AN1318: IV Update in a Bluetooth Mesh Network

This application note provides background information on the sequence number and IV index in a Bluetooth mesh network and the IV Update and IV Index Recovery procedures. It then discusses how to implement IV Update functionality in a Bluetooth mesh application.

KEY POINTS

- Short introduction to Bluetooth mesh security and replay attacks
- Sequence number and IV index
- IV Update procedure
- IV Index Recovery procedure
- Bluetooth mesh APIs for IV update and recovery

1 Introduction

Security is mandatory in Bluetooth mesh and cannot be switched off or reduced in any way. The network security, application security, and device security are all addressed independently. Security in Bluetooth mesh networking protects against various threats and issues, including:

- Replay attacks, which are prevented by judicious use of sequence numbers.
- Man-in-the-middle attacks, which are protected against by using asymmetrical cryptography, such as the Elliptic Curve Diffie-Hellman (ECDH) key agreement protocol, during provisioning of a new device into the network.
- Trash-can attacks, which exploit discarded devices, by ensuring security keys get refreshed when necessary.

A replay attack is a technique whereby an eavesdropper intercepts and captures one or more messages and simply retransmits them later, with the goal of tricking the recipient into carrying out something which the attacking device is not authorized to do. An example, commonly cited, is that of a car's keyless entry system being compromised by an attacker, intercepting the authentication sequence between the car's owner and the car, and later replaying those messages to gain entry to the car and steal it.

Bluetooth mesh has protection against replay attacks. The basis for this protection is the use of two cryptographic values in the mesh network called the Sequence Number (SEQ) and IV Index, respectively. Nodes keep track of the sequence numbers in the messages they receive, and together with the IV index of the network can detect and discard replayed messages.

1.1 Prerequisites

You should have:

- A general understanding of Bluetooth mesh concepts such as nodes and elements.
- Installed and be familiar with using the following:
  - Simplicity Studio v5.0.0 or above
  - Bluetooth Mesh SDK v2.0.0 or above

If you need to familiarize yourself with any of these concepts, the following may be useful:

- Bluetooth Mesh Networking - An Introduction for Developers
- QSG176: Bluetooth® Mesh SDK v2.x Quick-Start Guide

2 Sequence Number and IV Index

A Nonce is a number which may only be used once. Each time a message is encrypted, it is given a new nonce value. The nonce has various parts to it, including a sequence number and a value known as the IV Index. To ensure nonce values are unique for each new message encryption, the sequence number inside a nonce must not be allowed to wrap around while IV index remains unchanged.

The sequence number is a 24-bit value that allows an element to transmit 16,777,216 messages before repeating a nonce. If an element transmits a message on average once every second, then these sequence numbers would be exhausted after 194 days. To enable a mesh network to operate for longer periods of time than the sequence number space allows, an additional 4-octet value called the IV (Initialization Vector) Index is defined that is included in the security nonce.

The IV Index is a 32-bit value that is a shared network resource (that is, all nodes in a mesh network share the same IV Index value and use it for all subnets they belong to). Its purpose is to provide entropy (randomness) in the calculation of message Nonce values. At the same frequency of one message every second, the lifetime of the network using the IV Index would measure in billions of years.

Read Sections 3.8.3 and 3.8.4 of the Bluetooth Mesh Profile Specification v1.0.1 for further information.

Before the sequence number approaches the maximum value (0xFFFFFF), the element updates the IV Index using the IV Update procedure. The IV Index starts at 0x00000000 and is incremented during the IV Update procedure.

2.1 Sequence Number Increments

Each element increases the sequence number by one for every Network Protocol Data Unit (PDU) sent out to the network. The sequence number must be stored in non-volatile memory to ensure the sequence numbers that have been used will not be reused. However, writing every sequence number increment to flash memory is likely too many writes and would probably wear out flash memory during the lifetime of a product. The solution for this issue depends on Bluetooth mesh stack implementations.

In our implementation, the stack stores the sequence number in flash memory at a fixed interval when the sequence number is a multiple of a configurable value. Given that a device may lose power unexpectedly during the interval, the stored sequence number may be behind the sequence number actually used last, and using the stored value directly would lead to sequence number reuse. To avoid using the sequence numbers that have been used, the stack increases the stored sequence number by the configured interval value on a reset. The stack won't increase the sequence number if the increment of the stored sequence number was 0.

The stack will not transmit any Network PDUs from an element if its sequence number has reached the max value 0xFFFFFF. A node should initiate the IV Update procedure whenever it determines it is at risk of exhausting its sequence number values within a maximum of 96 hours, or if it determines that another node is in this situation.

2.2 IV Update Procedure

The IV Update procedure should be performed before the sequence number is exhausted. The procedure updates the IV index to a new value that will be used for subsequent communication in the mesh network. Once the IV Update procedure completes, the sequence number is reset to 0 on every element of every node in the network.

The IV Update procedure is initiated by any node in a primary subnet. If a node on a primary subnet receives an update on the primary subnet, it propagates the IV update to all other subnets. If a node on a primary subnet receives an IV update on any other subnet, the update is ignored.

The IV Index is shared within a mesh network via Secure Network beacons. IV updates received on a subnet are processed and propagated to that subnet. The propagation happens by the node transmitting Secure Network beacons with the updated IV Index for that particular subnet. A node that receives an IV update may or may not be able to update its IV index depending on the IV Update procedure state and the time since the last IV update.

- When a node is added to a mesh network, it is in the Normal Operation state. During the Normal Operation state, the IV Update Flag in the Secure Network beacon and in the Friend Update message is set to 0.
- After 96 hours of operating in the Normal Operation state, a node may initiate the IV Update procedure by transitioning to the IV Update in Progress state. When a node transitions from the Normal Operation state to the IV Update in Progress state, the IV Index on the node is incremented by one.
- During the IV Update in Progress state, the IV Update Flag in the Secure Network beacon and in the Friend Update message is set to 1.
- A node in the Normal Operation state receives a Secure Network beacon with the IV Update Flag set to 1 and an IV index equal to its current IV index + 1. The node then transitions to the IV Update in Progress state and updates its current IV index.

- After at least 96 hours and before 144 hours of operating in the IV Update in Progress state, the node transitions back to the Normal Operation state and resets its sequence number to 0x000000.
- A node must not start an IV Update procedure more often than once every 192 hours.

Read Section 3.10.5 of the Bluetooth Mesh Profile Specification v1.0.1 for further information.

2.3 IV Index Recovery Procedure

If a node is absent from a mesh network for a period of time, it may have missed IV index updates and is not able to communicate with other nodes. In this case, the node can scan for a Secure Network beacon, which contains the Network ID and the current IV Index, to recover the IV index. The IV Index Recovery procedure sets the current IV index and the IV Update procedure state of a node from the values in the Secure Network beacon.

- A node in the Normal Operation state receives a Secure Network beacon with an IV index greater than its current IV index + 1. It then may initiate an IV Index Recovery procedure.
- A node in the Normal Operation state receives a Secure Network beacon with the IV Update Flag set to 0 and an IV index equal to its current IV index + 1. It then may update its IV index without going to the IV Update in Progress state, or it may initiate an IV Index Recovery procedure, or it may ignore the Secure Network beacon. The node makes the choice depending on the time since the last IV update.
- After the IV Index Recovery procedure has updated the IV Index, the 96-hour time limits for changing the IV Update procedure state, as defined in the IV Update procedure, do not apply.
- A node must not execute more than one IV Index Recovery procedure within a period of 192 hours.
- A node in the Normal Operation state receives a Secure Network beacon with an IV index less than its current IV index or greater than its current IV index + 42. It ignores the Secure Network beacon. This allows a node to be away from the mesh network for 48 weeks. A node that is away from a network for longer than 48 weeks must be reprovisioned.

Read Section 3.10.6 of the Bluetooth Mesh Profile Specification v1.0.1 for further information.

2.4 IV Update Procedure Example

A summary of the IV Update procedure is provided in the table below.

IV Index	IV Update Flag	IV Update Procedure State	IV Index Accepted	IV Index used when transmitting
n	0	Normal	n-1, n	n
m (m=n+1)	1	In Progress	m-1, m	m-1
m	0	Normal	m-1, m	m

The following example illustrates the transition of the IV Update procedure state and IV index updates in the IV Update procedure. It will help explain the table above. You will also see from this example that the nodes before, during and after the IV Update procedure are able to communicate based on their Tx and Rx IV indices.

1. Node A and B are in the Normal Operation state and have the IV index 80 (n = 80, IV Update Flag = 0).

Node A	Normal Operation		Node B
IV Index = 80 (Normal) Tx: 80, Rx: 79, 80	Secure Network Beacon (IV Index = **80**, **Normal**)	→	IV Index = 80 (Normal) Tx: 80, Rx: 79, 80

2. Node A initiates the IV Update procedure and then Node B performs the IV Update procedure (m = 81, IV Update Flag = 1).
    1. Node A transitions to the IV Update in Progress state and increments its IV index by 1.
    2. Node B receives a Secure Network beacon with IV index = 81 and IV Update Flag = 1, transitions to the IV Update in Progress state and increments its IV index by 1.

Node A	Start IV Update		Node B
IV Index = 81 (Update) Tx: 80, Rx: 80, 81			IV Index = **80** (**Normal**) Tx: 80, Rx: 79, 80
IV Index = 81 (Update) Tx: 80, Rx: 80, 81	Secure Network Beacon (IV Index = **81**, **Update**)	→	IV Index = **81** (**Update**) Tx: 80, Rx: 80, 81

3. Node A and B transition to the Normal Operation state (m = 81, IV Update Flag = 0).
    1. Node A transitions to the Normal Operation state.
    2. Node B receives a Secure Network beacon with IV index = 81 and IV Update Flag = 0 and transitions to the Normal Operation state.

Node A	Finish IV Update		Node B
IV Index = 81 (Normal) Tx: 81, Rx: 80, 81			IV Index = **81** (**Update**) Tx: 80, Rx: 80, 81
IV Index = 81 (Normal) Tx: 81, Rx: 80, 81	Secure Network Beacon (IV Index = **81**, **Normal**)	→	IV Index = **81** (**Normal**) Tx: 81, Rx: 80, 81

3 IV Update Functionality Implementation

The IV Update procedure described in Section 2.2 IV Update Procedure and the IV Index Recovery procedure described in Section 2.3 IV Index Recovery Procedure have been implemented in the Bluetooth mesh stack. However, the stack cannot determine how long the remaining sequence numbers might last. The application should determine when a node may run out of sequence numbers and should initiate the IV Update procedure before sequence numbers are exhausted. If a node is absent from a mesh network for a period of time, the application should enable IV index recovery mode to set the IV Index value autonomously.

Secure network beacons are used to carry required information and are transmitted in the primary network. The nodes in the primary network that need to request IV index updates must enable secure network beacon broadcasting. This is usually configured by the provisioner. Note that some Bluetooth mesh mobile applications may not support configuration of the secure network beacon state. The application can call the `sl_btmesh_test_set_local_config()` API to set the state locally.

The following is a summary of the items that should be included in your system to implement IV Update functionality properly:

- Secure network beacons enabled on nodes.
- Code to track time across resets and set the time since the last IV update at booting.
- Code to determine sequence number exhaustion and trigger an IV index update.
- Code to check the network IV index and handle IV index recovery mode.

See the *Silicon Labs Bluetooth Mesh API Reference Manual* for all the BGAPI information.

3.1 Tracking Time Across Resets

The stack uses a timer to calculate the duration required to execute an IV Update or IV Index Recovery procedure. The timer is maintained in RAM and is reset whenever the device resets. This may cause an IV Update or IV Index Recovery procedure to execute later than expected. IV updates would not occur if frequent resets or power cycles are encountered in the application.

The application should track time across resets or power cycles. This is usually done by storing the time in flash memory regularly. The application then should call the following API to set the timer in the stack.

- `sl_btmesh_node_set_iv_update_age()`

The stack generates the following event when the IV Update procedure state has changed. If you did not call the `sl_btmesh_test_set_ivupdate_state()` API to change the state forcefully, this event indicates the IV Update procedure is ongoing (the value of the state is 1) or an IV update has completed (the value of the state is 0). The timer in the stack is reset to 0 when an IV update completes. The application should synchronize the time with the timer on this event when the value of the state is 0.

- `sl_btmesh_evt_node_changed_ivupdate_state_id`

3.2 Determining Sequence Number Exhaustion

As described in Section 2.1 Sequence Number Increments, the stack uses an interval value to determine sequence number storing and to add to the stored sequence number on a reset. The interval value can be configured by the structure member `pstore_write_interval_elem_seq` of the `__mesh_memory_config` variable in *sl_btmesh_dcd.c*. The value of this setting must be a power of 2. You may decrease the value if the device resets or power cycles regularly.

Every element of a node keeps a sequence number counter. The application should call the following APIs to get the number of remaining sequence numbers or the current sequence number of an element.

- `sl_btmesh_node_get_seq_remaining()`
- `sl_btmesh_node_get_element_seqnum()`

The maximum value of the sequence number is 0xFFFFFF. The application should know how often messages are transmitted to the network and therefore be able to determine if any element is about to exhaust its available sequence numbers. The application then should request the stack to initiate the IV Update procedure. The application must ensure the available sequence numbers can run for at least 96 hours when initiating the IV Update procedure. This time period can be extended to a safe period due to regular device resets or irregularity of message transmission.

3.3 Initiating IV Update

The application should call the following API to request the stack to initiate the IV Update procedure.

- `sl_btmesh_node_request_ivupdate()`

The `sl_btmesh_evt_node_changed_ivupdate_state_id` event is generated when the IV Update procedure has started or completed. The application can use this event to monitor the IV Update procedure state and the current IV index.

3.4 Recovering IV Index

The application should call the following API to enable the IV index recovery mode.

- `sl_btmesh_node_set_ivrecovery_mode()`

The IV index recovery mode should be enabled when the node may have missed IV index updates and need to perform the IV Index Recovery procedure. In this case, the following event is generated. The application should check the received network IV index and the current IV index that come with the event and enable the IV index recovery mode only when the received network IV index is greater than the current IV index and is equal to or less than the current IV index + 42.

- `sl_btmesh_evt_node_ivrecovery_needed_id`

If a real time clock is available in the device, the application can also check the IV index increment with the time elapsed. The time since the last IV update should be greater than the IV index increment multiplied by 192 hours minus 96 hours.

The `sl_btmesh_evt_node_changed_ivupdate_state_id` event is generated when the IV Index Recovery procedure has completed. The application should disable the IV index recovery mode on this event.

3.5 IV Update Testing

There are time limits on the IV Update and IV Index Recovery procedures so testing on Bluetooth mesh applications would be inefficient. The IV Update test mode removes these time limits. The application can call the following API to enable the IV Update test mode.

- `sl_btmesh_test_set_ivupdate_test_mode()`

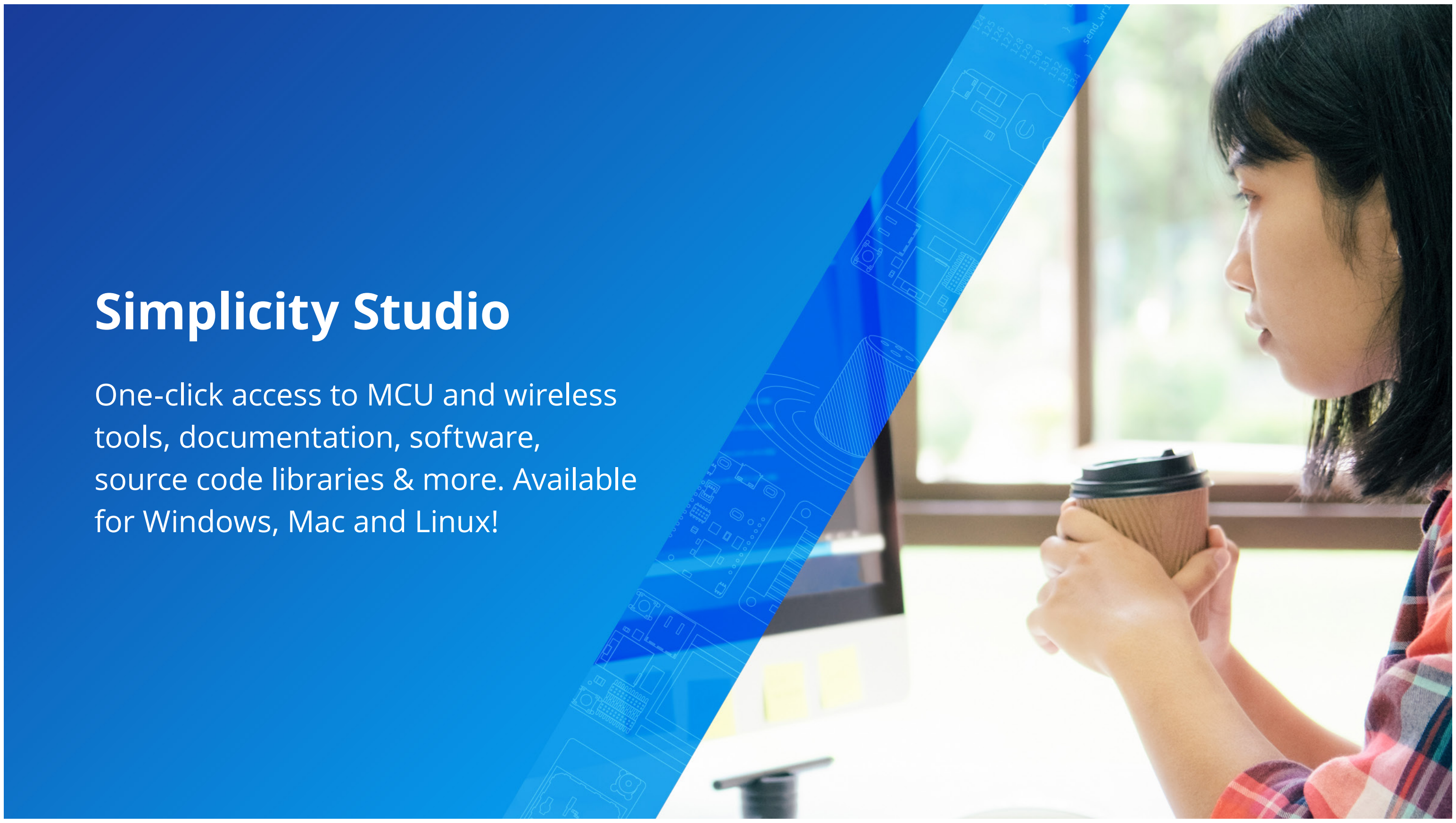
The following APIs may be useful when testing the IV Update functionality.

- `sl_btmesh_test_set_ivupdate_state()` – forces the node to transition to the IV Update in Progress state or to the Normal Operation state.
- `sl_btmesh_test_set_iv_index()` – changes the node's current IV index to any value.
- `sl_btmesh_test_set_local_config()` – changes a state of the Configuration Server model in the node. With the `sl_btmesh_node_beacon` parameter, enables or disables Secure Network Beacons broadcasting.
- `sl_btmesh_node_set_beacon_reporting()` – enables reporting to monitor secure network beacons. This can be used to diagnose the network IV index and the IV Update procedure state.

4 Additional Resources

Consult the following resources for additional information.

- QSG176: Bluetooth® Mesh SDK v2.x Quick-Start Guide
- Silicon Labs Bluetooth Mesh API Reference Manual
- Bluetooth Mesh Networking - An Introduction for Developers
- Bluetooth Mesh Profile Specification v1.0.1


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

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

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

www.silabs.com