AN1316: Bluetooth Mesh Parameter Tuning for Network Optimization

This document describes in detail how the Bluetooth mesh toplogy can influence network operation. Additionally, tips are provided on how to tune your network and its nodes to achieve best performance.

For more detail on how to monitor Bluetooth Low Energy and Mesh data traffic, refer to *AN1317: Using the Network Analyzer with Bluetooth Low Energy and Mesh*. The present document is based on Bluetooth Mesh profile and model 1.0.1 specifications.

As a prerequisite, it is assumed that you are already familiar with Bluetooth Low Energy concepts such as GATT protocol, connections, advertising, broadcasting, and scanning. It is also assumed that you are familiar with the Bluetooth mesh terminology and are accustomed to the meaning of terms “node” and “provisioner”. For more detail, please refer to the Bluetooth Mesh profile specification 1.0.1.

KEY POINTS

- Bluetooth mesh networking
- Network topology
- Networking parameters.
- Advertising bearer versus GATT bearer

Table of Contents

1 Introduction....1
1.1 Bluetooth Mesh Topology....1
2 Bluetooth Mesh Node Principles and Optimization....2
2.1 Bluetooth Mesh Stack Layers....2
2.2 Node Features....3
2.2.1 Relay....3
2.2.2 Proxy....3
2.2.3 Friend....3
2.2.4 Low Power (LP) Node....4
2.3 Node Message Handling....4
2.3.1 Network Interfaces and Filtering....4
2.3.2 Network Message Cache....5
2.4 Bluetooth Mesh Stack Component....6
2.5 Radio Range....7
3 Bluetooth Mesh Networking Principles....8
3.1 Bluetooth Mesh Network Layer....8
3.1.1 IV Index....9
3.1.2 Time to Live (TTL)....9
3.1.3 SEQ....9
3.1.4 SRC and DST....9
3.2 Message Replay Protection....10
3.2.1 Replay Protection List....10
3.3 Relays....11
3.3.1 Network Repetitions and Repetition Delays....12

1 Introduction

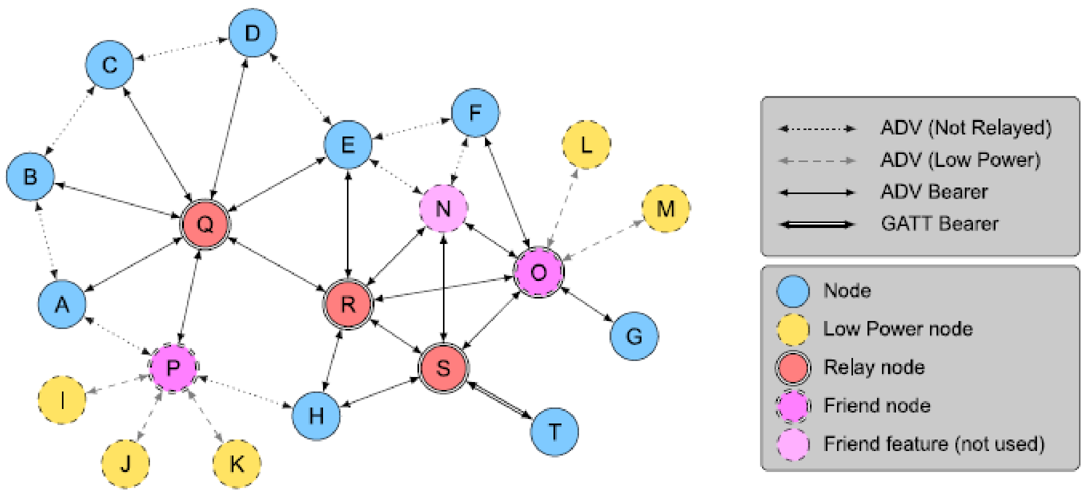
This section gives an overview of Bluetooth mesh topology concepts.

1.1 Bluetooth Mesh Topology

Although the Bluetooth mesh technology uses Bluetooth Low Energy (Bluetooth LE) for message transport, when it comes to data flow, there are significant differences.

Bluetooth mesh does not use a point-to-point communication scheme. Rather, it relies on advertising packets being sent and relayed through the mesh network. A node can identify if a network packet is addressed to itself via its destination mesh address (unicast, group or virtual address) which is different from a regular public or private Bluetooth LE address. For more information, please refer to the Bluetooth mesh profile specification.

The following example illustrates a Bluetooth mesh network with the various roles and bearers (taken from the Bluetooth mesh profile specification 1.0.1):



As opposed to Bluetooth LE, Bluetooth mesh nodes have a “many-to-many” relationship with mesh devices within their radio range. In the case where two nodes are not within radio range, the data is relayed by nodes supporting that feature from the emitter to the re-ceiver.

Additionally, nodes in a network have different energy consumption requirements. This impacts the network design, as a node taking part would constantly need to scan for incoming advertising packets and transmit, if necessary, which requires a relatively large amount of power. To mitigate that, nodes can be put to sleep and rely on other nodes within radio range and with fewer power constraints, to store the messages addressed to them.

Finally, Bluetooth mesh relies on the advertising/scanning state of the Bluetooth LE Link Layer. As a result, no low-level collision control exists. If a device does not support the advertising bearer, the GATT bearer can be used (relying on connection). This allows a device to communicate with nodes of a mesh network that support both bearers and that can operate as a proxy between the advertising and GATT bearer.

When building a Bluetooth mesh network, it is important to consider aspects such as these to ensure maximum efficiency.

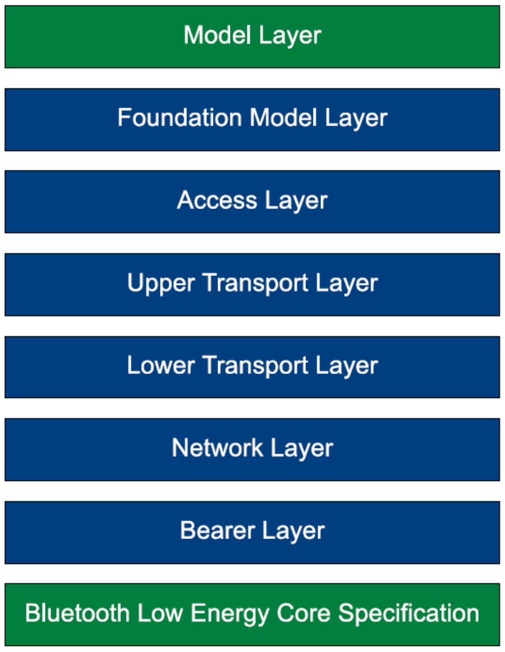
This document lays out the basic principles and discusses the main parameters that influence network operation.

2 Bluetooth Mesh Node Principles and Optimization

This section focuses on the network operation from the node perspective.

2.1 Bluetooth Mesh Stack Layers

The current Bluetooth mesh stack (Bluetooth Mesh profile/model spec 1.0.1) is built on top of the Bluetooth Low Energy core specification. The following image illustrates the layers composing the Bluetooth mesh stack running on each node:



The Bluetooth mesh specification defines two bearers over which mesh may be transported:

- An advertising bearer, using advertising packets as defined in the Bluetooth Low Energy core specification with dedicated mesh AD types.
- A GATT bearer, which allows devices not supporting the advertising bearer to participate to a mesh network. The GATT bearer can transmit and receive network layer packets, encapsulated in what is called the “Proxy protocol”, through regular Bluetooth Low Energy connections.

The network layer defines the Network packet data unit (PDU) format that allows data to be transported by the mesh bearers. It decrypts, authenticates, and forwards packets upstream, to the Lower Transport Layer, and downstream, to the bearer layer. For more detail on the Network layer and the Networks PDUs, see section 3 Bluetooth Mesh networking principles.

The Lower Transport Layer and Upper Transport Layer handle Transport PDUs. The Lower Transport Layer implements message segmentation and reassembly of Transport PDUs. Also at that level application keys associated with the message are identified. In the case of a friend node, a list of messages is stored for each associated low power (LP) node at this level. For more detail, see section 2.2.3 Friend.

The Upper Transport Layer implements message integrity checking and encryption/decryption.

Finally, the Access Layer defines the format of application data such as models. It also handles model publishing and subscribing. The access layer mainly handles application data and therefore will not be described extensively in this document, as it focuses on mesh networking operation.

2.2 Node Features

Node functionality is determined by the features that they support. Each node can support one or several of the features described in the Introduction, that is:

- Relay node
- Proxy node
- Friend node
- Low Power node

This section exposes the meaning of each feature with some practical recommendations.

2.2.1 Relay

Bluetooth mesh expands the range of the network by relaying messages. Any Bluetooth mesh device supporting the feature may be configured to act as a relay, but it is not mandatory for a network to have relay nodes (in case of small networks for example).

This relaying is undirected and is referred to as “message flooding”. It ensures a high probability of message delivery, without requiring any information on the network topology.

The Bluetooth mesh profile specification does not provide any routing mechanisms. As a result, all messages are forwarded by all relays until a counter-based time-to-live (TTL) mechanism reaches zero (more detail on TTL can be found in the dedicated section in the Bluetooth mesh profile specification). To avoid messages being forwarded by the same relays over and over, all Bluetooth mesh devices maintain a message cache, used for filtering out packets that the device has already handled.

Recommendations:

The flooding-based approach to message relaying can cause a lot of redundant traffic on air, which may impact the throughput and reliability of the network. Therefore, it is highly recommended to limit the number of relays in a network to restrict this effect.

The number of relay-enabled devices in the network is a trade-off between message redundancy and reliability. It should be tuned according to:

- Network size (number of nodes)
- Traffic volume
- Requirements for reliability and responsiveness

2.2.2 Proxy

To enable support for legacy Bluetooth LE devices that do not support receiving mesh network packets, Bluetooth mesh defines a separate protocol for tunneling mesh messages over the Bluetooth LE GATT protocol. For this purpose, the Bluetooth mesh profile specification defines a GATT bearer and the corresponding GATT Proxy Protocol. This protocol allows legacy Bluetooth LE devices to participate in the Bluetooth mesh network by establishing a GATT connection to a Bluetooth mesh device that has the proxy feature enabled.

The legacy device gets assigned an address and the necessary keys to become a full-fledged member of the network. The device receives the security credentials through the regular provisioning procedure or through some out-of-band mechanism.

Recommendations:

Unprovisioned devices should support both the advertising bearer and the GATT bearer.

A provisioner should support at least one of the bearers. It is highly recommended that it support the advertising bearer.

For a provisioner, when provisioning over the GATT bearer (PB-GATT) it is highly recommended that the connection interval for the connection between a Provisioner and device be between 250 and 1000 milliseconds (implementation-specific). This enables very low power operation for the device and allows the device to calculate the Diffie-Hellman shared secret without wasting significant energy to maintain an idle link.

2.2.3 Friend

To enable broadcast-based communication, the devices must continuously keep their radio in listening mode. This causes significantly higher power consumption than in a typical Bluetooth LE device.

To enable low power devices to take part in the mesh network, Bluetooth mesh contains a friendship feature. This protocol lets low power devices establish a relationship with a regular Bluetooth mesh device, which will then cache and forward messages to the low power device at regular intervals. This saves the low power device from having to stay on to listen for incoming messages.

Recommendations:

Limit the number of friend relationship a node can have, as this has direct effects on the message queue the friend node needs to maintain for the low power nodes it is associated with. If the friend queue is full and a new message needs to be stored, the oldest entries of the friend message queue will be discarded to make room for the new message.

2.2.4 Low Power (LP) Node

Some types of nodes have limited power resources, such as coin cell batteries. Those battery-powered devices need to conserve energy as much as possible. The low power node feature allows nodes that support it to establish an association with other nodes within radio range, to store mesh packets for them while they are on energy saving mode (deep sleep).

The devices at the other end of that relationship are called friend nodes. An LP node can only have one friend node, but a friend node can have a friend relationship with several LP nodes.

Furthermore, devices of this type may be predominantly concerned with sending messages but still have a need to occasionally receive messages.

Recommendations:

Low power edge nodes that mostly sends requests, such as light switches, do not need to act as a relay. Adjust the time-to-live counter (TTL) setting based on location within radio range of devices it controls.

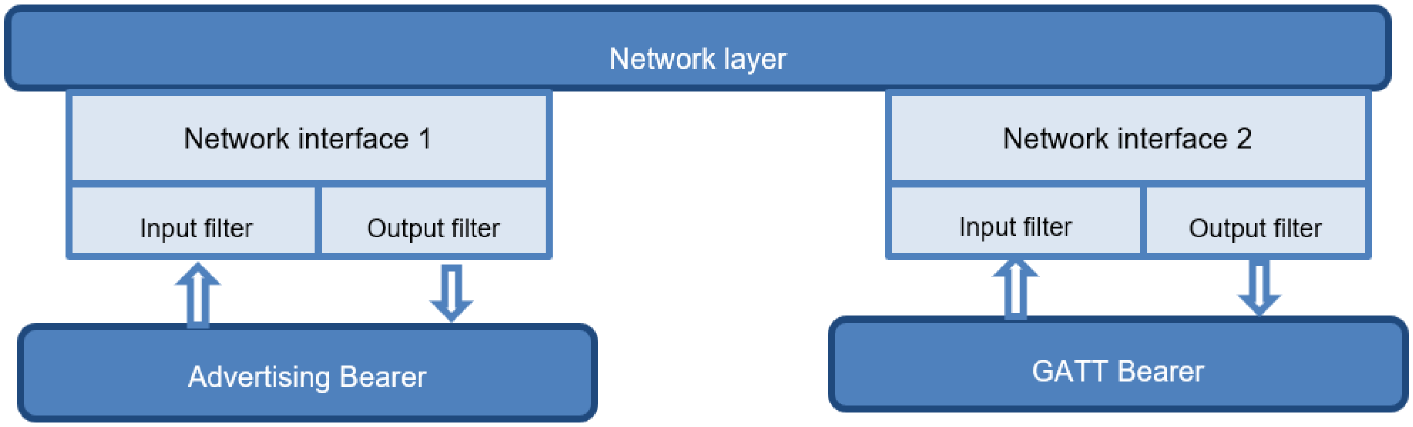
2.3 Node Message Handling

This section lays out the mechanisms put in place by the Bluetooth mesh stack for message handling on a node.

2.3.1 Network Interfaces and Filtering

On each node, the network layer supports sending and receiving messages via multiple bearers. As a result, multiple instances of a bearer might be present. Each of these instances interacts with the mesh network via a network interface.

For example, a node may have two network interfaces: one for Network PDU handling via the advertising bearer and another one for Network PDU handling via a GATT bearer. This is illustrated in the following figure.



The interface output filter of each interface makes sure only messages passing the defined rules are passed to the network layer and bearers. The output filter of the interface connected to advertising or GATT bearers must drop all messages with TTL values set to 1.

The following routines can be used to set up the input/output filtering rules for GATT bearers. For more information, see the Silicon Labs Bluetooth mesh API documentation:

- `sl_btmesh_proxy_allow()`
- `sl_btmesh_proxy_deny()`
- `sl_btmesh_proxy_set_filter_type()`

2.3.2 Network Message Cache

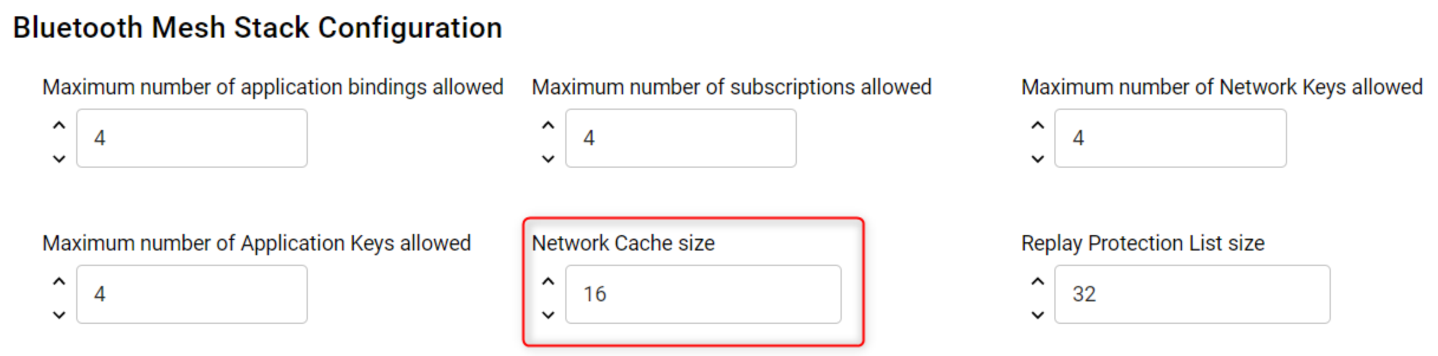
To reduce unnecessary security checks and excessive relaying, nodes include a network message cache of all recently seen Network PDUs. If a Network PDU is received that is already in the network message cache, then the Network PDU is not processed (that is, immediately discarded). If a Network PDU is received and that Network PDU is not in the message cache, then the Network PDU is processed (for example, checked against network security and relayed) and, if it is a valid Network PDU, it is stored in the network message cache.

It is not mandatory, nor is it necessary, to store entire Network PDUs in the network cache. For instance, the network cache as implemented in the Silicon Labs Bluetooth mesh stack consists of a ring buffer of cache entries stored internally by the stack in RAM. For resource optimization purposes, only some of the Network PDU information is stored in cache, rather than the encapsulated Transport PDU.

A cache entry, in the Silicon Labs Bluetooth mesh stack, is structured as:

- Network key index (2 bytes)
- IV index and SEQ number (4 bytes)
- Mesh source address (2 bytes)

That is 8 bytes in total. The total memory space allocated to the network cache is determined by the **Network Cache size** parameter in the **Bluetooth Mesh Stack** component. By default, this is set to 16 cache entries, or 128 bytes.

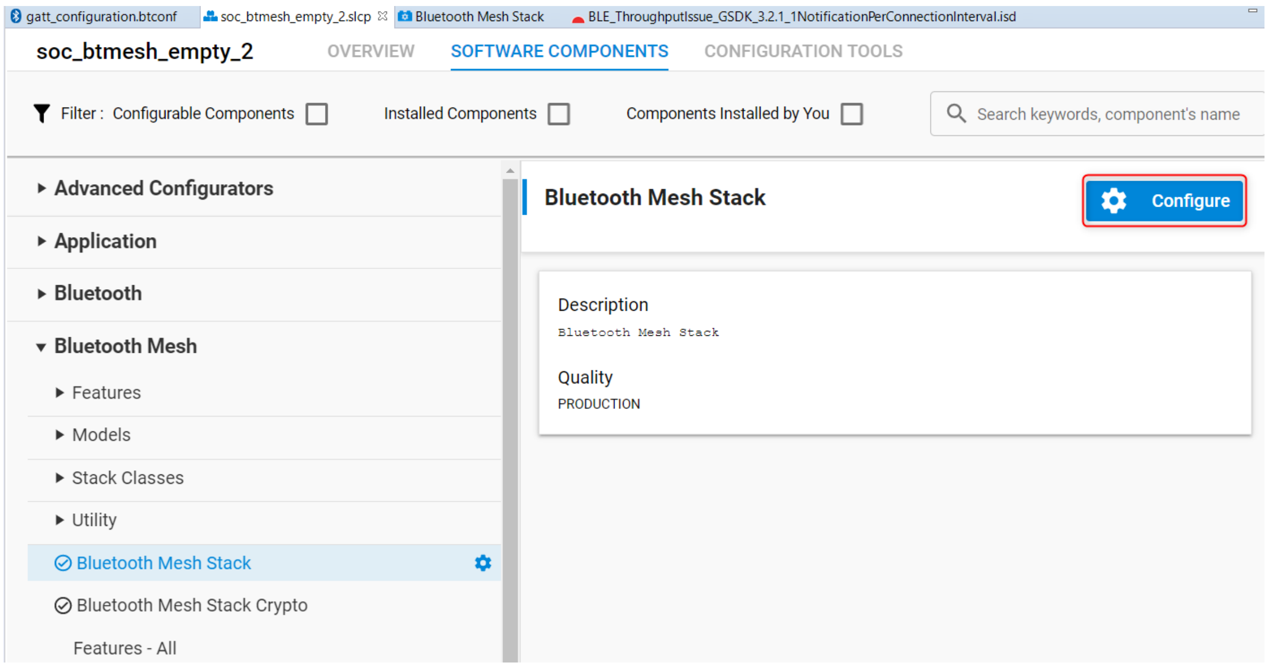

Bluetooth Mesh Stack Configuration

Maximum number of application bindings allowed: 4	Maximum number of subscriptions allowed: 4	Maximum number of Network Keys allowed: 4
Maximum number of Application Keys allowed: 4	Network Cache size: 16	Replay Protection List size: 32

Typically, it is good practice to set a large cache size for all nodes that support relaying and have it enabled.

2.4 Bluetooth Mesh Stack Component

As mentioned earlier, the Bluetooth mesh stack on each node can be configured to optimize network operation. This is done through the **Bluetooth Mesh Stack** component. Open the project by double-clicking the .slcp file, go to the SOFTWARE COMPONENTS tab, and search for and select the **Bluetooth Mesh Stack** component. Click **Configure**.

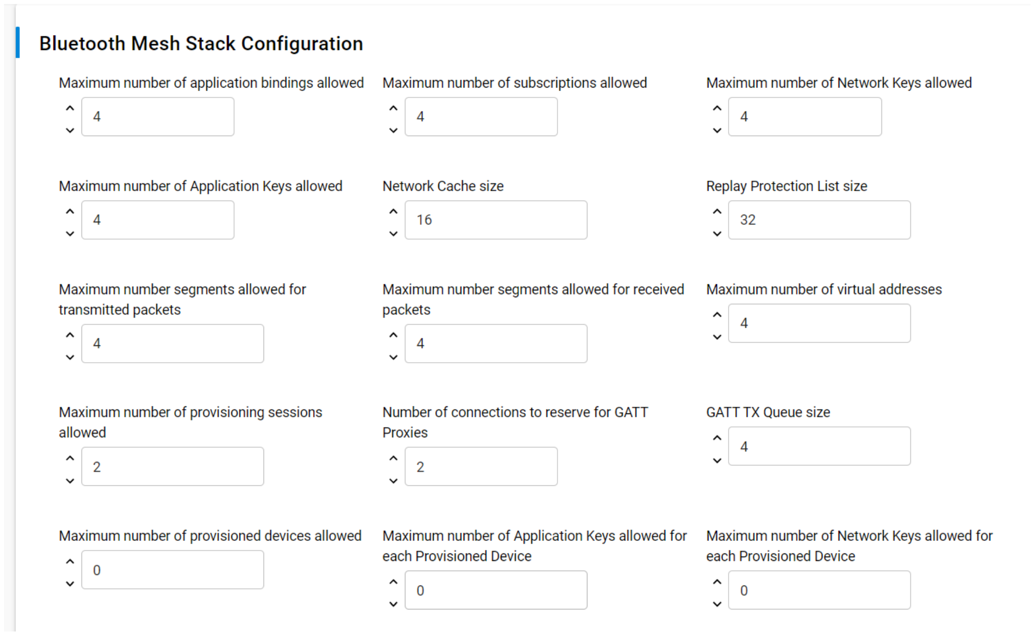


Bluetooth Mesh Stack

Description
Bluetooth Mesh Stack

Quality
PRODUCTION

The Component Editor opens.



Bluetooth Mesh Stack Configuration

Maximum number of application bindings allowed: 4	Maximum number of subscriptions allowed: 4	Maximum number of Network Keys allowed: 4
Maximum number of Application Keys allowed: 4	Network Cache size: 16	Replay Protection List size: 32
Maximum number segments allowed for transmitted packets: 4	Maximum number segments allowed for received packets: 4	Maximum number of virtual addresses: 4
Maximum number of provisioning sessions allowed: 2	Number of connections to reserve for GATT Proxies: 2	GATT TX Queue size: 4
Maximum number of provisioned devices allowed: 0	Maximum number of Application Keys allowed for each Provisioned Device: 0	Maximum number of Network Keys allowed for each Provisioned Device: 0

Among the configurable parameters, some are especially relevant for network efficiency optimization. A list of parameters that should be considered follows. For an exhaustive description of all the parameters, refer to UG472: Bluetooth Mesh Stack and Configurator User's Guide for SDK v2.x.

Network Cache size

For more detail on this, see section 2.3.2 Network Message Cache. It is good practice to increase the size of the cache on relay nodes. The number entered is the number of cache entries

Replay Protection List Size

For more detail on the replay protection list size see section 3 Bluetooth Mesh Networking Principles. The number entered is the number of replay protection list entries.

Maximum Number of Segments Allowed for Transmitted/Received Packets

At the transport level, messages can be segmented, meaning that large length messages can be sliced into smaller entities called segments and sent/received in bursts (or batches).

In the case of a receiver, *Maximum number segments allowed for received packets* determines the number of segments the node's stack receive task can process in one burst. To maximize throughput, it is good practice to increase the value of that parameter.

In the case of a transmitter, *Maximum number segments allowed for transmitted packets* determines the number of segments the node's stack transmit task can process in one burst. To maximize throughput, it is good practice to increase the value of that parameter.

The maximum number of segments for both receiving and transmitting is 32, corresponding to the maximum length of the access layer PDUs payload.

Do not interfere significantly with segmentation as this could have negative consequences. The use of unsegmented messages is always preferred.

For more detail on segmentation, see the Bluetooth mesh profile specification.

Maximum Number of Friendship Allowed

This parameter indicates, on a node that supports the Friend feature, the maximum number of friendships that can be established. This should be adjusted on each node.

Maximum Size of Total Friend Cache

On a node that supports the Friend feature, this parameter indicates the amount of RAM dedicated for the LP node message queues. This is does not have a direct effect on network operation but is strongly correlated to the following parameter *Maximum size of Cache for a single Friendship*.

Maximum Size of Cache for a Single Friendship

On a node that supports the Friend feature, this parameter indicates the length of the logical structure used by the stack to store LP node messages. The length of the message queue should be adjusted to the LP nodes' poll time, which is the duration they periodically spend in deep sleep.

Maximum Number of Connections to Reserve for GATT Proxies

On a node that supports the Proxy feature, this indicates maximum number of GATT bearer-based connections that can be made. This can be set to 0 when only the advertising bearer is used.

2.5 Radio Range

One parameter to take in account when setting up a mesh network is the reach of each node. Nodes that do not have a strong power constraint can increase the TX output power to enhance their reach. As a result, on such nodes, setting transmit power to the maximum legal value is recommended. This is, of course, not the case for LP nodes. In case of relay overlap, it is good practice to increase radio range.

It is recommended to keep the node's advertising rate as low as possible. Devices will usually have some advertising configured, including unprovisioned device/proxy advertisements and other miscellaneous advertising. As the number of nodes in the network increases, the rate of these advertisements can have a very detrimental effect on provisioning/configuration performance and network performance. The rate of these advertisements must be carefully selected by considering the expected size of the network and the application requirements (that is,. how quickly a mobile app needs to connect to a node, and so on).

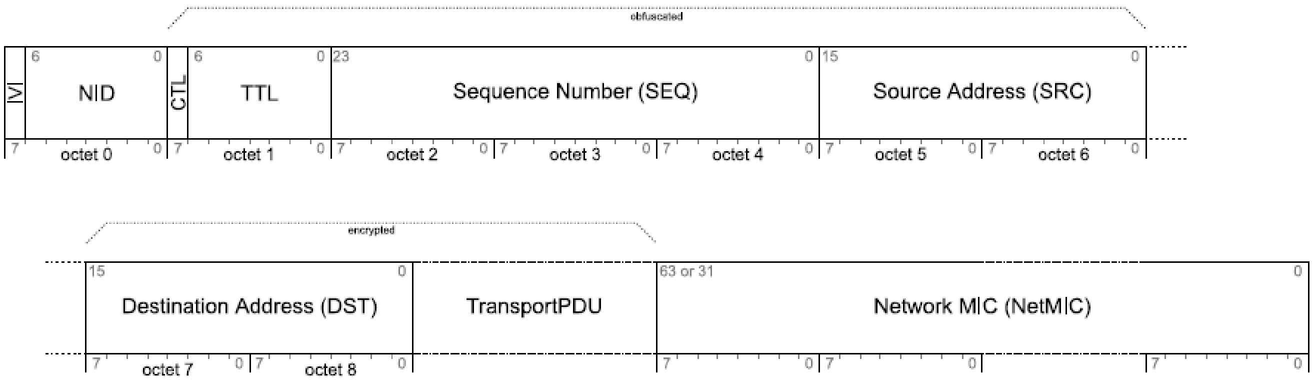
3 Bluetooth Mesh Networking Principles

This section describes Bluetooth mesh networking and the parameters influencing good operation. The information presented in this section focus on the following principles:

- **Scalability**. Adding and removing a node from a network should be a seamless procedure.
- **Robustness**. All information needed by a node for normal operation should reach it.
- **Efficiency**. Irrelevant radio communication should be reduced as much as possible to avoid collisions and unnecessary power consumption.
- **Security**. No weakness should be introduced in the network due to incorrect parameter settings.

3.1 Bluetooth Mesh Network Layer

The following description defines the Network PDU format:



The following table describes the fields:

Field Name	Bits	Description
IVI	1	Least significant bit of the IV index (see the Bluetooth mesh profile specification for more detail)
NID	7	Value derived from the network key used to identify the internal security keys used to secure this Network PDU (see the Bluetooth mesh profile specification for more detail)
CTL (1)	1	Network control
TTL (1)	7	Time to live
SEQ (1)	24	Sequence number
SRC (1)	16	Source address
DST (2)	16	Destination address
TransportPDU (2)	8 to 128	Transport protocol data unit
NetMIC	32 to 64	Message integrity check for network

(1) Fields are obfuscated, meaning that they are combined with the result of a single encryption function designed to prevent a passive eavesdropping.

(2) Fields are encrypted and authenticated. Authentication and encryption in this context is equivalent to Bluetooth LE terminology. For more detail on Network PDU encryption and obfuscation, refer to the Bluetooth mesh profile specification.

In the Network PDUs, not all fields are relevant for network operation optimization. The following parameters are the ones that should be considered first.

3.1.1 IV Index

The IV index is a 32-bit value known and shared by all nodes of a network. It is used for authentication and encryption at various levels in the stack (network, transport and application layers). For more information, see AN1318: IV Update in a Bluetooth Mesh Network.

3.1.2 Time to Live (TTL)

Every device acting as a relay will decrement the time-to-live (TTL) value in received messages and forward them if the TTL equals two or higher.

The following values are defined for the time-to-live counter:

- 0 – A Network PDU has never been relayed and will not be relayed.
- 1 – The Network PDU has been relayed but will not be relayed. All Network PDUs with this TTL value will be dropped by the node's network output filter.
- 2 to 126 – The Network PDU has been relayed and can be relayed.
- 127 – The Network PDU has not been relayed and will not be relayed.

The default TTL value of a node can be accessed and modified in the application code by any node supporting the configuration client model (typically the provisioner, but not necessarily):

- `sl_btmesh_config_client_get_default_ttl(net_key_index,node_address,pointer_to_handle)`
- `sl_btmesh_config_client_set_default_ttl(net_key_index,node_address,default_ttl_value,pointer_to_handle)`

This can also be done on model basis:

- `sl_btmesh_config_client_get_model_pub() /* See BT Mesh API documentation for the arguments */`
- `sl_btmesh_config_client_set_model_pub() /* See BT Mesh API documentation for the arguments */`

Note that the test API in the Silicon labs Bluetooth mesh API also allows modification of the TTL value for debugging purposes.

Adjust the default TTL value of a node carefully, depending on the supported features.

Typically, publishing model TTL value should be tailored to the application's need. As a trivial example, a lightness client model controlling a room within direct radio range should be set to publish with TTL=0. A sensor that sends data to a collector node somewhere in the network should publish with a TTL value big enough to just reach the collector across a chain of relays with and addition of 1 or 2 hops for relay redundancy.

Default TTL should be set so that Network PDUs can reach the Provisioner. If the provisioner is a mobile device accessing the network via a proxy node, the default TTL needs to be set so that the node reaches the whole network.

3.1.3 SEQ

Each Network PDU increments a sequence counter (SEQ) to protect the receiver node from replay attacks. It is a 24-bit value that can be combined with the source address and IV index of a network to identify messages individually. This value is part of the network cache and replay protection list entries in the Silicon Labs Bluetooth mesh stack.

3.1.4 SRC and DST

The SRC and DST fields are 16-bit values identifying the source and destination of the Network PDU. The SRC field is a unicast address, while the DST field may be a unicast, group, or virtual address.

Both fields are untouched when the PDU is relayed.

3.2 Message Replay Protection

A malicious device can passively receive all traffic whether encrypted or not, and then replay one or a sequence of messages to take advantage of the infrastructure. In common security terms, this is called a replay attack.

Since the originating element has encrypted and authenticated the message using the correct keys, the receiver node has no means to determine whether it is under a replay attack.

To increase protection against replay attacks, each element increases the sequence number (SEQ) for each new message that it sends. If a valid message has been received from an originating element with a specific sequence number, any future messages from the same originating element that contain numerically lower or equal sequence numbers than the last valid sequence number are very likely re-played messages and should be discarded. Therefore, messages are delivered to the access layer in sequence number order.

To guard against malicious devices replaying previous messages, every device keeps a running sequence number, which is used for outbound messages. Each Bluetooth mesh message is sent with a unique pair of sequence number and source address. When receiving a message, the receiving device stores the sequence number and makes sure that it is more recent than the last sequence number it received from the same source address. This is handled by the replay protection list described In the following section.

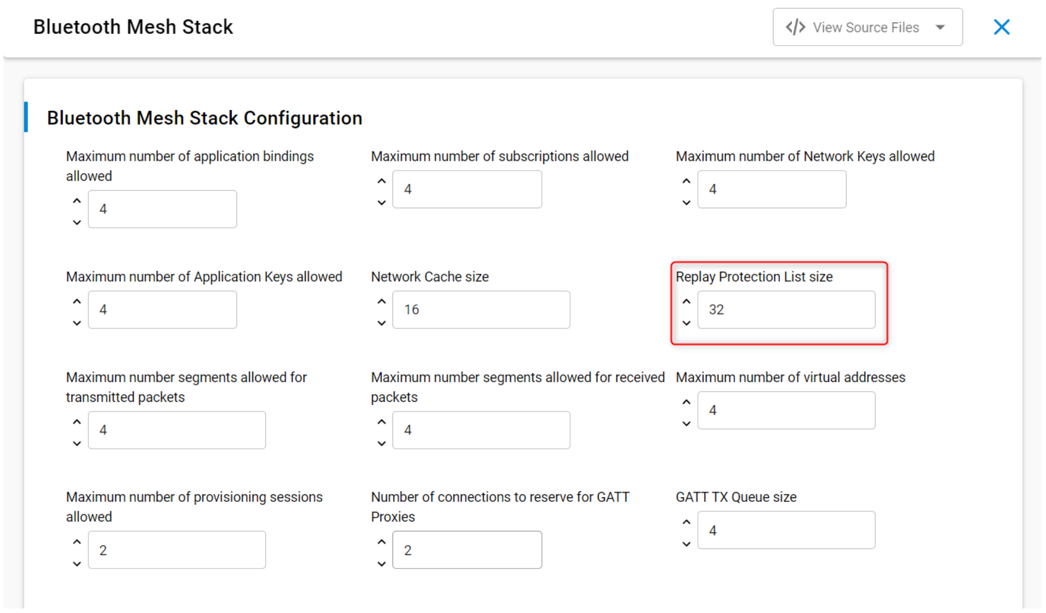
3.2.1 Replay Protection List

A replay protection list entry in the Silicon Labs Bluetooth mesh stack, is structured as:

- Source address (2 bytes)
- IV index and SEQ number (7 bytes)
- IV index and SEQ number authenticated (7 bytes)
- Saved flag (1 byte)

That totals 17 bytes. The total memory space allocated to the network replay protection list (RPL) is determined by the **Replay Protection List size** parameter in the **Bluetooth Mesh stack** configuration component. The default setting for the replay protection list is 32.

The RPL size sets the upper limit for the number of devices (or more accurately, the number of elements) this device can communicate with. When the RPL cannot be amended with an entry corresponding to a new received message, because the RPL is full, any such message will be dropped



Bluetooth Mesh Stack Configuration

Maximum number of application bindings allowed	4
Maximum number of subscriptions allowed	4
Maximum number of Network Keys allowed	4
Maximum number of Application Keys allowed	4
Network Cache size	16
Replay Protection List size	32
Maximum number segments allowed for transmitted packets	4
Maximum number segments allowed for received packets	4
Maximum number of virtual addresses	4
Maximum number of provisioning sessions allowed	2
Number of connections to reserve for GATT Proxies	2
GATT TX Queue size	4

The composition data page 0, part of what is referred to as Device Composition Data (DCD) in the Silicon Labs Bluetooth mesh stack, contains a field indicating the minimum number of entries in a node replay protection list. The following table illustrate the content:

Field	Size (bytes)	Description
CID	2	16-bit company identifier.
PID	2	16-bit vendor-assigned product identifier.
VID	2	16-bit vendor-assigned product version identifier.
CRPL	2	16-bit value representing the minimum number of replay protection list entries in a device.
Features	2	Contains a bit field indicating the supported features of the device.
Elements	variable	A sequence of elements

The CRPL value corresponds to the value set in the **Bluetooth Mesh Stack** component.

Silicon Labs' stack allows you to save the replay protection list in non-volatile memory (NVM) to ensure replay protection over power cycles. The following routines can be used:

- `sl_btmesh_node_save_replay_protection_list()`
- `sl_btmesh_node_get_replay_protection_list_status(uint16_t * total_entries, uint16_t * unsaved_en-tries)`

The replay protection list should be saved regularly, especially after a topology change or an IV index update, for example. When a node is removed or added in the network. It is good practice to save the RPL before the devices power off.

The publish rate of a model on each node (set by the configuration client) is also a critical parameter. If the publish rate is too high, it can drastically increase the traffic on the network and cause sluggish performance.

For more information, see the Silicon Labs Bluetooth mesh API documentation.

3.3 Relays

Relaying in Bluetooth mesh networks is important, but often misuse of that feature leads to problems. This section clarifies the functionality and suggests ways to improve robustness and scalability of your network.

As mentioned earlier in this document, message flooding is the mechanism put in place to ensure two nodes unable to reach each other directly by radio, can still communicate. This means that messages coming from the origin node will be received by relay nodes within radio range and then retransmitted to other relay nodes until the information finally reaches its original destination node.

This is a very effective and simple way to solve the network scalability issue but comes with some drawbacks:

- If too many relay nodes are enabled in the network, the redundant data traffic may increase in an uncontrollable way.
- Relay nodes must constantly listen for incoming advertising packets and retransmit them, which is very costly in term of power.
- A lot more information travels over the air, potentially making attacks easier.

As a result, a network designer must take caution when using the relay feature and limit the number of relay nodes to what is necessary.

For example:

- Devices powered by a building's electric system, such as light bulbs, are not power-constrained and most likely will be active on the mesh network. As a result, it makes sense for these nodes to support the relay feature. The network cache and replay protection list size should be adjusted depending on how many nodes surrounds it. Additionally, the TTL value should be set based on the hop count needed to reach its configuration device.
- An LP node would typically never need to support the relay feature, as it is incompatible with its power constraints.

3.3.1 Network Repetitions and Repetition Delays

Relays retransmit network PDUs. The configuration client (provisioner, configurator) determines whether a device behaves as a relay along with the number of retransmissions and the delay between retransmissions. The Silicon Labs configuration client commands for these are as follows:

- `sl_btmesh_config_client_get_relay(net_key_index,node_address)`
- `sl_btmesh_config_client_set_relay(net_key_index,node_address,value,retransmit_count,retrans-mit_interval)`

The number of retransmission count is encoded on 3 bits and goes from 0b000, meaning the messages are retransmitted once, to 0b111, meaning one transmission and 7 retransmissions.

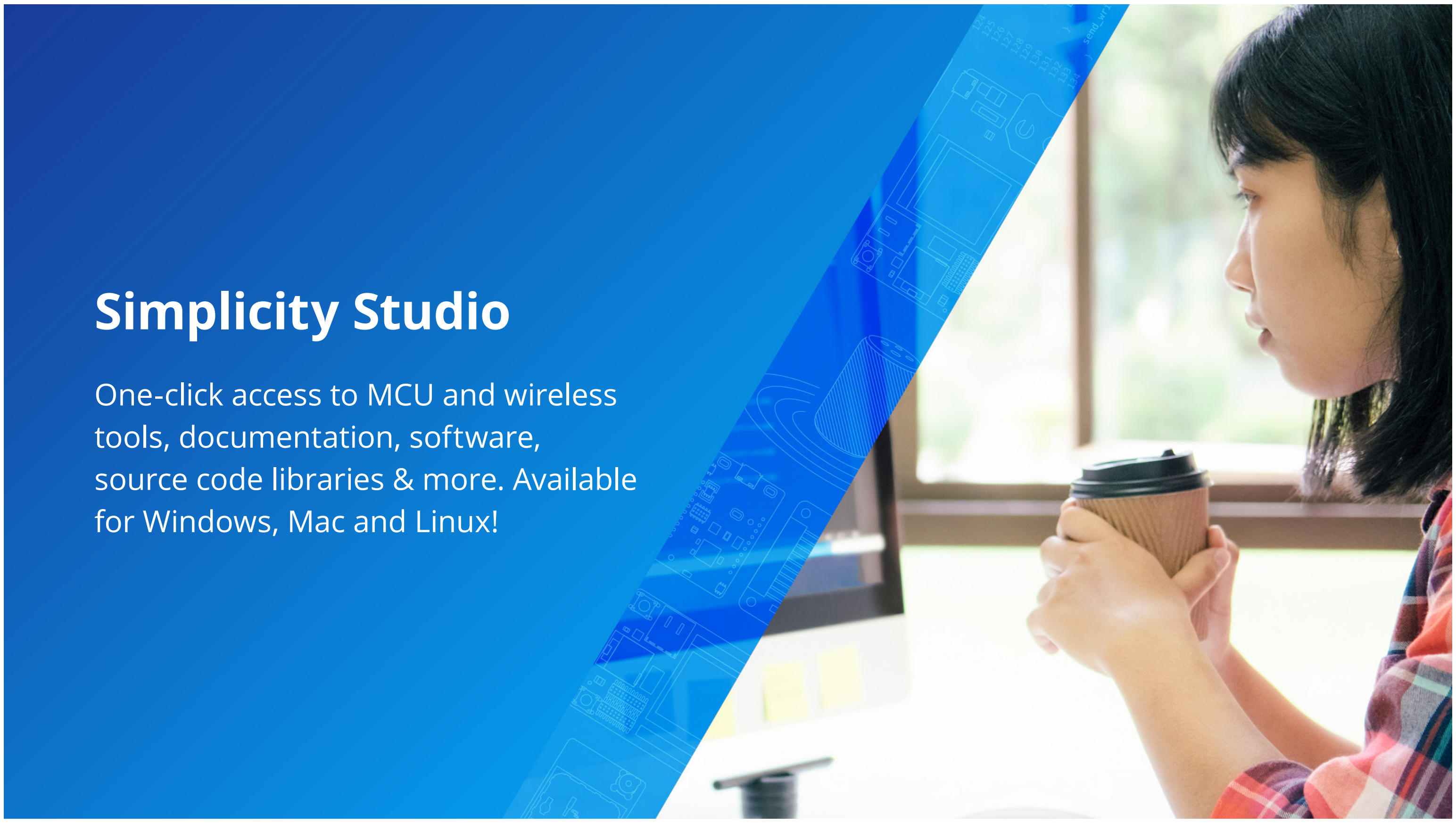
The time interval between retransmissions is encoded on a 5-bit value multiplied by a 10 ms time step. The value 0b00000 represent 10 ms: (0b00000 +1) * 10 ms. The value 0b11111 represent 320 ms: (0b11111 + 1) * 10 ms.

This is described in the Bluetooth mesh profile specification as relay state and should be carefully set to achieve maximum network performance. Additionally, a similar mechanism at the network layer level exists for non-relay nodes This is described in the Bluetooth mesh profile specification as network transmit state.

- `sl_btmesh_config_client_get_network_transmit(net_key_index,node_address,pointer_to_handle)`
- `sl_btmesh_config_client_set_network_transmit(net_key_index,node_address,transmit_count,trans-mit_interval,pointer_to_handle)`

Those routines allow a configuring node that is supporting the configuration client model to set the network transmit state. For more information, see the Silicon Labs Bluetooth mesh API documentation.

Only in areas with very few relays should the network/relay repetitions be enabled.


Simplicity Studio
One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com