



Bluetooth mesh

Principles and Commissioning

Rick Chung

Customer Application
Engineer

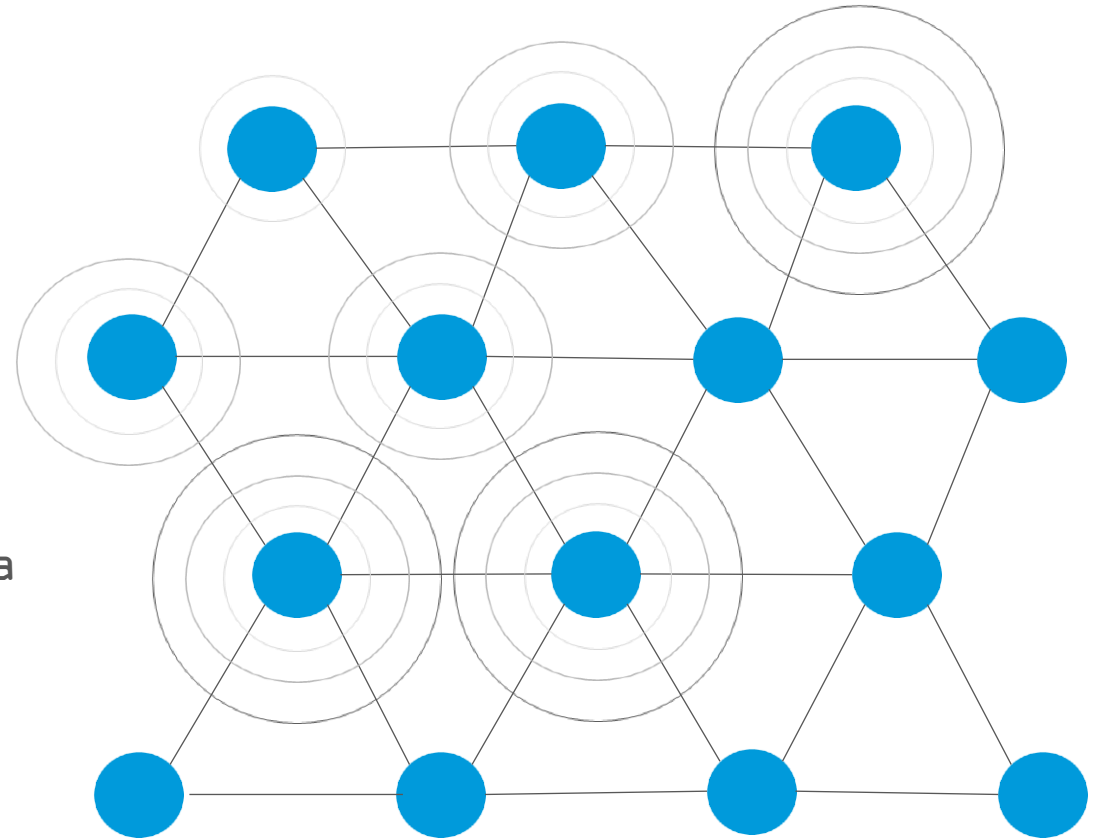
Bluetooth Asia 2018

Why we need mesh?

- If we want to communicate with a device which is far away
 - Enlarge the power of transmission?
- What if there are some physical obstacle?
 - Can we enlarge the power of transmission again?
- If we want to communicate with multiple devices at the same time
 - May be we can use multi-link?
 - Or we can use observer and broadcaster?
- What if we want to communicate with lots of device which spread in an area?
- What if we want devices can communicate to each others?
(decentralization)

What Bluetooth mesh brings

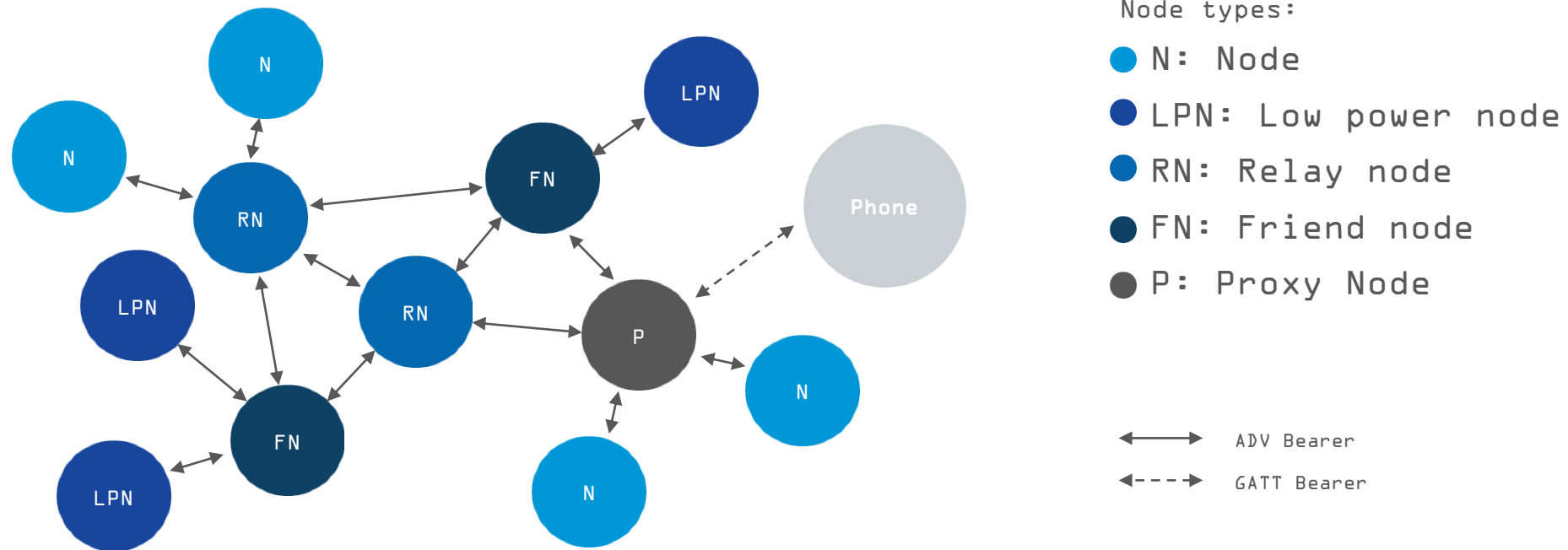
- Multi-hopping feature
 - Multi-hopping to avoid walls blocking radio signal
 - Extending the coverage
- From point-to-point to multi-point control
 - Every device can send/receive data to/from any other device
- Scenario
 - Sensor network
 - Building automation



Bluetooth mesh Specification

- The Bluetooth mesh specification is divided in three parts
 - Mesh Profile - Fundamental requirements for an interoperable mesh network based on Bluetooth LE
 - Mesh Model - Basic functionality of mesh network nodes
 - Mesh Device Properties - Device properties required for the Mesh Model specification.
- Based on Bluetooth v4.0
 - Bluetooth mesh is not a wireless communications technology. It's a networking technology
 - Compatible with any Bluetooth v4.0 controllers
 - Use of Bluetooth 5 feature will break the compatibility

Bluetooth mesh roles and topology



Mesh Key features highlights

- Managed flooding
 - Using managed flooding, easy to implement
 - Using “advertising” in Bluetooth LE for communication among mesh network
- Power consumption
 - Bluetooth LE radio. Low Power Nodes possible with Friend feature
- Grouping concept
 - Group addressing, identified at node level
- Built-in security
 - Security mechanism is not optional, is mandatory
 - Traffic encrypted by AES-128

Different kind of mesh ?

- Flooding mesh
 - All the devices are broadcasting data and also listening the broadcasting from the nearby devices
 - Easy to implement
 - Collision may happen when traffic is high
- Routing mesh
 - Maintain connection between devices
 - Possible to send data along a selected path
 - Single point of failure may happen

What if there is a better “flooding”



Bluetooth mesh is using “managed flooding”

Message Cache

- If a message has been received before, it will be store in the message cache
- Node checks the incoming message, if it already in the message cache, node will drop it simply
- This prevent the duplicated relay message between nodes

Time to Live

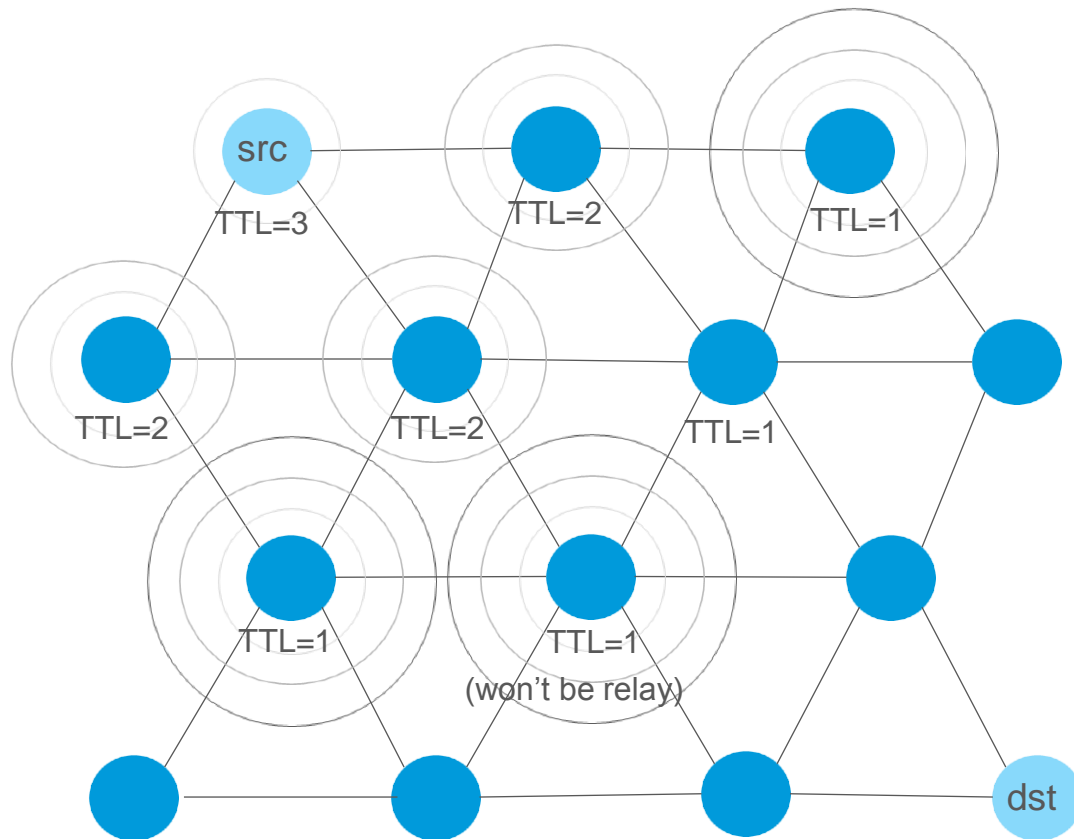
- There is a parameter “TTL” inside every packets
- The TTL range is from 0 ~ 127
- The TTL of one message will decrease every time when the message been relayed
- If $TTL \leq 1$, message won't be relayed
- TTL can reduce the traffic loading for small area message transmission

Different node

features

- Not all the nodes relay messages
- Some nodes can just listen and transmit the message when necessary

How to choose the TTL value ?



If TTL is too small, the message cannot reach destination

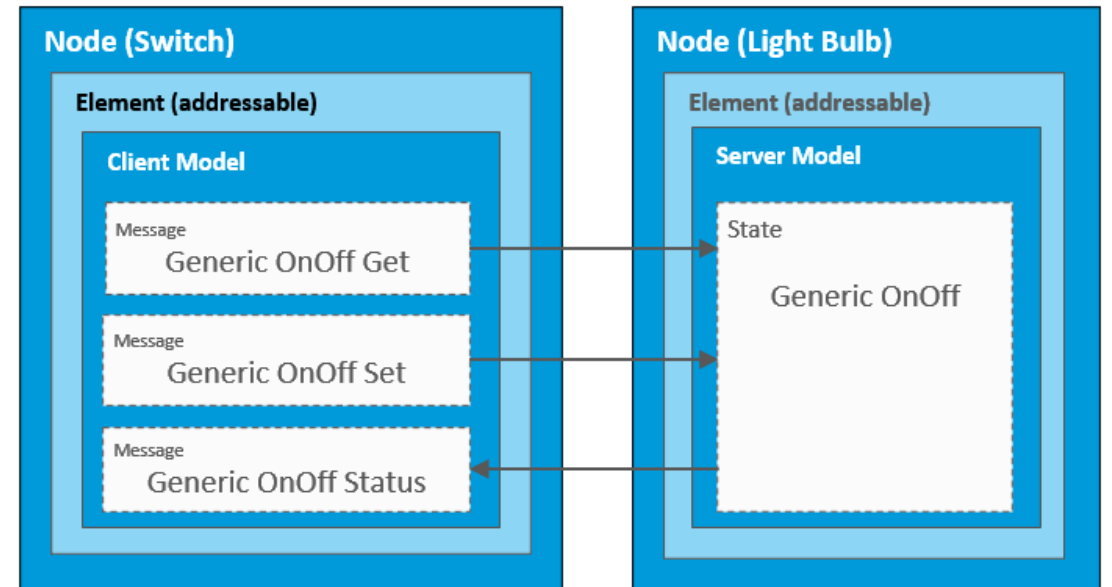
- We can set the TTL number with a fixed value
 - If TTL is too small, the message will not be able to reach the destination
 - If TTL is too large, the mesh traffic will become heavy
- “Heartbeats” message can do help!
 - Can be sent periodically
 - Provide TTL information from source to destination
 - Possible to optimize the TTL

Be careful of “traffic jam”

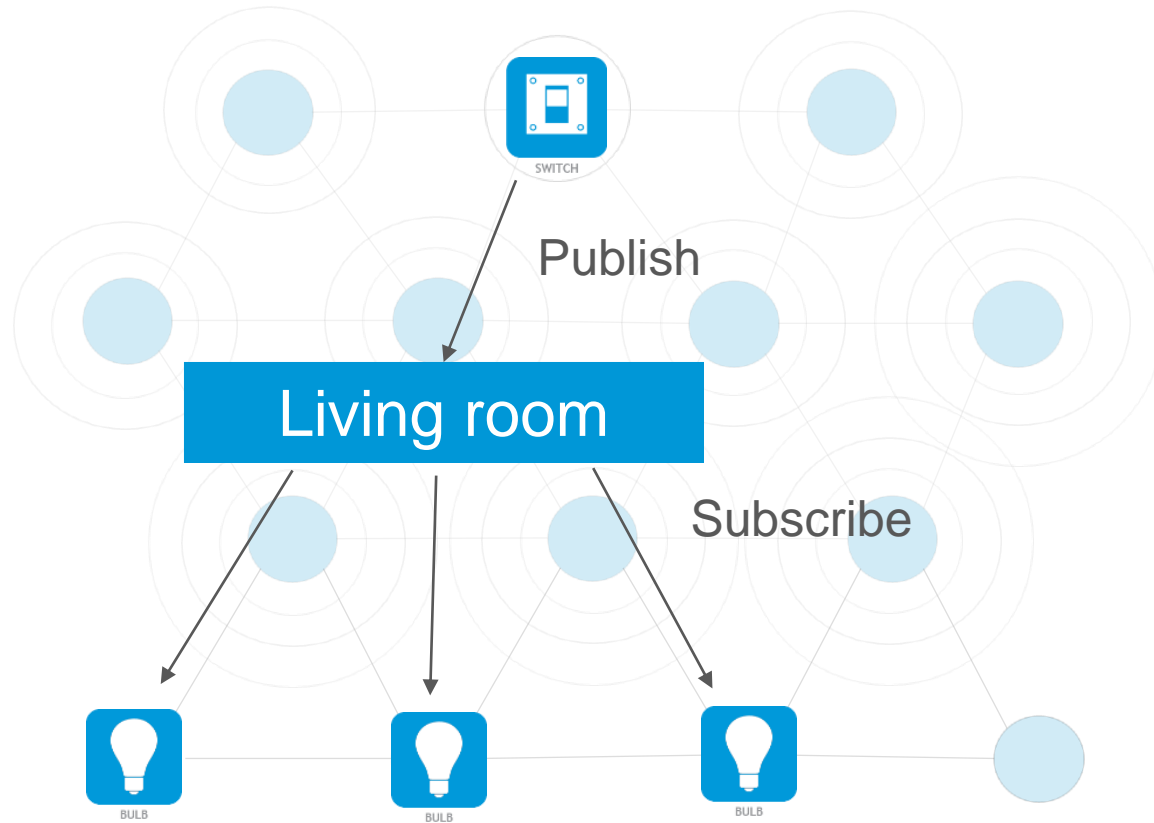
- Bluetooth mesh uses advertising channels to exchange information
 - Only 3 channels are used (channel 37, 38, 39)
 - Channel congestion in case of heavy traffic (e.g. dense network)
 - Additional Wi-Fi or other 2.4 GHz signals interference
- Once the traffic is saturated or interference happens
 - The communication latency will increase
 - Some messages may be lost
- We need to use some mechanisms to solve this
 - using Acknowledge Message in the access layer
 - Setting a filter to focus on advertising packet with mesh types

How device “talk” to each other

- Client-Server model
 - Equivalent to GATT profiles
 - Everything is a **state**
 - Using **publish/subscribe** to exchange information
- Foundation models:
 - Configuration model
 - Health model
- Model specification
 - Generic on/off
 - Lighting
 - Sensors

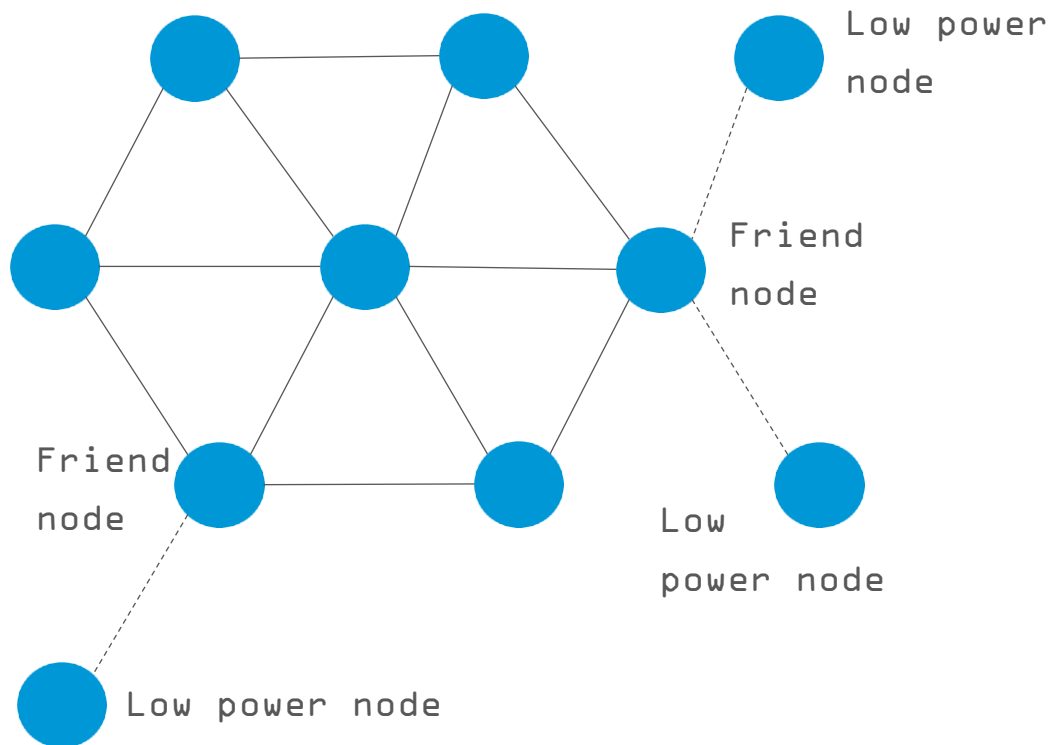


Group addressing



- Built-in group and virtual addressing
- Publish to a group address
- Subscribers receives the message
- Easier to replace nodes

Power consumption and management

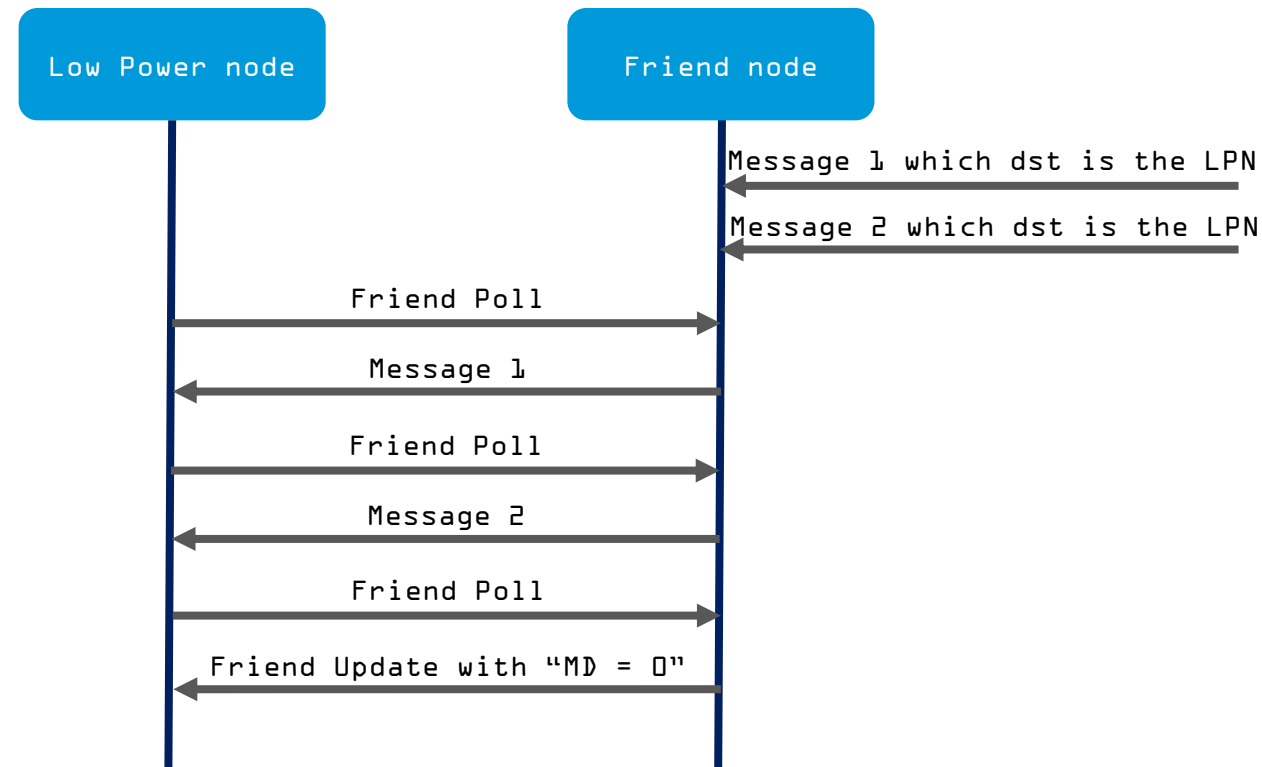


- Nodes continuously in RX
- Normally mains powered
- Friend feature for low power operation
- Friend caches and forwards
- Low power nodes sleep

How friendship works?

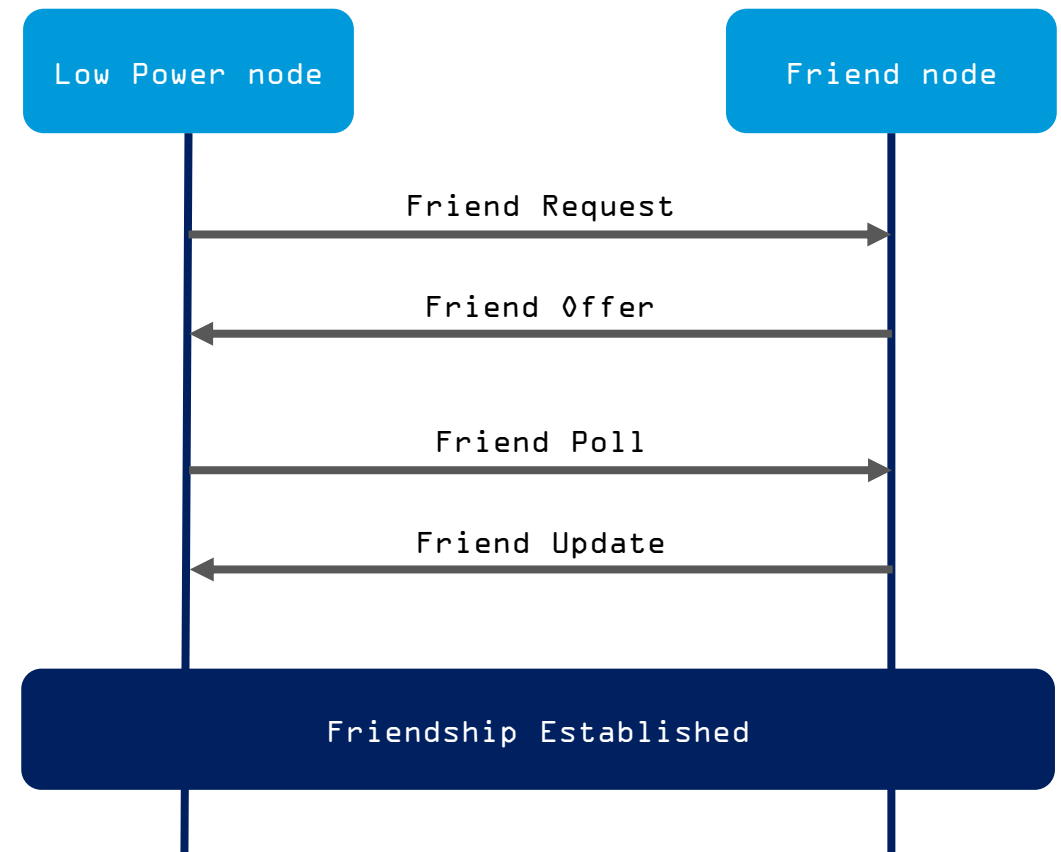
Once the friendship established

- The friend node will help low power node to keep the messages
- When the low power node wakes up
 - Low power node sends friend poll
 - Friend node sends the queued message
 - Low power node continue sends friend poll
 - When there's no message, friend node sends the "friend update" with "MD=0"



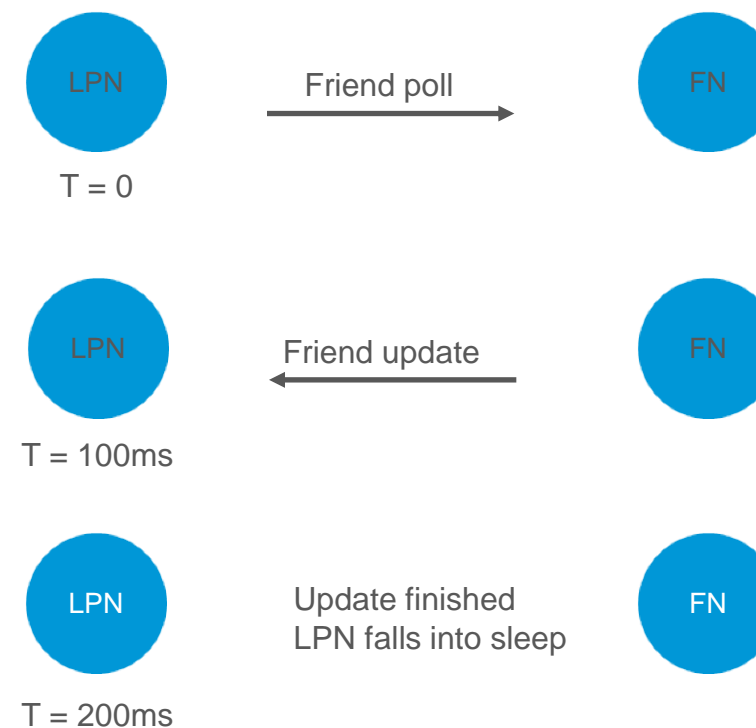
How to establish friendship?

- LPN send "Friend Request" to nearby friend nodes
 - Receive delay
 - Receive window
 - Poll timeout
- Friend send "Friend Offer"
 - Received window size supported
 - Message queue size available
 - Subscription list size available
 - RSSI
- LPN select the friend node by custom algorithm
- LPN send "Friend Poll" to the friend node
- Friend node replies "Friend Update"



How can low power nodes help ?

- For example:
 - If a low power node sends a "friend poll" to friend node to receive stored data every 15 minutes
 - And friend node responds after 100 ms, finishing the update in 100 ms
- The duty cycle for a day will be
 - $0.101 * (60/15) * 24 = 9.696$ sec in a day
 - 0.01 duty cycle



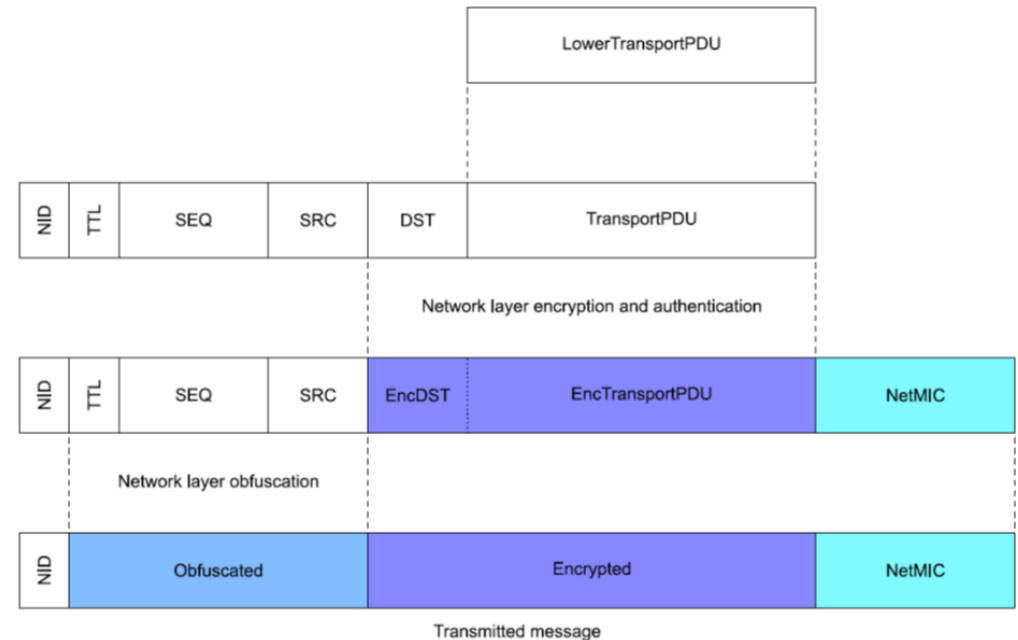
Built-in security



- All traffic encrypted with AES-128
- Only provisioned nodes can communicate
- Additional encryption per application
- Refresh procedures for encryption keys
- Additional protection against replay attacks and trash can attacks

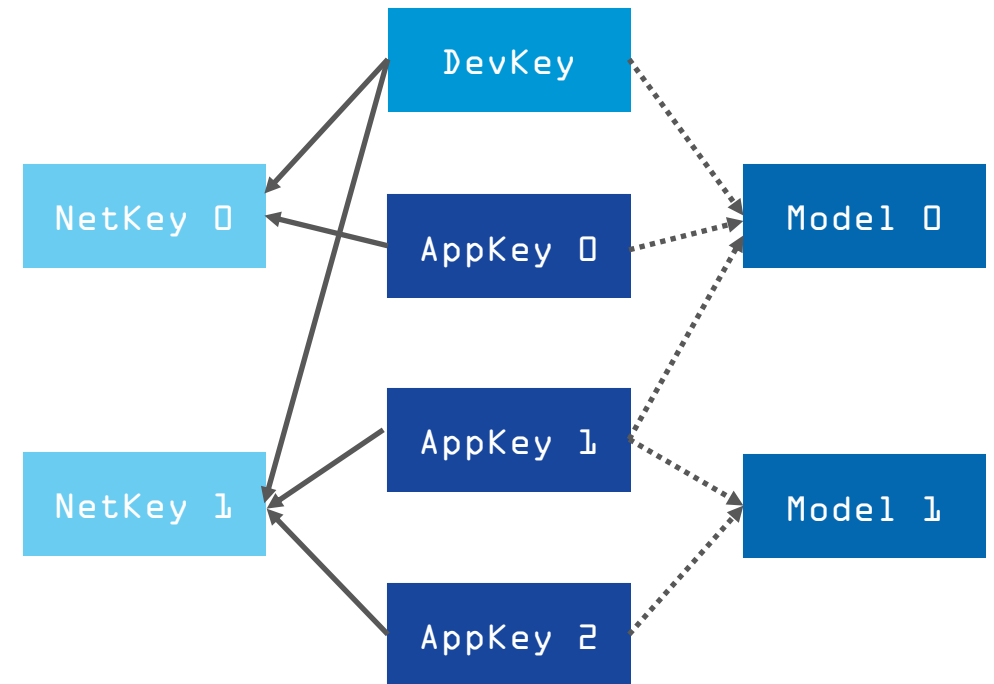
More deep about security

- Not every device can join a mesh network
 - Provisioning is necessary
- 2 layer data encryption using AES-128
 - Network layer - Network key
 - Access layer - Application key
- Split the application and configuration
 - Config client/server negotiated using Device key



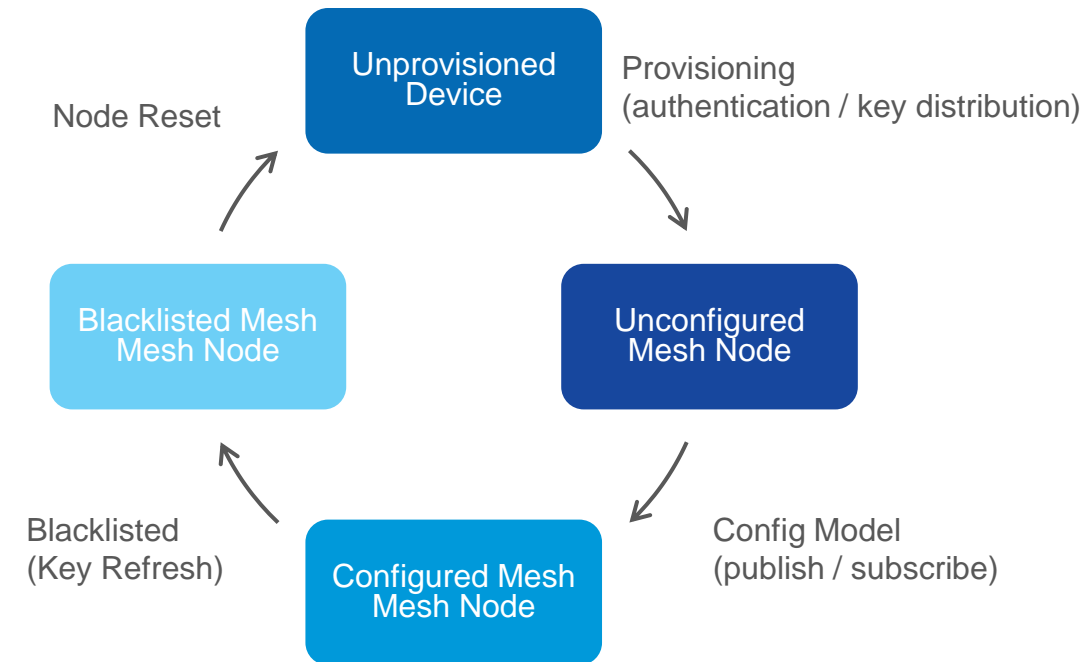
More deep about security

- Keys
 - Application key
 - Only the application with the correct key can access the data
 - You don't want your light bulb to be able to unlock your door
 - Network key
 - Only the message with correct network key can be relayed
 - If the network key is not correct, the packets will be discarded
 - Device key
 - Only the configuration client (provisioner) with correct device key can configure the nodes



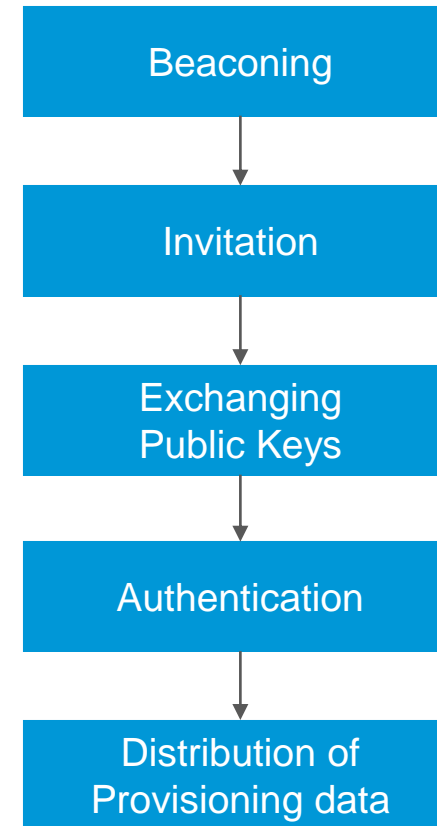
More deep about security

- Provisioning
 - Every device needs to be provisioned before joining a mesh network
 - Authentication is also performed during provisioning
 - ECDH based provisioning provides the protection from man in the middle attack
- Once the device needs to be removed
 - Key refresh procedure for preventing trash-can attack



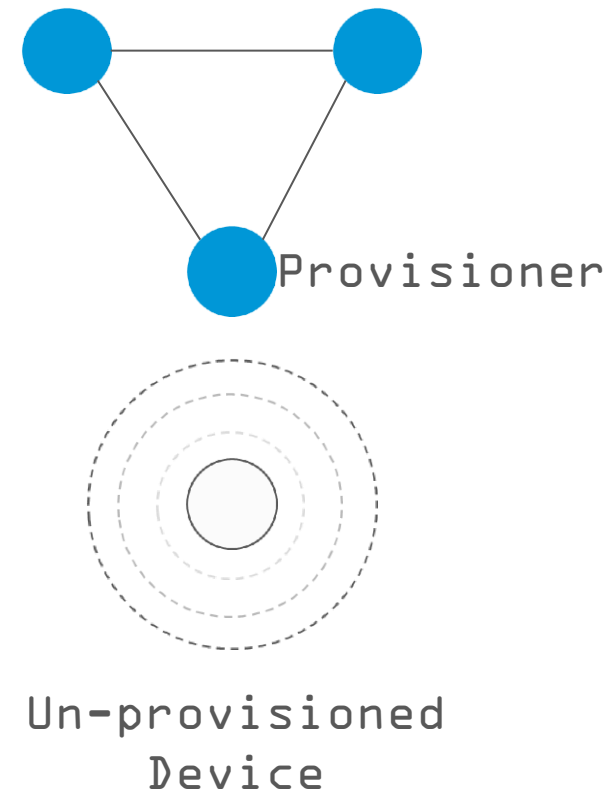
Provisioning

- Provisioning procedure in short:
- Provisioner listens for un-provisioned devices
- Provisioner initiates the Provisioning process by inviting the un-provisioned device
- Provisioner and device establish a secure tunnel for exchanging information and providing provisioning data
 - Unicast Address
 - Device Key



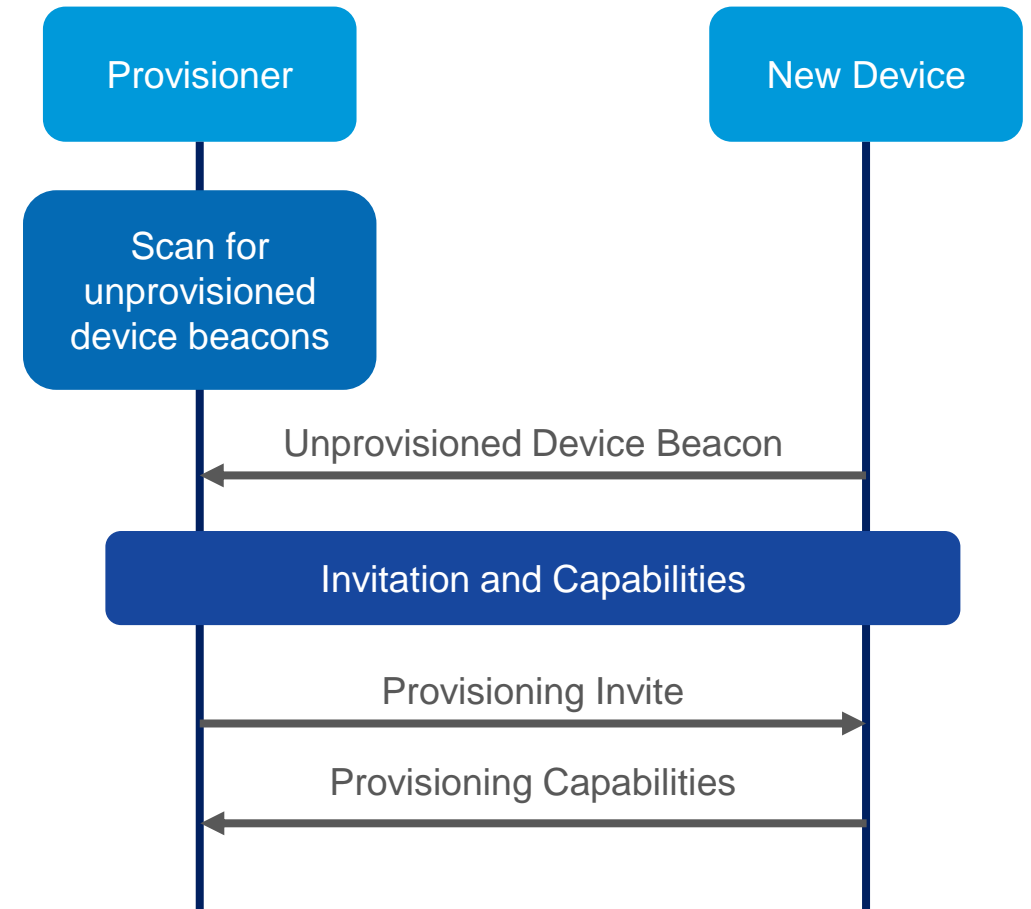
Beaconing

- The unprovisioned device should advertising a beacon signal
 - To indicate the provisioner for provisioning
 - The beacon signal may also contain the OOB information
 - QR code
 - Bar code
 - NFC
 - ...



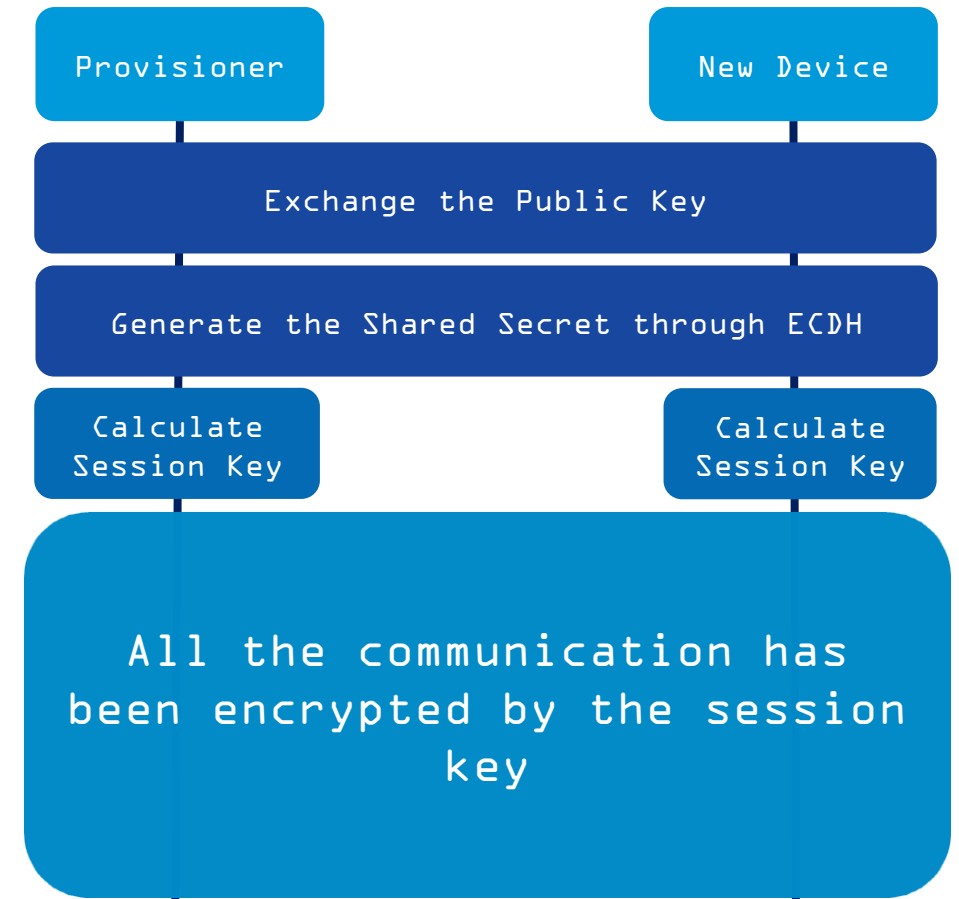
Invitation

- The provisioner can choose the interested device for provisioning
 - Provisioner will send the invitation to the device which contains
 - Attention Timer
 - The device should attract people's attention before time's up
 - The device should reply the IO capabilities
 - Like we see in the Bluetooth LE device pairing



Exchange public keys

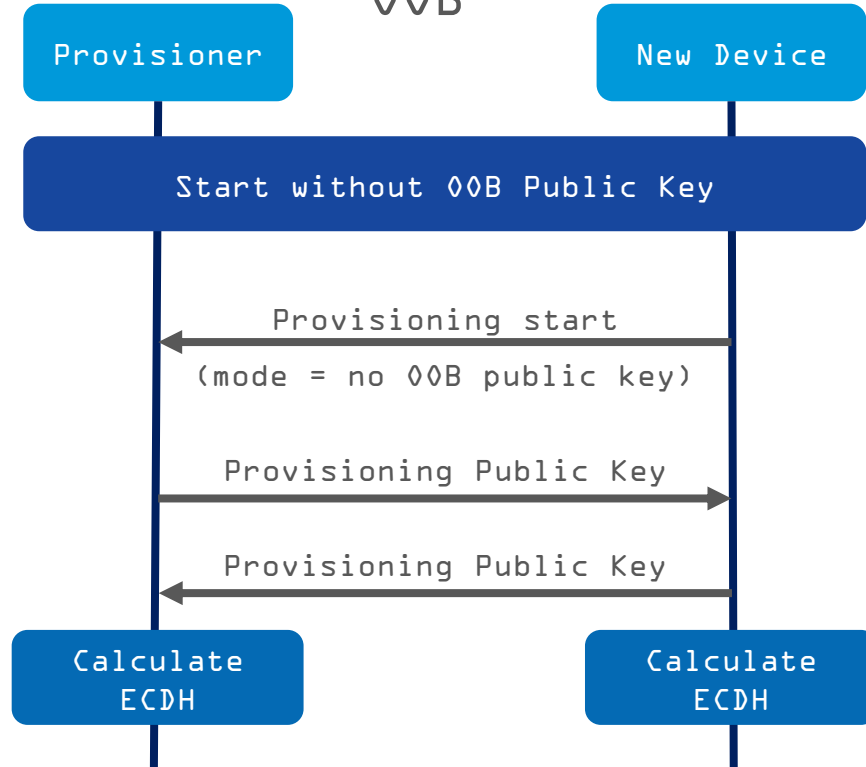
- Why we need this ?
- It is difficult to send the keys securely
 - Someone may sniff keys
- We can use asymmetric encryption to solve this problem
 - Bluetooth mesh use Elliptic Curve Diffie-Hellman (ECDH) to establish a shared secret over an insecure channel
 - Use this shared secret to create a session key for encrypting the



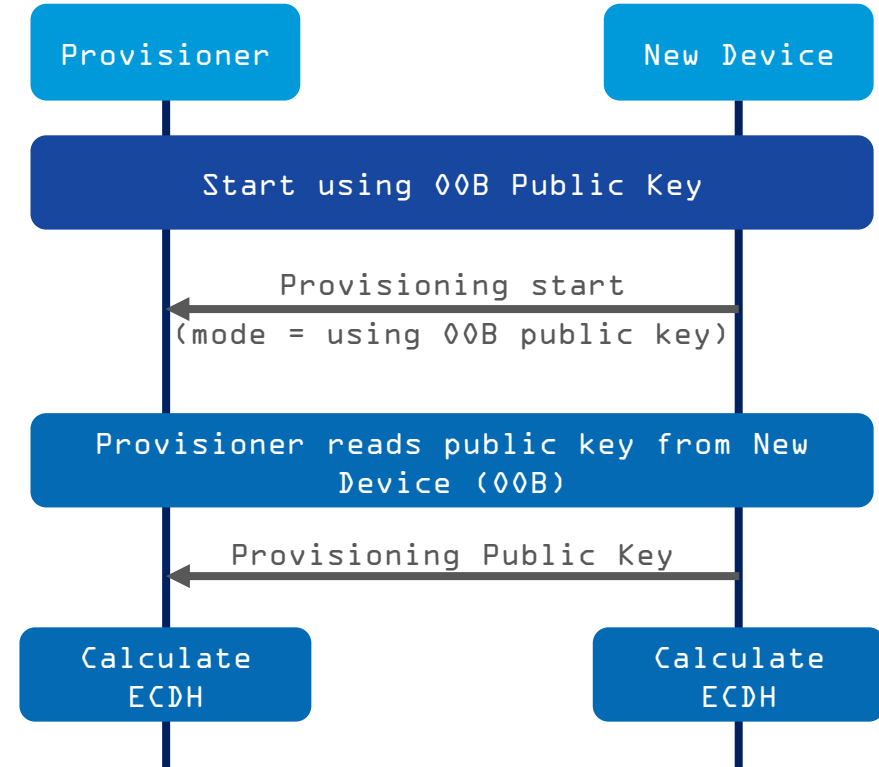
Exchange public keys

Exchange public key without

00B

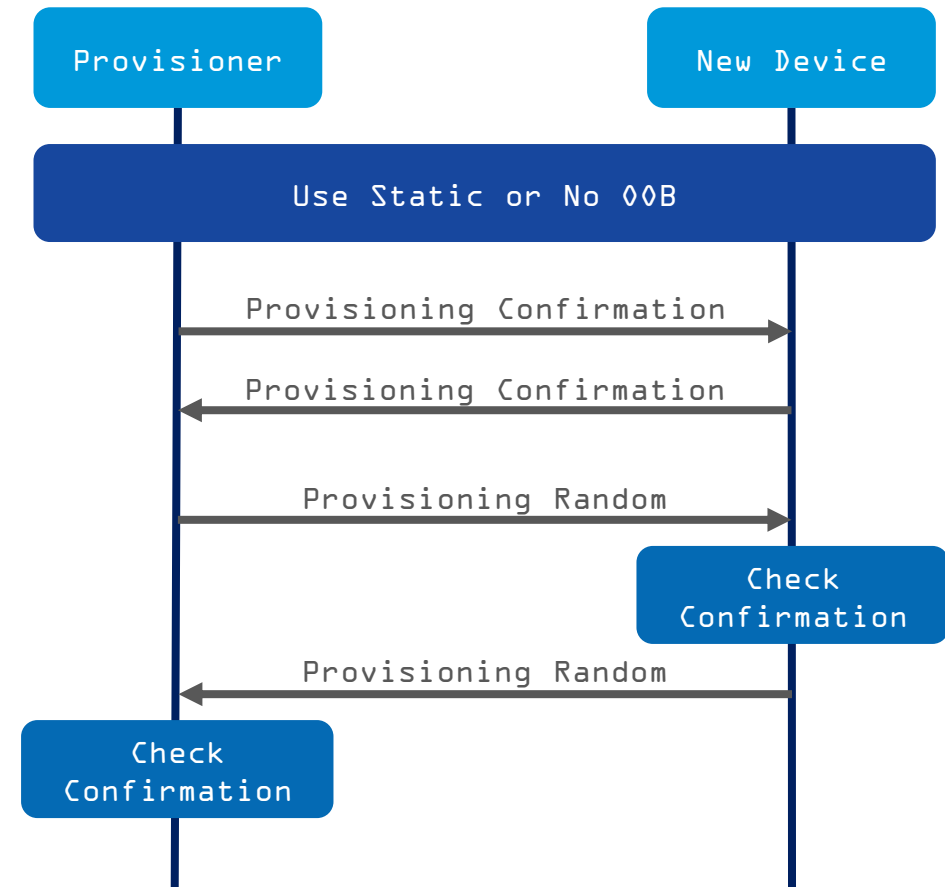


Exchange public key in 00B



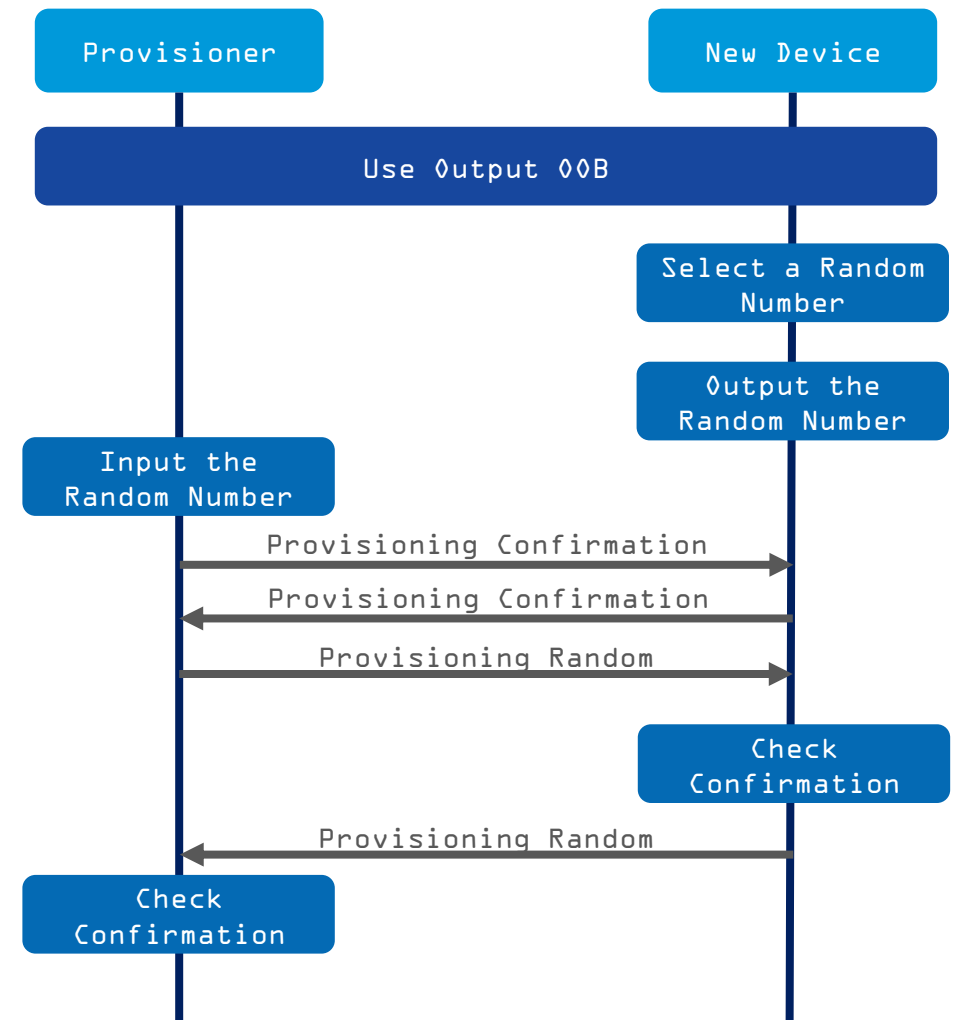
Authentication - Static method

- We need to make sure the device which we are provisioning is the correct one
- Since the device has sent the IO capabilities to the provisioner, the provisioner can choose the way to authenticate the device
- There are three ways to authenticate the device
 - Static
 - Input (perspective of device)
 - Output (perspective of device)



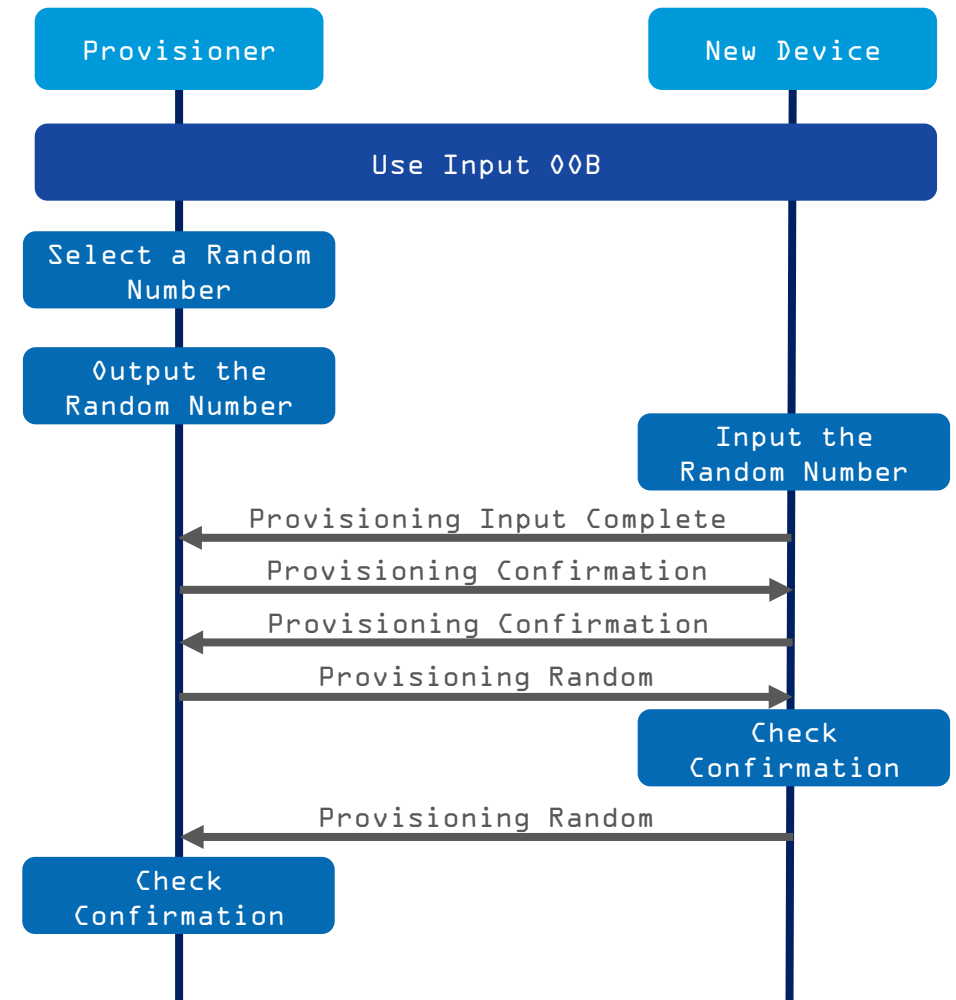
Authentication - Output method

- If device has the output interface, and provisioner has the input interface
 - Device generates a random number
 - LCD shows that random number
 - Blink that LED random times
 - Beep the buzzer
 - User needs to input what he/her saw on the provisioner
 - Push the button several times
 - Enter the number if key pad exists
 - Both of the provisioner and device will use a complex procedure to calculate the result generated by the random number, and confirm with each other



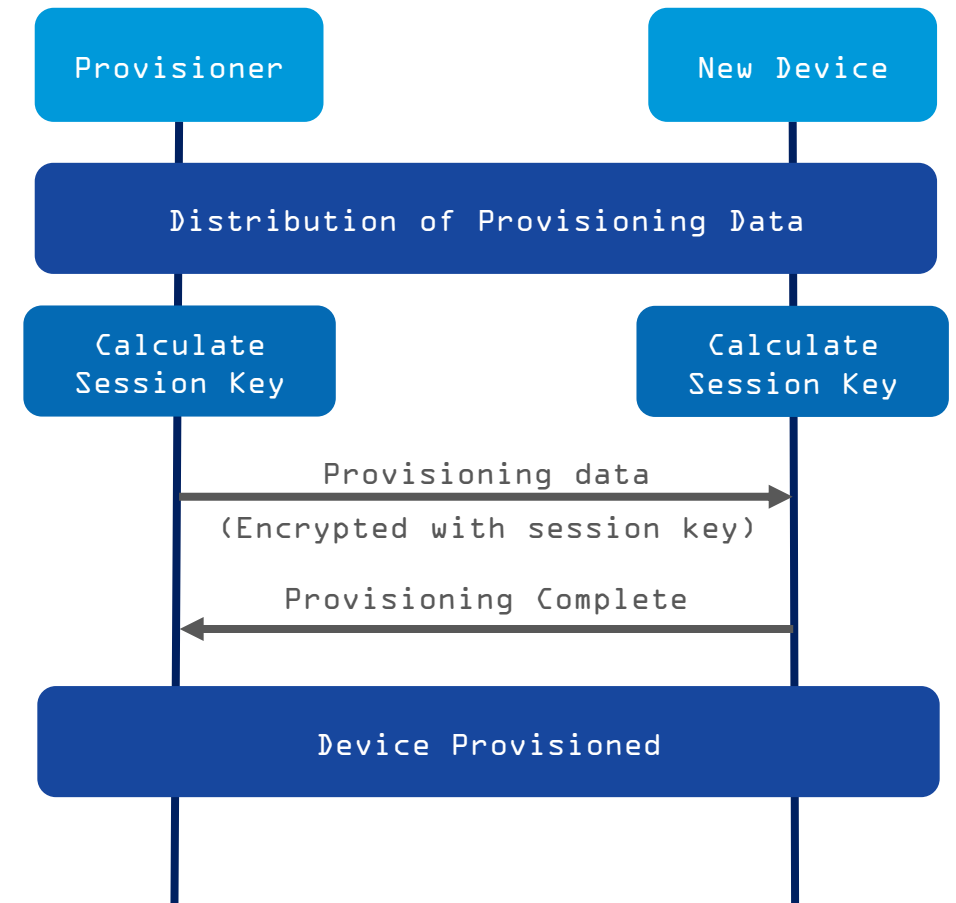
Authentication - Input method

- If device has the input interface, and provisioner has the output interface
 - Provisioner generates a random number
 - LCD shows that random number
 - Blink that LED random times
 - Beep the buzzer
 - User needs to input what he/her saw on the device
 - Push the button several times
 - Enter the number if key pad exists
 - Both of the provisioner and device will use a complex procedure to calculate the result generated by the random number, and confirm with



Distribution of provisioning data

- After the confirmation is finished
 - We can use the “**shared secret**” which generated by ECDH to get a session key through the KDF (key derivation function)
 - For now, both of the provisioner and device need to use the session key to encrypt/decrypt the provisioning data
 - Network key
 - Device key
 - IV index
 - Unicast address



The best security set for provisioning

- How to reach the best security set?
 - Since man-in-the-middle (MITM) attack may happen
 - The specification provides a suggestion like this:

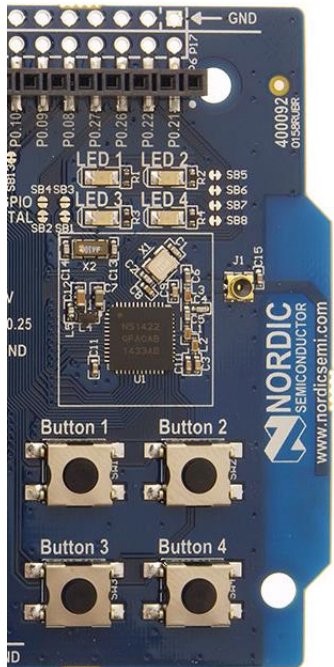
Provisioning may be secure or insecure. Secure Provisioning requires the following method:

- FIPS P-256 Elliptic Curve Algorithm, a Public Key Type that is not transferred in band (i.e., "OOB Public Key is used" is selected), and a Static OOB of any size.
- FIPS P-256 Elliptic Curve Algorithm; OOB Action of Input Numeric, Input Alphanumeric, Output Numeric, or Output Alphanumeric; and OOB Size of at least 6 octets.

Otherwise, provisioning is Insecure Provisioning.

Working example

Device



Device



Device

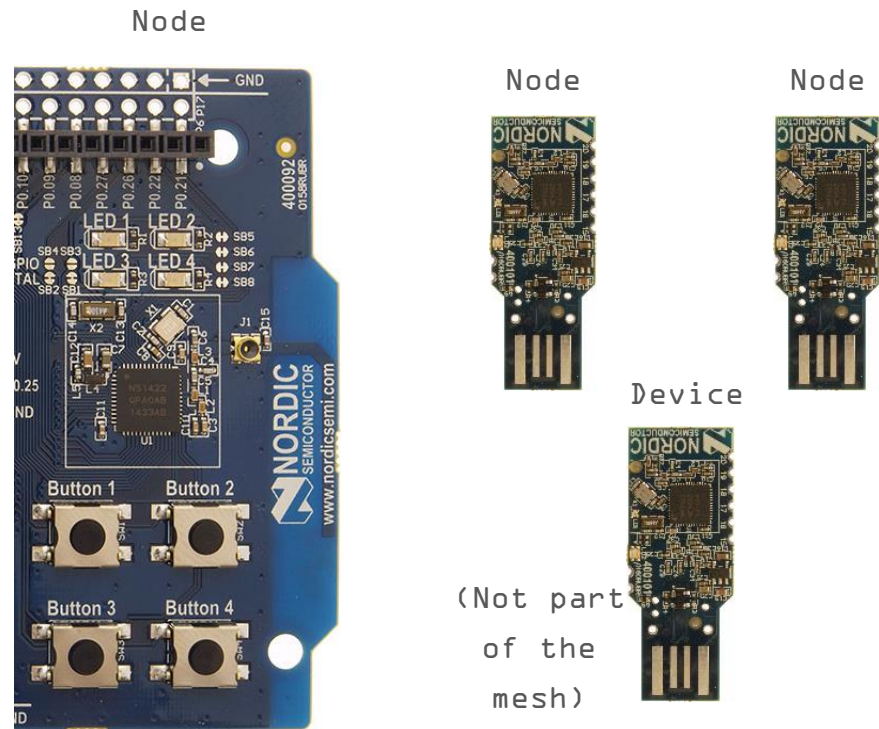


Device



- Entities capable of being provisioned are called devices.
- Devices advertise a beacon signal.
- Devices can be provisioned by a provisioner.
- The technical meaning of provisioning is to distribute addresses and encryption keys

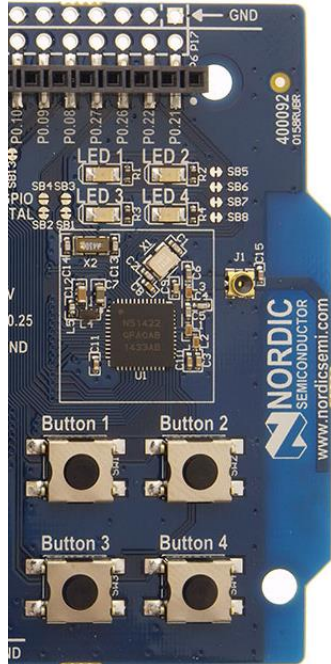
Working example



- Provisioned devices are called nodes.
- Nodes part of the network share the same address space.
- An addressable entity is called an element.
- Node has one or more elements.

Working example - configuration

Lighting control



Light 1



Light 2



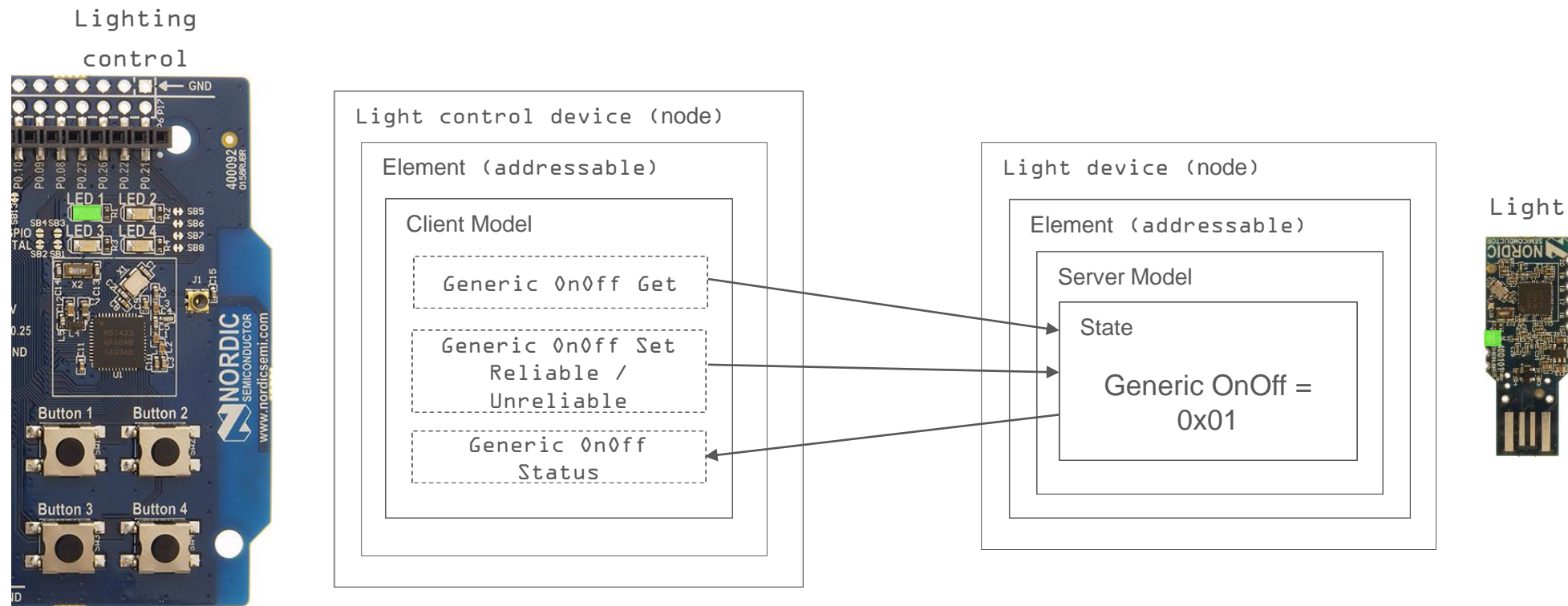
Device



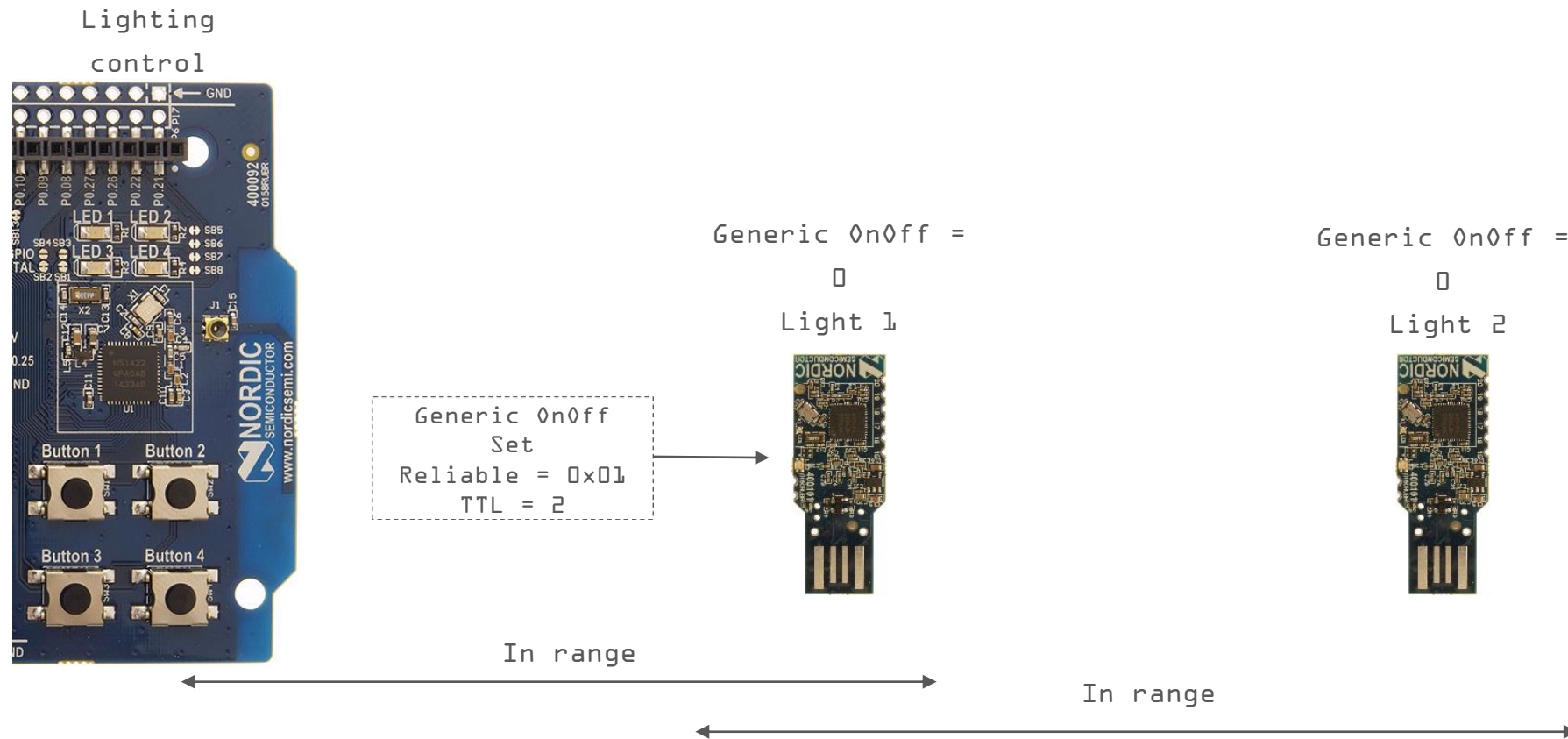
(Not part
of the
mesh)

- Provide application keys
- Configure how nodes do
 - Publish
 - Subscribe

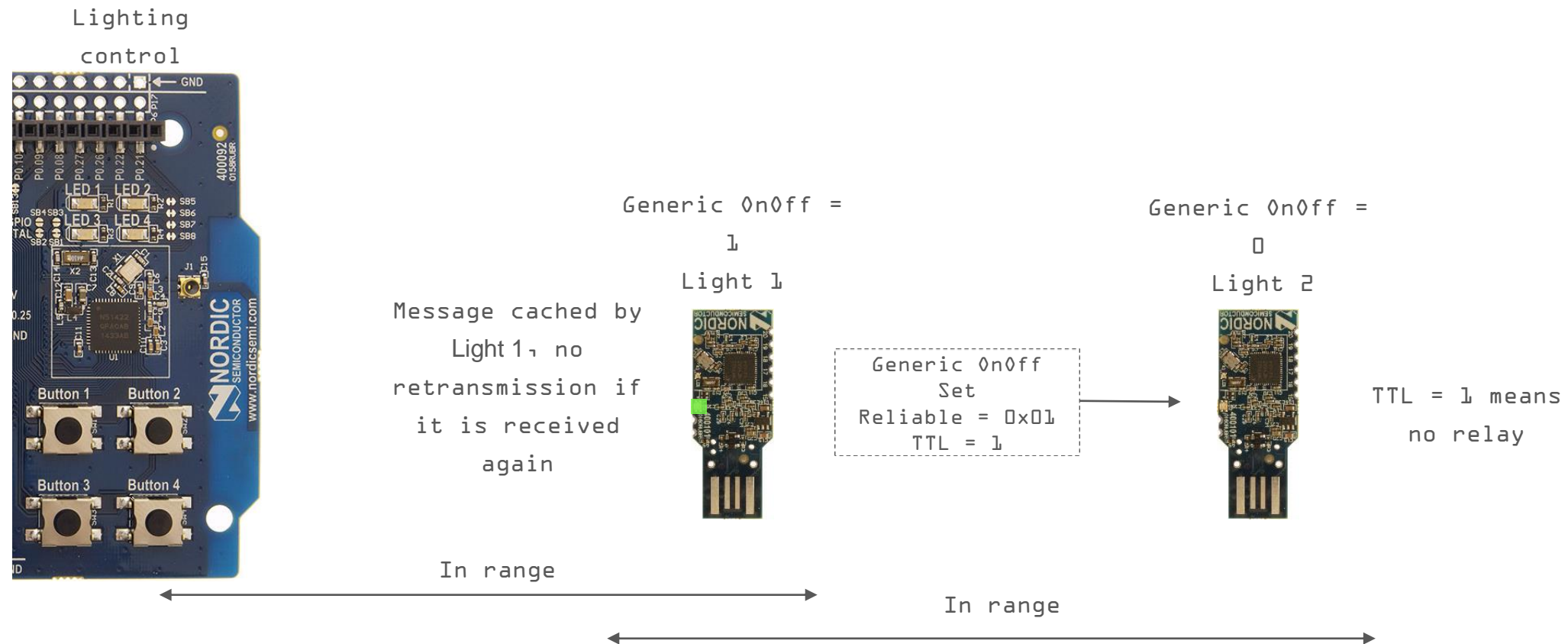
Working example - Controlling a lighting



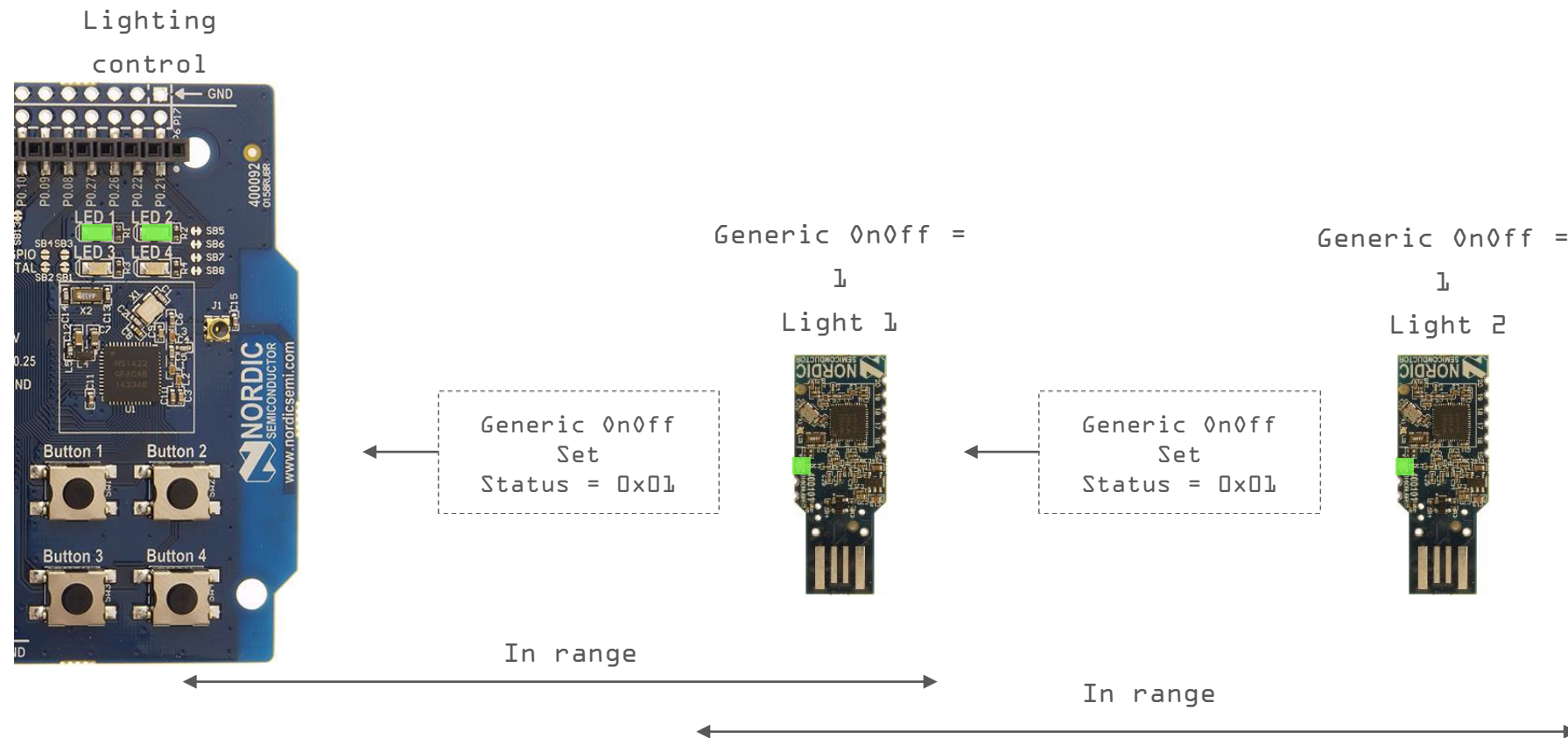
Working example - message transmission



Working example - message transmission



Working example - message transmission



Some key parameter for Bluetooth mesh

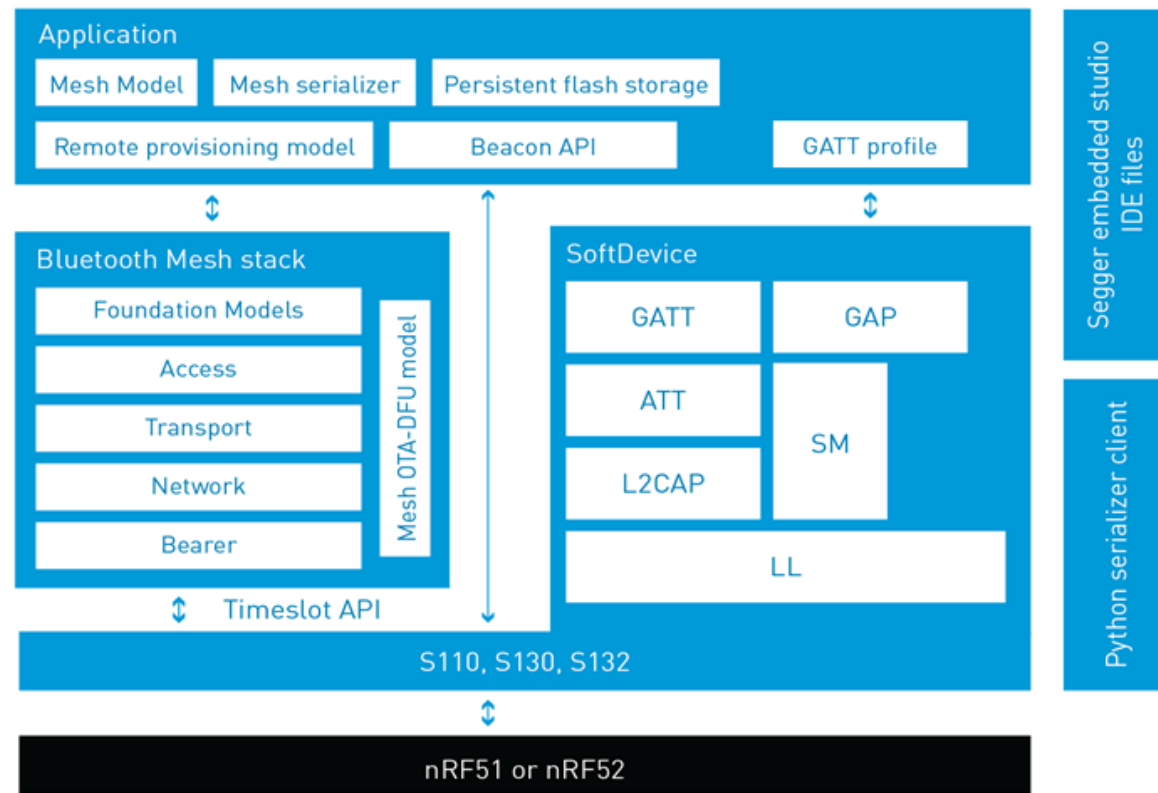
- The maximum configuration of TTL is 127
 - If $TTL = 1$, the message will not be relayed
- 32767 elements in a mesh network
 - The unicast address space is 2^{15}
- 16384 group addresses in a mesh network
 - The group address space is 2^{14}
- 70 trillion virtual addresses in a mesh network
- 340 undecillion (2^{128}) mesh networks
- 4096 applications per mesh network

nRF5 SDK for Mesh

Bluetooth mesh from Nordic Semiconductor

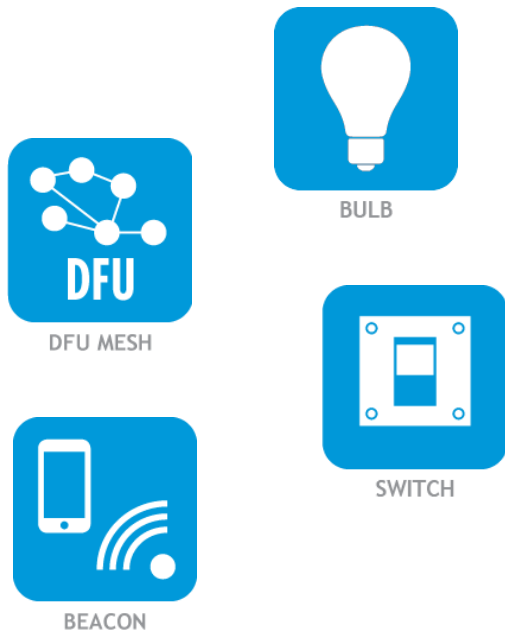
nRF5 SDK for Mesh

nRF5 SDK for Mesh



- Source code release
- Mesh examples
- DFU solution
- Serialization solution
- Python tools

Bluetooth mesh examples



- Light control
 - With/without proxy features
 - PB-ADV provisioning
 - PB-GATT provisioning
- Serial interface
 - With interactive Python interface
- PB-Remote provisioning
- Beaconing
- DFU example

Mesh DFU solution

- Update mesh nodes with new firmware image
- Background mode: No interruption of service
- Nordic specific feature



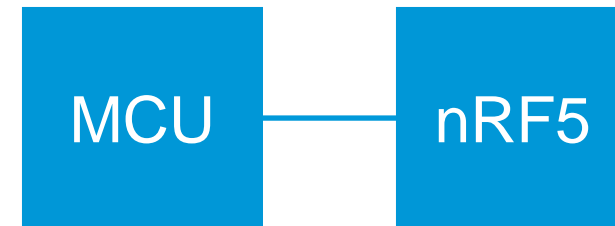
Mesh serial interface

Gateway



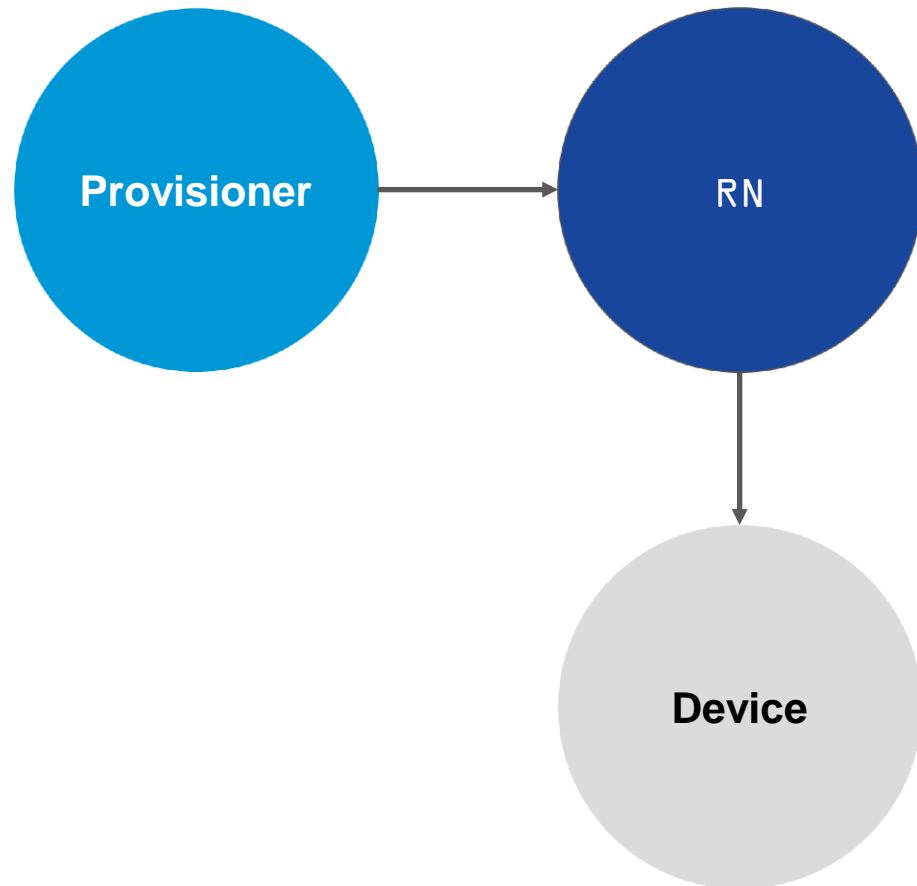
Gateway to Internet
Ethernet or WiFi capable
Application processor

3rd party MCU/CPU



Existing or specialized
application
DALI, DMX, 0-10V
General Purpose MCU

PB-Remote - Remote provisioning



- Provision device across the mesh network
- No need to have device in range of the provisioner
- Automate large installation
- Nordic specific feature

nRF5 SDK for Mesh and other resources

- Available now at
 - <https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF5-SDK-for-Mesh>
 - Or search "nRF5 SDK for Mesh"
- All the documents are "open", no NDA needed
 - <https://infocenter.nordicsemi.com>
- For any technical discussions, we invite you to join our DevZone
 - <https://devzone.nordicsemi.com>
- For more product information or news, please check
 - <https://www.nordicsemi.com>
- You can also follow us on Weibo
 - <https://www.weibo.com/NordicSemi>



Bluetooth mesh

Principles and Commissioning

Rick Chung

Customer Application
Engineer

Bluetooth Asia 2018