Generic Attribute Profile (GATT)

Bluetooth® Test Suite

- **Revision**: GATT.TS.5.1.1
- **Revision Date**: 2019-08-01
- **Group Prepared By**: BTI
- **Feedback Email**: bti-main@bluetooth.org
This document, regardless of its title or content, is not a Bluetooth Specification subject to the licenses granted by the Bluetooth SIG Inc. ("Bluetooth SIG") and its members under the Bluetooth Patent/Copyright License Agreement and Bluetooth Trademark License Agreement.

THIS DOCUMENT IS PROVIDED "AS IS" AND BLUETOOTH SIG, ITS MEMBERS, AND THEIR AFFILIATES MAKE NO REPRESENTATIONS OR WARRANTIES AND DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, THAT THE CONTENT OF THIS DOCUMENT IS FREE OF ERRORS.

TO THE EXTENT NOT PROHIBITED BY LAW, BLUETOOTH SIG, ITS MEMBERS, AND THEIR AFFILIATES DISCLAIM ALL LIABILITY ARISING OUT OF OR RELATING TO USE OF THIS DOCUMENT AND ANY INFORMATION CONTAINED IN THIS DOCUMENT, INCLUDING LOST REVENUE, PROFITS, DATA OR PROGRAMS, OR BUSINESS INTERRUPTION, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, AND EVEN IF BLUETOOTH SIG, ITS MEMBERS, OR THEIR AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document is proprietary to Bluetooth SIG. This document may contain or cover subject matter that is intellectual property of Bluetooth SIG and its members. The furnishing of this document does not grant any license to any intellectual property of Bluetooth SIG or its members.

This document is subject to change without notice.

Copyright © 2010–2019 by Bluetooth SIG, Inc. The Bluetooth word mark and logos are owned by Bluetooth SIG, Inc. Other third-party brands and names are the property of their respective owners.
Contents

1 Scope .................................................................................................................................................. 8

2 References, Definitions, and Abbreviations ...................................................................................... 9
   2.1 References ................................................................................................................................... 9

3 Test Suite Structure (TSS) .................................................................................................................. 10
   3.1 Overview ................................................................................................................................... 10
   3.2 Test Strategy ............................................................................................................................... 10
   3.2.1 Server Testing Configuration ............................................................................................... 11
   3.2.2 Client Testing Configuration ................................................................................................. 12
   3.3 Test Library and Test Databases ............................................................................................... 12
   3.3.1 Test Database Library ........................................................................................................... 12
   3.3.2 Test Database Requirements ............................................................................................... 13
   3.4 Test Groups ................................................................................................................................ 14
   3.4.1 Server Configuration ............................................................................................................. 14
   3.4.2 Discover Services and Characteristics .................................................................................. 14
   3.4.3 Read Characteristic Values and Characteristic Descriptors .............................................. 14
   3.4.4 Write Characteristic Values and Characteristic Descriptors ............................................. 14
   3.4.5 Notify or Indicate Characteristic Values ............................................................................... 14
   3.4.6 Generic Attribute Profile Services ....................................................................................... 14
   3.4.7 GATT Timeouts ..................................................................................................................... 14
   3.4.8 Generic Profile Attributes .................................................................................................... 14
   3.5 Attribute Protocol Testing .......................................................................................................... 14

4 Test Cases .......................................................................................................................................... 15
   4.1 Introduction ................................................................................................................................. 15
   4.1.1 Test Case Identification Conventions ................................................................................... 15
   4.1.2 Conformance .......................................................................................................................... 15
   4.1.3 Pass/Fail Verdict Conventions ............................................................................................... 16
   4.2 Setup Preambles .......................................................................................................................... 16
   4.2.1 ATT Bearer .............................................................................................................................. 16
   4.2.1.1 Setup ATT Bearer over BR/EDR ....................................................................................... 17
   4.2.1.2 Setup ATT Bearer over LE ................................................................................................. 17
   4.2.2 Characteristic Configuration .................................................................................................. 17
   4.2.2.1 Characteristic Configuration for Notification – IUT as Server ...................................... 17
   4.2.2.2 Characteristic Configuration for Indication – IUT as Server ........................................... 17
   4.2.2.3 Characteristic Configuration for Notification – IUT as Client ...................................... 17
   4.2.2.4 Characteristic Configuration for Indication – IUT as Client ........................................... 18
   4.2.3 Encryption Key Size ................................................................................................................ 18
   4.2.4 Exchange MTU ......................................................................................................................... 18
   4.3 Server Configuration ..................................................................................................................... 19
   4.3.1 GATT/CL/GAC/BV-01-C [Server Configuration - by Client] ............................................. 19
   4.3.2 GATT/SR/GAC/BV-01-C [Server Configuration - of Server] .............................................. 20
   4.4 Discovery ..................................................................................................................................... 22
   4.4.1 GATT/CL/GAD/BV-01-C [Discover All Primary Services - by Client] ............................... 22
   4.4.2 GATT/SR/GAD/BV-01-C [Discover All Primary Services - from Server] ........................... 24
   4.4.3 GATT/CL/GAD/BV-02-C [Discover Primary Services by Service UUID – by Client] ....... 25
   4.4.4 GATT/SR/GAD/BV-02-C [Discover Primary Services by Service UUID - from server] ....... 27
   4.4.5 GATT/CL/GAD/BV-03-C [Find Included Services – by client] ........................................... 28
4.4.6 GATT/SR/GAD/BV-03-C [Find Included Services – from Server] ..................................................31
4.4.7 GATT/CL/GAD/BV-04-C [Discover All Characteristics of a Service – by Client] ..........................32
4.4.8 GATT/SR/GAD/BV-04-C [Discover All Characteristics of a Service – from Server] ..................34
4.4.9 GATT/CL/GAD/BV-05-C [Discover Characteristics by UUID – by Client] .................................35
4.4.10 GATT/SR/GAD/BV-05-C [Discover Characteristics by UUID – from server] ............................37
4.4.11 GATT/CL/GAD/BV-06-C [Discover All Characteristic Descriptors – by client] ..........................39
4.4.12 GATT/SR/GAD/BV-07-C [Discover Primary Services using SDP - by client] ............................40
4.4.13 GATT/CL/GAD/BV-08-C [Discover Services by UUID using SDP - by Client] ........................42
4.4.14 GATT/SR/GAD/BV-08-C [Discover All Characteristic Descriptors – from Server] ..................44
4.4.15 GATT/SR/GAD/BV-07-C [Discover Primary Services using SDP - from Server] ......................46
4.4.16 GATT/SR/GAD/BV-08-C [Discover Services by UUID using SDP - from Server] ......................48
4.5 Read .................................................................51
4.5.1 GATT/CL/GAR/BV-01-C [Read Characteristic Value - by Client] ..................................................51
4.5.2 GATT/CL/GAR/BI-01-C [Read Characteristic Value – Invalid Handle] ........................................52
4.5.3 GATT/CL/GAR/BI-02-C [Read Characteristic Value – Read Not Permitted] .................................53
4.5.4 GATT/CL/GAR/BI-03-C [Read Characteristic Value – Insufficient Authorization] .....................55
4.5.5 GATT/CL/GAR/BI-04-C [Read Characteristic Value – Insufficient Authentication] .................56
4.5.6 GATT/CL/GAR/BI-05-C [Read Characteristic Value – Insufficient Encryption Key Size] ............57
4.5.7 GATT/SR/GAR/BV-01-C [Read Characteristic Value - from Server] .........................................58
4.5.8 GATT/SR/GAR/BI-01-C [Read Characteristic Value- Read Not Permitted Response] ..................59
4.5.9 GATT/SR/GAR/BI-02-C [Read Characteristic Value - Invalid Handle Response] ......................60
4.5.10 GATT/SR/GAR/BI-03-C [Read Characteristic Value – Insufficient Authorization] .................61
4.5.11 GATT/SR/GAR/BI-04-C [Read Characteristic Value – Insufficient Authentication] ...............62
4.5.12 GATT/SR/GAR/BI-05-C [Read Characteristic Value – Insufficient Encryption Key Size] ............63
4.5.13 GATT/CL/GAR/BV-03-C [Read Using Characteristic UUID - by Client] ..................................64
4.5.14 GATT/CL/GAR/BI-06-C [Read Using Characteristic UUID – Read Not Permitted] ..................66
4.5.15 GATT/CL/GAR/BI-07-C [Read Using Characteristic UUID – Attribute Not Found] ...............67
4.5.16 GATT/CL/GAR/BI-09-C [Read Using Characteristic UUID – Insufficient Authorization] ............68
4.5.17 GATT/CL/GAR/BI-10-C [Read Using Characteristic UUID – Insufficient Authentication] .........69
4.5.18 GATT/CL/GAR/BI-11-C [Read Using Characteristic UUID – Insufficient Encryption Key Size] ....70
4.5.19 GATT/SR/GAR/BV-03-C [Read using Characteristic UUID - from Server] .............................71
4.5.20 GATT/SR/GAR/BI-06-C [Read Characteristic by UUID - Read Not Permitted Response] ............73
4.5.21 GATT/SR/GAR/BI-07-C [Read Characteristic by UUID - Attribute Not Found Response] ...........74
4.5.22 GATT/SR/GAR/BI-08-C [Read Characteristic by UUID – Invalid Handle Response] .................75
4.5.23 GATT/SR/GAR/BI-09-C [Read Using Characteristic UUID – Insufficient Authorization] ............76
4.5.25 GATT/SR/GAR/BI-11-C [Read Using Characteristic UUID – Insufficient Encryption Key Size] ......78
4.5.26 GATT/CL/GAR/BV-04-C [Read Long Characteristic Value - by Client] .................................79
4.5.27 GATT/CL/GAR/BI-12-C [Read Long Characteristic Value – Read not permitted] ....................83
4.5.28 GATT/CL/GAR/BI-13-C [Read Long Characteristic Value – Invalid Offset] .............................84
4.5.29 GATT/CL/GAR/BI-14-C [Read Long Characteristic Value – Invalid Handle] ..........................85
4.5.31 GATT/CL/GAR/BI-16-C [Read Long Characteristic Value – Insufficient Authentication] ..........88
4.5.32 GATT/CL/GAR/BI-17-C [Read Long Characteristic Value – Insufficient Encryption Key Size] .......89
4.5.33 GATT/SR/GAR/BV-04-C [Read Long Characteristic Value - from Server] ..............................90
4.5.34 GATT/SR/GAR/BI-12-C [Read Long Characteristic Value - Read Not Permitted Response] ......92
4.5.35 GATT/SR/GAR/BI-13-C [Read Long Characteristic Value - Invalid Offset Response] ............93
4.5.36 GATT/SR/GAR/BI-14-C [Read Long Characteristic Value - Invalid Handle Response] ............94
4.5.39 GATT/SR/GAR/BI-17-C [Read Long Characteristic Value – Insufficient Encryption Key Size] ........97
4.5.40 GATT/CL/GAR/BV-05-C [Read Multiple Characteristic Values – by client] ................................98
4.5.41 GATT/CL/GAR/BI-18-C [Read Multiple Characteristic Values – Read not permitted] ..........99
4.5.42 GATT/CL/GAR/BI-19-C [Read Multiple Characteristic Values – Invalid Handle] .................100
4.5.43 GATT/CL/GAR/BI-20-C [Read Multiple Characteristic Values – Insufficient Authorization] ....101
4.5.44 GATT/CL/GAR/BI-21-C [Read Multiple Characteristic Values – Insufficient Authentication] ...102
4.5.45 GATT/CL/GAR/BI-22-C [Read Multiple Characteristic Values – Insufficient Encryption Key Size] ....103
4.5.46 GATT/SR/GAR/BV-05-C [Read Multiple Characteristic Values – from Server] .....................104
4.5.47 GATT/SR/GAR/BI-18-C [Read Multiple Characteristic Values – Read not permitted] ..........106
4.5.48 GATT/SR/GAR/BI-19-C [Read Multiple Characteristic Values – Invalid Handle] .................107
4.5.49 GATT/SR/GAR/BI-20-C [Read Multiple Characteristic Values – Insufficient Authorization] ....108
4.5.50 GATT/SR/GAR/BI-21-C [Read Multiple Characteristic Values – Insufficient Authentication] ....109
4.5.51 GATT/SR/GAR/BI-22-C [Read Multiple Characteristic Values – Insufficient Encryption Key Size] ....110
4.5.52 GATT/CL/GAR/BV-06-C [Read Characteristic Descriptors – by Client] ...............................111
4.5.53 GATT/SR/GAR/BV-06-C [Read Characteristic Descriptors – from Server] .........................112
4.5.54 GATT/CL/GAR/BV-07-C [Read Long Characteristic Descriptor - by Client] .......................113
4.5.55 GATT/CL/GAR/BI-34-C [Read Characteristic Value – Invalid Transport Access over BR/EDR] ....117
4.5.56 GATT/CL/GAR/BI-35-C [Read Characteristic Value – Invalid Transport Access over LE] ........118
4.5.57 GATT/SR/GAR/BV-07-C [Read Long Characteristic Descriptor - from Server] ....................120
4.5.58 GATT/SR/GAR/BV-08-C [Read Behind Long Characteristic Descriptor - from Server] ..........121
4.5.59 GATT/SR/GAR/BI-34-C [Read Multiple Characteristic Value - Invalid Transport Access over LE] ....122
4.5.60 GATT/SR/GAR/BI-35-C [Read Characteristic Value - Invalid Transport Access over BR/EDR] ....124
4.6 Write........................................................................................................................................125
4.6.1 GATT/CL/GAW/BV-01-C [Write without Response - by Client] .............................................125
4.6.2 GATT/SR/GAW/BV-01-C [Write Without Response - to Server] ..........................................126
4.6.3 GATT/CL/GAW/BV-02-C [Write without Response with Authentication - by Client] ............127
4.6.5 GATT/SR/GAW/BI-01-C [Write without Response with Authentication – Invalid Signature] ....130
4.6.6 GATT/CL/GAW/BV-03-C [Write Characteristic Value - by Client] .........................................131
4.6.7 GATT/CL/GAW/BI-02-C [Write Characteristic Value – Invalid handle] ...............................133
4.6.8 GATT/CL/GAW/BI-03-C [Write Characteristic Value – Write Not Permitted] .......................134
4.6.11 GATT/CL/GAW/BI-06-C [Write Characteristic Value – Insufficient Encryption Key Size] ....137
4.6.13 GATT/SR/GAW/BI-02-C [Write Characteristic Value – Invalid Handle Response] ............139
4.6.18 GATT/CL/GAW/BV-05-C [Write Long Characteristic Value - by Client] .............................144
4.6.20 GATT/CL/GAW/BI-08-C [Write Long Characteristic Value – Write Not Permitted] ............146
4.6.23 GATT/CL/GAW/BI-12-C [Write Long Characteristic Value – Insufficient Authentication] ....150
4.6.27 GATT/SR/GAW/BI-08-C [Write Long Characteristic Value – Write Not Permitted Response] ....155
4.6.32  GATT/CL/GAW/BV-06-C [Characteristic Value Reliable Write - by Client] ........................................ 160
4.6.33  GATT/SR/GAW/BV-06-C [Characteristic Value Reliable Writes - to Server] ................................. 162
4.6.34  GATT/SR/GAW/BV-10-C [Nested Long Characteristic Value Reliable Writes - to Server] .............. 163
4.6.35  GATT/SR/GAW/BV-11-C [Characteristic Value Reliable Writes - No Pending Prepared Write Requests].......................................................................................................................................................... 165
4.6.36  GATT/SR/GAW/BV-07-C [Cancel Reliable Write Characteristic – from Server] ........................................ 166
4.6.37  GATT/CL/GAW/BV-08-C [Write Characteristic Descriptors – by Client] ........................................ 167
4.6.38  GATT/SR/GAW/BV-08-C [Write Characteristic Descriptors – from Server] ........................................ 168
4.6.40  GATT/CL/GAW/BI-32-C [Cancel Reliable Write Characteristic – by Client] ........................................ 171
4.6.43  GATT/SR/GAW/BV-09-C [Write Long Characteristic Descriptors – from Server] ................................. 175
4.7  Notification and Indication ......................................................................................................................... 179
4.7.1  GATT/CL/GAN/BV-01-C [Characteristic Value Notification - to Client] ............................................... 179
4.7.2  GATT/SR/GAN/BV-01-C [Characteristic Value Notification - by Server] ........................................ 180
4.7.3  GATT/SR/GAI/BV-01-C [Characteristic Value Indication - by Server] ........................................ 181
4.7.4  GATT/CL/GAI/BV-01-C [Confirm Characteristic Value Indication - by Client] ........................................ 182
4.8  Generic Attribute Profile Services ............................................................................................................. 183
4.8.1  GATT/CL/GAS/BV-01-C [Service Changed Characteristic – to Client] .................................................. 183
4.8.2  GATT/CL/GAS/BV-02-C [Reading the Database Hash Characteristic] ................................................. 184
4.8.3  GATT/CL/GAS/BV-03-C [Enabling the Robust Caching] ................................................................. 185
4.8.4  GATT/SR/GAS/BV-01-C [Service Changed Characteristic – from Server] .......................................... 186
4.8.5  GATT/SR/GAS/BV-02-C [Computing and Returning the Database Hash Characteristic] ..................... 187
4.8.6  GATT/SR/GAS/BV-03-C [Maintaining a Client Supported Features Characteristic Instance for each Client] 189
4.8.7  GATT/SR/GAS/BV-04-C [Maintaining Client Supported Features Characteristic Values for Bonded Devices] 190
4.8.8  GATT/SR/GAS/BV-05-C [Handling Client Requests on Unsynchronized Database] ......................... 192
4.8.9  GATT/SR/GAS/BV-06-C [Handling Client Requests after a Hash Read or Service Changed Indication – Database Changed during Connection] ..................................................... 195
4.8.10 GATT/SR/GAS/BV-07-C [Handling Client Requests after a Hash Read or Service Changed Indication – Database Changed between Connections] ........................................ 197
4.9  GATT Transaction Timeouts ...................................................................................................................... 199
4.9.1  GATT/CL/GAT/BV-01-C [Read Characteristic Value – Server Timeout] ............................................. 199
4.9.2  GATT/CL/GAT/BV-02-C [Write Characteristic Value – Server Timeout] ........................................ 201
4.9.3  GATT/SR/GAT/BV-01-C [Handle Value Indication – Client Timeout] ............................................. 202
4.10  Generic Profile Attributes ....................................................................................................................... 203
4.10.1 GATT/CL/GPA/BV-12-C [Characteristic Format Descriptors – from Client] ........................................ 203
4.10.2 GATT/SR/GPA/BV-12-C [Characteristic Presentation Format Descriptors – from Client] ............ 208
4.10.3 GATT/CL/GPA/BV-11-C [Characteristic Aggregate Format – by Client] ........................................ 211
4.11  Multiple ATT Bearer Support .................................................................................................................. 217
4.11.1 GATT/SR/GPM/BV-01-C [Client Configuration Characteristic per ATT Bearer] ............................ 217
4.12  Unsupported Requests and Commands ................................................................................................... 218

Bluetooth SIG Proprietary
4.12.1 GATT/SR/UNS/BI-01-C [Unsupported ATT Requests on Server]........................................................................218
4.12.2 GATT/SR/UNS/BI-02-C [Unsupported ATT Commands on Server]..........................................................219

5 Test Case Mapping........................................................................................................................................220

6 ANNEX: Generic GATT Integrated Tests (GGIT) .........................................................................................227
6.1 Identification Conventions.........................................................................................................................227
6.2 GGIT Inputs ..................................................................................................................................................227
6.2.1 Example Usage – Blood Pressure Service TS.......................................................................................228
6.3 Server test procedures (SGGIT)..................................................................................................................229
6.3.1 SGGIT/SER [Service GGIT]....................................................................................................................229
6.3.2 SGGIT/CHA [Characteristic GGIT]........................................................................................................229
6.3.3 SGGIT/DES [Descriptor GGIT]................................................................................................................230
6.4 Client test procedures (CGGIT)..................................................................................................................230
6.4.1 CGGIT/SER [Service GGIT]....................................................................................................................231
6.4.2 CGGIT/CHA [Characteristic GGIT]........................................................................................................231
6.4.3 CGGIT/DES [Descriptor GGIT]................................................................................................................232

7 Revision History and Contributors ...........................................................................................................233
1 Scope

This Bluetooth document contains the Test Suite Structure (TSS) and Test Cases (TC) to test the Bluetooth Generic Attribute Profile Specification (GATT).

This material is also used to verify the Bluetooth Attribute Protocol Specification (ATT).

The objective of this test suite is to provide a basis for interoperability for Bluetooth devices giving a high probability of air interface interoperability between different manufacturers’ Bluetooth devices.
2 References, Definitions, and Abbreviations

2.1 References

This Bluetooth document incorporates, by dated or undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For the purpose of this Bluetooth document, the definitions and abbreviations in [1] and [2] apply.

[1] Generic Attribute Profile, Bluetooth 4.0 and later, Volume 3 Part G
[3] ICS Proforma for Generic Attribute Profile, GATT.ICS
[5] Attribute Protocol 4.0 and later, Volume 3 Part F
[8] ICS Proforma for Attribute Protocol, ATT.ICS
[10] Bluetooth Core Specification IXIT
3 Test Suite Structure (TSS)

3.1 Overview

The Generic Attribute Profile (GATT) is a user (client) of the Attribute Protocol (ATT) and the Generic Access Protocol (GAP). This is illustrated in Figure 3.1.

In Figure 3.1 the GATT layer under test may support only those GATT features needed by the included GATT Profile functionality, or it may be complete enough to support any GATT profile; e.g., a downloaded application.

3.2 Test Strategy

The test objectives are to verify functionality of the Generic Attribute Profile within a Bluetooth Host and enable interoperability between Bluetooth Hosts on different devices. The testing approach is to cover mandatory and optional requirements in the Profile specification and to match these to the support of the IUT as described in the ICS proforma.

Conformance testing is the appropriate test method to meet these intents. The basis for the test approach are the general concepts and conformance testing principles defined in ISO/IEC 9646-1 and ISO/IEC 9646-2, both are part of the OSI Conformance Testing Methodology and Framework (CTMF).

The conformance test equipment shall provide an implementation of the Radio Controller and the parts of the Host needed to perform the test cases defined in the Generic Attribute Profile Test Suite. For some test cases, it is necessary to stimulate the IUT from an Upper Tester. In practice, this may be implemented as a special test interface, an MMI, or another interface supported by the IUT.
The Generic Attribute Profile test suite contains Valid Behavior (BV) tests complemented with Invalid Behavior (BI) tests where required. The test coverage mirrored in the test suite structure is the result of a process that started with catalogued specification requirements that were logically grouped and assessed for testability enabling coverage in defined test cases.

The test suite structure is a tree:

1. Profile operations:
   - Discover Characteristics
   - Reading a Characteristic
   - Writing a Characteristic
   - Notification of Characteristics
   - Indicating a Characteristic, with Confirmation
2. Use of protocols:
   - Attribute Protocol
3. Generic Profile Attributes:
   - Characteristic Formats

### 3.2.1 Server Testing Configuration

The following configuration is recommended for testing server IUT:

![Sample database or database defined by IUT](image)

Figure 3.2: Generic Attribute Profile Server Test Configuration

The sample database of Generic Profile Attributes (services, characteristics, and descriptors) used by the IUT may be:

- Specified for use in testing [7].
- Specified by the IUT manufacturer in the IXIT [10].
3.2.2 Client Testing Configuration

The following configuration is recommended for testing client IUT:

![Diagram of Client Testing Configuration]

The sample database of Generic Profile Attributes (services, characteristics and descriptors) used by the Lower Tester is specified in Section 3.3. In tests of Discovery (Section 4.4) four or more distinct Sample Databases shall be used, derived from the Library of valid GATT database elements for Compliance (-C) testing as defined in Section 3.3.

The interface between the IUT and the Upper Tester may be:

- A man-machine interface
- Provided by the IUT manufacturer e.g., a wired connection, an API, etc.

3.3 Test Library and Test Databases

Tests of GATT clients shall use four or more sample databases derived from a Library.

3.3.1 Test Database Library

A Library of valid GATT database elements shall be assembled. These database elements shall include services, characteristics and characteristic descriptors. This Library shall include features required and optional in GATT servers:

1. At least 10 complex Services
2. Multiple included Services
3. Multiple included Characteristics
4. Complex Characteristics with multiple descriptors: name, properties, format, units
5. Variety of formats (Table 3-16[1])
6. Variety of properties (3.3.1.1[1])
7. Aggregate characteristics (3.3.3.6[1])
This Library will be updated for two events:

1. Add services and characteristics included in Adopted GATT-based profiles.
2. Add services and characteristics included in Qualified GATT servers.

### 3.3.2 Test Database Requirements

The following requirements apply to the set of databases used for testing:

1. One small test database whose services fit into a single minimum sized PDU, with only 16-bit UUIDs.
2. A set of three or more larger test databases, with 16-bit and 128-bit service UUIDs AND characteristic UUIDs in pseudo-random order. In one, the 16-bit ones are early and 128-bit ones are late; in one, that is reversed. In the third, they are completely random.
3. At least one of those has a Primary Service at the MAX handle (0xFFFF).
4. At least one primary service without any include or characteristic declarations which is not located at MAX handle (0xFFFF).
5. At least one of those has a Characteristic at the MAX handle (0xFFFF).
6. At least one of those has a Secondary Service.
7. Each has at least one each of 16-bit and 128-bit Primary Service UUID with multiple instances (the IOP database does); in the latter case, there should be more than enough for the found UUID/handle pairs to exceed the minimum ATT_MTU and overflow into multiple PDU.
8. All have some services that are simple, and some that include other services.
9. Each has at least one instance where the handle of an included service is before the handle of the including service.
10. At least one with one or more services that contain Included Services with both 16-bit and 128-bit Service UUIDs.
11. Each has instances of simple characteristics (no descriptors) and complex characteristics (multiple descriptors)
12. Each has instances of complex characteristics with 16-bit and 128-bit characteristic descriptor UUIDs, and these can be in scrambled order.
13. For any GATT client IUT whose ICS indicates support for a GATT-based Service, any test databases which are larger (see item 2 above) shall contain at least one instantiation of each GATT-based Service supported.
14. For any instantiation of a GATT-based Service defined by the Bluetooth SIG in a large test database (see item 2 above), the database shall contain at least two additional characteristics; the UUID for these characteristics shall be chosen randomly from 16-bit values not yet defined in Assigned Numbers:
   a) An additional characteristic whose handle is between the defined mandatory characteristics and any defined optional characteristics.
   b) An additional characteristic whose handle is after any defined optional characteristic.
15. For any instantiation of a GATT-based Service defined by the Bluetooth SIG in a large test database (see item 2 above), the database shall contain at least two additional characteristic descriptors; the UUID for these characteristic descriptors shall be chosen randomly from 16-bit values not yet defined in Assigned Numbers:
   a) An additional characteristic descriptor added to a defined mandatory characteristic.
   b) An additional characteristic descriptor added to an optional characteristic.
3.4 Test Groups

The following test groups have been defined.

3.4.1 Server Configuration
This test group covers Generic Attribute Profile Server configuration.

3.4.2 Discover Services and Characteristics
This test group covers Service and Characteristic discovery by Generic Attribute Profile clients and support of discovery by Generic Attribute Profile servers.

3.4.3 Read Characteristic Values and Characteristic Descriptors
This test group covers reading Characteristic Values and Descriptors by Generic Attribute Profile clients and support of reading these Characteristic Values and Descriptors by Generic Attribute Profile servers.

3.4.4 Write Characteristic Values and Characteristic Descriptors
This test group covers writing Characteristic Values and Descriptors by Generic Attribute Profile clients and support of writing those Characteristic Values and Descriptors by Generic Attribute Profile servers.

3.4.5 Notify or Indicate Characteristic Values
This test group covers Characteristic notification and indication by Generic Attribute Profile servers.

3.4.6 Generic Attribute Profile Services
This test group covers exposure of Generic Attribute Profile Characteristics by Generic Attribute Profile servers.

3.4.7 GATT Timeouts
This test group covers recovery from transaction timeouts.

3.4.8 Generic Profile Attributes
This test group covers the data structures defined for Generic Profile Attributes with GATT Characteristic Formats.

3.5 Attribute Protocol Testing
The Generic Attribute Profile can be used to test the Attribute Protocol. ATT requires use of GATT over BR/EDR or LE transports. When testing ATT, GATT is required as part of the Upper Tester. After analysis it was demonstrated that all features listed in the Attribute Protocol ICS can be tested using Test Cases in this Test Suite. In consideration of this, each test case in this specification contains references to both ATT and GATT.
4 Test Cases

4.1 Introduction

4.1.1 Test Case Identification Conventions

Test cases shall be assigned unique identifiers per the conventions in [2]. The convention used here is `<spec abbreviation>/<IUT role>/<class>/<feat>/<func>/<subfunc>/<cap>/<xx><nn><y>`.

Bolded ID parts shall appear in the order prescribed. Non-bolded ID parts (if applicable) shall appear between the bolded parts. The order of the non-bolded parts may vary from test suite to test suite, but shall be consistent within each individual test suite.

<table>
<thead>
<tr>
<th>Identifier Abbreviation</th>
<th>Spec Identifier &lt;spec abbreviation&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>GATT</td>
<td>Generic Attribute Profile</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Identifier Abbreviation</th>
<th>Role Identifier &lt;IUT role&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR</td>
<td>Server Role</td>
</tr>
<tr>
<td>CL</td>
<td>Client Role</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Identifier Abbreviation</th>
<th>Class Identifier &lt;class&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAC</td>
<td>Generic Attribute Configuration</td>
</tr>
<tr>
<td>GAD</td>
<td>Generic Attribute Discovery</td>
</tr>
<tr>
<td>GAR</td>
<td>Generic Attribute Read</td>
</tr>
<tr>
<td>GAW</td>
<td>Generic Attribute Write</td>
</tr>
<tr>
<td>GAN</td>
<td>Generic Attribute Notification</td>
</tr>
<tr>
<td>GAI</td>
<td>Generic Attribute Indication</td>
</tr>
<tr>
<td>GAS</td>
<td>Generic Attribute Profile Services</td>
</tr>
<tr>
<td>GAT</td>
<td>GATT Transaction Timeouts</td>
</tr>
<tr>
<td>GPA</td>
<td>Generic Profile Attributes</td>
</tr>
<tr>
<td>GPM</td>
<td>Generic Attribute Multiple ATT Bearers</td>
</tr>
<tr>
<td>UNS</td>
<td>Unsupported Requests and Commands</td>
</tr>
</tbody>
</table>

Table 4.1: GATT TC Class Naming Convention

4.1.2 Conformance

When conformance is claimed, all capabilities indicated as mandatory for this Specification shall be supported in the specified manner (process-mandatory). This also applies for all optional and conditional
capabilities for which support is indicated. All mandatory capabilities, and optional and conditional capabilities for which support is indicated, are subject to verification as part of the Bluetooth Qualification program.

The Bluetooth Qualification Program may employ tests to verify implementation robustness. The level of implementation robustness that is verified varies from one Specification to another and may be revised for cause based on interoperability issues found in the market.

Such tests may verify:

- That claimed capabilities may be used in any order and any number of repetitions that is not excluded by the Specification, OR
- That capabilities enabled by the implementations are sustained over durations expected by the use case, OR
- That the implementation gracefully handles any quantity of data expected by the use case, OR
- That in cases where more than one valid interpretation of the Specification exist, the implementation complies with at least one interpretation and gracefully handles other interpretations OR
- That the implementation is immune to attempted security exploits.

A single execution of each of the required tests is required in order to constitute a pass verdict. However, it is noted that in order to provide a foundation for interoperability, it is necessary that a qualified implementation consistently and repeatedly pass any of the applicable tests.

In any case, where a member finds an issue with the Test Plan Generator, the Test Case as described in the Test Suite, or with the Test System utilized, the Member is required to notify the responsible party via an errata request such that the issue may be addressed.

4.1.3 Pass/Fail Verdict Conventions

Each test case has an Expected Outcome section, which outlines all the detailed pass criteria conditions that shall be met by the IUT to merit a Pass Verdict.

The convention in this test suite is that, unless there is a specific set of fail conditions outlined in the test case, the IUT fails the test case as soon one of the pass criteria conditions cannot be met. If this occurs the outcome of the test shall be the Fail Verdict.

4.2 Setup Preambles

The procedures defined in this section are provided for information, as they are used in achieving the Initial Conditions in certain tests.

4.2.1 ATT Bearer

The ICS [3] specifies which transports are supported.

- If BR/EDR is specified in the ICS [3], the setup procedure defined in 4.2.1.1 is executed.
- If LE is specified in the ICS [3], the setup procedure defined in 4.2.1.2 is executed.
- If both BR/EDR and LE are specified, the test case shall be executed twice, once for each transport, unless a specific transport is specified in the test case initial conditions.
4.2.1.1  **Setup ATT Bearer over BR/EDR**

Preamble procedure:

1. Establish a BR/EDR transport connection between the IUT and the Lower Tester.
2. Establish an L2CAP channel (PSM 0x001F) between the IUT and the Lower Tester over that BR/EDR transport.

4.2.1.2  **Setup ATT Bearer over LE**

Preamble procedure:

1. Establish an LE transport connection between the IUT and the Lower Tester.
2. Establish an L2CAP channel 0x0004 between the IUT and the Lower Tester over that LE transport.

4.2.2  **Characteristic Configuration**

Tests for Characteristic Notification or Characteristic Indication may require characteristic configuration; see [1] Section 3.3.3.3. If required, the Lower Tester begins the test by configuring the server IUT to generate a Notification or an Indication of the selected Characteristic. If the server does not use the Client Characteristic Configuration descriptor then these preambles do not need to be executed.

All client test cases, which use a configuration as show in Figure 3.3, contain test procedure descriptions, Message Sequence Charts and expected results. These in turn use example message syntax between the Upper tester and the IUT. Those example messages are generic; there is no normative specification for these messages. The normative specifications are the functional descriptions for the test procedures and the expected results.

4.2.2.1  **Characteristic Configuration for Notification – IUT as Server**

Preamble procedure:

The Lower Tester has the necessary security permissions from the IUT to write the Client Characteristic Configuration descriptor.

The Lower Tester sends an `ATT_Write_Request` to the IUT. The handle selects the Client Characteristic Configuration descriptor. The value is set to 0x0001, to enable Notification.

4.2.2.2  **Characteristic Configuration for Indication – IUT as Server**

Preamble procedure:

The Lower Tester has the necessary security permissions from the IUT to write the Client Characteristic Configuration descriptor.

The Lower Tester sends an `ATT_Write_Request` to the IUT. The handle selects the Client Characteristic Configuration descriptor. The value is set to 0x0002, to enable Indication.

4.2.2.3  **Characteristic Configuration for Notification – IUT as Client**

Preamble procedure:

The IUT has the necessary security permissions from the Lower Tester to write the Client Characteristic Configuration descriptor.
The Upper Tester orders the IUT to send an *ATT_Write_Request* to the Lower Tester. The handle selects the Client Characteristic Configuration descriptor. The value is set to 0x0001, to enable Notification.

### 4.2.2.4 Characteristic Configuration for Indication – IUT as Client

Preamble procedure:

The IUT has the necessary security permissions from the Lower Tester to write the Client Characteristic Configuration descriptor.

The Upper Tester orders the IUT to send an *ATT_Write_Request* to the Lower Tester. The handle selects the Client Characteristic Configuration descriptor. The value is set to 0x0002, to enable Indication.

### 4.2.3 Encryption Key Size

Tests for insufficient encryption key size require an encrypted link with a key size less than the size required by an attribute. The encryption key size requirements of attributes are determined by the associated Profile.

Preamble procedure:

Establish an encrypted link over the LE transport between the IUT and the Lower Tester. For example, see test SM/MAS/EKS/BV-01-C in [6]. The key size is less than the size required by an attribute.

### 4.2.4 Exchange MTU

The GATT Exchange MTU sub-procedure is only applicable over the LE transport since the MTU size is negotiated using L2CAP channel configuration procedures for a BR/EDR physical link.

- **Reference**
  
  [1] 4.3.1

- **Preamble procedure:**
  
  ATT_MTU may be exchanged between IUT and the Lower Tester using the procedure described in the MSC below. The IUT may support either the GATT Client or GATT Server role as applicable to the test case. The IUT sets its RX MTU parameter to the value of TSPX_iut_max_rx_mtu in the IXIT [10]. This preamble can only be initiated once during a connection.
4.3 Server Configuration

The test group objective is to verify Generic Attribute Profile Server Configuration.

4.3.1 GATT/CL/GAC/BV-01-C [Server Configuration - by Client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can generate an Exchange MTU Request command to configure ATT_MTU over LE.

- **Reference**
  [1] 4.3.1
  [5] 3.4.2.1, 3.4.2.2

- **Initial Condition**
  The ICS indicates that the IUT supports ATT_MTU exchange.
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.
  The IXIT [10] specifies the value of maximum size of the attribute protocol message that the IUT can receive.
  The ICS indicates if the IUT can support ATT_Prepare_Write_Request.
  Lower Tester contains a writeable long characteristic of maximum size (512).

- **Test Procedure**
  1. The Lower Tester starts with the maximum value allowed (512). The sequence is:
  2. Send a request from the Upper Tester to the IUT to initiate MTU exchange; this may be of the form GATT_ExchangeMTURequest.
  3. The Lower Tester waits for an ATT_Exchange_MTU_Request from IUT.
  4. The Lower Tester sends an ATT_Exchange_MTU_Response to the IUT containing an ATT_MTU parameter set as specified in [5].
  5. If the IUT supports ATT_Prepare_Write_Request, the upper tests sends a request to the IUT to generate a prepare write request to a specified handle in the Lower Tester, providing more data than ATT_MTU.
  6. The Lower Tester waits for an ATT_Prepare_Write_Request, and checks its size.
• Expected Outcome

**Pass Verdict**

The IUT sends an ATT_Exchange_MTU_Request command to the Lower Tester with the client RX MTU size set to the value indicated in the IXIT [10].

The IUT receives an ATT_Exchange_MTU_Response sent by the Lower Tester, and recovers the ATT_MTU value.

If the IUT supports the feature, the IUT sends an ATT_Prepare_Write_Request with ATT_MTU-5 of data.

### 4.3.2 GATT/SR/GAC/BV-01-C [Server Configuration - of Server]

• **Test Purpose**

Verify that a Generic Attribute Profile server can accept an Exchange MTU Request and respond with an Exchange MTU Response over LE.

• **Reference**

[1] 4.3.1

[5] 3.4.2.1, 3.4.2.2

• **Initial Condition**

The ICS indicates that the IUT supports ATT_MTU exchange.

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

The IXIT [10] specifies the value of maximum size of the attribute protocol message that the IUT can receive.
The IXIT [10] indicates the handle, type and size of a readable long characteristic if supported by the IUT.

The IUT has at least one characteristic value that is:

a) longer than its (MTU - 1) if MTU < 512
b) 512 if MTU >= 512

- **Test Procedure**
  - This Test Procedure shall be run.
    - Once, where the Lower Tester starts with its ATT_MTU set to the minimum value (23), and
    - Once, where the Lower Tester starts with its ATT_MTU set to 512.
  
  For each test, the sequence is:

  Send ATT_Exchange_MTU_Request from the Lower Tester to the IUT to inform the server of the client’s maximum Rx MTU size (23 or 512).

  The Lower Tester waits for an ATT_Exchange_MTU_Response containing the ATT_MTU parameter.

  The Lower Tester sends an ATT_Read_Request to the IUT, to the handle specified in the IXIT [10].

  The Lower Tester waits for an ATT_Read_Response from the IUT.

  L2CAP Connection Established over selected transport (3.2).

  ATT_Exchange_MTU_Request
  (Code = 0x02, MTU = 0x0017 or 0x0200)

  ATT_Exchange_MTU_Response
  (Code = 0x03, ATT_MTU)

  ATT_Read_Request
  (Code = 0xA, handle)

  ATT_Read_Response
  (Code = 0x0B, value for size up to ATT_MTU-1)

  The Lower Tester verifies that MTU has been set as expected
• Expected Outcome

Pass Verdict

For each iteration:

- Once the ATT_Exchange_MTU request and response messages have been exchanged, verify that the ATT_MTU has been set to the minimum of the Client Rx MTU and Server Rx MTU values.
- The IUT responds to the ATT_Read_Request with an ATT_Read_Response. The Lower Tester verifies that MTU was exchanged successfully.

4.4 Discovery

The test group objective is to verify Generic Attribute Profile Discovery of Services and Service Characteristics.

4.4.1 GATT/CL/GAD/BV-01-C [Discover All Primary Services - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client discovers Primary Services in a GATT server.

• Reference

[1] 4.4.1
[5] 3.4.1.1, 3.4.4.9, 3.4.4.10

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure

The Upper Tester sends a request to the IUT to initiate Primary Service Discovery; this may be of the form GATT_DiscAllServices(starting handle, ending handle). The starting handle is 0x0001 and the ending handle is 0xFFFF.

This Test Procedure shall be run at least four times, using distinct Sample Databases derived from the Library defined in Section 3.3.
• Expected Outcome

Pass Verdict

The IUT sends ATT_Read_By_Group_Type_Requests to the Lower Tester and discovers services in the Lower Tester. The first command has starting handle = 0x0001, ending handle = 0xFFFF, and UUID = <primary service>. For any subsequent ATT_Read_By_Group_Type_Requests, the starting handle shall be set to one greater than the last End Group Handle in the ATT_Read_By_Type_Response.

The IUT sends ATT_Read_By_Group_Type_Requests to the Lower Tester until a response with End Group Handle = 0xFFFF is received, a desired service is found, or an ATT_Error_Response is received.

At the end of the test, the IUT indicates a completion message, e.g., GATT_DiscComplete. Prior to that completion message, the IUT shall report all Primary Service data: handle pairs and values (see [5] Table 3.25).
• Notes

Prior to each test performance, the IUT should clear the client’s attribute cache. This may be automated by the test equipment and user interaction may not be required.

4.4.2 GATT/SR/GAD/BV-01-C [Discover All Primary Services - from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support a search for all Primary Services it contains.

• Reference

[1] 4.4.1
[5] 3.4.1.1, 3.4.4.9, 3.4.4.10

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure

Send an initial ATT_Read_By_Group_Type_Request (starting handle=0x0001, Ending handle=0xFFFF, UUID = «Primary Service») from the Lower Tester to the IUT.

If the IUT returns an ATT_Read_By_Group_Type_Response, continue the test by sending another ATT_Read_By_Group_Type_Request (next starting handle, Ending handle=0xFFFF, UUID = «Primary Service») from the Lower Tester to the IUT.

Repeat sending ATT_Read_By_Group_Type_Requests until the IUT returns an ATT_Error_Response indicating AttributeNotFound, or the IUT returns an ATT_Read_By_Group_Type_Response containing handle 0xFFFF.
- Expected Outcome
  Pass Verdict

  For each `ATT_Read_By_Group_Type_Request`, the IUT sends a correctly formatted `ATT_Read_By_Group_Type_Response` to the Lower Tester or an `ATT_Error_Response` if there is no handle/UUID pair matching the request.

  The IUT reports all service UUIDs known to be contained in its database (via IXIT [10]).

  The IUT sends all `ATT_Read_by_Group_Type_Responses` to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).

4.4.3 GATT/CL/GAD/BV-02-C [Discover Primary Services by Service UUID – by Client]
- Test Purpose
  Verify that a Generic Attribute Profile client can discover Primary Services selected by service UUID, using 16-bit and 128-bit UUIDs.

- Reference
  [1] 4.4.2
  [5] 3.4.3.3, 3.4.3.4

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
• **Test Procedure**

  The IUT sends `ATT_Find_By_Type_Value_Requests` (starting handle, ending handle, attribute type, attribute value) to the Lower Tester, with attribute type = «Primary Service», attribute value = UUID of the Service to search for; the handle values will range from 0x0001 to 0xFFFF. At the end of the test, the IUT will indicate a completion message, e.g., `GATT_Command_Complete`.

  This Test Procedure shall be run four or eight times:

  - with a 16-bit UUID, and with a 128-bit UUID if supported by the IUT.
  - using at least four distinct Sample Databases derived from the Library defined in Section 3.3.

• **Expected Outcome**

  **Pass Verdict**

  The IUT sends `ATT_Find_By_Type_Value_Requests` to the Lower Tester and discovers Services in the Lower Tester. The first command has starting handle = 0x0001, ending handle = 0xFFFF, attribute type = «primary service» and attribute value = UUID of the service to search for. For any subsequent `ATT_Find_By_Type_Value_Requests`, the starting handle shall be set to one greater than the last End Group Handle in the `ATT_Find_By_Type_Value_Response`.

  The IUT sends `ATT_Find_By_Type_Value_Requests` to the Lower Tester until a response with End Group Handle = 0xFFFF is received, a desired service is found, or an `ATT_Error_Response` is received.
• Notes
Prior to each test performance, the IUT should clear the client’s attribute cache. This may be automated by the test equipment and user interaction may not be required.

4.4.4 GATT/SR/GAD/BV-02-C [Discover Primary Services by Service UUID - from server]

• Test Purpose
Verify that a Generic Attribute Profile server can support discovery of all particular Primary Services selected by service UUID, using 16-bit UUIDs, and using 128-bit UUIDs where supported.

• Reference
[1] 4.4.2
[5] 3.4.3.3, 3.4.3.4

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
The Lower Tester sends an ATT_Find_By_Type_Value_Requests (starting handle, ending handle) to the IUT, with type set to «Primary Service» and Value set to a particular UUID, until all Primary Services with a matching service UUID are found, and the IUT returns an ATT_Error_Response with the error code AttributeNotFound or an ATT_Find_By_Type_Value_Response containing an End Group Handle – 0xFFFF.

This Test Procedure shall be run:

  a) once with a 16-bit UUID, and

  b) once with a 128-bit UUID where supported by the IUT.
4.4.5 GATT/CL/GAD/BV-03-C [Find Included Services – by client]

• Test Purpose

Verify that a Generic Attribute Profile client can find include service declarations within a specified service definition on a server.

• Reference

[1] 4.5.1
[5] 3.4.1.1, 3.4.4.1, 3.4.4.2
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

The Upper Tester has access to the test database contained in the Lower Tester, so that the Upper Tester can specify the service handle range.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT may have performed the procedure in GATT/CL/GAD/BV-01-C [Discover All Primary Services - by Client] or GATT/CL/GAD/BV-02-C [Discover Primary Services by Service UUID – by Client] and cached the attribute handles and Service UUID’s of all Primary Services in the Lower Tester database.

• Test Procedure

The Upper Tester issues a command to find included services, specifying the starting and ending handles of the service definition (e.g., GATT_DiscIncludedServices(service handle range)).

The IUT sends ATT_Read_By_Type_Requests (starting handle, ending handle, attribute type) commands to the Lower Tester, with attribute type = <<Include>>. The first request will contain the starting and ending handles of the specified service. For subsequent requests the starting handle is set to one greater than the last attribute handle in the ATT_Read_By_Type_Response.

When an include declaration returned in the ATT_Read_By_Type_Response does not contain the service UUID (ALT 2 in the message sequence chart), indicating the service UUID is a 128-bit UUID, the IUT shall get the service UUID by sending an ATT_Read_Request with the handle set to the attribute handle of the included service, if the IUT supports 128-bit UUIDs, and the IUT has not previously obtained the service UUID via Primary Service Discovery.

The test will continue until all Included Services for the specified service are found, and either a) the Lower Tester returns an ATT_Error_Response with the error code AttributeNotFound or b) the Lower Tester returns a handle equal to the ending handle for the service. At the end of the test, the IUT will indicate a completion message, e.g., GATT_Command_Complete.

This Test Procedure shall be run at least four times using four or more distinct Sample Databases derived from the Library defined Section 3.3, where at least one of the inquired service handle ranges contains a mix of Included Services with 16-bit and 128-bit service UUIDs.
Expected Outcome

Pass Verdict

The IUT sends ATT_Read_By_Type_Requests to the Lower Tester until all include service declarations in the specified handle range are returned.

If the IUT supports 128-bit service UUIDs, when an include declaration returned in the ATT_Read_By_Type_Response does not contain the service UUID, and the IUT has not previously discovered the service UUID via Primary Service Discovery, the IUT gets the service UUID by sending an ATT_Read_Request with the handle set to the attribute handle of the included service. If the IUT has previously discovered the service UUID via Primary Service Discovery and/or Relationship Discovery, the IUT may optionally send ATT_Read_Requests to get all included service UUIDs.

The IUT does not report any included services that should not be part of the primary service.
4.4.6 GATT/SR/GAD/BV-03-C [Find Included Services – from Server]

- **Test Purpose**
  
  Verify that a Generic Attribute Profile server can support a search for all Included Services in a specified handle range.

- **Reference**
  
  [1] 4.5.1
  
  [5] 3.4.1.1, 3.4.4.1, 3.4.4.2

- **Initial Condition**
  
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  
  The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

- **Test Procedure**
  
  The Lower Tester sends ATT_Read_By_Type_Requests (starting handle, ending handle, attribute type) to the IUT, with attribute type set to <<Include>>, until all include declarations are found, and either a) the IUT returns an ATT_Error_Response with the error code AttributeNotFound or b) the IUT returns an attribute handle equal to the ending handle.

  The Lower Tester checks if any of the returned include declarations refer to the service being searched (i.e., a circular Reference).
• Expected Outcome

Pass Verdict

The IUT replies with an `ATT_Read_By_Type_Response` to the Lower Tester for each received `ATT_Read_By_Type_Request`, if the IUT has include service declarations within the handle range of the request.

The `ATT_Read_By_Type_Responses` have complete handle-value pairs and the handle-value pairs are returned sequentially based on the attribute handle.

The IUT reports all include service declarations known to be contained in its database within the specified handle range.

The response size does not exceed any negotiated ATT_MTU.

The IUT sends all `ATT_Read_By_Type_Responses` to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).

The IUT sends an `ATT_Error_Response` with error code AttributeNotFound if there are no include service declarations within the handle range of the request.

The IUT does not report any included services that should not be part of the primary service.

4.4.7 GATT/CL/GAD/BV-04-C [Discover All Characteristics of a Service – by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can discover characteristic declarations within a specified service definition.

• Reference

[1] 4.6.1

[5] 3.4.1.1, 3.4.4.1, 3.4.4.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Upper Tester has access to the test database contained in the Lower Tester, so that the Upper Tester can specify the service handle range.

The desired characteristic declaration, if used by the IUT, is in the IXIT [10].

• Test Procedure

The Upper Tester issues a command to discover service characteristics, specifying the starting and ending handles of the service definition (e.g., GATT_DiscServiceChar (service handle range).
The IUT sends ATT_Read_By_Type_Requests (starting handle, ending handle, attribute type) to the Lower Tester, with attribute type = <Characteristic>. The first request will contain the starting and ending handles of the specified service. For subsequent requests, the starting handle is set to one greater than the last attribute handle in the ATT_Read_By_Type_Response. At the end of the test, the IUT will indicate a completion message, e.g., GATT_Command_Complete.

This Test Procedure shall be run four times using distinct Sample Databases derived from the Library defined in Section 3.3.

**Expected Outcome**

**Pass Verdict**

The IUT sends ATT_Read_By_Type_Requests to the Lower Tester and discovers Characteristics in the Lower Tester. The first command has starting handle set to the starting handle of the specified service, ending handle ending handle set to the ending handle of the specified service, and attribute type = <Characteristic>. For any subsequent ATT_Read_By_Type_Requests, the starting handle shall be set to one greater than the last attribute handle in the ATT_Read_By_Type_Response.
The IUT sends \texttt{ATT\_Read\_By\_Type\_Requests} to the Lower Tester until a response with attribute handle = the ending handle is received, a desired characteristic declaration is found, or an \texttt{ATT\_Error\_Response} is received.

4.4.8 GATT/SR/GAD/BV-04-C [Discover All Characteristics of a Service – from Server]

- **Test Purpose**
  
  Verify that a Generic Attribute Profile server can support a search for all characteristics of a specified Service, and report all of those characteristics.

- **Reference**

  [1] 4.6.1

  [5] 3.4.1.1, 3.4.4.1, 3.4.4.2

- **Initial Condition**

  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

- **Test Procedure**

  The Lower Tester sends \texttt{ATT\_Read\_By\_Type\_Requests} (starting handle, ending handle, attribute type) to the IUT, with attribute type set to \texttt{<Characteristic>} and the initial handle range set to that for the specified service definition, until all characteristics in that handle range are found, and either a) the IUT returns an \texttt{ATT\_Error\_Response} with the error code AttributeNotFound or b) the IUT returns an attribute handle equal to the ending handle.
• Expected Outcome

**Pass Verdict**

The IUT sends one `ATT_Read_By_Type_Response` to the Lower Tester for each received `ATT_Read_By_Type_Request`, if it has characteristic declarations within the handle range.

The IUT reports all characteristics of the specified service.

The `ATT_Read_By_Type_Responses` have complete handle-value pairs and the handle-value pairs are returned sequentially based on the attribute handle.

The response size does not exceed any negotiated `ATT_MTU`.

The IUT sends all `ATT_Read_By_Type_Responses` to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).

The IUT sends an `ATT_Error_Response` if there are no further characteristic declarations within the handle range of the request.

### 4.4.9 GATT/CL/GAD/BV-05-C [Discover Characteristics by UUID – by Client]

• **Test Purpose**

Verify that a Generic Attribute Profile client can discover characteristics of a specified service, using 16-bit and 128-bit characteristic UUIDs.

• **Reference**

[1] 4.6.2

[5] 3.4.1.1, 3.4.4.1, 3.4.4.2
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

The Upper Tester has access to the test database contained in the Lower Tester, so that the Upper Tester can specify the service handle range and characteristic UUID.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure

The Upper Tester issues a command to discover service characteristics, specifying the starting and ending handles of the service definition and the characteristic UUID (e.g., GATT_DiscServiceCharUUID (service handle range, UUID).

The IUT sends ATT_Read_By_Type_Requests (starting handle, ending handle, attribute type) commands to the Lower Tester, with attribute type = <Characteristic>. The first request will contain the starting and ending handles of the specified service. For subsequent requests, the starting handle is set to one greater than the last attribute handle in the ATT_Read_By_Type_Response. The test will continue until all characteristic declarations for that Service are found, and either a) the IUT returns an ATT_Error_Response with the error code AttributeNotFound or b) the IUT returns a handle value equal to the last specified handle. At the end of the test, the IUT will indicate a completion message, e.g., GATT_Command_Complete.

Each ATT_Read_By_Type_Response returns a list of attribute handle and attribute value pairs. Each attribute value is a characteristic declaration containing the characteristic UUID. The IUT checks the UUID in each attribute value for a match to the characteristic UUID being discovered.

This Test Procedure shall be run eight times:

- With a 16-bit UUID and with a 128-bit UUID
- Using at least four distinct Sample Databases derived from the Library defined Section 3.3.
**4.4.10 GATT/SR/GAD/BV-05-C [Discover Characteristics by UUID – from server]**

- **Test Purpose**
  Verify that a Generic Attribute Profile server can discover service Characteristics of a specified UUID, using 16-bit UUoIDs, and using 128-bit UUoIDs.

- **Reference**
  [1] 4.6.2
  [5] 3.4.1.1, 3.4.4.1, 3.4.4.2

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

**Generic Attribute Profile (GATT) / Test Suite**
• Test Procedure

The Lower Tester sends ATT_Read_By_Type_Requests (starting handle, ending handle, attribute type) to the IUT, with attribute type set to <Characteristic> and the initial handle range set to that for the specified Service definition, until all Characteristics with matching UUID in that handle range are found, and either a) the IUT returns an ATT_Error_Response with the error code AttributeNotFound or b) the IUT returns an attribute handle equal to the ending handle.

This Test Procedure shall be run:

a) Once with a 16-bit UUID, and
b) Once with a 128-bit UUID where supported.

• Expected Outcome

Pass Verdict

The IUT sends one ATT_Read_By_Type_Response to the Lower Tester for each received ATT_Read_By_Type_Request, if it has characteristic declarations within the handle range.

The IUT reports all Characteristics of the specified Service.

The ATT_Read_By_Type_Responses have complete handle-value pairs and the handle-value pairs are returned sequentially based on the attribute handle.

The response size does not exceed any negotiated ATT_MTU.

The IUT sends all ATT_Read_By_Type_Responses to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).

The IUT sends an ATT_Error_Response if there are no characteristic declarations within the handle range of the request.
4.4.11 GATT/CL/GAD/BV-06-C [Discover All Characteristic Descriptors – by client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can find all Descriptors of a specified Characteristic.

- **Reference**
  [1] 4.7.1
  [5] 3.4.1.1, 3.4.3.1, 3.4.3.2

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  The desired characteristic descriptor, if used by the IUT, is in the IXIT [10].

- **Test Procedure**
  The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_DiscDescriptors.
  The IUT in turn sends ATT_Find Information_Requests (starting handle, ending handle) to the Lower Tester. The first request will set the starting handle to the handle of the specified Characteristic Value + 1 and the ending handle to the ending handle of the specified characteristic. For subsequent requests, the starting handle is set to one greater than the last attribute handle in the ATT_Find Information_Response. The test will continue until all Characteristic Descriptors for that Characteristic are found, and either a) the IUT returns an ATT_Error_Response with the error code AttributeNotFound or the IUT returns a handle equal to the ending handle. At the end of the test, the IUT will indicate a completion message, e.g., GATT_Command_Complete.

  This Test Procedure shall be run four times using all of the distinct Sample Databases derived from the Library defined in [7] and in Section 3.3.
• Expected Outcome
  Pass Verdict
  The IUT sends `ATT.Find.Information.Requests` to the Lower Tester until all a desired characteristic descriptor is found, or all descriptors in the specified handle range are found.

• Notes
  Prior to each test performance, the IUT should clear the client’s attribute cache. This may be automated by the test equipment and user interaction may not be required.

4.4.12 GATT/CL/GAD/BV-07-C [Discover Primary Services using SDP - by client]

• Test Purpose
  Verify that a Service Discovery Protocol client IUT discovers Primary GATT Services in a GATT server using SDP over BR/EDR.

• Reference
  [1] 4.4.1, 9
  [8] 2.6, 4.5

• Initial Condition
  A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over the BR/EDR transport.

  The Lower Tester instantiates a Service Database including GATT Services. The Upper Tester knows the handles and UUIDs of those GATT Services. All SDP records for GATT Services conform to the requirements in Table 9.1 in [1].
Test Procedure

The Upper Tester sends a request to the IUT to initiate Primary GATT Service Discovery over SDP, e.g., ‘SDP_DiscoverGATTServices’.

This Test Procedure shall be run four times, using distinct Sample Databases derived from the Library defined in Section 3.3.

Both MSCs below are example procedures and may be used as references:

ATT Bearer established over BR/EDR transport (4.2).

First request
- E.g. SDP_ServiceSearchRequest → SDP_DiscoverGATTServices

If Service found
- E.g. SDP_ServiceSearchResponse
- ALT 1: continuation !=0x00
- E.g. SDP_ServiceSearchRequest → SDP_DiscoverGATTServices

Last service found
- E.g. SDP_ServiceSearchResponse
- ALT 2: continuation = 0x00
- SDP_DiscoverGATTServices (handle list)
  - SDP_GATTDiscComplete

If no remaining Services
- E.g SDP_ServiceSearchResponse
- ALT 3: no more services
  - SDP_GATTDiscComplete

E.g. SDP_ServiceSearchResponse.
Upper Tester
IUT
Lower Tester

ATT Bearer established over BR/EDR transport (4.2).
• Expected Outcome

**Pass Verdict**

The IUT executes an SDP search for Primary GATT based services, which use the ATT protocol. The SDP commands sent by the IUT are correctly formatted. The IUT receives the response from the Lower Tester, and responds to the Upper Tester with the list of GATT services. The Upper Tester verifies that the list is as expected.

The IUT should continue the search until the Lower Tester indicates that the search is complete, and then it shall report that the search is complete to the Upper Tester, e.g., SDP_GATTDiscComplete.

**4.4.13 GATT/CL/GAD/BV-08-C [Discover Services by UUID using SDP - by Client]**

• **Test Purpose**

Verify that a Service Discovery Protocol client IUT discovers GATT Services contained in a GATT server IUT selected by Service UUID.

• **Reference**

[1] 4.4.1, 9

[9] 2.6, 4.5
• Initial Condition

A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over the BR/EDR transport.

The Lower Tester instantiates a Service Database, including GATT Services. The Upper Tester knows the handles and UUIDs of those GATT Services.

• Test Procedure

The Upper Tester sends a request to the IUT to initiate Primary GATT Service Discovery by UUID over SDP, e.g., ‘SDP_DiscoverGATTServicesByUUID’; that request will specify the Service UUID of one of the GATT Services contained in the Lower Tester’s database.

This Test Procedure shall be run at least four times, using distinct Sample Databases derived from the Library defined in Section 3.3.

Both MSCs below are example procedures and may be used as references:
generic attribute profile (gatt) / test suite

ATT bearer established over BR/EDR transport (4.2).

- **Expected Outcome**

  **Pass Verdict**

  The IUT executes an SDP search for Primary GATT services specified by the Service UUID, which use the ATT Protocol. The SDP commands sent by the IUT are correctly formatted.

  The IUT receives the response from the Lower Tester, and responds to the Upper Tester with the list of GATT services. The Upper Tester shall verify that the list contains only the GATT services with the requested service UUID.

  The IUT should continue the search until the Lower Tester indicates that the search is complete, then, it shall report that the search is complete to the Upper Tester, e.g., SDP_GATTDiscComplete.

4.4.14 GATT/SR/GAD/BV-06-C [Discover All Characteristic Descriptors – from Server]

- **Test Purpose**

  Verify that a Generic Attribute Profile server can support a search for all descriptors of a specified characteristic.

- **Reference**

  [1] 4.7.1
• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7].

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
The Lower Tester sends ATT_Read_Find_Information_Requests (starting handle, ending handle) to the IUT and the initial handle range set to that for the specified Characteristic definition, until all Characteristic Descriptors with matching UUID in that handle range are found, and either a) the IUT returns an ATT_Error_Response with the error code AttributeNotFound or b) the IUT returns a handle equal to the end handle specified.

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Find_Information_Response to the Lower Tester for each received ATT_Find_Information_Request if it has characteristic descriptors within the handle range.

The IUT reports all descriptors defined for the specified characteristic.

The ATT_Find_Information_Responses have complete handle-UUID pairs. A handle-UUID pair fits into one response packet. The handle-UUID pairs are returned in ascending order of attribute handles.

The response size does not exceed any negotiated ATT_MTU.
The IUT sends all `ATT_Find_Information_Responses` to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).

The IUT sends an `ATT_Error_Response` when there are no characteristic descriptors within the handle range.

### 4.4.15 GATT/SR/GAD/BV-07-C [Discover Primary Services using SDP - from Server]

- **Test Purpose**
  Verify that a Generic Attribute Profile server IUT can support a search for all GATT Services it contains.

- **Reference**
  [1] 4.4.1, 9
  [9] 2.6, 4.5

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over the BR/EDR transport.

  The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7]. That server database includes at least one GATT Service.

- **Test Procedure**
  The Lower Tester sends an initial SDP_ServiceSearchAttributeRequest (code=0x06, a service search parameter UUID of `<ATT>`), a non-zero value for maximum attribute byte count parameter, an AttributeIDList parameter of AID 0x0001 and AID 0x0004, and a continuation state parameter of 0x00.

  If the IUT returns an SDP_ServiceSearchAttributeResponse, including a non-zero continuation state parameter, continue the test by sending another SDP_ServiceSearchAttributeRequest, including the returned continuation state parameter, from the Lower Tester to the IUT.

  If the IUT returns an SDP_ServiceSearchAttributeResponse, with a zero continuation state parameter, terminate the test.
Both MSCs below are example procedures and may be used as references:

**ATT Bearer established over BR/EDR transport (4.2).**

**First request**  
If Service found:
- **SDP ServiceSearchRequest**  
  (Code=0x02, ServiceSearchPattern = <ATT>, max count, continuation=0)
- **SDP ServiceSearchResponse.**  
  (Code = 0x03, total record count, Handle list, continuation)

**Last service found**  
ALT 1: continuation !=0x00

If no remaining Services:
- **SDP ServiceSearchResponse.**  
  (Code = 0x03, total record count, Handle list, continuation=0x00)

ALT 2: continuation = 0x00

ALT 3: no more services

ALT 1: continuation !=0x00

ALT 2: continuation = 0x00

ALT 3: no more services
ATT Bearer established over BR/EDR transport (4.2).

**Expected Outcome**

**Pass Verdict**

For each `SDP_ServiceSearchAttributeRequest`, the IUT sends a correctly formatted `SDP_ServiceSearchAttributeResponse` to the Lower Tester.

The IUT reports service UUID (either UUID16 or UUID128), GATT Start handle (Uint16) and GATT End handle (Uint16) for each GATT Services known by the Lower Tester to be contained in its database (via IXIT [10]).

### 4.4.16 GATT/SR/GAD/BV-08-C [Discover Services by UUID using SDP - from Server]

**Test Purpose**

Verify that a Service Discovery Protocol server IUT can support a search for all GATT Services it contains.

**Reference**

[1] 4.4.1, 9

[9] 2.6, 4.5
• Initial Condition

A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over the BR/EDR transport.

The Upper Tester has access to the IUT server database structure, either from IXIT [10] or from use of a predefined database [7]. That server database includes at least one GATT Service; the <Service UUID> of that service is known by the Lower Tester.

• Test Procedure

The Lower Tester sends an initial SDP_ServiceSearchAttributeRequest (code=0x06, a service search parameter UUID of <ATT> and the <Service UUID>, a non-zero value for maximum attribute byte count parameter, an AttributeIDList parameter of AID 0x0001 and AID 0x0004, and a continuation state parameter of 0x00.

If the IUT returns an SDP_ServiceSearchAttributeResponse, including a non-zero continuation state parameter, continue the test by sending another SDP_ServiceSearchAttributeRequest, including the returned continuation state parameter, from the Lower Tester to the IUT.

If the IUT returns an SDP_ServiceSearchAttributeResponse, with a zero continuation state parameter, terminate the test.
Both MSCs below are example procedures and may be used as references:

**Upper Tester**

**IUT**

**Lower Tester**

**ATT Bearer established over BR/EDR transport (4.2).**

**First request**

- **SDP_ServiceSearchRequest**
  (Code=0x02, ServiceSearchPattern =,<ATT> <Service UUID>, max count, continuation=0)

**If Service found**

- **SDP_ServiceSearchResponse**
  (Code = 0x03, total record count, Handle list, continuation)

- **SDP_ServiceSearchRequest**
  (Code=0x02, ServiceSearchPattern =,<ATT> <Service UUID>, max count, continuation)

**Last service found**

- **SDP_ServiceSearchResponse**
  (Code = 0x03, total record count, Handle list, continuation=0x00)

**If no remaining Services**

- **SDP_ServiceSearchResponse**
  (Code = 0x03, total record count = 0, Empty handle list, continuation=0x00)

**ALT 1: continuation !=0x00**

**ALT 2: continuation = 0x00**

**ALT 3: no more services**
• Expected Outcome

**Pass Verdict**

For each `SDP_ServiceSearchAttributeRequest`, the IUT sends a correctly formatted `SDP_ServiceSearchAttributeResponse` to the Lower Tester.

The IUT reports Service UUID (either UUID16 or UUID128), GATT Start handle (Uint16) and GATT End handle (Uint16) for each GATT Services known by the Lower Tester to be contained in its database (via IXIT [10]).

### 4.5 Read

The test group objective is to verify Generic Attribute Profile reading of Characteristic Values and Characteristic Descriptors.

#### 4.5.1 GATT/CL/GAR/BV-01-C [Read Characteristic Value - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can read a Characteristic Value selected by handle.
• Reference
[1] 4.8.1

[5] 3.4.4.3, 3.4.4.4

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure
The Upper Tester will specify the handle of a Characteristic Value contained in the Lower Tester.

Send a request from the Upper Tester to the IUT to read a Characteristic Value from the Lower Tester by specifying the characteristic handle e.g., GATT_ReadReq.

- ATT_Read_Request (Code = 0x0A, handle)
- ATT_Read_Response (Code = 0x0B, value)

• Expected Outcome
Pass Verdict

The IUT sends a correctly formatted ATT_Read_Request command (0x0A) to the Lower Tester.

The Characteristic handle parameter is set to the handle specified by the Upper Tester.

When the ATT_Read_Response is received, the IUT sends the received response with the correct value to the Upper Tester, e.g., GATT_ReadRes.

4.5.2 GATT/CL/GAR/BI-01-C [Read Characteristic Value – Invalid Handle]

• Test Purpose
Verify Generic Attribute Profile client behavior when the Read Characteristic Value procedure fails due to invalid handle.
• Reference
[1] 4.8.1
[5] 3.4.1.1, 3.4.4.3

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq (handle). The IUT sends an ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadRes.

• Expected Outcome
Pass Verdict

The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.3 GATT/CL/GAR/BI-02-C [Read Characteristic Value – Read Not Permitted]
• Test Purpose
Verify Generic Attribute Profile client behavior when the Read Characteristic Value procedure fails due to read not permitted.
• Reference
  [1] 4.8.1
  [5] 3.4.1.1, 3.4.4.3

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester is selected.

• Test Procedure
The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle). The IUT sends an ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadRes.

• Expected Outcome
  Pass Verdict

  The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester.

  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.
4.5.4 GATT/CL/GAR/BI-03-C [Read Characteristic Value – Insufficient Authorization]

- **Test Purpose**
  Verify Generic Attribute Profile client behavior when the Read Characteristic Value procedure fails due to insufficient authorization.

- **Reference**
  [1] 4.8.1  
  [5] 3.4.1.1, 3.4.4.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that requires read authorization is selected.

  No authorization procedure has been performed between the IUT and the Lower Tester.

- **Test Procedure**
  The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle). The IUT sends an ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadRes.
• Expected Outcome
Pass Verdict

The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.5 GATT/CL/GAR/BI-04-C [Read Characteristic Value – Insufficient Authentication]

• Test Purpose
Verify Generic Attribute Profile client behavior when the Read Characteristic Value procedure fails due to insufficient authentication.

• Reference
[1] 4.8.1
[5] 3.4.1.1, 3.4.4.3

• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester that requires read authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure
The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle). The IUT sends an ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadRes.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted `ATT_Read_Request` to the Lower Tester.

Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadRes`.

### 4.5.6 GATT/CL/GAR/BI-05-C [Read Characteristic Value – Insufficient Encryption Key Size]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Read Characteristic Value procedure fails due to insufficient encryption key size.

• Reference

[1] 4.8.1

[5] 3.4.1.1, 3.4.4.3

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester that requires encryption with a key longer than the key used to establish the encrypted link is selected.
• Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle). The IUT sends an ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadRes.

**Expected Outcome**

**Pass Verdict**

The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.7 GATT/SR/GAR/BV-01-C [Read Characteristic Value - from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support reading a Characteristic Value selected by handle.

• Reference

[1] 4.8.1

[5] 3.4.4.3, 3.4.4.4

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester has the necessary security permissions from the IUT to read a characteristic.
• Test Procedure

Send an ATT_Read_Request from the Lower Tester to the IUT to read a Characteristic Value by specifying the Characteristic Value Handle.

- ATT_Read_Request (Code = 0x0A, handle)

- ATT_Read_Response (Code = 0x0B, value)

• Expected Outcome

Pass Verdict

The IUT sends ATT_Read_Response (code 0x0B) to the Lower Tester.

The Characteristic Value is set to the value of the characteristic identified by the Characteristic Value Handle in the ATT_Read_Request.

The response size does not exceed any negotiated ATT_MTU. If the Characteristic Value is longer than (ATT_MTU – 1) then the first (ATT_MTU – 1) octets are included in this response.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.8 GATT/SR/GAR/BI-01-C [Read Characteristic Value- Read Not Permitted Response]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request to a non-readable Characteristic Value and issue a Read Not Permitted Response.

• Reference

[1] 4.8.1

[5] 3.4.1.1, 3.4.4.3

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that does not permit reading is selected.

- **Test Procedure**
  Send an ATT_read_Request from the Lower Tester to the IUT using the selected handle.

  ![Diagram](image)

  - **ATT_read_Request** (Code = 0x0A, handle)
  - **ATT_error_Response** (Code = 0x01, 0x0A, handle, Error Code = 0x02)

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an **ATT_error_Response** (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x02, Read Not Permitted.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.5.9 GATT/SR/GAR/BI-02-C [Read Characteristic Value - Invalid Handle Response]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request with an unsupported Characteristic Value handle and issue an Invalid Handle Response.

- **Reference**
  [1] 4.8.1
  [5] 3.4.1.1, 3.4.4.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester selects a handle which is known to be invalid.

- **Test Procedure**

The Lower Tester sends an ATT_Read_Request to the IUT to using the selected handle.

![Diagram](Diagram)

- **Expected Outcome**

**Pass Verdict**

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x01, Invalid Handle.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.5.10 GATT/SR/GAR/Bi-03-C [Read Characteristic Value – Insufficient Authorization]

- **Test Purpose**

Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request and issue an Insufficient Authorization Response.

- **Reference**

[1] 4.8.1

[5] 3.4.1.1, 3.4.4.3

- **Initial Condition**

A preamble procedure defined in Section 4.2.4.2 is used to set up the transport and L2CAP channel over LE.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that requires read authorization is selected.

No authorization procedure has been performed between the IUT and the Lower Tester.

- Test Procedure
  The Lower Tester sends an ATT_Read_Request to the IUT to using the selected handle.

  • Expected Outcome
  Pass Verdict
  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x08, Insufficient Authorization.
  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.11 GATT/SR/GAR/BI-04-C [Read Characteristic Value – Insufficient Authentication]

- Test Purpose
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request and issue an Insufficient Authentication Response.

- Reference
  [1] 4.8.1
  [5] 3.4.1.1, 3.4.4.3
• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that requires read authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure
The Lower Tester sends an ATT_Read_Request to the IUT to using the selected handle.

ATT_Read_Request (Code = 0x0A, handle)
ATT_Error_Response (Code = 0x01, 0x0A, handle, Error Code = 0x05)

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.12 GATT/SR/GAR/BI-05-C [Read Characteristic Value – Insufficient Encryption Key Size]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request and issue an Insufficient Encryption Key Size Response.

• Reference
[1] 4.8.1
3.4.1.1, 3.4.4.3

- Initial Condition
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.
  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the IUT that requires encryption with a key longer than the key used to establish the encrypted link is selected.

- Test Procedure
  The Lower Tester sends an ATT_Read_Request to the IUT to using the selected handle.

  **Expected Outcome**
  
  **Pass Verdict**

  The IUT sends an **ATT_Error_Response** (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x0C, Insufficient Encryption Key Size.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.5.13 GATT/CL/GAR/BV-03-C [Read Using Characteristic UUID - by Client]

- Test Purpose
  Verify that a Generic Attribute Profile client can read a Characteristic Value selected by UUID, using a 16-bit UUID and using a 128-bit UUID.
• Reference

[1] 4.8.2
[5] 3.4.4.1, 3.4.4.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A UUID for a characteristic in the Lower Tester that permits reading is selected.

If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure

Send a request from the Upper Tester to the IUT to read a Characteristic from the Lower Tester by specifying the Characteristic UUID e.g., GATT_ReadByTypeReq (starting handle, ending handle, characteristic UUID). The starting and ending handles are set to the range for the service to which the characteristic belongs. The Lower Tester will support at least one characteristic of the type specified by the UUID.

This Test Procedure shall be run once or twice, with a 16-bit UUID, and with a 128-bit UUID if supported by the IUT.

• Expected Outcome

Pass verdict

The IUT sends a correctly formatted ATT_Read_By_Type_Request to the Lower Tester.

The IUT receives the ATT_Read_By_Type_Response sent by the Lower Tester.
The values reported to the Upper Tester match the values delivered by the Lower Tester.

4.5.14 GATT/CL/GAR/BI-06-C [Read Using Characteristic UUID – Read Not Permitted]

• Test Purpose
  Verify Generic Attribute Profile client behavior when the Read Using Characteristic UUID procedure fails due to read not permitted.

• Reference
  [1] 4.8.2
  [5] 3.4.4.1, 3.4.4.2

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  A UUID for a characteristic in the Lower Tester is selected.

• Test Procedure
  Send a request from the Upper Tester to the IUT to read a characteristic from the Lower Tester by specifying the characteristic UUID e.g., GATT_ReadByTypeReq (starting handle, ending handle, characteristic UUID).

• Expected Outcome
  Pass Verdict
  The IUT sends a correctly formatted ATT_Read_By_Type_Request to the Lower Tester.
Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadRes`.

### 4.5.15 GATT/CL/GAR/BI-07-C [Read Using Characteristic UUID – Attribute Not Found]

**Test Purpose**

Verify Generic Attribute Profile client behavior when the Read Using Characteristic UUID procedure fails due to attribute not found.

**Reference**

[1] 4.8.2  
[5] 3.4.4.1, 3.4.4.2

**Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A UUID that is not used in the Lower Tester is selected.

The starting handle is set to 0x0001 and the ending handle is set to 0xFFFF.

**Test Procedure**

Send a request from the Upper Tester to the IUT to read a characteristic from the Lower Tester by specifying the characteristic UUID e.g., `GATT_ReadByTypeReq` (starting handle, ending handle, characteristic UUID).
• Expected Outcome
  
  Pass Verdict
  
  The IUT sends a correctly formatted ATT_Read_By_Type_Request to the Lower Tester.
  
  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.16 GATT/CL/GAR/BI-09-C [Read Using Characteristic UUID – Insufficient Authorization]

• Test Purpose
  
  Verify Generic Attribute Profile client behavior when the Read Using Characteristic UUID procedure fails due to insufficient authorization.

• Reference
  
  [1] 4.8.2
  
  [5] 3.4.4.1, 3.4.4.2

• Initial Condition
  
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  
  A UUID for a characteristic in the Lower Tester that requires read authorization is selected.
  
  A handle range is selected in which the selected characteristic UUID occurs only once.
  
  No authorization procedure has been performed between the IUT and the Lower Tester.

• Test Procedure
  
  Send a request from the Upper Tester to the IUT to read a characteristic from the Lower Tester by specifying the characteristic UUID e.g., GATT_ReadByTypeReq (starting handle, ending handle, characteristic UUID).
• Expected Outcome
  
  **Pass Verdict**

  The IUT sends a correctly formatted `ATT_Read_By_Type_Request` to the Lower Tester.

  Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadRes`.

4.5.17 GATT/CL/GAR/BI-10-C [Read Using Characteristic UUID – Insufficient Authentication]

• Test Purpose

  Verify Generic Attribute Profile client behavior when the Read Using Characteristic UUID procedure fails due to insufficient authentication.

• Reference

  [1] 4.8.2

  [5] 3.4.4.1, 3.4.4.2

• Initial Condition

  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A UUID for a characteristic in the Lower Tester that requires read authentication is selected.

  A handle range is selected in which the selected characteristic UUID occurs only once.

  No authentication procedure has been performed between the IUT and the Lower Tester.
• **Test Procedure**

Send a request from the Upper Tester to the IUT to read a characteristic from the Lower Tester by specifying the characteristic UUID e.g., GATT_ReadByTypeReq (starting handle, ending handle, characteristic UUID).

![Diagram](chart.png)

- **Expected Outcome**
  - **Pass Verdict**

  The IUT sends a correctly formatted `ATT_Read_By_Type_Request` to the Lower Tester.

  Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.18 GATT/CL/GAR/BI-11-C [Read Using Characteristic UUID – Insufficient Encryption Key Size]

• **Test Purpose**

  Verify Generic Attribute Profile client behavior when the Read Using Characteristic UUID procedure fails due to insufficient encryption key size.

• **Reference**

  [1] 4.8.2

  [5] 3.4.4.1, 3.4.4.2

• **Initial Condition**

  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A UUID for a characteristic in the Lower Tester that requires encryption with a key longer than the key used to establish the encrypted link is selected.

A handle range is selected in which the selected characteristic UUID occurs only once.

• Test Procedure

Send a request from the Upper Tester to the IUT to read a characteristic from the Lower Tester by specifying the characteristic UUID e.g., GATT_ReadByTypeReq (starting handle, ending handle, characteristic UUID).

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_By_Type_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester, the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.19 GATT/SR/GAR/BV-03-C [Read using Characteristic UUID - from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support the Read using Characteristic UUID procedure, using a 16-bit UUID and using a 128-bit UUID if supported.

• Reference

[1] 4.8.2

[5] 3.4.4.1, 3.4.4.2
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester has the necessary security permissions from the IUT to read a characteristic.

• Test Procedure

Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT with a specified UUID and handle range.

This Test Procedure shall be run:

a) With a 128-bit UUID type value specified by the manufacturer in the IXIT [10], if supported by the IUT, and

b) With a 16-bit UUID value required to be supported by a GATT server specified by the manufacturer in the IXIT [10].

• Expected Outcome

Pass Verdict

For each pass:

- The IUT sends a correctly formatted ATT_Read_By_Type_Response (result code 0x09) command to the Lower Tester, containing a list of Attribute Handle and Attribute Value pairs corresponding to the characteristics contained in the handle range provided.

- If the Attribute Value is longer than (ATT_MTU – 4) or 253, whichever is smaller, then the first (ATT_MTU – 4) or 253 octets are included in this response. The response does not exceed the ATT_MTU.

- The IUT sends each ATT_Read_By_Type_Response to the Lower Tester within the applicable timeout (3.3.3/ATT [5]).
4.5.20 GATT/SR/GAR/BI-06-C [Read Characteristic by UUID - Read Not Permitted Response]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request to a non-readable Characteristic Value and issue a Read Not Permitted Response.

• Reference
[1] 4.8.2
[5] 3.4.1.1, 3.4.4.1

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A UUID for a characteristic in the IUT that does not permit reading is selected.

A handle range is selected in which the selected characteristic UUID occurs only once.

• Test Procedure
Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.

• Expected Outcome
Pass Verdict
The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the handle of the selected attribute. The Error code is set to 0x02, Read Not Permitted.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.21 GATT/SR/GAR/BI-07-C [Read Characteristic by UUID - Attribute Not Found Response]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request to an unsupported characteristic and issue an Attribute Not Found Response.

• Reference
[1] 4.8.2
[5] 3.4.1.1, 3.4.4.1

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A UUID that is not used in the Lower Tester is selected.

The starting handle is set to 0x0001 and the ending handle is set to 0xFFFF.

• Test Procedure
Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.

ATT_Read_By_Type_Request
(Code = 0x08, starting handle, ending handle, characteristic UUID)

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the starting handle. The Error code is set to 0x0A, Attribute Not Found.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.22 GATT/SR/GAR/BI-08-C [Read Characteristic by UUID - Invalid Handle Response]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request with an invalid handle range and issue an Invalid Handle Response.

- **Reference**
  [1] 4.8.2
  [5] 3.4.1.1, 3.4.4.1

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  The UUID is set to the UUID for Primary Service.

  The starting handle is set to 0x0002 and the ending handle is set to 0x0001 such that the ending handle is less than the starting handle.

- **Test Procedure**
  Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the starting handle. The Error code is set to 0x01, Invalid Handle.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.23 GATT/SR/GAR/BI-09-C [Read Using Characteristic UUID – Insufficient Authorization]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request and issue an Insufficient Authorization Response.

- **Reference**
  [1] 4.8.2
  [5] 3.4.1.1, 3.4.4.1

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A UUID for a characteristic in the IUT that requires read authorization is selected.

  A handle range is selected in which the selected characteristic UUID occurs only once.

  No authorization procedure has been performed between the IUT and the Lower Tester.

- **Test Procedure**
  Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.

  ![Diagram](image)

  ATT_Read_By_Type_Request
  (Code = 0x08, starting handle, ending handle, characteristic UUID)

  ATT_Error_Response
  (Code = 0x01, 0x08, handle, Error Code=0x08)

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the handle of the selected attribute. The Error code is set to 0x08, Insufficient Authorization.
The IUT responds within the applicable timeout (3.3.3/ATT [5]).

**4.5.24 GATT/SR/GAR/BI-10-C [Read Using Characteristic UUID – Insufficient Authentication]**

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request and issue an Insufficient Authentication Response.

- **Reference**
  [1] 4.8.2
  [5] 3.4.1.1, 3.4.4.1

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A UUID for a characteristic in the IUT that requires read authentication is selected.

  A handle range is selected in which the selected characteristic UUID occurs only once.

  No authentication procedure has been performed between the IUT and the Lower Tester.

- **Test Procedure**
  Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.

```
ATT_Read_By_Type_Request
(Code = 0x08, starting handle, ending handle, characteristic UUID)
```

Lower Tester IUT Upper Tester

L2CAP Connection Established over LE transport (4.2.1.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

```
ATT_Error_Response
(Code = 0x01, 0x08, handle, Error Code=0x05)
```

ATT_Read_By_Type_Request
(Code = 0x08, starting handle, ending handle, characteristic UUID)

ATT_Error_Response
(Code = 0x01, 0x08, handle, Error Code=0x05)
• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the handle of the selected attribute. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.25 GATT/SR/GAR/BI-11-C [Read Using Characteristic UUID – Insufficient Encryption Key Size]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic by UUID Request and issue an Insufficient Encryption Key Size Response.

• Reference

[1] 4.8.2

[5] 3.4.1.1, 3.4.4.1

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A UUID for a characteristic in the IUT that requires encryption with a key longer than the key used to establish the encrypted link is selected.

A handle range is selected in which the selected characteristic UUID occurs only once.

• Test Procedure

Send an ATT_Read_By_Type_Request from the Lower Tester to the IUT using the selected UUID and handle range.
• Expected Outcome

**Pass Verdict**

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x08. The Attribute Handle in Error parameter is set to the handle of the selected attribute. The Error code is set to 0x0C, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.5.26 GATT/CL/GAR/BV-04-C [Read Long Characteristic Value - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can read a Characteristic Value by selected handle. The Characteristic Value length is unknown to the client and might be long.

• Reference

[1] 4.8.3

[5] 3.4.4.5, 3.4.4.6

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure

Send a command from the Upper Tester to the IUT to read a potentially long Characteristic Value from the Lower Tester by specifying the Characteristic handle e.g., GATT_ReadLongRequest (handle) in each of the following four steps.
Step 1: Characteristic Value is long, length \( m \times \text{ATT}_{-} \text{MTU-1} + n \) octets

The Upper Tester will specify the handle of a long Characteristic Value with a length of \( m \times \text{ATT}_{-} \text{MTU-1} + n \) with \( m \geq 1 \) and \( 1 \leq n < \text{ATT}_{-} \text{MTU-1} \) contained in the Lower Tester.
Step 2: Characteristic Value is long, length $m \times \text{ATT}_\text{MTU} - 1$ octets

The Upper Tester will specify the handle of a long Characteristic Value with a length of $m \times \text{ATT}_\text{MTU} - 1$ with $m \geq 2$ contained in the Lower Tester.
Step 3: Characteristic Value is short, length ATT_MTU-1 octets

The Upper Tester will specify the handle of a short Characteristic Value with a length of ATT_MTU-1 contained in the Lower Tester.

Step 4: Characteristic Value is short, length less than ATT_MTU-1 octets

The Upper Tester will specify the handle of a short Characteristic Value with a length less than ATT_MTU-1 contained in the Lower Tester.
Note: In steps 3 and 4, the Lower Tester chooses to treat the Characteristic as of variable length and thus does not respond with ATT_Error_Response (Attribute Not Long) to an ATT_Read_Blob_Request.

Note: It is recommended to execute steps 1 to 4 in an arbitrary sequence not known to the IUT.

- Expected Outcome

**Pass Verdict**

The IUT sends correctly formatted ATT_Read_Blob_Request commands (0x0C) to the Lower Tester. Note that the first request may be an ATT_Read_Request; in that case the Lower Tester shall reply with an ATT_Read_Response, and the IUT shall detect that the Characteristic Value is long, and continue with ATT_Read_Blob_Requests.

The ATT_Read_Blob_Request and optional first ATT_Read_Request specifies the handle of the Characteristic Value to be read and the offset value of the first octet to be read. The offset for the first request is 0x0000; subsequent offset values are sequential values of (ATT_MTU-1). The IUT shall detect the end of the long Characteristic Value by the size of the last part attribute value which is then less than (ATT_MTU-1).

The complete Characteristic Value reported to the Upper Tester matches the value reported by the Lower Tester. On detection of the last blob, the IUT shall report the complete Long Characteristic Value to the Upper Tester.

**4.5.27 GATT/CL/GAR/BI-12-C [Read Long Characteristic Value – Read not permitted]**

- Test Purpose

Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to read not permitted.

- Reference

[1] 4.8.3

[5] 3.4.1.1, 3.4.4.5

- Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester is selected.

- Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadLongRequest (handle). The IUT sends an ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadLongRes. If an ATT_Read_Blob_Request is used, the MSC is shown below; if an
ATT_Read_Request is used, the MSC is shown in GATT/CL/GAR/BI-02-C [Read Characteristic Value – Read Not Permitted].

**Expected Outcome**

**Pass Verdict**

The IUT sends a correctly formatted ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadLongRes.

### 4.5.28 GATT/CL/GAR/BI-13-C [Read Long Characteristic Value – Invalid Offset]

**Test Purpose**

Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to invalid offset.

**Reference**

[1] 4.8.3

[5] 3.4.1.1, 3.4.4.5

**Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
A handle of a Characteristic Value in the Lower Tester that permits reading is selected. The offset is set to a value greater than the length of the selected Characteristic Value.

• Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadLongRequest (handle, offset). The IUT sends an ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadLongRes. If an ATT_Read_Blob_Request is used, the MSC is shown below; if an ATT_Read_Request is used, the MSC is shown in GATT/CL/GAR/BI-01-C [Read Characteristic Value – Invalid Handle].

• Expected Outcome

   Pass Verdict

   The IUT sends a correctly formatted ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester.

   Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadLongRes.

4.5.29 GATT/CL/GAR/BI-14-C [Read Long Characteristic Value – Invalid Handle]

• Test Purpose

   Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to invalid handle.
• Reference
  [1] 4.8.3
  [5] 3.4.1.1, 3.4.4.5

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
  The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadLongRequest (handle). The IUT sends an ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadLongRes. If an ATT_Read_Blob_Request is used, the MSC is shown below; if an ATT_Read_Request is used, the MSC is shown in GATT/CL/GAR/BI-01-C [Read Characteristic Value – Invalid Handle].

  ![Diagram]

• Expected Outcome
  Pass Verdict
  The IUT sends a correctly formatted ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester.

  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadLongRes.
4.5.30 GATT/CL/GAR/BI-15-C [Read Long Characteristic Value – Insufficient Authorization]

- **Test Purpose**
  Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to insufficient authorization.

- **Reference**
  
  [1] 4.8.3
  
  [5] 3.4.1.1, 3.4.4.5

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  
  A handle of a Characteristic Value in the Lower Tester that requires read authorization is selected.
  
  No authorization procedure has been performed between the IUT and the Lower Tester.

- **Test Procedure**
  The Upper Tester sends a command to the IUT to initiate the test, e.g., `GATT_ReadLongRequest (handle)`. The IUT sends an `ATT_Read_Blob_Request` or `ATT_Read_Request` to the Lower Tester. When the IUT receives an `ATT_Error_Response` it sends the result to the Upper Tester, e.g., `GATT_ReadLongRes`. If an `ATT_Read_Blob_Request` is used, the MSC is shown below; if an `ATT_Read_Request` is used, the MSC is shown in GATT/CL/GAR/BI-03-C [Read Characteristic Value – Insufficient Authorization].

![Diagram of test procedure](image-url)
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadLongRes.

4.5.31 GATT/CL/GAR/BI-16-C [Read Long Characteristic Value – Insufficient Authentication]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to insufficient authentication.

• Reference

[1] 4.8.3
[5] 3.4.1.1, 3.4.4.5

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester that requires read authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadLongRequest (handle). The IUT sends an ATT_Read_Blob_Request or ATT_Read_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadLongRes. If an ATT_Read_Blob_Request is used, the MSC is shown below; if an ATT_Read_Request is used, the MSC is shown in GATT/CL/GAR/BI-04-C [Read Characteristic Value – Insufficient Authentication].
• Expected Outcome

**Pass Verdict**

The IUT sends a correctly formatted `ATT_Read_Blob_Request` or `ATT_Read_Request` to the Lower Tester.

Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadLongRes`.

**4.5.32 GATT/CL/GAR/BI-17-C [Read Long Characteristic Value – Insufficient Encryption Key Size]**

• Test Purpose

Verify Generic Attribute Profile client behavior when the Read Long Characteristic Value procedure fails due to insufficient encryption key size.

• Reference

[1] 4.8.3

[5] 3.4.1.1, 3.4.4.5

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
A handle of a Characteristic Value in the Lower Tester that requires encryption with a key longer than the key used to establish the encrypted link is selected.

- **Test Procedure**

  The Upper Tester sends a command to the IUT to initiate the test, e.g., `GATT_ReadLongRequest (handle)`. The IUT sends an `ATT_Read_Blob_Request` or `ATT_Read_Request` to the Lower Tester. When the IUT receives an `ATT_Error_Response` it sends the result to the Upper Tester, e.g., `GATT_ReadLongRes`. If an `ATT_Read_Blob_Request` is used, the MSC is shown below; if an `ATT_Read_Request` is used, the MSC is shown in GATT/CL/GAR/BI-05-C [Read Characteristic Value – Insufficient Encryption Key Size].

  - **Expected Outcome**

  **Pass Verdict**

  The IUT sends a correctly formatted `ATT_Read_Blob_Request` or `ATT_Read_Request` to the Lower Tester.

  Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadLongRes`.

  **4.5.33 GATT/SR/GAR/BV-04-C [Read Long Characteristic Value - from Server]**

- **Test Purpose**

  Verify that a Generic Attribute Profile server can support reading a long Characteristic Value selected by handle.

- **Reference**

  [1] 4.8.3

  [5] 3.4.4.5, 3.4.4.6
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains at least one valid long Characteristic and the handle of this Characteristic is known to the test system.

The Lower Tester has the necessary security permissions to read the value of the Characteristic from the IUT.

• Test Procedure

Send a series of ATT_Read_Blob_Request (handle, offset) commands from the Lower Tester to the IUT to read all parts of the value of a long characteristic.

• Expected Outcome

Pass Verdict

The IUT sends correctly formatted ATT_Read_Blob_Responses (0x0D) to the Lower Tester. The responses contain all parts of the value of the long characteristic that has been read.

If the Lower Tester sends an ATT_Read_Blob_Request with the offset greater than the length of the Characteristic Value then the IUT sends an ATT_Read_Blob_Responses with a part attribute value of length 0 to the Lower Tester. If the value offset of the Read Blob Request is equal to the length of the attribute value, then the length of the part attribute value in the response shall be zero.

Each response does not exceed any negotiated ATT_MTU.

Each IUT response occurs within the applicable timeout (3.3.3/ATT [5]).
4.5.34 GATT/SR/GAR/BI-12-C [Read Long Characteristic Value - Read Not Permitted Response]

• Test Purpose
  Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request to a non-readable Characteristic Value and issue a Read Not Permitted Response.

• Reference
  [1] 4.8.3
  [5] 3.4.1.1, 3.4.4.5

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a long Characteristic Value in the IUT that does not permit reading is selected.

• Test Procedure
  Send an ATT_Read_Blob_Request (handle, offset=0x0000) from the Lower Tester to request IUT using the selected handle.

  ATT_Bearer established over selected transport (4.2).
  ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

  ATT_Read_Blob_Request (Code = 0x0C, handle, Offset = 0x0000) → IUT
  ATT_Error_Response (Code = 0x01, 0x0C, handle, Error Code = 0x02)

• Expected Outcome
  Pass Verdict

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x02, Read Not Permitted.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.35 GATT/SR/GAR/BI-13-C [Read Long Characteristic Value - Invalid Offset Response]

- Test Purpose
  Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request with an invalid offset and issue an Invalid Offset Response.

- Reference
  [1] 4.8.3
  [5] 3.4.1.1, 3.4.4.5

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  A handle of a long Characteristic Value in the IUT that permits reading is selected. The offset is set to a value greater than the length of the Characteristic Value.

- Test Procedure
  Send an ATT_Read_Blob_Request (handle, offset) command from the Lower Tester to the IUT.

```
+-----------------+          +-----------------+          +-----------------+
| Lower Tester    |          | IUT             |          | Upper Tester     |
+-----------------+          +-----------------+          +-----------------+
          +---------------------------+                      +---------------------------+
          | ATT_Bearer established      | ATT_MTBU may have been exchanged between IUT and Lower Tester (4.2). |
          +---------------------------+                      +---------------------------+
          | ATT_Read_Blob_Request      |                      | ATT_Error_Response         |
          | (Code = 0x0C, handle, offset) |                      | (Code = 0x01, 0x0C, handle, Error Code = 0x07) |
          +---------------------------+                      +---------------------------+
```

- Expected Outcome
  **Pass Verdict**
  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x07, Invalid Offset.
  The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.36 GATT/SR/GAR/BI-14-C [Read Long Characteristic Value - Invalid Handle Response]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request to an invalid handle and issue an Invalid Handle Response.

- **Reference**
  [1] 4.8.3
  [5] 3.4.1.1, 3.4.4.5

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  The Lower Tester selects a handle which is known to be invalid.

- **Test Procedure**
  Send an ATT_Read_Blob_Request (handle, offset=0x0000) command from the Lower Tester to the IUT.

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x01, Invalid Handle.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request and issue an Insufficient Authorization Response.

• Reference
[1] 4.8.3
[5] 3.4.1.1, 3.4.4.5

• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a long Characteristic Value in the IUT that requires read authorization is selected.

No authorization procedure has been performed between the IUT and the Lower Tester.

• Test Procedure
Send an ATT_Read_Blob_Request (handle, offset=0x0000) command from the Lower Tester to the IUT.

ATT_Bearer established over LE transport (4.2.1.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

ATT_Read_Blob_Request
(Code = 0x0C, handle, Offset = 0x0000)

ATT_Error_Response
(Code = 0x01, 0x0C, handle, Error Code = 0x08)

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x08, Insufficient Authorization.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.38 GATT/SR/GAR/BI-16-C [Read Long Characteristic Value – Insufficient Authentication]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request and issue an Insufficient Authentication Response.

• Reference
[1] 4.8.3
[5] 3.4.1.1, 3.4.4.5

• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a long Characteristic Value in the IUT that requires read authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure
Send an ATT_Read_Blob_Request (handle, offset=0x0000) command from the Lower Tester to the IUT.

```
Lower Tester    IUT    Upper Tester
ATT Bearer established over LE transport (4.2.1.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).
ATT_Read_Blob_Request
(Code = 0x0C, handle, Offset = 0x0000)
ATT_Error_Response
(Code = 0x01, 0x0C, handle, Error Code = 0x05)
```

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.39 GATT/SR/GAR/BI-17-C [Read Long Characteristic Value – Insufficient Encryption Key Size]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Long Characteristic Value Request and issue an Insufficient Encryption Key Size Response.

- **Reference**
  [1] 4.8.3
  [5] 3.4.1.1, 3.4.4.5

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a long Characteristic Value in the IUT that requires encryption with a key longer than the key used to establish the encrypted link is selected.

- **Test Procedure**
  Send an ATT_Read_Blob_Request (handle, offset=0x0000) command from the Lower Tester to the IUT.

- **Expected Outcome**
  Pass Verdict

  The IUT sends an *ATT_Error_Response* (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0C. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x0C, Insufficient Encryption Key Size.
The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.40 GATT/CL/GAR/BV-05-C [Read Multiple Characteristic Values – by client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can read multiple Characteristic Values selected by a set of handles.

- **Reference**
  - [1] 4.8.4
  - [5] 3.4.4.7, 3.4.4.8

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A set of at least two handles of Characteristic Values in the Lower Tester that permit reading is selected. The combined length of the set of Characteristic Values is less than (ATT_MTU – 1).

  If the characteristics permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  Send a request from the Upper Tester to the IUT to read a set of Characteristic Values selected by a set of handles, e.g., GATT_ReadMultReq.

  ![Diagram](https://via.placeholder.com/150)

  **ATT_Read_Multiple_Request** (Code = 0x0E, set of handles)

  **ATT_Read_Multiple_Response** (Code = 0x0F, set of values)

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Read_Multiple_Request (0x0E) to the Lower Tester.
The characteristic handles are set to valid handles delivered by the Upper Tester.

The IUT receives the `ATT_Read_Multiple_Response` (0x0F) sent by the Lower Tester.

The IUT sends the received responses to the Upper Tester, e.g., `GATT_ReaMultRes` with the same set of values reported by the Lower Tester.

4.5.41 GATT/CL/GAR/BI-18-C [Read Multiple Characteristic Values – Read not permitted]

- **Test Purpose**
  Verify Generic Attribute Profile client behavior when the Read Multiple Characteristic Values procedure fails due to read not permitted.

- **Reference**
  
  [1] 4.8.4
  
  [5] 3.4.1.1, 3.4.4.7

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A set of at least two handles of Characteristic Values in the Lower Tester is selected.

- **Test Procedure**
  The Upper Tester sends a command to the IUT to initiate the test, e.g., `GATT_ReadMultReq(set of handles)`. The IUT sends an `ATT_Read_Multiple_Request` to the Lower Tester. When the IUT receives an `ATT_Error_Response` it sends the result to the Upper Tester, e.g., `GATT_ReadMultRes`.
• **Expected Outcome**

**Pass Verdict**

The IUT sends a correctly formatted ATT_Read_Multiple_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

### 4.5.42 GATT/CL/GAR/BI-19-C [Read Multiple Characteristic Values – Invalid Handle]

• **Test Purpose**

Verify Generic Attribute Profile client behavior when the Read Multiple Characteristic Values procedure fails due to invalid handle.

• **Reference**

[1] 4.8.4

[5] 3.4.1.1, 3.4.4.7

• **Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• **Test Procedure**

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadMultReq(set of handles). The IUT sends an ATT_Read_Multiple_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadMultRes.
• Expected Outcome

**Pass verdict**

The IUT sends a correctly formatted `ATT_Read_Multiple_Request` to the Lower Tester.

Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_ReadMultRes`.

4.5.43 GATT/CL/GAR/BI-20-C [Read Multiple Characteristic Values – Insufficient Authorization]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Read Multiple Characteristic Values procedure fails due to insufficient authorization.

• Reference

[1] 4.8.4

[5] 3.4.1.1, 3.4.4.7

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of at least two handles of Characteristic Values in the Lower Tester of which one requires read authorization is selected.

• Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., `GATT_ReadMultReq(set of handles)`. The IUT sends an `ATT_Read_Multiple_Request` to the Lower Tester. When the IUT receives an `ATT_Error_Response` it sends the result to the Upper Tester, e.g., `GATT_ReadMultRes`. 
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_Multiple_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

4.5.44 GATT/CL/GAR/BI-21-C [Read Multiple Characteristic Values – Insufficient Authentication]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Read Multiple Characteristic Values procedure fails due to insufficient authentication.

• Reference

[1] 4.8.4

[5] 3.4.1.1, 3.4.4.7

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of at least two handles of Characteristic Values in the Lower Tester of which one requires read authentication is selected.
**Test Procedure**

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadMultReq(set of handles). The IUT sends an ATT_Read_Multiple_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

- **Expected Outcome**
  
  **Pass Verdict**
  
  The IUT sends a correctly formatted ATT_Read_Multiple_Request to the Lower Tester.

  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

### 4.5.45 GATT/CL/GAR/BI-22-C [Read Multiple Characteristic Values – Insufficient Encryption Key Size]

- **Test Purpose**
  
  Verify Generic Attribute Profile client behavior when the Read Multiple Characteristic Values procedure fails due to encryption key size.

- **Reference**
  
  [1] 4.8.4

  [5] 3.4.1.1, 3.4.4.7

- **Initial Condition**
  
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of at least two handles of Characteristic Values in the Lower Tester of which one requires encryption with a key longer than the key used to establish the encrypted link is selected.

• Test Procedure

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadMultReq(set of handles). The IUT sends an ATT_Read_Multiple_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_Multiple_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_ReadMultRes.

4.5.46 GATT/SR/GAR/BV-05-C [Read Multiple Characteristic Values – from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support reading multiple Characteristic Values selected by a set of handles.

• Reference

[1] 4.8.4

[5] 3.4.4.7, 3.4.4.8
• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of at two handles of Characteristic Values in the IUT that permit reading is selected.

The Lower Tester has the necessary security permissions from the IUT to read the characteristics.

• Test Procedure
Send an ATT_Read_Multiple_Request (0x0E) from the Lower Tester to the IUT using the selected set of handles.

ATT_Read_Multiple_Request (Code = 0x0E, set of handles)

ATT_Read_Multiple_Response (Code = 0x0F, set of values)

• Expected Outcome
  Pass Verdict
The IUT sends ATT_Read_Multiple_Response (0x0F) to the Lower Tester.

The Characteristic Values are set to the values of the characteristics selected by the set of handles in the ATT_Read_Multiple_Request.

The response size does not exceed any negotiated ATT_MTU. If the combined length of the Characteristic Values are longer than (ATT_MTU – 1) then the first (ATT_MTU – 1) octets are included in this response.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.47 GATT/SR/GAR/BI-18-C [Read Multiple Characteristic Values – Read not permitted]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Multiple Characteristic Values Request including any non-readable Characteristic Value and issue a Read Not Permitted Response.

• Reference
[1] 4.8.4
[5] 3.4.4.7, 3.4.4.8

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of at two handles of Characteristic Values in the IUT of which one does not permit reading is selected.

• Test Procedure
Send an ATT_Read_Multiple_Request from the Lower Tester to the IUT using the selected set of handles.

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0E. The Attribute Handle in Error parameter is set to the first handle causing the error. The Error code is set to 0x02, Read Not Permitted.
The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.48 GATT/SR/GAR/BI-19-C [Read Multiple Characteristic Values – Invalid Handle]

- Test Purpose
  Verify that a Generic Attribute Profile server can detect and reject a Read Multiple Characteristic Value Request including an unsupported Characteristic Value handle and issue an Invalid Handle Response.

- Reference
  [1] 4.8.4
  [5] 3.4.4.7, 3.4.4.8

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  The Lower Tester selects a set of two handles of which one is known to be invalid.

- Test Procedure
  Send an ATT_Read_Multiple_Request from the Lower Tester to the IUT using the selected set of handles.
• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0E. The Attribute Handle in Error parameter is set to the first handle causing the error. The Error code is set to 0x01, Invalid Handle.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.49 GATT/SR/GAR/BI-20-C [Read MultipleCharacteristic Values – Insufficient Authorization]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Read Multiple Characteristic Value Request and issue an Insufficient Authorization Response.

• Reference

[1] 4.8.4

[5] 3.4.4.7, 3.4.4.8

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of two handles of Characteristic Values in the IUT of which one requires read authorization is selected.

No authorization procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

Send an ATT_Read_Multiple_Request from the Lower Tester to the IUT using the selected set of handles.
• Expected Outcome

Pass Verdict

The IUT sends an \texttt{ATT\_Error\_Response} (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0E. The Attribute Handle in Error parameter is set to the first handle causing the error. The Error code is set to 0x08, Insufficient Authorization.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.50 GATT/SR/GAR/BI-21-C [Read Multiple Characteristic Values – Insufficient Authentication]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Read Multiple Characteristic Value Request and issue an Insufficient Authentication Response.

• Reference

[1] 4.8.4

[5] 3.4.4.7, 3.4.4.8

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of two handles of Characteristic Values in the IUT of which one requires read authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.
• Test Procedure
Send an ATT_Read_Multiple_Request from the Lower Tester to the IUT using the selected set of handles.

L2CAP Connection Established over LE transport (4.2.1.2). ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

ATT_Read_Multiple_Request
(Code = 0x0E, set of handles)

ATT_Error_Response
(Code = 0x01, 0x0E, Handle, error code = 0x05)

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0E. The Attribute Handle in Error parameter is set to the first handle causing the error. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.51 GATT/SR/GAR/BI-22-C [Read Multiple Characteristic Values – Insufficient Encryption Key Size]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Multiple Characteristic Value Request and issue an Insufficient Encryption Key Size Response.

• Reference
[1] 4.8.4
[5] 3.4.4.7, 3.4.4.8

• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A set of two handles of Characteristic Values in the IUT of which one requires encryption with a key longer than the key used to establish the encrypted link is selected.

- **Test Procedure**
  
  Send an ATT_Read_Multiple_Request from the Lower Tester to the IUT using the selected set of handles.

  ![Diagram]

- **Expected Outcome**
  
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0E. The Attribute Handle in Error parameter is set to the first handle causing the error. The Error code is set to 0x0C, Insufficient Encryption Key Size.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

**4.5.52 GATT/CL/GAR/BV-06-C [Read Characteristic Descriptors – by Client]**

- **Test Purpose**
  
  Verify that a Generic Attribute Profile client can read a characteristic descriptor selected by handle.

- **Reference**
  
  [1] 4.12.1
  
  [5] 3.4.4.3, 3.4.4.4

- **Initial Condition**
  
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

If the characteristic descriptor permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure

Send a request from the Upper Tester to the IUT to read a characteristic descriptor value from the Lower Tester by specifying the descriptor handle e.g., GATT_ReadReq.

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_Request (0x0A) to the Lower Tester.

The characteristic handle parameter is set to a valid handle sent by the Upper Tester.

The IUT receives the ATT_Read_Response (0x0B) sent by the Lower Tester.

The IUT sends the received response to the Upper Tester. e.g., GATT_ReadRes

The descriptor value matches the value delivered by the Lower Tester.

4.5.53 GATT/SR/GAR/BV-06-C [Read Characteristic Descriptors – from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support reading a characteristic descriptor selected by handle.

• Reference

[1] 4.12.1

[5] 3.4.4.3, 3.4.4.4
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester has the necessary security permissions from the IUT to read the characteristic descriptor.

• Test Procedure

Send an ATT_Read_Request from the Lower Tester to the IUT to read a characteristic descriptor value by specifying the Descriptor handle e.g., ATT_Read_Request.

• Expected Outcome

Pass Verdict

The IUT sends ATT_Read_Response (0x0B) to the Lower Tester.

The Descriptor value is set to the value of the Descriptor identified by the attribute handle in the ATT_Read_Request.

The response size does not exceed any negotiated ATT_MTU. If the Characteristic Value is longer than (ATT_MTU – 1) then the first (ATT_MTU – 1) octets are included in this response.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.5.54 GATT/CL/GAR/BV-07-C [Read Long Characteristic Descriptor - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can read a characteristic descriptor by selected handle. The Characteristic Descriptor length is unknown to the client and might be long.
• Reference
  [1] 4.12.2
  [5] 3.4.4.5, 3.4.4.6

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  If the characteristic descriptor permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure
  Send a command from the Upper Tester to the IUT to read a potentially long characteristic descriptor from the Lower Tester by specifying the characteristic handle e.g., GATT_ReadLongDescriptorReq (handle) in each of the following four steps:

  Step 1: Characteristic Descriptor is long, length $m \times $ATT_MTU-1 + n octets

  The Upper Tester will specify the handle of a long characteristic descriptor with a length of $m \times $ATT_MTU-1 + n with $m \geq 1$ and $1 \leq n < $ATT_MTU-1 contained in the Lower Tester.
Step 2: Characteristic Descriptor is long, length $m \times \text{ATT_MTU} - 1$ octets

The Upper Tester will specify the handle of a long Characteristic Descriptor with a length of $m \times \text{ATT_MTU} - 1$ with $m \geq 2$ contained in the Lower Tester.
Step 3: Characteristic Descriptor is short, length ATT_MTU - 1 octets

The Upper Tester will specify the handle of a long Characteristic Descriptor with a length of ATT_MTU - 1 contained in the Lower Tester.

Step 4: Characteristic Descriptor is SHORT, length less than ATT_MTU - 1 octets

The Upper Tester will specify the handle of a long Characteristic Descriptor with a length of less than ATT_MTU - 1 contained in the Lower Tester.
Note: In steps 3 and 4, the Lower Tester chooses to treat the Characteristic Descriptor as of variable length and thus does not respond with ATT_Error_Response (Attribute Not Long) to an ATT_Read_Blob_Request.

Note: It is recommended to execute steps 1 to 4 in an arbitrary sequence not known to the IUT.

- Expected Outcome

**Pass Verdict**

The IUT sends correctly formatted ATT_Read_Blob_Request commands (0x0C) to the Lower Tester. Note that the first request may be an ATT_Read Request; in that case the Lower Tester shall reply with an ATT_Read_Response and the IUT shall detect that the Characteristic Descriptor is long, and continue with ATT_Read_Blob_Requests.

The ATT_Read_Blob_Request and optional first ATT_Read_Request specifies the handle of the Characteristic Descriptor to be read and the offset value of the first octet to be read. The offset for the first request is 0x0000; subsequent offset values are sequential values of (ATT_MTU-1).

The IUT shall detect the end of the long Characteristic Descriptor by the size of the last partial attribute value which is then less than (ATT_MTU-1).

When the IUT receives the responses from the Lower Tester, it may concatenate them into a single long Characteristic Descriptor and send it (e.g., a GATT_ReadLongDescriptorRes) to the Upper Tester.

The complete Long Characteristic Descriptor reported to the Upper Tester matches the value reported by the Lower Tester.

### 4.5.55 GATT/CL/GAR/BI-34-C [Read Characteristic Value – Invalid Transport Access over BR/EDR]

- **Test Purpose**

Verify Generic Attribute Profile client behavior when an attempt to use BR/EDR transport to execute the Read Characteristic Value procedure on a characteristic contained within a service defined for use only over LE transport.

- **Reference**

[1] 4.4, 4.8.1

[5] 3.4.1.1, 3.4.4.3

- **Initial Condition**

A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over BR/EDR.

The Lower Tester is a BR/EDR/LE device.

The Lower Tester implements a GATT-based service defined for use only over the LE transport. That GATT-based profile contains a readable characteristic. The handle of that characteristic is available to the Upper Tester.
• Test Procedure
The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle); the Upper Tester provides the handle of the characteristic contained within the GATT-based profile which is defined of use only the LE transport.

![Diagram of ATT Bearer established over BR/EDR transport]

• Expected Outcome
Pass Verdict
The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester, including the handle specified by the Upper Tester.

Upon receiving an ATT_Error_Response from the Lower Tester containing the <Application Error> the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.

4.5.56 GATT/CL/GAR/BI-35-C [Read Characteristic Value – Invalid Transport Access over LE]

• Test Purpose
Verify Generic Attribute Profile client behavior when an attempt to use LE transport to execute the Read Characteristic Value procedure on a characteristic contained within a service defined for use only over BR/EDR transport.

• Reference
[1] 4.4, 4.8.1

[5] 3.4.1.1, 3.4.4.3
• **Initial Condition**

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester is a BR/EDR/LE device.

The Lower Tester implements a GATT-based service defined for use only over the BR/EDR transport. That GATT-based profile contains a readable characteristic. The handle of that characteristic is available to the Upper Tester.

• **Test Procedure**

The Upper Tester sends a command to the IUT to initiate the test, e.g., GATT_ReadReq(handle); the Upper Tester provides the handle of the characteristic contained within the GATT-based profile which is defined of use only the BR/EDR transport.

• **Expected Outcome**

**Pass Verdict**

The IUT sends a correctly formatted ATT_Read_Request to the Lower Tester, including the handle specified by the Upper Tester.

Upon receiving an ATT_Error_Response from the Lower Tester containing the <Application Error> the IUT sends the result to the Upper Tester, e.g., GATT_ReadRes.
4.5.57 GATT/SR/GAR/BV-07-C [Read Long Characteristic Descriptor - from Server]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can support reading a Long Characteristic Descriptor selected by handle.

- **Reference**
  [1] 4.12.2
  [5] 3.4.4.5, 3.4.4.6

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  
  The IUT contains at least one valid long characteristic and the handle of this characteristic is known to the test system, from IXIT [10] or from Discovery (Section 4.4).
  
  The Lower Tester has the necessary security permissions to read the value of the characteristic descriptor from the IUT.

- **Test Procedure**
  Send a series of ATT_Read_Blob_Request (handle, offset) commands from the Lower Tester to the IUT to read all parts of the value of a long characteristic descriptor.

- **Expected Outcome**
  **Pass Verdict**
  
  The IUT sends correctly formatted ATT_Read_Blob_Responses (0x0D) or ATT_Read_Request to the Lower Tester.
The responses contain all parts of the value of the Long Characteristic Descriptor.

If the Lower Tester sends an ATT_Read_Blob_Request with the offset greater than the length of the Characteristic Value then the IUT sends an ATT_Read_Blob_Response to the Lower Tester. If the value offset of the Read Blob Request is equal to the length of the attribute value, then the length of the part attribute value in the response shall be zero.

Each response does not exceed any negotiated ATT_MTU.

Each IUT response occurs within the applicable timeout (3.3.3/ATT [5]).

4.5.58 GATT/SR/GAR/BV-08-C [Read Behind Long Characteristic Descriptor - from Server]

• Test Purpose
Verify that a Generic Attribute Profile server returns a 0-length part value when reading a Long Characteristic Descriptor selected by handle with an offset equal to the length of the Long Characteristic Value.

• Reference

[5] 3.4.4.5, 3.4.4.6

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains at least one valid Long Characteristic and the handle of this Characteristic is known to the test system, from IXIT [10] or from Discovery (Section 4.4).

The Lower Tester has the necessary security permissions to read the value of the Characteristic Descriptor from the IUT.

• Test Procedure
Send a series of ATT_Read_Blob_Request (handle, offset) commands from the Lower Tester to the IUT to read all parts of the value of a Long Characteristic Descriptor.

When the size of the part attribute value in the received ATT_Read_Blob_Request is shorter than ATT_MTU-1, the Lower Tester calculates the length of the Characteristic Value and sends an ATT_Read_Blob_Request with the offset set to that length.
**4.5.59 GATT/SR/GAR/BI-34-C [Read Characteristic Value - Invalid Transport Access over LE]**

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request, delivered over LE transport, to a readable Characteristic Value included in a service defined only over BR/EDR transport, and issue an Application Error Response.

- **Expected Outcome**
  **Pass Verdict**

The IUT sends correctly formatted ATT_Read_Blob_Responses (0x0D) or ATT_Read_Request to the Lower Tester.

The responses contain all parts of the value of the Long Characteristic Descriptor.

The ATT_Read_Blob_Response (0x0D) to the last ATT_Read_Blob_Request contains a 0-length part attribute value.

Each response does not exceed any negotiated ATT_MTU.

Each IUT response occurs within the applicable timeout (3.3.3/ATT [5]).

- **Reference**
  [1] 4.4, 4.8.1

  [5] 3.4.1.1, 3.4.4.3
• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT and Lower Tester are BR/EDR/LE devices.

The IUT implements a GATT-based service defined for use only over the BR/EDR transport. That GATT-based profile contains a readable characteristic. The handle of that characteristic is available to the Upper Tester via IXIT [10].

• Test Procedure

Send an ATT_Read_Request from the Lower Tester to the IUT using the selected handle over the LE transport.

- ATT_Error_Response (Code = 0x01, 0x0A, handle, Error Code = <Application Error>)

• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to an <Application Error> code as defined in 3.3/ATT [5].

The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.5.60 GATT/SR/GAR/BI-35-C [Read Characteristic Value - Invalid Transport Access over BR/EDR]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Read Characteristic Value Request, delivered over BR/EDR transport, to a readable Characteristic Value included in a service defined only over LE transport, and issue an Application Error Response.

• Reference
[1] 4.4, 4.8.1
[5] 3.4.1.1, 3.4.4.3

• Initial Condition
A preamble procedure defined in Section 4.2.1.1 is used to set up the transport and L2CAP channel over BR/EDR.

The IUT and Lower Tester are BR/EDR/LE devices.

The IUT implements a GATT-based service defined for use only over the LE transport. That GATT-based profile contains a readable characteristic. The handle of that characteristic is available to the Upper Tester via IXIT [10].

• Test Procedure
Send an ATT_Read_Request from the Lower Tester to the IUT using the selected handle over BR/EDR transport.
• Expected Outcome
  
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x0A. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to an &lt;Application Error&gt; code as defined in 3.3/ATT.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.6 Write

The test group objective is to verify Generic Attribute Profile Writing of Characteristic Values and Characteristic Descriptors.

#### 4.6.1 GATT/CL/GAW/BV-01-C [Write without Response - by Client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can issue a characteristic Write without response.

- **Reference**
  - [1] 4.9.1
  - [5] 3.4.5.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that permits writing is selected.

  If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  Send a command from the Upper Tester to request the IUT to write a value of a characteristic in the Lower Tester specifying the handle that is to be written e.g., GATT_Write_Command (handle, value).
**4.6.2  GATT/SR/GAW/BV-01-C [Write Without Response - to Server]**

- **Test Purpose**
  Verify that a Generic Attribute Profile server can support a write to a characteristic without response.

- **Reference**
  [1] 4.9.1
  [5] 3.4.5.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the IUT that permits writing is selected, if indicated in the IXIT [10]. The IXIT [10] also indicates if this characteristic is readable. If that characteristic also supports reading, then ALT 2 may be used.

  The Lower Tester has the necessary security permissions from the IUT to read and write a characteristic.
• Test Procedure

Send an ATT_Write_Command (handle, value) from the Lower Tester to the IUT.

If the Characteristic is readable, send an ATT_Read_Request (handle) to verify that value specified in the ATT_Write_Command has been written in the IUT.

• Expected Outcome

Pass Verdict

ALT 1: The IUT receives the ATT_Write_Command and indicates that via message to the Upper Tester (e.g., GATT_Write_Command (handle, value)).

ALT 2: If the characteristic is readable, the IUT sends an ATT_Read_Response reporting the same value delivered in the ATT_Write_Command within the applicable timeout (3.3.3/ATT [5]).

4.6.3 GATT/CL/GAW/BV-02-C [Write without Response with Authentication - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can write a characteristic with authentication without response.

• Reference

[1] 4.9.2

[5] 3.4.5.3

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that permits writing and requires authentication is selected.

The ATT Bearer is not encrypted.

The Lower Tester is bonded with the IUT, so it has the necessary security permissions from the IUT to write a Characteristic Value.

- **Test Procedure**

  Send a command from the Upper Tester to request the IUT to write a value of a characteristic in the Lower Tester specifying the handle that is to be written e.g., GATT_Signed_Write_Command (handle, value).

  The Lower Tester verifies the signature of the received ATT_Signed_Write_Command.

  ![Diagram](image)

- **Expected Outcome**

  **Pass Verdict**

  The IUT sends a correctly formatted signed ATT_Signed_Write_Command to the Lower Tester and the signature is verified by the Lower Tester.

  The Attribute Handle parameter shall be set to the handle of the characteristic that is to be written. The Characteristic Value of the Signed Write Command is an authenticated value, as defined in the Security Manager (Volume 6 [9]), and matches the value specified by the Upper Tester.

  The size of the ATT_Signed_Write_Command does not exceed any negotiated ATT_MTU.
4.6.4 GATT/SR/GAW/BV-02-C [Write without Response with Authentication – to Server]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can support a characteristic Write with authentication without response.

- **Reference**
  [1] 4.9.2
  [5] 3.4.5.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the IUT that permits writing and requires authentication is selected, if indicated in the IXIT [10]. The IXIT [10] also indicates if this characteristic is readable. If that characteristic also supports reading, then ALT 2 may be used.

  The ATT Bearer is not encrypted.

  The Lower Tester is bonded with the IUT, so it has the necessary security permissions from the IUT to read and write a Characteristic Value.

- **Test Procedure**
  Send an ATT_Signed_Write_Command (0xD2, handle, value, signature) from the Lower Tester to the IUT.

  If the characteristic is readable, send an ATT_Read_Request (0x0A, handle) command to verify that value specified in the ATT_Signed_Write_Command has been written in the IUT.
• Expected Outcome

Pass Verdict

ALT1: The IUT receives the ATT_Signed_Write_Command and indicates that via message to the Upper Tester (e.g., GATT_Write_Command (handle, value)).

ALT 2: If the characteristic is readable, the IUT sends an ATT_Read_Response reporting the same value delivered in the ATT_Signed_Write_Command within the applicable timeout (3.3.3/ATT [5]).

4.6.5 GATT/SR/GAW/BI-01-C [Write without Response with Authentication – Invalid Signature]

• Test Purpose
For a readable characteristic, verify that a Generic Attribute Profile server can detect a characteristic Write with authentication without response in which the signature is invalid and ignore the write command.

• Reference
[1] 4.9.2

• Initial Condition
A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that permits writing and requires authentication is selected, if indicated in the IXIT [10]. The IXIT [10] also indicates if this characteristic is readable.

The Lower Tester and the IUT are bonded so that the Lower Tester can generate ATT_Signed_Write_Commands with valid signatures.
• Test Procedure

ALT 2: If the characteristic is readable, the Lower Tester will issue an ATT_Read_Request to determine the current value of the Characteristic Value. It will wait for an ATT_Read_Response containing the current Characteristic Value.

The Lower Tester will issue an ATT_Signed_Write_Command (0xD2, handle, value, signature) to that characteristic in the IUT using an invalid signature and with a different Characteristic Value than the value reported in the previous step.

ALT 2: If this characteristic is readable, the Lower Tester will issue another ATT_Read_Request to the same characteristic.

• Expected Outcomes

Pass Verdict

ALT 1: The IUT reports an error message to the Upper Tester. Example: a generic ‘GATT_Error’ message that indicates what kind of event happened (signed write command) and the kind of error (insufficient authentication).

ALT 2: If the characteristic is readable, in response to the first ATT_Read_Request the IUT reports an initial value for the selected characteristic in an ATT_Read_Response.

ALT 2: If the characteristic is readable, in response to the second ATT_Read_Request the IUT reports the same value for the selected characteristic in a second ATT_Read_Response.

4.6.6 GATT/CL/GAW/BV-03-C [Write Characteristic Value - by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can write a Characteristic Value selected by handle.

• Reference

[1] 4.9.3
3.4.5.1, 3.4.5.2

- **Initial Condition**

  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that permits writing is selected.

  If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**

  Send a command from the Upper Tester to request the IUT to write the value of a characteristic in the Lower Tester e.g., GATT_WriteRequest (handle, value).

  ![Diagram of L2CAP Connection Established over selected channel (4.2). ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).](image)

  **ATT_Write_Request**
  (Code = 0x12, handle, new value)

  **ATT_Write_Response**
  (Code = 0x13)

  **GATT_WriteRequest**
  (handle, new value)

  **GATT_WriteResponse**

- **Expected Outcome**

  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Write_Request (0x12) to the Lower Tester, specifying the value that is to be written from the Upper Tester.

  The Characteristic handle parameter is set to a valid handle.

  The size of the ATT_Write_Request does not exceed any negotiated ATT_MTU.

  The IUT receives the response from the Lower Tester and sends the GATT_WriteResponse to the Upper Tester.
4.6.7  GATT/CL/GAW/BI-02-C [Write Characteristic Value – Invalid handle]

- Test Purpose
  Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to invalid handle.

- Reference
  [1] 4.9.3
  [5] 3.4.5.1, 3.4.5.2

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

- Test Procedure
  Send a command from the Upper Tester to request the IUT to initiate the test, e.g., GATT_WriteRequest (handle, value). When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteResponse.

- Expected Outcome
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester.

  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.
4.6.8 GATT/CL/GAW/BI-03-C [Write Characteristic Value – Write Not Permitted]

- Test Purpose
  Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to write not permitted.

- Reference
  [1] 4.9.3
  [5] 3.4.5.1, 3.4.5.2

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  A handle of a Characteristic Value in the Lower Tester is selected.

- Test Procedure
  Send a command from the Upper Tester to request the IUT to initiate the test, e.g., GATT_WriteRequest (handle, value). When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteResponse.

- Expected Outcome
  Pass Verdict
  The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester.
  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.
4.6.9 GATT/CL/GAW/BI-04-C [Write Characteristic Value – Insufficient Authorization]

- **Test Purpose**
  Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to insufficient authorization.

- **Reference**
  [1] 4.9.3
  [5] 3.4.5.1, 3.4.5.2

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that requires write authorization is selected.

- **Test Procedure**
  Send a command from the Upper Tester to request the IUT to initiate the test, e.g., GATT_WriteRequest (handle, value). When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteResponse.

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester.
Upon receiving an ATT_Error_Response from the Lower Tester, the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.

4.6.10 GATT/CL/GAW/BI-05-C [Write Characteristic Value – Insufficient Authentication]

- **Test Purpose**
  Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to insufficient authentication.

- **Reference**
  
  [1] 4.9.3
  
  [5] 3.4.5.1, 3.4.5.2

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that requires write authentication is selected.

- **Test Procedure**
  Send a command from the Upper Tester to request the IUT to initiate the test, e.g., GATT_WriteRequest (handle, value). When the IUT receives an ATT_Error_Response, it sends the result to the Upper Tester, e.g., GATT_WriteResponse.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.

4.6.11 GATT/CL/GAW/BI-06-C [Write Characteristic Value – Insufficient Encryption Key Size]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to insufficient encryption key size.

• Reference

[1] 4.9.3

[5] 3.4.5.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester that requires encryption with a key longer than the key used to establish the encrypted link is selected.

• Test Procedure

Send a command from the Upper Tester to request the IUT to initiate the test, e.g., GATT_WriteRequest (handle, value). When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteResponse.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.

4.6.12 GATT/SR/GAW/BV-03-C [Write Characteristic Value - to Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support writing a Characteristic Value selected by handle.

• Reference

[1] 4.9.3
[5] 3.4.5.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT_MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that permits writing is selected.

The Lower Tester has the necessary security permissions from the IUT to read and write the characteristic.

• Test Procedure

Send an ATT_Write_Request (handle, value) from the Lower Tester to the IUT.
Send an ATT_Read_Request (handle) to verify that the value specified in the ATT_Write_Request has been written in the IUT.

* Expected Outcome

**Pass Verdict**

The IUT sends a correctly formatted ATT_Write_Response to the Lower Tester.

The IUT sends an ATT_Read_Response reporting the same value delivered in the ATT_Write_Request.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

**4.6.13 GATT/SR/GAW/BI-02-C [Write Characteristic Value – Invalid Handle Response]**

* Test Purpose

Verify that a Generic Attribute Profile server can detect a Write Characteristic Value Request with an unsupported Characteristic Value handle and issue an Invalid Handle Response.

* Reference

[1] 4.9.3

[5] 3.4.1.1, 3.4.5.2

* Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester selects a handle which is known to be invalid.
• Test Procedure
The Lower Tester sends an ATT_Write_Request to the IUT to using the selected handle.

![Diagram showing L2CAP Connection Established over selected transport (4.2), ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).]

ATT_Write_Request
(Code = 0x12, handle, value)
ATT_Error_Response
(Code = 0x01, 0x12, Handle, Error Code = 0x01)

• Expected Outcome
Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x01, Invalid Handle.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).


• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Write Characteristic Value Request to a non-writeable Characteristic Value and issue a Write Not Permitted Response.

• Reference
[1] 4.9.3
[5] 3.4.1.1, 3.4.5.2

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that does not permit writing is selected.

• Test Procedure
The Lower Tester sends an ATT_Write_Request to the IUT using the selected handle.
• Expected Outcome

**Pass Verdict**

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x03, Write not Permitted.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.15 GATT/SR/GAW/BI-04-C [Write Characteristic Value – Insufficient Authorization]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Write Characteristic Value Request and issue an Insufficient Authorization Response.

• References

[1] 4.9.3

[5] 3.4.1.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that requires write authorization is selected.

No authorization procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

The Lower Tester sends an ATT_Write_Request to the IUT using the selected handle.
• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x08, Insufficient Authorization.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.16 GATT/SR/GAW/BI-05-C [Write Characteristic Value – Insufficient Authentication]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Write Characteristic Value Request and issue an Insufficient Authentication Response.

• References

[1] 4.9.3
[5] 3.4.1.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that requires write authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

The Lower Tester sends an ATT_Write_Request to the IUT using the selected handle.
• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.17 GATT/SR/GAW/BI-06-C [Write Characteristic Value – Insufficient Encryption Key Size]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Write Characteristic Value Request and issue an Insufficient Encryption Key Size Response.

• References

[1] 4.9.3

[5] 3.4.1.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that requires encryption with a key longer than the key used to establish the encrypted link is selected.

• Test Procedure

The Lower Tester sends an ATT_Write_Request to the IUT using the selected handle.
**Expected Outcome**

**Pass Verdict**

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x0C, Insufficient Encryption Key Size.

The IUT responds within the applicable timeout (3.3.3/ATT[5]).

### 4.6.18 GATT/CL/GAW/BV-05-C [Write Long Characteristic Value - by Client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can write a long Characteristic Value selected by handle.

- **Reference**
  [1] 4.9.4  
  [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.  
  MTU shall be smaller than 512, and smaller than the value that has to be written.  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.  
  The Lower Tester contains at least one valid long characteristic and the handle of this long characteristic is known to the IUT.  
  If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest.
• Expected Outcome

**Pass Verdict**

The IUT sends correctly formatted ATT_Prepare_Write_Request commands (0x16) to the Lower Tester.

Each ATT_Prepare_Write_Request specifies the handle of the long characteristic to be written, the offset of the first octet to be written, and part of the value of the long characteristic to be written. The offset parameters are all values of Nx(ATT_MTU-5), with N ranging from 0 (for the first part value) to the last value sufficient to write the entire long characteristic.

After sending all the ATT_Prepare_Write_Requests, the IUT sends a correctly formatted ATT_Execute_Write_Request (0x18).

The size of each ATT_Prepare_Write_Request does not exceed any negotiated ATT_MTU.

### 4.6.19 GATT/CL/GAW/BI-07-C [Write Long Characteristic Value – Invalid Handle]

• **Test Purpose**

Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to invalid handle.

• **Reference**

[1] 4.9.4

[5] 3.4.6.1, 3.4.6.2

• **Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
MTU shall be smaller than 512, and smaller than the value that has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

- Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest (handle, value). The IUT sends an ATT_Prepare_Write_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

4.6.20 GATT/CL/GAW/BI-08-C [Write Long Characteristic Value – Write Not Permitted]

- Test Purpose

Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to write not permitted.

- Reference

[1] 4.9.4

[5] 3.4.6.1, 3.4.6.2
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

MTU shall be smaller than 512, and smaller than the value that has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester is selected.

• Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest (handle, value). The IUT sends an ATT_Prepare_Write_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Prepare_Write_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

4.6.21 GATT/CL/GAW/BI-09-C [Write Long Characteristic Value – Invalid Offset]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to invalid offset.
• Reference

[1] 4.9.4

[5] 3.4.6.1, 3.4.6.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

MTU shall be smaller than 512, and smaller than the value that has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a long Characteristic Value in the Lower Tester that permits writing is selected. The offset is set to a value greater than the length of the selected Characteristic Value.

• Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest (handle, value). The IUT sends an ATT_Prepare_Write_Request to the Lower Tester, When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Prepare_Write_Request to the Lower Tester.

After receiving an ATT_Prepare_Write_Response from the Lower Tester, the IUT sends a correctly formatted ATT_Execute_Write_Request to the Lower Tester.
Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

### 4.6.22 GATT/CL/GAW/BI-11-C [Write Long Characteristic Value – Insufficient Authorization]

- **Test Purpose**
  
  Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to insufficient authorization.

- **Reference**
  
  [1] 4.9.4  
  [5] 3.4.6.1, 3.4.6.2

- **Initial Condition**
  
  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  MTU shall be smaller than 512, and smaller than the value that has to be written.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a Characteristic Value in the Lower Tester that requires write authorization is selected.

  No authorization procedure has been performed between the IUT and the Lower Tester.

- **Test Procedure**
  
  Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest (handle, value). The IUT sends an ATT_Prepare_Write_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Prepare_Write_Request to the Lower Tester.

Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

4.6.23 GATT/CL/GAW/BI-12-C [Write Long Characteristic Value – Insufficient Authentication]

• Test Purpose

Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to insufficient authentication.

• Reference

[1] 4.9.4

[5] 3.4.6.1, 3.4.6.2

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

MTU shall be smaller than 512, and smaller than the value that has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the Lower Tester that requires write authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest (handle, value). The IUT sends an ATT_Prepare_Write_Request to the Lower Tester. When the IUT receives an ATT_Error_Response it sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.
Generic Attribute Profile (GATT) / Test Suite

ATT Bearer established over LE transport (4.2.1.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

- Expected Outcome
  
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Prepare_Write_Request to the Lower Tester.

  Upon receiving an ATT_Error_Response from the Lower Tester the IUT sends the result to the Upper Tester, e.g., GATT_WriteLongResponse.

  **4.6.24 GATT/CL/GAW/BI-13-C [Write Long Characteristic Value – Insufficient Encryption Key Size]**

- Test Purpose

  Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to insufficient encryption key size.

- Reference

  [1] 4.9.4

  [5] 3.4.6.1, 3.4.6.2

- Initial Condition

  A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

  MTU shall be smaller than 512, and smaller than the value that has to be written.

  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT_MTU between the IUT and the Lower Tester.
A handle of a Characteristic Value in the Lower Tester that requires encryption with a key longer than the key used to establish the encrypted link is selected.

- **Test Procedure**
  
  Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., `GATT_WriteLongRequest (handle, value)`. The IUT sends an `ATT_Prepare_Write_Request` to the Lower Tester. When the IUT receives an `ATT_Error_Response` it sends the result to the Upper Tester, e.g., `GATT_WriteLongResponse`.

- **Expected Outcome**

  **Pass Verdict**

  The IUT sends a correctly formatted `ATT_Prepare_Write_Request` to the Lower Tester.

  Upon receiving an `ATT_Error_Response` from the Lower Tester the IUT sends the result to the Upper Tester, e.g., `GATT_WriteLongResponse`.

  **4.6.25 GATT/SR/GAW/BV-05-C [Write Long Characteristic Value - to Server]**

  - **Test Purpose**

    Verify that a Generic Attribute Profile server can support writing a long Characteristic Value selected by handle.

  - **Reference**

    [1] 4.9.4

    [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

  - **Initial Condition**

    A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains a valid characteristic and the handle of this characteristic is known to the Lower Tester.

The Lower Tester has the necessary security permissions to write the value of the characteristic of the IUT.

- **Test Procedure**

Send ATT_Prepare_Write_Requests from the Lower Tester to the IUT to write all parts of the value of a long characteristic, setting the offset to Nx(ATT_MTU-5), with N starting from 0. The handle of the characteristic to be written and the part value to be written is also specified.

The Lower Tester recovers the returned offset and part value of each ATT_Prepare_Write_Response.

After receiving the last ATT_Prepare_Write_Response the Lower Tester sends an ATT_Execute_Write_Request to the IUT.

- **Expected Outcome**

**Pass Verdict**

The IUT sends a series of correctly formatted ATT_Prepare_Write_Responses to the Lower Tester. The offset and part value sent in each response match the offset and part value sent in the corresponding request.

The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.
The responses do not exceed any negotiated ATT_MTU.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

**4.6.26 GATT/SR/GAW/BI-07-C [Write Long Characteristic Value – Invalid Handle Response]**

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect a Write Long Characteristic Request with an invalid characteristic handle and issue an Invalid Handle Response.

- **Reference**
  [1] 4.9.4
  [5] 3.4.1.1, 3.4.6.1

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  The Lower Tester selects a handle that is known to be invalid.

- **Test Procedure**
  The Lower Tester sends an ATT_Prepare_Write_Request (handle, offset = 0x0000, part value) to the IUT.

  ATT_Prepare_Write_Request
  (Code = 0x16, handle, offset = 0x0000, part value)

  ATT_Error_Response
  (Code = 0x01, 0x16, handle, Error Code = 0x01)

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x01, Invalid Handle.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.6.27 GATT/SR/GAW/BI-08-C [Write Long Characteristic Value – Write Not Permitted Response]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can detect and reject a Write Long Characteristic Value Request to a non-writeable long Characteristic Value and issue a Write Not Permitted Response.

- **Reference**
  [1] 4.9.4
  [5] 3.4.1.1, 3.4.6.1

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A handle of a long Characteristic Value in the IUT that does not permit writing is selected.

- **Test Procedure**
  The Lower Tester sends an ATT_Prepare_Write_Request (handle, offset = 0x0000, part value) to the IUT.

  ATT_Bearer established over selected transport (4.2). ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

- **Expected Outcome**
  **Pass Verdict**
  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x03, Write Not Permitted.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).
4.6.28 GATT/SR/GAW/BI-09-C [Write Long Characteristic Value – Invalid Offset Response]

• Test Purpose
Verify that a Generic Attribute Profile server can detect and reject a Write Long Characteristic Request with an invalid offset and issue an Invalid Offset Response.

• Reference
[1] 4.9.4
[5] 3.4.1.1, 3.4.6.1

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a long Characteristic Value in the IUT that permits writing is selected. The offset is set to a value greater than the length of the Characteristic Value.

• Test Procedure
The Lower Tester sends an ATT_Prepare_Write_Request(handle, offset, part value) to the IUT. In response, the IUT sends an ATT_Prepare_Write_Response to the Lower Tester. Then, the Lower Tester sends an ATT_Execute_Write_Request to the IUT.

### Diagram

```
   Lower Tester  IUT    Upper Tester

ATT_Bearer established over selected transport (4.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

ATT_Prepare_Write_Request
(Code = 0x16, handle, offset, part value)

ATT_Prepare_Write_Response
(Code = 0x17, handle, Offset = 0x0000, 1st part value)

ATT_Execute_Write_Request
(Code = 0x16, handle, offset, part value)

ATT_Error_Response
(Code = 0x01, 0x16, handle, Error Code = 0x07)
```
• Expected Outcome

   Pass Verdict

   The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x07, Invalid Offset.

   The IUT responds within the applicable timeout (3.3.3/ATT [5]).


• Test Purpose

   Verify that a Generic Attribute Profile server can detect and reject a Write Long Characteristic Request and issue an Insufficient Authorization Response.

• Reference

   [1] 4.9.4

   [5] 3.4.1.1, 3.4.6.1

• Initial Condition

   A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

   A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

   A handle of a long Characteristic Value in the IUT that requires write authorization is selected.

   No authorization procedure has been performed between the IUT and the Lower Tester.

• Test Procedure

   The Lower Tester sends an ATT_Prepare_Write_Request (handle, offset = 0x0000, part value) to the IUT.
• Expected Outcome

Pass Verdict

The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x08, Insufficient Authorization.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.30 GATT/SR/GAW/BI-12-C [Write Long Characteristic Value – Insufficient Authentication]

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Write Long Characteristic Request and issue an Insufficient Authentication Response.

• Reference

[1] 4.9.4

[5] 3.4.1.1, 3.4.6.1

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a long Characteristic Value in the IUT that requires write authentication is selected.

No authentication procedure has been performed between the IUT and the Lower Tester.
• Test Procedure

The Lower Tester sends an `ATT_Prepare_Write_Request(handle, offset = 0x0000, part value)` to the IUT.

```
Lower Tester   IUT    Upper Tester

ATT Bearer established over LE transport (4.2.1.2).
ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).
```

ATT_Prepare_Write_Request
(Code = 0x16, handle, 
offset =0x0000, part value)

ATT_Error_Response
(Code = 0x01, 0x16, 
handle, Error Code = 0x05)

• Expected Outcome

**Pass Verdict**

The IUT sends an `ATT_Error_Response (code 0x01)` to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x05, Insufficient Authentication.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

**4.6.31 GATT/SR/GAW/BI-13-C [Write Long Characteristic Value – Insufficient Encryption Key Size]**

• Test Purpose

Verify that a Generic Attribute Profile server can detect and reject a Write Long Characteristic Request and issue an Insufficient Encryption Key Size Response.

• Reference

[1] 4.9.4

[5] 3.4.1.1, 3.4.6.1

• Initial Condition

A preamble procedure defined in Section 4.2.1.2 is used to set up the transport and L2CAP channel over LE.

A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
A handle of a long Characteristic Value in the IUT that requires encryption with a key longer than the key used to establish the encrypted link is selected.

- **Test Procedure**
  The Lower Tester sends an ATT_Prepare_Write_Request(handle, offset = 0x0000, part value) to the IUT.

  ![Diagram](image)

  ATT Bearer established over LE transport (4.2.1.2).
  Encrypted link established with insufficient key size (4.2.3).
  ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).

- **Expected Outcome**
  **Pass Verdict**

  The IUT sends an ATT_Error_Response (code 0x01) to the Lower Tester. The Request Opcode in Error parameter is set to 0x16. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x0C, Insufficient Encryption Key Size.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

### 4.6.32 GATT/CL/GAW/BV-06-C [Characteristic Value Reliable Write - by Client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client can reliably write a characteristic selected by handle.

- **Reference**
  [1] 4.9.5
  [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.3 is used to set up an encrypted link.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
The Lower Tester contains at least one valid characteristic and the handle of this characteristic is known to the IUT.

The characteristic has sufficient permissions to allow writing.

If the characteristic permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  1. Send a command from the Upper Tester to the IUT to request the Lower Tester to reliably write a new value to the selected characteristic e.g., `GATT_ReliableWriteRequest` (handle, new value).
  2. When IUT receives an `ATT_Prepare_Write_Response (0x17)` from tester, the IUT issues an `ATT_Execute_Write_Request (0x18)` to the Lower Tester.
  3. When the IUT receives a valid `ATT_Execute_Write_Response`, it reports this to the Upper Tester, e.g., `GATT_ReliableWriteResponse`.

- **Expected Outcome**
  **Pass Verdict**

  As a result of Step 1 in the Test Procedure, the IUT sends a correctly formatted `ATT_Prepare_Write_Request (0x16)` to the Lower Tester. The command specifies the handle, offset, and the value of the characteristic to be written.

  As a result of Step 2 in the Test Procedure, the IUT sends a correctly formatted `ATT_Execute_Write_Request command (0x18)` to the Lower Tester. The flags are set to 0x01 (Immediately write all pending prepared values).

  The IUT receives the response from the Lower Tester and sends the `GATT_ReliableWriteResponse` to the Upper Tester.

  The size of the `ATT_Prepare_Write_Request` does not exceed any negotiated `ATT_MTU`. 
4.6.33 GATT/SR/GAW/BV-06-C [Characteristic Value Reliable Writes - to Server]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can support reliable writing to a Characteristic Value selected by handle.

- **Reference**
  [1] 4.9.5
  [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  The IUT contains a valid characteristic and the handle of this characteristic is known to the test system.
  The Lower Tester has the necessary security permissions to write the value of the characteristic of the IUT.

- **Test Procedure**
  Send an ATT_Prepare_Write_Request command from the Lower Tester to the IUT to prepare to write the value of a characteristic. The ATT_Prepare_Write_Request specifies the handle of the characteristic to be written and the value of the characteristic to be written.
  Send an ATT_Execute_Write_Request command from the Lower Tester to the IUT with flags = 0x01.
• Expected Outcome

**Pass Verdict**

The IUT sends a correctly formatted ATT_Prepare_Write_Response to the Lower Tester. The offset, handle, and value sent in the response match those sent in the request.

The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

4.6.34 GATT/SR/GAW/BV-10-C [Nested Long Characteristic Value Reliable Writes - to Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support nested reliable writing to long Characteristic Values selected by handle.

• Reference

[1] 4.9.5

[5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

• Initial Condition

For this test, the server has to support a sufficient number of ATT_Prepare_Write_Request commands and two prepare queues.

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains two writeable valid long characteristics and the handles of these characteristics are known to the test system.

The Lower Tester has the necessary security permissions to write the values of the characteristics of the IUT.

• Test Procedure

Send an ATT_Prepare_Write_Request command from the Lower Tester to the IUT to prepare to write the 1st part value of a first characteristic. The ATT_Prepare_Write_Request specifies the handle of the first characteristic to be written and the value of the characteristic to be written.

Send an ATT_Prepare_Write_Request command from the Lower Tester to the IUT to prepare to write the 1st part value of a second characteristic. The ATT_Prepare_Write_Request specifies the handle of the second characteristic to be written and the value of the characteristic to be written.

Send ATT_Prepare_Write_Requests from the Lower Tester to the IUT to write the remaining part values of the first characteristic, setting the offset to Nx(ATT_MTU-5), with N starting from 1. The handle of the characteristic to be written and the part value to be written are also specified.
Send ATT_Prepare_Write_Requests from the Lower Tester to the IUT to write the remaining part values of the second characteristic, setting the offset to \(N \times (\text{ATT_MTU} - 5)\), with \(N\) starting from 1. The handle of the characteristic to be written and the part value to be written are also specified.

Send an ATT_Execute_Write_Request command from the Lower Tester to the IUT with flags = 0x01.

- **Expected Outcome**

  **Pass Verdict**

  The IUT sends a series of correctly formatted ATT_Prepare_Write_Responses to the Lower Tester. The offset, handle, and value sent in each response match those sent in the corresponding request.

  The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.
The responses do not exceed any negotiated ATT_MTU.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

The values of the long characteristics stored at handle 1 and handle 2 in the IUT correspond to the values written by the Lower Tester.

### 4.6.35 GATT/SR/GAW/BV-11-C [Characteristic Value Reliable Writes - No Pending Prepared Write Requests]

- **Test Purpose**
  Verify that a Generic Attribute Profile server can support reliable writing to a Characteristic Value selected by handle when there are no pending Prepared Write requests.

- **Reference**
  [1] 4.9.5
  [5] 3.4.6.3, 3.4.6.4

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  The IUT contains a valid characteristic and the handle of this characteristic is known to the test system.

  The Lower Tester has the necessary security permissions to write the value of the characteristic of the IUT.

- **Test Procedure**
  1. The Lower Tester sends an ATT_Execute_Write_Request command to the IUT with flags = 0x01.
  2. The Lower Tester expects an ATT_Execute_Write_Response from the IUT.
  3. The Upper Tester verifies that no new values are written to the target characteristic.
• Expected Outcome

**Pass Verdict**

The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.

The IUT does not write new values to the target characteristic.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

### 4.6.36 GATT/SR/GAW/BV-07-C [Cancel Reliable Write Characteristic – from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support cancel of a reliable write.

• Reference

[1] 4.9.5

[5] 3.4.6.1, 3.4.6.2, 3.4.6.3

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT has at least one valid characteristic. The Lower Tester has sufficient permissions for writing value of this characteristic.

• Test Procedure

Send an ATT_Prepare_Write_Request from the Lower Tester to the IUT to prepare to write the value of a characteristic. The ATT_Prepare_Write_Request specifies the handle of the characteristic to be written, the offset (0x0000) and the value of the characteristic to be written.

Send an ATT_Execute_Write_Request command from the Lower Tester to the IUT with the flags set to 0x00, to cancel the pending reliable write.

Send an ATT_Read_Request command from the Lower Tester to verify that the Characteristic Value has not been written.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Prepare_Write_Response to the Lower Tester. The offset, handle, and value sent the response match those sent in the request.

The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.

The IUT sends a correctly formatted ATT_Read_Response to the Lower Tester containing a value different from the one sent in the ATT_Prepare_Write_Request.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

4.6.37 GATT/CL/GAW/BV-08-C [Write Characteristic Descriptors – by Client]

• Test Purpose

Verify that a Generic Attribute Profile client can write a characteristic descriptor selected by handle.

• Reference

[1] 4.12.3

[5] 3.4.5.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a characteristic descriptor in the Lower Tester that permits writing is selected.
If the characteristic descriptor permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  
  Send a command from the Upper Tester to request IUT to write the value of a characteristic in the Lower Tester e.g., GATT_WriteRequest (handle, value).

  ![Diagram](https://example.com/diagram)

  **ATT_Write_Request**
  (Code = 0x12, handle, new value)

  **ATT_Write_Response**
  (Code = 0x13)

  **GATT_WriteRequest**
  (handle, new value)

  **GATT_WriteResponse**

- **Expected Outcome**
  
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester, specifying the value that is to be written from the Upper Tester.

  The Characteristic handle parameter is set to a valid handle.

  The size of the ATT_Write_Request does not exceed any negotiated ATT_MTU.

  The IUT receives the response from the Lower Tester and sends the GATT_WriteResponse to the Upper Tester.

4.6.38 GATT/SR/GAW/BV-08-C [Write Characteristic Descriptors – from Server]

- **Test Purpose**
  
  Verify that a Generic Attribute Profile server can support writing a characteristic descriptor selected by handle.

- **Reference**
  
  [1] 4.12.3

  [5] 3.4.5.1, 3.4.5.2
• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a characteristic descriptor in the IUT that permits writing is selected.

The Lower Tester has the necessary security permissions from the IUT to read and write the characteristic descriptor.

• Test Procedure
Send an ATT_Write_Request (handle, value) from the Lower Tester to the IUT.

Send an ATT_Read_Request (handle) to verify that the value specified in the ATT_Write_Request has been written in the IUT.

• Expected Outcome
Pass Verdict

The IUT sends a correctly formatted ATT_Write_Response to the Lower Tester.

The IUT sends an ATT_Read_Response reporting the same value delivered in the ATT_Write_Request.

The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.39 GATT/CL/GAW/BV-09-C [Write Long Characteristic Descriptors – by Client]

• Test Purpose
Verify that a Generic Attribute Profile client can write a long characteristic descriptor selected by handle.
• Reference

[1] 4.12.4

[5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

MTU shall be smaller than 512, and smaller than the value which has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester contains at least one valid long characteristic descriptor and the handle of this long characteristic descriptor is known to the IUT.

If the characteristic descriptor permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

• Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic descriptor in the Lower Tester e.g., GATT_WriteLongRequest.

• Expected Outcome

Pass Verdict

The IUT sends correctly formatted ATT_Prepare_Write_Request commands (0x16) to the Lower Tester.
Each ATT_Prepare_Write_Request specifies the handle of the long characteristic to be written, the offset of the first octet to be written, and part of the value of the long characteristic to be written. The offset parameters are all values of \( N \times (\text{ATT_MTU} - 5) \), with \( N \) ranging from 0 (for the first part value) to the last value sufficient to write the entire long characteristic.

After sending all the ATT_Prepare_Write_Requests, the IUT sends a correctly formatted ATT_Execute_Write_Request (0x18).

The size of each ATT_Prepare_Write_Request does not exceed any negotiated ATT_MTU.

### 4.6.40 GATT/CL/GAW/BI-32-C [Cancel Reliable Write Characteristic – by Client]

- **Test Purpose**
  Verify that a Generic Attribute Profile client will cancel a reliable write after receiving an invalid ATT_Prepare_Write_Request from the server.

- **Reference**
  [1] 4.9.5
  [5] 3.4.6

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  
  A handle of a Characteristic Value in the Lower Tester that permits writing is selected.
  
  If the characteristic descriptor permissions require a specific security mode and level, execute pairing and/or bonding with the required security mode and level.

- **Test Procedure**
  A command is sent from the Upper Tester to the IUT to request the Lower Tester to prepare to write a value to that characteristic e.g., GATT_PrepareWriteRequest (handle, offset, new value).
  
  The Lower Tester responds with an ATT_Prepare_Write_Response (0x17) message where the new value parameter is corrupted i.e. not equal to the value parameter sent in the corresponding ATT_Prepare_Write_Request.
  
  The IUT should send an ATT_Execute_Write_Request (0x18) with the flag parameter set to 0x00 (cancel all prepared writes).
  
  After receiving the ATT_Execute_Write_Response (0x19) from the Lower Tester, the IUT should inform the Upper Tester that the requested write operation was cancelled.
• Expected Outcome

**Pass Verdict**

As a result of an Upper Tester command the IUT sends a correctly formatted ATT_Prepare_Write_Request (0x16) to the Lower Tester. The command specifies the handle of the characteristic to be written, the offset (0x0000) and the value of the characteristic to be written.

After the IUT receives the ATT_Prepare_Write_Response with an incorrect value parameter, it sends an ATT_Execute_Write_Request (0x18) with the flag parameter = 0x00, to cancel the write operation.

The IUT receives the ATT_Execute_Write_Response from the Lower Tester and sends an acknowledgement to the Upper Tester, e.g., GATT_ReliableWriteResponse with a result indicating that the write was not completed.

### 4.6.41 GATT/CL/GAW/BI-33-C [Write Characteristic Value – Invalid Attribute Value Length]

• **Test Purpose**

Verify Generic Attribute Profile client behavior when the Write Characteristic Value procedure fails due to invalid attribute value length.

• **Reference**

[1] 4.9.3

[5] 3.4.5.1, 3.4.5.2

• **Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
A handle of a Characteristic Value in the Lower Tester that permits writing is selected. The size of the characteristic is known; the IUT will deliberately send a value that is longer or shorter than the size specified for that characteristic.

- **Test Procedure**
  Send a command from the Upper Tester to request the IUT to write the value of a characteristic in the Lower Tester, with the size set incorrectly, e.g., GATT_WriteRequest (handle, value).

  ![Diagram](image)

  • **Expected Outcome**
  **Pass Verdict**

  The IUT sends a correctly formatted ATT_Write_Request to the Lower Tester with a Characteristic Value of the wrong size.

  The Characteristic handle parameter is set to a valid handle.

  Upon receiving an ATT_Error_Response from the Lower Tester, the IUT sends the result to the Upper Tester, e.g., GATT_WriteResponse.

  **4.6.42 GATT/CL/GAW/BI-34-C [Write Long Characteristic Value – Invalid Attribute Value Length]**

  • **Test Purpose**

  Verify Generic Attribute Profile client behavior when the Write Long Characteristic Value procedure fails due to invalid attribute value length.

  • **Reference**

  [1] 4.9.4

  [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

MTU shall be smaller than 512, and smaller than the value that has to be written.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The Lower Tester contains at least one valid long characteristic and the handle of this long characteristic is known to the IUT. The size of the characteristic is known; the Lower Tester will deliberately send a value that is longer than the size specified for that characteristic.

• Test Procedure

Send a command from the Upper Tester to request the IUT to write the value of a long characteristic in the Lower Tester e.g., GATT_WriteLongRequest.

The IUT sends correctly formatted ATT_Prepare_Write_Request commands (0x16) to the Lower Tester.

Each ATT_Prepare_Write_Request specifies the handle of the long characteristic to be written, a valid offset of the first octet to be written, and part of the value of the long characteristic to be written. The offset parameters are all values of Nx(ATT_MTU-5), with N ranging from 0 (for the first part value) to the last value sufficient to write the entire long characteristic.

After sending all the ATT_Prepare_Write_Requests, the IUT sends a correctly formatted ATT_Execute_Write_Request.

• Expected Outcome

Pass Verdict

The IUT sends correctly formatted ATT_Prepare_Write_Request commands (0x16) to the Lower Tester.

Each ATT_Prepare_Write_Request specifies the handle of the long characteristic to be written, a valid offset of the first octet to be written, and part of the value of the long characteristic to be written. The offset parameters are all values of Nx(ATT_MTU-5), with N ranging from 0 (for the first part value) to the last value sufficient to write the entire long characteristic.
Note: the IUT detects size problems on execution of the Execute Write Request command.

The size of each ATT_Prepare_Write_Request does not exceed any negotiated ATT_MTU.

4.6.43 GATT/SR/GAW/BV-09-C [Write Long Characteristic Descriptors – from Server]

• Test Purpose

Verify that a Generic Attribute Profile server can support writing a long characteristic descriptor selected by handle.

• Reference

[1] 4.12.4

[5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains a valid characteristic descriptor and the handle of this characteristic descriptor is known to the test system.

The Lower Tester has the necessary security permissions to write the value of the characteristic descriptor of the IUT.

• Test Procedure

Send ATT_Prepare_Write_Requests from the Lower Tester to the IUT to write all parts of the value of a long characteristic descriptor, setting the offset to Nx(ATT_MTU-5), with N starting from 0. The handle of the characteristic to be written and the part value to be written is also specified.

The Lower Tester recovers the returned offset and part value of each ATT_Prepare_Write_Response.

After receiving the last ATT_Prepare_Write_Response the Lower Tester sends an ATT_Execute_Write_Request to the IUT.
• Expected Outcome

Pass Verdict

The IUT sends a series of correctly formatted ATT_Prepare_Write_Responses to the Lower Tester. The offset and part value sent in each response match the offset and part value sent in the corresponding request.

The IUT sends a correctly formatted ATT_Execute_Write_Response to the Lower Tester.

The responses do not exceed any negotiated ATT_MTU.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

4.6.44 GATT/SR/GAW/BI-32-C [Write Characteristic Value – Invalid Attribute Value Length]

• Test Purpose

Verify that a Generic Attribute Profile server can detect a Write Characteristic Request with an invalid characteristic/descriptor value length and issue an Invalid Attribute Value Length response.

• Reference

[1] 4.9.3
[5] 3.4.5.1, 3.4.5.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A handle of a Characteristic Value in the IUT that permits writing is selected. The size of the characteristic is known; the Lower Tester will deliberately send a value that is longer than the size specified for that characteristic.

The Lower Tester has the necessary security permissions from the IUT to read and write the characteristic.

• Test Procedure
  Send an ATT_Write_Request (handle, value) from the Lower Tester to the IUT with the size deliberately set incorrectly.

  \[\text{ATT}_\text{Write}\_\text{Request} \rightarrow \begin{array}{c}
  \text{IUT} \\
  \text{Lower Tester}
\end{array} \]

  \[\begin{array}{c}
  \text{ATT}_\text{Error}\_\text{Response} \\
  \text{Upper Tester}
\end{array} \rightarrow \text{IUT} \]

  ATT_Write_Request
  (Code = 0x12, handle, value -wrong size-)

  ATT_Error_Response
  (Code = 0x01, 0x12, Handle, Error Code = 0x0D)

• Expected Outcome
  Pass Verdict

  The IUT sends a correctly formatted ATT_Write_Response to the Lower Tester. The Request Opcode in Error parameter is set to 0x12. The Attribute Handle in Error parameter is set to the handle sent by the Lower Tester. The Error code is set to 0x0D, Invalid Attribute Value Length.

  The IUT responds within the applicable timeout (3.3.3/ATT [5]).

4.6.45 GATT/SR/GAW/BI-33-C [Write Long Characteristic Value – Invalid Attribute Value Length]

• Test Purpose
  Verify that a Generic Attribute Profile server can detect a Write Long Characteristic Request with an invalid characteristic/descriptor value length and issue an Invalid Attribute Value Length response.

• Reference
  [1] 4.9.4

  [5] 3.4.6.1, 3.4.6.2, 3.4.6.3, 3.4.6.4

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The IUT contains a valid characteristic and the handle of this characteristic is known to the test system. The size of the characteristic is known; the Lower Tester will deliberately send a value that is longer than the size specified for that characteristic.

The Lower Tester has the necessary security permissions to write the value of the characteristic of the IUT.

- Test Procedure

Send ATT_Prepare_Write_Requests from the Lower Tester to the IUT to write all parts of the value of a long characteristic, setting a valid offset to \( N \times (ATT_{MTU} - 5) \), with \( N \) starting from 0. The handle of the characteristic to be written and the part value to be written is also specified. The complete aggregate characteristic will be longer than that supported by the IUT.

The Lower Tester recovers the returned offset and part value of each ATT_PREPARE_WRITE_RESPONSE.

After receiving the last ATT_PREPARE_WRITE_RESPONSE the Lower Tester sends an ATT_EXECUTE_WRITE_REQUEST to the IUT.

- Expected Outcome

Pass Verdict

The IUT sends a series of correctly formatted ATT_PREPARE_WRITEResponses to the Lower Tester. The offset and part value sent in each response match the offset and part value sent in the corresponding request.
The IUT sends a correctly formatted ATT_Execute Write_Response to the Lower Tester. The IUT should respond with an ATT_Error_Response flagging the Invalid Attribute Value Length.

The responses do not exceed any negotiated ATT_MTU.

All of the IUT responses occur within the applicable timeout (3.3.3/ATT [5]).

4.7 Notification and Indication

The test group objective is to verify Generic Attribute Profile Notification, Indication and Confirmation of Characteristic Values.

4.7.1 GATT/CL/GAN/BV-01-C [Characteristic Value Notification - to Client]

- Test Purpose
  Verify that a Generic Attribute Profile client can receive a Characteristic Value Notification and report that to the Upper Tester.

- Reference
  [1] 4.10.1, 4.14
  [5] 3.3.3, 3.4.7.1

- Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A preamble procedure defined in Section 4.2.2.3 is used to configure the characteristic for Notification.

- Test Procedure
  Send a Handle Value Notification from the Lower Tester to the IUT; e.g., GATT_HandleValueNotification, with a handle value to specify the characteristic.
Expected Outcome

Pass Verdict

IUT receives an ATT_Handle_Value_Notification (0x1B) command from the Lower Tester, and generates a message to the Upper Tester containing the characteristic handle and value included in that Handle Value Notification, before the Attribute Protocol Transaction timeout expires.

4.7.2 GATT/SR/GAN/BV-01-C [Characteristic Value Notification - by Server]

Test Purpose

Verify that a Generic Attribute Profile server can send a Characteristic Value Notification.

Reference

[1] 4.10.1
[5] 3.4.7.1

Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

A preamble procedure defined in Section 4.2.2.1 is used to configure the characteristic for Notification.

Test Procedure

Send a command from the Upper Tester to the IUT to request the IUT to send a Handle Value Notification to the Lower Tester e.g., GATT_HandleValueNotification, with a handle value to specify the characteristic.
**Expected Outcome**

**Pass Verdict**

IUT sends a correctly formatted ATT_Handle_Value_Notification (0x1B) command to the Lower Tester.

The Characteristic Handle is set to a valid handle.

The Characteristic Value is set to the value of a characteristic contained in the IUT identified by the characteristic handle.

The command size does not exceed any negotiated ATT_MTU.

The IUT is successfully configured for notification using the procedure specified in Section 4.2.2.1.

### 4.7.3 GATT/SR/GAI/BV-01-C [Characteristic Value Indication - by Server]

**Test Purpose**

Verify that a Generic Attribute Profile server can send a Characteristic Value Indication.

**Reference**

[1] 4.11.1

[5] 3.4.7.2

**Initial Condition**

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT_MTU between the IUT and the Lower Tester.

A preamble procedure defined in Section 4.2.2.2 is used to configure the characteristic for Indication.
• Test Procedure
Send a command from the Upper Tester to the IUT to request the IUT to send a Handle Value Indication to the Lower Tester e.g., `GATT_HandleValueIndication`.

- **Expected Outcome**
  
  **Pass Verdict**
  IUT sends a correctly formatted `ATT_Handle_Value_Indication` (0x1D) command to the Lower Tester. The Characteristic Handle shall be set to a valid handle. The Characteristic Value shall be set to the value of characteristic identified by the characteristic handle.
  IUT receives `ATT_Handle_Value_Confirmation` (0x1E) from the Lower Tester. IUT sends `GATT_HandleValueConfirmation` to the Upper Tester. The command size does not exceed any negotiated `ATT_MTU`.
  The IUT is successfully configured for indication using the procedure specified in Section 4.2.2.2.

4.7.4 **GATT/CL/GAI/BV-01-C [Confirm Characteristic Value Indication - by Client]**

- **Test Purpose**
Verify that a Generic Attribute Profile client can respond with a Confirmation to a Characteristic Value Indication.

- **Reference**
[1] 4.11.1

[5] 3.4.7.3
• Initial Condition
    A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
    A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
    Send an ATT_Handle_Value_Indication command from the Lower Tester to the IUT.

    ![Diagram showing L2CAP Connection Established over selected transport (4.2). ATT_MTU may have been exchanged between IUT and Lower Tester (4.2.4).]

• Expected Outcome
    Pass Verdict
    IUT sends an ATT_Handle_Value_Confirmation (0x1E) command to the Lower Tester.
    The IUT is successfully configured for indication using the procedure specified in Section 4.2.2.4.

4.8 Generic Attribute Profile Services
Verify the correct implementation of the Generic Attribute Profile Services. All of these test cases apply to Generic Attribute Profile servers.

4.8.1 GATT/CL/GAS/BV-01-C [Service Changed Characteristic – to Client]
• Test Purpose
    Verify that the IUT client behavior when a Service Changed Indication is received from a server.

• References
    [1] 2.5.2, 7.1
    [5] 3.4.5.3, 3.4.7.2, 3.4.7.3
• Initial Conditions

IUT has performed service discovery during a previous connection and has cached handles. The cached handles are identified in the IXIT [10].

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure

The Lower Tester sends an ATT_Handle_Value_Indication to the IUT containing the Service Changed characteristic. The Characteristic Value contains a handle range that includes the handles cached by the IUT.

The IUT sends an ATT_Handle_Value_Confirmation.

• Expected Outcome

Pass Verdict

IUT sends an ATT_Handle_Value_Confirmation (0x1E) command to the Lower Tester.

4.8.2 GATT/CL/GAS/BV-02-C [Reading the Database Hash Characteristic]

• Test Purpose

Verify that the IUT acting as GATT Client can read the value of the Database Hash characteristic.

• References

[11] 2.5.2, 7.3

• Initial Conditions

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.
ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.

- **Test Procedure**

  The Upper Tester orders the IUT to read the Database Hash characteristic. The Lower Tester expects the IUT to perform the GATT Read Using Characteristic UUID sub-procedure.

  ![Connection established between IUT and Lower Tester]

  - **Expected Outcome**

    **Pass Verdict**

    The IUT executes the GATT Read Using Characteristic UUID sub-procedure, with the characteristic UUID set to «Database Hash», and the handle range 0x0001-0xFFFF.

4.8.3 **GATT/CL/GAS/BV-03-C [Enabling the Robust Caching]**

- **Test Purpose**

  Verify that the IUT acting as GATT Client can enable the Robust Caching feature on the Server.

- **References**

  [11] 2.5.2.1, 7.2

- **Initial Conditions**

  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.

  ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.

  The IUT has discovered the Client Supported Features characteristic and has saved the handle of the characteristic value.

  There is no bond between the IUT and the Lower Tester.
• Test Procedure

The Upper Tester orders the IUT to enable the Robust Caching feature. The Lower Tester expects the IUT to perform a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1 and all RFU bits to 0.

• Expected Outcome

Pass Verdict

The IUT executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1, and setting all RFU bits to 0.

4.8.4 GATT/SR/GAS/BV-01-C [Service Changed Characteristic – from Server]

• Test Purpose

Verify that if the IUT supports the Service Changed Characteristic and that it can generate an indication.

• References

[1] 2.5.2, 7.1

[5] 3.4.5.2, 3.4.7.2, 3.4.7.3

• Initial Conditions

The Lower Tester has bonded with the IUT during a previous connection and both the Lower Tester and IUT have valid stored bonding information.

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
A preamble procedure defined in Section 4.2.2.2 is used to configure the characteristic for Indication.

- **Test Procedure**
  
  Send a command from the Upper Tester to the IUT to initiate a service change. This may be performed before or after the connection is established.

  The IUT sends an ATT_Handle_Value_Indication to the Lower Tester containing the Service Changed characteristic.

  The Lower Tester sends an ATT_Handle_Value_Confirmation.

- **Expected Outcome**
  
  **Pass Verdict**

  The IUT sends a properly formatted ATT_Handle_Value_Indication containing the Service Changed characteristic.

  The IUT is successfully configured for indication using the procedure specified in Section 4.2.2.2.

### 4.8.5 GATT/SR/GAS/BV-02-C [Computing and Returning the Database Hash Characteristic]

- **Test Purpose**
  
  Verify that the IUT acting as GATT Server can return and properly update the value of the Database Hash characteristic.

- **References**
  
  [11] 2.5.2, 7.3

- **Initial Conditions**

  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.
ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.

- **Test Procedure**
  1. The Lower Tester obtains the complete GATT database from the IUT, by executing service and characteristic discovery procedures, then reads the value of the **Database Hash** characteristic.
  2. The Lower Tester computes the correct value of the database hash and verifies that it matches the value of the **Database Hash** characteristic obtained in step 1. If the values do not match, the test ends with a **Fail Verdict**.
  3. The Upper Tester orders the IUT to change its GATT database, by adding or removing services and/or characteristics.
  4. Repeat steps 1–2.

**Expected Outcome**

- **Pass Verdict**

  The IUT returns the correct value of the **Database Hash** characteristic in both iterations of step 2.
4.8.6 GATT/SR/GAS/BV-03-C [Maintaining a Client Supported Features Characteristic Instance for each Client]

- Test Purpose
  Verify that the IUT acting as GATT Server can update the value of the Client Supported Features characteristic and, if applicable, can maintain a separate instance for each Client.

- References
  [11] 2.5.2, 7.2

- Initial Conditions
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and Lower Tester 1.

  ATT_MTU may have been exchanged between IUT and Lower Tester 1 using the GATT Exchange MTU sub-procedure.

  Lower Tester 1 has discovered the Client Supported Features characteristic and has saved the handle of the characteristic value.

  There is no bond between the IUT and any of the Lower Testers.

- Test Procedure
  1. Lower Tester 1 executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1, setting all RFU bits to 1 and setting all other bits to 0.
  2. Lower Tester 1 executes a GATT Characteristic Value Read procedure on the Client Supported Features characteristic, expecting the Robust Caching bit to be set to 1 and all other bits to 0.
  3. If the IUT does not support multiple simultaneous LE connections, end the test, otherwise continue with step 4.
  4. Lower Tester 2 establishes a connection to the IUT and executes a GATT Characteristic Value Read procedure on the Client Supported Features characteristic, expecting the Robust Caching bit to be set to 0.
Connection established between IUT and Lower Tester 1. The Client Supported Features characteristic on the IUT has been discovered.

- Expected Outcome
  
  **Pass Verdict**

  The IUT updates the value of the *Client Supported Features* characteristic as requested by the Client, ignoring the values of the RFU bits.

  If the IUT supports multiple simultaneous LE connections, it maintains separate characteristic instances for each Client.

4.8.7 GATT/SR/GAS/BV-04-C [Maintaining Client Supported Features Characteristic Values for Bonded Devices]

- Test Purpose
  
  Verify that the IUT acting as GATT Server saves and restore the value of the *Client Supported Features* characteristic for bonded devices across connections.

- References
  
  [11] 2.5.2, 7.2
• Initial Conditions
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.

ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.

The IUT and the Lower Tester have paired and bonded.

The Lower Tester has discovered the Client Supported Features characteristic and has saved the handle of the characteristic value.

• Test Procedure
1. The Lower Tester executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1.
2. The Lower Tester terminates the connection with the IUT, then a new connection is established.
3. The Lower Tester executes a GATT Characteristic Value Read procedure on the Client Supported Features characteristic, expecting the Robust Caching bit to be set to 1.

• Expected Outcome
Pass Verdict

The IUT maintains the value of the Client Supported Features characteristic across connections.
4.8.8 GATT/SR/GAS/BV-05-C [Handling Client Requests on Unsynchronized Database]

- **Test Purpose**
  Verify that the IUT acting as GATT Server handles Client requests on an unsynchronized database based on the Robust Caching feature being enabled or disabled.

- **References**
  [11] 2.5.2, 7.2

- **Initial Conditions**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.
  
  ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.
  
  The GATT database on the IUT contains at least one characteristic that supports the GATT Characteristic Value Read sub-procedure and at least one characteristic that supports the GATT Write Without Response sub-procedure. Any change on the GATT database during the test procedure will not affect these characteristics.
  
  The Lower Tester has discovered the characteristics on the IUT and has saved the handles of their values.
  
  The Lower Tester does not enable indications on the Service Changed characteristic.
  
  The Lower Tester has not enabled the Robust Caching feature.
  
  There is no bond between the IUT and the Lower Tester.

- **Test Procedure**
  1. The Upper Tester orders the IUT to change its GATT database, by adding or removing services and/or characteristics.
  2. The Lower Tester executes the GATT Characteristic Value Read sub-procedure on a characteristic (other than the Database Hash characteristic) and expects the IUT to return either the characteristic value or a different error code. If the Database Out of Sync error is returned, the test ends with a Fail Verdict.
  3. The Lower Tester executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1.
  4. Repeat step 1.
  5. The Lower Tester executes the GATT Characteristic Value Read sub-procedure on a characteristic (other than the Database Hash characteristic) and expects the IUT to return the Database Out of Sync error.
  6. The Lower Tester tries to read the same characteristic as in the previous step, and this time expects the IUT to return either the characteristic value or a different error code. If the Database Out of Sync error is returned, the test ends with a Fail Verdict.
  7. The Upper Tester reads the value of a characteristic on the IUT that supports the GATT Write Without Response sub-procedure.
  8. Repeat step 1, then repeat step 5.
9. The Lower Tester executes a GATT Write Without Response sub-procedure on the characteristic whose value is known from step 7, trying to change the characteristic to a new, valid value. The IUT is expected to ignore the command.

10. Repeat step 7, expecting the IUT to return the same characteristic value. If the IUT has changed the value of the characteristic, then the test ends with a Fail Verdict.

11. The Lower Tester executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 0, and expects the IUT to return the Value Not Allowed error.

12. The Lower Tester executes a GATT Characteristic Value Read procedure on the Client Supported Features characteristic, expecting the Robust Caching bit to be set to 1.

13. Repeat step 1.

14. The Lower Tester executes the GATT Characteristic Value Read sub-procedure on the Database Hash characteristic and expects the IUT to return the Database Out of Sync error.

Connection established between IUT and Lower Tester.
Lower Tester has discovered the database on the IUT.
IUT and Lower Tester have not bonded.
Lower Tester has not enabled the Robust Caching feature.
Lower Tester has not enabled Service Changed indications.

Change the database

ATT_Read_Request
(Handle != Database Hash characteristic)

ATT_Read/Error_Response
(Value or Error != Database Out of Sync)

ATT_Write_Request
(Handle of Client Supported Features,
Value = 0x01)

ATT_Write_Response

Change the database

ATT_Read_Request
(Handle != Database Hash characteristic)

ATT_Error_Response
(Database Out of Sync)

ATT_Read_Request
(Handle != Database Hash characteristic)

ATT_Read/Error_Response
(Value or Error != Database Out of Sync)

Get value of writable characteristic

Value of requested characteristic

Change the database
• **Expected Outcome**

**Pass Verdict**

In both executions of step 5, the IUT returns the Database Out of Sync error.

In steps 2 and 6, the IUT does not return the Database Out of Sync error.

In step 9, the IUT ignores the ATT command from the Lower Tester.

In step 11, the IUT returns the Value Not Allowed error and does not change the value of the characteristic.

In step 14, the IUT returns the Database Out of Sync error.
4.8.9 GATT/SR/GAS/BV-06-C [Handling Client Requests after a Hash Read or Service Changed Indication – Database Changed during Connection]

- **Test Purpose**
  Verify that the IUT acting as GATT Server does not return the Database Out of Sync error after a hash read when the Robust Caching feature is enabled. The database is changed during the same connection.

- **References**
  [11] 2.5.2, 7.2

- **Initial Conditions**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel between the IUT and the Lower Tester.

  ATT_MTU may have been exchanged between the IUT and the Lower Tester using the GATT Exchange MTU sub-procedure.

  The GATT database on the IUT contains at least one characteristic that supports the GATT Characteristic Value Read sub-procedure. Any change on the GATT database during the test procedure will not affect this characteristic.

  The Lower Tester has discovered the characteristics on the IUT and has saved the handles of their values.

  The Lower Tester does not enable indications on the Service Changed characteristic unless specifically directed to do so.

  There is no bond between the IUT and the Lower Tester.

- **Test Procedure**
  1. The Lower Tester executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 1.
  2. The Upper Tester orders the IUT to change its GATT database, by adding or removing services and/or characteristics.
  3. The Lower Tester executes the GATT Read Using Characteristic UUID sub-procedure on the Database Hash characteristic.
  4. The Lower Tester executes the GATT Characteristic Value Read sub-procedure on another characteristic (different than the Database Hash characteristic) and expects the IUT to return either the characteristic value or a different error code. If the Database Out of Sync error is returned, the test ends with a Fail Verdict.
  5. The Lower Tester enables indications on the Service Changed characteristic.
  6. Repeat step 2 and wait for the IUT to send an indication on the Service Changed characteristic. If the indication is not received within 30 seconds, the test ends with a Fail Verdict.
  7. The Lower Tester confirms the indication, then repeat step 4.
  9. The Lower Tester does not confirm the indication, but instead executes the GATT Characteristic Value Read sub-procedure on another characteristic (different than the Database Hash characteristic) and expects the IUT to return the Database Out of Sync error.
Connection established between IUT and Lower Tester. Lower Tester has discovered the database on the IUT. IUT and Lower Tester have not bonded. Lower Tester has not enabled Service Changed indications.

ATT_Write_Request
(Handle of Client Supported Features, Value = 0x01)
ATT_Write_Response

ATT_Read_By_Type_Request
(<<Database Hash>>, 0x0000-0xFFFF)
ATT_Read_By_Type_Response
(Handle and Value of Database Hash)
ATT_Read_Request
(Handle != Database Hash characteristic)
ATT_Read/Error_Response
(Value of Error != Database Out of Sync)

Enable Service Changed indications

ATT_Handle_Value_Indication
(Handle of Service Changed)
ATT_Handle_Value_Confirmation
ATT_Read_Request
(Handle != Database Hash characteristic)
ATT_Read/Error_Response
(Value or Error != Database Out of Sync)

ATT_Handle_Value_Indication
(Handle of Service Changed)
ATT_Read_Request
(Handle != Database Hash characteristic)
ATT_Error_Response
(Database Out of Sync)
• Expected Outcome

Pass Verdict

In both executions of step 4, the IUT does not return the Database Out of Sync error.

In step 9, the IUT returns the Database Out of Sync error.

4.8.10 GATT/SR/GAS/BV-07-C [Handling Client Requests after a Hash Read or Service Changed Indication – Database Changed between Connections]

• Test Purpose

Verify that the IUT acting as GATT Server does not return the Database Out of Sync error after a hash read when the Robust Caching feature is enabled. The database is changed between connections after a bond has been created.

• References

[11] 2.5.2, 7.2

• Initial Conditions

The GATT database on the IUT contains at least one characteristic that supports the GATT Characteristic Value Read sub-procedure. Any change on the GATT database during the test procedure will not affect this characteristic.

The IUT and the Lower Tester have bonded and the connection has been terminated.

The Lower Tester has discovered the characteristics on the IUT and has saved the handles of their values.

The Lower Tester has enabled the Robust Caching feature.

The Lower Tester has not enabled indications on the Service Changed characteristic.

• Test Procedure

1. The Upper Tester orders the IUT to change its GATT database, by adding or removing services and/or characteristics, then the Lower Tester creates a connection with the IUT.
2. The Lower Tester executes the GATT Read Using Characteristic UUID sub-procedure on the Database Hash characteristic.
3. The Lower Tester executes the GATT Characteristic Value Read sub-procedure on another characteristic (different than the Database Hash characteristic) and expects the IUT to return either the characteristic value or a different error code. If the Database Out of Sync error is returned, the test ends with a Fail Verdict.
4. The Lower Tester enables indications on the Service Changed characteristic, then it terminates the connection.
5. Repeat step 1 and wait for the IUT to send an indication on the Service Changed characteristic. If the indication is not received within 30 seconds, the test ends with a Fail Verdict.
6. The Lower Tester confirms the indication, then repeat step 3.
7. The Lower Tester terminates the connection, then reconnects, and then it executes a GATT Characteristic Value Write procedure on the Client Supported Features characteristic, setting the Robust Caching bit to 0, and expects the IUT to return the Value Not Allowed error.
8. The Lower Tester disables indications on the Service Changed characteristic, then it terminates the connection, then repeat step 1, and then the Lower Tester executes the GATT Characteristic Value Read sub-procedure on another characteristic (different than the Database Hash characteristic) and expects the IUT to return the Database Out of Sync error.

9. The Lower Tester terminates the connection, then reconnects, and then it executes the GATT Characteristic Value Read sub-procedure on another characteristic (different than the Database Hash characteristic) and expects the IUT to return the Database Out of Sync error.

---

Connect

- ATT_Read_By_Type_Request
  - <<Database Hash>>, 0x0001-0xFFFF
- ATT_Read_By_Type_Response
  - (Handle and Value of Database Hash)
- ATT_Read_Request
  - (Handle != Database Hash characteristic)
- ATT_Read/Error_Response
  - (Value or Error != Database Out of Sync)

Enable Service Changed indications, then disconnect

- ATT_Handle_Value_Indication
  - (Handle of Service Changed)
- ATT_Handle_Value_Confirmation

Reconnect

- ATT_Read_Request
  - (Handle != Database Hash characteristic)
- ATT_Read/Error_Response
  - (Value or Error != Database Out of Sync)

Disconnect and reconnect
• Expected Outcome

Pass Verdict

In both executions of step 3, the IUT does not return the Database Out of Sync error.

In step 7, the IUT returns the Value Not Allowed error.

In steps 8 and 9, the IUT returns the Database Out Of Sync error.

4.9 GATT Transaction Timeouts

Test cases in this group check IUT response to GATT transaction timeouts.

4.9.1 GATT/CL/GAT/BV-01-C [Read Characteristic Value – Server Timeout]

• Test Purpose

Verify that a client IUT detects server timeout after a read request, notifies the Upper Tester, and does not retry the read request.
• Reference

[1] 4.8.1, 4.14

[5] 3.3.3, 3.4.4.3

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

• Test Procedure

The Upper Tester will specify the handle of a Characteristic Value contained in the Lower Tester.

Send a request from the Upper Tester to the IUT to read a Characteristic Value from the Lower Tester by specifying the characteristic handle e.g., GATT_ReadReq.

The Lower Tester detects the ATT_Read_Request and notifies the Upper Tester.

• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Read_Request command (0x0A) to the Lower Tester.

The characteristic handle parameter is set to the valid handle specified by the Upper Tester and waits for the ATT_Read_Response.

After the specified timeout (3.3.3/ATT [5]) the IUT sends a notification to the Upper Tester. A test system may allow up to a +/- 1 second deviation of the 30 second timeout to accommodate timer synchronization issues caused by the testing environment.
4.9.2 GATT/CL/GAT/BV-02-C [Write Characteristic Value – Server Timeout]

• Test Purpose
Verify that a client IUT detects server timeout after a Write Request, notifies the Upper Tester, and does not retry the Write Request.

• Reference
[1] 4.9.3, 4.14
[5] 3.3.3, 3.4.5.2

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel. The Lower Tester contains at least one valid characteristic and the handle of this characteristic is known to the Upper Tester. The Lower Tester has the necessary security permissions from the IUT to write a characteristic.

• Test Procedure
Send a command from the Upper Tester to request IUT to write the value of a characteristic in the Lower Tester e.g., GATT_WriteRequest (handle, value).

The Lower Tester detects the ATT_Write_Request and notifies the Upper Tester.
• Expected Outcome

Pass Verdict

The IUT sends a correctly formatted ATT_Write_Request command (0x12) to the Lower Tester, specifying the value that is to be written from the Upper Tester.

The characteristic handle parameter is set to a valid handle.

After the specified timeout (3.3.3/ATT [5]) the IUT sends a notification to the Upper Tester. A test system may allow up to a +/- 1 second deviation of the 30 second timeout to accommodate timer synchronization issues caused by the testing environment.

### 4.9.3 GATT/SR/GAT/BV-01-C [Handle Value Indication – Client Timeout]

• Test Purpose

Verify that a server IUT detects client timeout after a Handle Value Indication, notifies the Upper Tester, and does not retry the Handle Value Indication.

• Reference

[1] 4.11.1, 4.14

[5] 3.3.3, 3.4.7.2

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.2.2 is used to configure the characteristic for Indication.

• Test Procedure

Send a command from the Upper Tester to the IUT to request the IUT to send a Handle Value Indication to the Lower Tester e.g., GATT_HandleValueIndication.

The Lower Tester detects the ATT_Handle_Value_Indication and notifies the Upper Tester.
• Expected Outcome

Pass Verdict

IUT sends a correctly formatted ATT_Handle_Value_Indication (0x1D) command to the Lower Tester.

The characteristic handle shall be set to a valid handle.

The Characteristic Value shall be set to the value of characteristic identified by the characteristic handle.

After the specified timeout (3.3.3/ATT [5]) the IUT sends a notification to the Upper Tester. A test system may allow up to a +/- 1 second deviation of the 30 second timeout to accommodate timer synchronization issues caused by the testing environment.

4.10 Generic Profile Attributes

The test group objective is to verify format and integrity of generic attributes when read by a client or when provided by a server.

4.10.1 GATT/CL/GPA/BV-12-C [Characteristic Format Descriptors – from Client]

• Test Purpose

Verify that a Generic Attribute Profile client (IUT) can read and interpret a Characteristic Value whose format and unit is defined by the Characteristic Presentation Format. The format, unit and value are not known to the IUT before the test.

• Reference

[1] 3.3, 4.7.1, 4.8

[5] 3.3.1, 3.4.4.3
• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The handle range of the characteristic used for the test is communicated to the IUT.

• Test Procedure

Execute the following procedure 5 times with different, random characteristic, formats and exponents—where applicable—presented by the Lower Tester.

1. Send a command from the Upper Tester to the IUT to request the IUT to discover the characteristic properties, read the Characteristic Value and format and present it with correct formatting and unit to the Upper Tester.

2. Characteristic Value, value length where applicable, exponent and unit are randomly chosen by the Lower Tester among the options in the following Characteristic Format Descriptor table:

<table>
<thead>
<tr>
<th>Characteristic Format Descriptor</th>
<th>Value Range (hex)</th>
<th>Value Length (octets)</th>
<th>Exponent Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>0</td>
<td>0x1</td>
<td>1</td>
</tr>
<tr>
<td>2 bit</td>
<td>0</td>
<td>0x3</td>
<td>1</td>
</tr>
<tr>
<td>nibble</td>
<td>0</td>
<td>0xF</td>
<td>1</td>
</tr>
<tr>
<td>uint8</td>
<td>0</td>
<td>0xFF</td>
<td>1</td>
</tr>
<tr>
<td>uint12</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
<tr>
<td>uint16</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
<tr>
<td>uint24</td>
<td>0</td>
<td>0xFF.FFFF</td>
<td>3</td>
</tr>
<tr>
<td>uint32</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>4</td>
</tr>
<tr>
<td>uint48</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>6</td>
</tr>
<tr>
<td>uint64</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>8</td>
</tr>
<tr>
<td>Characteristic Format Descriptor</td>
<td>Value Range (hex)</td>
<td>Value Length (octets)</td>
<td>Exponent Range (decimal)</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
<td>-----------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>unit128</td>
<td>0</td>
<td>0xFFFF. FFFF. FFFF. FFFF. FFFF. FFFF. FFFF. FFFF. FFFF</td>
<td>16</td>
</tr>
<tr>
<td>sint8</td>
<td>0</td>
<td>0xFF</td>
<td>1</td>
</tr>
<tr>
<td>sint12</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
<tr>
<td>sint16</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>2</td>
</tr>
<tr>
<td>sint24</td>
<td>0</td>
<td>0xFF. FFFF</td>
<td>3</td>
</tr>
<tr>
<td>sint32</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>4</td>
</tr>
<tr>
<td>sint48</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>6</td>
</tr>
<tr>
<td>sint64</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>8</td>
</tr>
<tr>
<td>sint128</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>16</td>
</tr>
<tr>
<td>float32</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>4</td>
</tr>
<tr>
<td>float64</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>8</td>
</tr>
<tr>
<td>SFLOAT</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
</tbody>
</table>
### Table 4.2: Characteristic format descriptor for random choice in test cases - for client IUT

In execution of these tests, the client IUT might use procedures similar to those defined for the following tests:

- **GATT/CL/GAR/BV-06-C** [Read Characteristic Descriptors – by Client]
- **GATT/CL/GAR/BV-01-C** [Read Characteristic Value - by Client]
- **GATT/CL/GAR/BV-04-C** [Read Long Characteristic Value - by Client]

**• Expected Outcome**

**Pass Verdict**

In the case of utf8s, utf16s and struct formats, the number of individual elements (e.g. characters) shall match the number of elements presented by the Lower Tester.

In the case of struct format, the IUT is not required to decompose the contents of the structure.

**• Notes**

The test assumes that the GATT upper edge provides sufficient capabilities to verify the validity of the Characteristic Value whose format and unit is defined by the Characteristic Presentation Format.

The presentation of value, exponent and unit is implementation dependent and may vary from the presentation of e.g. separate value, exponent and unit up to strings containing a human-readable interpretation of value and unit.

As long as this may be verified the capabilities are up to the implementer of the IUT to define, among other alternatives implementers may choose raw format or complete interpreted structure.
Non-exhaustive Examples:

#1 [Characteristic sint16 Format Descriptor, by client]:
The Lower Tester presents the following field values to the IUT:

- Characteristic Value = 0xF025
- Format = 0x0E
- Exponent = 0xFB
- Unit = 0x27AC

The result presented by the IUT to the Upper Tester is:

- -0.04059 °F

#2 [Characteristic sint16 Format Descriptor, by client]:
The Lower Tester presents the following field values to the IUT:

- Characteristic Value = 0xF025
- Format = 0x0E
- Exponent = 0xFB
- Unit = 0x27AC

The result presented by the IUT to the Upper Tester is in the form of four individual elements:

- [value, format, exponent, unit] = [0xF025, 0x0E, 0xFB, 0x27AC]

#3 [Characteristic FLOAT Format Descriptor, by client]:
The Lower Tester presents the following field values to the IUT:

- Characteristic Value = 0x4162.2B4A.BB55.CD44
- Format = 0x15
- Exponent = n/a
- Unit = 0x2782

The result presented by the IUT to the Upper Tester is:

- 9.525845854238547e+6 Ångström

#4 [Characteristic utf16s Format Descriptor, by client]:
The Lower Tester presents the following field values to the IUT:

Characteristic Value = 0x1F08.03C1.03C7.03B9.03BC.03AE.03B4.03B7.03C2

Format = 0x1A

Exponent = n/a

Unit = 0x2700
The result presented by the IUT to the Upper Tester is:

- Ἀρχιμήδης
  ('Archimedes' in Greek letters)

4.10.2 GATT/SR/GPA/BV-12-C [Characteristic Presentation Format Descriptors – from Server]

• Test Purpose
  Verify that a Generic Attribute Profile server (IUT) can report a Characteristic Value whose format and unit is defined by the Characteristic Presentation Format.

• Reference
  [1] 3.3, 3.3.3.5, 4.7, 4.8
  [5] 3.3.1, 3.4.4.2, 3.4.4.4, 3.4.4.6

• Initial Condition
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.
  
  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.
  
  The server IUT is configured with a test database which meets conditions described in Table 4.3; e.g., the database specified in [7]. The Upper Tester and the Lower Tester have access to the same database.

• Test Procedure
  Execute the following procedure with a, random Characteristic Value and format requested by the Lower Tester.

  1. Send an ATT_Find_Information_Request from the Lower Tester to read the characteristic properties, using a handle range that will select the Characteristic Presentation Format Descriptor from the test database, and the type set to «Characteristic Format».
  2. On receipt of the IUT response, send an ATT_Read_Request, and/or a series of ATT_Read_Blob_Requests, from the Lower Tester to the IUT to read the Characteristic Value, or Long Characteristic Value. The Lower Tester will request random combinations of Characteristic Value, exponent and unit within the ranges defined by Table 4.3.
  3. Characteristic Value, exponent and unit are randomly specified in the test database by the Lower Tester according to the following Test Case table:

<table>
<thead>
<tr>
<th>Valid Format</th>
<th>Value Range (hex)</th>
<th>Value Length (octets)</th>
<th>Exponent Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>0</td>
<td>0x1</td>
<td>n/a</td>
</tr>
<tr>
<td>Valid Format</td>
<td>Value Range (hex)</td>
<td>Value Length</td>
<td>Exponent Range</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>--------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>(octets)</td>
</tr>
<tr>
<td>2bit</td>
<td>0</td>
<td>0x3</td>
<td>1</td>
</tr>
<tr>
<td>nibble</td>
<td>0</td>
<td>0xF</td>
<td>1</td>
</tr>
<tr>
<td>uint8</td>
<td>0</td>
<td>0xFF</td>
<td>1</td>
</tr>
<tr>
<td>uint12</td>
<td>0</td>
<td>0xFFF</td>
<td>2</td>
</tr>
<tr>
<td>uint16</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
<tr>
<td>uint24</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>3</td>
</tr>
<tr>
<td>uint32</td>
<td>0</td>
<td>0x0FFFF.FFFF</td>
<td>4</td>
</tr>
<tr>
<td>uint48</td>
<td>0</td>
<td>0x0FFFF.FFFF.</td>
<td>6</td>
</tr>
<tr>
<td>uint64</td>
<td>0</td>
<td>0x0FFFF.FFFF.</td>
<td>8</td>
</tr>
<tr>
<td>unit128</td>
<td>0</td>
<td>0x0FFFF.FFFF.</td>
<td>16</td>
</tr>
<tr>
<td>sint8</td>
<td>0</td>
<td>0xFF</td>
<td>1</td>
</tr>
<tr>
<td>sint12</td>
<td>0</td>
<td>0xFFF</td>
<td>2</td>
</tr>
<tr>
<td>sint16</td>
<td>0</td>
<td>0xFFFF</td>
<td>2</td>
</tr>
<tr>
<td>sint24</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>3</td>
</tr>
<tr>
<td>sint32</td>
<td>0</td>
<td>0xFFFF.FFFF</td>
<td>4</td>
</tr>
<tr>
<td>Valid Format</td>
<td>Value Range (hex)</td>
<td>Value Length (octets)</td>
<td>Exponent Range (decimal)</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>-------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>sint48</td>
<td>0x0000.0000.0000</td>
<td>6</td>
<td>-128 - 127</td>
</tr>
<tr>
<td>sint64</td>
<td>0x0000.0000.0000</td>
<td>8</td>
<td>-128 - 127</td>
</tr>
<tr>
<td>sint128</td>
<td>0x0000.0000.0000</td>
<td>16</td>
<td>-128 - 127</td>
</tr>
<tr>
<td>float32</td>
<td>0x0000.0000.0000</td>
<td>4</td>
<td>n/a</td>
</tr>
<tr>
<td>float64</td>
<td>0x0000.0000.0000</td>
<td>8</td>
<td>n/a</td>
</tr>
<tr>
<td>SFLOAT</td>
<td>0x0000</td>
<td>2</td>
<td>n/a</td>
</tr>
<tr>
<td>FLOAT</td>
<td>0x0000</td>
<td>4</td>
<td>n/a</td>
</tr>
<tr>
<td>duint16</td>
<td>0x0000, 0x0000</td>
<td>4</td>
<td>n/a</td>
</tr>
<tr>
<td>utf8</td>
<td>n/a</td>
<td>variable</td>
<td>n/a</td>
</tr>
<tr>
<td>utf16</td>
<td>n/a</td>
<td>variable</td>
<td>n/a</td>
</tr>
<tr>
<td>struct</td>
<td>n/a</td>
<td>variable</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 4.3: Characteristic Format Descriptor Test Cases - for server IUT
• Expected Outcome

Pass Verdict

In response to the ATT_Find_information_Request the IUT sends a correctly formatted Characteristic Presentation Format Descriptor. The length of any variable length characteristics sent by the IUT has to match the one in the database.

In response to the ATT_Read_Request, or series of ATT_Read_Blob_Requests, the IUT sends a Characteristic Value correctly formatted to match the Characteristic Format contained in the Characteristic Presentation Format Descriptor.

• Notes

The test assumes that the GATT upper edge provides sufficient capabilities to verify the validity of the Characteristic Value whose format and unit is defined by the Characteristic Presentation Format.

The presentation of value, exponent and unit is implementation dependent and may vary from the presentation of e.g. separate value, exponent and unit up to strings containing a human-readable interpretation of value and unit.

As long as this may be verified, the capabilities are up to the implementer of the IUT to define, among other alternatives implementers may choose raw format or complete interpreted structure.

4.10.3 GATT/CL/GPA/BV-11-C [Characteristic Aggregate Format – by Client]

• Test Purpose

Verify that a Generic Attribute Profile client (IUT) can read and interpret a Characteristic Value whose format and unit is defined by several Characteristic Presentation Formats that are concatenated by a Characteristic Aggregate Format descriptor. The formats, units and values are not known to the IUT before the test.

• Reference

[1] 3.3.3.6, 4.7.1, 4.8

[5] 3.3.1, 3.4.4.3

• Initial Condition

A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The handle range of the Characteristic used for the test is communicated to the IUT.

• Test Procedure

Execute the following procedure 5 times, each time with a random number (two or more) of individual values concatenated by a Characteristic Aggregate Format descriptor. The Characteristic format and exponents for each of the values as well as the values themselves are randomly chosen from the following table and presented by the Lower Tester:
<table>
<thead>
<tr>
<th>Value Format</th>
<th>Value Range (hex)</th>
<th>Value Length</th>
<th>Exponent Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>0 0x1</td>
<td>1 bit</td>
<td>n/a</td>
</tr>
<tr>
<td>2bit</td>
<td>0 0x3</td>
<td>2 bit</td>
<td>n/a</td>
</tr>
<tr>
<td>nibble</td>
<td>0 0xF</td>
<td>4 bit</td>
<td>n/a</td>
</tr>
<tr>
<td>uint8</td>
<td>0 0xFF</td>
<td>1 octet</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint12</td>
<td>0 0xFFFF</td>
<td>2 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint16</td>
<td>0 0xFFFFF</td>
<td>2 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint24</td>
<td>0 0xFFF.FFFF</td>
<td>3 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint32</td>
<td>0 0xFFFF.FFFF</td>
<td>4 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint48</td>
<td>0 0xFFFF.FFFF.FFFF</td>
<td>6 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint64</td>
<td>0 0xFFFF.FFFF.FFFF</td>
<td>8 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>uint128</td>
<td>0 0xFFFF.FFFF.FFFF.FFFF</td>
<td>16 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint8</td>
<td>0 0xFF</td>
<td>1 octet</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint12</td>
<td>0 0xFFF</td>
<td>2 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint16</td>
<td>0 0xFFFF</td>
<td>2 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint24</td>
<td>0 0xFFF.FFFF</td>
<td>3 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint32</td>
<td>0 0xFFFF.FFFF</td>
<td>4 octets</td>
<td>-128 127</td>
</tr>
</tbody>
</table>
### Table 4.4: Characteristic Format Selection Table - for client IUT

<table>
<thead>
<tr>
<th>Value Format</th>
<th>Value Range (hex)</th>
<th>Value Length</th>
<th>Exponent Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>sint48</td>
<td>0 0xFFFF. FFFF. FFFF</td>
<td>6 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint64</td>
<td>0 0xFFFF. FFFF. FFFF</td>
<td>8 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>sint128</td>
<td>0 0xFFFF. FFFF. FFFF. FFFF</td>
<td>16 octets</td>
<td>-128 127</td>
</tr>
<tr>
<td>float32</td>
<td>0 0xFFFF. FFFF</td>
<td>4 octets</td>
<td>n/a</td>
</tr>
<tr>
<td>float64</td>
<td>0 0xFFFF. FFFF</td>
<td>8 octets</td>
<td>n/a</td>
</tr>
<tr>
<td>SFLOAT</td>
<td>0 0xFFFF</td>
<td>2 octets</td>
<td>n/a</td>
</tr>
<tr>
<td>FLOAT</td>
<td>0 0xFFFF. FFFF</td>
<td>4 octets</td>
<td>n/a</td>
</tr>
<tr>
<td>duint16</td>
<td>0, 0 0xFFFF, 0xFFFF</td>
<td>4 octets</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Send a command from the Upper Tester to the IUT to request the IUT to discover the characteristic properties, read the Characteristic Value, aggregate format and format descriptors and present it with correct formatting and unit to the Upper Tester.

In execution of these tests, the client IUT might use procedures similar to those defined the following tests:

- GATT/CL/GAR/BV-06-C [Read Characteristic Descriptors – by Client]
- GATT/CL/GAR/BV-01-C [Read Characteristic Value - by Client]
- GATT/CL/GAR/BV-04-C [Read Long Characteristic Value - by Client]

Note: The Lower Tester shall store all Characteristic Value data as specified in 3.3.3.5.2/GATT [1].
• Expected Outcome
Pass Verdict

The IUT presents the correct amount of Characteristic Values with the exponent applied (if defined for that format) correctly formatted according to the Format field with unit to the Upper Tester.

4.10.4 GATT/SR/GPA/BV-11-C [Characteristic Aggregate Format – by Server]

• Test Purpose
Verify that a Generic Attribute Profile server (IUT) can report an aggregate Characteristic Value whose format and unit is defined by concatenating several Characteristic Presentation Formats.

• Reference
[1] 3.3.3.6, 4.7.1, 4.8
[5] 3.3.1, 3.4.4.2, 3.4.4.4, 3.4.4.6

• Initial Condition
A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

The server IUT is configured with a test database, which meets conditions described in Table 4.5, e.g., the database specified in [7]. The Upper Tester and the Lower Tester have access to the same database.

• Test Procedure
Execute the following procedure 5 times, each time with a random number (two or more) of individual values concatenated by a Characteristic Aggregate Format descriptor.

Lower Tester send an ATT_Read_by_Type_Request to the IUT to read the characteristic aggregate format descriptor, using a handle range defined in the test database with the type set to «Characteristic Aggregate Format».

IUT send an ATT_Read_by_Type_Response with characteristic aggregate descriptor to the Lower Tester.

Lower Tester sends an ATT_Read_by_Type_Request, and/or a series of ATT_Read_Blob_Requests to the IUT to read the Characteristic Aggregate Values (including any Long Characteristic Values).

IUT sends an ATT_Read_Response with characteristic aggregate values to the Lower Tester.
The Characteristic format and exponents for each of the aggregated values as well as the values themselves are randomly chosen from the following table and presented by the Lower Tester:

<table>
<thead>
<tr>
<th>Value Format</th>
<th>Value Range (hex)</th>
<th>Value Length</th>
<th>Exponent Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>boolean</td>
<td>0</td>
<td>0x1</td>
<td>1 bit</td>
</tr>
<tr>
<td>2bit</td>
<td>0</td>
<td>0x3</td>
<td>2 bit</td>
</tr>
<tr>
<td>nibble</td>
<td>0</td>
<td>0xF</td>
<td>4 bit</td>
</tr>
<tr>
<td>uint8</td>
<td>0</td>
<td>0xFF</td>
<td>1 octet</td>
</tr>
<tr>
<td>uint12</td>
<td>0</td>
<td>0xFFF</td>
<td>2 octets</td>
</tr>
<tr>
<td>uint16</td>
<td>0</td>
<td>0xFFFF</td>
<td>2 octets</td>
</tr>
<tr>
<td>uint24</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>3 octets</td>
</tr>
<tr>
<td>uint32</td>
<td>0</td>
<td>0xFFFF. FFFF</td>
<td>4 octets</td>
</tr>
<tr>
<td>uint48</td>
<td>0</td>
<td>0xFFFF. FFFF. FFFF</td>
<td>6 octets</td>
</tr>
<tr>
<td>uint64</td>
<td>0</td>
<td>0xFFFF. FFFF. FFFF. FFFF</td>
<td>8 octets</td>
</tr>
<tr>
<td>uint128</td>
<td>0</td>
<td>0xFFFF. FFFF. FFFF. FFFF. FFFF. FFFF. FFFF. FFFF</td>
<td>16 octets</td>
</tr>
<tr>
<td>sint8</td>
<td>0</td>
<td>0xFF</td>
<td>1 octet</td>
</tr>
<tr>
<td>sint12</td>
<td>0</td>
<td>0xFFF</td>
<td>2 octets</td>
</tr>
<tr>
<td>sint16</td>
<td>0</td>
<td>0xFFFF</td>
<td>2 octets</td>
</tr>
<tr>
<td>sint24</td>
<td>0</td>
<td>0xFFF. FFFF</td>
<td>3 octets</td>
</tr>
</tbody>
</table>
Table 4.5: Characteristic Format Selection Table - for server IUT

- Expected Outcome

**Pass Verdict**

In response to the ATT_Read_Request, and/or a series of ATT_Read_Blob_Requests the IUT presents correctly a correctly formatted Aggregated Characteristic Value according to the Format fields in the reported Aggregate Characteristic Descriptor, including applicable exponent fields, with units to the Upper Tester. The IUT shall store all Characteristic value data as specified in 3.3.3.5.2/GATT [1].

- Notes

The test assumes that the GATT value whose format and unit is defined by the Characteristic Presentation Format.
The presentation of value, exponent and unit is implementation dependent and may vary from the presentation of e.g. separate value, exponent and unit up to strings containing a human-readable interpretation of value and unit.

As long as this may be verified, the capabilities are up to the implementer of the IUT to define, among other alternatives implementers may choose raw format or complete interpreted structure.

4.11 Multiple ATT Bearer Support

This test group verifies that the Generic Attribute Profile server properly handles the client configuration characteristic with a single device accessing the Generic Attribute Profile server using multiple Attribute Protocol bearers.

4.11.1 GATT/SR/GPM/BV-01-C [Client Configuration Characteristic per ATT Bearer]

- **Test Purpose**
  Verify that a Generic Attribute Profile server has separate client configuration characteristics for each Attribute Protocol bearer.

- **Reference**
  [5] 3.3.3

- **Initial Condition**
  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel over LE

  A preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel over BR/EDR

  A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

  A preamble procedure defined in Section 4.2.2.1 is used to configure the characteristic for Notification on the ATT bearer over LE

  A preamble procedure defined in Section 4.2.2.1 is used to configure the characteristic for Indications on the ATT bearer over BR/EDR.

  The two bearers are set up to be active simultaneously.

- **Test Procedure**
  Send an ATT_Read_Request from the lower tester to the IUT to read the Client Configuration Characteristic Descriptor value by specifying the Descriptor handle e.g., ATT_Read_Request on the ATT bearer over LE.

  Send an ATT_Read_Request from the lower tester to the IUT to read the Client Configuration Characteristic Descriptor value by specifying the Descriptor handle e.g., ATT_Read_Request on the ATT bearer over BR/EDR.
• Expected Outcome

Pass Verdict

The IUT sends ATT_Read_Response (code 0x0B) to the lower tester for both requests on each transport.

The Descriptor value is set to the value of the Descriptor identified by the attribute handle in the ATT_Read_Request for both requests on each transport.

The IUT responds within the applicable timeout ([5] 3.3.3).

The Client Configuration Characteristic Descriptor value for the value returned on the LE ATT bearer shall only have the notification bit set.

The Client Configuration Characteristic Descriptor value for the value returned on the BR/EDR ATT bearer shall only have the indication bit set.

4.12 Unsupported Requests and Commands

4.12.1 GATT/SR/UNS/BI-01-C [Unsupported ATT Requests on Server]

• Test Purpose

Verify that a Generic Attribute Profile server responds to ATT Requests with an unsupported opcode with a proper ATT Error Response.

• Reference

[5] 3.3

• Initial Condition

The ATT Requests supported by the IUT can be derived from the ICS [3] [8].

The preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure

Lower Tester sends an Attribute PDU with the Attribute Opcode field set to a randomly generated request opcode, with the Command Flag set to 0, that is not in the list of ATT Requests supported by the IUT (if the IUT supports all the request opcodes defined by the specification, then the generated opcode will be an RFU value).

If the value of the Authentication Signature Flag is 1, then the Authentication Signature field of the Attribute PDU is set to a random 12-octet value.

The Attribute Parameters field of the Attribute PDU is set to an array of random octets, of size limited by the ATT_MTU and the authentication signature.
• Expected Outcome
Pass verdict

The IUT responds with an ATT Error Response PDU, with the Request Opcode In Error field set to the opcode sent by the Lower Tester, the Attribute Handle In Error field set to 0x0000, and the Error Code field set to 0x06 (Request Not Supported).

4.12.2 GATT/SR/UNS/BI-02-C [Unsupported ATT Commands on Server]

• Test Purpose
Verify that a Generic Attribute Profile server silently ignores ATT Commands with an unsupported opcode.

• Reference
[5] 3.3

• Initial Condition
The ATT Commands supported by the IUT can be derived from the ICS [3] [8].

The preamble procedure defined in Section 4.2.1 is used to set up the transport and L2CAP channel.

A preamble procedure defined in Section 4.2.4 may have been used to exchange ATT MTU between the IUT and the Lower Tester.

• Test Procedure
Lower Tester sends an Attribute PDU with the Attribute Opcode field set to a randomly generated request opcode, with the Command Flag set to 1, that is not in the list of ATT Commands supported by the IUT (if the IUT supports all the command opcodes defined by the specification, then the generated opcode will be an RFU value).

If the value of the Authentication Signature Flag is 1, then the Authentication Signature field of the Attribute PDU is set to a random 12-octet value.

The Attribute Parameters field of the Attribute PDU is set to an array of random octets, of size limited by the ATT_MTU and the authentication signature.

• Expected Outcome
Pass verdict

The IUT silently ignores the command from the Lower Tester.
# 5 Test Case Mapping

The Test Case Mapping Table (TCMT) maps test cases to specific capabilities in the ICS. Profiles, protocols and services may define multiple roles, and it is possible that a product may implement more than one role. The product shall be tested in all roles for which support is declared in the ICS document.

The columns for the TCMT are defined as follows:

**Item**: contains an y/x reference, where y corresponds to the table number and x corresponds to the feature number as defined in the GATT ICS Proforma [3]. If the item is defined with Protocol, Profile or Service abbreviation before y/x, the table and feature number referenced are defined in the abbreviated ICS proforma document.

**Feature**: recommended to be the primary feature defined in the ICS being tested or may be the test case name.

**Test Case(s)**: the applicable test case identifiers required for Bluetooth Qualification if the corresponding y/x references defined in the Item column are supported.


<table>
<thead>
<tr>
<th>Item</th>
<th>Feature</th>
<th>Test Case(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GATT 1/2</td>
<td>Generic Attribute Profile Server - Unsupported Requests and Commands</td>
<td>GATT/SR/UNS/BI-01-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/UNS/BI-02-C</td>
</tr>
<tr>
<td>GATT 3/1</td>
<td>Exchange MTU</td>
<td>GATT/CL/GAC/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/1</td>
<td>Exchange MTU</td>
<td>GATT/SR/GAC/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/2</td>
<td>Discover All Primary Services</td>
<td>GATT/CL/GAD/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/2</td>
<td>Discover All Primary Services</td>
<td>GATT/SR/GAD/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/3</td>
<td>Discover Services by Service UUID</td>
<td>GATT/CL/GAD/BV-02-C</td>
</tr>
<tr>
<td>GATT 4/3</td>
<td>Discover Services by Service UUID</td>
<td>GATT/SR/GAD/BV-02-C</td>
</tr>
<tr>
<td>GATT 3/4</td>
<td>Find Included Services</td>
<td>GATT/CL/GAD/BV-03-C</td>
</tr>
<tr>
<td>GATT 4/4</td>
<td>Find Included Services</td>
<td>GATT/SR/GAD/BV-03-C</td>
</tr>
<tr>
<td>GATT 3/5</td>
<td>Discover All Characteristics of a Service</td>
<td>GATT/CL/GAD/BV-04-C</td>
</tr>
<tr>
<td>GATT 4/5</td>
<td>Discover All Characteristics of a Service</td>
<td>GATT/SR/GAD/BV-04-C</td>
</tr>
<tr>
<td>GATT 3/6</td>
<td>Discover Characteristics by UUID</td>
<td>GATT/CL/GAD/BV-05-C</td>
</tr>
<tr>
<td>GATT 4/6</td>
<td>Discover Characteristics by UUID</td>
<td>GATT/SR/GAD/BV-05-C</td>
</tr>
<tr>
<td>GATT 3/7</td>
<td>Discover All Characteristic Descriptors</td>
<td>GATT/CL/GAD/BV-06-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td>GATT 4/7</td>
<td>Discover All Characteristic Descriptors</td>
<td>GATT/SR/GAD/BV-06-C</td>
</tr>
<tr>
<td>GATT 6/2 AND GATT 1/1</td>
<td>Discover GATT Services over BR/EDR using SDP</td>
<td>GATT/CL/GAD/BV-07-C, GATT/CL/GAD/BV-08-C</td>
</tr>
<tr>
<td>GATT 6/3 AND GATT 1/2</td>
<td>Publish SDP record for GATT Services over BR/EDR</td>
<td>GATT/SR/GAD/BV-07-C, GATT/SR/GAD/BV-08-C</td>
</tr>
<tr>
<td>GATT 3/8 AND GATT 7/7</td>
<td>Read Characteristic Value and Insufficient Authorization</td>
<td>GATT/CL/GAR/BI-03-C</td>
</tr>
<tr>
<td>GATT 3/8 AND GATT 7/4</td>
<td>Read Characteristic Value and Insufficient Authentication</td>
<td>GATT/CL/GAR/BI-04-C</td>
</tr>
<tr>
<td>GATT 3/8 AND GATT 7/2</td>
<td>Read Characteristic Value and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAR/BI-05-C</td>
</tr>
<tr>
<td>GATT 4/8 AND GATT 7/7</td>
<td>Read Characteristic Value and Insufficient Authorization</td>
<td>GATT/SR/GAR/BI-03-C</td>
</tr>
<tr>
<td>GATT 4/8 AND GATT 7/4</td>
<td>Read Characteristic Value and Insufficient Authentication</td>
<td>GATT/SR/GAR/BI-04-C</td>
</tr>
<tr>
<td>GATT 4/8 AND GATT 7/2</td>
<td>Read Characteristic Value and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAR/BI-05-C</td>
</tr>
<tr>
<td>GATT 3/9 AND GATT 7/7</td>
<td>Read using Characteristic UUID and Insufficient Authorization</td>
<td>GATT/CL/GAR/BI-09-C</td>
</tr>
<tr>
<td>GATT 3/9 AND GATT 7/4</td>
<td>Read using Characteristic UUID and Insufficient Authentication</td>
<td>GATT/CL/GAR/BI-10-C</td>
</tr>
<tr>
<td>GATT 3/9 AND GATT 7/2</td>
<td>Read using Characteristic UUID and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAR/BI-11-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td>GATT 4/9 AND GATT 7/7</td>
<td>Read using Characteristic UUID and Insufficient Authorization</td>
<td>GATT/SR/GAR/BI-09-C</td>
</tr>
<tr>
<td>GATT 4/9 AND GATT 7/4</td>
<td>Read using Characteristic UUID and Insufficient Authentication</td>
<td>GATT/SR/GAR/BI-10-C</td>
</tr>
<tr>
<td>GATT 4/9 AND GATT 7/2</td>
<td>Read using Characteristic UUID and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAR/BI-11-C</td>
</tr>
<tr>
<td>GATT 3/10 AND GATT 7/4</td>
<td>Read Long Characteristic Values and Insufficient Authentication</td>
<td>GATT/CL/GAR/BI-16-C</td>
</tr>
<tr>
<td>GATT 3/10 AND GATT 7/2</td>
<td>Read Long Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAR/BI-17-C</td>
</tr>
<tr>
<td>GATT 4/10 AND GATT 7/4</td>
<td>Read Long Characteristic Values and Insufficient Authentication</td>
<td>GATT/SR/GAR/BI-16-C</td>
</tr>
<tr>
<td>GATT 4/10 AND GATT 7/2</td>
<td>Read Long Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAR/BI-17-C</td>
</tr>
<tr>
<td>GATT 3/11 AND GATT 7/7</td>
<td>Read Multiple Characteristic Values and Insufficient Authorization</td>
<td>GATT/CL/GAR/BI-20-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>GATT 3/11 AND GATT 7/4</td>
<td>Read Multiple Characteristic Values and Insufficient Authentication</td>
<td>GATT/CL/GAR/BI-21-C</td>
</tr>
<tr>
<td>GATT 3/11 AND GATT 7/2</td>
<td>Read Multiple Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAR/BI-22-C</td>
</tr>
<tr>
<td>GATT 4/11 AND GATT 7/7</td>
<td>Read Multiple Characteristic Values and Insufficient Authorization</td>
<td>GATT/SR/GAR/BI-20-C</td>
</tr>
<tr>
<td>GATT 4/11 AND GATT 7/4</td>
<td>Read Multiple Characteristic Values and Insufficient Authentication</td>
<td>GATT/SR/GAR/BI-21-C</td>
</tr>
<tr>
<td>GATT 4/11 AND GATT 7/2</td>
<td>Read Multiple Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAR/BI-22-C</td>
</tr>
<tr>
<td>GATT 3/12</td>
<td>Write without Response</td>
<td>GATT/CL/GAW/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/12</td>
<td>Write without Response</td>
<td>GATT/SR/GAW/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/13 AND GATT 2/2</td>
<td>Signed Write Without Response</td>
<td>GATT/CL/GAW/BV-02-C</td>
</tr>
<tr>
<td>GATT 4/16 AND GATT 4/25</td>
<td>Characteristic Value Reliable Writes, Execute Write Request with no pending prepared write values</td>
<td>GATT/SR/GAW/BV-11-C</td>
</tr>
<tr>
<td>GATT 3/14 AND GATT 7/7</td>
<td>Write Characteristic Value and Insufficient Authorization</td>
<td>GATT/CL/GAW/BI-04-C</td>
</tr>
<tr>
<td>GATT 3/14 AND GATT 7/4</td>
<td>Write Characteristic Value and Insufficient Authentication</td>
<td>GATT/CL/GAW/BI-05-C</td>
</tr>
<tr>
<td>GATT 3/14 AND GATT 7/2</td>
<td>Write Characteristic Value and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAW/BI-06-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>GATT 4/14</td>
<td>Write Characteristic Value</td>
<td>GATT/SR/GAW/BV-03-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-02-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-03-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-32-C</td>
</tr>
<tr>
<td>GATT 4/14 AND GATT 7/7</td>
<td>Write Characteristic Value and Insufficient Authorization</td>
<td>GATT/SR/GAW/BI-04-C</td>
</tr>
<tr>
<td>GATT 4/14 AND GATT 7/4</td>
<td>Write Characteristic Value and Insufficient Authentication</td>
<td>GATT/SR/GAW/BI-05-C</td>
</tr>
<tr>
<td>GATT 4/14 AND GATT 7/2</td>
<td>Write Characteristic Value and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAW/BI-06-C</td>
</tr>
<tr>
<td>GATT 3/15</td>
<td>Write Long Characteristic Values</td>
<td>GATT/CL/GAW/BV-05-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/CL/GAW/BI-07-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/CL/GAW/BI-08-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/CL/GAW/BI-09-C</td>
</tr>
<tr>
<td>GATT 3/15 AND GATT 7/7</td>
<td>Write Long Characteristic Values and Insufficient Authorization</td>
<td>GATT/CL/GAW/BI-11-C</td>
</tr>
<tr>
<td>GATT 3/15 AND GATT 7/4</td>
<td>Write Long Characteristic Values and Insufficient Authentication</td>
<td>GATT/CL/GAW/BI-12-C</td>
</tr>
<tr>
<td>GATT 3/15 AND GATT 7/2</td>
<td>Write Long Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/CL/GAW/BI-13-C</td>
</tr>
<tr>
<td>GATT 4/15</td>
<td>Write Long Characteristic Values</td>
<td>GATT/SR/GAW/BV-05-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-07-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-08-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BI-09-C</td>
</tr>
<tr>
<td>GATT 4/15 AND GATT 7/7</td>
<td>Write Long Characteristic Values and Insufficient Authorization</td>
<td>GATT/SR/GAW/BI-11-C</td>
</tr>
<tr>
<td>GATT 4/15 AND GATT 7/4</td>
<td>Write Long Characteristic Values and Insufficient Authentication</td>
<td>GATT/SR/GAW/BI-12-C</td>
</tr>
<tr>
<td>GATT 4/15 AND GATT 7/2</td>
<td>Write Long Characteristic Values and Insufficient Encryption Key Size</td>
<td>GATT/SR/GAW/BI-13-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>GATT 3/16</td>
<td>Characteristic Value Reliable Writes</td>
<td>GATT/CL/GAW/BV-06-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/CL/GAW/BI-32-C</td>
</tr>
<tr>
<td>GATT 4/16</td>
<td>Characteristic Value Reliable Writes</td>
<td>GATT/SR/GAW/BV-06-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BV-07-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAW/BV-10-C</td>
</tr>
<tr>
<td>GATT 3/17</td>
<td>Notifications</td>
<td>GATT/CL/GAN/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/17</td>
<td>Notifications</td>
<td>GATT/SR/GAN/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/18</td>
<td>Indications</td>
<td>GATT/CL/GAI/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/18</td>
<td>Indications</td>
<td>GATT/SR/GAI/BV-01-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAT/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/19</td>
<td>Read Characteristic Descriptors</td>
<td>GATT/CL/GAR/BV-06-C</td>
</tr>
<tr>
<td>GATT 4/19</td>
<td>Read Characteristic Descriptors</td>
<td>GATT/SR/GAR/BV-06-C</td>
</tr>
<tr>
<td>GATT 3/20</td>
<td>Write Characteristic Descriptors</td>
<td>GATT/CL/GAW/BV-08-C</td>
</tr>
<tr>
<td>GATT 4/20</td>
<td>Write Characteristic Descriptors</td>
<td>GATT/SR/GAW/BV-08-C</td>
</tr>
<tr>
<td>GATT 3/20</td>
<td>Read Long Characteristic Descriptors</td>
<td>GATT/CL/GAR/BV-07-C</td>
</tr>
<tr>
<td>GATT 3/20 AND</td>
<td>Read Characteristic Value and Invalid</td>
<td>GATT/CL/GAR/BI-34-C</td>
</tr>
<tr>
<td>GATT 2/1</td>
<td>transport access over BR/EDR</td>
<td>GATT/CL/GAR/BI-35-C</td>
</tr>
<tr>
<td>GATT 3/20 AND</td>
<td>Read Characteristic Value and Invalid</td>
<td>GATT/CL/GAR/BI-34-C</td>
</tr>
<tr>
<td>GATT 2/2</td>
<td>transport access over LE</td>
<td>GATT/CL/GAR/BI-35-C</td>
</tr>
<tr>
<td>GATT 4/20</td>
<td>Read Long Characteristic Descriptors</td>
<td>GATT/SR/GAR/BV-07-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GATT/SR/GAR/BV-08-C</td>
</tr>
<tr>
<td>GATT 4/20 AND</td>
<td>Read Characteristic Value and Invalid</td>
<td>GATT/SR/GAR/BI-34-C</td>
</tr>
<tr>
<td>GATT 2/1 AND</td>
<td>transport access over supported transport</td>
<td>GATT/SR/GAR/BI-35-C</td>
</tr>
<tr>
<td>GATT 2/2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GATT 3/22</td>
<td>Write Long Characteristic Descriptors</td>
<td>GATT/CL/GAW/BV-09-C</td>
</tr>
<tr>
<td>GATT 4/22</td>
<td>Write Long Characteristic Descriptors</td>
<td>GATT/SR/GAW/BV-09-C</td>
</tr>
<tr>
<td>GATT 3/23</td>
<td>Service Changed</td>
<td>GATT/CL/GAS/BV-01-C</td>
</tr>
<tr>
<td>GATT 4/23</td>
<td>Service Changed</td>
<td>GATT/SR/GAS/BV-01-C</td>
</tr>
<tr>
<td>GATT 3/26</td>
<td>Database Hash Characteristic</td>
<td>GATT/CL/GAS/BV-02-C</td>
</tr>
<tr>
<td>Item</td>
<td>Feature</td>
<td>Test Case(s)</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| GATT 4/27 | Database Hash Characteristic | GATT/SR/GAS/BV-02-C  
| | | GATT/SR/GAS/BV-05-C  
| | | GATT/SR/GAS/BV-06-C  
| GATT 4/27 AND (GAP 24/2 OR GAP 24/3 OR GAP 34/2 OR GAP 34/3) | Database Hash Characteristic  
| | Bonding | GATT/SR/GAS/BV-04-C  
| | | GATT/SR/GAS/BV-07-C  
| GATT 3/25 | Client Supported Features Characteristic | GATT/CL/GAS/BV-03-C  
| GATT 4/26 | Client Supported Features Characteristic | GATT/SR/GAS/BV-03-C  
| GATT 3/28 | Characteristic Aggregate Format Descriptor | GATT/CL/GPA/BV-11-C  
| GATT 4/29 | Characteristic Aggregate Format Descriptor | GATT/SR/GPA/BV-11-C  
| GATT 3/27 | Characteristic Format Descriptor by Client | GATT/CL/GPA/BV-12-C  
| GATT 4/28 | Characteristic Format Descriptor by Server | GATT/SR/GPA/BV-12-C  
| GATT 8/1 | Support for Multiple ATT bearers from same device | GATT/SR/GPM/BV-01-C  

*Table 5.1: Test Case Mapping*
6 ANNEX: Generic GATT Integrated Tests (GGIT)

This annex defines generic test procedures that integrate various GATT tests that are required to be executed for each supported service and characteristic.

6.1 Identification Conventions

In addition to the conventions defined in 4.1.1 Test Case Identification Conventions, the following identifiers are introduced in this annex.

<table>
<thead>
<tr>
<th>Identifier Abbreviation</th>
<th>Group Identifier &lt;class&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>GGIT</td>
<td>Generic GATT Integrated Tests</td>
</tr>
<tr>
<td>SGGIT</td>
<td>Server GGIT test procedures</td>
</tr>
<tr>
<td>CGGIT</td>
<td>Client GGIT test procedures</td>
</tr>
<tr>
<td>SER</td>
<td>Service</td>
</tr>
<tr>
<td>CHA</td>
<td>Characteristic</td>
</tr>
<tr>
<td>DES</td>
<td>Descriptor</td>
</tr>
</tbody>
</table>

Table 6.1: Identification Convention

6.2 GGIT Inputs

The GGIT procedures are executed with an input consisting of a table; the table lists all the services and characteristics that are being tested, as well as descriptors and parameters. The CCCD is not listed as a descriptor in the table; its presence is inferred from the characteristic properties (the CCCD is present when the characteristic supports indications or notifications).

The GGIT table format is defined below:

<table>
<thead>
<tr>
<th>TCID</th>
<th>Service / Characteristic / Descriptor &lt;func&gt;</th>
<th>Reference</th>
<th>Characteristic Properties</th>
<th>Value Length (Octets)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCID</td>
<td>Service_1</td>
<td>[SPEC] X.Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TCID</td>
<td>Characteristic_11</td>
<td>[SPEC] X.Y</td>
<td>0xZZ</td>
<td>N, Min-Max or Skip</td>
</tr>
<tr>
<td>TCID</td>
<td>Descriptor_111</td>
<td>[SPEC] X.Y</td>
<td>-</td>
<td>N or Min-Max</td>
</tr>
<tr>
<td>TCID</td>
<td>Descriptor_112</td>
<td>[SPEC] X.Y</td>
<td>-</td>
<td>N or Min-Max</td>
</tr>
</tbody>
</table>
Table 6.2: GGIT Input Table Format

The recommended format for the TCID of these tests is:

<spec abbreviation>/ROLE/SGGIT/<func>/BV-XX-C, (for Server Tests), and

<spec abbreviation>/ROLE/CGGIT/<func>/BV-XX-C, (for Client Tests), and

6.2.1 Example Usage – Blood Pressure Service TS

[This is an extract showing how the Blood Pressure Service TS can reference the GGIT as an alternative to creating test cases for each service and characteristic related procedures.]

Execute the Generic GATT Integrated Tests defined in [GATT.TS.REF#] 6.3 Server test procedures (SGGIT) using Table 6.3 below as input:

[REF#] below is the reference to the related specification (in this case Blood Pressure Service) as the BLS.TS defines it.

<table>
<thead>
<tr>
<th>TCID</th>
<th>Service / Characteristic / Descriptor &lt;func&gt;</th>
<th>Reference</th>
<th>Characteristic Properties</th>
<th>Value Length (Octets)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCID</td>
<td>Characteristic_12</td>
<td>[SPEC] X.Y</td>
<td>0xZZ</td>
<td>N, Min-Max or Skip</td>
</tr>
<tr>
<td>TCID</td>
<td>Descriptor_121</td>
<td>[SPEC] X.Y</td>
<td>-</td>
<td>N or Min-Max</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>TCID</td>
<td>Service_2</td>
<td>[SPEC] X.Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TCID</td>
<td>Characteristic_21</td>
<td>[SPEC] X.Y</td>
<td>0xZZ</td>
<td>N, Min-Max or Skip</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TCID</th>
<th>Service / Characteristic / Descriptor &lt;func&gt;</th>
<th>Reference</th>
<th>Properties</th>
<th>Value Length (Octets)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLS/SEN/SGGIT/SER/BV-01-C [Service GGIT – Blood Pressure]</td>
<td>Blood Pressure Service</td>
<td>[REF#] 2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BLS/SEN/SGGIT/CHA/BV-02-C [Characteristic GGIT – Blood Pressure Measurement]</td>
<td>Blood Pressure Measurement Characteristic</td>
<td>[REF#] 3</td>
<td>0x20 (Notificatio n)</td>
<td>7-19</td>
</tr>
<tr>
<td>TCID</td>
<td>Service / Characteristic / Descriptor</td>
<td>Reference</td>
<td>Properties</td>
<td>Value Length (Octets)</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------</td>
<td>-----------</td>
<td>------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>BLS/SEN/SGGIT/CHA/BV-03-C [Characteristic GGIT – Intermediate Cuff Pressure]</td>
<td>Intermediate Cuff Pressure Characteristic</td>
<td>[REF#] 3</td>
<td>0x10</td>
<td>7-19</td>
</tr>
<tr>
<td>BLS/SEN/SGGIT/CHA/BV-04-C [Characteristic GGIT – Blood Pressure Feature]</td>
<td>Blood Pressure Feature Characteristic</td>
<td>[REF#] 3</td>
<td>0x02</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 6.3: Input for the GGIT Server Test Procedure

### 6.3 Server test procedures (SGGIT)

For each row of the Server GGIT input table:

- if the row defines a service, then the test sequence in 6.3.1 SGGIT/SER [Service GGIT] is executed.
- if the row defines a characteristic, then the test sequence in 6.3.2 SGGIT/CHA [Characteristic GGIT] is executed.
- if the row defines a characteristic descriptor other than the CCCD, then the test sequence in 6.3.3 SGGIT/DES [Descriptor GGIT] is executed.

#### 6.3.1 SGGIT/SER [Service GGIT]

Execute all the following tests using the Service UUID from the current row of the input table, or until one of the referenced tests fails:

1. Execute GATT/SR/GAD/BV-01-C [Discover All Primary Services - from Server] or GATT/SR/GAD/BV-07-C [Discover Primary Services using SDP - from Server], depending on the transports tested for this Service UUID, to verify that the Service UUID is present in the list of Primary Services retrieved from the IUT.
2. Execute GATT/SR/GAD/BV-02-C [Discover Primary Services by Service UUID - from server] or GATT/SR/GAD/BV-08-C [Discover Services by UUID using SDP - from Server] with the Service UUID, depending on the transports tested for this Service UUID, to verify that the UUID is found in the database of the IUT. Save the service handle range for the next step.
3. Execute GATT/SR/GAD/BV-04-C [Discover All Characteristics of a Service – from Server] with the service handle range to verify that all the characteristics listed under the Service UUID in the input table are found in the database of the IUT.

#### 6.3.2 SGGIT/CHA [Characteristic GGIT]

Execute all the following tests using the Characteristic UUID from the current row of the input table, or until one of the referenced tests fails:

1. Execute GATT/SR/GAD/BV-05-C [Discover Characteristics by UUID – from server] with the Characteristic UUID to verify that the UUID is found in the database of the IUT. Save the characteristic properties and handle range for the next steps.
2. Execute GATT/SR/GAD/BV-06-C [Discover All Characteristic Descriptors – from Server] with the characteristic handle range to verify that all the characteristic descriptors listed under the
Characteristic UUID in the input table are found in the database of the IUT. If the characteristic supports Notifications or Indications, then expect the CCCD to be present in the list of descriptors. Save all handle ranges for the next steps.

3. If the *Value Length (Octets)* column specifies “Skip” for this characteristic, skip steps 4-8.

4. If the characteristic supports the Read property, then:
   a. execute GATT/SR/GAR/BV-01-C [Read Characteristic Value - from Server] if the value length allows small reads.
   b. execute GATT/SR/GAR/BV-03-C [Read using Characteristic UUID - from Server].
   c. execute GATT/SR/GAR/BV-04-C [Read Long Characteristic Value - from Server] if the value length allows long reads.

5. If the characteristic supports the Write Without Response property, then execute GATT/SR/GAW/BV-01-C [Write Without Response - to Server].

6. If the characteristic supports the Write property, then:
   a. execute GATT/SR/GAW/BV-03-C [Write Characteristic Value - to Server] if the value length allows small writes.

7. If the characteristic supports the Notification property, then execute GATT/SR/GAN/BV-01-C [Characteristic Value Notification - by Server] and as required, perform a service-specific action to generate a new characteristic value.

8. If the characteristic supports the Indication property, then execute GATT/SR/GAI/BV-01-C [Characteristic Value Indication - by Server] and as required, perform a service-specific action to generate a new characteristic value.

6.3.3 SGGIT/DES [Descriptor GGIT]

Execute all the following tests using the discovered handle of the Descriptor UUID from the current row of the input table, or until one of the referenced tests fails:

1. execute GATT/SR/GAR/BV-06-C [Read Characteristic Descriptors – from Server] if the value length allows small reads.
2. execute GATT/SR/GAR/BV-07-C [Read Long Characteristic Descriptor - from Server] if the value length allows long reads.
3. execute GATT/SR/GAW/BV-08-C [Write Characteristic Descriptors – from Server] if the value length allows small writes.

6.4 Client test procedures (CGGIT)

For each row of the Client GGIT input table:

- if the row defines a service, then the test sequence in 6.4.1 CGGIT/SER [Service GGIT] is executed.
- if the row defines a characteristic, then the test sequence in 6.4.2 CGGIT/CHA [Characteristic GGIT] is executed.
- if the row defines a characteristic descriptor other than the CCCD, then the test sequence in 6.4.3 CGGIT/DES [Descriptor GGIT] is executed.
6.4.1 CGGIT/SER [Service GGIT]

Execute all the following tests using the Service UUID from the current row of the input table, or until one of the referenced tests fails:

1. If the IUT supports the GATT Discover All Primary Services sub-procedure, then execute GATT/CL/GAD/BV-01-C [Discover All Primary Services - by Client] or GATT/CL/GAD/BV-07-C [Discover Primary Services using SDP - by client], depending on the transports tested for this Service UUID, to verify that the IUT discovers all the primary services and reports that the Service UUID is present in the list of Primary Services retrieved from the Lower Tester.

2. If the IUT supports the GATT Discover Primary Services by UUID sub-procedure, then execute GATT/CL/GAD/BV-02-C [Discover Primary Services by Service UUID – by Client] or GATT/CL/GAD/BV-08-C [Discover Services by UUID using SDP - by Client] with the Service UUID, depending on the transports tested for this Service UUID, to verify that the IUT finds the UUID in the database of the Lower Tester and reports the correct handle range.

3. If the IUT supports the GATT Discover All Characteristics of a Service sub-procedure, then execute GATT/CL/GAD/BV-04-C [Discover All Characteristics of a Service – by Client] with the service handle range to verify that all the characteristics listed under the Service UUID in the input table are found by the IUT in the database of the Lower Tester.

6.4.2 CGGIT/CHA [Characteristic GGIT]

Execute all the following tests using the Characteristic UUID from the current row of the input table, or until one of the referenced tests fails:

1. If the IUT supports the GATT Discover Characteristics by UUID sub-procedure, then execute GATT/CL/GAD/BV-05-C [Discover Characteristics by UUID – by Client] with the Characteristic UUID to verify that the IUT finds the UUID in the database of the IUT and reports the correct characteristic properties and handle range.

2. If the IUT supports the GATT Discover All Characteristic Descriptors sub-procedure, then execute GATT/CL/GAD/BV-06-C [Discover All Characteristic Descriptors – by client] with the characteristic handle range to verify that all the characteristic descriptors listed under the Characteristic UUID in the input table are found by the IUT in the database of the Lower Tester and that the correct handle ranges are reported.

3. If the Value Length (Octets) column specifies "Skip" for this characteristic, skip steps 4-6.

4. If the characteristic supports the Read property, then:
   a. if the IUT supports the GATT Read Characteristic Value sub-procedure, then execute GATT/CL/GAR/BV-01-C [Read Characteristic Value - by Client] if the value length allows small reads.
   b. if the IUT supports the GATT Read Using Characteristic UUID sub-procedure, then execute GATT/CL/GAR/BV-03-C [Read Using Characteristic UUID - by Client].
   c. if the IUT supports the GATT Read Long Characteristic Value sub-procedure, then execute GATT/CL/GAR/BV-04-C [Read Long Characteristic Value - by Client] if the value length allows long reads.

5. If the characteristic supports the Write Without Response property and if the IUT supports the GATT Write Without Response sub-procedure, then execute GATT/CL/GAW/BV-01-C [Write without Response - by Client].
6. If the characteristic supports the Write property, then:
   a. if the IUT supports the GATT Write Characteristic Value sub-procedure, then execute 
      GATT/SR/GAW/BV-03-C [Write Characteristic Value - to Server] if the value length 
      allows small writes.
   b. if the IUT supports the GATT Write Long Characteristic Value sub-procedure, then 
      execute GATT/CL/GAW/BV-05-C [Write Long Characteristic Value - by Client] if the value 
      length allows long writes.

6.4.3 CGGIT/DES [Descriptor GGIT]
Execute all the following tests using the discovered handle of the Descriptor UUID from the current row of 
the input table, or until one of the referenced tests fails:
1. if the IUT supports the GATT Read Characteristic Descriptors sub-procedure, then execute 
   GATT/CL/GAR/BV-06-C [Read Characteristic Descriptors – by Client] if the value length allows small 
   reads.
2. if the IUT supports the GATT Read Long Characteristic Descriptors sub-procedure, then execute 
   GATT/CL/GAR/BV-07-C [Read Long Characteristic Descriptor - by Client] if the value length allows 
   long reads.
3. if the IUT supports the GATT Write Characteristic Descriptors sub-procedure, then execute 
   GATT/CL/GAW/BV-08-C [Write Characteristic Descriptors – by Client] if the value length allows small 
   writes.
4. if the IUT supports the GATT Write Long Characteristic Descriptors sub-procedure, then execute 
   GATT/CL/GAW/BV-09-C [Write Long Characteristic Descriptors – by Client] if the value length allows 
   long writes.
### 7 Revision History and Contributors

**Revision History**

<table>
<thead>
<tr>
<th>Revision History</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.0.0</td>
<td>2010-06-30</td>
<td>Publication.</td>
</tr>
<tr>
<td>4.0.1r00</td>
<td>2010-08-24</td>
<td>TSE 3427: new text in 4.2.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 3911: new test TP/GAN/CL/BV-01-C (GATT/CL/GAN/BV-01-C after ID conversion) and TCMT entry for 3/17</td>
</tr>
<tr>
<td>4.0.1r01 – 4.0.1r12</td>
<td>2010-10-10 – 2011-06-13</td>
<td>TSE 3859: edit 4.2.3; add missing references to 4.2.3.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 3911: add timer to TP/GAN/CL/BV-01-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 3937: correct opcode for <code>$ATT_Write_Command</code> in TP/GAW/xx/BV-01-C, TP and TP/GAW/xx/BV-02-C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4106: 128-bit UUID optional in TP/GAD/CL/BV-01-C (GATT/CL/GAD/BV-01-C after ID conversion), TP/GAR/CL/BV-03-C (GATT/CL/GAR/BV-03-C after ID conversion) and TP/GAD/CL/BV-02-I</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4107 &amp; 4122: explicitly state the option to use <code>$ATT_Read_Request</code> as the first operation in Read Long Characteristic tests: TP/GAR/CL/BV-04-C (GATT/CL/GAR/BV-04-C after ID conversion) and TP/GAR/SR/BI-12-C through TP/GAR/SR/BI-17-C (GATT/SR/GAR/BI-12-C through GATT/SR/BI-17-C after ID conversion).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4132: edit initial condition in TP/GAR/SR/BV-03-C (GATT/SR/GAR/BV-03-C after ID conversion).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4134: add <code>$ATT_Prepae_Write_Response</code> and <code>$ATT_Execute_Write_Request</code>, in test procedure, MSC and Pass Verdict.</td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>TSE 4123: Update hyperlinks page 16, 26, 37, 43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4132: TP/GAR/SR/BV-03-C: update Test procedure (GATT/SR/GAR/BV-03-C after ID conversion)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4160: TP/GAW/CLBI-32;C new test case (GATT/CL/GAW/BI-32-C after ID conversion)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4170: Update references from 4.2.1 and 4.2.1.2 to 3.2.1 and 3.2.1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4179: TP/GAW/CL/BI-09-C (GATT/CL/GAW/BI-09-C after ID conversion): edit Test Procedure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>TSE 4237: Duplicate of TSE 4170</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update TCMT per TSE 4236 comment ID 10166</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4251: TP/GAC/CL/BV-01-C (GATT/CL/GAC/BV-01-C after ID conversion): Test Procedure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4307: TP/GAW/CL/BV-02-C (GATT/CL/GAW/BV-02-C after ID conversion): Update TCMT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposed edits for TSE 4331:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include ATT in the scope (1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add reference to ATT.ICS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Redirect the (2.5) reference to ATT TCMT to the new 4.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Split Test Case Mapping into 4.1 (GATT) and 4.2 (ATT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Replicate the ATT TCMT from ATT.TS 4.0.1 r3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Editorial cleanup in the references</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4326: GATT TCMT changes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Input review comments for TSE 4331</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Jdecuir’s comments for italics, xref colors, section</td>
<td></td>
<td></td>
</tr>
<tr>
<td>numbering, missing graphic, comment deletion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Added header to ATT TCMT.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Changed formatting to “ATT:” and “GATT:” in TCMTs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4386: Change mapping for TP/GAT/SR/BV-01-C from ATT: 4/30 to</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATT: 4/27 (GATT/SR/GAT/BV-01-C after ID conversion)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Separate GAS/CL and GAS/SR to have separate mappings</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCMT: Switch server/client timeout features based on changes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>to ATT ICS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4401: Remove duplicate entries in TCMT by consolidating</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rows</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prepare for publication.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4210: update UUIDs for TP/GAD/SR/BV-01-C,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4327: update pass verdicts for TP/GAR/SR/BV-04-C,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4366: update TP/GAR/CL/BV-04-C, TP/GAR/SR/BV-04-C,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cases TP/GAW/SR/BV-08-C, TP/GAW/SR/BV-10-C (GATT/SR/GAW/BV-08-C, GATT/SR/GAW/BV-10-C after ID conversion)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4384: Update section 3.3.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4503: TP/GAD/CL/BV-03-C, TP/GAD/CL/BV-03-I,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP/GAD/SR/BV-03-C (GATT/CL/GAD/BV-03-C and GATT/SR/GAD/BV-03-C after ID conversion); add additional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pass Verdict</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4540: TCMT updates for TP/GAR/SR/BI-34-C,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 4564: TP/GAW/CL/BV-08-C, TP/GAW/CL/BI-20-C,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4787: Changes to test case TP/GAD/CL/BV-03-C, and TP/GAD/CL/BV-03-I (GATT/CL/GAD/BV-03-C after ID conversion)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4841: Changes to test case TP/GPA/CL/BV-12-C (GATT/CL/GPA/BV-12-C after ID conversion)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4810: Added a paragraph to the initial condition of TP/GPA/CL/BV-03-C (GATT/CL/GPA/BV-03-C after ID conversion) to correct the &quot;can’t specifically look for Secondary Services in GATT” problem.</td>
</tr>
<tr>
<td>4.0.4r2</td>
<td>2012-11-12</td>
<td>Edits to TSE 4943, and 4847.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 4688: Changes to TP/GAD/CL/BV-07-C, TP/GAD/CL/BV-08-C, TP/GAD/SR/BV-07-C, and TP/GAD/SR/BV-08-C (GATT/CL/GAD/BV-07-C, GATT/CL/GAD/BV-08-C, GATT/SR/GAD/BV-07-C, and GATT/SR/GAD/BV-08-C after ID conversion). Added MSCs (examples) and removed the “All SDP records for GATT Services conform to the requirements in Table 9.1 in [1]….” In these test cases.</td>
</tr>
<tr>
<td>4.0.4r3</td>
<td>2012-11-15</td>
<td>TSE 5012: Edits to TP/GAW/CL/BV-02-C and TP/GAW/SR/BV-02-C (GATT/CL/GAW/BV-02-C and GATT/SR/GAW/BV-02-C after ID conversion) to change ATT_Write_Command to ATT_Signed_Write_Command.</td>
</tr>
<tr>
<td>4.0.4</td>
<td>2012-11-16</td>
<td>Prepare for Publication</td>
</tr>
<tr>
<td>4.0.5rT</td>
<td>2013-05-24</td>
<td>Template Conversion: - Update of language to match BTI approved wording (example, fail verdicts) - Removal of Test Subgroup Objectives - Removal of sections marked “N/A”</td>
</tr>
<tr>
<td>4.0.5r1</td>
<td>2013-05-29</td>
<td>TSE 5011: Updated MSC in TP/GAW/CL/BV-01-C (GATT/CL/GAW/BV-01-C after ID conversion).</td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 5142: Deleted the following test cases: TP/GAD/CL/BV-01-I, TP/GAD/CL/BV-02-I, TP/GAD/CL/BV-03-I, TP/GAD/CL/BV-04-I, TP/GAD/CL/BV-05-I and TP/GAD/CL/BV-06-I. The implications of this removal are that the corresponding –C test cases will become category B, as tracked in the TCRL.</td>
</tr>
<tr>
<td>4.0.5r2</td>
<td>2013-06-06</td>
<td>BTI Review, Comments from Magnus and Alicia</td>
</tr>
<tr>
<td>4.0.5</td>
<td>2013-07-02</td>
<td>Prepare for Publication</td>
</tr>
<tr>
<td>4.0.6rT, 4.0.6rTr3</td>
<td>2013-07-03</td>
<td>Template Conversion:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Update of language to match BTI approved wording (example, removed fail verdicts)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Removal of Test Subgroup Objectives and sections marked “N/A”</td>
</tr>
<tr>
<td>4.0.6rTr4</td>
<td>2013-09-27</td>
<td>Template Review Comment Resolution &amp; Changes Accepted.</td>
</tr>
<tr>
<td>4.1.0r02</td>
<td>2013-10-04</td>
<td>Dual Mode Topology CR</td>
</tr>
<tr>
<td>4.1.0r03</td>
<td>2013-10-07</td>
<td>TSE 5290: Removed TP/GAD/CL/BV-02-I from the TCMT.</td>
</tr>
<tr>
<td>4.1.0r04</td>
<td>2013-11-08</td>
<td>Added IXIT to references</td>
</tr>
</tbody>
</table>
## Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.0</td>
<td>2013-12-03</td>
<td>Prepare for Publication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TSE 5487: Updated the discovery command in step one of the test procedure and pass verdict of TP/GPA/SR/BV-12-C (GATT/SR/GPA/BV-12-C after ID conversion) from &quot;ATT_Read_by_Type_Request&quot; to &quot;ATT_Find_Information_Request.&quot;</td>
</tr>
<tr>
<td>4.1.1r01</td>
<td>2014-06-17</td>
<td>BTI Review, Magnus, editorial corrections</td>
</tr>
<tr>
<td>4.1.1</td>
<td>2014-07-07</td>
<td>TCRL 2014-1 Publication</td>
</tr>
<tr>
<td>4.2.0r00</td>
<td>2014-11-24</td>
<td>Revved version to align with Core 4.2 release</td>
</tr>
<tr>
<td>4.2.0</td>
<td>2014-12-04</td>
<td>Prepared for TCRL 2014-2 publication</td>
</tr>
<tr>
<td>4.2.1r00</td>
<td>2015-05-05</td>
<td>TSE 6274: Corrected references in TP/GPM/SR/BV-01-C (GATT/SR/GPM/BV-01-C after ID conversion)</td>
</tr>
<tr>
<td>4.2.1</td>
<td>2015-07-14</td>
<td>Prepared for TCRL 2015-1 publication</td>
</tr>
<tr>
<td>4.2.2r00</td>
<td>2015-10-07</td>
<td>TSE 6271: Corrected maximum value allowed for MTU size; added additional initial condition for TP/GAC/SR/BV-01-C (GATT/SR/GAC/BV-01-C after ID conversion); corrected ATT_Exchange_MTU_Request value in MSC; and corrected reference to ATT in Section 4.3.2.</td>
</tr>
<tr>
<td>4.2.2r01</td>
<td>2015-10-12</td>
<td>TSE 6392: Corrected test case mapping for TP/GAW/CL/BV-02-C (GATT/CL/GAW/BV-02-C after ID conversion)</td>
</tr>
<tr>
<td>4.2.2r02</td>
<td>2015-10-23</td>
<td>Reviewed by Alicia Courtney. Updated MSC for TP/GAC/SR/BV-01-C (GATT/SR/GAC/BV-01-C after ID conversion); editorial corrections to TCMT.</td>
</tr>
<tr>
<td>4.2.2</td>
<td>2015-12-22</td>
<td>Prepared for TCRL 2015-2 publication.</td>
</tr>
<tr>
<td>4.2.3r00, 4.2.3r01, 4.2.3r02, 4.2.3r03</td>
<td>2016-02-01, 2016-02-17, 2016-02-20, 2016-02-23</td>
<td>Converted test case IDs to new convention defined in TSTO v4.1 and transfer over to new template</td>
</tr>
<tr>
<td>Revision History</td>
<td>Date</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 4.2.3r04         | 2016-03-01 | TSE 6790: Expanded Initial Condition for test case GATT/SR/GPA/BV-02-C.  
TSE 5483: Major change. Rewrite of Sections GATT Declarations and Descriptors – by Client (test cases GATT/CL/GPA/BV-01-C through GATT/CL/GPA/BV-08-C) and GATT Declarations and Descriptors – from Server (test cases GATT/SR/GPA/BV-01-C through GATT/SR/GPA/BV-08-C) (Sections 4.10.1 and 4.10.2). |
| 4.2.3r05         | 2016-03-09 | TSE 6651: TCMT Items updated to remove ATT ICS Mapping which has been moved instead to the ICS conditionals.                                                                                                                                                                                                                           |
| 4.2.3r06         | 2016-04-27 | Corrected some omissions from the conversion to the new test case ID conventions.                                                                                                                                                                                                                                                       |
| 4.2.3            | 2016-07-13 | Prepared for TCRL 2016-1 publication.                                                                                                                                                                                                                                                                                                  |
| 5.0.0r00         | 2016-10-07 | TSE 7558 (erratum 4654) - Clarification for octet alignment in Characteristic Aggregate Format tests GATT/CL/GPA/BV-11-C and GATT/SR/GPA/BV-11-C.  
TSE 7798: Corrected mapping for GATT/CL/GAW/BV-02-C from GATT 0/2 to 2/2  
Advanced document number to 5.0.0 to align with Bluetooth Core Spec 5.0 release.                                                                                                                                                                                                 |
| 5.0.0r01         | 2016-10-17 | TSE 7573 (erratum 5610): Added new test case GATT/SR/GAW/BV-11-C for Characteristic Value Reliable Writes - No Pending Prepared Write Requests. Updated TCMT with new entry.                                                                                                                                                                        |
| 5.0.0r02         | 2016-10-19 | Completed conversion to new Test Case ID conventions as defined in TSTO v4.1                                                                                                                                                                                                                                                       |
| 5.0.0            | 2016-12-13 | Approved by BTI. Prepared for TCRL 2016-2 publication.                                                                                                                                                                                                                                                                             |
| 5.0.1r00         | 2017-04-10 | TSE 8719: Changed initial condition of GATT/CL/GAR/BI-34-C and GATT/SR/GAR/BI-35-C to “Section 4.2.1.1”.  
Changed initial condition of GATT/CL/GAR/BI-35-C, GATT/SR/GAR/BI-34-C to “Section 4.2.1.2”.  
Cleaned up TCMT for tests GATT/SR/GAR/BI-34-C and GATT/SR/GAR/BI-35-C by merging rows in TCMT and updating feature to add “supported transport” as reviewers pointed out the test case mapping was the same for the two tests. |
<p>| 5.0.1            | 2017-07-05 | Approved by BTI. Prepared for TCRL 2017-1 publication.                                                                                                                                                                                                                                                                             |
| 5.0.2r00         | 2017-08-29 | TSE 9718: Added UNS to Table 4.1. Added new test cases GATT/SR/UNS/BI-01-C and GATT/SR/UNS/BI-02-C in new section “Unsupported Requests and Commands” and in the TCMT.                                                                                                                                                             |</p>
<table>
<thead>
<tr>
<th>Revision History</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0.2</td>
<td>2017-12-07</td>
<td>Approved by BTI. Prepared for TCRL 2017-2 publication.</td>
</tr>
</tbody>
</table>
| 5.0.3r00-04      | 2018-02-20 – 2018-05-10 | TSE 10284 (rating 1): Updated Reference for GATT/SR/UNS/BI-01-C.  
TSE 10347 (rating 3): Simplified the Initial Condition, Test Procedure and MSC, and the Pass Verdict of test case GATT/SR/GAC/BV-01-C to show IUT doesn't support Read Blob.  
TSE 10525 (rating 3): Added a preamble for Exchange MTU. Updated the initial conditions globally to remove references to GATT/SR/GAC/BV-01-C and GATT/CL/GAC/BV-01-C and instead replaced with a statement referencing the new preamble section.  
TSE 10522 (rating 3): Removed invalid behavior from the initial condition of test cases GATT/CL/GAW/BI-02-C and 07-C; and GATT/CL/GAR/BI-14-C, 19-C, and 01-C.  
TSE 10369 (rating 3): Changed name of test case GATT/SR/GPA/BV-12-C to [Characteristic Presentation Format Descriptors – from Server]. Added 3.3.3.5 to Reference 1. Updated test procedure and pass verdict to include "Presentation" in the test case name.  
TSE 10657 (rating 3): Removed invalid behavior from initial condition for test cases GATT/CL/GAR/BI-02-C, 06-C, 12-C, and 18-C; and GATT/CL/GAW/BI-03-C and 08-C. |
| 5.0.3            | 2018-07-02 | Approved by BTI. Prepared for TCRL 2018-1 publication. |
| 5.0.4r00-r03     | 2018-07-27 - 2018-11-12 | Incorporated GATT Caching TEST CR r09. Added new TCIDs GATT/CL/GAS/BV-02-C and 03-C; GATT/SR/GAS/BV-02-C through 07-C; and added them to the TCMT.  
TSE 10616 (rating 2): In the TCMT, removed test case GATT/SR/GAR/BV-08-C from GATT 4/10, and added test case GATT/SR/GAR/BV-08-C to GATT 4/20.  
Added new reference for Bluetooth Core Specification version 5.1. |
<table>
<thead>
<tr>
<th>Revision History</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1.0r00-r01</td>
<td>2018-11-13 - 2018-11-21</td>
<td>Updated revision number from 5.0.4 to 5.1.0 to align with the adoption of Core Specification version 5.1. TSE 11290 (rating 4): Addition of ANNEX: Generic GATT Integrated Tests (GGIT), 6.1 GGIT Inputs, 6.2 Server Test Procedures (SGGIT), 6.3 Client Test Procedures (CGGIT) and associated subsections.</td>
</tr>
<tr>
<td>5.1.0</td>
<td>2018-12-07</td>
<td>Approved by BTI. Prepared for TCRL 2018-2 publication.</td>
</tr>
</tbody>
</table>
| 5.1.1r00–r06     | 2019-02-19–2019-06-18 | TSE 11481 (rating 3): Updated steps 6 and 7 in SGGIT/CHA [Characteristic GGIT] to be service-specific. TSE 11466 (rating 3): Updated section heading and preamble procedure for "Characteristic Configuration for Notification" and "Characteristic Configuration for Indication". Added new sections "Characteristic Configuration for Notification – IUT as Client" and "Characteristic Configuration for Indication – IUT as Client". Updated initial condition for test cases GATT/CL/GAR/BV-01-C, 03-C, 04-C, 07-C; GATT/SR/GAR/BV-03-C; GATT/CL/GAW/BV-01-C, 08-C; GATT/SR/GAW/BV-01-C; GATT/CL/GAN/BV-01-C. Updated initial condition and pass verdict for test cases GATT/CL/GAR/BV-05-C, 06-C; GATT/SR/GAR/BV-06-C; GATT/CL/GAW/BV-03-C, 05-C, 09-C. Updated initial condition, test procedure, and pass verdict for test cases GATT/SR/GAR/BV-05-C; GATT/CL/GAW/BV-06-C; GATT/CL/GAW/BI-32-C. Updated pass verdict for test case GATT/CL/GAI/BV-01-C. Updated Characteristic_11, 12, and 21 in Table 6.2 GGIT Input Table Format. Updated 2nd paragraph in section "Example Usage – Blood Pressure Service TS". Updated steps 1 and 2 in section "SGGIT/SER [Service GGIT]". Added new step 3 in section "SGGIT/CHA [Characteristic GGIT]". Updated steps 1 and 2 in section "CGGIT/SER [Service GGIT]". Added new step 3 and removed steps 7 and 8 in section "CGGIT/CHA [Characteristic GGIT]". TSE 11617 (rating 3): Updated text in Encryption Key Size section; revised the initial conditions (and MSCs where
Table: Revision History

<table>
<thead>
<tr>
<th>Revision History</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSE 11538 (rating 4): Removed Complete GATT and redundant test cases no longer needed that rely on higher layer profiles or are implementation specific (deleted test cases GATT/CL/GPA/BV-01-C – -08-C and GATT/SR/GPA/BV-01-C – -08-C). Revised TCMT for GATT/SR/GPA/BV-11-C and -12-C and GATT/CL/GPA/BV-11-C and -12-C based on ICS changes from this TSE.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 11775 (rating 1): Updated the MSC for test case GATT/SR/GAS/BV-01-C to add further explanation to the sequence diagram.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE 11943 (rating 1): Replaced the MSC for test case GATT/CL/GAW/BI-09-C to fix an invalid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1.1</td>
<td>2019-08-01</td>
<td>Approved by BTI. Prepared for TCRL 2019-1 publication.</td>
</tr>
</tbody>
</table>

Contributors

<table>
<thead>
<tr>
<th>Name</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elisa Rincón</td>
<td>AT4 wireless</td>
</tr>
<tr>
<td>Angel Romero</td>
<td>AT4 wireless</td>
</tr>
<tr>
<td>Juanma Hidalgo</td>
<td>AT4 wireless</td>
</tr>
<tr>
<td>Elizabeth Dominguez</td>
<td>AT4 wireless</td>
</tr>
<tr>
<td>Tim Eichelberger</td>
<td>Bluetooth SIG</td>
</tr>
<tr>
<td>Manivannan Elangovan</td>
<td>Bluetooth SIG</td>
</tr>
<tr>
<td>Aravind Narasimhan</td>
<td>Bluetooth SIG</td>
</tr>
<tr>
<td>Yung Ming Kung</td>
<td>Bluetooth SIG</td>
</tr>
<tr>
<td>Name</td>
<td>Company</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Meagan Schuver</td>
<td>Bluetooth SIG</td>
</tr>
<tr>
<td>Alicia Courtney</td>
<td>Broadcom</td>
</tr>
<tr>
<td>Norbert Grunert</td>
<td>Broadcom</td>
</tr>
<tr>
<td>Joe Decuir</td>
<td>CSR</td>
</tr>
<tr>
<td>Robin Heydon</td>
<td>CSR</td>
</tr>
<tr>
<td>Magnus Sommansson</td>
<td>CSR</td>
</tr>
<tr>
<td>Rasmus Abildgren</td>
<td>CSR</td>
</tr>
<tr>
<td>Tim Howes</td>
<td>Nokia</td>
</tr>
<tr>
<td>Miika Laaksonen</td>
<td>Nokia</td>
</tr>
<tr>
<td>Frank Karlsen</td>
<td>Nordic Semiconductor</td>
</tr>
<tr>
<td>Sebastien Mackaie-Blanchi</td>
<td>Nordic Semiconductor</td>
</tr>
<tr>
<td>Miles Smith</td>
<td>Nordic Semiconductor</td>
</tr>
<tr>
<td>Terry Bourk</td>
<td>Qualcomm</td>
</tr>
<tr>
<td>Brian A. Redding</td>
<td>Qualcomm</td>
</tr>
<tr>
<td>Morteza Rahchamani</td>
<td>TI</td>
</tr>
<tr>
<td>Jason Hillyard</td>
<td>Wicentric</td>
</tr>
<tr>
<td>Magnus Sommansson</td>
<td>Qualcomm</td>
</tr>
</tbody>
</table>