Developer Study Guide

Bluetooth® Internet Gateways

Orientation Guide

Release	:	2.0.1
Document Version:		2.0.0
Last updated	:	24th June 2021

Contents

1. Revision History

Version	Date	Author	Changes
2.0.0	24th June 2021	Martin Woolley Bluetooth SIG	**Release:** Added new module covering gateways for Bluetooth mesh networks. Modularised the study guide to accommodate the two main cases of LE Peripherals and mesh networks. Updated to be based on Python 3 rather than Python 2. **Document**: This document is new in this release.

2. Introduction and Orientation

Welcome to the Bluetooth® Internet Gateway Developer Study Guide.

This is a self-paced, educational resource for software developers and technical architects. It aims to introduce the reader to the topic of *gateways*, provide some insight into their potential capabilities and probable architecture, and offers some experience in building two types of Bluetooth internet gateway from the ground up.

The study guide is modular in design and this orientation guide will help you decide where to start and what path to follow, depending on your area of interest.

Module 02, *First Steps* covers the basic concepts and issues, requirements and logical architecture of a gateway and you should start with this module regardless of your particular area of interest.

Physical architecture and hands-on implementation steps are then covered in separate modules depending on whether you want your gateway to be for LE Peripheral devices or Bluetooth mesh networks. Module 03 covers Bluetooth internet gateways which enable internet access to Bluetooth LE Peripheral devices and module 04 covers Bluetooth internet gateways which enable the exchange of messages with Bluetooth mesh networks from the internet.

For both scenarios, full working solutions together with associated web applications which use the example Bluetooth internet gateways are also provided in this study guide and they act as complete, end to end illustrations of gateways and their use.

Generally we'll follow the sequence of a standard development project, starting by considering possible requirements for a Bluetooth internet gateway and then thinking about both the logical and physical architecture for a solution to those requirements. At various stages, you will be asked to offer your own thoughts on some key questions, before proceeding to the next part of the guide.

With a solution architecture determined, you will next be invited to assemble, integrate or develop, components of a gateway and an associated application, which uses the gateway and provides a user interface.

Once we have a working gateway and application, we'll spend some time considering security, identifying issues and countermeasures, many of which you will then implement to some extent.

Finally, the topic of scalability as it applies to Bluetooth internet gateways will be explored.

Security and scalability are covered in modules 05 and 06 respectively and both of them should be considered common modules, equally applicable to gateways for LE Peripherals and to gateways for Bluetooth mesh networks.

3. Goals

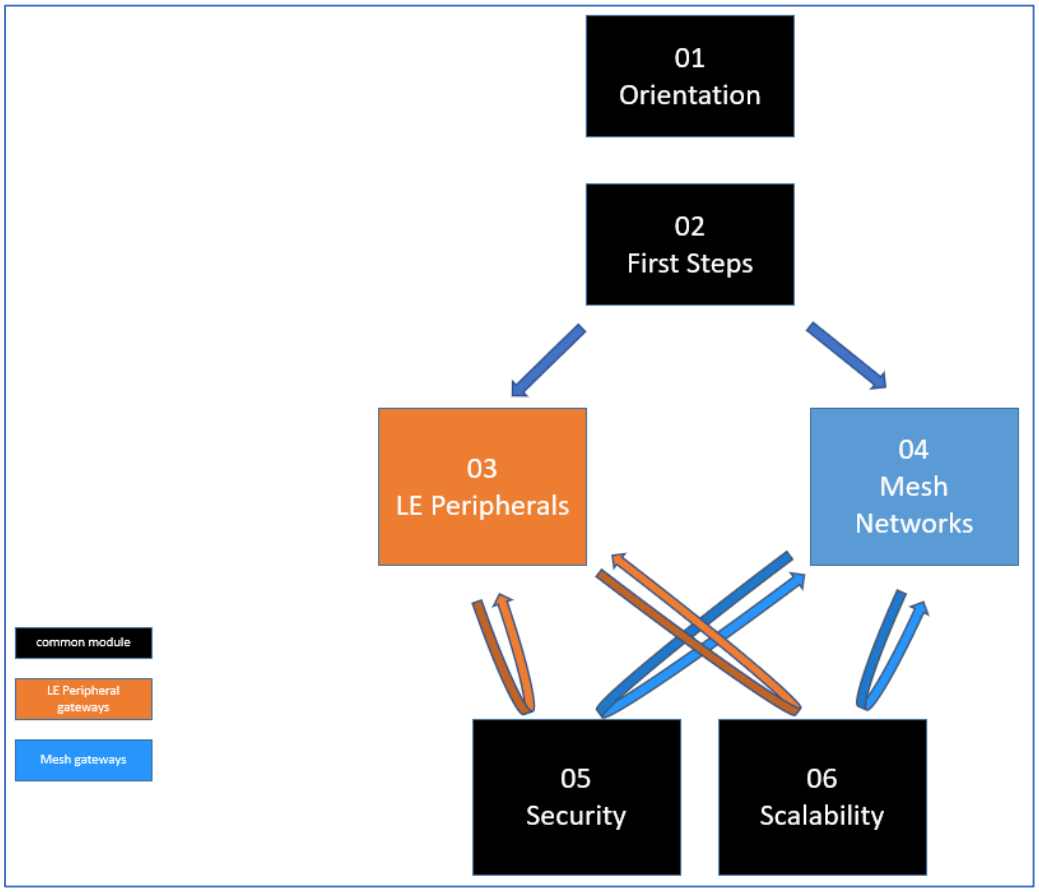
After completing the work in this study guide, you should:

- Be able to answer the question "what is a Bluetooth internet gateway"?
- Be able to explain various ways in which Bluetooth devices could be interfaced to the Internet or other TCP/IP network.

- Understand that requirements for a gateway may vary and be able to describe those which are important to you.
- Be able to describe the various types of Bluetooth device whose particular capabilities might require particular measures to accommodate them in a gateway architecture.
- Have an appreciation of what can be involved in designing and implementing your own gateway solution from scratch.
- Understand the importance of security in gateways and be able to identify and describe some of the key issues. You should be able to discuss some ways in which these issues can be addressed.
- Understand what the term “scalability” means in the context of a Bluetooth internet gateway and have some knowledge of architectures which are inherently more scalable than others.

4. Module Sequence

The following diagram illustrates the relationship between modules and the order in which you should study them. When studying modules 03 or 04, you’ll be directed to look at common modules 05 (security) and 06 (scalabilty), after which you should return to your point of departure in module 03 or 04.



You should now proceed to study module 02 *First Steps*. Good luck with your studies!

Appendix A - Python 2 to Python 3 Migration

Version 1 of this study guide used Python 2. When version 2 of the study guide was created, it was decided to migrate all code to use Python 3 instead since Python 2 is essentially deprecated. For readers who made use of version 1 of the study guide and have a gateway for LE Peripherals already implemented, the following notes should make it easy to migrate the code to use Python 3.

Only three types of change were required to migrate the study gudie code from Python 2 syntax to Python 3.

A1. Shebang

The old Python 2 shebang at the top of scripts looked like this:

```
#!/usr/bin/python
```

For Python 3 this must be changed to:

```
#!/usr/bin/python3
```

A2. Print statements

Python 2 print statements do not require arguments to be inside parantheses. For example:

```
if 'REQUEST_METHOD' in os.environ:
    result = {}
    if os.environ['REQUEST_METHOD'] != 'GET':
        print 'Status: 405 Method Not Allowed'
        print
        print "Status-Line: HTTP/1.0 405 Method Not Allowed"
        print
    else:
        print "Content-Type: application/json;charset=utf-8"
        print

else:
    print "ERROR: Not called by HTTP"
```

Python 3 requires print statements to use paranetheses so the example code must become:

```
if 'REQUEST_METHOD' in os.environ:
    result = {}
    if os.environ['REQUEST_METHOD'] != 'GET':
        print('Status: 405 Method Not Allowed')
        print()
        print("Status-Line: HTTP/1.0 405 Method Not Allowed")
        print()
    else:
        print("Content-Type: application/json;charset=utf-8")
        print()

else:
    print("ERROR: Not called by HTTP")
```

A3. Exceptions

Python 2 exception handling syntax looks like this:

```
try:
    services_discovered = bluetooth_gatt.get_services(bdaddr)

except bluetooth_exceptions.StateError, e:
    result = {}
    result['result'] = e.args[0]
```

The syntax of the *except* clause changed at Python 3 and migrated code now looks like this:

```
try:
    services_discovered = bluetooth_gatt.get_services(bdaddr)

except bluetooth_exceptions.StateError as e:
    result = {}
    result['result'] = e.args[0]
```