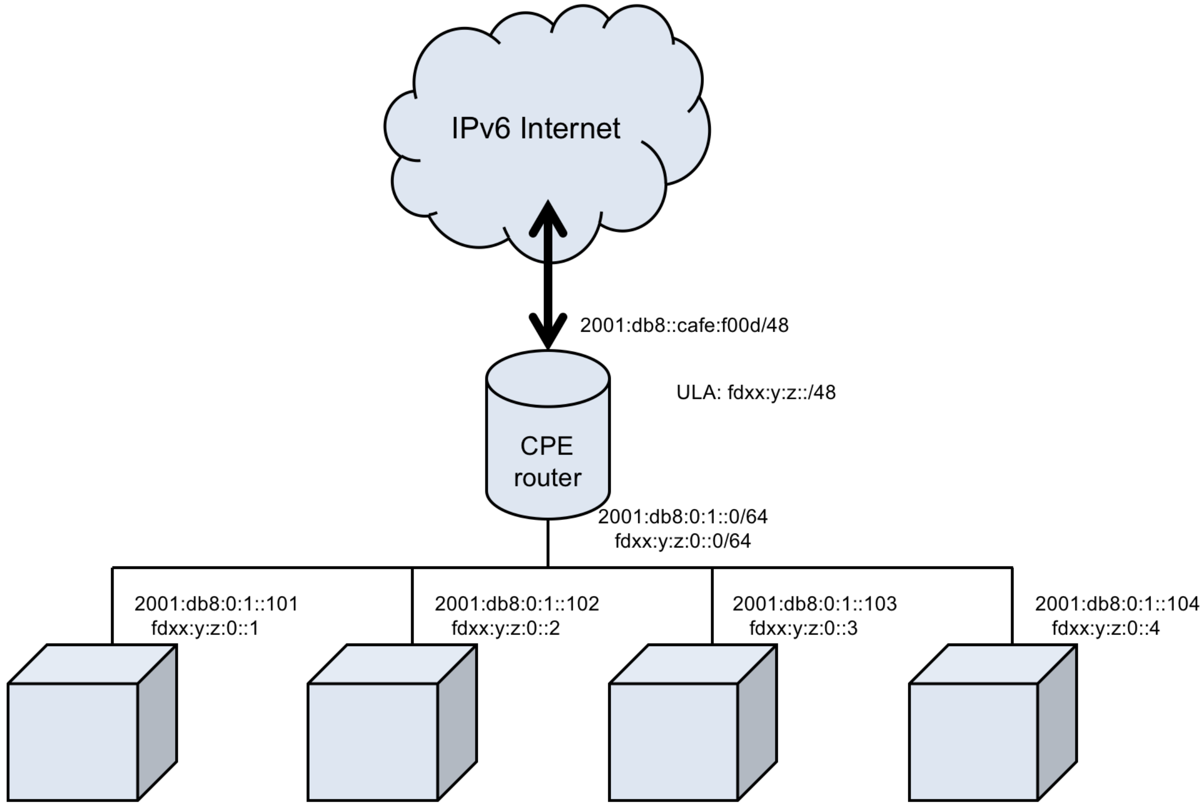
Routing Differences for IPv6 Connectivity Compared to IPv4

The following diagram depicts the typical way an IPv4 subnet is routed to the public Internet.



In this diagram, an ISP assigns a single global scope IPv4 address to the site (198.51.100.23), and network address translation (NAT) is used to share it with a subnet numbered with private scope addresses (192.168/24). Typical variations include ISPs that assign to each site a single private scope address from the Shared Address Space [RFC6598] and use a Carrier Grade Network Address Translator (CGN) to share a pool of public addresses among multiple subscribers. Only in very rare cases are subnets ever numbered with public global scope IPv4 addresses.

In contrast, the following diagram depicts a simplification of the typical way an IPv6 subnet is routed to the public Internet.



In this diagram, an ISP assigns a global scope IPv6 routing prefix to the site (2001:db8::/48) and a global scope address containing that prefix (2001:db8::cafe:f00d) to its wide-area link interface. The local subnet has a global scope subnet prefix (2001:db8:0:1/64) chosen by the CPE router.

Additionally, the CPE router also self-assigns a routing prefix to the site (fd*xx*:*y*:*z*::/48) with *Unique Local IPv6 Addresses (ULA)* [RFC4193]. It similarly chooses a subnet prefix with that routing prefix (fd*xx*:*y*:*z*:1::/64) for its local area network link.

Because there is no shortage of IPv6 addresses, network address translation (NAT) is not operationally required for address amplification (re-use) purposes as is typical with IPv4, and so third-party address referral is a feature of IPv6 not available to IPv4 applications. Third-party address referral describes the scenario of “host A sends the address of host B to host C,” which is significantly complicated by the presence of network address translators between hosts.

Additionally, the IPv6 subnet model differs from the IPv4 model in one important respect not shown in the above diagram: the distinction between an “on-link” prefix, which is used for routing purposes, and a “subnet” prefix, which is used for address configuration purposes. With IPv6 networking, every link has one or more “on-link” prefixes (always including the link-local prefix). Multiple distinct subnet prefixes may be assigned to the same link. Also, a subnet prefix can be divided into longer sub-prefixes that can then be assigned to (delegated to) multiple subordinate links. The 6LoWPAN [RFC6282] standard that Thread uses, in addition allows multiple links to share the same subnet prefix.

The distinctions between IPv4 and IPv6/6LoWPAN subnet models make the considerations for forwarding and routing between IPv6 links important to understand. Any particular Thread network 1) may have its own subnet apart from any non-Thread networks to which a Border Router is also connected, and/or 2) may share a subnet with other non-Thread networks through a Border Router. The following subsections describe each of these possibilities in detail.

IPv6 Routing and Forwarding

In the IPv4-only case, the prototypical home local-area network is a single IP link, usually Wi-Fi and sometimes also bridged with IEEE 802.1 to Ethernet, numbered with IPv4 addresses allocated locally by *Address Allocation for Private Internets* [RFC1918], with the public Internet made reachable via Network Address Translation (NAT) functions, see [RFC4787], [RFC5382] and [RFC5508], either in the CPE router or in the carrier network (or both).

Some home local-area networks consist of a cascade of NAT routers in series starting at the top with a CPE router, and with one or more interior links dependent for upstream IPv4 reachability on the NAT functions of their first-hop interior router.

In all cases with IPv4, unsolicited flows from upstream links to downstream links are blocked by the stateful process in the NAT router.

When the infrastructure supports IPv6, the typical home local-area network is still a single link, usually Wi-Fi and sometimes bridged with IEEE 802.1 to Ethernet, numbered with IPv4 in the manner described above, but also served with an IPv6 router according to *Basic Requirements for IPv6 Customer Edge Routers* [RFC7084]. For IPv6 connectivity, no NAT functions are required, although *Simple Security in IPv6 Gateway CPE* [RFC6092] solutions are sometimes deployed to block unsolicited flows from upstream links to downstream links in a comparable way as IPv4 NAT does.

In the IPv6 case, non-Thread hosts on each link of a home local-area network may communicate directly with one another using link-local scope IPv6 addresses, without facilitation or knowledge of any IPv6-capable router. However, communication between any two non-Thread hosts on different links of a home local-area network is generally made possible only when both hosts are addressed with global-scope addresses and a forwarding path between their respective links is operational via one or more interior routers.

How Thread Border Routers participate in the process of delegating global IPv6 routing prefixes from CPE routers and subsequently allocating and assigning subnet prefixes to home local-area network links is the subject of the following subsections.

IPv6 Prefix Allocation and Delegation

According to the requirements described in *Basic Requirements for IPv6 Customer Edge Routers* [RFC7084], links in home local-area networks have zero or more globally-unique IPv6 address routing prefixes delegated by the service provider to the CPE router. They also autonomously generate and persistently store a single unique-local IPv6 address prefix using *Unique Local IPv6 Addresses (ULA)* [RFC4193].

This subsection describes the available methods for automatically allocating subnet prefixes to home local-area network links.

Prefix Delegation with Dynamic Host Configuration Protocol (DHCPv6-PD)

Thread Border Routers SHOULD connect Thread networks with home local-area networks by participating in the available routing prefix allocation and delegation methods used for providing subnets with IPv6 addresses. The typical way that IPv6 subnet prefix allocation works today is that interior routers use clients that implement *Prefix Delegation with Dynamic Host Configuration Protocol (DHCPv6-PD)* [RFC3633]. They request a prefix of their chosen length, usually less than 64 bits, and the server allocates a prefix of that length or longer, and inserts a route in its routing table towards the requester for the newly delegated prefix, and then responds with a lease that includes a valid lifetime. The client then renews the lease on the prefix as necessary.

Where there is a cascade of IPv6 interior routers downstream from the CPE router, routing prefixes delegated by service providers can be exhausted quickly, and DHCPv6-PD clients may not be offered a prefix when they request one. This is an inherent risk based on the way prefixes are delegated and the Thread Border Router cannot resolve this. Moreover, due to the typical lack of an interior dynamic routing protocol in home networks, paths from hosts on one subnet to hosts on another subnet always take a suboptimal route, passing through the lowest common router in the cascade that was used to delegate a prefix for each of the two subnets.

Prefix Allocation with Home Network Control Protocol (HNCP)

Home Network Control Protocol (HNCP) [RFC7788] is a distributed consensus protocol that is more efficient than a cascade of DHCPv6-PD services for the function of allocating subnet prefixes to each link that shares a shorter routing prefix delegated by a service provider. An important feature of this protocol is that it facilitates the resolution of a single automatically generated home local-area ULA routing prefix, that is used for numbering each subnet in the home with exactly one ULA subnet prefix.

Thread Border Routers MAY implement HNCP [RFC7788] to coordinate with peers on non-Thread interfaces, including peers that are other Thread Border Routers, on the allocation of home local-area ULA subnets. On home local-area networks served by CPE routers that use HNCP to allocate prefixes delegated by service providers, Border Routers then allocate globally-unique subnet prefixes to each Thread network for each service provider prefix.

Because Thread networks are not suitable for transit, Thread Border Routers MUST NOT advertise HNCP agents on their Thread interfaces. It is therefore sufficient to implement the more efficient stub variant of the HNCP service on the non-Thread interfaces of the Border Router.

Mixed HNCP and DHCPv6 Prefix Delegation

Because [RFC7084] is widely implemented in commercial residential gateways, but HNCP is not yet as widely deployed (at the time of this document publication), Thread Border Routers MAY offer both an HNCP agent and a DHCPv6 prefix delegation client on non-Thread interfaces.

Thread Border Routers which are offered a ULA prefix in DHCPv6-PD MUST NOT advertise any additional self-delegated ULA prefix with HNCP.

Routing Distinct IPv6 Subnets

Thread Border Routers SHOULD obtain IPv6 subnet prefixes for their Thread interfaces by one or more of the prefix allocation and route distribution methods described in the previous section. Each distinct IPv6 subnet prefix they obtain by one of these methods SHOULD be advertised in the Thread mesh network via Network Data.

A Thread network will always have its designated ULA that is used for mesh-local addressing. If a Thread Border Router wants to join or create a home-local subnet that is based on a ULA prefix then it MUST either obtain a ULA routing prefix of sufficient size to allocate a /64 ULA subnet prefix to its Thread interface, or self-delegate a ULA prefix of 48 bits length (/48). In either case, the ULA subnet prefixes MUST NOT be shared by the Neighbor Discovery Proxy. In this operating mode, Thread Border Routers MUST send IPv6 Neighbor Discovery Neighbor Advertisement messages on the non-Thread interface with the R flag set, to indicate that it is an IPv6 router.

Router Advertisements on Non-Thread Links

Thread Border Routers in this configuration MAY send Neighbor Discovery RA messages on their non-Thread interfaces. If a Thread Border Router sends RA messages on its non-Thread interfaces, then it SHOULD set Router Lifetime to zero and send a Route Information Option (RIO) containing the specific prefixes it has acquired to facilitate hosts that process them with selecting the optimal first-hop router for destinations on Thread networks.

Dynamic Routing with Babel

Where multiple Thread Border Routers connect to non-Thread networks where no other IPv6 routers are present, a dynamic routing protocol MAY facilitate transit over non-Thread networks between hosts on Thread networks.

Thread Border Routers MAY implement the HOMENET profile of the Babel dynamic routing service [I-D.ietf-homenet-babel-profile] on their non-Thread interfaces. Prefixes allocated either by DHCPv6-PD or by HNCP should be advertised via Babel on non-Thread networks. Because Thread networks are generally not useful for transit, it is sufficient to implement the more efficient Babel routing agent for stub networks in Thread Border Routers that have constrained resources.

IPv6 Thread Servers and Non-Thread Clients

Passively listening IPv6 services on Thread networks (or anywhere elsewhere in the local domain) MAY use the *Port Control Protocol (PCP)* [RFC6887] to obtain leases on IPv6 firewalls that are configured by default to deny all unsolicited inbound IP flows from exterior hosts. Zero, one or more of such firewalls may be located anywhere in the network, e.g. in subscriber gateways and/or elsewhere in the ISP network.

Thread Border Routers that implement IPv6 firewalls SHOULD provide the capability to map ports using the PCP service (see above), and the PCP Proxy function SHOULD provide a *PCP Relay* Service [RFC7648] for mapping requests to upstream PCP services at the *PCP Anycast Address* [RFC7723].

Alternatively, these services MAY rely on the facility described in Section 3.2.4 “IPsec and Internet Key Exchange (IKE)” of *Simple Security in IPv6 Gateway CPE* [RFC6092] that allows all unsolicited inbound IKE and IPsec flows as an exception to the usual “default deny” policy.

Security for Thread Management Framework Communications

The Thread Management Framework (TMF) as defined in [Thread 1.1.1] is the protocol that takes care of the correct operation of the Thread mesh network, using the CoAP

[RFC7252] protocol over UDP between Thread devices in the mesh – including Border Routers. These communications are within the mesh all protected by the mesh network’s link-layer encryption. However, when a Border Router connects the Thread IPv6 nodes to external IPv6 networks, an additional risk of attacks on TMF from external (e.g., compromised) IPv6 nodes emerges.

To mitigate such attack vectors, a Thread Border Router MUST NOT ingress i.e. forward into the mesh any UDP packets destined to port 61631, with the exception of such UDP packets when received from a Commissioner over the DTLS-secured commissioning session. This specific port is the Thread Network Management port (:MM, Section 10.13 in [Thread 1.1.1]) used by TMF. This rule also implies that any application layer protocols cannot re-use this UDP port for their messages, for communication across Thread Border Routers.

IP Multicast

As Thread networks are relatively low bandwidth compared to other networks typically present in the home, the Border Router by default should not forward all IP multicast traffic from these external networks into the Thread network. Any inbound forwarding that is done should be rate-limited to avoid flooding the Thread network. It is recommended that a Thread Border Router can be administratively configured to allow specific IP multicast ingress policies; with settings that certain IPv6 multicast groups, multicast protocols (e.g. CoAP discovery [RFC7252]) or scopes (e.g. the admin-local scope 4) are to be forwarded into the mesh.

Outbound forwarding of IPv6 multicast is relatively safe and can be done if the multicast scope is appropriate; specifically the site-local scope 5 or higher scopes can always be forwarded while the realm-local scope 3 [RFC7346] MUST NOT be forwarded across Border Routers. The admin-local scope 4 is again administratively configurable.

Appendix: Application Layer Specific Services

Thread networks are an IPv6-bearer medium to facilitate the routing and communication of one or more IPv6-based network applications. Consequently, conscious design choices must be made to ensure that generalized network facilities such as those listed earlier in this document are harmonized and work constructively with application layer-specific optional services that may be operating within or at the boundaries of a Thread network such as:

- Service Discovery and Application Layer Proxies
- Virtual Private Networks (VPNs)

Service Discovery and Application Layer Proxies

The Border Router can play an important role in service discovery and application layer proxy services.

Devices arriving into home networks are interested in discovering services on the local Thread network, on other subnets within the home, discovering and connecting to cloud services, or to vendor-specific device maintenance or management services. Devices discovering services on the Internet generally use standard DNS services, which are not further discussed here.

Service discovery queries for local IoT devices and their services are typically not using DNS but IP multicast, where any device offering the requested service then unicasts a response to the requester. CoAP discovery [RFC7252] is an example of such a protocol. As multicasts may be filtered by the Border Router (see section “IP Multicast” in this white paper), it is beneficial if a Border Router can offer a service to respond to service discovery requests from adjacent subnets on behalf of a registered device on its own Thread network that offers those requested services. Even for use within its own Thread network, a Border Router could host a service such as a Resource Directory [I-D.ietf-core-resource-directory] to allow unicast discovery queries for services, instead of multicasts. For discovery of Sleepy End Devices (SEDs) and their services, such a Resource Directory can be an important function to enable discovery since these devices cannot respond for themselves while in a sleep cycle. Application layer protocols such as *dotdot* and *OCF* use CoAP discovery services and also specify how a Resource Directory can be used. Although this Resource Directory could be placed on any device within a Thread network, the Border Router is a very suitable location as it provides more memory, more processing capabilities, and can easily respond to requests from adjacent networks as well.

Application layer proxies can serve a similar function as they can respond to remote requests for (sensor) information or device status without directly waking a Sleepy End

Device. The sleepy device can be configured to report state changes event-based, or periodically, to an application-layer proxy which then services any requests for the "cached" information. Therefore, also these proxies could be conveniently located on a Thread Border Router.

Virtual Private Networks (VPNs)

Employing VPNs bring a number of benefits to Thread networks, their devices, and their application layers. The Thread Border Router is a suitable host for a VPN tunnel termination point.

IPv4 / IPv6 service and device deployment

Regardless of whether a Thread deployment site and/or a related cloud service are IPv6-enabled or not, a VPN can be established using IPv4 or IPv6, resulting in an IPv6 over IPv4 tunnel for IPv4-only deployments or an IPv6 over IPv6 tunnel for IPv6-enabled deployments, mitigating the issue of fragmented IPv6 deployment by ISPs.

Public Internet attack surface

Unless a device or ecosystem vendor chooses to do so, devices need not take on publicly-routable Internet addresses. Without such a publicly-routable Internet address, the attack surface for confidentiality, privacy, replay protection, etc. for an on-site device is dramatically narrowed from the open, public Internet to a tightly-controlled cloud service plus the other on-site devices that the Thread device is associated with.

Network Address Translation (NAT) and service provider traversal and keep-alive and asynchronous Internet ingress traffic

Most site customers obtain internet access through DSL or DOCSIS cable, provided by the local exchange carrier (LEC) telephone company or the cable television company, respectively. These companies typically provide the site customer with an access point and router, which typically include both firewall and network address translation NAT capabilities.

The firewall and NAT functionalities, in particular, prove troublesome for asynchronous, unsolicited ingress traffic from an Internet-based service into the site since typically, these technologies only allow, by default, asynchronous, unsolicited **egress** traffic **to** an Internet-based service. For some limited time after, typically extended by traffic *liveness*, the firewall and NAT will allow ingress solicited response traffic from the Internet into the site.

By initiating a VPN and by maintaining an always-fresh and *-live* associated tunnel, rather than or in contrast to an on-demand tunnel, independent of any traffic flowing over the tunnel, the VPN removes the burden of maintaining NAT liveness while also allowing, at the application layer, asynchronous, unsolicited ingress traffic from a cloud service to any arbitrary ecosystem device within the site. This is a particularly important use case for cloud service-based controllers, in which the on-site device is receiving an unsolicited request from the cloud rather than initiating it and making a request to the cloud.

API and interaction pervasiveness and consistency

With the cloud service and address virtualization provided by a VPN, device developers no longer need to treat access to other devices, mobile devices, or a cloud service as separate, discrete interaction models and APIs. Developers may simply identify a resource to make a request to – and without regard to its location, compose a message or command, address it, and send it.

This may also obviate the need for end devices to deal with DNS, saving that code space for features and other more value-added functionality while eliminating DNS as a common single-point-of-failure and target of attacks.

Path redundancy

VPN-based approaches have the benefit of being able to support multiple VPN tunnels between the site and the VPN termination point, providing redundant paths for packets to flow into and out of the site network. Because routing is not stateful, packets can be dynamically re-routed via an alternate path without disrupting application-level communication.

VPN Benefits to Cloud Service Providers

In addition to the benefits delivered to Thread networks, their devices, and their application layers, VPNs can also bring additional benefits to cloud service providers.

TCP connection scaling

Assuming an Internet of Things (IoT) based on the traditional web-technologies of TCP and HTTP, with market projections exceeding 10x or more of the current mobile device population, IoT devices pose daunting challenges to cloud service providers as they look to scale the (cloud-hosted) termination of TCP connections from such IoT devices.

However, if we assume that the scale of IoT is greater with more Thread-based constrained, low-power, and low-bandwidth devices and fewer mains-powered, more capable, high-bandwidth devices running and initiating VPNs, a VPN-based system architecture then provides a cloud service TCP connection scaling benefit. Cloud services TCP connection handling must scale with the number of VPN devices on the site rather than with the (larger) total number of a given ecosystem's IoT devices on the site.

Routing and Addressing

When dealing with supplemental Unique Local Addresses, whether used in conjunction with VPNs or not, care and consideration must be taken when dealing with routes for these addresses, ensuring that they are handled in preference to or with priority over default routes.

Consider nodes 6, B, D, i, and ii in the figure below, in which we assume that prefix delegation from a service provider (not depicted) is in effect with the IPv6 prefix 2001::/64.

Node B
This node is a Thread Border Router running two application-specific services, one of which is a VPN. It has the following addresses:

- **LLA:** FE80::0B%thread
- **MLA:** FC54::0B
- **ULA:** FD65::0B
- **GUA:** 2001::0B
- **IPv4:** 192.168.100.11

It may form either an IPv6-over-IPv4 or an IPv6-over-IPv6 tunnel with Node D.

Node D
This node is an application-specific VPN terminator / concentrator operating in a managed cloud on behalf of the application-specific service provider. It has the following addresses:

- **ULA:** FD65::0D
- **GUA:** 200D::0D
- **IPv4:** 10.251.50.13

where the ULA FD65::0D is a virtualized address afforded by any VPNs created with either an IPv6-over-IPv4 or an IPv6-over-IPv6 tunnel.

Node 6
This node is a Thread End Device running two application-specific services, one of which is bound to the application-specific supplemental ULA prefix FD65::/64. It has the following addresses:

- **LLA:** FE80::06%thread
- **MLA:** FC54::06
- **ULA:** FD65::06
- **GUA:** 2001::06

Node i
This node is an application-specific service endpoint running within the application service provider’s managed cloud infrastructure, reachable only via the VPN provided by link BD. It has the following address:

- **ULA:** FD65::11

Node ii
This node is an application-specific service endpoint running within the application service provider’s managed cloud infrastructure and is globally reachable, either by IPv6 or IPv4. It has the following addresses:

- **GUA:** 2012::12
- **IPv4:** 10.251.50.18

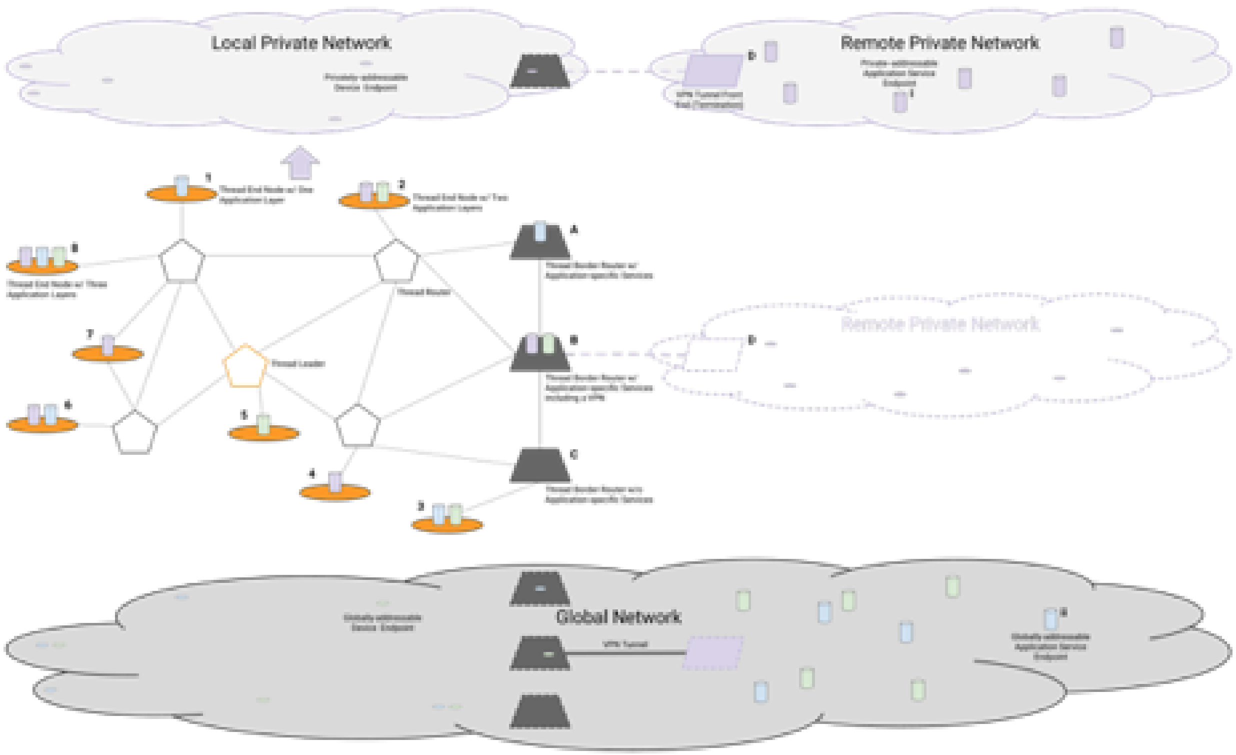

Local Private Network
Privately addressable Device Endpoint
Remote Private Network
Private addressable Application Service Endpoint
VPN Tunnel Point End (Termination)
1
Thread End Node w/ One Application Layer
2
Thread End Node w/ Two Application Layers
8
Thread End Node w/ Three Application Layers
A
Thread Border Router w/ Application-specific Services
Thread Router
7
Thread Leader
B
Thread Border Router w/ Application-specific Services including a VPN
Remote Private Network
6
5
4
C
Thread Border Router w/o Application-specific Services
3
Global Network
Globally addressable Device Endpoint
VPN Tunnel
Globally addressable Application Service Endpoint

Figure. Thread network with VPN-enabled application-specific services.

For communications traffic from Node 6 with the source address 2001::06 to Node ii with the destination address 2012::12, unsolicited traffic may egress the Thread network via any Border Router, A, B, or C. In theory, any solicited return traffic in the opposite direction may ingress the Thread network via any of the same Border Routers. However, in practice, some or all of these Border Routers may also be running stateful firewall policies which dictates that the solicited return traffic must ingress the same Border Router as the egress for the unsolicited traffic.

However, for communications traffic from Node 6 with the source address FD65::06 to Node i with the destination address FD65::11, traffic **must** egress the Thread network via the Border Router B, since it is the only Border Router with an off network route to FD65::/64. Likewise, any traffic in the opposite direction **must**, by definition and design, ingress the Thread network via Border Router B since it is the only node with an established VPN tunnel.

References

Thread Group, Thread 1.1.1 Specification, February 2017. Link

Thread Group, Thread Border Routers white paper, v2.0 July 2015. Link

Internet Engineering Task Force (IETF) standards documents:

RFC 1918 – Address Allocation for Private Internets
RFC 2131 – Dynamic Host Configuration Protocol
RFC 2993 – Architectural Implications of NAT
RFC 3424 – IAB Considerations for UNilateral Self-Address Fixing (UNSAF) …
RFC 3633 – IPv6 Prefix Options for DHCPv6
RFC 3879 – Deprecating Site-local Addresses
RFC 4193 – Unique Local IPv6 Unicast Addresses
RFC 4389 – Neighbor Discovery Proxies (ND Proxy)
RFC 6052 – IPv6 Addressing of IPv4/IPv6 Translators
RFC 6092 – Recommended Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service
RFC 6146 – Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers
RFC 6147 – DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers
RFC 6282 – Compression Format for IPv6 Datagrams over IEEE 802.15.4 Networks
RFC 6598 – IANA-Reserved IPv4 Prefix for Shared Address Space
RFC 6603 – Prefix Exclude Option for DHCPv6-based Prefix Delegation
RFC 6751 – Native IPv6 behind IPv4-to-IPv4 NAT Customer Premises Equipment
RFC 6763 – DNS-Based Service Discovery
RFC 6830 – The Locator/ID Separation Protocol (LISP)
RFC 6887 – Port Control Protocol (PCP)
RFC 6890 – Special-Purpose IP Address Registries
RFC 7084 – Basic Requirements for IPv6 Customer Edge Routers (CPE)
RFC 7252 –The Constrained Application Protocol (CoAP)
RFC 7346 – IPv6 Multicast Address Scopes
RFC 7526 – Deprecating the Anycast Prefix for 6to4 Relay Routers
RFC 7648 – Port Control Protocol (PCP) Proxy Function
RFC 7723 – Port Control Protocol (PCP) Anycast Addresses

RFC 7788 – Home Networking Control Protocol
IETF Working Group draft I-D.ietf-v6ops-ula-usage-considerations, March 2017
IETF Working Group draft I-D.ietf-homenet-babel-profile, February 2018
IETF Working Group draft I-D.ietf-core-resource-directory, March 2018
IETF draft I-D.sekar-dns-llq, August 2006.