Thread Network Fundamentals White Paper

September 2022

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.

Thread Network Fundaments White Paper

September 2022

Revision History

Revision	Date	Comments
1.0	November 29, 2014	Initial Release
2.0	July 13, 2015	Public Release
3.0	May 12, 2020	Updated Release
4.0	September 28, 2022	Updated Release for Accuracy

Table of Contents:

1. Introduction

The Internet of Things (IoT) aims to transform people's lives through smart homes and businesses. In the home, the goal is a network of connected appliances, lights, climate controls, security, and entertainment systems, all of which work together to make life more convenient and rewarding for consumers. In commercial buildings, IoT aims to improve the efficiency, functionality, automation, and safety of buildings such as offices, healthcare facilities, hotels, and schools. This white paper will discuss the details around each of the fundamental aspects of Thread to give users a deeper understanding of how a Thread network operates.

1.1. General Characteristics

The Thread Specification is an open standard for reliable, cost-effective, low-power, secure, wireless IPv6 communication. It is designed specifically for connected home and commercial applications where IP-based networking is desired and a variety of application layers can be used on the same network.

These are the general characteristics of Thread:

- Simple network installation, start-up and operation: The simple protocols for forming, joining, and maintaining Thread Networks allow systems to self-configure, dynamically optimize and heal.

- Secure: Devices can not join the Thread Network unless authorized and all network communication is encrypted and secure.

- Small and large networks: Home networks vary from several devices to hundreds of devices communicating seamlessly. Commercial networks can host thousands of devices at the same time. The Thread network layer is designed to optimize the network operation based on the expected use.

- Range: Typical devices in conjunction with mesh networking provide more than enough range to cover a normal home. Backbone Border Routers (BBRs) enable even larger IPv6 subnets composed of Thread devices, unifying multiple Thread networks into a single IPv6 subnet for the commercial market. Spread spectrum technology is used at the physical layer to provide good immunity to interference.

- No single point of failure: Thread Networks are auto-configuring and self-healing, so will continue to provide secure and reliable communication even if individual devices fail.
- Low power: Host devices can operate for several years on small batteries using suitable duty cycles.
- Built on open and proven standards: The Thread Specification uses well-defined standards from the IEEE and IETF as the foundation for a robust, modern protocol that doesn't seek to reinvent the wheel.
- Application-layer agnostic: Thread is a networking layer solution based on IPv6. Any low bandwidth application layer that can run over IPv6 can run over Thread, and multiple application layers can share the same network.

Figure 1 illustrates an overview of the Thread Specification.

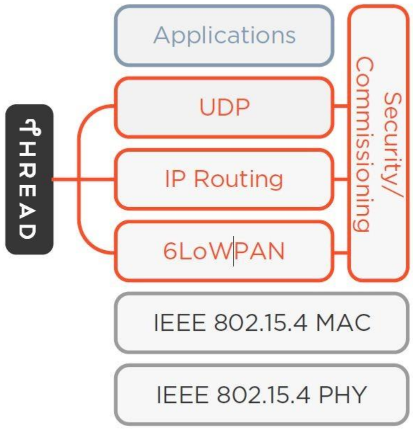

Figure 1. Overview of Thread Specification

1.2. IEEE 802.15.4 PHY/MAC

The Thread Specification uses the IEEE 802.15.4 [IEEE802154] PHY (Physical) and MAC (Media Access Control) layers operating at 250 kbps in the 2.4 GHz band for link layer communication. The IEEE 802.15.4-2006 and IEEE 802.15.4-2015 versions of the specification are used for Thread.

IEEE 802.15.4 is used as the foundation of the Thread Specification, providing reliable message transmission between individual Thread Devices at the link level. IEEE 802.15.4 provides a CSMA-CA (Carrier Sense Multiple Access – Collision Avoidance) mechanisms to allow multiple Thread Devices to utilize the shared 2.4GHz bandwidth by waiting for a clear channel before transmission. IEEE 802.15.4 uses link-layer acknowledgments and retries, enabling the reliable transmission of individual messages. Encryption, authentication and replay protection are also provided to guarantee secure communication. For low-power devices, routines are defined to extract messages buffered by always-on nodes. IEEE 802.15.4 is a widely deployed, well-tested protocol that the Thread Specification builds upon to provide reliable end-to-end communication.

1.3. No Single Point of Failure

In a system comprised of Thread Devices, none of these devices represent a single point of failure. While there are devices in the system that perform special functions, the Thread Network operates such that they can be replaced automatically without impacting the ongoing communication within the Network. For example, a Sleepy End Device (SED) requires a Parent for communication, so this Parent represents a single point of failure. However, the SED can and will select another Parent if its current Parent is unavailable, so this failure should not be visible to the user.

While the system is designed for no single point of failure, under certain topologies there will be individual devices that do not have backup capabilities. For example, in a system with a single Border Router, if the Border Router loses power, there is no means to switch to an alternative.

A single Router will assume the Leader role for certain functions in the Thread Network. This Leader is required to make decisions within the network. For example, the Leader assigns Router addresses and allows new Router requests. The Leader role is dynamically elected and if the Leader fails, another Router assumes the role. It is this autonomous operation that ensures there is no single point of failure.

2. Thread Device Types and Roles

There are two types of Thread Devices defined by the specification: the Full Thread Device (FTD), and the Minimal Thread Device (MTD). The MTD has the lowest requirements on device hardware (e.g. memory size) and power consumption, while the FTD is most versatile in the roles that it can play in a Thread Network. These roles are further detailed in the sections below.

2.1. Routing Full Thread Devices

Router

A Thread Router provides routing services to Thread Devices in the network. Routers also provide joining and security services for devices trying to join the network. Routers are not designed to sleep. Routers can downgrade their functionality and become REEDs (Router-eligible End Devices).

Leader

The Leader is an additional role of one Router in a Thread network. The Leader is an elected role of one Router, which takes certain decisions in the Thread network such as allowing REEDs to upgrade to Routers. If the Leader of a Thread Network fails, another Router will be dynamically selected to resume the role. All Routers have the required Thread Network Data to seamlessly assume this role.

2.2. Non-Routing Full Thread Devices

Router-Eligible End Device (REED)

REEDs have the capability to become Routers but due to the network topology or conditions are not acting as Routers. The Thread Network manages REEDs becoming Routers through the Leader, without user interaction.

Full End Device (FED)

FEDs are end devices similar to REEDs, however they do not have the capability to be a Router, so will never become a Routing Thread Device or Leader.

2.3. Non-Routing Minimal Thread Devices

Minimal End Device (MED)

Minimal End Devices (MEDs) communicate only through their Parent Router and cannot forward messages for other devices. An MED has its radio turned on, even when idle.

Sleepy End Device (SED)

Sleepy End Devices (SEDs) communicate only through their Parent Router and cannot forward messages for other devices. An SED has its radio turned off during idle periods and wakes periodically to communicate with its parent.

Synchronized Sleepy End Device (SSED)

Synchronized Sleepy End Devices (SSEDs) communicate only through their Parent Router and cannot forward messages for other devices. An SSED has its radio turned off during idle periods and wakes periodically to listen for messages from its parent at scheduled intervals.

2.4. Border Router

A Border Router is a role of a Thread Device that provides connectivity from the Thread Network to adjacent networks on other physical layers (for example, Wi-Fi or Ethernet). Border Routers provide services for devices within the Thread Network, including routing services for off-network operations. There may be several Border Routers in one Thread Network. Any FTD can provide Border Router services, even if the device is not acting as a Router in the Thread Network.

3. IPv6 Stack Fundamentals

3.1. Addressing

Devices in the Thread Network support the IPv6 addressing architecture specified in [RFC 4291]. The Thread Network automatically configures each device with certain IPv6 addresses by default as shown in Table 1. Each Thread Device can also configure additional addresses as required by the application.

Address Name	Description
Leader Anycast	The Leader of a Thread Network assigns itself an IPv6 'Anycast Address', which only belongs to the current Leader of the partition. This address is a defined well-known value in the Thread Specification, and any IPv6 packet sent to this address will be delivered to the Leader.
Mesh-Local EID	The Mesh-Local Endpoint Identifier (ML-EID) is a topology-independent IPv6 address that is routable on the Thread Network.
Mesh-Local RLOC	The Mesh-Local Routing Locator (ML-RLOC) is an IPv6 address that is dynamic and topology-dependent. It expresses the routing location of a device using the final 16 bits of the address as described in Figure 2.
Link-Local Interface Identifier	All devices have a link-local IPv6 address that is not routable. This address is used internally by MLE to configure Thread links with immediate neighbors.

Table 1. Thread autoconfigured IPv6 Addresses

The device that forms the network generates a /64 prefix that is used throughout the Thread Network. This prefix is known as the 'Mesh Local Prefix', and is used by every device in the network to generate two 'Mesh Local Addresses'. The Mesh Local Addresses are IPv6 ULAs [RFC 4193], and are used for communication within the Thread Network. For mesh establishment and link maintenance, every Thread Device configures a link-local Interface Identifier with a FE80::/64 prefix using its Extended Address as defined in section 6 of [RFC 4944].

The Thread Network may also contain one or more Border Routers that may provide additional GUA/ULA prefixes. Thread Devices may configure additional IPv6 addresses based on these prefixes using either SLAAC [RFC 7217] or DHCPv6 [RFC 3315] , depending on the Border Router configuration.

The devices also support appropriate multicast IPv6 addresses. By default, this includes link-local all nodes multicast, link-local all routers multicast, realm-local all nodes multicast and realm-local all routers multicast, depending on the device type. A Thread Device can also subscribe to additional multicast addresses as required by the application. Each device joining the Thread Network is assigned a 16-bit short address as specified in [IEEE802154]. In Thread, this is known as an RLOC16. For Routers, this address is assigned using the high bits in the address field with the lower bits set to 0, indicating a Router address. Children are then allocated a 16-bit short address using their Parent Router's high bits and the appropriate lower bits for their address. This allows any other device in the Thread Network to understand the Child's routing location simply by using the high bits of its address field.

Figure 2 illustrates the Thread short address.

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**1 0**	**1**	**2**	**3**	**4**	**5**
Router ID						**R**	**Child ID**								

Figure 2. Thread RLOC16

3.2. 6LoWPAN

Thread devices use 6LoWPAN, the standard as defined in [RFC 4944] and [RFC 6282], for transmission of IPv6 Packets over IEEE 802.15.4 networks.

Header compression is used within the Thread Network and devices transmitting messages compress the IPv6 header as much as possible to minimize the size of the transmitted packet.

The mesh header is supported for more efficient link-layer forwarding as discussed in the **Routing and Network Connectivity** section. The mesh header as used in Thread also allows efficient end-to-end fragmentation of messages rather than the hop-by-hop fragmentation specified in [RFC 4944].

The devices do not need IPv6 neighbor discovery as specified in [RFC 6775] as MLE messages (see **MLE Messages** section) are already used to discover neighboring nodes and routers.

Further details on 6LoWPAN usage and configuration are contained in the "Thread Usage of 6LoWPAN" white paper. Chapter 3 of the Thread specification details the specific 6LoWPAN configuration used.

3.3. ICMP

Devices support the ICMPv6 (Internet Control Message Protocol version 6) protocol [RFC 4443] and ICMPv6 error messages, as well as the echo request (ping) and echo reply messages.

3.3. UDP and TCP

Thread Devices use UDP (User Datagram Protocol) as defined in [RFC 768] for messaging between devices for mesh establishment and maintenance. Thread Networks also support TCP (or any other IPv6-based transport protocol) for application layer communication.

4. Network Topology

4.1. Overview

The Thread Specification enables full mesh connectivity between all Routers in a Thread Network.

The actual topology is based on the number of Routers in the Thread Network. If there is only one Router, then a basic star topology with a single Router is formed. If there is more than one Router then a mesh topology is automatically formed.

Figure 3 illustrates the basic topology of a Thread Network and the types of devices.

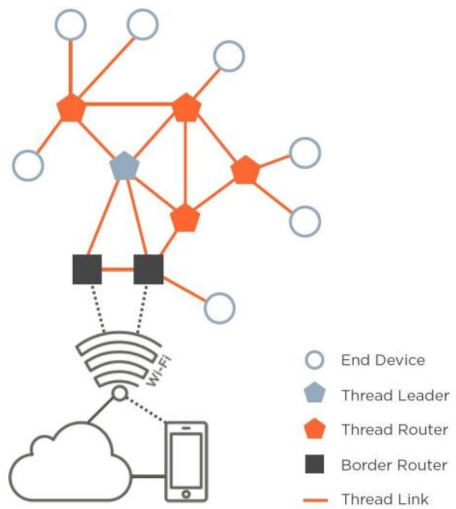

Figure 3. Basic Thread Network Topology and Devices

4.2. Mesh Networks

Mesh networks make wireless systems more reliable by allowing radios to forward messages for other radios. For example, if a node cannot send a message directly to another node, the mesh network forwards the message through one or more intermediary nodes. If an individual link fails, messages can be forwarded along an alternate path. As discussed in the Routing and Network Connectivity section below, the nature of the Thread Network is that all Router nodes maintain routes and connectivity with each other so the mesh is constantly maintained and connected. There is a limit of 32 active Routers in the Thread Network. However, 64 Router addresses are used to allow recycling of Router addresses.

In a Thread mesh network, the end devices do not route for other devices. These devices communicate via a Parent Router that handles the routing operations for its Child devices.

4.3. Routing and Network Connectivity

The Thread Network has up to 32 active Routers that use next-hop routing for messages based on the device's link-layer routing table. The link-layer routing table is maintained by the Thread Device to ensure all Routers have connectivity and up-to-date paths for any other Router in the Thread Network. Thread uses a distance-vector routing protocol similar to RIPng [RFC 2080], but with very compact message formats. All Routers exchange with other Routers their cost of routing to other Routers in the Thread Network in a compressed format using MLE (Mesh Link Establishment).

Note: From an IP standpoint, the Thread Network supports Routers and hosts. Hosts are any End Device, including REEDs, FEDs, MEDs, SEDs, SSEDs.

MLE (Mesh Link Establishment) Messages

MLE messages (see [draft-kelsey-intarea-mesh-link-establishment-06] which is extended for Thread in Chapter 4, Mesh Link Establishment, of the Thread Specification) are used for establishing and configuring secure radio links, detecting neighboring devices, and maintaining routing costs between devices in the Thread Network. MLE messages are transported using single-hop link-local unicast and multicast between Thread Devices.

MLE messages are used for identifying, configuring, and securing links to neighboring devices as the topology and physical environment change. MLE is also used to distribute configuration values that are shared across the Thread Network such as the channel and the Personal Area Network ID (PAN ID). These messages are forwarded with controlled flooding as specified by Multicast Protocol for Low-power and Lossy Networks (MPL) [RFC 7731].

MLE messages also ensure asymmetric link costs are considered when establishing routing costs between two devices. Asymmetric link qualities are common in IEEE 802.15.4 networks. To ensure two-way messaging is reliable, Thread Devices consider bidirectional link quality.

Route Discovery and Repair

Thread uses a proactive, distance-vector routing protocol to establish routes among Routers. On-demand (or reactive) route discovery is commonly used in low-power IEEE

802.15.4 mesh networks. However, on-demand route discovery is costly in terms of network overhead and bandwidth due to route discovery requests flooding the network.

In a Thread Network, all Routers periodically exchange single-hop MLE advertisement packets containing link quality and link cost information for all neighboring Routers, and path cost information for all other Routers in the Thread Network. Through these periodic, local updates, all Routers have up-to-date route cost information to any other Router in the Thread Network. If a route is no longer usable, Routers select the next most suitable route to the destination. This self-healing routing mechanism allows Routers to quickly detect when other Routers have dropped off the Thread Network, and to calculate the best routes to maintain connectivity to all other devices in the Thread Network.

The link quality in each direction is based on the link margin on incoming messages from that neighboring device. This incoming link margin is mapped to a link quality from 0 to 3. A value of 0 means unknown or infinite cost. The link margin is a measure of RSSI (Received Signal Strength Indicator) of received messages in dBm above the radio noise floor. The link quality value can then be mapped onto the link cost to use in route cost computations.

Table 2 summarizes the link quality and link cost.

Link Margin	**Link Quality**	**Link Cost**
None (~0 dB)	0	Unknown/infinite
Poor	1	4
Reasonable	2	2
Good	3	1

Table 2. Link Quality and Link Cost

The path cost to any other node in the Thread Network is then the minimum sum of link costs to reach that node. Routers monitor these costs as the radio link quality or topology of the network changes and propagate the new costs through the Thread Network using the periodic MLE advertisement messages.

To illustrate a simplified example, imagine a pre-commissioned network with shared security material where all devices are powered on at the same time. Each Router would periodically send an advertisement initially populated only with costs to single-hop neighbors. Internally, each Router would store next hop information that is not sent in the advertisement.

The first few advertisements would have path cost equal to link cost, because the only Routers that are known are immediate neighbors.

But as Routers start hearing advertisements from their neighbors that contain costs to other Routers that are two or more hops away, their tables populate with multi-hop path costs which then propagate even further out, until eventually there is connectivity information between all Routers in the network.

Routing to Child devices is done by looking at the high bits of the Child's RLOC16 address to determine the Parent Router address. Once the device knows the Parent Router, it has the path cost information and next-hop routing information for that destination device.

Forwarding

Devices use IP routing to forward packets. A device routing table is populated with a compressed form of a mesh-local address for each Router and the appropriate next hop.

When forwarding on the Thread Network, the upper 6 bits of the RLOC16 address define the Router address of the Destination Router. If the lower bits of the destination address are 0, then the Router is the final destination. Otherwise, the Destination Router is responsible for forwarding to the final destination, one of its Child nodes based on the lower 9 bits of the RLOC16 destination address.

For forwarding beyond the Thread Network, Border Routers are used. Each Border Router notifies the Leader of the particular IPv6 prefix(es) it serves and this information is distributed by the Leader as Thread Network Data using MLE messages. The Thread Network Data includes prefix data (which includes the prefix itself), the 6LoWPAN context for each prefix, the Border Routers serving each prefix and, optionally, a DHCPv6 server or DNS information for a prefix. If a device is to configure an IPv6 address using that prefix, it either uses SLAAC (Stateless Address Auto configuration) or contacts the appropriate DHCPv6 (Dynamic Host Configuration Protocol) server for this. The Thread Network Data also includes a list of RLOC addresses of Border Routers for routing packets to off-mesh destinations.

4.3. Retries and Acknowledgements

While UDP messaging is used in the Thread Specification for all mesh establishment and maintenance communication, reliable message delivery is still a requirement. This is done using a series of lightweight mechanisms as follows:

- MAC-level retries: Each device uses IEEE 802.15.4 MAC Acknowledgements from the next hop and will retry a message at the MAC layer if the MAC Acknowledgement is not received.
- Application-level retries: The application level can determine if message reliability is a critical parameter and may implement its own retry mechanism if necessary. Thread uses both MLE and CoAP to maintain the Thread network, both of which provide their own reliability mechanisms.

5. Adding a New Device to a Thread Network

There are three phases a new device must go through before it can participate in a Thread Network:

1. Discovery
2. Commissioning
3. Attaching

Once attached, a device is fully participating in the Thread Network and can exchange application layer information with other devices and services within and beyond the Thread Network.

5.1. Discovery

Before a Thread Device can participate in a Thread Network, it must first discover and establish contact with a Joiner Router for commissioning. The joining device iterates through all channels, issues an MLE Discovery Request on each channel, and waits for MLE Discovery Responses. The Discovery Response contains a payload including the network name and steering data, to steer devices into joining the Thread Network where they are expected. Once a device has discovered the Thread Network, it uses a link-local channel to the Joiner Router to establish a connection to the commissioning application and perform commissioning.

Discovery and commissioning are only required for the very first attachment of a Thread Device to a Thread Network. Every Thread device stores the Network Credentials in non-volatile memory for subsequent attachments.

5.2. Commissioning

Thread Commissioning is the process of authenticating a new device and providing it with the Network Credentials. For this, an authenticated DTLS session is established between a joining Thread Device and a commissioning application on a smartphone, tablet, or webpage. This session is used to securely authenticate the Joiner. If this process is successful, the Joiner Router securely provides the Joiner with the Thread Network Credentials, so that it can attach to the Thread Network.

5.3. Attaching

A detached Thread Device with Network Credentials will periodically attempt to attach to a Thread Network by multicasting MLE Parent Requests to nearby Routers and REEDs. If required, a REED will, upon hearing the parent request, upgrade to a Router role to support the connectivity of the newly attaching Thread Device. The attaching Thread Device and the Thread Router then use MLE Messages to configure a secure link and provision IPv6 addresses. A Thread Device will always attach as an End Device, and can upgrade to a Router later by requesting a Router ID from the Leader.

This attach process is illustrated in Figure 4.

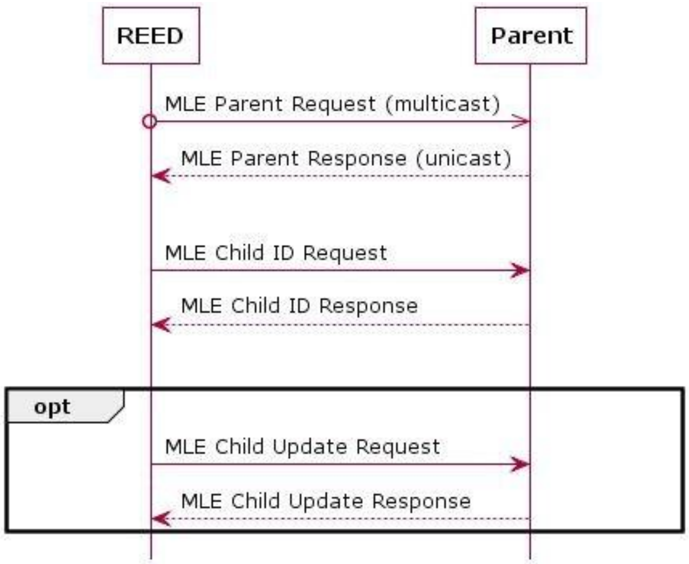

Figure 4. End Device (REED) with Thread Network Credentials attaching to a Parent Router

5.4. MLE Messages

Once a device attaches to a Thread Network, it needs certain information to maintain its participation in the network. MLE provides services to distribute network data throughout the network and exchange link costs and security frame counters between neighbors.

The MLE messages distribute or exchange the following information:

- The 16-bit short and 64-bit extended address of neighboring devices
- Device capabilities information including if it is a Sleepy End Device and the sleep cycle of the sleepy host device
- Neighbor link quality and link costs (if a Router)
- Security material and frame counters between devices
- Routing costs to all other Routers in the Thread Network
- Updates to Operational Dataset such as the channel or PAN ID

Note: MLE messages are encrypted except during discovery before the joining device has obtained the required Network Credentials. However, sensitive security materials (such as keys) are never sent in the clear within an unencrypted MLE message.

6. Network Management

6.1. ICMPv6

Thread devices support ICMPv6 error message processing, as well as the ICMPv6 echo request (ping) and echo reply messages.

6.2. Device Management and Diagnostics

The application layer on a Thread device has access to a set of device management and diagnostics information that can be used locally or collected and sent to other management devices. A commissioning application is also able to request this information from Thread devices.

The information used by Thread from the IEEE 802.15.4 MAC layer includes:

- 64-bit extended address
- 16-bit short address
- Capability information
- PAN ID
- Packets sent and received
- Packets dropped on transmit or receive
- Security errors
- Number of MAC retries

The information used by Thread from the network layer includes:

- IPv6 address list
- Neighbor table
- Child table
- Routing table

6. Conclusion

Thread is designed to address the unique interoperability, security, power, and architecture challenges of the IoT.

- Thread is a low-power wireless mesh networking protocol, based on the universally-supported Internet Protocol (IP), and built using open and proven standards.

- Thread enables device-to-device and device-to-cloud communications and reliably connects hundreds (or thousands) of products and includes mandatory-to-implement security features.
- Thread networks have no single point of failure, can self-heal and reconfigure when a device is added or removed, and are simple to set up and use.
- Thread is based on the broadly supported IEEE 802.15.4 radio standard, which is designed from the ground up for extremely low power consumption and low latency.