SILICON LABS

Bluetooth Mesh Technology

Wireless Technology for the World of IoT

Bluetooth Mesh: Wireless Technology for the World of IoT

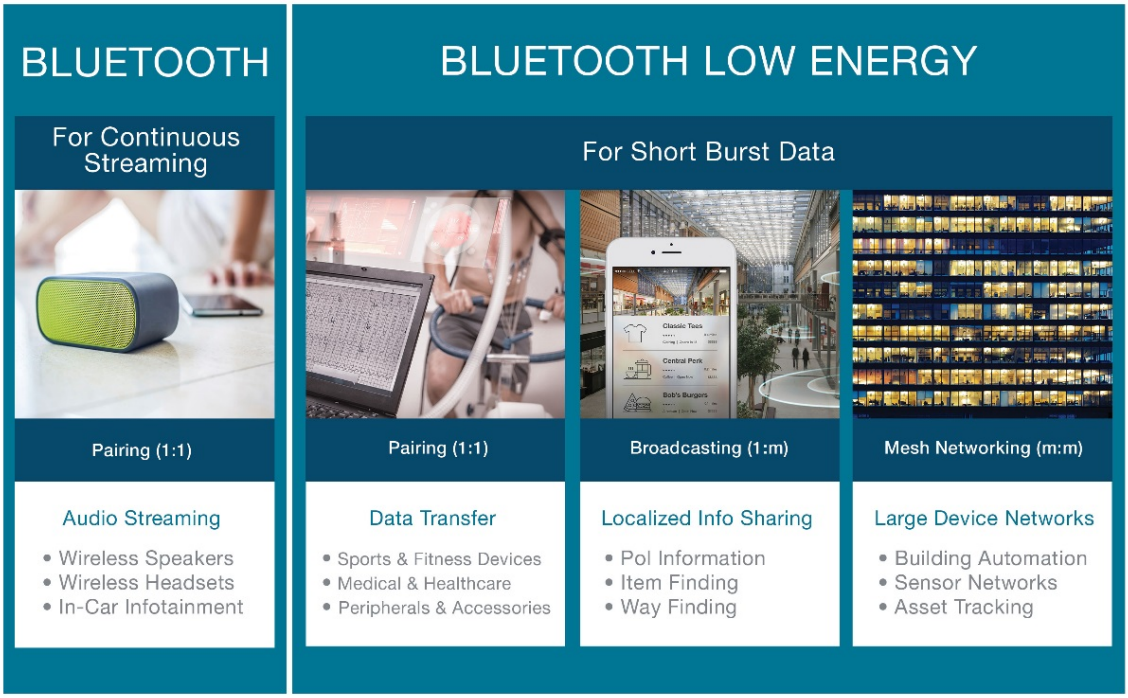
Henrik Snellman, Paul Benford, Hannu Mallat, Mikko Savolainen

Bluetooth mesh networking is a new topology available for Bluetooth LE devices that enables many-to-many communications. It is optimized for creating large scale device networks and ideally suited for building automation, sensor network, and asset tracking solutions. This article takes a look at this new exciting Bluetooth standard.

Table of Contents:

Bluetooth BR/EDR technology was originally developed to fulfill the need for a short-range wireless point-to-point connectivity between PCs, mobile phones and accessories like headsets, keyboards and mice. Since it's introduction this technology has been constantly improved and enhanced, but it has also become the de facto wireless technology for voice and audio streaming and use cases like wireless speakers, headphones and in-car infotainment.

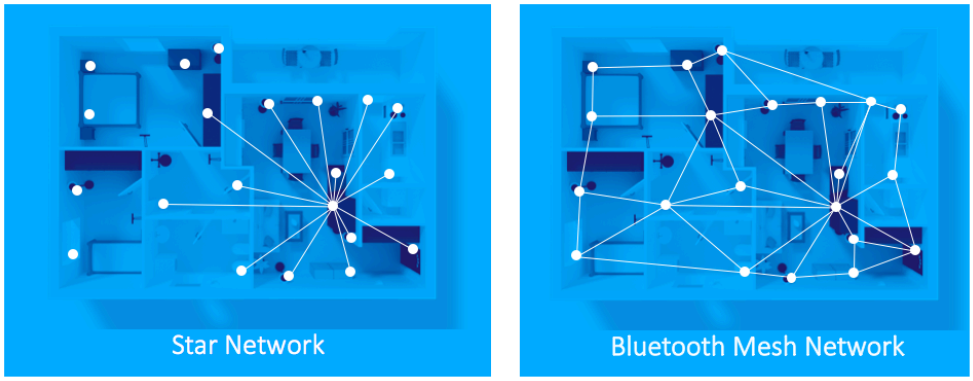
However, Bluetooth BR/EDR technology was not optimized for low power applications such as sports and fitness sensors, wearable devices and low-power PC accessories. To address these markets Bluetooth SIG developed the Bluetooth Low Energy (LE) technology which enables short burst wireless connections and data broadcast. Since it's introduction Bluetooth LE was quickly adopted into smart phones and tablets and it enabled new applications such as wearable devices, sports and fitness sensors and beacon applications and helped create new markets for Bluetooth technology. With the introduction of Bluetooth mesh Bluetooth SIG addresses the needs of the quickly growing IoT market. The figure on next page summarizes the evolution of the Bluetooth standard with Bluetooth mesh at the far right.



Development of Bluetooth standard from Bluetooth BR/EDR (left) to LE and Bluetooth mesh (right)

New Topology Paradigm

Bluetooth LE technology is based on point-to-point, star, or broadcast based network topologies. Bluetooth mesh changes this as it allows Bluetooth devices to be used in a mesh networking topology (see figure below). This major topology paradigm change will provide many new use cases for Bluetooth technology and boost innovation.



Bluetooth BR/EDR topology (left) and Bluetooth mesh topology (right)

Bluetooth mesh allows us to move away from the typical personal area network (PAN) which Bluetooth is known for and extends the range and number of devices a Bluetooth network can contain. A smart home is a great example and use case for Bluetooth mesh. Smart homes typically require 30 - 50 devices which usually are not all within a single-hop distance from each other. Bluetooth mesh enables networking all devices into a single mesh network which can cover the entire household. The really exciting point with Bluetooth mesh is the fact that its nodes can also support existing Bluetooth LE topologies and use cases such as point-to-point connectivity and Bluetooth beaconing. This allows smart phones to be connected to the Bluetooth mesh network to control and monitor nodes. Furthermore,

some really exciting new use cases such as indoor positioning and asset tracking are also feasible. Bluetooth SIG hast just released a collection of specifications related to the Bluetooth mesh version 1.0 version [1], [2], [3] and anyone can now start to develop applications based on this new Bluetooth standard. This white paper introduces the key features and capabilities of Bluetooth mesh technology and contains information about selected features designed for sensor, low power and lighting applications.

Managed Flooding

The initial Bluetooth mesh specification is based on a flooding type mesh. Broadcast channels are used to transmit messages from nodes. Messages are received by other nodes and relayed further. This makes it possible to extend the range by adding more nodes. If the density of the mesh nodes is sufficient any node can send messages to any other node directly (full mesh) or first to nodes close-by after which the nodes relay the messages onward to further nodes (partial mesh). Mesh networks are most often of the partial type with messages being relayed from node to node until all nodes have received the original message. Flooding-based mesh networks are considered easier to implement and more resilient if compared with routing-based mesh networks. On the other hand routing-based networks have better scalability.

Bluetooth mesh specification includes several features which warrant the term “managed flooding”. These features are listed in the table below.

Feature	Description
Heartbeats	• *Nodes send heartbeat messages periodically to indicate activity.* • *Heartbeat messages contain data to allow receiving nodes to determine the number of hops between the nodes which can be taken into account using the Time to Live (TTL) parameter (see below).*
Time to Live (TTL)	• *All Bluetooth mesh PDU’s contain this field.* • *Controls maximum number of hops.*
Network Message Cache	• *Compulsory for each node.* • *All recently received messages are stored in the cache.* • *If the received message is already in the cache it is an indication that the message has been processed already and is therefore discarded immediately.*
Friendship	• *Used in combination with Low Power nodes.* • *Friendship node receives and buffers addressed to the configured Low Power node and forwards buffered messages to the Low Power node when it becomes active after waking up.*
Pre-emptive “filtering”	• *Received messages are passed up the stack and various checks are performed by the Network Layer to decide whether to discard the message or pass further to higher layers.* • *PDU Network ID field is used to decide if the NetKey used to encrypt the message is recognized or not. If the NetKey is not known the PDU is discarded.* • *If the Message Integrity Check (MIC) check fails when using the NetKey corresponding to the PDU’s Network ID the message is discarded.* • *Optimizes filtering of messages not addressed to the node.* • *AppKey of message is used in Upper Transport Layer with the help of Application Identifier.* • *If the Application ID is not recognized the PDU is discarded by the Upper Transport Layer.* • *If the transport Message Integrity Check (MIC) is not succesful the message is discarded.*

Downward Compatibility using GATT

Currently used “traditional” Bluetooth devices would not be able to send Bluetooth mesh messages nor would they be able to interpret them. Updating all existing Bluetooth devices would not be feasible so Bluetooth SIG has provided downward compatibility by including GATT features enabling connectivity with any device supporting GATT. This guarantees downward compatibility without the need for hardware or software updates as long as the device is compatible with Bluetooth 4.0 or later. Mobile phones which do not support Bluetooth mesh may also be used with Bluetooth mesh networks if the Bluetooth stack of the mobile device supports Bluetooth GATT connectivity.

Architecture Overview

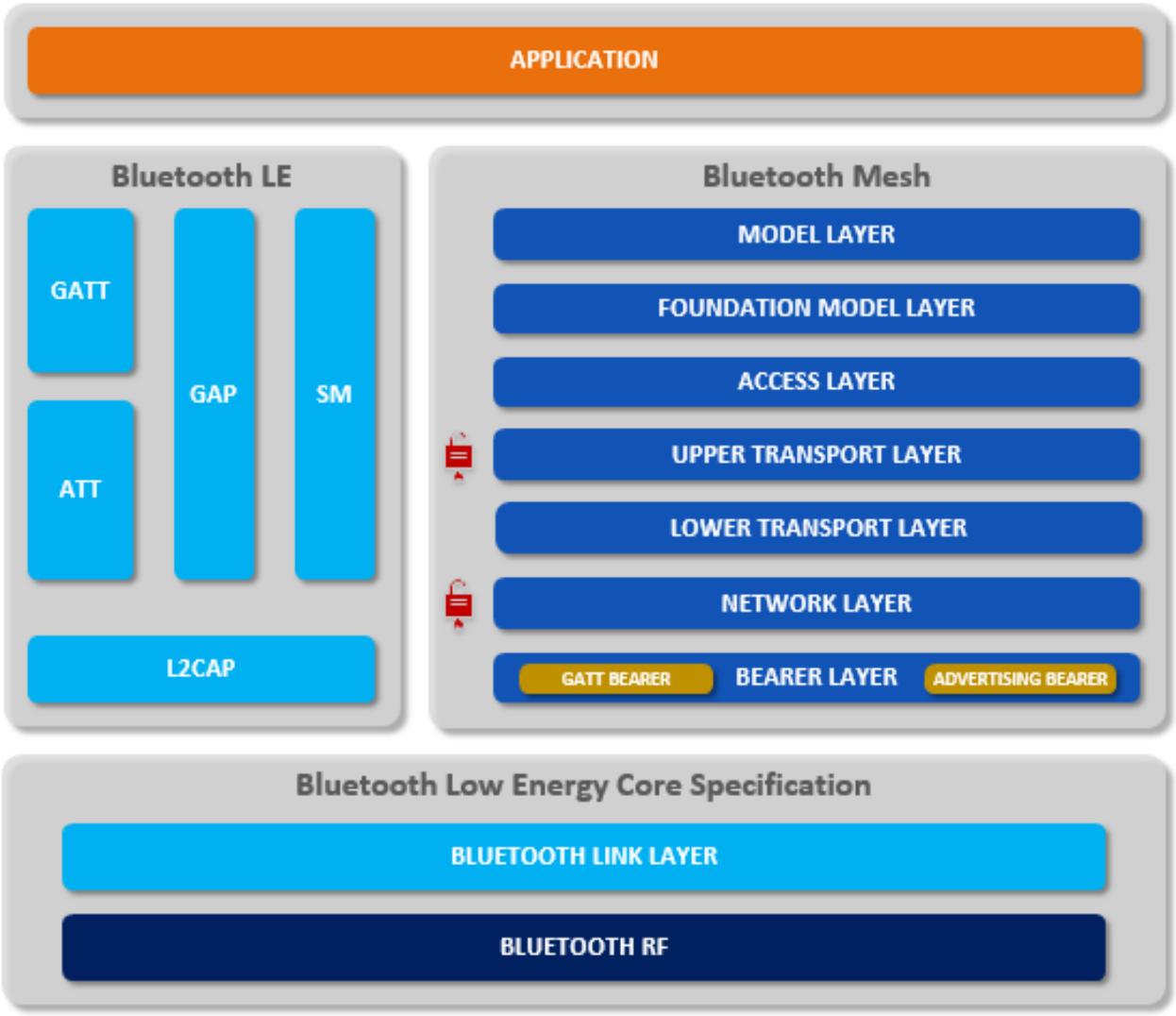
Bluetooth mesh architecture consists of seven layers on top of the Bluetooth Low Energy Core Specification, as shown in the figure on next page.

The Model Layer defines models which standardize typical user scenario operations as defined in the Bluetooth Mesh Model specification [2] and other possible higher layer specifications. These include among others model specifications for lighting and sensors. The Foundation Model Layer defines the states, messages and models needed for configuring and managing a Bluetooth mesh network. Higher Layer application usage of the Upper Transport Layer is defined in the Access Layer while the Upper Transport Layer guarantees confidentiality of access messages through encryption, decryption and authentication as well as defines the way transport control messages are used for managing the Upper Transport Layer between nodes, also for the Friend feature.

Segmentation and reassembly of Upper Transport Layer messages is defined by the Lower Transport Layer. Segmentation enables the transfer or large Upper Transport Layer messages to other nodes by splitting large PDU’s into multiple Lower Transport PDU’s. Single control messages for managing segmentation and reassembly are also defined at this level.

Addressing of transport messages is defined by the Network Layer and also defines the Network Message format used to allow Transport PDU’s to be handled by the Bearer Layer. Rejection of messages, decision to relay messages or to further process the messages instead of rejecting them is managed by the Network layer. Bearer Layer defines the transportation of network messages between nodes. An important point is the inclusion of two different bearers, one for Advertising and one for GATT.

According to Bluetooth SIG there may be additional Bearers introduced in the future. The inclusion of the GATT Bearer makes it possible to carry out messaging with any Bluetooth device compatible with GATT. The above listed seven layers are designed on top of Bluetooth Low Energy Core Specification (Bluetooth 4.0 Low Energy).



Bluetooth mesh architecture (right) with Bluetooth LE architecture (left)

Introduction to Nodes

Basically a Bluetooth mesh network is a collection of nodes capable of transmitting and receiving data. Thus, to understand how a Bluetooth mesh network really functions requires an understanding of how individual nodes operate and which features they support. This section introduces the main concepts. For a full description of node functionality see the Bluetooth mesh Model Specification available from Bluetooth SIG [2].

Features

In addition to the basic capability of receiving and transmitting messages Bluetooth mesh nodes can support one or more of the following four features: Relay, Proxy, Friend and Low Power. Bluetooth mesh specification mentions also a fifth “feature” or node type called Gateway. It is a node which translates messages between a Bluetooth mesh network and other wireless technology networks.

These features are listed in the table on the following page.

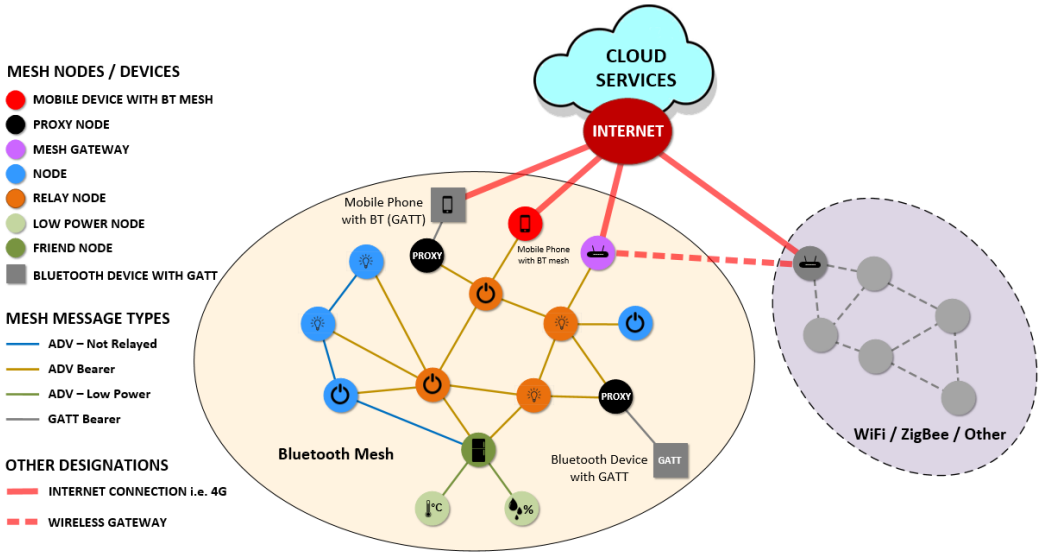
Optional node feature	Description
Relay	Receive and retransmit mesh messages by using the Advertising Bearer to enable larger networks
Proxy	Receive and retransmit mesh messages between GATT and Advertisement Bearers
Low Power	Operation at significantly reduced receiver duty cycle in conjunction with a node supporting the Friend feature
Friend	Enables helping a node supporting the Low Power feature by storing messages on its behalf

Additionally

Gateway	Used between a Bluetooth mesh and a non-Bluetooth wireless network to allow data sharing based on protocol conversion

Relay nodes are typically powered from an electrical outlet and are awake all or most of the time to be able to receive and relay messages at any time. Proxy nodes are used to enable the use of older Bluetooth devices as part of a Bluetooth mesh network. The prerequisite here is that the device in question must support Bluetooth GATT functionality. For example an older mobile phone with a Bluetooth stack supporting GATT can be connected to a Bluetooth mesh network through a Proxy node. Low Power nodes are powered typically by battery power or may use energy harvesting methods. They are programmed to wake up every now and then for receiving buffered messages from a Friend node and for transmitting messages to other nodes in the mesh. Friend nodes help the Low Power nodes minimize power consumption by acting as data storage buffers. It should be noted that a node may be programmed to use one or more of the listed features as required. If no feature is activated the node will still be able to receive and send mesh messages.

The figure below shows a small Bluetooth mesh network as an example. Color coding is used to distinguish the supported feature of each node. The small example network includes some switches, lights, a refrigerator, a temperature sensor and humidity sensor, proxy nodes to enable connection of older Bluetooth devices and a Gateway node to enable data exchange with a neighboring non-Bluetooth wireless network. The mesh network is also connected to the Internet through the mobile phones. One of the mobile phones is older but its software stack supports GATT. The Gateway node device may also include mobile phone functionality (e.g. 4G) to allow a connection to the Internet but mobile phones can also be used for that purpose.



Bluetooth mesh node roles and related message types

Resources

Nodes share four common resources which are Network Addresses, Network Keys, Application Keys and IV Index. Network Addresses are required for identifying message sources and destinations. Network Keys are for securing and authenticating messages at the Network Layer. Application Keys are used for a similar purpose but on Access Layer. IV Index is used to extend or limit the lifetime of the network. These node resources are listed in the table below.

Node resource	Description
Network Addresses	Identification of message sources and destinations
Network Keys	Securing and authentication of messages at the Network Layer
Application Keys	Securing and authentication of messages at the Access Layer
IV Index	Network lifetime extension

Network Keys and Application Keys both are generated by using a random number generator as defined in the Bluetooth Core Specification [4]. Both of these keys are shared between nodes. It should be noted that Application Keys are related to Network Keys and an Application Key is bound to a single Network Key. That is, an Application Key is valid only when used with the proper Network Key.

The IV Index field in the network PDU is a 1-bit value which is the least significant bit of the actual IV Index value used in the nonce to authenticate and encrypt the related Network PDU. IV Index is a 32-bit value shared between all nodes and subnets of a particular mesh network. Since IV Index is changed on a more or less regular basis and because its value is included in the obfuscation the benefit is that possible obfuscation attacks would need to be started again.

Elements

Each node must have at least one defined element called the Primary Element. In addition to the Primary Element nodes may have one or more additional elements called Secondary Elements. As long as a node is part of a mesh network the number of its elements remains the same. Provisioner assigns an address to the Primary Element of the node, and the Secondary Elements are implicitly assigned consecutive addresses. This unicast address points to the Primary Element of the node. Unicast element addresses are used by nodes to identify which element in a node is transmitting or receiving a message.

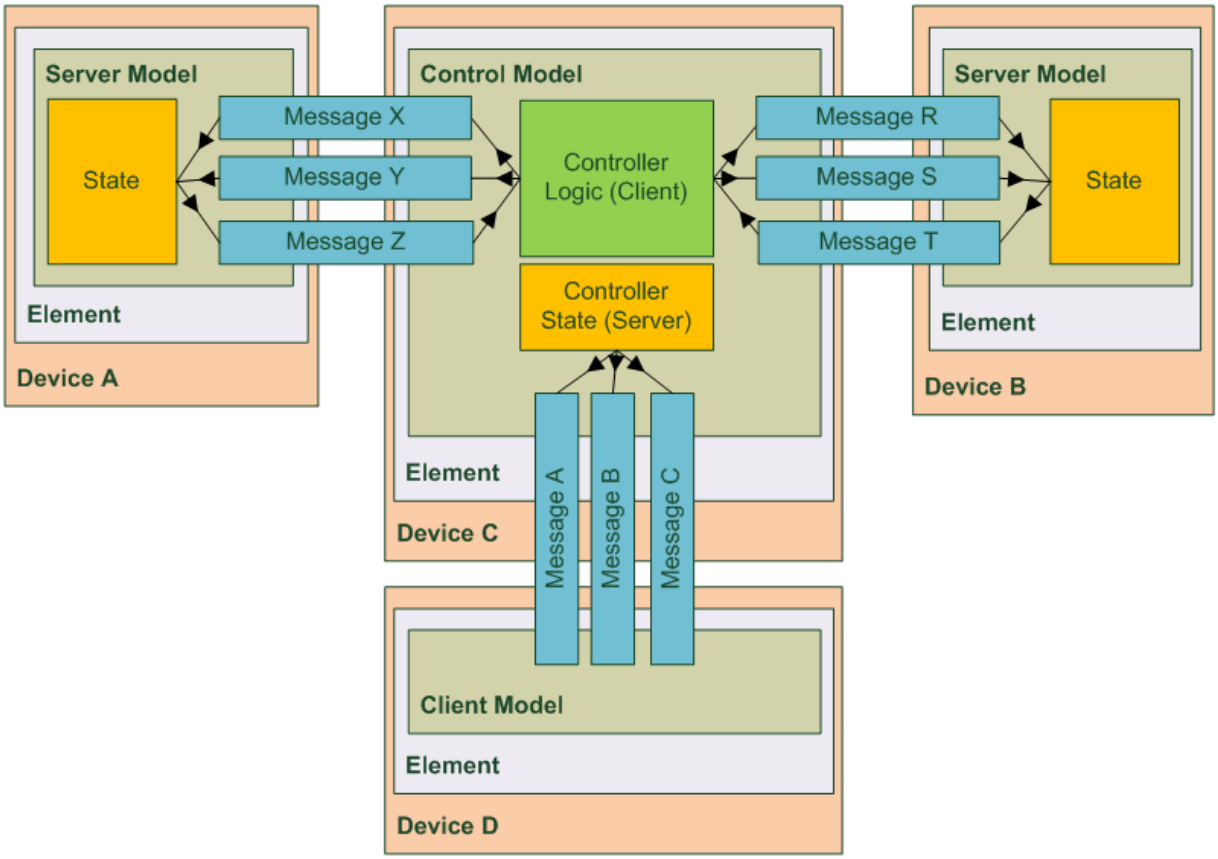
All elements also have a GATT Bluetooth Namespace Descriptor value to help determine the part of the node the element in question represents. Namespace descriptor value definitions are the same as in GATT. As an example, a temperature sensor has the values “inside” and “outside”. This also makes using older Bluetooth devices supporting GATT functionality as part of a Bluetooth mesh network possible.

Server, Client, and Control Models

The basic functionality of a node is defined by models, which define the required states, related messages and other behavior. Bluetooth mesh defines three types of models: Server, Client and Control. A single device can contain one, two or all three types of models. Mesh applications are defined using a client-server architecture communicating using a publish-subscribe paradigm.

Server Models are compositions of one or more states spanning over one or more elements. In addition Server Models also define State Transitions, State Bindings and transmittable and receivable messages of elements within the Model. Furthermore Server Models define Message, State and State Transition behavior. Client Models on the other hand do not define any states and define a set of messages, mandatory and optional, used by a Client to

request, change or consume related Server states. Control models are combinations of the above described Models and can therefore contain Client Model functionality for communicating with other Server Models as well as Server Model functionality for communicating with other Client Models. In addition Control Models may contain control logic which consists of a set of rules and behaviors for coordinating interactions between other models to which the said Control Model is connected to. New models can be defined based on existing models, basic models being named Root Models. The concept of the above mentioned models and their relationships to each other is shown in the figure below.



Control Model communication with Client and Server Models [2]

Publish and Subscribe

Devices in a Bluetooth mesh network can send information to multiple devices in the network. The term to use is publishing. Devices can also be configured to follow information from one or more sources. This is called subscribing. A feature called periodic publishing enables sending messages periodically regardless of whether the state has changed or not.

Let us consider a case in which a house has a room with two light switches and two lights. Each light needs to be controlled by a switch. In this case the other switch publishes its state to the appropriate light and that light subscribes to the said switch. The other switch-light pair is managed in similar fashion.

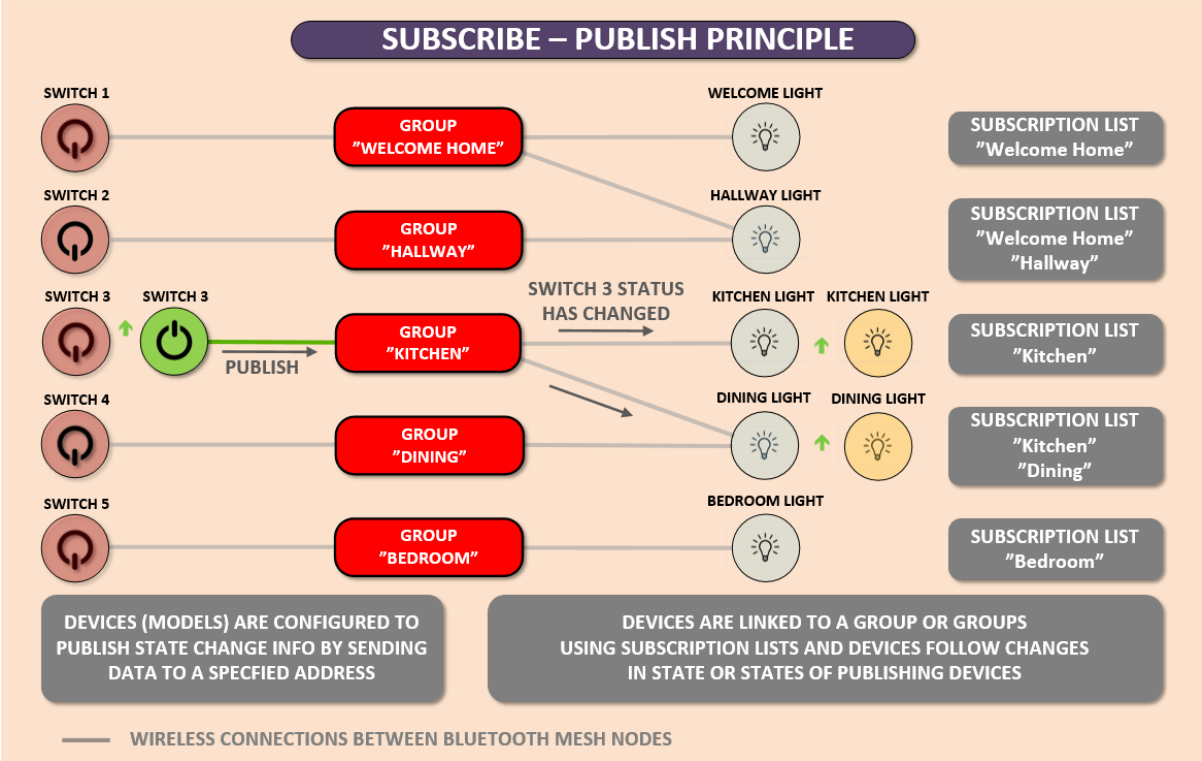
Groups

The previously described case is simple due to the small number of devices. Let us now consider a more complex case in which a house has five light switches and five lights. Some of the lights need to be controlled by just a single switch while several other lights need to be controlled by more than just one switch. This is basically easy to set up but what if a broken switch needs to be replaced or the configuration needs to be changed? One would need to

configure the new switch with all of the addresses of the lights to be controlled by that switch. Clearly this will become unpractical or even unmanageable if the number of switches and lights increases.

The above mentioned problem can be solved by taking into use the concept of Groups to designate a virtual "switchboard" level. Now switches can be configured to send their state to a Group and also lights can be configured to subscribe to Groups.

Configuration becomes much simpler because now the new replacement switch needs to be configured to publish data only to a single Group instead of a multitude of lights. The figure below depicts the described example and the Publish-Subscribe principle using Groups. Messages generated by devices, or more specifically, elements within nodes, send their messages to mesh addresses (unicast, group, or broadcast address) from a single unicast address. These messages are sequenced uniquely to help protect against replay attacks. Acknowledged messages will cause a response message while unacknowledged will not.



Subscribe-Publish principle and the use of Groups

Provisioning

Devices which are not yet part of any Bluetooth mesh network are called unprovisioned devices. Unprovisioned devices advertise their presence and can be added to the mesh to become nodes of the network by a process called provisioning. During provisioning the Provisioner, which would typically be a mobile device, provides the node with data consisting of the Network Key, the current IV Index and a unicast address for each element in the node.

The use of multiple Provisioners is allowed but the specifics of cached data sharing etc. need to be defined in the implementation. During provisioning network resources are managed and allocated to nodes by a Provisioner. The Provisioner also allocates node addresses making sure there will be no duplicates of unicast addresses. Device keys are known only to the device itself and the Provisioner and they are used only during configuration.

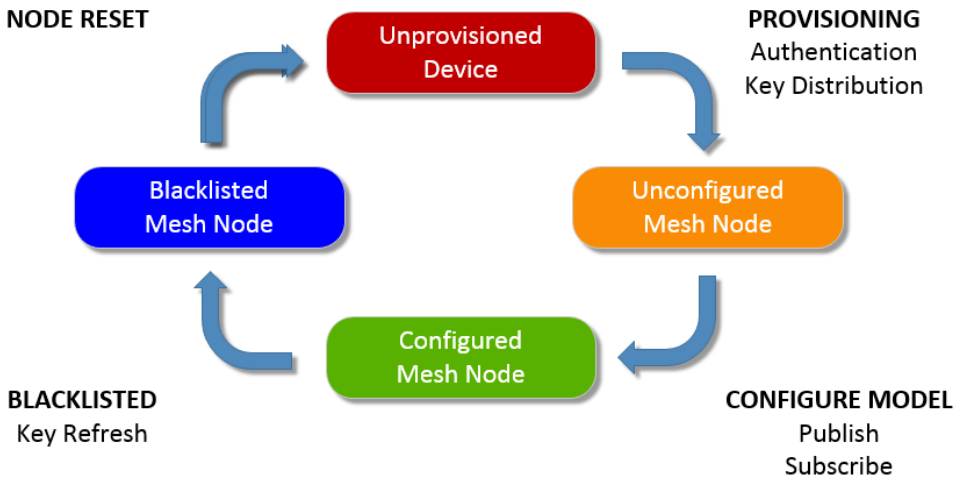
Device Authentication using OOB

For secure authentication during provisioning information must be passed between devices without using the actual Bluetooth RF channels. This is referred to as Out-of-Band (OOB) authentication. Bluetooth mesh defines three types of OOB authentication methods, listed in the table below.

Method	Description
Input	Provisioner outputs a value – user inputs the value into the device
Output	Device outputs a value – user inputs the value into the provisioner
Out-of-band	Device communicates the value by non-Bluetooth means such as NFC

Lifecycle

A Bluetooth mesh device lifecycle model is shown in the figure below. First the Provisioner must detect an unprovisioned device and establish a provisioning bearer. Then the Provisioner and the device use the Elliptic Curve Diffie-Hellman (ECDH) anonymous key agreement protocol to establish a shared secret. After this the device needs to be authenticated. Using Out-of-Band (OOB) information like a public key of the device, a long secret or a value which is output or input using the device is strongly recommended. Once authentication has succeeded both the Provisioner and the device together will generate a session key based based on the shared secret. The session key is then used to encrypt and authenticate provisioning data.



Bluetooth mesh device lifecycle

All devices have a configuration block which includes information such as Company ID, Product ID and Model information. The Provisioner needs to read the data contained in the configuration block after which the Provisioner can proceed with the configuration according to the capabilities and Models present on the device. These keys also need to be bound to the appropriate Network Keys. Finally, the Provisioner needs to configure how the node publishes information and which information it needs to subscribe to.

If a node is removed from the mesh the keys of all remaining nodes need to be changed to prevent the possibility of a “trash-can attack”. Nodes which may have been compromised can be blacklisted to prevent their use in the mesh. Network Keys, Application Keys and all confidential data derived from them can be refreshed using a Key Refresh procedure.

States and Scenes

States are values representing conditions of node elements. When an element is exposing a state (value) representing the condition of the element that element is called a Server and implements the Server Model. When accessing a state of an node element the accessing node is called a Client, which implements the Client Model. There are three different methods for controlling element states.

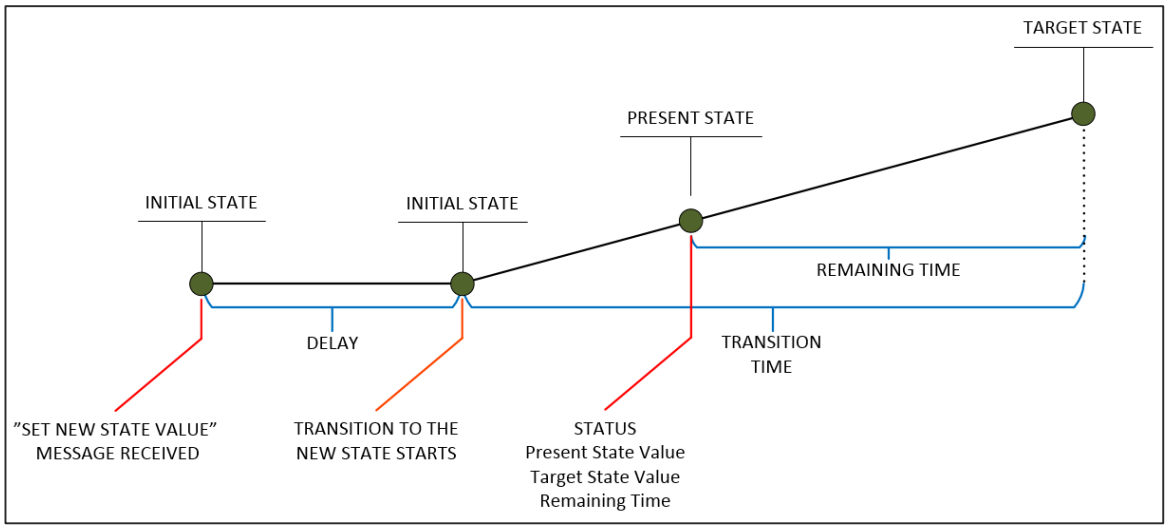
The first is a state-changing message which is sent to a Server which then processes it. The second is an asynchronous event e.g. a scheduler action which is executed. The third is a local event e.g. the pressing of a button physically located on a lamp which is part of the mesh network. Bluetooth mesh devices publish state (value) changes to other nodes but devices can subscribe to only certain interesting bits of information. Publishing can be directed to a single group while subscription can be set up to follow multiple groups.

To store states of elements Bluetooth mesh provides Scenes. A Scene Register is a 17-element, zero-based, indexed array of 16-bit values, the indexes serving as handles to state storage containers in which the associated state information is stored.

Initial, Present and Target States

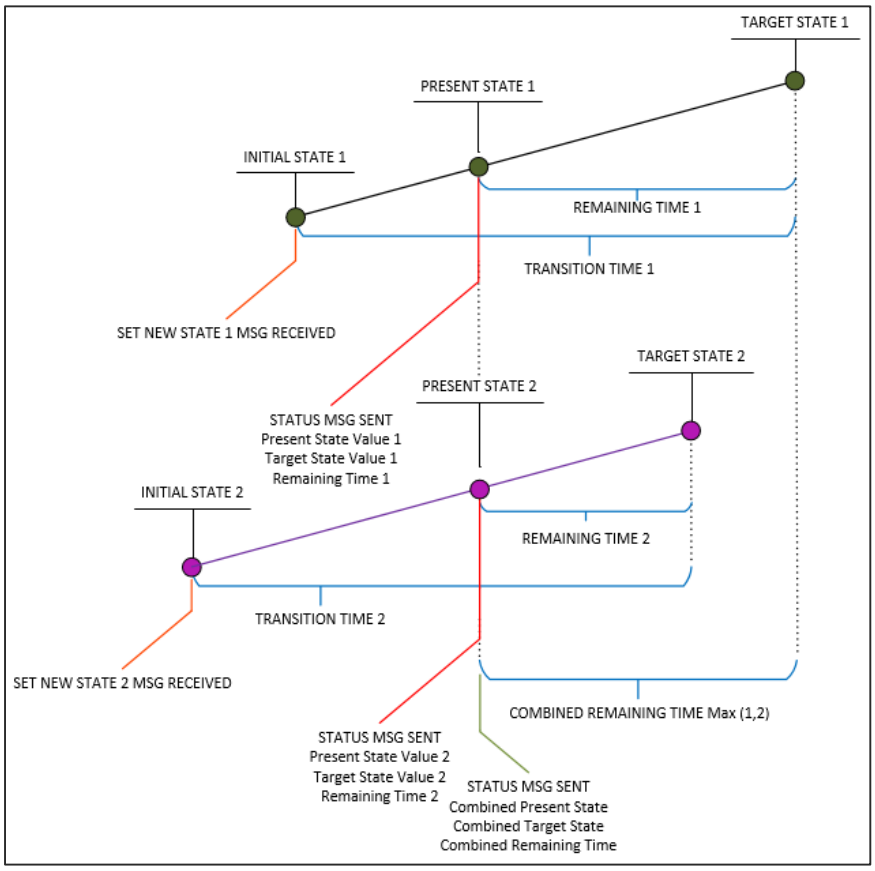
States can change instantaneously or within a certain time period, this time period being called Transition Time. The change begins from an Initial State ending to the Target State. Present State is the current state of an element. Bluetooth mesh defines a very flexible method for defining how states can be changed using several different parameters and has a selection of specially optimized commands for various applications such as lighting control, sensors etc. The user can utilize a default transition time or alternatively define the time by changing the appropriate field in the related message. One can also add a Delay which defines the length of time between the receipt of the message and the actual start of the state transition. This is useful for synchronizing actions with multiple receivers.

With some states messages can be used to report the state of the transition, either the present state, or including both the present and target states together with the remaining time when the target state will be reached. The figure below shows the relation of these concepts in a one-dimensional case.



State change using delay and transition time with related status info

States can be one-dimensional or multi-dimensional. For example, in light control applications one might be interested in controlling the hue, saturation and lightness. The figure below demonstrates a multi-dimensional case.



Example of a two-dimensional state

Bound States

States can also be bound together in which case a change in any of the states bound to each other will result in a change in the other states bound together. An example from lighting applications would be a case in which the light is dimmed to zero which will then cause the OnOff state to change to off-state too. It should be noted that bound states do not need to be from the same model nor do they have to be related to the same element. Binding can be defined as bi-directional or uni-directional. This gives a lot of freedom for creative application development.

Composite States

To make the detection of changes in states as flexible as possible the user can group together multiple states. These kind of groups are called Composite States. As an example, consider the Light Lightness state, which is actually composed of three states name the Light Lightness Actual State, the Light Lightness Last State and the Light Lightness Default State. If an indication of a change in any of the listed three states is required one can just refer to the Light Lightness state instead of referring to all three states separately.

Scenes

Bluetooth mesh includes a useful feature called Scene with which it is possible to recall a set of States for a group of Nodes. A practical example for a home application utilizing Scenes would be a light control by which a parent can control the lights in all of the children's bedrooms to the desired level. For example, in the evening the user could set

all the lights in the bedrooms off except for a dimmed comfort light which slowly dims and finally switches off completely.

Messaging

All communication in a Bluetooth mesh network is based on sending messages which operate on states. All states have a defined set of messages supported by a Server. Clients can use these same messages for requesting the value of state or to change the value of a state. Information on states and/or the changing of states may be sent by a Server without any external request. Messages consist of an opcode and related parameters. Single octet size messages are used with special messages when maximum payload size is required, two octet messages are standard messages and three octet messages are reserved for vendor-specific messages.

Messages are either acknowledged or unacknowledged. Unacknowledged messages require a response whereas unacknowledged messages do not. Set, Clear, Recall, and Store messages can be either acknowledged or unacknowledged, which means that even though the semantics are the same their opcodes are different.

Broadcast messages to all nodes listening at any time are based on the Advertising Bearer using LE non-connectable advertising. Messages are sent using broadcast channels. Point-to-point messages are based on the GATT Bearer using LE connections and standard GATT service. Any node may support either or both bearers.

Segmentation and Reassembly

The Transport Layer defines the total message size. Segmentation and Reassembly (SAR) can be used. In an optimal case only single segments are used. With SAR a maximum of 32 segments is the limit which translates to 384 octets. With a single octet opcode this means that 379 octets are available for parameters, and with two or three octets 378 or 377 octets correspondingly. Bluetooth mesh messages are sent inside models using opcodes and element addresses.

With large PDU's exceeding the maximum size the Lower Transport Layer of the transmitting node takes the PDU from the Upper Transport Layer and segments it by splitting the PDU into several Transport PDU's. When the receiving node receives these PDU's the Lower Transport Layer reassembles the original PDU and then passes it upward to the Upper Transport Layer in the receiving node.

Segments are identified by using a Segment Offset Number (SegO) where the first segment has the value 0, the second has the value 1 and so on. The offset of the last segment needs to be also defined with the Last Segment Offset Number (SegN).

For example, if the message consists of four segments the value to use for SegN would be 3. In addition each message is identified by using a Sequence Authentication (SeqAuth) value which is used to encrypt or authenticate the access message. The normal procedure would be to use the Sequence Number (SEQ) of segment number 0.

There are four different message types defined for segmentation and reassembly (control and access segmented, control and access unsegmented), listed in the table on next page. With segmented messages Bluetooth mesh defines a Segment Acknowledgement procedure which enables retransmission of segments which have not been received during the first try. The mechanism allows indication of missed segments to make the source send only missed segments again. In case the receiving node is receiving messages on behalf of a defined Low Power node in a Friendship setup an OBO (On Behalf Of) acknowledgement message is used. The Friend node which buffers the received message replies with the OBO on behalf of the Low Power node.

SEGMENTATION TYPE	MESSAGE TYPE	
	Control	Access
Unsegmented	Unsegmented Control Message	Unsegmented Access Message
Segmented	Segmented Control Message	Segmented Access Message

Addressing

A multitude of IoT use cases require that the user is able to control a single node, a group of nodes or all of the nodes simultaneously. This is provided in Bluetooth mesh by using unicast, virtual and group addressing. A special value representing an unassigned address not used in messages is also available. An element is an addressable entity within a node and each node has at least one element called the Primary Element. In addition to the Primary Element a node may have one or more additional elements. This characteristics is stable throughout the lifetime of the node.

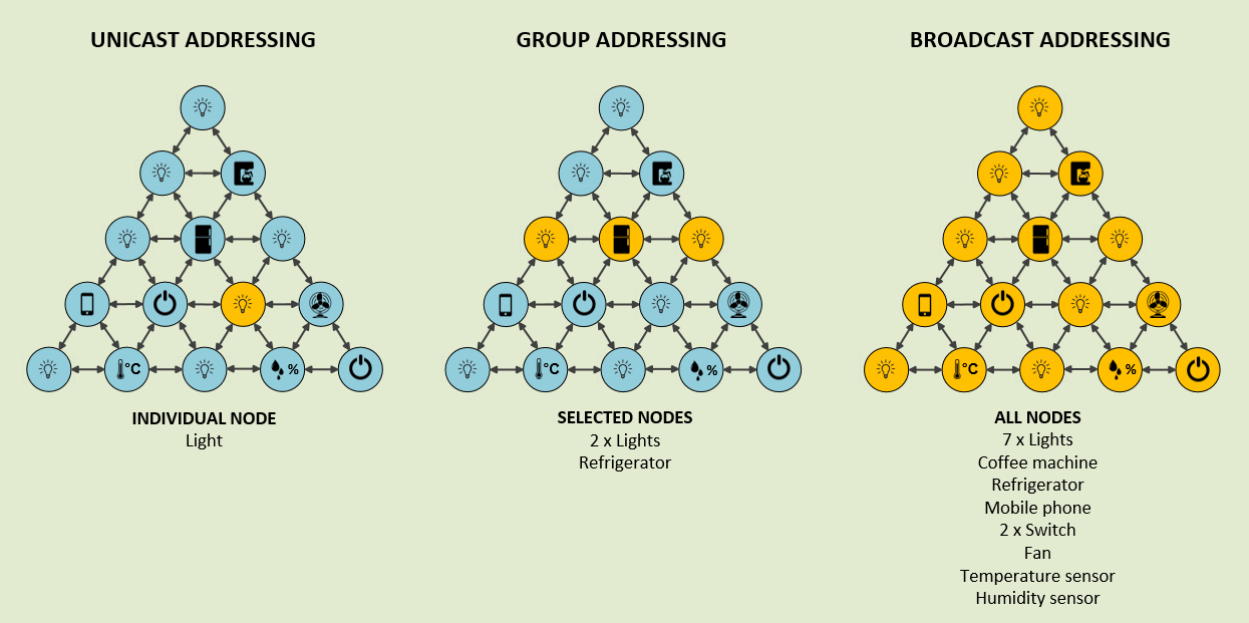
Unicast Addresses are allocated to Elements and they represent a single Element in a Node. A Bluetooth mesh network allows a maximum of 32767 unicast addresses. Virtual addresses are used for multicasting and they can represent multiple elements in one or more nodes.

A virtual address is a hash of a 128-bit Label UUID, meaning that the amount of available virtual addresses is very large even if they are represented as 14-bit quantities in network PDU's.

Group addresses are also used for multicasting and may represent several Elements in one or more nodes. A mesh network offers 16384 group addresses. The Bluetooth mesh standard defines a set of fixed group addresses which can be used to address a subset of all primary Elements of nodes of certain functionality. The rest are dynamically assigned group addresses and the user has 256 fixed group addresses and 16128 dynamically assignable group addresses to use.

Since each node in the mesh may contain one or more elements the Provisioner allocates each element with a unique unicast address. In addition to the unicast addresses for each element the Provisioner also allocates group addresses.

The principal difference between the above listed addressing options are shown in the figure on next page. It is important to note that Bluetooth mesh allows grouping of all kinds of devices to allow flexible grouping of mesh devices.



Bluetooth mesh enables addressing a single node, a group of nodes or all nodes by using Unicast, Group or Broadcast addressing respectively.

Security

Bluetooth SIG has taken security concerns seriously and made security a mandatory feature of Bluetooth mesh. All traffic is encrypted and sending of unencrypted messages is prohibited. This fact alone makes the security model used in Bluetooth mesh stronger compared to BR/EDR and LE.

Encrypted Layers and Multiple Keys

Achieving a high level of security is based on encryption and authentication both at network and transport layers. Message Integrity Check (MIC) is applied to traffic of both layers. Furthermore, all mesh messages are encrypted and authenticated using two different keys. Network traffic and application data traffic are both protected using AES-128 CCM. Application specific keys provide effective isolation of applications and application data.

Authentication and confidentiality of all data in Bluetooth mesh is provided by using three different keys which are the Device Key, the Network Key and the Application Key. There are also some other keys derived from these three main keys which as a group are called Derived Keys.

Device Key

Each individual node of a mesh network has its own unique Device Key. The value of the Device Key is calculated with the shared secret provided by the Elliptic Curve Diffie-Hellman key agreement between the two devices and is never transmitted over the air. In theory only the Provisioner has knowledge of the Device Key except if the Device Key has been handed over to another device for configuration purposes.

The Device Key is used to authenticate and encrypt communications between the Configuration Client and a single node. Other nodes have no way of interpreting communication between the said two devices. The Device Key is the most critical key in the mesh network since the Device Key is used to read information about the node, enable configuring how the node will communicate with other nodes in the mesh and in the distribution of security credentials.

Network Key

Mesh networking flexibility is provided by the fact that any node can be configured to use just one or multiple Network Keys simultaneously. This enables the use of one or more Subnets through the definition of Subnet Network Keys. Subnets are groups of nodes able to communicate with each other at the Network Layer. Alternatively a node may belong to two or more totally separate mesh networks. A good example of such a case would be a mobile phone which could be part of a home mesh network, an office mesh network and a vehicle mesh network.

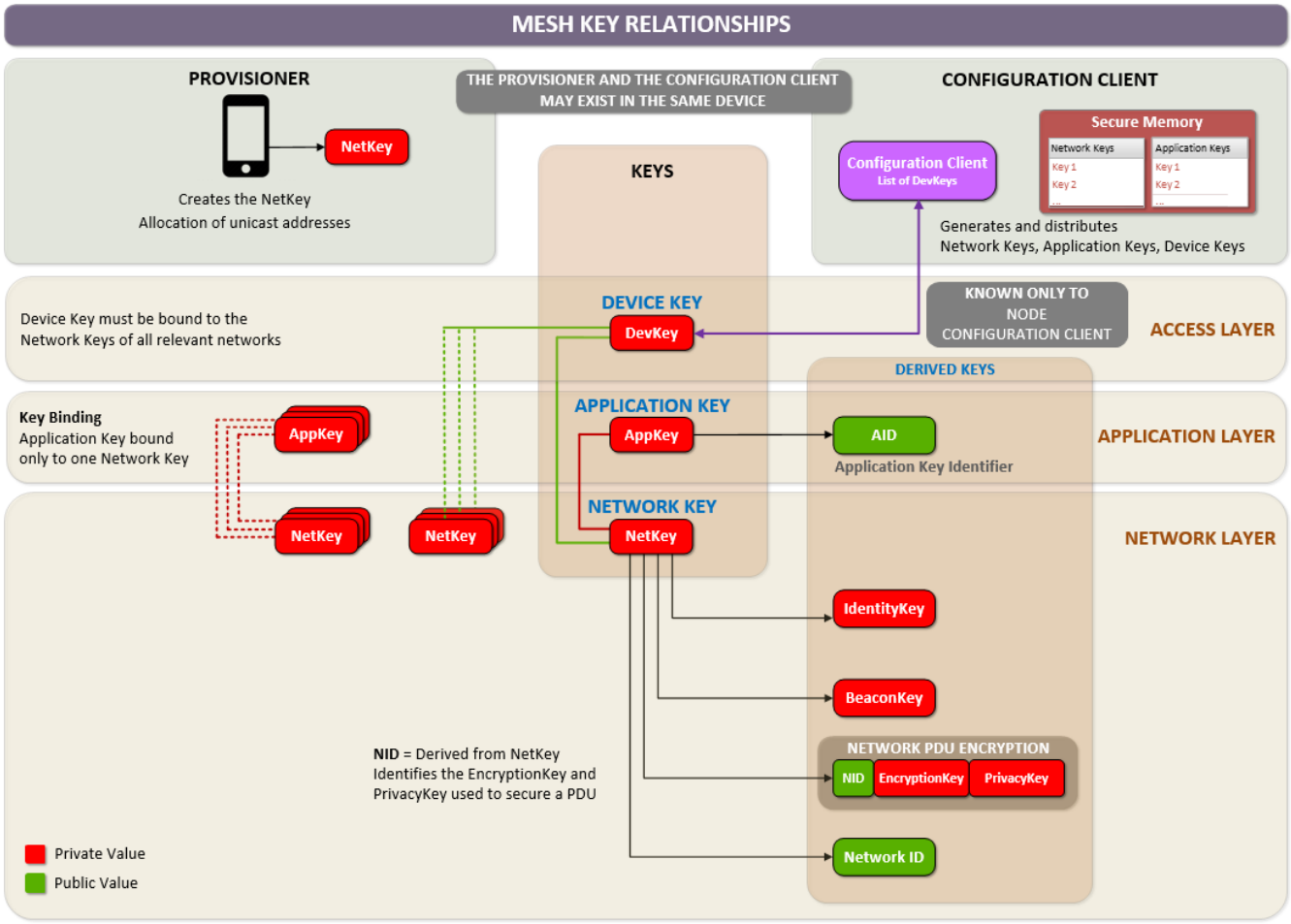
The maximum number of different Network Keys in any installation is 4096 which makes it possible to create secure partitions inside a network when required. There is no direct use for the Network Key. Instead, it is used to derive encryption and other keys using AES-CMAC hashing.

Application Key

A mesh network can contain one or more application domains each with a unique Application Key. Nodes which are configured with a particular Application Key can receive and transmit messages related to the said application while messages originating from nodes part of different application domains are just relayed. The above solution allows compartmentalization of applications which increases security. As an example consider a home with a wireless doorbell and an electrically controllable door lock. Having separate application domains for these two devices increases security since a compromised doorbell would still not make it possible to hack the door lock. One can think of application domains as virtual security areas which belong to a larger security area provided by the mesh network itself.

Derived Keys

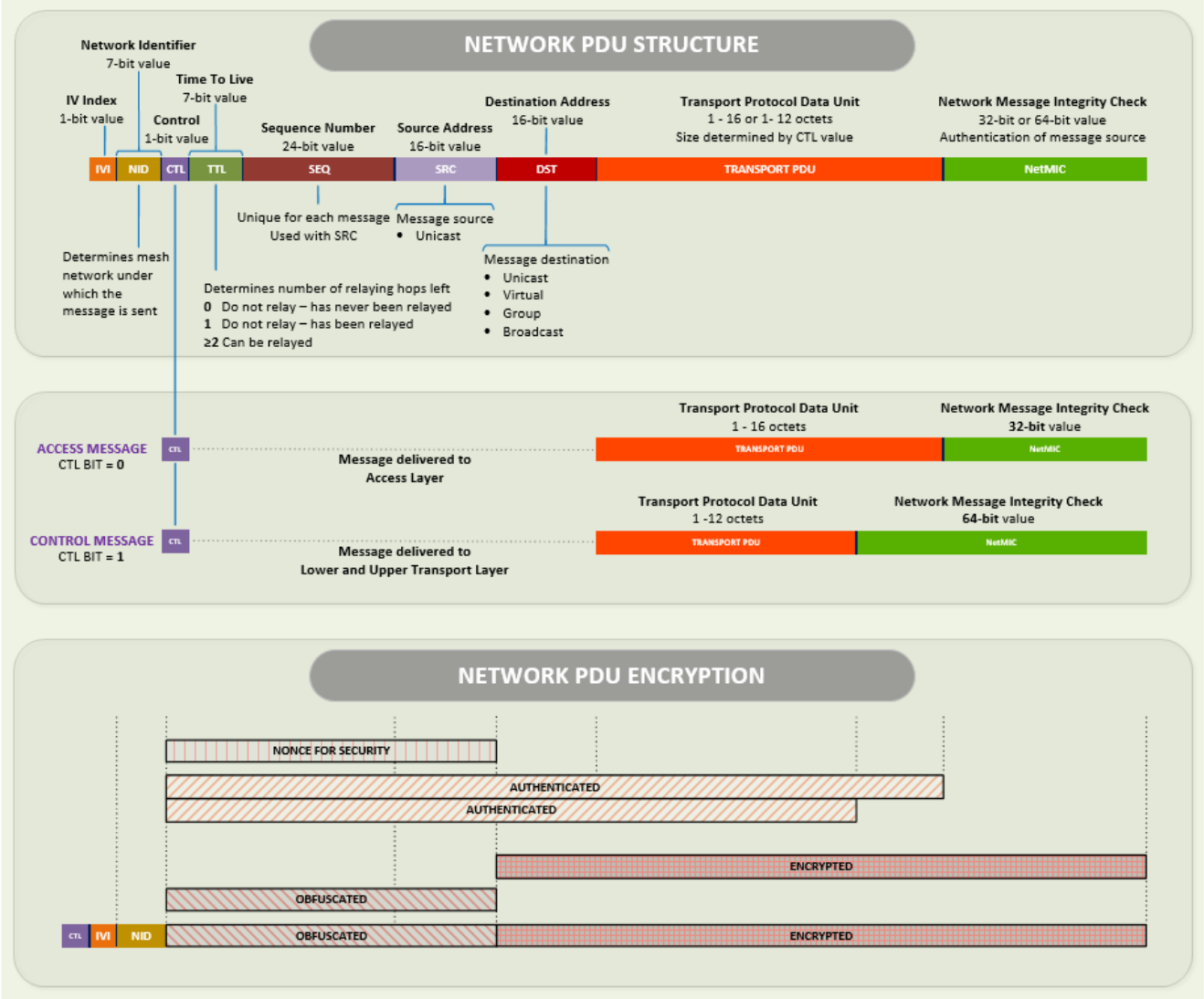
Due to the nature of the Device Key and the fact that it is not used directly requires that we have some other keys to use. These are established from the Device Key using a Key Derivation Function (KDF) as defined by the Bluetooth SIG. The idea here is to protect the Device Key even though a Derived Key is compromised for any reason. The relationship of Derived Keys with the three basic keys is shown in the figure on next page.



Bluetooth mesh keys and derived keys

Network PDU Obfuscation and Encryption

The only plaintext parts of the Bluetooth mesh PDU are the IVI Index (IVI) and the Network Identifier (NID) at the start of the PDU package. The rest of the PDU is obfuscated using AES-ECB or encrypted using AES-CCM. This is to make passive snooping of the network structure next to impossible since the header, TTL, sequence number, source address etc. are not sent as plain text. Nonce data is generated from sequence numbers and other network header information while message integrity is protected by a 32-bit or 64-bit MIC. To keep track of individual messages each message has a unique 24-bit Sequence Number. Replay protection is achieved by using message sequence numbering which the nodes keep track of based on source address. The figure on next page shows the structure of the network PDU and clarifies the principle of obfuscation and encryption applied to it.



Bluetooth mesh network PDU structure and encryption

Key Management

The network, device keys, addresses, human friendly names related to nodes and groups are all saved into a Provisioning Database. This solution enables the use of more than one Provisioner which is useful especially in large mesh networks. Individual databases must then be compiled into a single database at a suitable time.

Key Refresh and Blacklisting

To ensure security a Bluetooth mesh network needs to have its keys refreshed periodically. Also, whenever a node is removed from a Bluetooth mesh network intentionally, or should a network node become compromised for any reason, the keys need to be refreshed too. Bluetooth mesh provides a Key Refresh procedure for these purposes. The Provisioner/Configuration client can utilize this feature to decrease the likelihood of security problems by performing a key refresh of all known acceptable devices on a regular basis such as once a week.

Minimizing Security Threats

Bluetooth SIG has designed the security features of Bluetooth mesh to provide protection against many different types of security threats and attack methods. A summary of identified security threat types with related defense methods are summarized in the table on the following page.

Attack type	Defence method
Replay	Increasing message sequence number SEQ IV Index values within messages from a given element must always be equal to or greater than the last valid message from that element.
Bit Flipping	64-bit Message Integrity Check 32-bit Network layer / 32-bit Application layer
Eavesdropping	Encryption based on AES-CCM Encryption at Application layer Encryption at Network layer
Man-in-the Middle	Out-of-band authentication
Brute Force	Refresh of Network Key Refresh of Application Key
Physically Insecure Device	Complete separation between the network, subnetworks and application layer security
Trash-Can	Blacklisting and key refresh
Guest Access	Separate network and application keys for guests without key refresh option Limited lifetime
Privacy	Privacy built in into foundation of mesh nodes Obfuscation of identifying information on mesh payload Bluetooth address inclusion in advertising packets no longer required

Identified Security Threats and Related Defense Methods [5]

Practical Security Guidelines

Bluetooth mesh provides various means to maximize security. Generally speaking applications should be configured with individual Application Keys because this will limit problems if a node is compromised. Pairwise Application Keys can be configured between nodes which need to exchange data only between each other.

Device Keys represent the most critical information in the network. If a mobile phone is used as a Provisioner the Device Keys as well as Network Keys and Application Keys should be stored using secure storage API's available through the operating system of the mobile device. A lost mobile phone would then not present risks regarding the security of the mesh network. For embedded devices security credentials should be stored in secure memory elements.

In device provisioning using out-of-band input or output the user should use at the minimum a 6-octet numeric or alphanumeric value but the longer the value the stronger the security. Alternatively a device may have other non-Bluetooth data transfer methods available.

When devices are removed from the mesh the Provisioner should initiate a Key Refresh procedure throughout the whole mesh network for both the network and the Application Key. The removed device should also be blacklisted. There is also a need to refresh the above mentioned keys periodically to help maintain the strength of the obfuscation. Bluetooth SIG recommends this to be done every 14 days. In similar fashion a periodic refresh of the IV Index keeps the nonces applied to encryption and authentication fresh. Bluetooth SIG recommends the same 14 day period for IV Index refreshing. Programming fixed Network Keys during device manufacturing is strongly discouraged since refreshing as recommended above would not be possible.

Finally, since the Provisioning Database contains very sensitive information regarding the safety of a mesh network special attention to the transfer of the database is needed. Authentication using at the minimum 128-bit MIC is recommended. In addition using OOB methods will ensure that only intended devices can utilize the database.

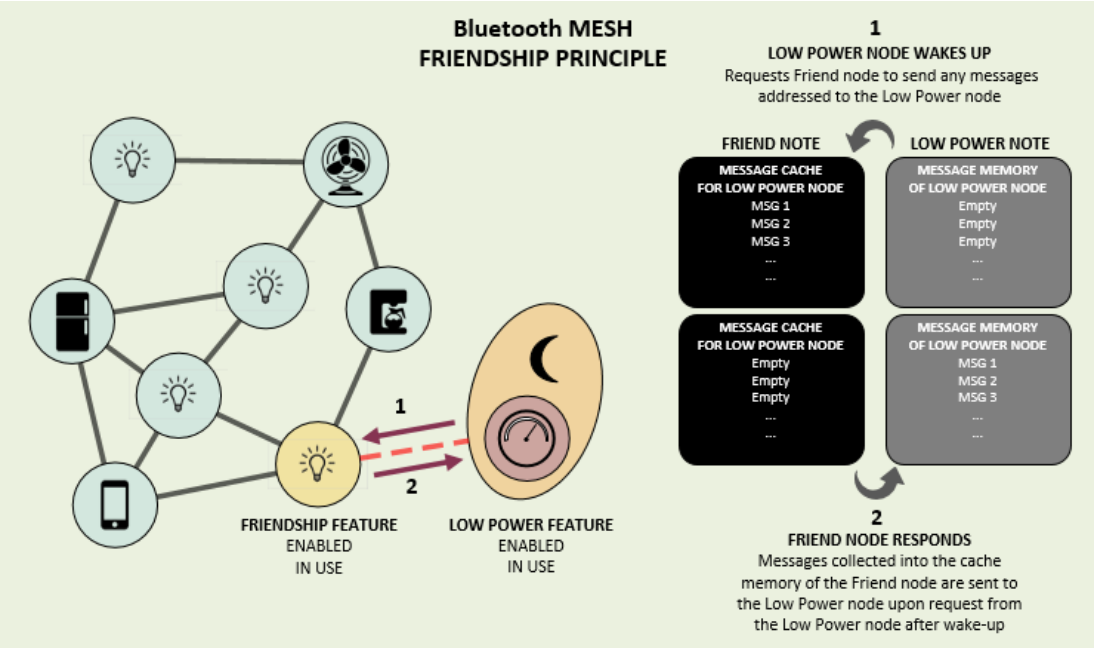
Low Power Support

One important detail in most mesh networks is how the network caters for very low power devices. For example sensors may be in deep sleep most of the time while gathering and saving data with the wireless radio circuitry switched off to save battery power. How can the user be sure that messages addressed to dormant nodes will eventually reach the node? In Bluetooth mesh this is managed by using the so called Friendship feature. A node which supports the Low Power feature must the feature enabled and furthermore must establish a friendship relationship with a node supporting the Friendship feature.

The Low Power feature reduces receiver duty cycles by allowing the Friend node to store messages intended to the Low Power node. When the Low Power node wakes up it polls the Friend node for any messages stored on its behalf and can receive them "batch style". The Friendship node will need to be of the "Always on" type so that is can receive messages addressed to the Low Power node at any time.

Friendship

Friendship is based on polling and there are some timing parameters defined at the setup of a Friendship connection which then are static until the connection is finally (if ever) cleared. It should be noted that the Friendship device's power consumption may increase when messages are received, acknowledged and stored and finally sent to the Low Power node. The relationship between a node with Low Power feature enabled and in use with a node with the Friendship feature enabled is shown in the figure below.



Relationship between a Low Power feature node and a Friend node.

Sensors and Multisensors

Sensors play an important part in almost all IoT systems and thus also Bluetooth mesh contains a versatile set of features available for sensors. Bluetooth mesh specification defines the *Sensor State* as a composite of four states which are the *Sensor Descriptor*, *Sensor Setting*, *Sensor Cadence* and *Sensor Data* states. Of these the *Sensor Descriptor* is fixed over the lifetime of the element it defines whereas the *Sensor Setting* and *Sensor Cadence* are configurable as needed. The *Sensor Data* state is a single data point or a column of a series of data points (histogram called *Sensor Series Column* state). *Sensor Data* follows the sensor output changes.

More flexibility is allowed by the fact that the same Model can be presented by multiple instances of sensor states. These states are distinguished from one another with a unique value (*Sensor Property ID*). This way a single sensor may send several measurements such as outside temperature, inside temperature and humidity etc. In such cases the sensor is called a *Multisensor*. The *Sensor Descriptor* state options are listed in more detail the table below.

Field	Size (bits)	Comments
Sensor Property ID	16	Value referencing a property describing the meaning and format of data reported by the sensor
Sensor Positive Tolerance	12	Value representing the magnitude of a possible positive error in the measurements
Sensor Negative Tolerance	12	Value representing the magnitude of a possible negative error in the measurements
Sensor Sampling Function	8	Defines the averaging or sampling function applied on the measured value Options: • *Unspecified* • *Instantaneous* • *Arithmetic mean* • *RMS* • *Maximum* • *Minimum* • *Accumulated* • *Count* • *Reserved for future use*
Sensor Measurement Period	16	Specifies one of the time spans listed below depending on the setting chosen above • *Averaging time span* • *Accumulation time* • *Measurement period*
Sensor Update Interval	16	Defines the internally refreshed measurement value reported by the sensor

Sensor Property ID options are listed in the following table.

Field	Size (octets)	Comments
Sensor Property ID	2	Same value as Sensor Property ID in the above table.
Sensor Setting Property ID	2	Identifies the property of a setting including format and representation of the Sensor Setting Raw field
Sensor Setting Access	1	Read/Write access rights for the setting
Sensor Setting Raw	variable	Raw value of a setting within a sensor

Sensor Cadence

The user can configure the sensor to send measurement values as the value changes either up or down by more than a configurable delta value. This feature actually saves energy since the sensor can be configured to report only when a value changes more than the configured delta value.

A more refined way of controlling how the sensor reports value changes is provided by Sensor Cadence state which controls when and how often the sensor reports using Sensor Status messages. Furthermore the user has the possibility to utilize a state called Sensor Cadence. As the name implies this state can be used to configure the sensor to send updated measurement values more often when the value changes more rapidly and vice versa. Thus the application can then be programmed to note the speed at which the value increases or decreases.

The Sensor Cadence feature provides a particularly useful method for alerting applications to changes in the monitored values only when there is a noteworthy change.

Lighting Control

Bluetooth mesh specification includes the possibility to control different types of light sources very flexibly. The simplest control method is related to turning a light source on or off or dimming a light. For tunable white light sources the color temperature can be controlled by setting the actual color temperature value referenced to the black body curve together with a setting determining the set point distance from the black body curve. In case the light type allows color changing in all three dimensions (hue, saturation and lightness) it is possible to set all three values independently.

The default model for color light control with mesh networks is generally based on the so called HSL (Hue, Saturation, Light) model instead of the RGB model typically used with computer monitors and printers. If desired the HSL color space can be converted to e.g. RGB.

For professional lighting applications colors seen by the human eye are often represented using coordinates (x, y and Y) as defined in the CIE1931 (Commission on Illumination 1931) color chart. The coordinates of the color on the chart are defined by x and y whereas Y defines the luminance. In mesh networks an xyL system is used instead where L corresponds to perceived lightness. To enable accurate “fine-tuning” of a color source all light control states have 16-bit precision.

Other Useful Features

Bluetooth mesh specification includes some other useful features such as Generic Location, Time and Scheduler.

Generic Location

For widely dispersed mesh networks and/or networks with a multitude of sensors and other devices the location of any particular device is often mandatory information required by many applications. Bluetooth mesh provides a generic state called Generic Location, which can be used to define the location of each device accurately. Global Latitude and Global Longitude fields determine the coordinates (WGS84) of the device. Global Altitude determines the altitude of the device above the WGS84 datum.

Furthermore, Local North and Local East fields can be used to describe the location of the device with reference to a local coordinate system on a predefined map with Local Altitude giving the altitude relative to the Generic Location Global Altitude. There is even a Floor Number field which determines the floor on which the device is installed in. For devices which might be moved or move autonomously the Uncertainty field provides a way of defining that the device is either stationary or mobile.

Time and Scheduler

It is beneficial if all nodes in a mesh type network can be aware of time. Knowledge of time is the basis of scheduled and timed actions such as delayed switching or adjustment of certain settings linearly from an initial state to an end

state in the defined time period etc. However, what is also needed is the capability to store and recall device states when needed. The basis of time in Bluetooth mesh is the International Atomic Time (TAI) represented as seconds after 00:00:00 on 2000-01-01. This information is passed on to all nodes in the network as long as one node has the time information.

States can be changed autonomously according to a programmed schedule based on the UTC time and an ISO 8601 calendar. A register provides the means to set the time points at which a change of state is to be carried out.

Performance

Since Bluetooth mesh has been officially released just a short time ago there is not much information yet available on its performance in real applications.

Preventing Mesh Saturation

As mentioned earlier in this paper Bluetooth mesh is based on flooding which is typically simple to implement but presents some issues regarding scalability as the network size and/or amount of traffic increases. In flooding messages injected into the mesh network are potentially forwarded by all relays nodes which receive the message. This may cause infinite retransmissions which will cause the mesh network to saturate. This is why Bluetooth mesh provides two methods, message caching and the Time-to-Live (TTL), to prevent the endless forwarding of messages in the mesh.

The network cache method is designed to prevent forwarding of messages already received by the node. The node compares the received message to already stored messages in the network layer cache of the said node. If the message has already been received the message is discarded. If the received message is new and intended for the receiving node it forwards the message to the higher architectural layers of the software in the node. If not the node simply relays the message forward to other nodes. List size is determined by the implementation.

Each message also includes a Time-to-Live (TTL) number. This number is originally determined by the source node but initially is assumed to be 64 (maximum value is 127). TTL number limits the number of forwards allowed for that particular message. A node which forwards the said message also decrements the TTL value by one. Relay nodes forward only messages which have TTL values larger than 1. Thus the “lifetime” of a forwarded message as it hops along in the network can be effectively limited.

Monitoring Heartbeats

Bluetooth mesh specification includes a useful message type called Heartbeats. Heartbeat messages can be used for checking whether a node is still active and for determining the distance to a node. Heartbeat messages can be configured to be sent periodically a limited number of times or infinitely. The destination address must be configured. Received Heartbeat messages can be counted and the results are indicative of reliability. Heartbeat messages also carry the TTL (Time-to-Live) value so that the number of retransmissions (hops) which can also be used to analyze reliability. Furthermore, this feature makes it possible to tune the TTL value to its optimal setting.

Service Ratio

When discussing mesh networks one of the most important factors determining performance is the service ratio. Service ratio is the ratio of transmitted packets reaching their end destinations compared to the number of all transmitted packets. Another important factor is Application layer packet delay. Adding relays increases performance only to a certain point after which the mesh network will start to congest especially with high traffic densities. Usually the more packets delivered successfully during the first or second hop the better.

One of the bottlenecks in Bluetooth mesh networks is related to the “first hop” i.e. the injection of messages into the mesh in the first place. After a node has been able to inject the message into the mesh the nearby nodes are most likely able to send the message further on through the mesh because there are typically several paths available. One method to help maximize performance is randomized advertising which can improve the service ratio even further.

Conclusion

Perhaps the most important technical points defined by the Bluetooth mesh specification are related to its security related features. Separate network and application level encryption makes it possible to manage overlapping networks and overlapping applications in a logical way while guaranteeing the necessary interoperability between nodes.

Easy configurability of nodes through the subscribe-publish mechanism provides ease-of-use useful in end-user applications. The inclusion of the GATT Bearer making it possible to use older Bluetooth devices as part of Bluetooth mesh networks will certainly speed up the adaptation of Bluetooth mesh technology. Bluetooth mesh is by definition the most secure Bluetooth standard introduced so far and provides efficient security features for IoT applications.

Bluetooth mesh demonstrates the resilience of Bluetooth as a wireless standard by providing a competitive and technically mature solution for many different types of IoT applications. Combined with the robust Bluetooth radio technology Bluetooth mesh is sure to be one of the main technologies the IoT market will utilize.

References

[1] “Mesh Profile Specification 1.0”, July 13th, 2017, Bluetooth SIG

[2] “Mesh Model Specification 1.0”, July 13th, 2017, Bluetooth SIG

[3] “Mesh Device Properties 1.0”, July 13th, 2017, Bluetooth SIG

[4] “Bluetooth Specification 4.2”, Bluetooth SIG

[5] “Bluetooth® mesh networking”, An Introduction for Developers”, Martin Woolley, Sarah Schmidt, Bluetooth SIG