ECEN 5823-001 / -001B

Internet of Things Embedded Firmware

Lecture #22

9 November 2017

Be Boulder.

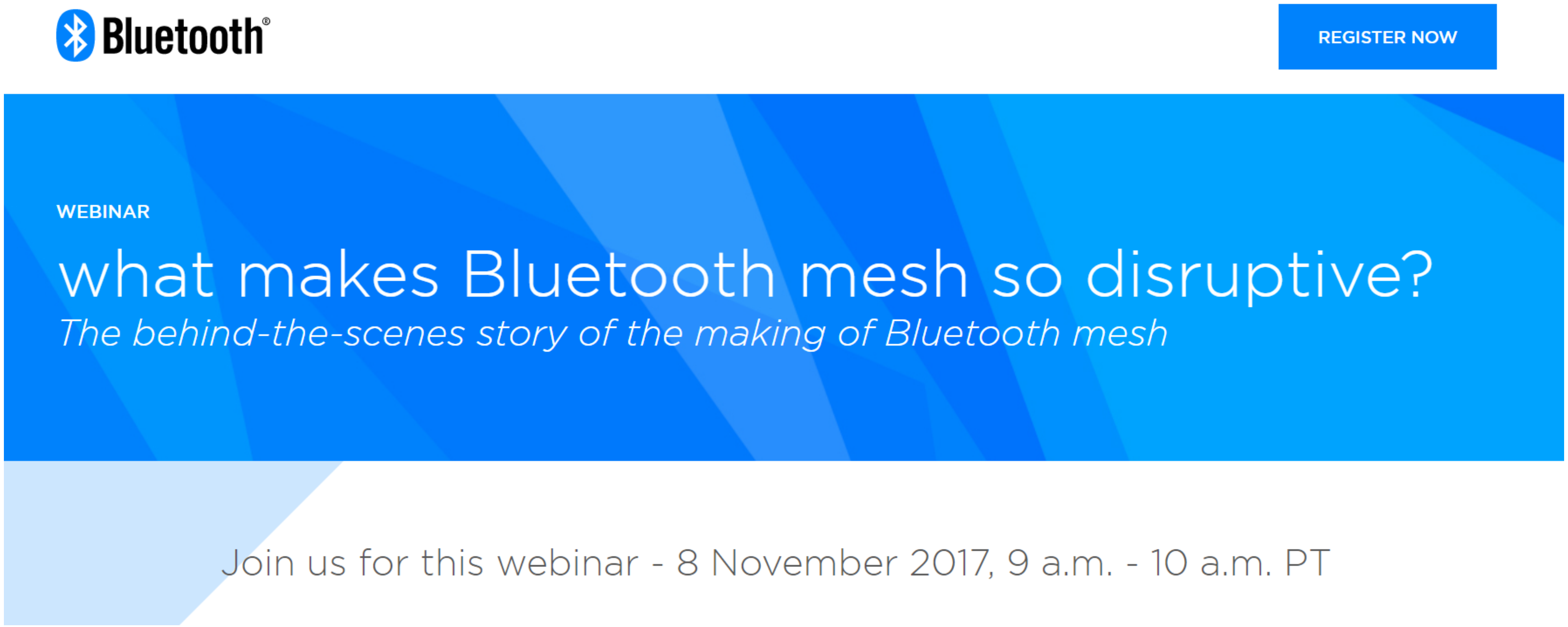
University of Colorado Boulder

Agenda

- Class announcements
- Bonus opportunity #3 – Persistent Data
- Bluetooth Mesh

Class Announcements

- Quiz #8 is due at 11:59pm this Sunday, November 125th, at 11:59pm
- Bonus #2, OTA, is due by 11:59pm this Saturday the 11th!
    - Since it is a bonus, it must be done during a convenient time for the instructing team and do not expect to get a time at 11:59 on the 11th to demo your OTA bonus assignment
    - This will not be used an excuse to request an extension
- Any questions regarding the Course Project?
- Any questions regarding Bluetooth Mesh?
- What to do while the Bluetooth boards are not hear?


Bluetooth®

REGISTER NOW

WEBINAR

what makes Bluetooth mesh so disruptive?

The behind-the-scenes story of the making of Bluetooth mesh

Join us for this webinar - 8 November 2017, 9 a.m. - 10 a.m. PT

http://pages.bluetooth.com/mesh-webinar.html?utm_campaign=mesh&utm_source=homepage-cta&utm_medium=homepage-cta&utm_term=homepage-hero&utm_content=homepage-hero&_ga=2.230830367.1233636139.1509419350-

What is Persistent Data?

- **Persistent Data** denotes information that is infrequently accessed and not likely to be modified. The opposite of this is dynamic **data** (also known as transactional **data**) where information is asynchronously changed as further updates to the information become available. (Wikipedia)
- Our definition is that the data is not modified frequently, and is non-volatile

Why do you require Persistent Data?

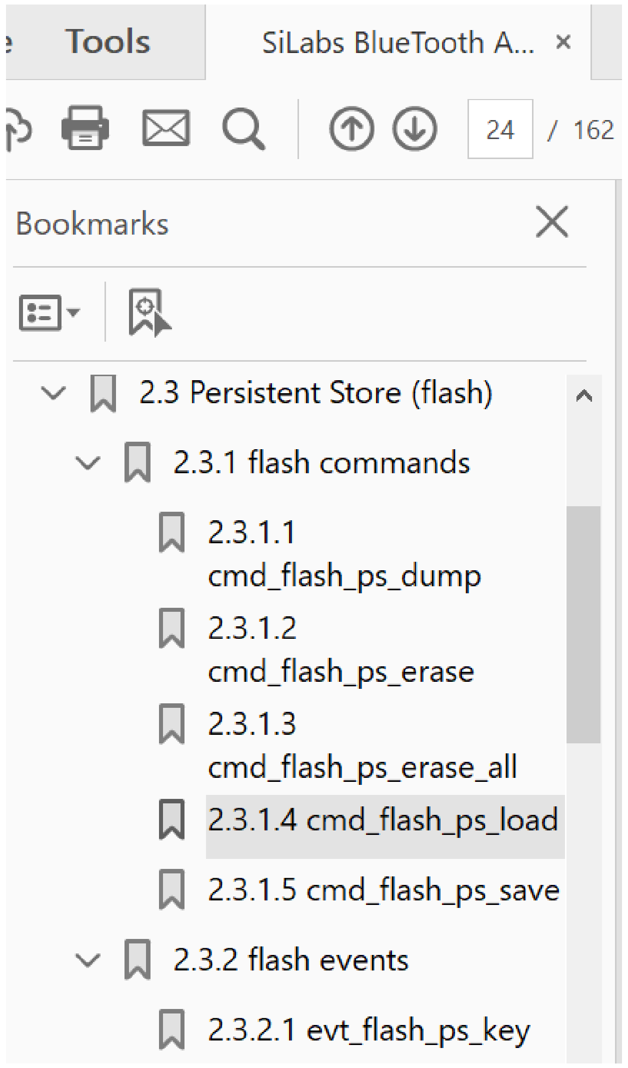
- Remove the need to reconfigure a device upon power loss and reboot
- Examples:
  - Temperature settings on a thermostat
  - Turn on and turn off times of a smart light
  - Critical set points of a sensor alarm
    - Setting the minimum temperature alarm
- Can you protect yourself from a power loss and not require non-volatile memory?
- When could you have loss of power?
  - When the batter is changed

What do I need to consider?

- I have just changed my program to use persistent data
- What happens the first time I boot up and try to read the persistent data?
  - It is not initialized
  - You must be able to recognized that it is not initialized and handle this case
    - Load some default initial value
- What event should you use to initialize your variables from persistent data?
  - gecko_evt_system_boot_id

What is required to program the BG Persistent Memory?

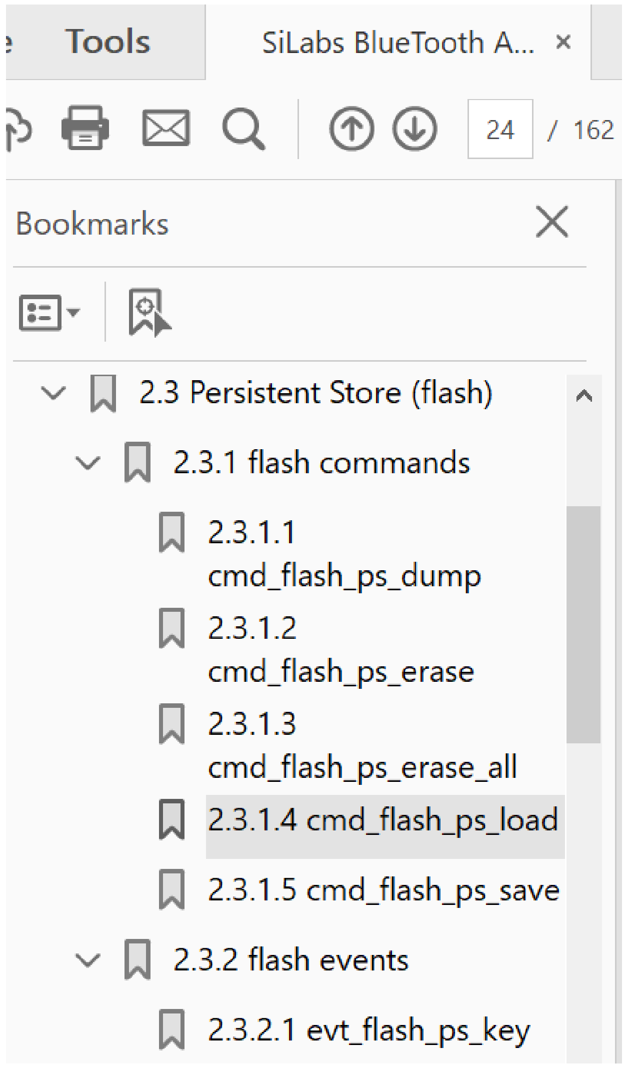
- Go to the Bluetooth API document
- Persistent Store commands can be used to manage the user data in the flash memory of the Bluetooth device
- User data stored in PS keys within the flash memory is persistent across reset and power cycling of the device
- The maximum user data size associated to a PS key is 56 bytes



2.3 Persistent Store (flash)
- 2.3.1 flash commands
  - 2.3.1.1 cmd_flash_ps_dump
  - 2.3.1.2 cmd_flash_ps_erase
  - 2.3.1.3 cmd_flash_ps_erase_all
  - 2.3.1.4 cmd_flash_ps_load
  - 2.3.1.5 cmd_flash_ps_save
- 2.3.2 flash events
  - 2.3.2.1 evt_flash_ps_key

What is required to program the BG Persistent Memory?

- Go to the Bluetooth API document
- Persistent Store commands can be used to manage the user data in the flash memory of the Bluetooth device
- User data stored in PS keys within the flash memory is persistent across reset and power cycling of the device
- The maximum user data size associated to a PS key is 56 bytes

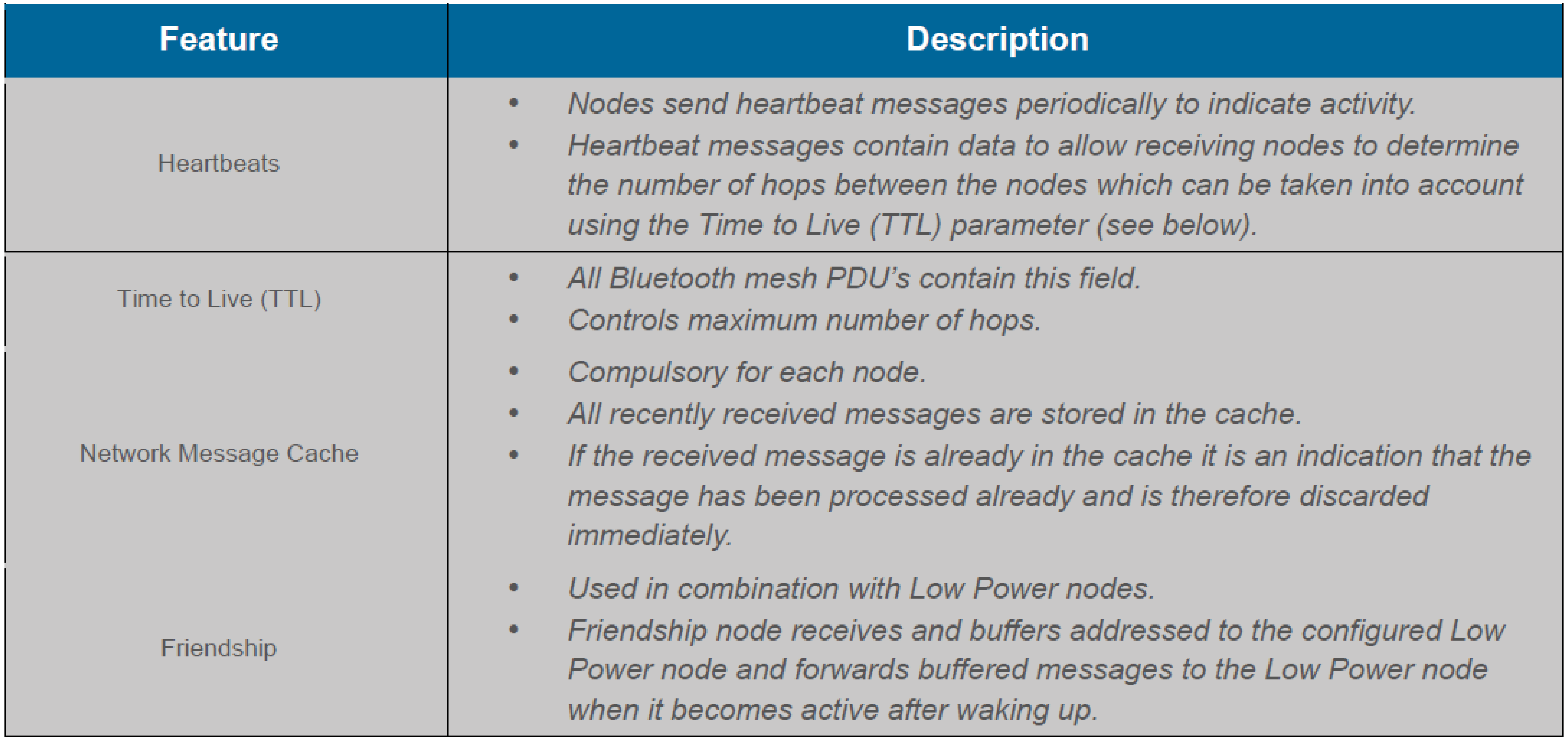


2.3 Persistent Store (flash)
2.3.1 flash commands
2.3.1.1 cmd_flash_ps_dump
2.3.1.2 cmd_flash_ps_erase
2.3.1.3 cmd_flash_ps_erase_all
2.3.1.4 cmd_flash_ps_load
2.3.1.5 cmd_flash_ps_save
2.3.2 flash events
2.3.2.1 evt_flash_ps_key

Course Project – 3rd Bonus Opportunity

- Implementing Persistent Data
- Opportunity is worth 1% towards your final grade!
- You will need to demonstrate persistent data by changing a value, removing power from your development board, applying power back to the board, and demonstrating different functionality based on the persistent data stored without down loading additional firmware
- Requirements to get the extra credit:
  - Must demo to one of the three instructing team members by Saturday, November 18th at 11:59pm
  - By being a bonus, you must demo during convenient instructing team availability and not expect availability at 11:58pm on November 18th
  - For distant students, the demo should be arranged over video chat, skype, etc.

Bluetooth Mesh – Managed Flooding



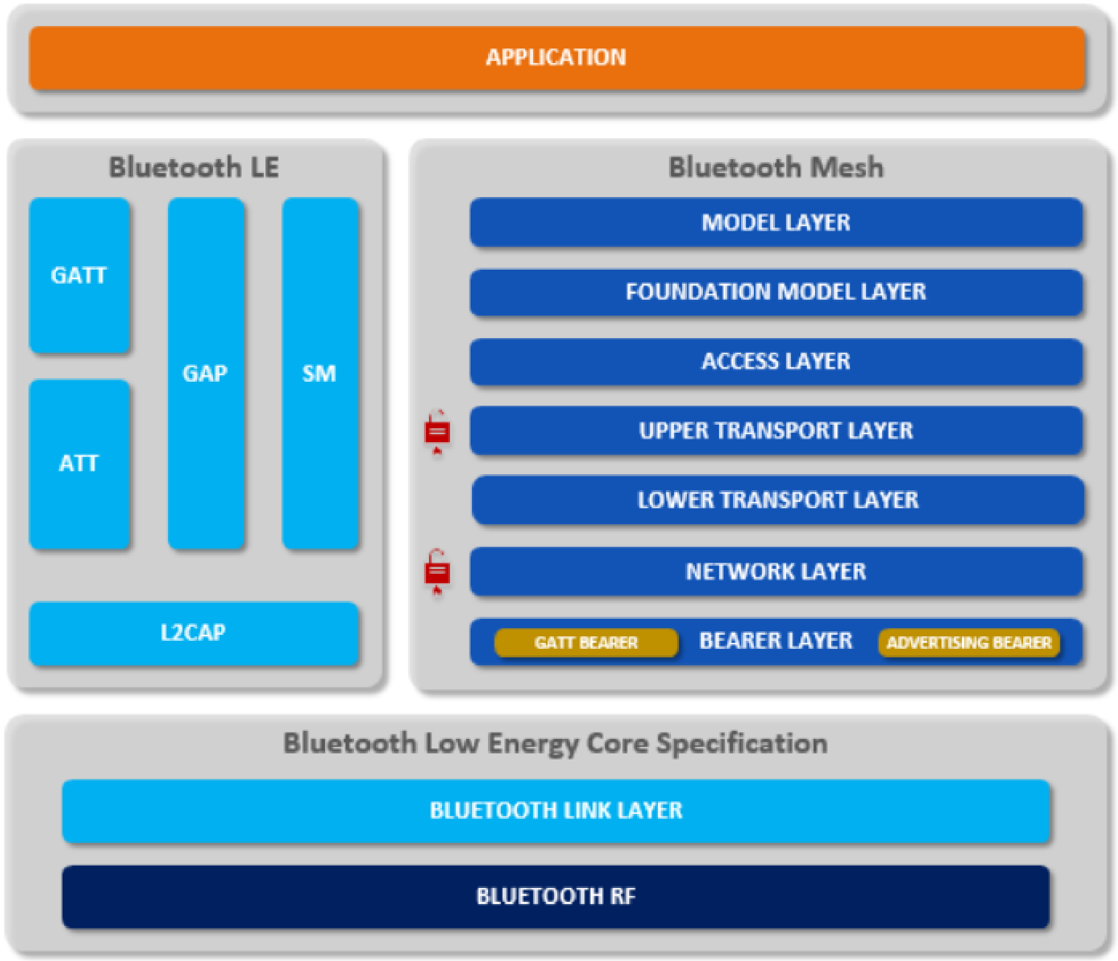
Feature	Description
Heartbeats	• *Nodes send heartbeat messages periodically to indicate activity.* • *Heartbeat messages contain data to allow receiving nodes to determine the number of hops between the nodes which can be taken into account using the Time to Live (TTL) parameter (see below).*
Time to 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.*

Bluetooth Mesh – Managed Flooding

Feature	Description
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.*

Bluetooth Mesh Software Architecture

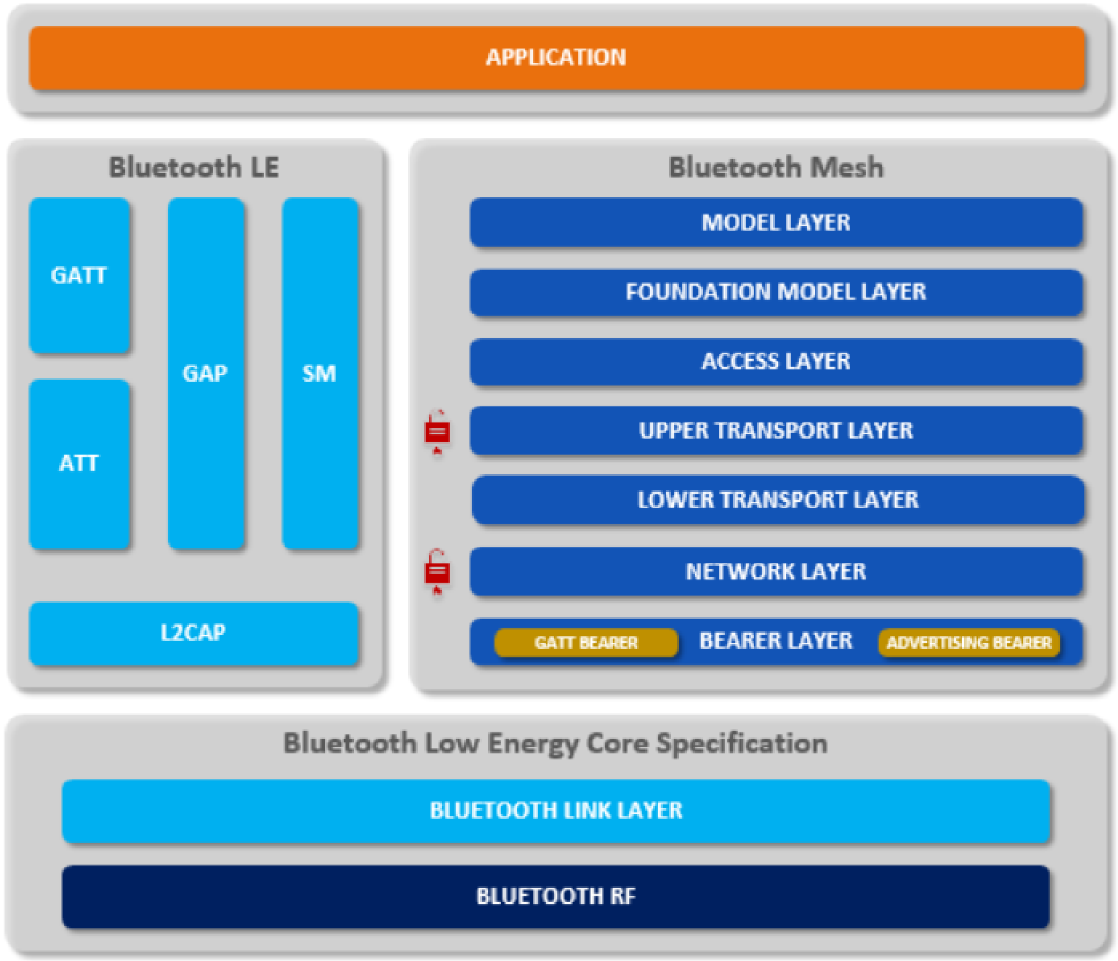
- Seven software layers on top of the Bluetooth Low Energy Core Specifications
- The Model Layer defines models which standardize typical user scenario operations as defined in the Bluetooth Mesh Model specification



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

Bluetooth Mesh Software Architecture

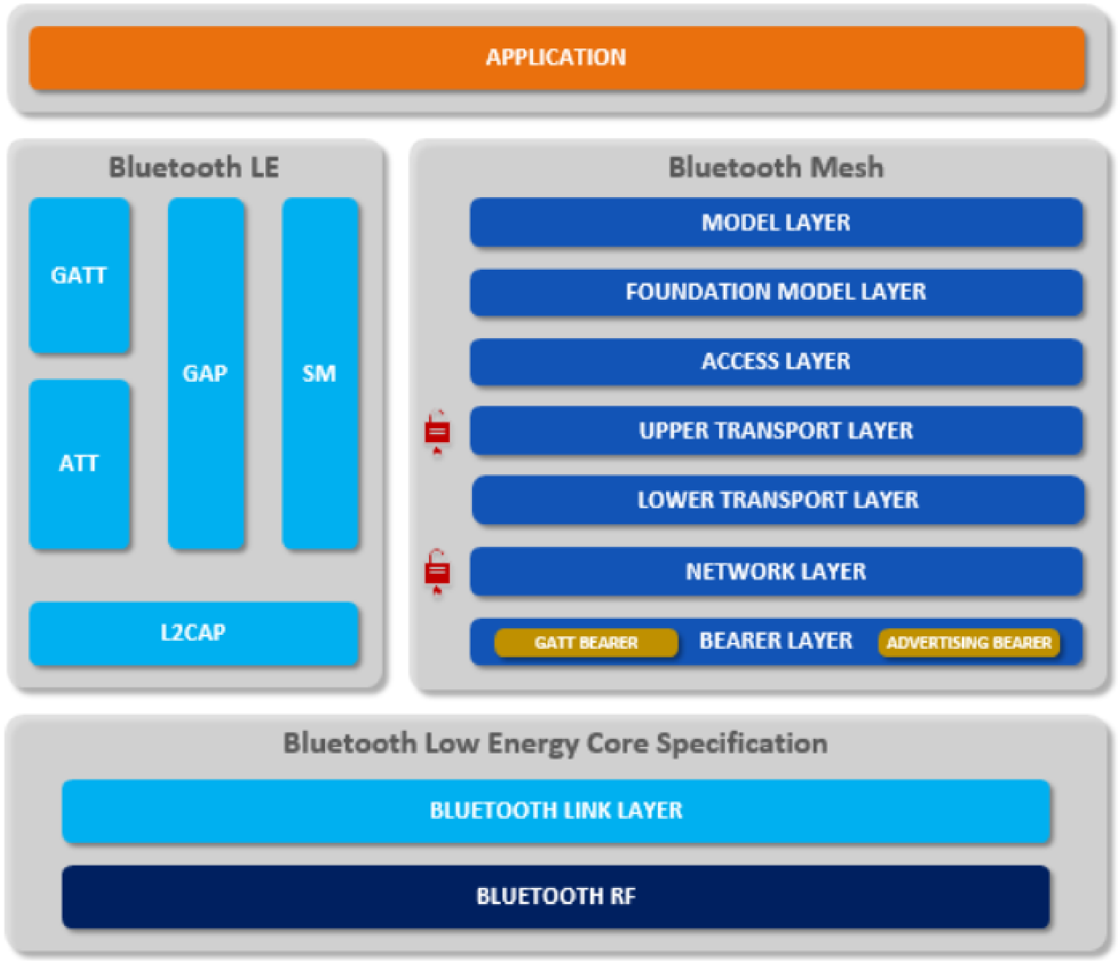
- The Foundation Model Layer defines the states, messages and models needed for configuring and managing a Bluetooth mesh network
- Higher Layer application usage of the Upper Transport Layer is defined in the Access Layer



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

Bluetooth Mesh Software Architecture

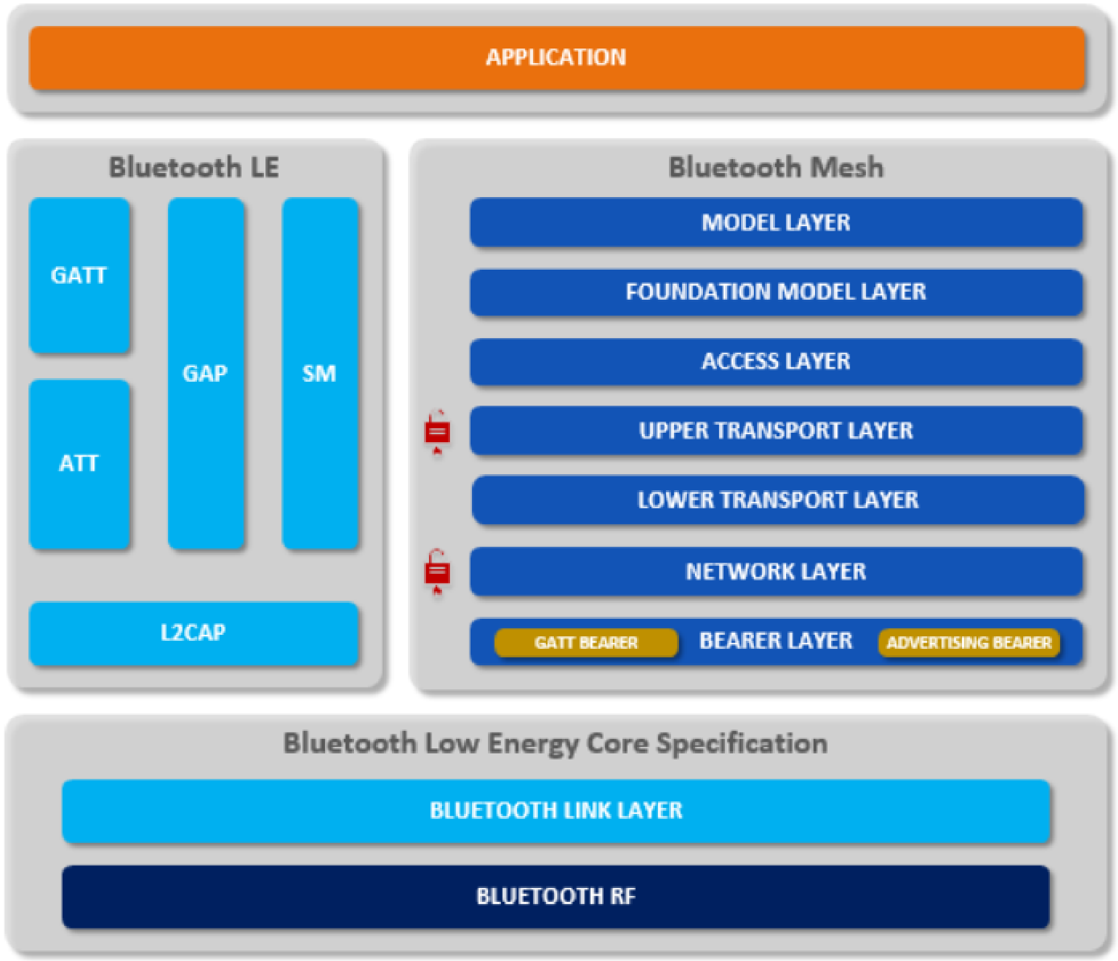
- Upper Transport Layer guarantees confidentiality of access messages through encryption, decryption and authentication as well as defines the way transport control messages are used for managing the Upper Transport Layer between nodes, also for the Friend feature



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

Bluetooth Mesh Software Architecture

- 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



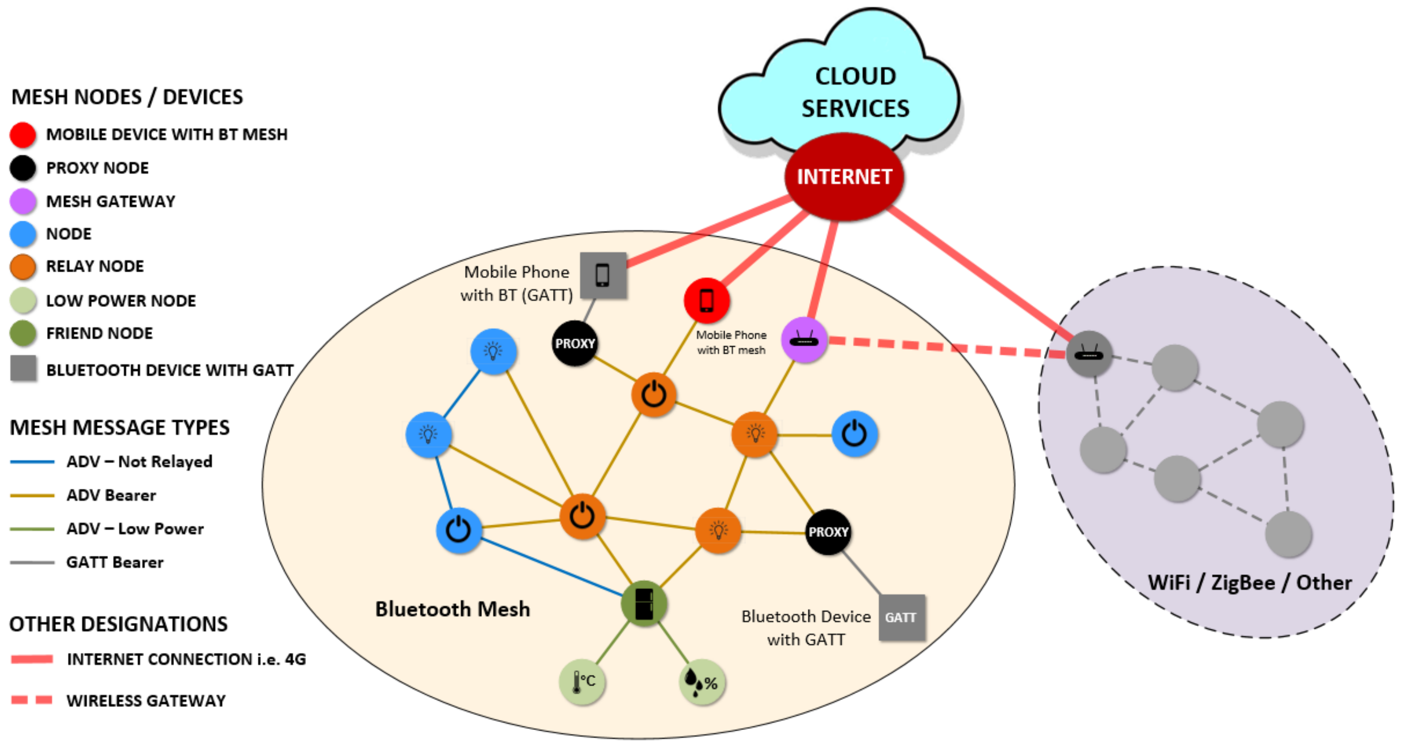
Bluetooth mesh architecture (right) with Bluetooth LE architecture (left)

Bluetooth Mesh - Nodes

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

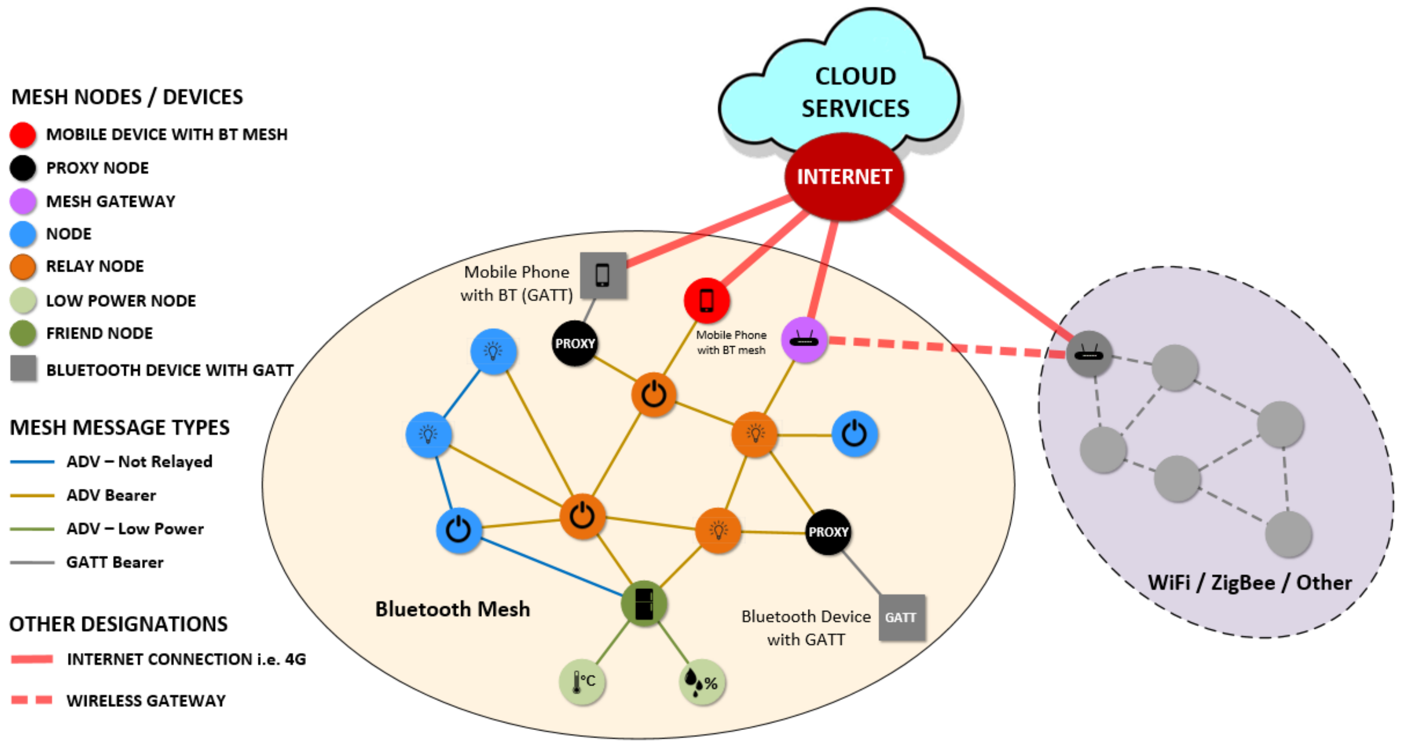
Bluetooth Mesh – Relay Node

- Relay nodes receive and retransmit mesh messages by using the Advertising Bearer to enable larger networks
- 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.



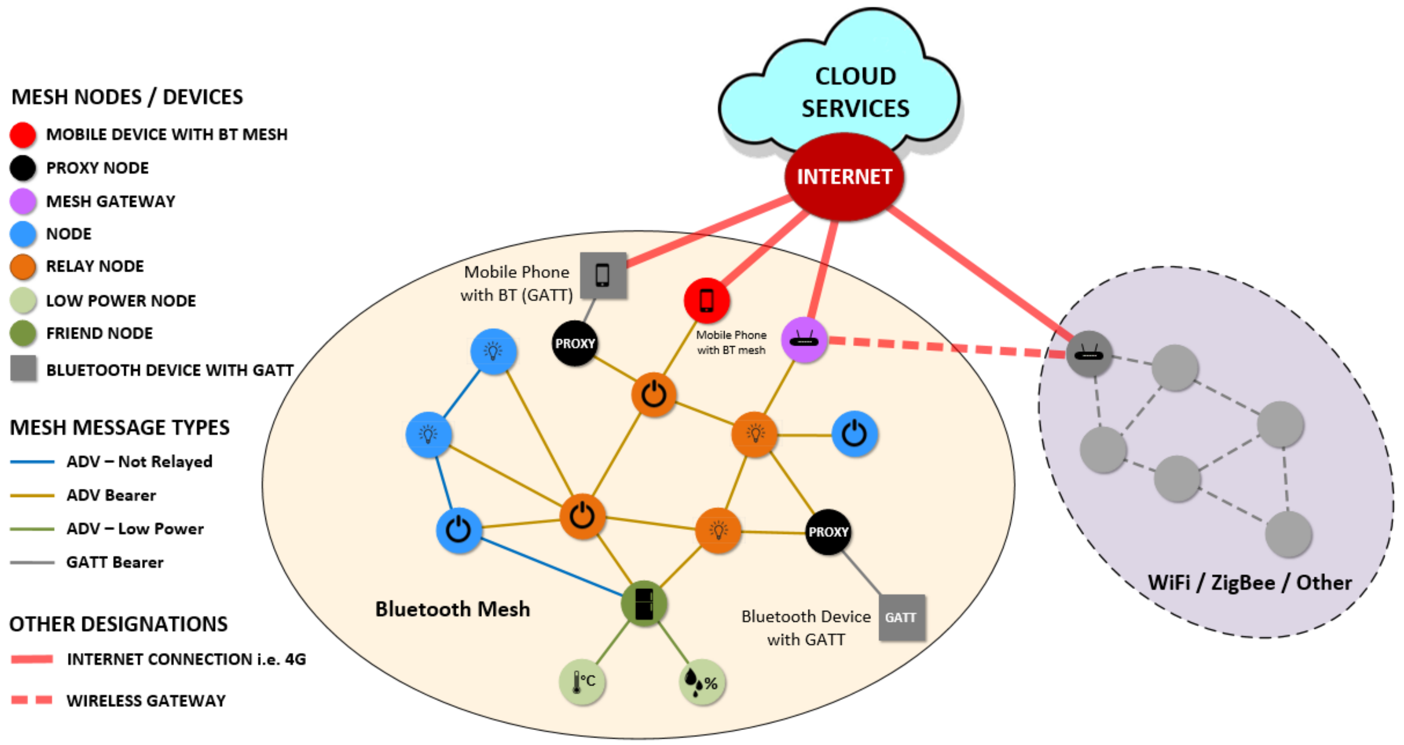
Bluetooth Mesh – Proxy Nodes

- Proxy nodes receive and retransmit mesh messages between GATT and Advertisement Bearers
- 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



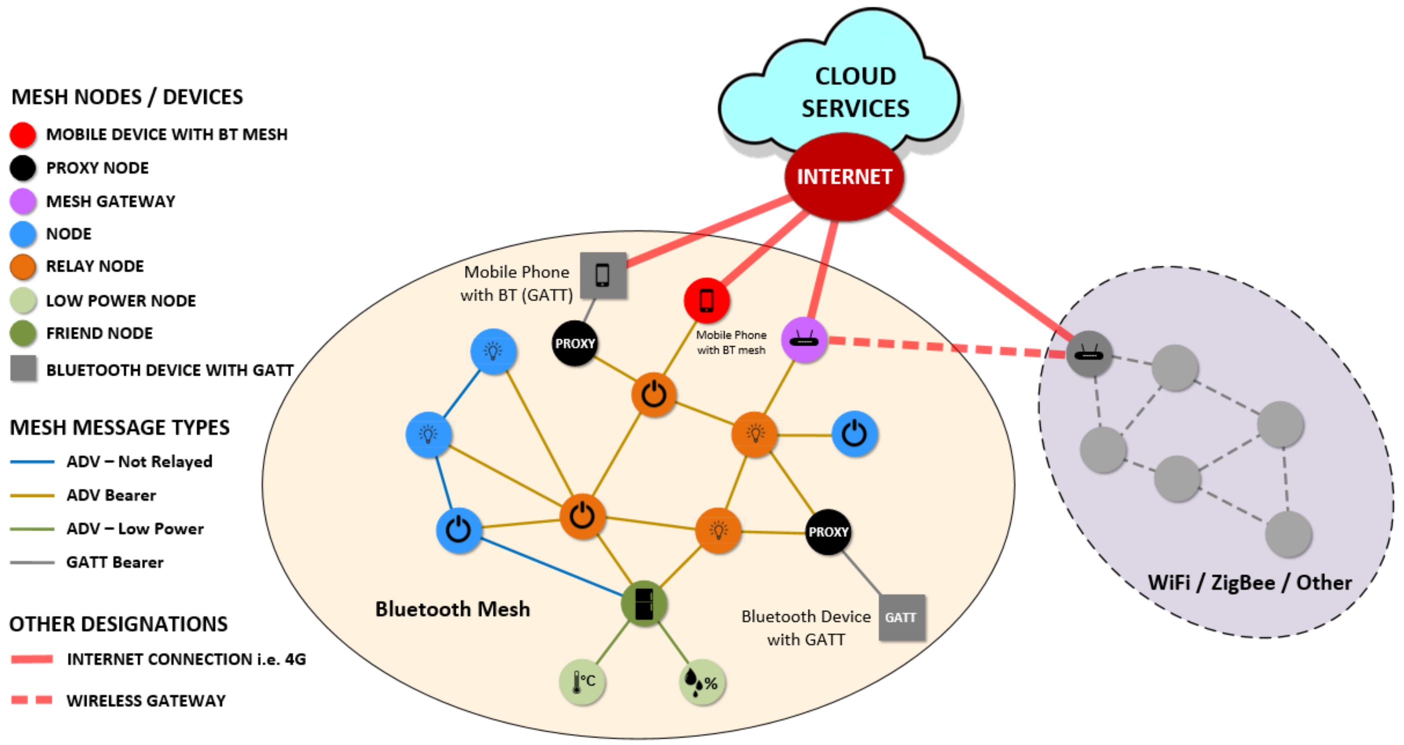
Bluetooth Mesh – Low Power Nodes

- Low Power nodes operate at significantly reduced receiver duty cycle in conjunction with a node supporting the Friend feature
- 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



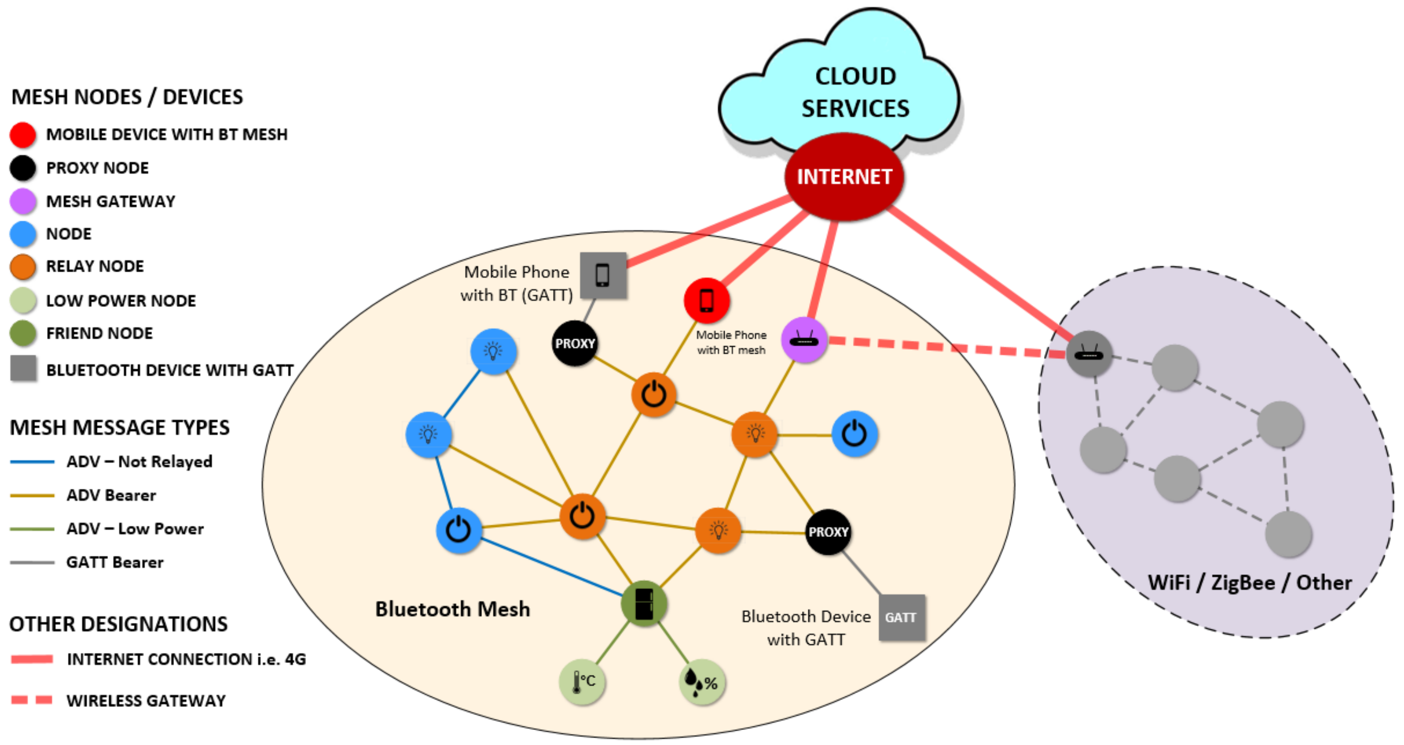
Bluetooth Mesh – Friend Node

- Friend nodes enable low power nodes by storing messages on its behalf (acting as data storage buffers)



Bluetooth Mesh – Gateway Nodes

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



Bluetooth Mesh - Resources

- All Bluetooth mesh nodes must have these resources:
  - Network Addresses:
    - Identification of message sources and destinations
  - Network Keys:
    - Securing and authentication of messages at the Network Layer
  - Access Keys:
    - Securing and authentication of messages at the Access Layer
  - IV Index:
    - extend or limit the lifetime of the network

Bluetooth Mesh - Resources

- Network Keys and Application Keys both are generated by using a random number generator as defined in the Bluetooth Core Specification
- Both of these keys are shared between nodes
- Application Keys are related to Network Keys and an Application Key is bound to a single Network Key
- Thus, an Application Key is valid only when used with the proper Network Key
    - The application is only valid on this network

Bluetooth Mesh - Resources

- 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 regular basis the benefit is that possible obfuscation attacks would need to be started again

Bluetooth Mesh - Elements

- Each node must have at least one defined element called 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
- The 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

Bluetooth Mesh - Elements

- 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

Bluetooth Mesh - Models

- Server Models are compositions of one or more states spanning over one or more elements
- Server Models also define State Transitions, State Bindings and transmittable and receivable messages of elements within the Model
- Server Models define Message, State and State Transition behavior

Similar to BLE – Service Declaration

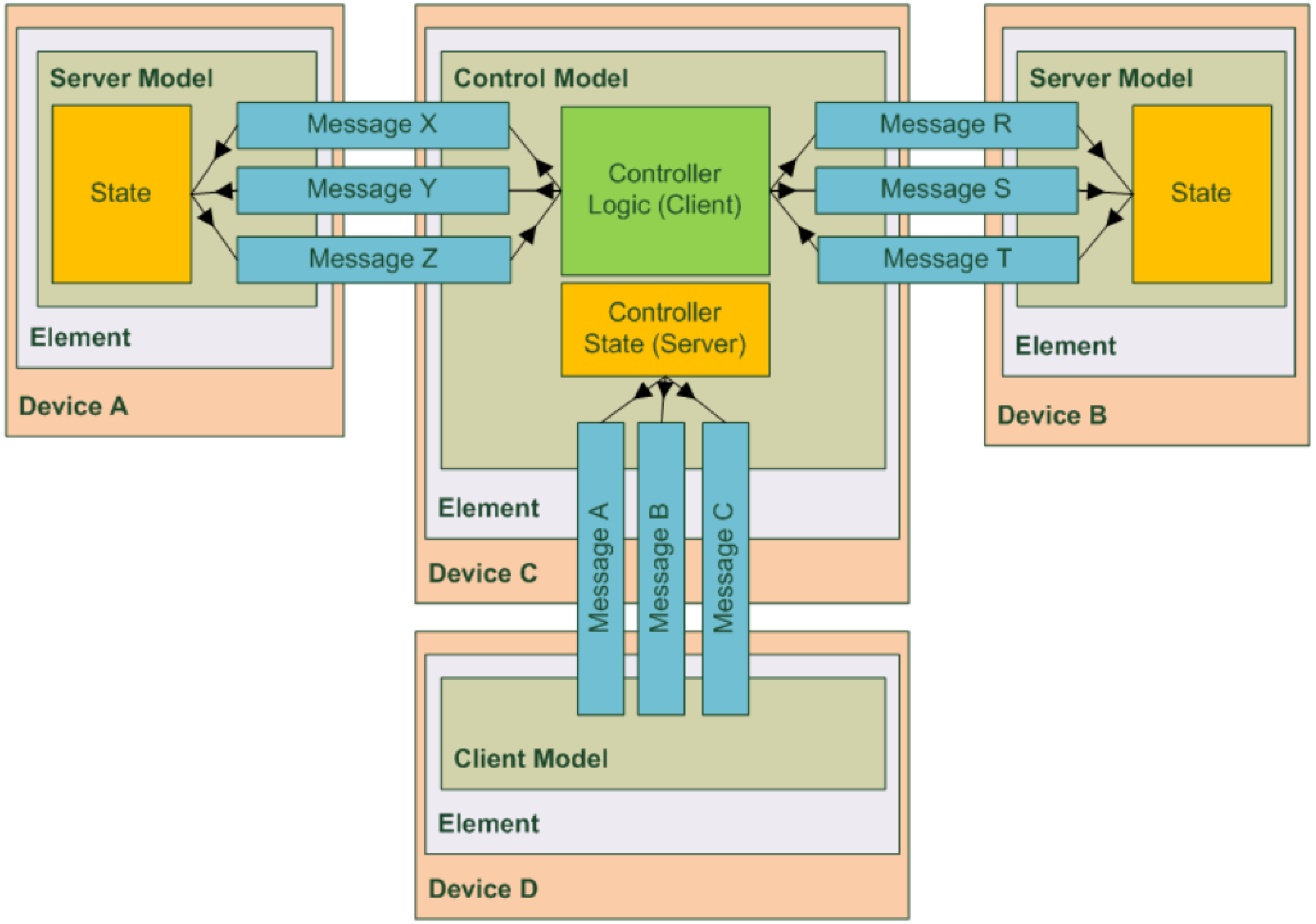
Bluetooth Mesh – Client Models

- Client models 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

Similar to BLE – Client Profile

Bluetooth Mesh – Control Models

- 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



Control Model communication with Client and Server Models [2]

Bluetooth Mesh - Publish and Subscribe

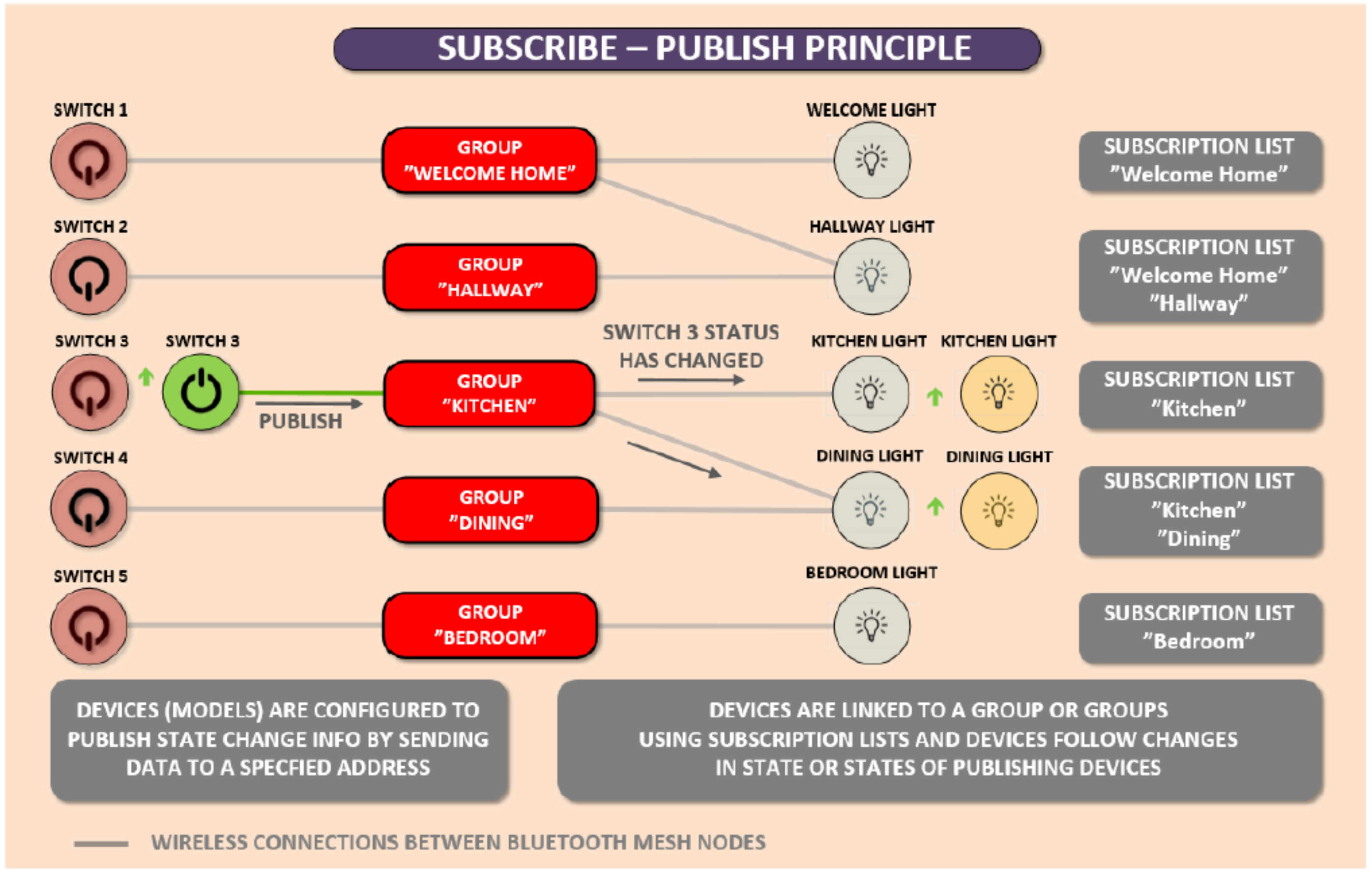
- Bluetooth mesh devices can publish information to multiple devices in the network
- Bluetooth mesh devices can also subscribe to follow information from one or more
- A feature called periodic publishing enables sending messages periodically regardless of whether the state has changed or not

Bluetooth Mesh - Publish and Subscribe

- An example of Bluetooth Mesh publish and subscribe
  - 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 first switch publishes its state light subscribes to the said switch
  - The light that subscribes to the first switch will act accordingly to the published state of the switch
  - The other switch-light pair is managed in similar fashion

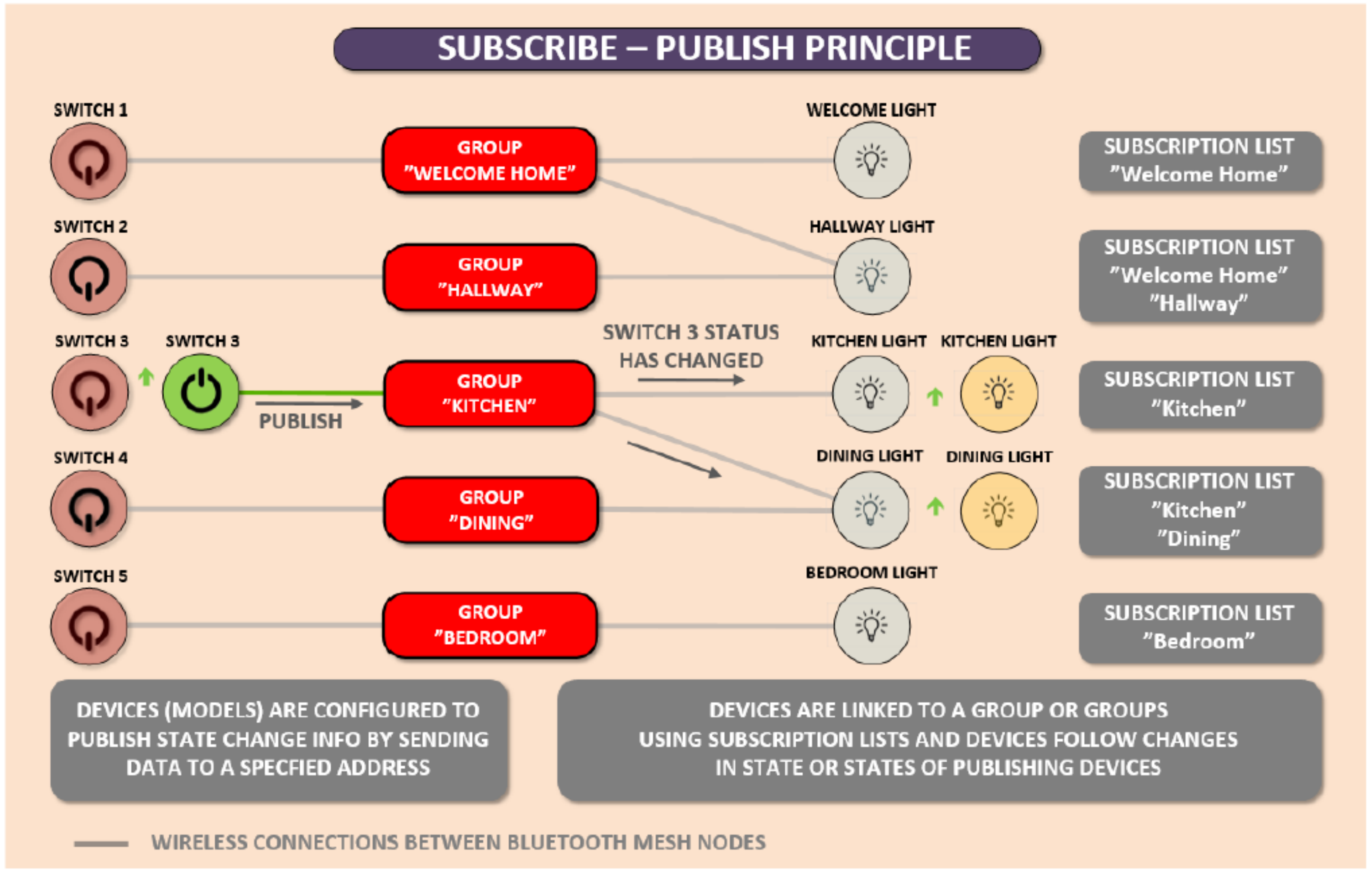
Bluetooth Mesh - Groups

- 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?


SUBSCRIBE – PUBLISH PRINCIPLE
SWITCH 1
SWITCH 2
SWITCH 3
SWITCH 3
SWITCH 4
SWITCH 5
PUBLISH
GROUP "WELCOME HOME"
GROUP "HALLWAY"
GROUP "KITCHEN"
GROUP "DINING"
GROUP "BEDROOM"
SWITCH 3 STATUS HAS CHANGED
WELCOME LIGHT
HALLWAY LIGHT
KITCHEN LIGHT
KITCHEN LIGHT
DINING LIGHT
DINING LIGHT
BEDROOM LIGHT
SUBSCRIPTION LIST "Welcome Home"
SUBSCRIPTION LIST "Welcome Home" "Hallway"
SUBSCRIPTION LIST "Kitchen"
SUBSCRIPTION LIST "Kitchen" "Dining"
SUBSCRIPTION LIST "Bedroom"
DEVICES (MODELS) ARE CONFIGURED TO PUBLISH STATE CHANGE INFO BY SENDING DATA TO A SPECFIED ADDRESS
DEVICES ARE LINKED TO A GROUP OR GROUPS USING SUBSCRIPTION LISTS AND DEVICES FOLLOW CHANGES IN STATE OR STATES OF PUBLISHING DEVICES
WIRELESS CONNECTIONS BETWEEN BLUETOOTH MESH NODES

Bluetooth Mesh - Groups

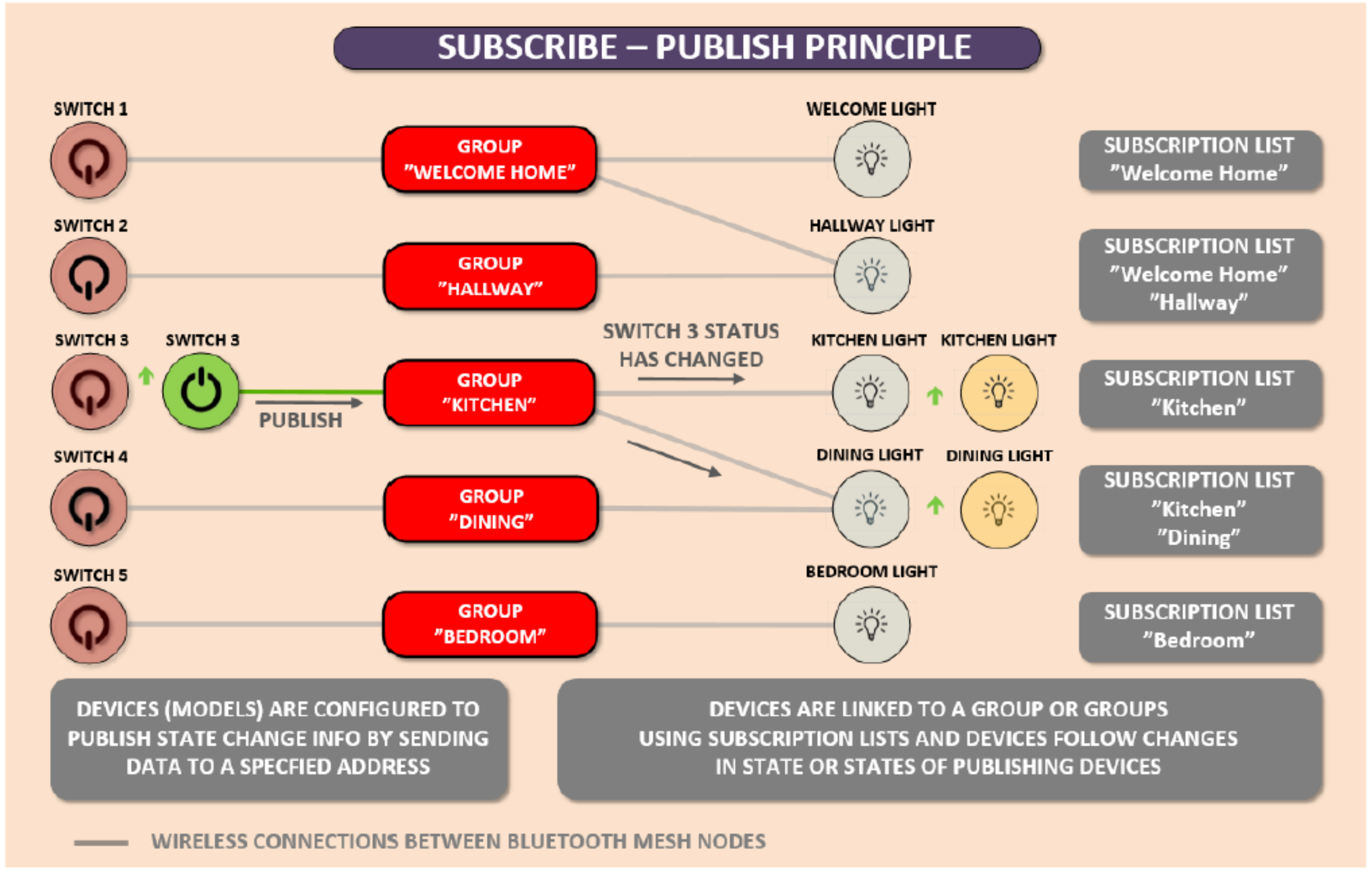
- In BLE, one would need to configure the new switch with all of the addresses of the lights to be controlled by that switch
- In an industrial or large deployment where the number of lights to be controlled increases, BLE could become unpractical
- Bluetooth Mesh solves this by enabling Groups to designate 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.


SUBSCRIBE – PUBLISH PRINCIPLE
SWITCH 1
SWITCH 2
SWITCH 3
SWITCH 3
SWITCH 4
SWITCH 5
PUBLISH
GROUP "WELCOME HOME"
GROUP "HALLWAY"
GROUP "KITCHEN"
GROUP "DINING"
GROUP "BEDROOM"
SWITCH 3 STATUS HAS CHANGED
WELCOME LIGHT
HALLWAY LIGHT
KITCHEN LIGHT
KITCHEN LIGHT
DINING LIGHT
DINING LIGHT
BEDROOM LIGHT
SUBSCRIPTION LIST "Welcome Home"
SUBSCRIPTION LIST "Welcome Home" "Hallway"
SUBSCRIPTION LIST "Kitchen"
SUBSCRIPTION LIST "Kitchen" "Dining"
SUBSCRIPTION LIST "Bedroom"
DEVICES (MODELS) ARE CONFIGURED TO PUBLISH STATE CHANGE INFO BY SENDING DATA TO A SPECFIED ADDRESS
DEVICES ARE LINKED TO A GROUP OR GROUPS USING SUBSCRIPTION LISTS AND DEVICES FOLLOW CHANGES IN STATE OR STATES OF PUBLISHING DEVICES
WIRELESS CONNECTIONS BETWEEN BLUETOOTH MESH NODES

Bluetooth Mesh - Groups

Enabling Industrial IoT applications!

- Bluetooth mesh configuration becomes much simpler with the replacement switch only needing to be configured to publish data only to the Group instead of a multitude of lights
- Using 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
- Acknowledged messages will cause a response message while unacknowledged will not



SUBSCRIBE – PUBLISH PRINCIPLE
SWITCH 1
SWITCH 2
SWITCH 3
SWITCH 3
SWITCH 4
SWITCH 5
PUBLISH
GROUP "WELCOME HOME"
GROUP "HALLWAY"
GROUP "KITCHEN"
GROUP "DINING"
GROUP "BEDROOM"
SWITCH 3 STATUS HAS CHANGED
WELCOME LIGHT
HALLWAY LIGHT
KITCHEN LIGHT
KITCHEN LIGHT
DINING LIGHT
DINING LIGHT
BEDROOM LIGHT
SUBSCRIPTION LIST "Welcome Home"
SUBSCRIPTION LIST "Welcome Home" "Hallway"
SUBSCRIPTION LIST "Kitchen"
SUBSCRIPTION LIST "Kitchen" "Dining"
SUBSCRIPTION LIST "Bedroom"
DEVICES (MODELS) ARE CONFIGURED TO PUBLISH STATE CHANGE INFO BY SENDING DATA TO A SPECFIED ADDRESS
DEVICES ARE LINKED TO A GROUP OR GROUPS USING SUBSCRIPTION LISTS AND DEVICES FOLLOW CHANGES IN STATE OR STATES OF PUBLISHING DEVICES
WIRELESS CONNECTIONS BETWEEN BLUETOOTH MESH NODES