Chapter 4: More WICED BLE

Time 2 Hours

After completing chapter 4 you will have all the required knowledge to create the most basic WICED Bluetooth Low Energy Peripheral.

3.1 Notify & Indicate 2

3.2 Other Characteristic Descriptors 2

3.1 Security – Pairing – Bonding 2

3.2 Authentication & Authorization 2

3.3 Wiced\_bt\_cfg.c 2

3.4 Buffer Pools 2

3.5 Advertising packet fields 2

3.6 GATT Service Discovery 2

3.7 Mesh 2

3.8 Central 2

3.9 Non-GATT DB Based Attribute Protocols 2

3.10 GATT MTU 3

3.11 More Bluetooth Management Events 3

3.12 More GATT Events 3

3.13 OTA Bootloading 3

3.14 Low Power 3

3.15 WICED Chips 3

3.16 Terms 3

# Notify & Indicate

# Other Characteristic Descriptors

## Client Characteristic Configuration

## Characteristic User Description

## CHARACTERISTIC\_PRESENTATION\_FORMAT

# Security – Pairing – Bonding

BLE has two security modes, and several levels in each mode. They are:

|  |  |  |  |
| --- | --- | --- | --- |
| **Security** | **Level 1** | **Level 2** | **Level 3** |
| Mode 1 | No security | Unauthenticated  Encrypted | Authenticated  Encrypted |
| Mode 2 | Unauthenticated  Data Signed | Authenticated  Data Signed | N/A |

Authentication is the process of identifying a device and deciding whether a connection will be allowed. It can be done in one of several ways depending on the capabilities of the devices. The possible capabilities are:

1. No Input, No Output
2. Display Only
3. Display
4. Display: Yes/No
5. Keyboard Only

Need to understand these better and explain the possible options depending on the capabilities. What is display only vs. display yes/no vs display?

Need to understand/explain what data signed means. How is this different from encryption?

Need details on authentication and encryption schemes.

Once two BLE devices have established a connection (including authentication and key exchange if necessary), they are considered Paired. If the authentication information and keys are stored in memory by both devices, then the devices are Bonded. Devices that are bonded can connect in the future without going through the pairing process again.

The whole process looks like this:



In Bluetooth v4.2, privacy 1.2 was introduced. This involves using a 48-bit resolvable private address (RPA) that can be changed frequently (every 1 second) to prevent tracking. Only peer devices that have the 128-bit identity resolving key (IRK) of a BLE device can connect to it.

# Authentication & Authorization

# Wiced\_bt\_cfg.c

# Buffer Pools

# Advertising packet fields

## Using the advertising packet to actually get connected

## iBeacon

## Eddystone

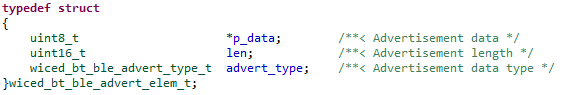
Lots of different information can be sent in an advertisement packet (e.g. device name, device appearance, primary service UUID etc.), but the total payload size is limited to a maximum of 31 bytes. The first element sent is the flags which is required. Each element (including the flags) uses the first byte to specify its payload length, the second byte to specify the type of data that follows, and the remaining bytes for the actual data.

For example, if you want to send flags, a 128-bit service UUID, and an 8-character device name, you would have:

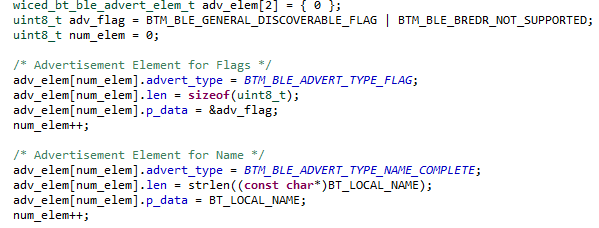
* Flags: 1-byte size + 1-byte Flag type + 1-byte Flag data = 3 bytes
* UUID: 1-byte size + 1-byte UUID type + 16-bytes for the 128-bit UUID data = 18 bytes
* Name: 1-byte size + 1-byte Name type + 8-byte Name data = 10 bytes
* Total = 3 + 18 + 10 = 31 bytes

You can send a shortened part of the name if necessary (i.e. *BTM\_BLE\_ADVERT\_TYPE\_NAME\_SHORT* instead of *BTM\_BLE\_ADVERT\_TYPE\_NAME\_COMPLETE*).

The advertisement data is set by creating an array of type *wiced\_ble\_advert\_elem\_t*. Each entry in the array is a structure that contains the type of the entry, the size of the entry, and a pointer to the data:



For example, a two-entry advertisement packet with Flags (for a BLE only device that is generally discoverable) and the name of the device would look like this:



Once the array is populated, the function *wiced\_bt\_ble\_set\_raw\_advertisement\_data* is called with the number of advertisement packet entries and a pointer to the array like this:



Finally, advertisements are started by calling *wiced\_bt\_start\_advertisements*:



# Scan Respose

# GATT Service Discovery

# Mesh

# Central

# Non-GATT DB Based Attribute Protocols

# GATT MTU

# More Bluetooth Management Events

# Privacy

# More GATT Events

# OTA Bootloading

# Multirole devices

# Direct Test Mode

# Low Power

# WICED Chips

# Terms

Central

Peripheral

GATT

GAP

Service

Profile

Characteristic

Attribute