Bluetooth Core Specification v5.1

M. Woolley

2019-01-28

Outline

Direction Finding
- Overview
- Technical Details

GATT Caching Enhancements
- Background
- Improved Caching Strategy
- Better State Management

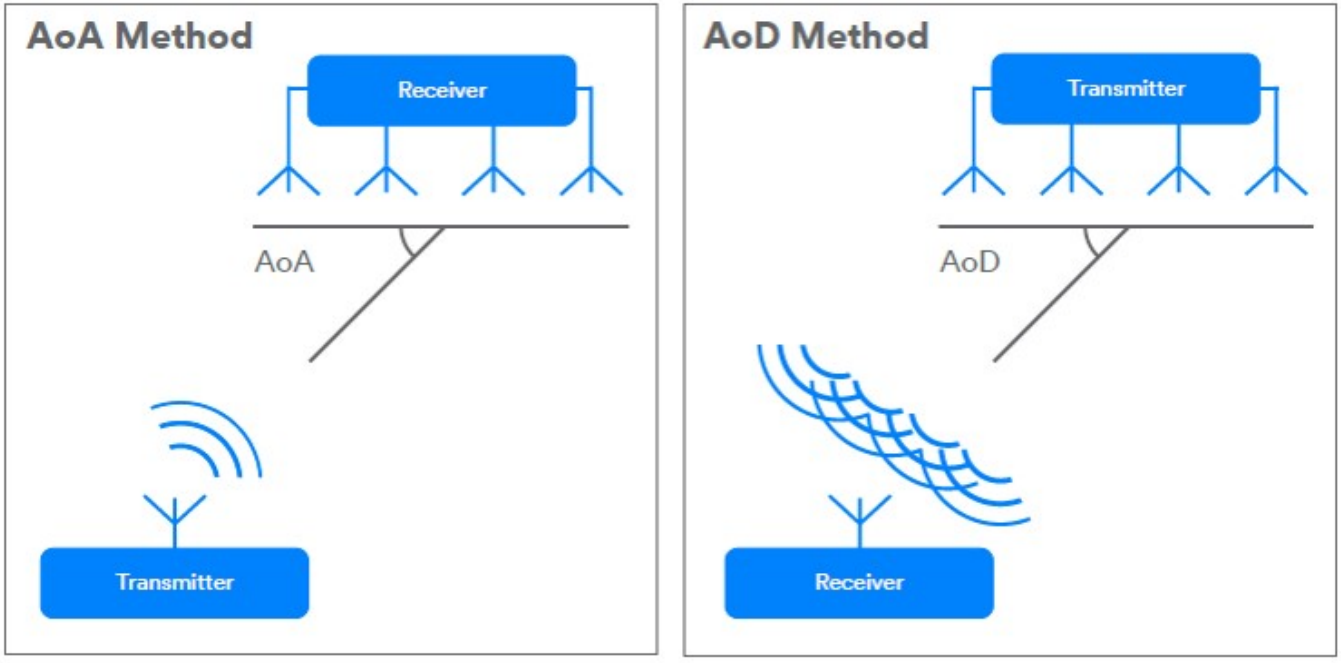
Advertising Enhancements 1
- Background
- Improved Packet Collision Avoidance

Advertising Enhancements 2
- Background
- The Power of Two

Direction Finding

Overview

- Bluetooth proximity solutions and positioning systems currently use signal **strength** to estimate **distance**
- A new direction finding feature in Bluetooth Core Specification v5.1 makes it possible for Bluetooth devices to determine the **direction** of a Bluetooth signal transmission



Direction Finding

Overview

- This new feature offers two different methods for determining the angle that a Bluetooth signal is being transmitted from with a high degree of accuracy
  - Angle of Arrival (AoA)
  - Angle of Departure (AoD)
- Antenna array included in the receiving device when the AoA method is used and in the transmitting device when using AoD
- Each of the techniques requires one of the two communicating devices to have an array of multiple antennae

Direction Finding

Overview

- Bluetooth v5.1 gives the BLE controller in the **receiving device** the ability to generate data that can then be used to calculate the directional angle to the transmitting device
  - When the associated profiles have been released, Bluetooth developers will be able to exploit it
  - E.g. For real-time locating systems (RTLS) and indoor positioning systems (IPS)
- It also has the potential to enhance Bluetooth **proximity** solutions by determining device direction
  - E.g. Directional item finding
  - E.g. Point of interest information solutions

Direction Finding

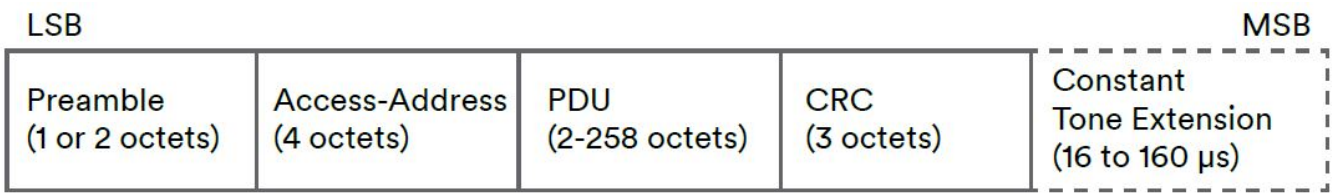
Technical Details

- The Bluetooth direction finding feature uses In-Phase and Quadrature (IQ) sampling to measure the phase of radio waves incident upon an antenna
- Sampled data is passed up the stack via the Host Controller Interface (HCI) where it will apply a suitable algorithm to the sampled data to calculate the direction
  - Once associated profiles are available, application developers will implement algorithms for calculating angles from IQ samples

Direction Finding

Technical Details

- To support IQ sampling and the use of IQ samples by higher layers in the stack, the link layer (LL) and HCI have each changed
- At the link layer, a new field called the Constant Tone Extension (CTE) has been defined
- This field contains a sequence of 1s, and is not included in the CRC calculation



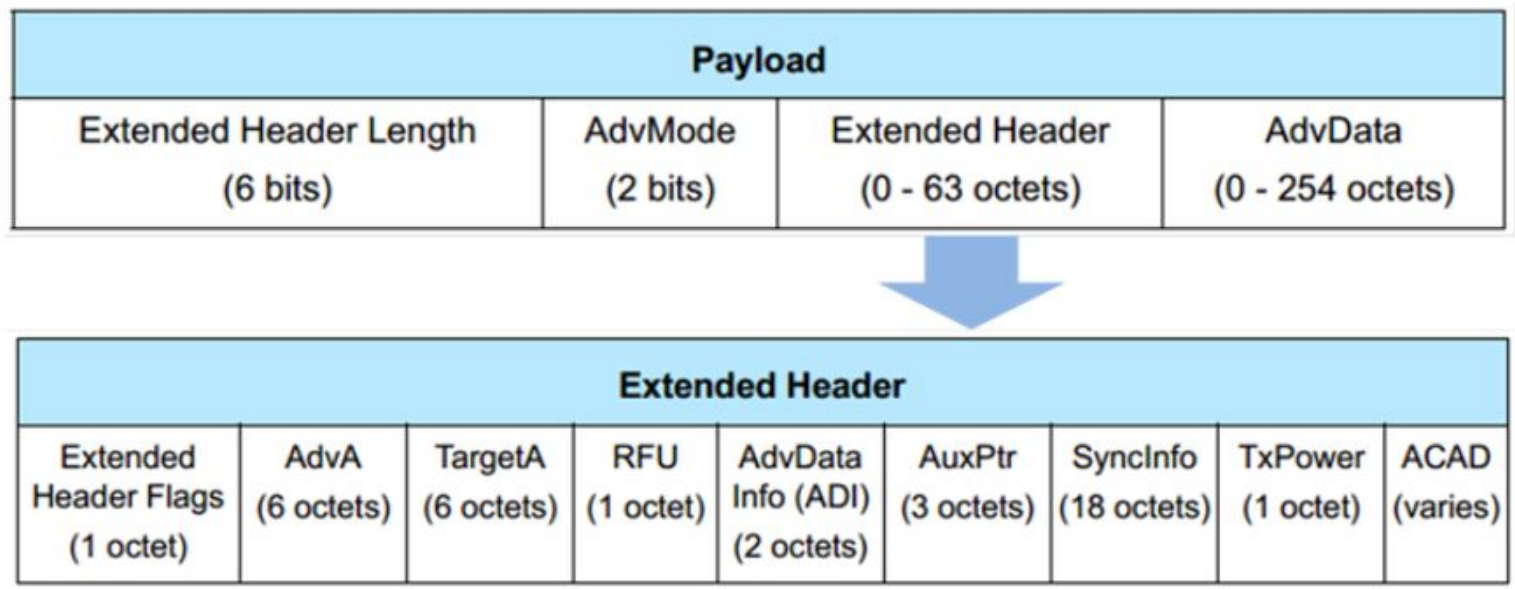
Direction Finding

Technical Details

- CTE can be used in both connectionless and connection-oriented scenarios
  - For connectionless use, the periodic advertising feature is required (since deterministic timing in the sampling process is important) and CTE is appended to AUX_SYNC_IND PDUs
  - For connection-oriented use, new PDUs LL_CTE_REQ and LL_CTE_RSP have been defined
- In either case, there are new HCI PDUs that allow the configuration of various aspects of CTE PDUs, such as the CTE length, length of the antenna switching pattern, and antenna IDs

참고: Advertisement Frames in v5.0

- Four PDU types
  - ADV_EXT_IND
  - AUX_ADV_IND
  - AUX_SYNC_IND
  - AUX_CHAIN_IND



Payload			
Extended Header Length (6 bits)	AdvMode (2 bits)	Extended Header (0 - 63 octets)	AdvData (0 - 254 octets)

Extended Header								
Extended Header Flags (1 octet)	AdvA (6 octets)	TargetA (6 octets)	RFU (1 octet)	AdvData Info (ADI) (2 octets)	AuxPtr (3 octets)	SyncInfo (18 octets)	TxPower (1 octet)	ACAD (varies)

GATT Caching Enhancements

Background

- All BLE connected devices use the Generic Attribute Profile (GATT)
- GATT devices contain a database known as the **attribute table**
- The attribute table contains GATT **service, characteristic, and descriptor** structural details and values
- Entries in the attribute table are identified by attribute handles

GATT Caching Enhancements

Background

- GATT clients must perform a procedure known as **service discovery** to acquire details of the attribute table on the remote GATT server device that the client has connected to
- The client can then use these details, including the identifying attribute handles in subsequent Attribute Protocol (ATT) interactions with the server

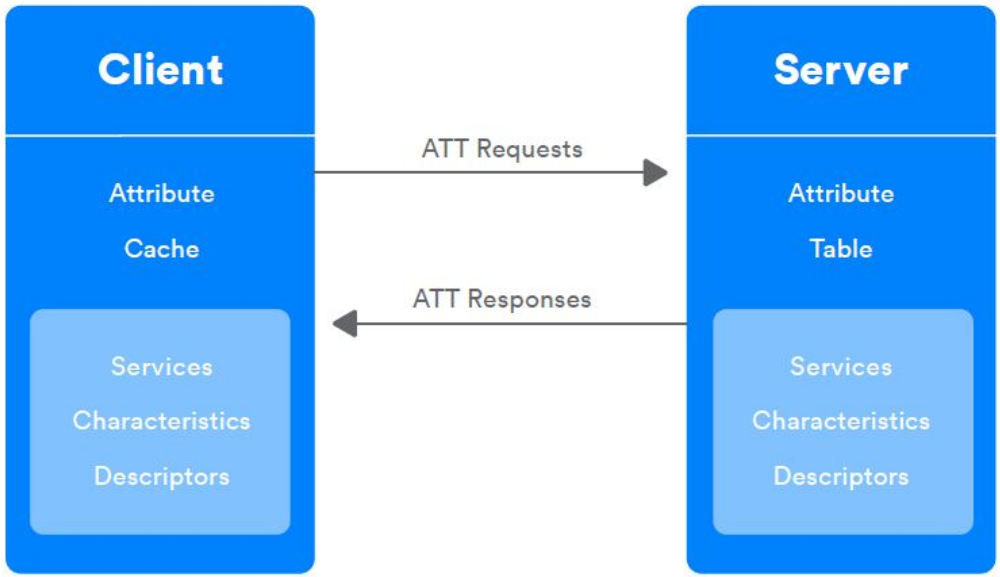


Figure 3 - Service Discovery and Attribute Caching

GATT Caching Enhancements

Background

- Some devices do not change their attribute table structure throughout their life
  - The GATT services, characteristics, and descriptors present in the table will always be the same, and only values of characteristics or descriptors will change
- Other devices do change their attribute table
- Previously, server had to keep track of every connected client to notify any change
- The GATT server could inform a connected client that its attribute table had changed by sending an ATT indication to the client

GATT Caching Enhancements

Background

- The client replied with an ATT confirmation and performed service discovery to synchronize its attribute cache with that of the server
- To avoid the GATT server needing to keep track of every client that ever connected to it, and whether or not each client had been informed of the latest attribute table change, clients and servers that have no trusted relationship (i.e. are not bonded) were required to perform service discovery **every time they connect**
  - This rule can cause energy efficiency and user experience issues for some types of products

GATT Caching Enhancements

Background

- In addition, beyond making a single attempt to inform the client that the attribute table had changed using the ATT Service Changed indication, there was no further state management carried out with respect to the client's view of the attribute table vs the server's
- The approach allowed a race condition in the communication between client and server, with respect to attribute table changes and general ATT interactions to exist
  - It was possible for a client to time-out whilst waiting for a Service Changed indication after connecting to the server, proceed to send general ATT PDUs, and then receive a Service Changed indication

GATT Caching Enhancements

Improved Caching Strategy

- To save time and energy for service discovery, v5.1 defines **attribute caching**
- Clients can skip service discovery if nothing has changed
  - Previously, server had to keep track of every connected client to notify any change
- Bluetooth v5.1 allows clients without a trusted relationship with a server to retain their attribute cache **across connections** and resolves the race condition issue
- Two new characteristics, each a member of the Generic Attribute Service, have been introduced: **Database Hash and Client Supported Features**

GATT Caching Enhancements

Improved Caching Strategy

- Clients which do not have a trusted relationship with the server may now cache the attribute table **across connections**
  - If the client supports the new Database Hash characteristic, as indicated by the client updating a flag in the server's Client Supported Features characteristic
- The Database Hash characteristic allows the client to ask the server if anything has changed, rather than relying on the server telling it using a Service Changed indication
- The client reads its value immediately after establishing a connection

GATT Caching Enhancements

Improved Caching Strategy

- The client may cache the Database Hash value and subsequently use it to determine whether or not the remote attribute table has changed
  - If it has changed, the client performs service discovery again
  - If it has not, it does not need to
- This offers a major user-experience and energy efficiency benefit to some device types
- If the database hash from **another** connected device is the same as the one associated with the client's attribute cache, there is no need to perform service discovery

GATT Caching Enhancements

Improved Caching Strategy

- A state machine defines whether or not the client view of the attribute table and the server view of its attribute table are in sync
  - As such, whether or not the client needs to perform service discovery
- The revised specification for attribute caching introduces the rigorously defined concept of Robust Caching that formalizes this state machine and introduces mechanisms for using it
- Clients are said to be in the **change-aware** state or are **change-unaware**

GATT Caching Enhancements

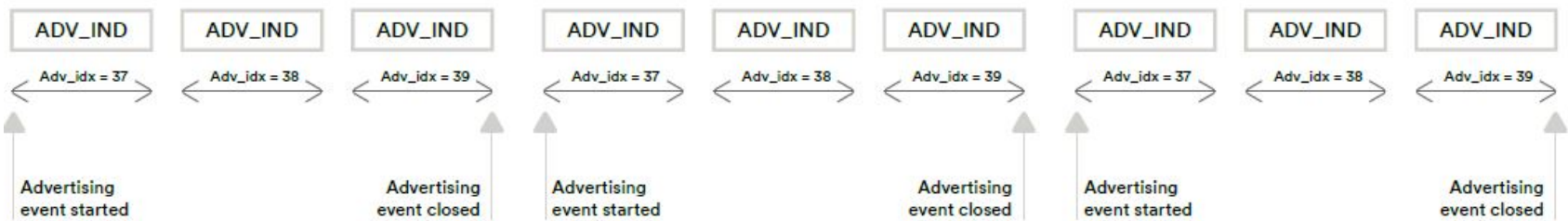
Improved Caching Strategy

- The server will ignore all ATT commands received from the client while it is in the change-unaware state
- A number of events can transition the client's state to change-aware
  - Server receiving an ATT confirmation to a Service Changed indication it had previously sent
  - Server having notified the client using the **Database Out Of Sync** error and subsequently receiving some other ATT PDU from the client
- From the client's point of view, if it moves to the change-unaware state it will not use its attribute cache, regarding it as invalid

Advertising Enhancements 1

Background

- In Bluetooth Core Specification v5.0, advertising uses channels in the sequence 37 then 38 then 39, in strict order
- To lessen the possibility of persistent packet collisions, where two or more devices advertise on the same channel in an overlapping time period, Bluetooth v5.0 stipulates that the time between consecutive advertising events must include a random delay of between 0 and 10ms



Advertising Enhancements 1

Improved Packet Collision Avoidance

- In Bluetooth 5.1, devices in the advertising state are no longer required to select advertising channels in a strict and unchanging sequence
- In busy radio environments, the randomization of advertising channel indices further reduces the potential for advertising packet collisions occurring



Advertising Enhancements 2

Background

- Bluetooth v5.0 introduced periodic advertising that uses deterministic scheduling of advertising events and provides a procedure that devices can use to synchronize their scanning with the advertising schedule of another device
  - Synchronization can make the scanning device more energy efficient and can make possible some use cases that require precise timing in the exchange of data
- Remote device advertises AUX_ADV_IND PDUs which contain a field called SyncInfo
  - SyncInfo contains everything the receiving device needs to know to synchronize with the periodic advertising of AUX_SYNC_IND PDUs performed by the remote device from that point on
- This periodic advertising synchronization procedure can be a relatively expensive operation, however

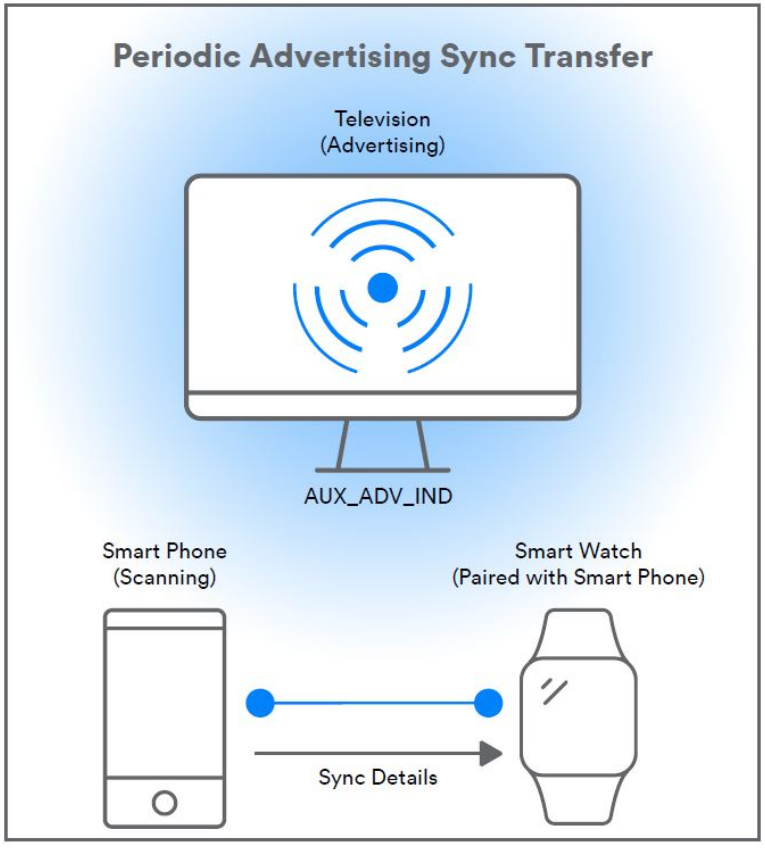
Advertising Enhancements 2

The Power of Two

- Some device types, with limited power, may not be able to afford the energy cost associated with the periodic advertising synchronization procedure or may have limitations in duty cycle or scan time that prevent it from working
- The new Periodic Advertising Sync Transfer (PAST) feature allows another, less constrained device to perform the synchronization procedure and then pass the acquired synchronization details over a point-to-point Bluetooth Low Energy connection to the other, constrained device

Advertising Enhancements 2

The Power of Two



- For example, a smartphone could scan for AUX_SYNC_IND packets from a TV and then pass them over a connection to an associated smart watch so that the watch can then benefit from using periodic advertising and scanning to acquire data from the TV