Audio/Video Distribution Transport Protocol (AVDTP)

***Bluetooth*® Test Suite**

- **Revision:** AVDTP.TS.p23
- **Revision Date:** 2025-07-08
- **Prepared By:** BTI
- **Published during TCRL:** TCRL.pkg100

Contents

1 Scope

This Bluetooth document contains the Test Suite Structure (TSS) and test cases to test the implementation of the Bluetooth Audio/Video Distribution Transport Protocol (AVDTP) Specification with the objective to provide a high probability of air interface interoperability between the tested implementation and other manufacturers' Bluetooth devices.

2 References, definitions, and abbreviations

2.1 References

This document incorporates provisions from other publications by dated or undated reference. These references are cited at the appropriate places in the text, and the publications are listed hereinafter. Additional definitions and abbreviations can be found in [5] and [6].

[1] OSI Conformance testing methodology and framework for protocol recommendations for ITU-T applications – General Concepts ITU-T Recommendation X.290 (04/95)

[2] OSI Conformance testing methodology and framework for protocol recommendations for ITU-T applications - Abstract Test Suite Specification ITU-T Recommendation X.291 (04/95)

[3] Specification of the Bluetooth System, Protocols, Audio/Video Distribution Transport Protocol, Versions 1.0 and 1.2

[4] ICS Proforma for Audio/Video Distribution Transport Protocol

[5] Specification of the Bluetooth System, Protocols, Audio/Video Distribution Transport Protocol, Version 1.3

[6] Test Strategy and Terminology Overview

2.2 Definitions

In this Bluetooth document, the definitions from [5] and [6] apply.

2.3 Acronyms and abbreviations

In this Bluetooth document, the definitions, acronyms, and abbreviations from [5] and [6] apply.

Acronyms and abbreviations	Definition
NSEP	Number of SEPs
NSEP_det	Number of SEPs detected

Table 2.1: Acronyms and abbreviations

3 Test Suite Structure (TSS)

3.1 Overview

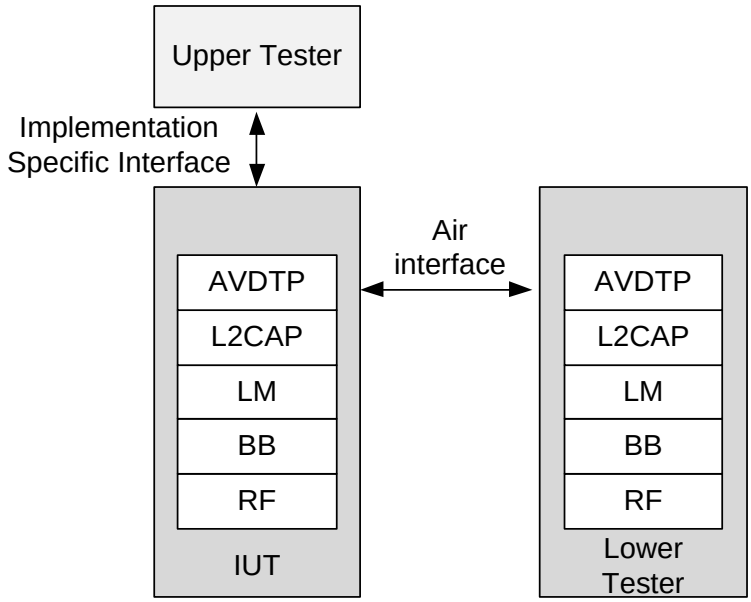

Figure 3.1: AVDTP Conformance Testing Configuration

3.2 Test Strategy

The test objectives are to verify functionality of AVDTP within a Bluetooth Host and enable interoperability between Bluetooth Hosts on different devices. The testing approach covers mandatory and optional requirements in the specification and matches these to the support of the IUT as described in the ICS. Any defined test herein is applicable to the IUT if the ICS logical expression defined in the Test Case Mapping Table (TCMT) evaluates to true.

The test equipment provides an implementation of the Radio Controller and the parts of the Host needed to perform the test cases defined in this Test Suite. A Lower Tester acts as the IUT's peer device and interacts with the IUT over-the-air interface. The configuration, including the IUT, needs to implement similar capabilities to communicate with the test equipment. For some test cases, it is necessary to stimulate the IUT from an Upper Tester. In practice, this could be implemented as a special test interface, a Man Machine Interface (MMI), or another interface supported by the IUT.

The Upper Tester and upper interface (as described in Section 13 of [3] and [5]) are provided for testing purposes only and may be used as a basis for other application specific implementations. Hence, all AVDTP service primitives and parameters (such as AVDT_Discover_Req) referred to in the tests are informational. As a consequence, the expected outcome of a test case does not depend on those service primitives and their (output) parameters, which are given for information only.

This 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 purposes. The test cases presented in the current document intend to set a SEP in all the possible states, but not to test exhaustively all the possible state transitions.

The test cases do not aim at substituting for manufacturer's tests of topics like robustness, behavior in limit cases or even exhaustive generation of all the possible error codes for all the signaling messages or service primitives.

The approach used for testing the invalid behavior is the following:

- Type 1: On the INT side: Verify that the IUT detects the rejection by the ACP of all the types of command messages.
- Type 2: On the ACP side: Verify that the IUT is able to issue reject response messages in answer to all the types of command messages received.
- Type 3: The AVDTP specification document ([3] and [5]) specifies in its Section 6.5 all the possible state transitions for a stream end point.

3.3 Test groups

The following test groups have been defined:

- Signaling procedures
  - Stream management service
  - Security signaling service
  - Signaling message fragmentation service
  - Synchronization service
- Transport procedures
  - Basic transport service
  - Reporting service
  - Recovery service
  - Multiplexing service
  - Robust header compression service

4 Test cases (TC)

4.1 Introduction

4.1.1 Test case identification conventions

Test cases are assigned unique identifiers per the conventions in [6]. The convention used here is: **<spec abbreviation>/<IUT role>/**<class>/**<feat>**/<func>/<subfunc>/<cap>/**<xx>-<nn>-<y>**.

Testing of AVDTP functionality includes tests that refer to multiple roles in combination referred to in the TCMT per the following convention: **<spec abbreviation>/<IUT role>/<AVDTP Secondary role>**/<feat>.

Identifier Abbreviation	**Spec Identifier <spec abbreviation>**
AVDTP	Audio/Video Distribution Transport Protocol
Identifier Abbreviation	**Role Identifier <IUT role>**
SNK	Sink role
SRC	Source role
Identifier Abbreviation	**Secondary Role Identifier <AVDTP Secondary role>**
ACP	Acceptor role
INT	Initiator role
Identifier Abbreviation	**Feature Identifier <feat>**
BM	Basic mode
BTR	Basic Transport Service
EM	Enhanced modes
ESRnn	Errata Service Release derived test where “nn” signifies the ESR release number (e.g., ESR02, ESR04, ESR05)
FRA	Message Fragmentation Service
L2C	L2CAP Configuration
MUX	Multiplexing Service
REC	Recovery Service
REP	Reporting Service
RHC	Robust Header Compression Service
SEC	Security Signaling Service
SIG	Signaling procedure
SMG	Stream Management Service
SYN	Synchronization
TRA	Transport procedure

Table 4.1: AVDTP TC feature naming conventions

4.1.2 Conformance

When conformance is claimed for a particular specification, all capabilities are to be supported in the specified manner. The mandated tests from this Test Suite depend on the capabilities to which conformance is claimed.

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 not excluded by the specification
- That capabilities enabled by the implementations are sustained over durations expected by the use case
- That the implementation gracefully handles any quantity of data expected by the use case
- That in cases where more than one valid interpretation of the specification exists, the implementation complies with at least one interpretation and gracefully handles other interpretations
- That the implementation is immune to attempted security exploits

A single execution of each of the required tests is required to constitute a Pass verdict. However, it is noted that 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 generated by the Bluetooth SIG qualification tool, with 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 erratum request such that the issue may be addressed.

4.1.3 Assumptions

Only one point-to-point configuration is considered.

To execute the AVDTP Test Suite, it is required that the Bluetooth reference stack layers below AVDTP on the IUT are in conformance with their respective specifications.

The IUT has the role of initiator, acceptor, source or sink depending on the test case. The selected role is specified for each test case.

An ACL link has been set up between the IUT and the Lower Tester.

One L2CAP channel is set-up and configured between the IUT and the Lower Tester.

4.1.4 Mapping of error codes and test purposes

The following table provides for each error code that is generated by AVDTP itself (see Sections 8.18.6.2 in [3] and 8.20.6.2 in [5] and the identification of the test purpose of type 2, as defined in Section 4.1.1 of this document) that verifies its implementation. Note that for some tests there are multiple error codes that may be applicable.

Error Mnemonic	Test Purpose
BAD_STATE	AVDTP/SRC/INT/SIG/SMG/BI-16-C AVDTP/SNK/INT/SIG/SMG/BI-16-C AVDTP/SRC/ACP/SIG/SMG/BI-17-C AVDTP/SNK/ACP/SIG/SMG/BI-17-C AVDTP/SRC/INT/SIG/SMG/BI-19-C AVDTP/SRC/ACP/SIG/SMG/BI-20-C AVDTP/SNK/ACP/SIG/SMG/BI-20-C AVDTP/SRC/INT/SIG/SMG/BI-22-C AVDTP/SNK/INT/SIG/SMG/BI-22-C AVDTP/SRC/INT/SIG/SMG/BI-25-C AVDTP/SRC/ACP/SIG/SMG/BI-26-C AVDTP/SNK/ACP/SIG/SMG/BI-26-C
BAD_HEADER_FORMAT	AVDTP/SRC/INT/SIG/SMG/BI-01-C AVDTP/SNK/INT/SIG/SMG/BI-01-C
BAD_LENGTH	AVDTP/SRC/INT/SIG/SMG/BI-04-C AVDTP/SNK/INT/SIG/SMG/BI-04-C AVDTP/SRC/ACP/SIG/SMG/BI-05-C AVDTP/SNK/ACP/SIG/SMG/BI-05-C AVDTP/SRC/INT/SIG/SMG/BI-32-C AVDTP/SNK/INT/SIG/SMG/BI-32-C AVDTP/SRC/ACP/SIG/SMG/BI-33-C AVDTP/SNK/ACP/SIG/SMG/BI-33-C AVDTP/SRC/ACP/TRA/REP/BI-02-C AVDTP/SNK/ACP/TRA/REP/BI-02-C
BAD_ACP_SEID	AVDTP/SRC/INT/SIG/SMG/BI-10-C AVDTP/SNK/INT/SIG/SMG/BI-10-C AVDTP/SRC/ACP/SIG/SMG/BI-11-C AVDTP/SNK/ACP/SIG/SMG/BI-11-C AVDTP/SRC/ACP/SIG/SMG/BI-14-C AVDTP/SNK/ACP/SIG/SMG/BI-14-C AVDTP/SRC/ACP/SIG/SMG/BI-23-C AVDTP/SNK/ACP/SIG/SMG/BI-23-C
SEP_IN_USE	AVDTP/SRC/INT/SIG/SMG/BI-07-C AVDTP/SNK/INT/SIG/SMG/BI-07-C AVDTP/SRC/ACP/SIG/SMG/BI-08-C AVDTP/SNK/ACP/SIG/SMG/BI-08-C
SEP_NOT_IN_USE	AVDTP/SRC/INT/SIG/SMG/BI-13-C AVDTP/SNK/INT/SIG/SMG/BI-13-C AVDTP/SRC/INT/SIG/SMG/BI-23-C AVDTP/SNK/INT/SIG/SMG/BI-23-C AVDTP/SRC/ACP/SIG/SMG/BI-14-C AVDTP/SNK/ACP/SIG/SMG/BI-14-C

Error Mnemonic	Test Purpose
BAD_SERV_CATEGORY	AVDTP/SRC/INT/SIG/SMG/BI-13-C AVDTP/SNK/INT/SIG/SMG/BI-13-C AVDTP/SRC/INT/SIG/SMG/BI-23-C AVDTP/SNK/INT/SIG/SMG/BI-23-C AVDTP/SRC/ACP/SIG/SMG/BI-14-C AVDTP/SNK/ACP/SIG/SMG/BI-14-C
NOT_SUPPORTED_COMMAND	None
UNSUPPORTED_CONFIGURATION	None
INVALID_CAPABILITIES	AVDTP/SRC/ACP/SIG/SMG/BI-14-C AVDTP/SNK/ACP/SIG/SMG/BI-14-C
BAD_REPORT_TYPE – Error Code not defined by the AVDTP Specification. In case the format of the Reporting Service capability is not correct, the BAD_LENGTH or BAD_PAYLOAD_FORMAT error code is used.	None
BAD_RECOVERY_TYPE	AVDTP/SRC/ACP/TRA/REC/BI-01-C AVDTP/SNK/ACP/TRA/REC/BI-01-C
BAD_MEDIA_TRANSPORT_FORMAT	AVDTP/SRC/ACP/TRA/BTR/BI-01-C AVDTP/SNK/ACP/TRA/BTR/BI-01-C
BAD_PAYLOAD_FORMAT	AVDTP/SRC/ACP/TRA/REP/BI-02-C AVDTP/SNK/ACP/TRA/REP/BI-02-C
BAD_RECOVERY_FORMAT	AVDTP/SRC/ACP/TRA/REC/BI-01-C AVDTP/SNK/ACP/TRA/REC/BI-01-C
BAD_ROHC_FORMAT	AVDTP/SRC/ACP/TRA/RHC/BI-01-C AVDTP/SNK/ACP/TRA/RHC/BI-01-C
BAD_CP_FORMAT	AVDTP/SRC/ACP/SIG/SEC/BI-01-C AVDTP/SNK/ACP/SIG/SEC/BI-01-C
BAD_MULTIPLEXING_FORMAT	AVDTP/SRC/ACP/TRA/MUX/BI-01-C AVDTP/SNK/ACP/TRA/MUX/BI-01-C

Table 4.2: Error Code Mapping

4.1.5 Pass/Fail verdict conventions

Each test case has an Expected Outcome section. The IUT is granted the Pass verdict when all the detailed pass criteria conditions within the Expected Outcome section are met.

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 as one of the pass criteria conditions cannot be met. If this occurs, the outcome of the test is a Fail verdict.

4.2 Setup preambles

4.2.1 Establishing an L2CAP channel for signaling

- Preamble Purpose

  Establish one L2CAP channel for signaling between the IUT and the Lower Tester.

- Reference

  [5] 5.4.6

- Preamble Procedure

  1. One ACL link is established between the IUT and the Lower Tester.
  2. One L2CAP channel for signaling is established between the IUT and the Lower Tester.

4.3 Signaling

Verify the correct implementation of the Signaling procedures.

4.3.1 Stream Management Service

Verify that the following procedures are implemented according to their specification in AVDTP:

- Stream end-point discovery
- Get capabilities
- Stream configuration and reconfiguration
- Stream establishment
- Stream start, suspend, and release

4.3.1.1 Stream Discover command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid Stream Discover command and report the replied SEPs and media types.

- Reference

  [3] 6.4, 6.6, 8.6, 13.1

  [5] 6.4, 6.6, 8.6, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-05-C [Stream Discover command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-05-C [Stream Discover command – INT]

Table 4.3: Stream Discover command – INT test cases

- Test Procedure

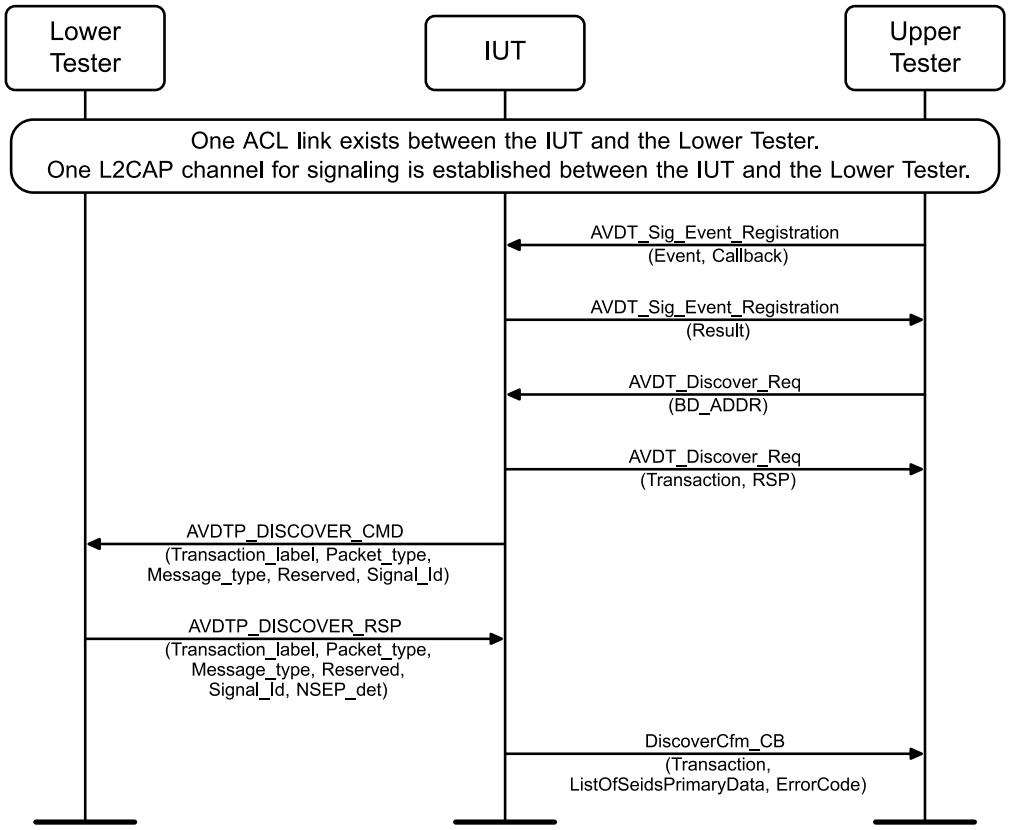

Figure 4.1: Stream Discover command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive with Event = AVDT_Discover_Cfm and Callback = DiscoverCfm_CB to the IUT.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Discover_Req service primitive with BD_ADDR = $BD\_ADDR_{LowerTester}$ to the IUT.
4. The IUT responds to the Upper Tester with AVDT_Discover_Req with Transaction and RSP.
5. The IUT sends an AVDTP_DISCOVER_CMD command to the Lower Tester with the Transaction_Label, Packet_type, Message_type, Signal_Id, and NSEP_det set to valid values.
6. The Lower Tester sends an AVDTP_DISCOVER_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), Signal_Id = 0x01 (AVDTP_DISCOVER), and NSEP_det = valid SEID values.
7. The IUT sends a DiscoverCfm_CB event to the Upper Tester with valid Transaction, ListOfSeidsPrimaryData, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

In Step 4, the output parameters of the AVDT_Discover_Req service primitive sent by the IUT to the Upper Tester are:

- Transaction = $Transaction_{IUT}$ = any value.
- RSP = REQUEST_ACCEPTED

In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{IUT}$
- Message_type = 00 (command)
- Signal_Id = 0x01 (AVDTP_DISCOVER)

In Step 7, the IUT invokes the DiscoverCfm_CB callback function with the following parameters:

- Transaction = $Transaction_{IUT}$
- ListOfSeidsPrimaryData = the values provided with AVDTP_DISCOVER_RSP
- ErrorCode = SUCCESS

4.3.1.2 Stream Discover command – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid Stream Discover command and replies the returned list of SEPs and media types.

- Reference

  [3] 6.4, 6.6, 8.6, 13.1

  [5] 6.4, 6.6, 8.6, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-06-C [Stream Discover command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-06-C [Stream Discover command – ACP]

Table 4.4: Stream Discover command – ACP test cases

- Test Procedure

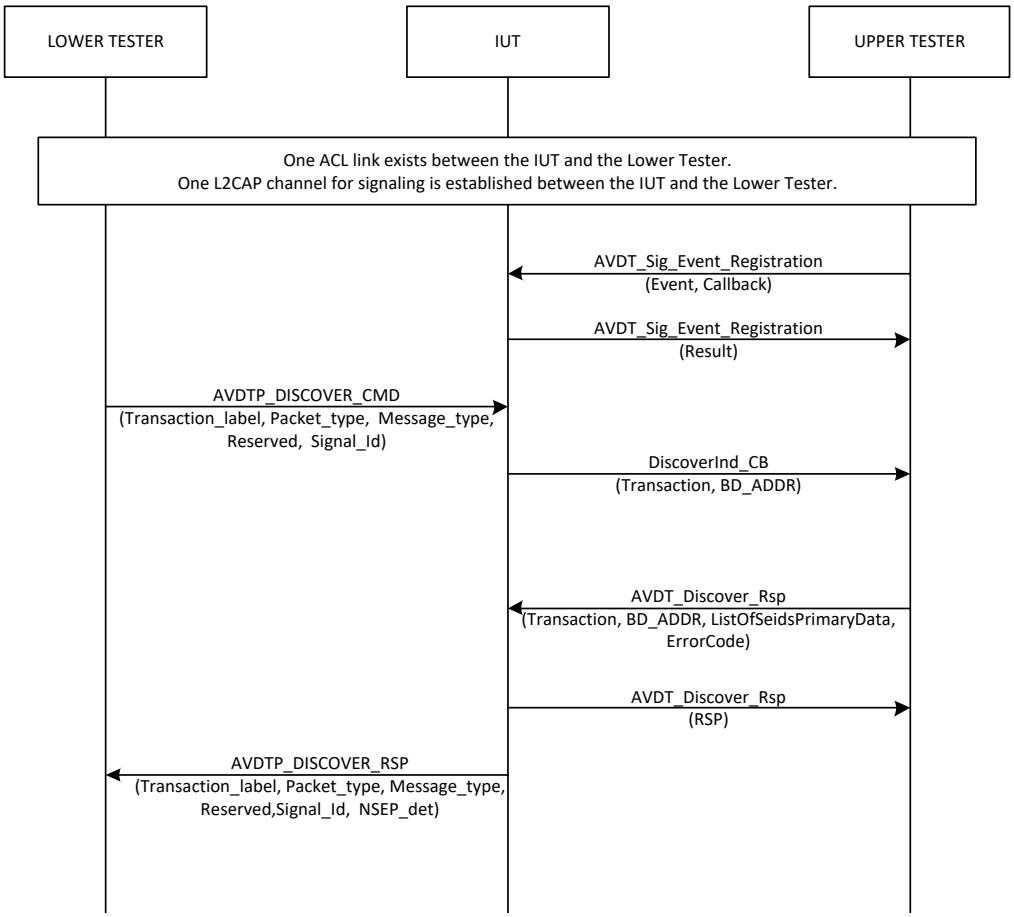

Figure 4.2: Stream Discover command – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Discover_Ind and Callback = DiscoverInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_DISCOVER_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x01 (AVDTP_DISCOVER).
4. The IUT sends a DiscoverInd_CB event to the Upper Tester with Transaction and BD_ADDR.
5. The Upper Tester sends the AVDT_DiscoverRsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, ListOfSeidsPrimaryData contains valid SEID values, and ErrorCode = SUCCESS.
6. The IUT responds to the Upper Tester with AVDT_Discover_Rsp with RSP.
7. The IUT sends an AVDTP_DISCOVER_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Signal_Id, and NSEP_det.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

In Step 4, the IUT invokes the DiscoverInd_CB callback function with the following parameters:

- Transaction = $Transaction_{LowerTester}$
- BD_ADDR = $BD\_ADDR_{LowerTester}$

In Step 6, the output parameter of the AVDT_Discover_Rsp service primitive sent from the IUT to the Upper Tester is:

- RSP = REQUEST_ACCEPTED

In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Message_type = 10 (response accept)
- Signal_Id = 0x01 (AVDTP_DISCOVER)
- NSEP_det = the values provided with AVDT_Discover_Rsp

4.3.1.3 Get Capabilities command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid query for remote SEP capabilities and reports the replied ones.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-07-C [Get Capabilities command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-07-C [Get Capabilities command – INT]

Table 4.5: Get Capabilities command – INT test cases

- Test Procedure

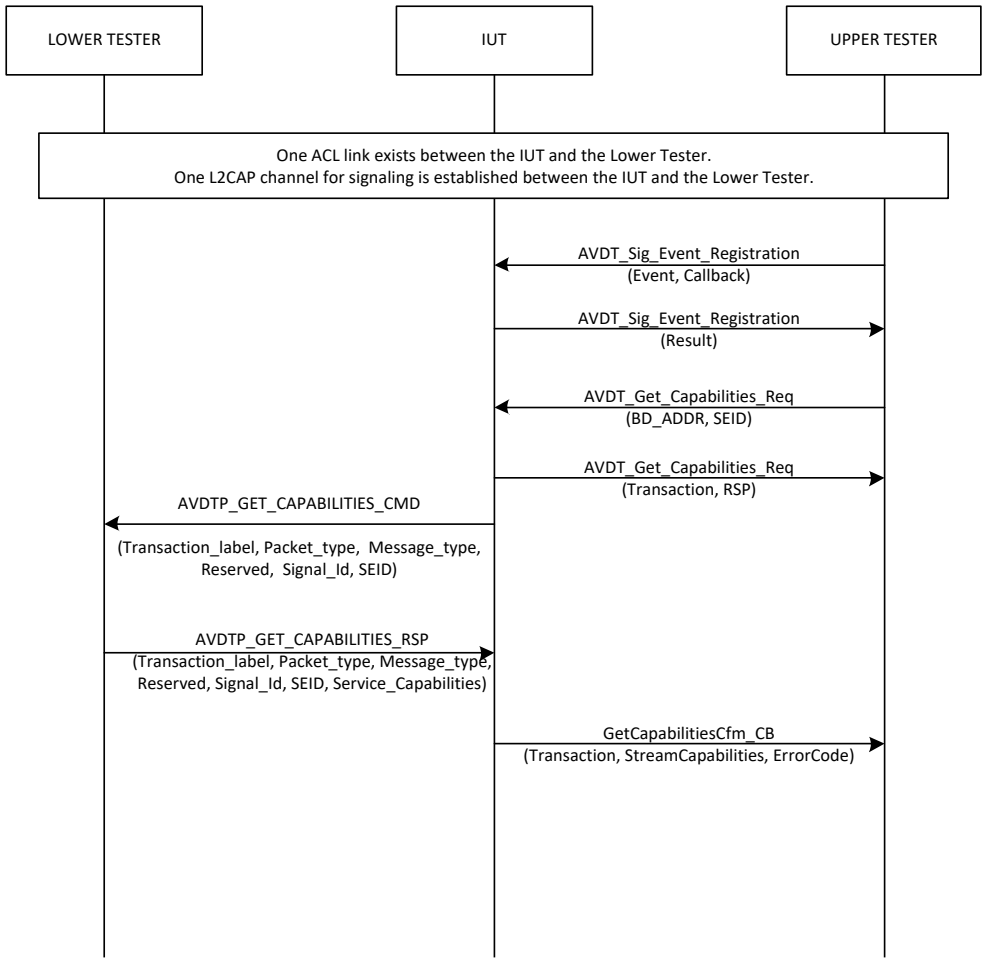

Figure 4.3: Get Capabilities command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_Capabilities_Cfm and Callback = GetCapabilitiesCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Get_Capabilities_Req service primitive with BD_ADDR = $BD\_ADDR_{LowerTester}$ and SEID = $SEID_{LowerTester}$ = any value.
4. The IUT responds to the Upper Tester with AVDT_Discover_Req with Transaction and RSP with valid values.
5. The IUT sends an AVDTP_GET_CAPABILITIES_CMD command to the Lower Tester with Transaction_Label, Packet_type, Message_type, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_GET_CAPABILITIES_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Service_Capabilities with valid capabilities values for each of the following types: Media transport, reporting, recovery, video codec or audio codec or multimedia codec, content protection, header compression, multiplexing.
7. The IUT sends a GetCapabilitiesCfm_CB event to the Upper Tester with valid Transaction, StreamCapabilities, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Get_Capabilities_Req service primitive to the Upper Tester with the following values:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - SEID = $SEID_{LowerTester}$

  In Step 7,the IUT invokes the GetCapabilitiesCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - Stream_capabilities = the values provided with AVDTP_GET_CAPABILITIES_RSP
  - ErrorCode = SUCCESS

4.3.1.4 Get Capabilities response – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid query for remote SEP capabilities and replies the returned capabilities.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-08-C [Get Capabilities response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-08-C [Get Capabilities response – ACP]

Table 4.6: Get Capabilities response – ACP test cases

- Test Procedure

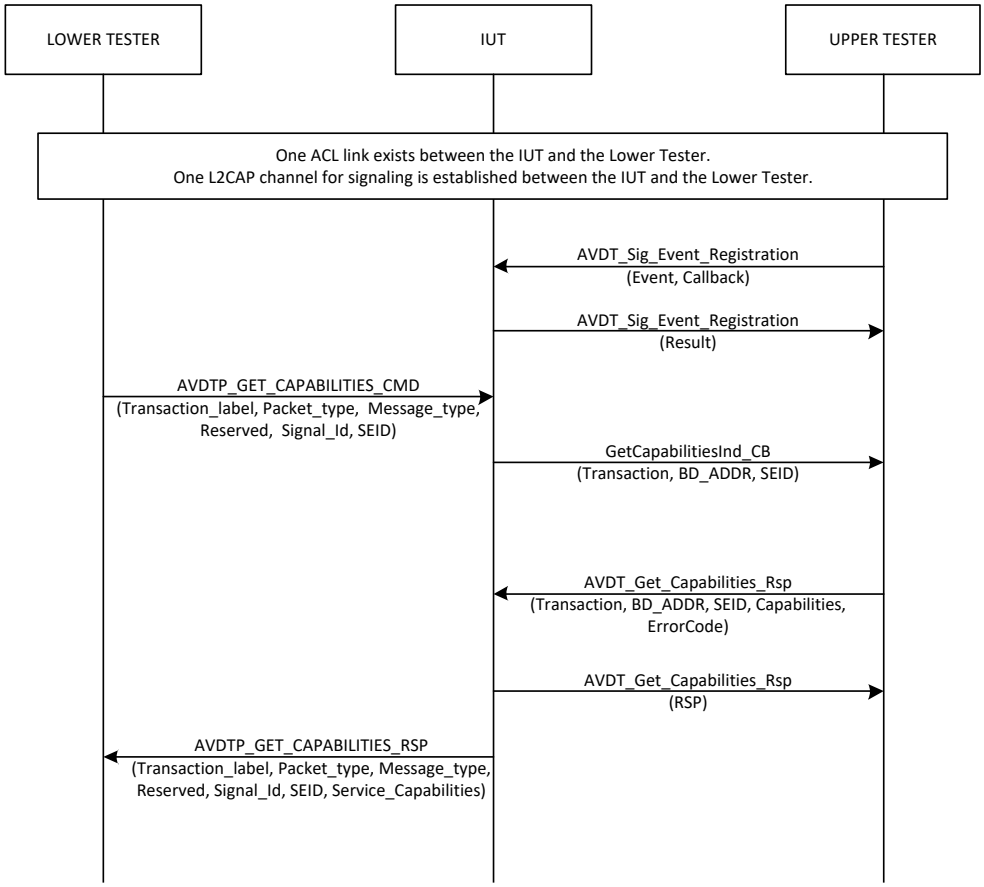

Figure 4.4: Get Capabilities response – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_Capabilities_Ind and Callback = GetCapabilitiesInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = $SEID_{IUT}$ = any value.
4. The IUT sends a GetCapabilitiesInd_CB event to the Upper Tester with valid Transaction, BD_ADDR, and SEID.
5. The Upper Tester sends an AVDT_Get_Capabilities_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, SEID = $SEID_{IUT}$, Capabilities = valid capabilities values for each category that the addressed SEP supports, and ErrorCode = SUCCESS.
6. The IUT responds to the Upper Tester with an AVDT_Get_Capabilities_Rsp response with RSP set to a valid value.
7. The IUT sends an AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with Transaction_Label, Packet_type, Message_type, Signal_Id, SEID and Service_Capabilities set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent from the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - BD_ADDR = $BD\_ADDR_{LowerTester}$
  - SEID = $SEID_{IUT}$

  In Step 6, the output parameter of the AVDT_Get_Capabilities_Rsp response sent from the IUT to the Upper Tester is:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - SEID = $SEID_{IUT}$
  - Service_Capabilities = the values provided with AVDT_Get_Capabilities_Rsp

4.3.1.5 Set Configuration command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid Set Configuration command for remote SEP and reports the replied confirmation.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEP is in the Idle state.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-09-C [Set Configuration command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-09-C [Set Configuration command – INT]

Table 4.7: Set Configuration command – INT test cases

- Test Procedure

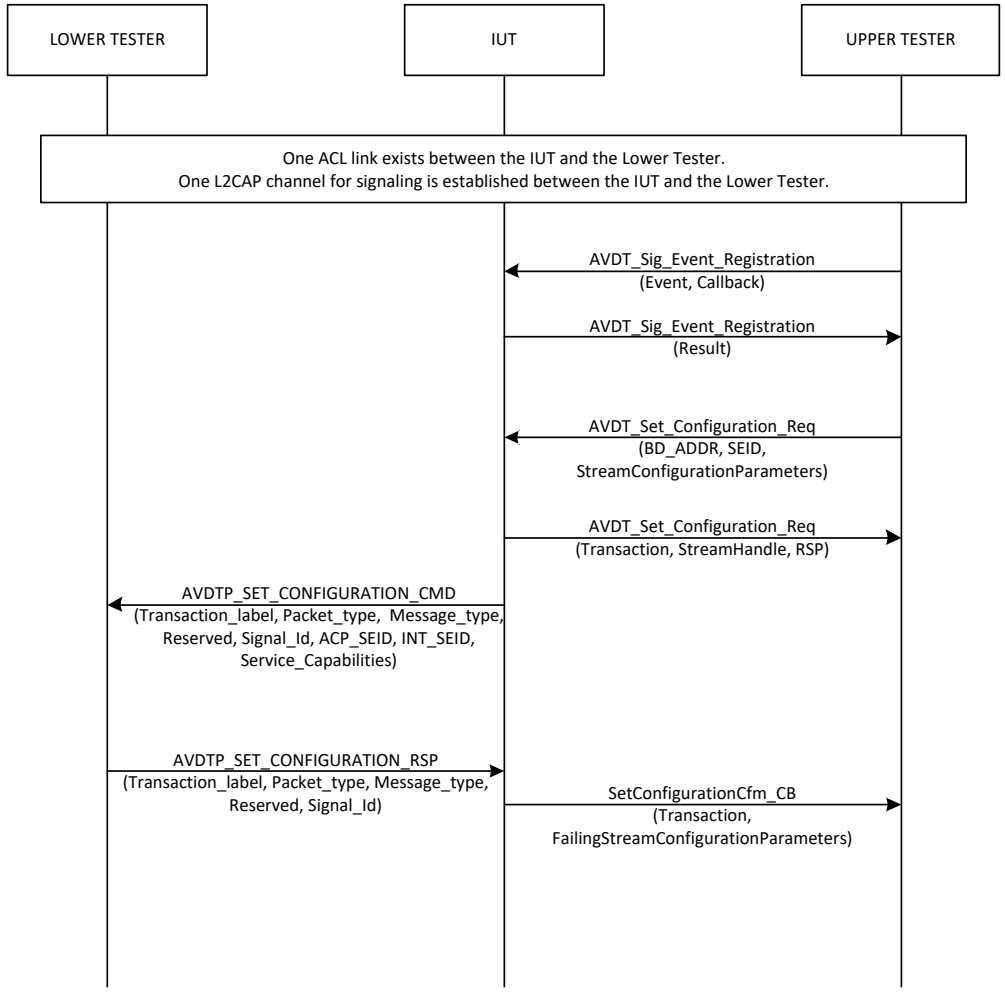

Figure 4.5: Set Configuration command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive with Event = AVDT_Set_Configuration_Cfm and Callback = SetConfigurationCfm_CB to the IUT.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Set_Configuration_Req service primitive with BD_ADDR = $BD\_ADDR_{LowerTester}$,SEID = $SEID_{LowerTester}$ = any value, and StreamConfigurationParameters = valid configurations for (a subset of) capabilities supported by the addressed Stream End Point. Except for Content protection, at most one Service Capabilities information element per category may appear in the payload of the message.
4. The IUT responds to the Upper Tester with AVDT_Set_Configuration_Req with Transaction, StreamHandle, and RSP set to valid values.
5. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with the Transaction_label, Packet_type, Message_type, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities set to valid values.
6. The Lower Tester sends an AVDTP_SET_CONFIGURATION_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION).
7. The IUT sends a SetConfigurationCfm_CB event to the Upper Tester with valid Transaction, and FailingStreamConfigurationParameters.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the output parameters of the AVDT_Set_Configuration_Req service primitive are:

  - Transaction = $Transaction_{IUT}$ = any value
  - StreamHandle = SH_{IUT} = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - ACP_SEID = $SEID_{LowerTester}$
  - INT_SEID = any value

  In Step 7, the IUT invokes the SetConfigurationCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - FailingStreamConfigurationParameters = Empty list

4.3.1.6 Set Configuration response – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid Set Configuration command for remote SEP, and configures the SEP as requested and replies the returned confirmation.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEP is in the Idle state.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-10-C [Set Configuration response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-10-C [Set Configuration response – ACP]

Table 4.8: Set Configuration response – ACP test cases

- Test Procedure

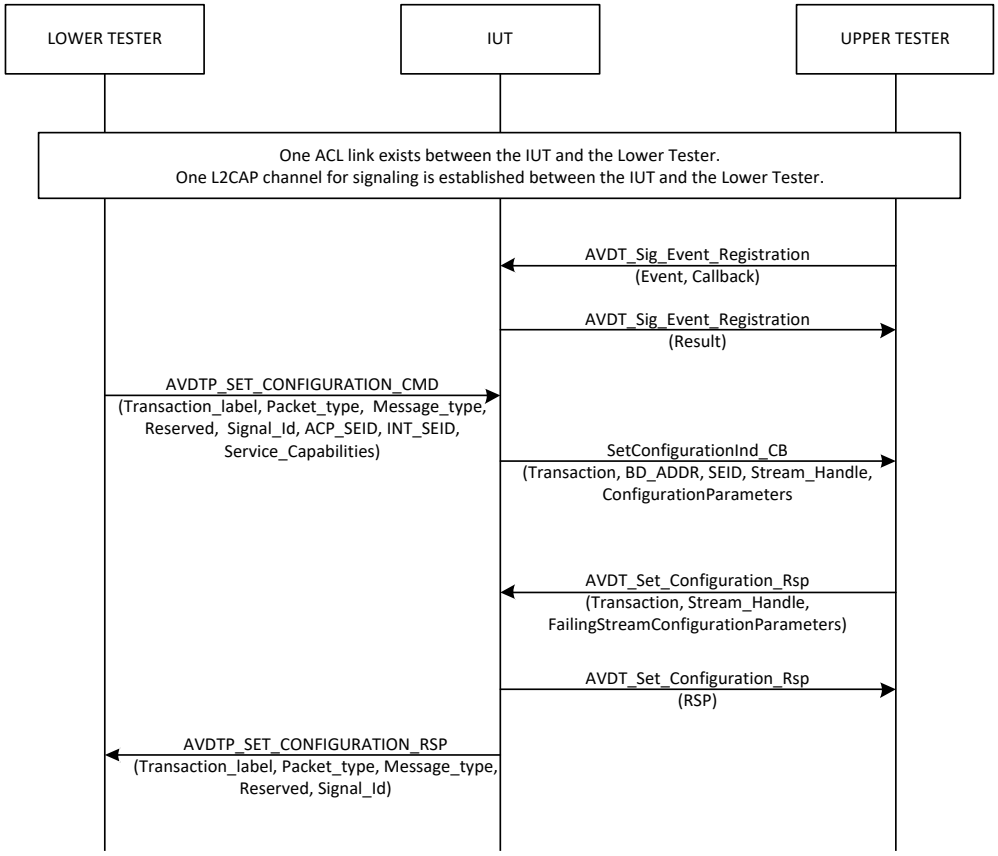

Figure 4.6: Set Configuration response – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive with Event = AVDT_Set_Configuration_Ind and Callback = SetConfigurationInd_CB to the IUT.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any value, and Service_Capabilities = valid capabilities values for each category that the addressed SEP supports.
4. The IUT sends a SetConfigurationInd_CB event to the Upper Tester with Transaction, BD_ADDR, SEID, Stream_Handle, and ConfigurationParameters set to valid values.
5. The Upper Tester sends an AVDT_Set_Configuration_Rsp response to the IUT with Transaction = $Transaction_{LowerTester}$, Stream_Handle = SH_{IUT}, and FailingStreamConfigurationParameters = Empty list.
6. The IUT sends an AVDT_Set_Configuration_Rsp response to the Upper Tester with a valid RSP.
7. The IUT sends an AVDTP_SET_CONFIGURATION_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

In Step 4, the SetConfigurationInd_CB is invoked with the following parameters:

- Transaction = $Transaction_{LowerTester}$
- BD_ADDR = $BD\_ADDR_{LowerTester}$
- SEID = $SEID_{IUT}$
- Stream_Handle = SH_{IUT} = any value
- ConfigurationParameters = the configuration values provided with AVDTP_SET_CONFIGURATION_CMD

In Step 6, the output parameters of the AVDT_Set_Configuration_Rsp service primitive are:

- RSP = REQUEST_ACCEPTED

In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Message_type = 10 (response accept)
- Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)

4.3.1.7 Get Configuration command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid Get Configuration command for remote SEP and report the replied configuration.

- Reference

  [3] 6.4, 6.9, 8.9, 13.1

  [5] 6.4, 6.10, 8.10, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT} (see Section 4.3.1.5, Set Configuration command – INT).

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-11-C [Get Configuration command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-11-C [Get Configuration command – INT]

Table 4.9: Get Configuration command – INT test cases

- Test Procedure

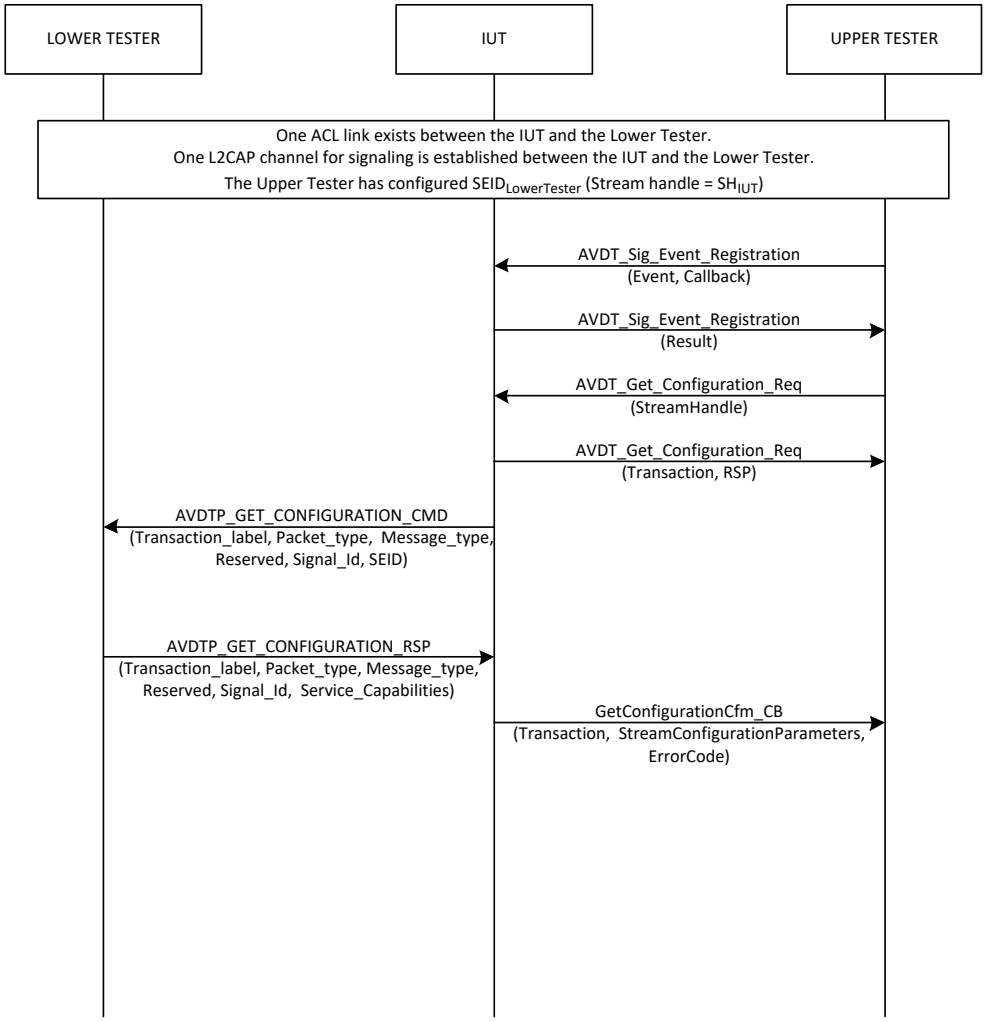

Figure 4.7: Get Configuration command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_Configuration_Cfm and Callback = GetConfigurationCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Get_Configuration_Req service primitive with StreamHandle = SH_{IUT} = the value corresponding to $SEID_{LowerTester}$.
4. The IUT responds to the Upper Tester with AVDT_Get_Configuration_Req with valid Transaction and RSP.
5. The IUT sends an AVDTP_GET_CONFIGURATION_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_GET_CONFIGURATION_RSP response with Transaction_label = TransactionIUT, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION), and Service_Capabilities = valid capabilities values for each category that the addressed SEP supports.
7. The IUT sends a GetConfigurationCfm_CB event to the Upper Tester with valid Transaction, StreamConfigurationParameters, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the output parameters of the AVDT_Get_Configuration_Req service primitive are:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION)
  - SEID = $SEID_{LowerTester}$

  In Step 7, the IUT invokes GetConfigurationCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - StreamConfigurationParameters = the values provided with AVDTP_GET_CONFIGURATION_RSP
  - ErrorCode = SUCCESS

4.3.1.8 Get Configuration response – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid Get Configuration command for remote SEP and replies the returned configuration.

- Reference

  [3] 6.4, 6.9, 8.9, 13.1

  [5] 6.4, 6.10, 8.10, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (see Section 4.3.1.6, Set Configuration response – ACP).

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-12-C [Get Configuration response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-12-C [Get Configuration response – ACP]

Table 4.10: Get Configuration response – ACP test cases

- Test Procedure

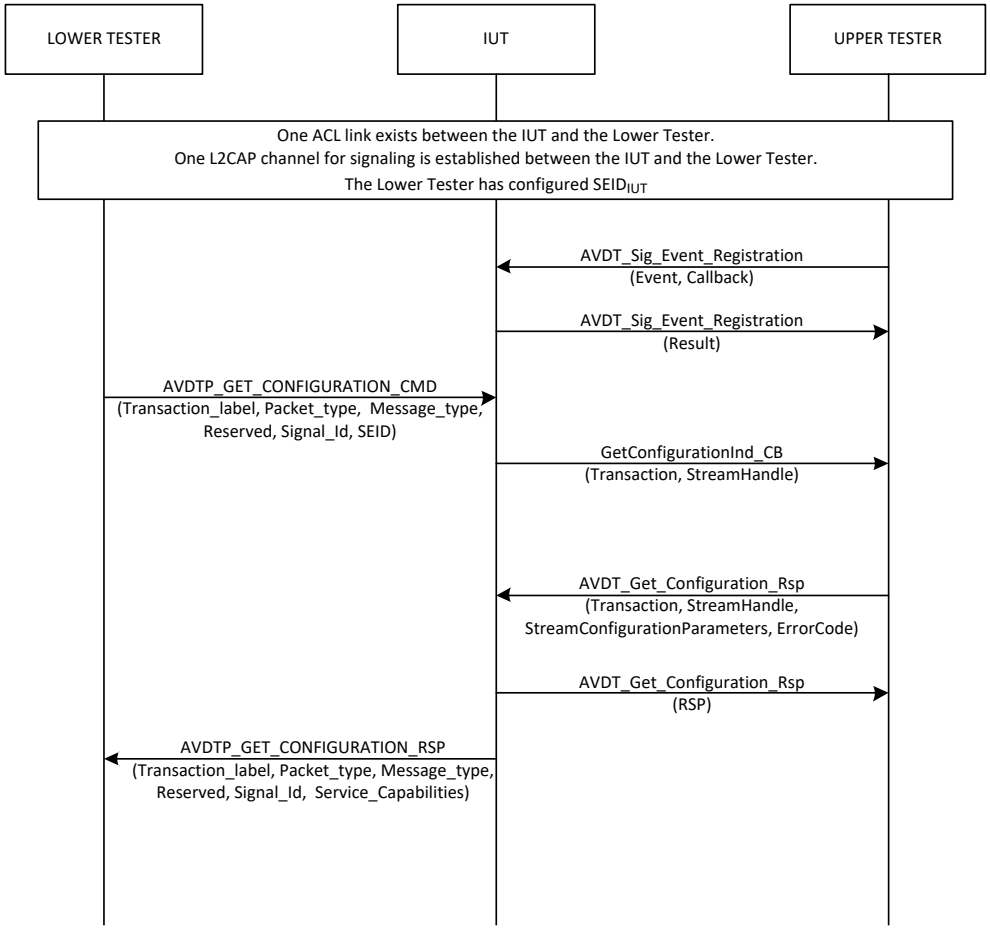

Figure 4.8: Get Configuration response – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_Configuration_Ind and Callback = GetConfigurationInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_GET_CONFIGURATION_CMD command with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION), and SEID = $SEID_{IUT}$ = any value.
4. The IUT sends a GetConfigurationInd_CB event to the Upper Tester with Transaction, and StreamHandle set to valid values.
5. The Upper Tester sends an AVDT_Get_Configuration_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value corresponding to $SEID_{IUT}$, Service_Capabilities = valid capabilities values for each category that the addressed SEP supports, and ErrorCode = SUCCESS.
6. The IUT responds to the Upper Tester with AVDT_Get_Configuration_Rsp and RSP.
7. The IUT sends an AVDTP_GET_CONFIGURATION_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Service_Capabilities set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the GetConfigurationInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - StreamHandle = SH_{IUT}

  In Step 6, the output parameters of the AVDT_Get_Configuration_Rsp service primitive are:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION)
  - Service_Capabilities = the values provided with AVDT_Get_Configuration_Rsp

4.3.1.9 Reconfigure command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid Reconfigure command for remote SEP and report the replied confirmation.

- Reference

  [3] 6.4, 6.14, 8.10, 13.1

  [5] 6.4, 6.15, 8.11, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured and is in the state given by SEP State in Table 4.11.

- Test Case Configuration

Test Case	SEP State
AVDTP/SRC/INT/SIG/SMG/BV-13-C [Reconfigure command following Suspend – INT]	Configured, opened, started, and suspended
AVDTP/SNK/INT/SIG/SMG/BV-13-C [Reconfigure command following Suspend – INT]	Configured, opened, started, and suspended
AVDTP/SRC/INT/SIG/SMG/BV-33-C [Reconfigure command following Open – INT]	Configured, opened
AVDTP/SNK/INT/SIG/SMG/BV-33-C [Reconfigure command following Open – INT]	Configured, opened

Table 4.11: Reconfigure command – INT test cases

- Test Procedure

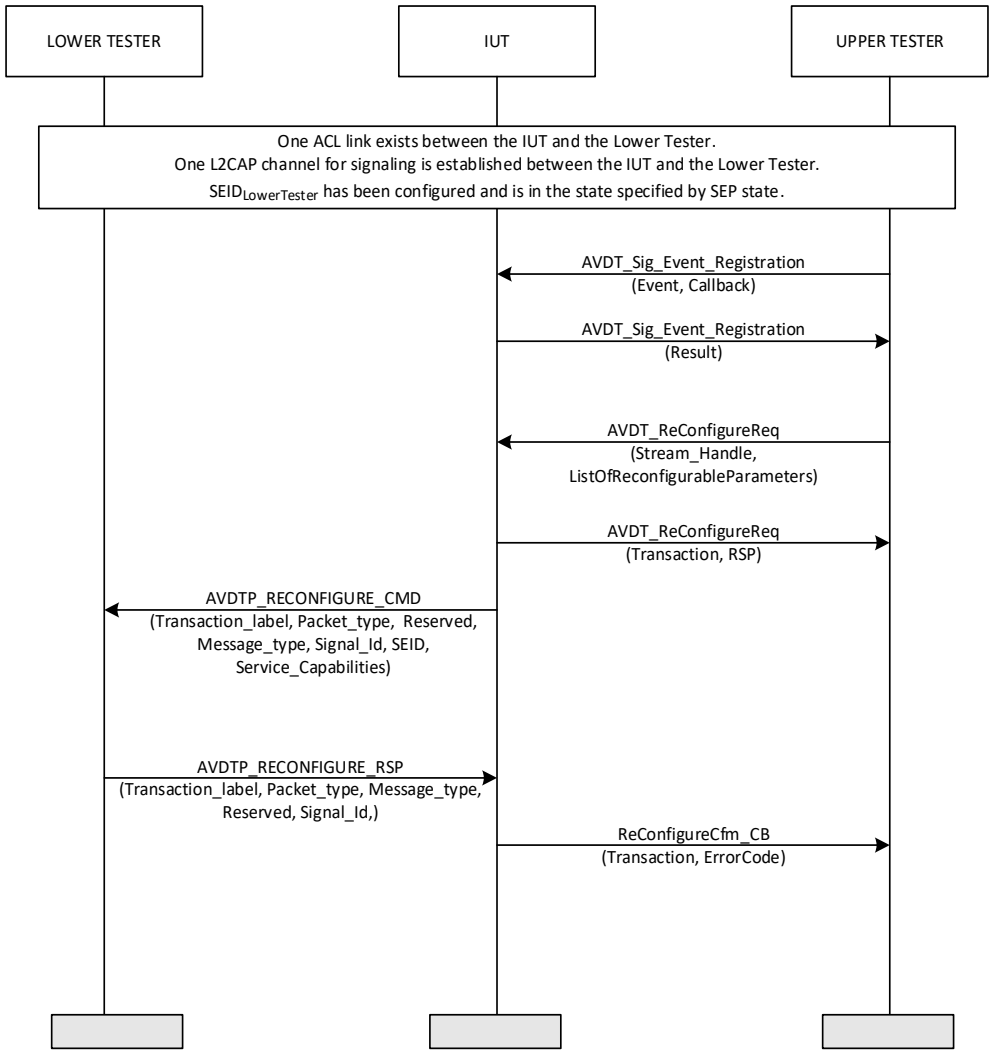

Figure 4.9: Reconfigure command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_ReConfigure_Cfm and Callback = ReConfigureCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_ReConfigure_Req service primitive with Stream_Handle = SH_{IUT} = the value returned when configuring SEP, and ListOfReconfigurableParameters = valid parameters.
4. The IUT sends an AVDT_ReConfigureReq with Transaction and RSP with valid values.
5. The IUT sends an AVDTP_RECONFIGURE_CMD command to the Lower Tester with Transaction_label, Packet_type, Reserved, Message_type, Signal_Id, SEID, and Service_Capabilities set to valid values.

6. The Lower Tester sends an AVDTP_RECONFIGURE_RSP response to the IUT with Transaction_label =$Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x05 (AVDTP_RECONFIGURE).
7. The IUT sends a ReConfigureCfm_CB event to the Upper Tester with Transaction, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the output parameters of the AVDT_ReConfigure_Req service primitive are:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)
  - SEID = $SEID_{LowerTester}$
  - Service_Capabilities = the reconfigurable parameters provided in AVDT_ReConfigure_Req

  In Step 7, the IUT invokes the ReConfigureCfm_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{IUT}$
  - ErrorCode = SUCCESS

4.3.1.10 Reconfigure response – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid Reconfigure command for remote SEP, reconfigures the SEP as requested and replies the returned confirmation.

- Reference

  [3] 6.4, 6.14, 8.10, 13.1

  [5] 6.5, 6.15, 8.11, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{IUT}$ has been configured and is in the state given by SEP State in Table 4.12.

- Test Case Configuration

Test Case	SEP State
AVDTP/SRC/ACP/SIG/SMG/BV-14-C [Reconfigure response following Suspend – ACP]	Configured, opened, started, and suspended by the Lower Tester
AVDTP/SNK/ACP/SIG/SMG/BV-14-C [Reconfigure response following Suspend – ACP]	Configured, opened, started, and suspended by the Lower Tester

Test Case	SEP State
AVDTP/SRC/ACP/SIG/SMG/BV-34-C [Reconfigure response following Open – ACP]	Configured, opened
AVDTP/SNK/ACP/SIG/SMG/BV-34-C [Reconfigure response following Open – ACP]	Configured, opened

Table 4.12: Reconfigure response – ACP test cases

- Test Procedure

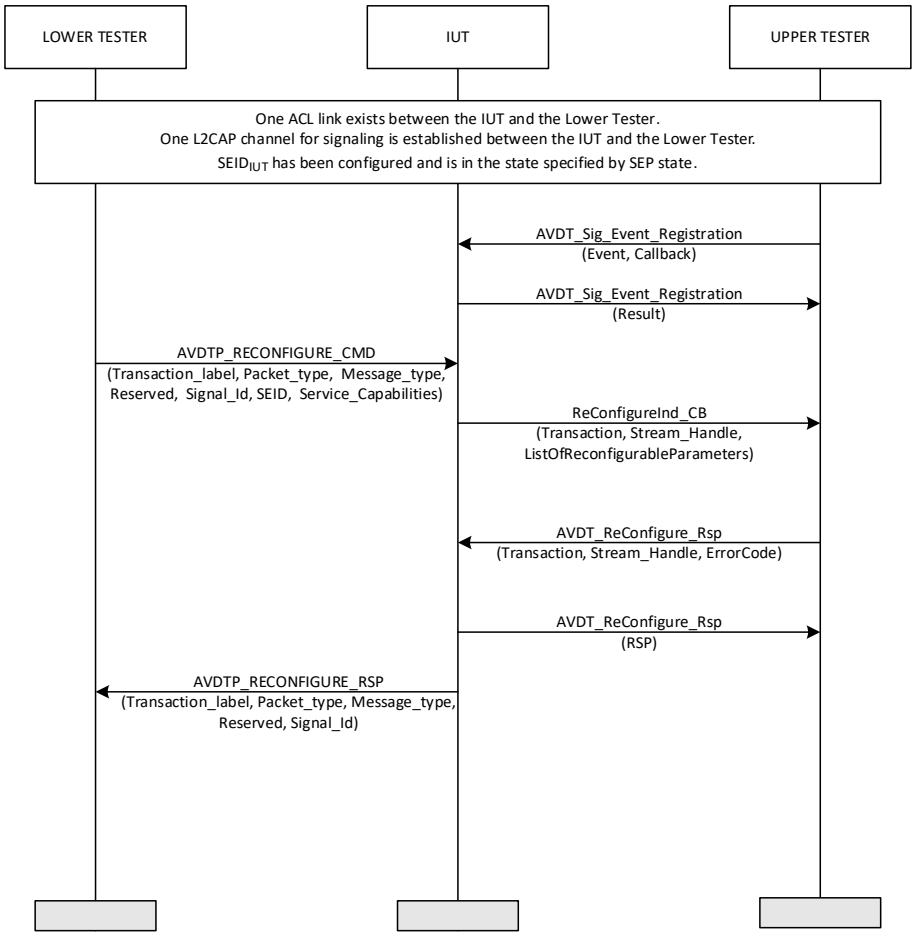

Figure 4.10: Reconfigure response – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_ReConfigure_Ind and Callback = ReConfigureInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_RECONFIGURE_CMD command with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x05 (AVDTP_RECONFIGURE), SEID = $SEID_{IUT}$, and Service_Capabilities = valid configurations.
4. The IUT sends a ReConfigureInd_CB event to the Upper Tester with Transaction, Stream_Handle, and ListOfReconfigurableParameters set to valid values.
5. The Upper Tester sends an AVDT_ReConfigureRsp service primitive to the IUT with Transaction = TransactionLowerTester, Stream_Handle = SHIUT = the value corresponding to SEIDIUT, and ErrorCode = SUCCESS.

6. The IUT sends an AVDT_ReConfigure_Rsp response to the Upper Tester with a valid RSP.
7. The IUT sends an AVDTP_RECONFIGURE_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the ReConfigureInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - Stream_Handle = SH_{IUT} = the value provided when $SEID_{IUT}$ was configured for the first time.
  - ListOfReconfigurableParameters = the parameters provided with AVDTP_RECONFIGURE_CMD

  In Step 6, the IUT sends the AVDT_ReConfigure_Rsp service primitive to the Upper Tester with the following parameters:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)

4.3.1.11 Open command – INT

- Test Purpose

  Verify that the IUT (INT) is able to establish a stream connection to a previously configured remote SEP by establishing one L2CAP channel (AVDTP_OPEN_CMD), and reports the related confirmation (no reporting, no recovery).

- Reference

  [3] 6.4, 6.10, 8.11, 13.1

  [5] 6.5, 6.11, 8.12, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) such as multiplexing is not selected, reporting is not selected and recovery is not selected (see Section 4.3.1.5, Set Configuration command – INT).

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-15-C [Open command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-15-C [Open command – INT]

Table 4.13: Open command – INT test cases

- Test Procedure

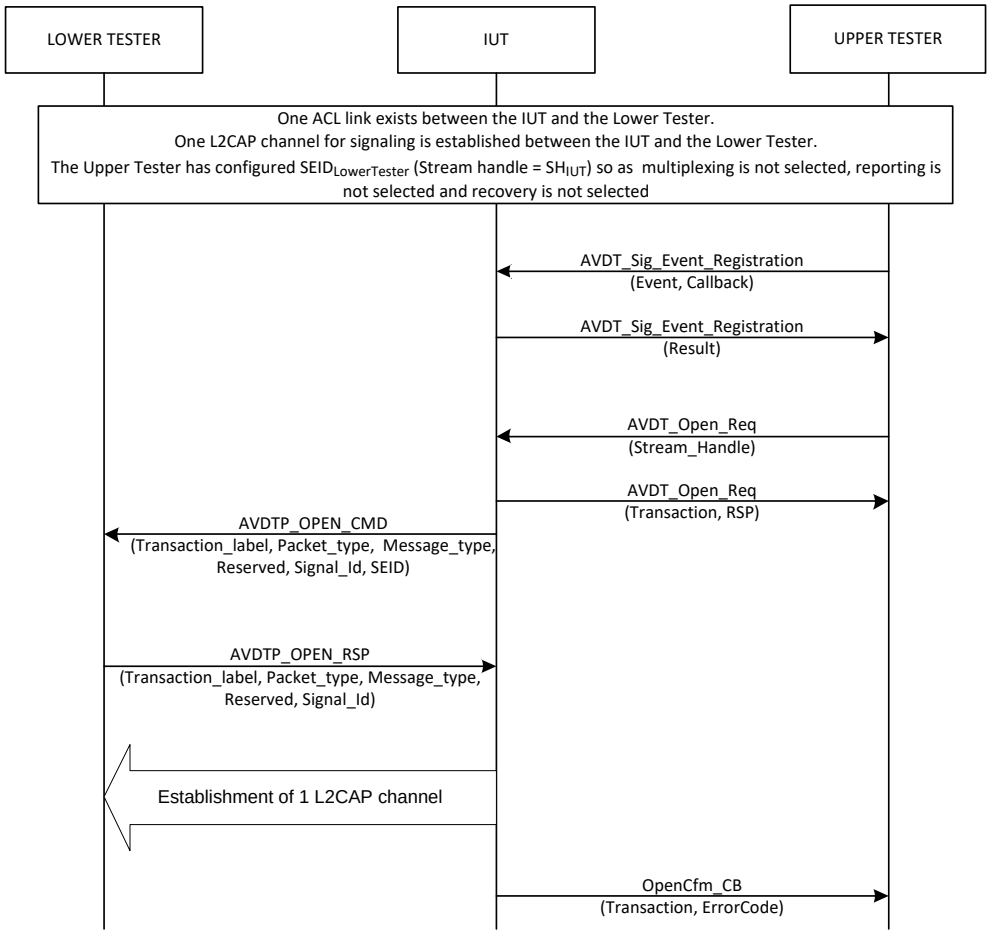

Figure 4.11: Open command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Open_Cfm and Callback = OpenCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Open_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring $SEID_{LowerTester}$.
4. The IUT responds to the Upper Tester with AVDT_Open_Req with Transaction and RSP set to valid values.
5. The IUT sends an AVDTP_OPEN_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_OPEN_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x06 (AVDTP_OPEN).
7. A single L2CAP channel is established between the IUT and the Lower Tester. The Lower Tester provides positive responses to the L2CAP_ConnectReq and L2CAP_ConfigReq messages issued by the IUT. The Lower Tester issues valid L2CAP_ConfigReq messages further to each such message issued by the IUT.
8. The IUT sends an OpenCfm_CB event to the Upper Tester with Transaction and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Open_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x06 (AVDTP_OPEN)
  - SEID = $SEID_{LowerTester}$ = the value used for stream end-point configuration.

  In Step 7, the IUT establishes one L2CAP channel (it issues L2CAP_ConnectReq followed by L2CAP_ConfigReq with valid parameters).

  In Step 8, the IUT invokes the OpenCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = SUCCESS

4.3.1.12 Open command – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of valid AVDTP_OPEN_CMD for transport sessions associated with a SEP and replies the returned confirmation.

- Reference

  [3] 6.4, 6.10, 8.11, 13.1

  [5] 6.4, 6.11, 8.12, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (see Section 4.3.1.6, Set Configuration response – ACP).

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-16-C [Open command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-16-C [Open command – ACP]

Table 4.14: Open command – ACP test cases

- Test Procedure

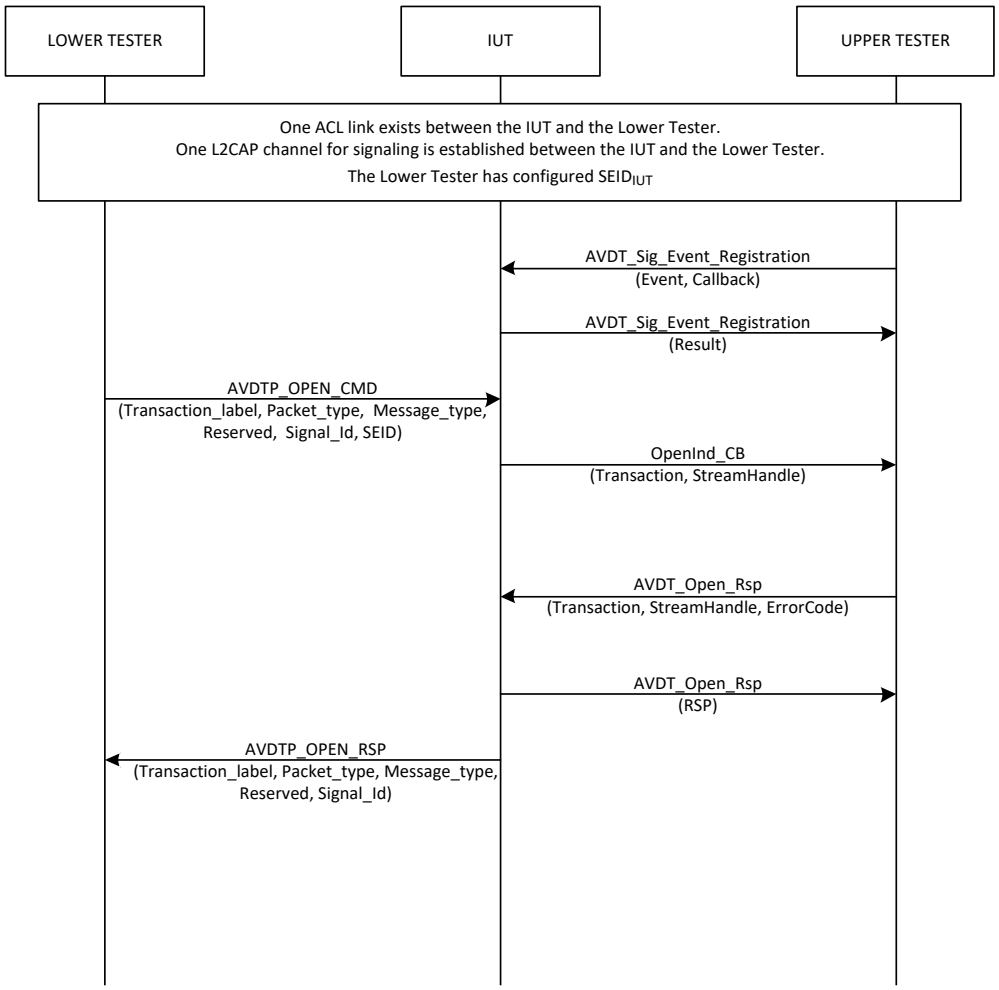

Figure 4.12: Open command – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Open_Ind and Callback = OpenInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_OPEN_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x06 (AVDTP_OPEN), and SEID = $SEID_{IUT}$ = the value used for configuring the SEP.
4. The IUT sends an OpenInd_CB event to the Upper Tester with Transaction and StreamHandle set to valid values.
5. The Upper Tester sends an AVDT_Open_Rsp service primitive to the IUT with Transaction = TransactionLowerTester, Stream_Handle = SHIUT = the value corresponding to SEIDIUT, and ErrorCode = SUCCESS.
6. The IUT responds to the Upper Tester with AVDT_Open_Rsp with a valid Result.
7. The IUT sends an AVDTP_OPEN_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the OpenInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - Stream_Handle = SH_{IUT} The value corresponding to $SEID_{IUT}$

  In Step 6, the IUT sends the AVDT_Open_Rsp service primitive to the Upper Tester with:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x06 (AVDTP_OPEN)

AVDTP/SRC/INT/SIG/SMG/BV-17-C [Start a Streaming procedure – INT]

- Test Purpose

  Verify that the IUT (INT, SRC) is able to start a Streaming procedure after the stream establishment is complete, by issuing the AVDTP_START_CMD, reporting the replied confirmation and streaming packets.

- Reference

  [3] 6.4, 6.11, 7.2.1, 8.12, 13.1

  [5] 6.4, 6.12, 7.2.1, 8.13, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) such as multiplexing is not selected, reporting is not selected and recovery is not selected, see Section 4.3.1.5, Set Configuration command – INT.
  - A stream connection has been established to the SEP identified by $SEID_{LowerTester}$ (Stream handle = SH_{IUT}), see Section 4.3.1.11, Open command – INT.

- Test Procedure

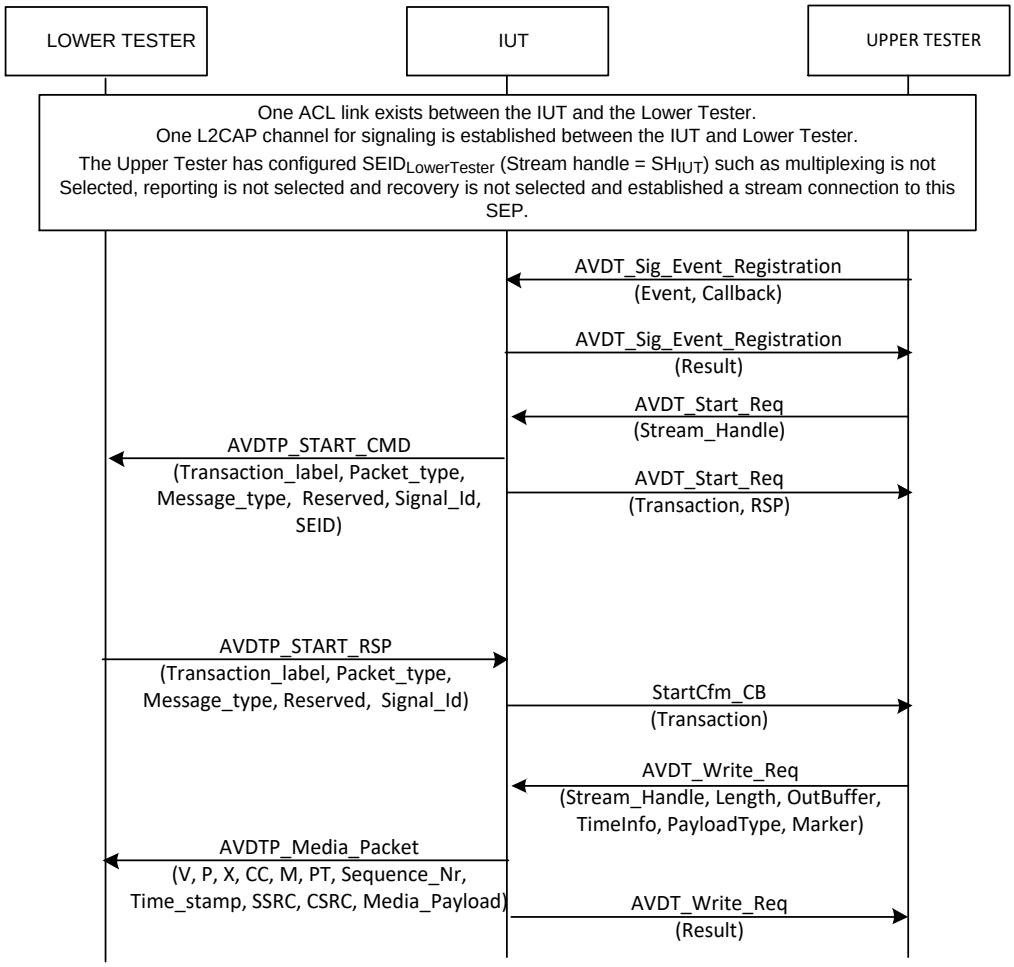

Figure 4.13: AVDTP/SRC/INT/SIG/SMG/BV-17-C [Start a Streaming procedure – INT] MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Start _Cfm and Callback = StartCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Start_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing a stream connection for $SEID_{LowerTester}$.
4. The IUT sends an AVDTP_START_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
5. The IUT sends an AVDT_Start_Req response to the Upper Tester with Transaction and RSP set to valid values.
6. The Lower Tester sends an AVDTP_START_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x07 (AVDTP_START).
7. The IUT sends the StartCfm_CB event to the Upper Tester with a valid Transaction.
8. The Upper Tester sends an AVDT_Write_Req to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing a stream connection for $SEID_{LowerTester}$, Length = L_{data_buffer} (stream data frame buffer length) = Media_Payload of AVDTP media frame, OutBuffer = $ADDR_{DataBuffer}$, TimingInfo = $Time\_Stamp_{Tester}$, and Marker = one of the possible values (1 or 0).
9. The IUT sends an AVDTP_Media_Packet to the Lower Tester with V, P, X, CC, M, PT, Sequence_Nr, Time_stamp, SSRC, CSRC, and Media_Payload set to valid values.
10. The IUT sends an AVDT_Write_Req response to the Upper Tester with a valid Result.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDTP_START_CMD signaling message with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x07 (AVDTP_START)
  - SEID = $SEID_{LowerTester}$ = the value used for stream end-point configuration.

  In Step 5, the IUT sends the AVDT_Start_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT invokes the StartCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$

  In Step 9, the IUT sends a media packet to the Lower Tester. See Section 7.2.1 in [3] [5] for Media Packet Format definitions.

4.3.1.13 Receive Start command – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of valid AVDTP_START_CMD and replies the returned confirmation.

- Reference

  [3] 6.4, 6.11, 8.12, 13.1

  [5] 6.4, 6.12, 8.13, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
  - The Lower Tester has established a stream connection to the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-18-C [Receive Start command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-18-C [Receive Start command – ACP]

Table 4.15: Receive Start command – ACP test cases

- Test Procedure

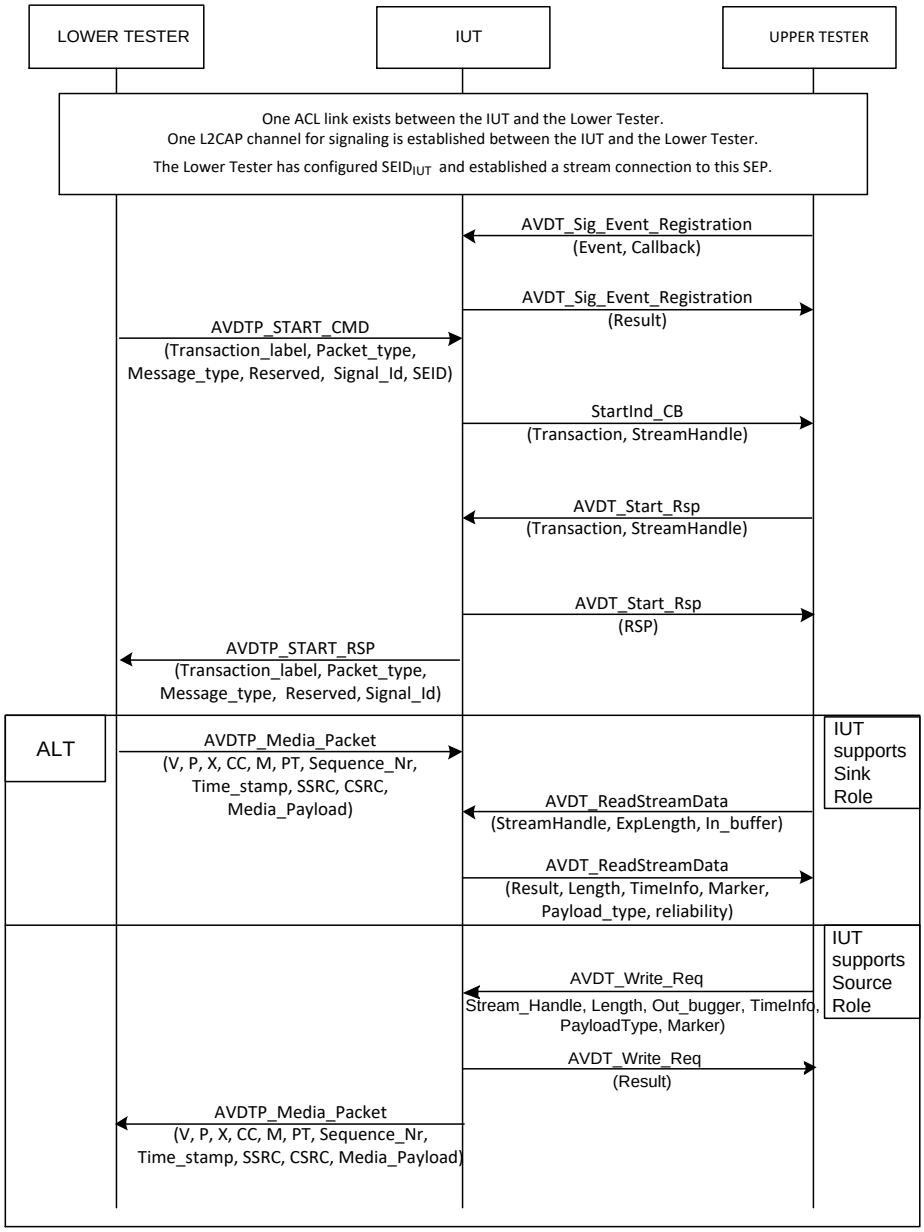

Figure 4.14: Receive Start command – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration to the IUT with service primitive Event = AVDT_Start _Ind and Callback = StartInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_START_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x07 (AVDTP_START), and SEID = $SEID_{IUT}$ = the value used for configuring the SEP.
4. The IUT sends a StartInd_CB event to the IUT with Transaction and StreamHandle set to valid values.
5. The Upper Tester sends an AVDT_Start_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$ and Stream_Handle = SH_{IUT} = the value corresponding to $SEID_{IUT}$.
6. The IUT sends an AVDT_Start_Rsp response to the Upper Tester with a valid RSP.

7. The IUT sends an AVDTP_START_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.
8. Execute either alternative 8A or 8B depending on whether the IUT supports the sink or the source role.

   Alternative 8A (the IUT supports the **sink** role):

   8A.1. The Lower Tester sends an AVDTP_Media_Packet with V, P, X, CC, M, PT, Sequence_Nr, Time_stamp, SSRC, CSRC, and Media_Payload set to valid values.

   8A.2. The Upper Tester sends an AVDT_Read_Stream_Data service primitive to the IUT with Stream_Handle = SH_{IUT}, ExpLength = $L_{DataBuffer}$ (stream data frame buffer length) = Media_Payload expected length of AVDTP media frame, and InBuffer = $ADDR_{DataBuffer}$.

   8A.3. The IUT sends an AVDT_ReadStreamData response to the Upper Tester with Result, Length, TimeInfo, Marker, Payload_type, and reliability set to valid values.

   Alternative 8B (the IUT supports the **source** role:

   8B.1. The Upper Tester sends an AVDT_Write_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing a stream connection for $SEID_{LowerTester}$, Length = $L_{DataBuffer}$ (stream data frame buffer length) = Media_Payload of AVDTP media frame, OutBuffer = $ADDR_{DataBuffer}$, TimingInfo = $Time\_Stamp_{Tester}$, and Marker = one of the possible values (1 or 0).

   8B.2. The IUT sends an AVDT_Write_Req response to the Upper Tester with a valid result.

   8B.3. The IUT sends an AVDTP_Media_Packet with V, P, X, CC, M, PT, Sequence_Nr, Time_stamp, SSRC, CSRC, and Media_Payload set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the StartInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - StreamHandle = SH_{IUT} = the value corresponding to $SEID_{IUT}$

  In Step 6, the IUT sends the AVDT_Start_Rsp service primitive to the Upper Tester with:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Upper Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x07 (AVDTP_START)

In Step 8A.3, for the IUT supporting the **sink** role, the IUT sends the output AVDT_ReadStreamData service primitive to the Upper Tester with:

- Result = 0 (successful data transfer)
- Length = Media_Payload length of AVDTP media packet
- Time_Info = $Time\_Stamp_{LowerTester}$
- Marker = one of the possible values (1 or 0)
- Payload type = PT_{IUT} = value chosen according to the implementation in the IUT
- Reliability = 0 (no errors)

In Step 8B.3, for an IUT supporting the **source** role, the IUT sends the Lower Tester a media packet. See 7.2.1 in [3] [5] for Media Packet Format definitions.

- Notes

  The IUT (if supporting the **source** role) can re-configure or change the SEP before the streaming is started.

4.3.1.14 Close command – INT

- Test Purpose

  Verify that the IUT (INT) is able to initiate the release of a Streaming procedure by issuing the AVDTP_CLOSE_CMD, releasing all the resources associated to the SEP after reception of the replied confirmation and reporting the closing.

- Reference

  [3] 6.4, 6.12, 8.13, 13.1

  [5] 6.4, 6.13, 8.14, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) such as multiplexing is not selected, reporting is not selected and recovery is not selected (see Section 4.3.1.11, Open command – INT).
  - The stream has been established.
  - The SEP identified by $SEID_{LowerTester}$ is in the Open state.
  - Either the IUT or the Lower Tester can act as the INT of the Stream Configuration procedure. The INT of the Stream Configuration procedure is also the INT of the Establishment and Start procedures.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-19-C [Close command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-19-C [Close command – INT]

Table 4.16: Close command – INT test cases

- Test Procedure

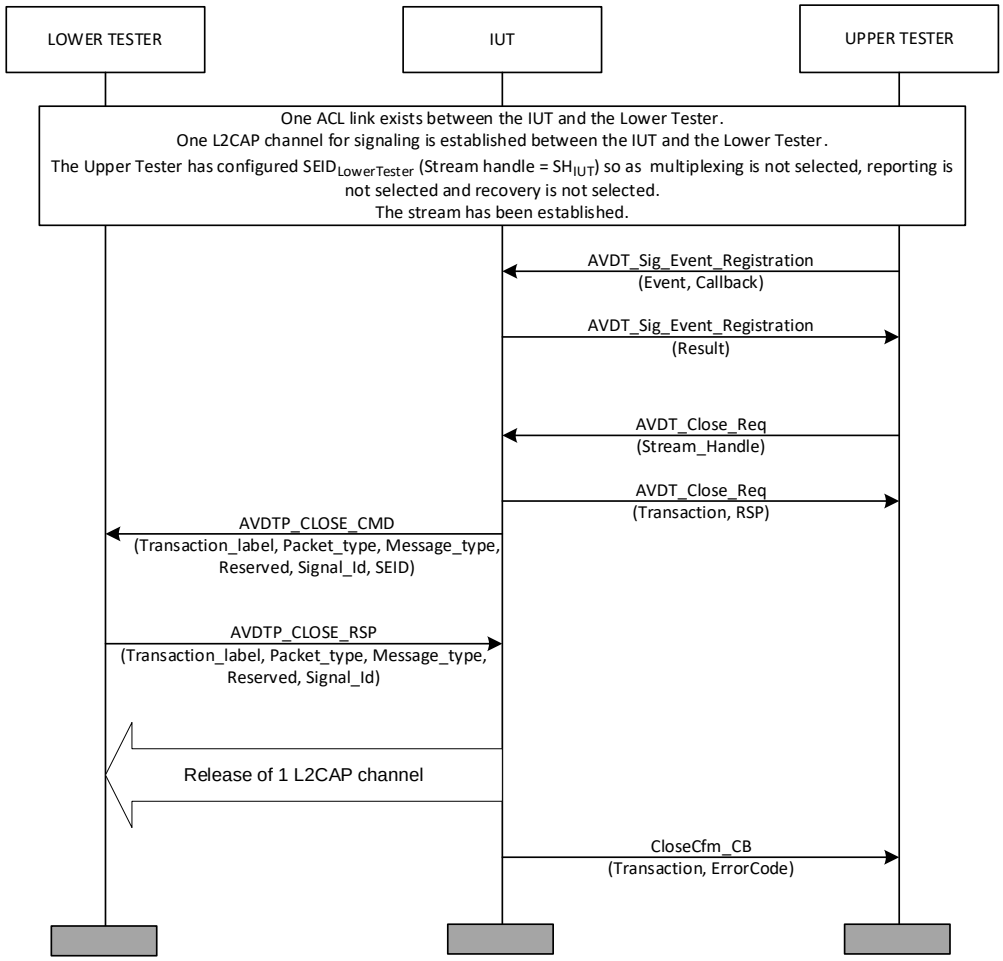

Figure 4.15: Close command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Close_Cfmand Callback = CloseCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Close_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing $SEID_{LowerTester}$.
4. The IUT responds to the Upper Tester with AVDT_Close_Req with valid Transaction and RSP.
5. The IUT sends an AVDTP_CLOSE_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester provides an AVDTP_CLOSE_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x08 (AVDTP_CLOSE).
7. One L2CAP channel is released. The Lower Tester provides positive responses to the L2CAP_DisconnectReq messages issued by the IUT.
8. The IUT sends a CloseCfm_CB event to the Upper Tester with valid Transaction and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

In Step 4, the IUT sends the AVDT_Close_Req service primitive to the Upper Tester with:

- Transaction = $Transaction_{IUT}$ = any value
- RSP = REQUEST_ACCEPTED

In Step 5, the IUT sends the AVDTP signaling message to the Upper Tester with the fields:

- Transaction_label = $Transaction_{IUT}$ = any value
- Message_type = 00 (command)
- Signal_Id = 0x08 (AVDTP_CLOSE)
- SEID = $SEID_{LowerTester}$ = the value used for stream end-point configuration and establishment.

In Step 7, the IUT releases the L2CAP channel connected and configured during the stream establishment (it issues L2CAP_DisonnectReq with valid parameters for the L2CAP channel).

In Step 8, the IUT invokes the CloseCfm_CB callback function with the following parameters:

- Transaction = $Transaction_{IUT}$
- ErrorCode = SUCCESS

4.3.1.15 Close command – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to accept an AVDTP_CLOSE_CMD after the Streaming procedure has been started, by releasing all the resources associated to the SEP, reporting the closing, and replying the returned confirmation.

- Reference

  [3] 6.4, 6.12, 8.13, 13.1

  [5] 6.4, 6.13, 8.14, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured and established the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.
  - The SEP identified by $SEID_{IUT}$ is in the Open state.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-20-C [Close command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-20-C [Close command – ACP]

Table 4.17: Close command – ACP test cases

- Test Procedure

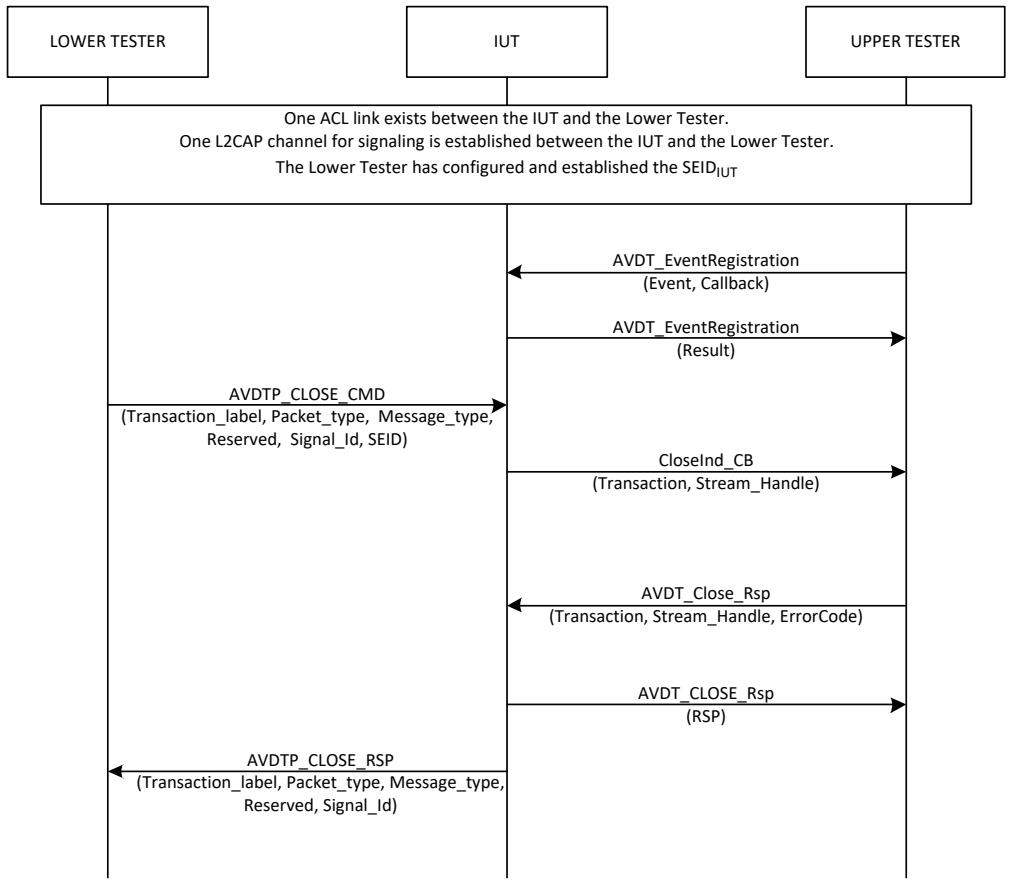

Figure 4.16: Close command – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Close_Ind and Callback = CloseInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_CLOSE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x08 (AVDTP_CLOSE), and SEID = $SEID_{IUT}$ = the value used for configuring and establishing the SEP.
4. The IUT sends a CloseInd_CB event to the Upper Tester with valid Transaction and Stream_Handle.
5. The Upper Tester sends an AVDT_Close_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, Stream_Handle = SH_{IUT} = the value used when configuring and establishing the SEP, and ErrorCode = SUCCESS.
6. The IUT sends an AVDT_Close_Rsp response to the Upper Tester with a valid RSP.
7. The IUT sends an AVDTP_CLOSE_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

In Step 4, the IUT invokes the CloseInd_CB callback function with the following parameters:

- Transaction = $Transaction_{LowerTester}$
- Stream_Handle = SH_{IUT} = the value used when configuring and establishing the SEP

In Step 6, the IUT sends the AVDT_Close_Rsp service primitive to the Upper Tester with:

- RSP = REQUEST_ACCEPTED

In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Message_type = 10 (response accept)
- Signal_Id = 0x08 (AVDTP_CLOSE)

AVDTP/SRC/INT/SIG/SMG/BV-21-C [Suspend command – INT]

- Test Purpose

  Verify that the IUT (INT, SRC) is able to initiate the suspension of a Streaming procedure, after the Streaming procedure has been started, by issuing the AVDTP_SUSPEND_CMD. Further streaming is refused.

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) such as multiplexing is not selected, reporting is not selected and recovery is not selected (see Section 4.3.1.7, Set Configuration command – INT).
  - A stream connection to the SEP identified by $SEID_{LowerTester}$ has been established (Stream handle = SH_{IUT}) and started streaming (see AVDTP/SRC/INT/SIG/SMG/BV-17-C [Start a Streaming procedure – INT]).
  - The SEP identified by $SEID_{LowerTester}$ is in the Streaming state.

- Test Procedure

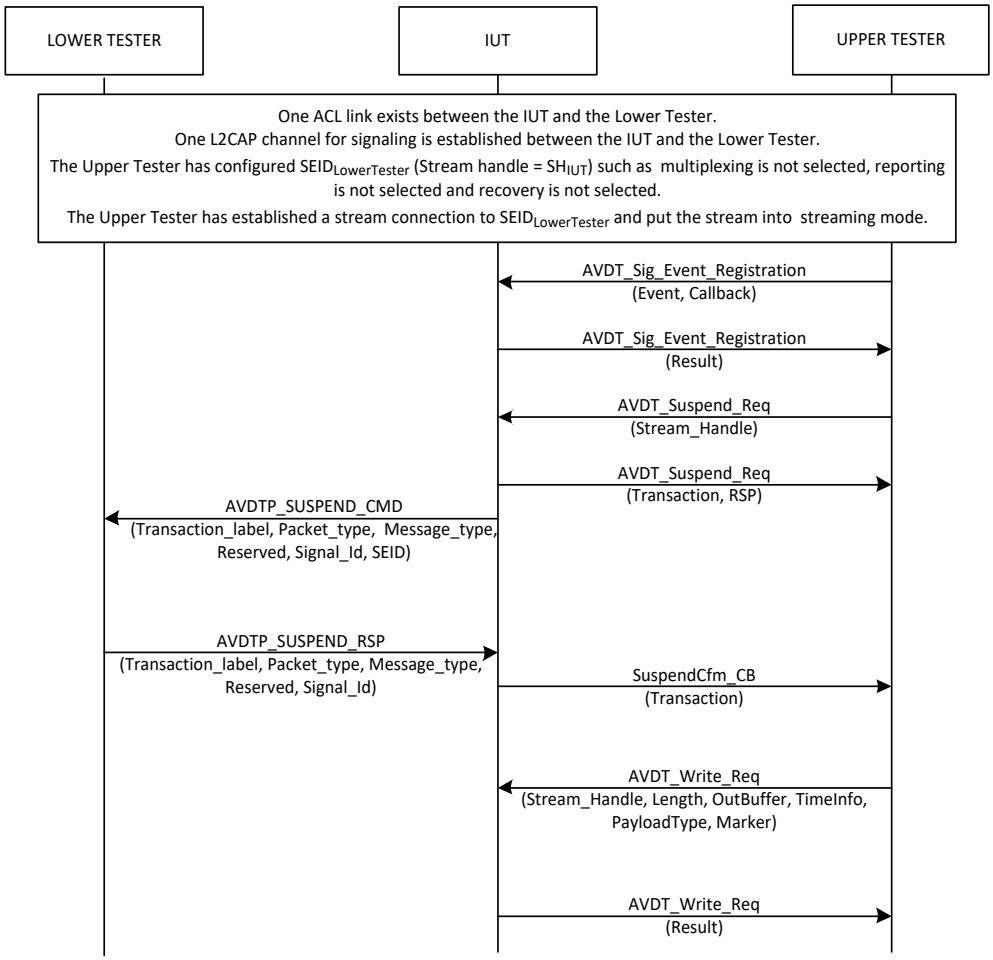

Figure 4.17: AVDTP/SRC/INT/SIG/SMG/BV-21-C [Suspend command – INT] MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Suspend_Cfm and Callback = SuspendCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester send an AVDT_Suspend_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring, establishing a stream connection for $SEID_{LowerTester}$ and starting streaming.
4. The IUT responds with an AVDT_Suspend_Req response to the Upper Tester with valid Transaction and RSP.
5. The IUT sends an AVDTP_SUSPEND_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_SUSPEND_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x09 (AVDTP_SUSPEND).
7. The IUT sends a SuspendCfm_CB event to the Upper Tester with valid Transaction.
8. The Upper Tester sends an AVDTP_Write_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing a stream connection for $SEID_{LowerTester}$, Length = L_{data_buffer} (stream data frame buffer length) = Media_Payload of AVDTP media frame,

OutBuffer = $ADDR_{DataBuffer}$, TimingInfo = $Time\_Stamp_{Tester}$, and Marker = 0 = one of the possible values (1 or 0).

9. The IUT sends AVDT_Write_Req to the Upper Tester with valid result.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent to the Upper Tester by the IUT is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Suspend_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x09 (AVDTP_SUSPEND)
  - SEID = SEIDLowerTester = the value used for stream end-point configuration

  In Step 7, the IUT invokes the SuspendCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$

  The Lower Tester receives no media packet.

  In Step 9, the output parameter 'Result' of the AVDT_Write_Req service primitive sent by the IUT to the Upper Tester indicates a rejection of the request.

4.3.1.16 Suspend command – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to accept an AVDTP_SUSPEND_CMD, after the Streaming procedure has been started, by reporting the suspension and replying the returned confirmation.

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (see Section 4.3.1.6, Set Configuration response – ACP).
  - The Lower Tester has established a stream connection to the SEP identified by $SEID_{IUT}$ and started the stream (see Section 4.3.1.13, Receive Start command – ACP).

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-22-C [Suspend command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-22-C [Suspend command – ACP]

Table 4.18: Suspend command – ACP test cases

- Test Procedure

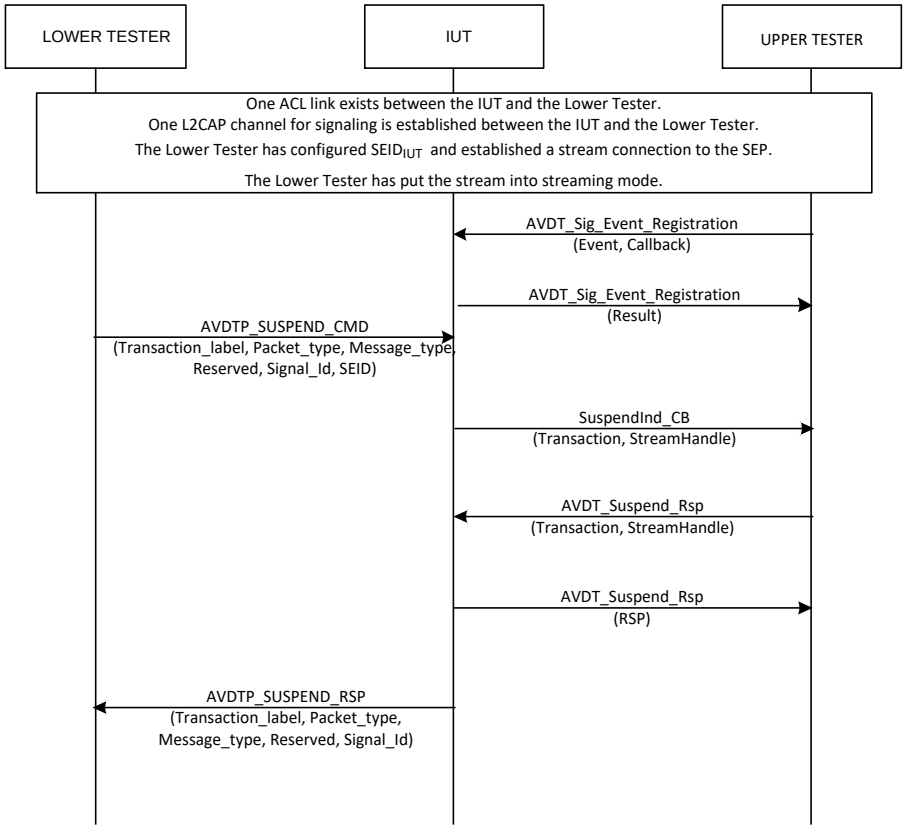

Figure 4.18: Suspend command – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Suspend_Ind and Callback = SuspendInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_SUSPEND_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x09 (AVDTP_SUSPEND), and SEID = $SEID_{IUT}$ = the value used for configuring the SEP.
4. The IUT sends the SuspendInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
5. The Upper Tester sends an AVDT_Suspend_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$ and Stream_Handle = SH_{IUT} = the value corresponding to $SEID_{IUT}$.
6. The IUT sends an AVDT_Suspend_Rsp response to the Upper Tester with a valid RSP.
7. The IUT sends an AVDTP_SUSPEND_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the SuspendInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - StreamHandle = SH_{IUT} = the value corresponding to $SEID_{IUT}$

  In Step 6, the IUT sends the AVDT_Suspend_Rsp service primitive to the Upper Tester with:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x09 (AVDTP_SUSPEND)

4.3.1.17 Abort command – INT

- Test Purpose

  Verify that the IUT (INT) is able to abort a stream by issuing the AVDTP_ABORT and reporting the replied confirmation.

- Reference

  [3] 6.4, 6.16, 8.15, 13.1

  [5] 6.4, 6.17, 8.16, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured.
  - Stream handle = SH_{IUT}.
  - See Section 4.3.1.5, Set Configuration command – INT.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-23-C [Abort command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-23-C [Abort command – INT]

Table 4.19: Abort command – INT test cases

- Test Procedure

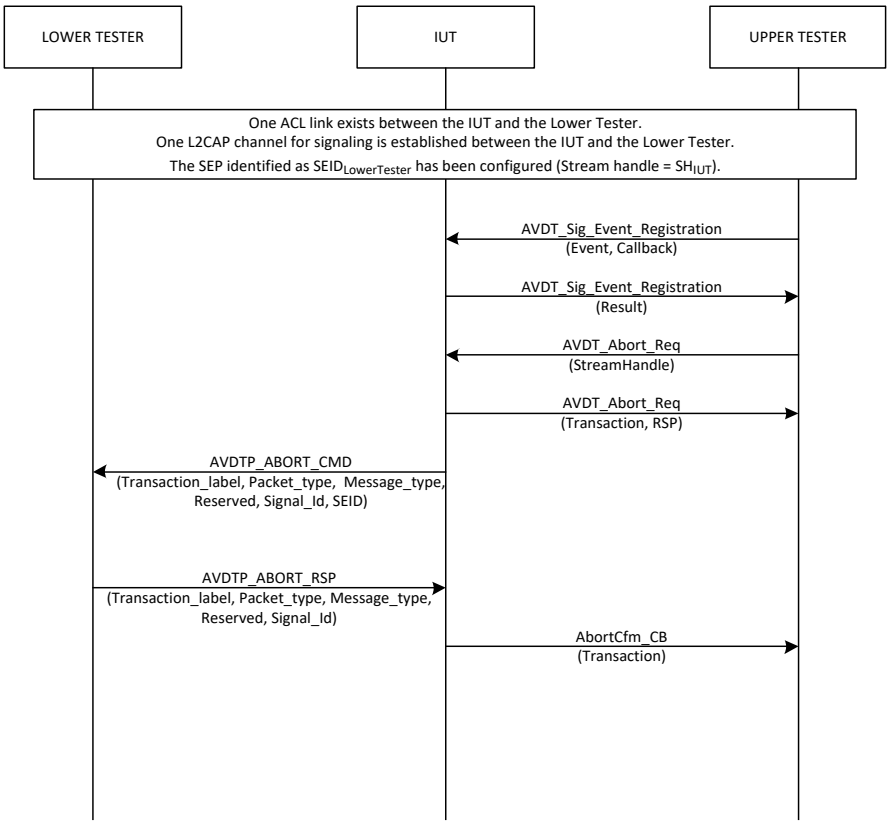

Figure 4.19: Abort command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Abort_Cfm and Callback = AbortCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Abort_Req service primitive to the IUT with StreamHandle = SH_{IUT} = the value corresponding to $SEID_{LowerTester}$.
4. The IUT sends an AVDT_Abort_Req response to the Upper Tester with valid Transaction and RSP.
5. The IUT sends an AVDTP_ABORT_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_ABORT_RSP response with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x0A (AVDTP_ABORT).
7. The IUT sends an AbortCfm_CB event to the Upper Tester with valid Transaction.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Abort_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{IUT}$
- Message_type = 00 (command)
- Signal_Id = 0x0A (AVDTP_ABORT)
- SEID = $SEID_{LowerTester}$

In Step 7, the IUT invokes the AbortCfm_CB callback function with the following parameters:

- Transaction = $Transaction_{IUT}$

4.3.1.18 Accept Abort – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to accept a command to abort a stream.

- Reference

  [3] 6.4, 6.16, 8.15, 13.1

  [5] 6.4, 6.17, 8.16, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-24-C [Accept Abort – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-24-C [Accept Abort – ACP]

Table 4.20: Accept Abort – ACP test cases

- Test Procedure

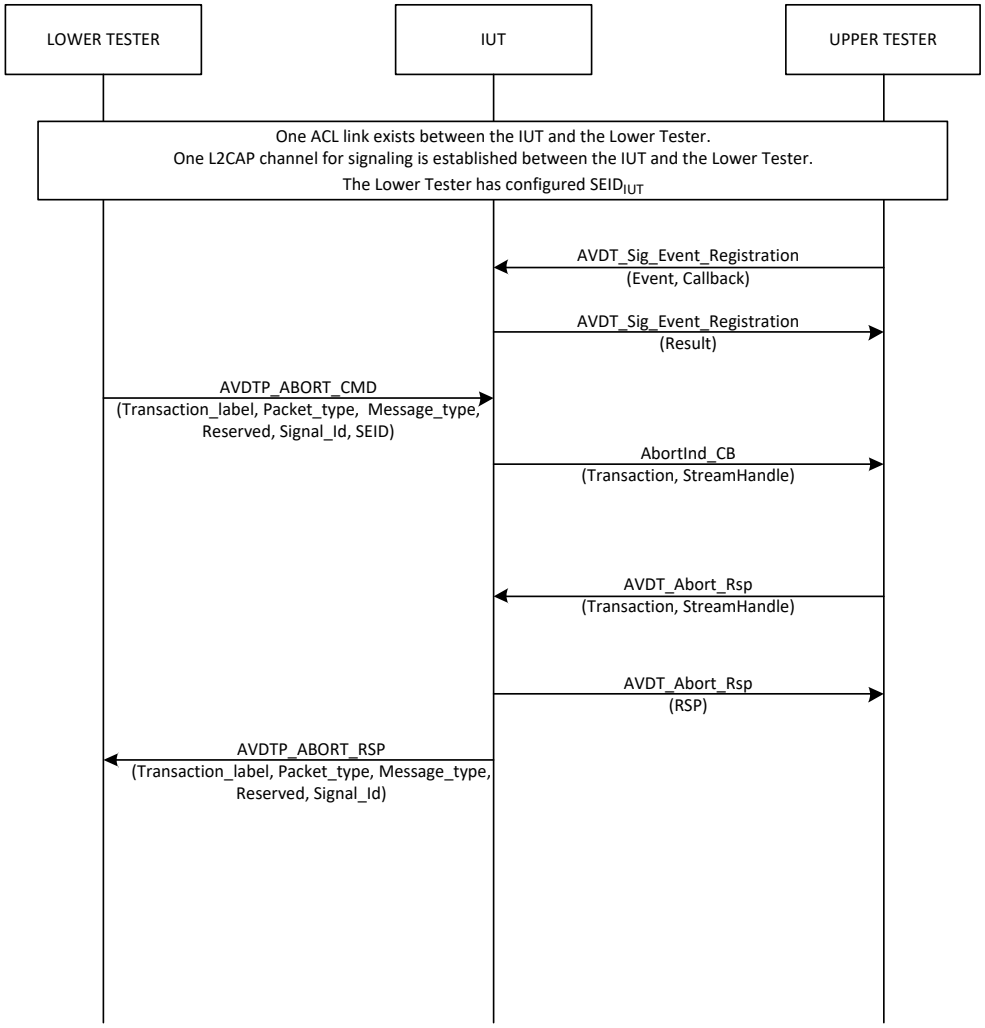

Figure 4.20: Accept Abort – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Abort_Ind and Callback = AbortInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_ABORT_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x0A (AVDTP_ABORT), and SEID = $SEID_{IUT}$ = any value.
4. The IUT sends an AbortInd_CB to the IUT with valid Transaction and StreamHandle.
5. The Upper Tester sends an AVDT_Abort_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, and StreamHandle = SH_{IUT} = the value corresponding to $SEID_{IUT}$.
6. The IUT sends an AVDT_Abort_Rsp response to the Upper Tester with valid RSP.
7. The IUT sends an AVDTP_ABORT_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the AbortInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - StreamHandle = SH_{IUT}

  In Step 6, the IUT sends the AVDT_Abort_Rsp service primitive to the Upper Tester with:

  - RSP = REQUEST_ACCEPTED

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x0A (AVDTP_ABORT)

4.3.1.19 Get All Capabilities command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid query for remote SEP capabilities and reports the replied ones using Get_All_Capabilities.

- Reference

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-25-C [Get All Capabilities command – INT]
AVDTP/SNK/INT/SIG/SMG/BV-25-C [Get All Capabilities command – INT]

Table 4.21: Get All Capabilities command – INT test cases

- Test Procedure

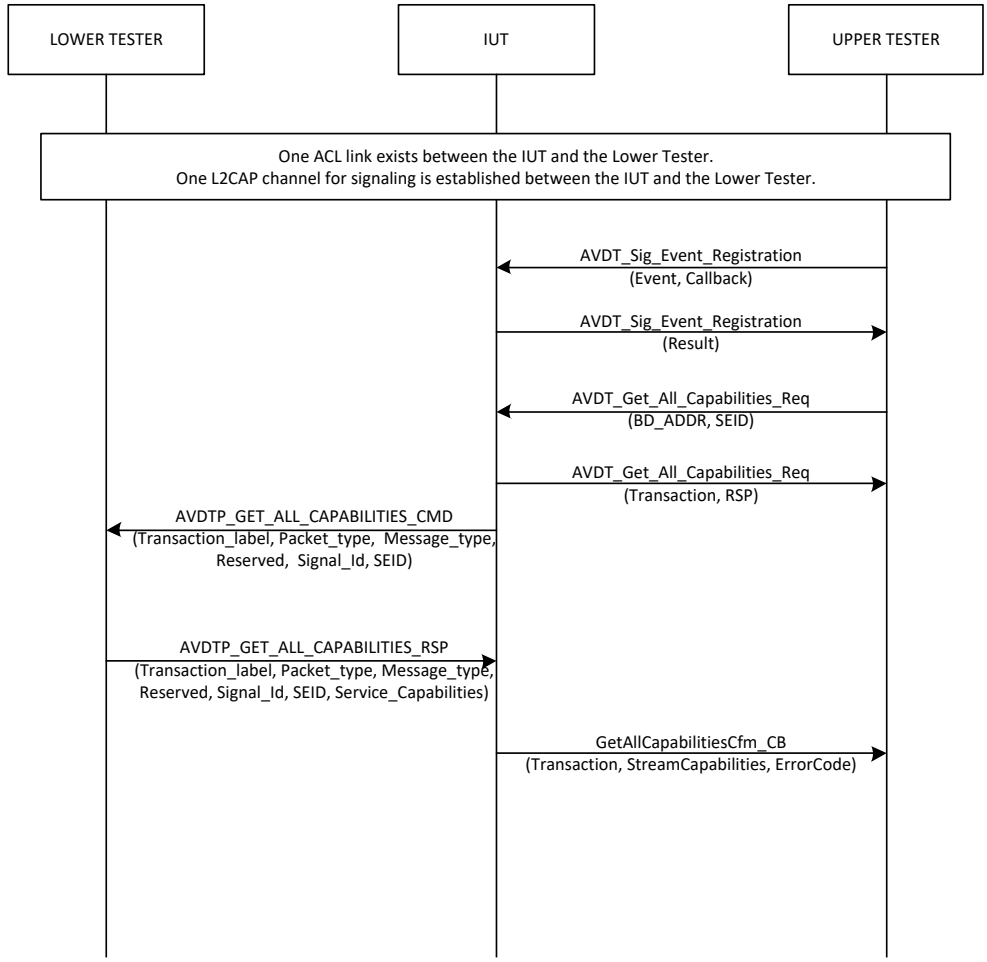

Figure 4.21: Get All Capabilities command – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_All_Capabilities_Cfm and Callback = GetAllCapabilitiesCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Get_All_Capabilities_Req service primitive to the IUT with BD_ADDR = BD_ADDRLowerTester and SEID = SEIDLowerTester = any value.
4. The IUT sends an AVDT_Get_All_Capabilities_Req response to the Upper Tester with valid Transaction and RSP.
5. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the IUT with Transaction_label = Transaction_{IUT}, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Service_Capabilities = valid capabilities values for each of the following types: Media

transport, reporting, recovery, video codec or audio codec or multimedia codec, content protection, header compression, multiplexing.

7. The IUT sends a GetAllCapabilitiesCfm_CB event to the Upper Tester with Transaction, StreamCapabilities, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Get_All_Capabilities_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
  - SEID = $SEID_{LowerTester}$

  In Step 7, the IUT invokes the GetAllCapabilitiesCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - Stream_capabilities = the values provided with AVDTP_GET_ALL_CAPABILITIES_RSP
  - ErrorCode = SUCCESS

4.3.1.20 Get All Capabilities response – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid query for remote SEP capabilities and replies the returned capabilities using Get_All_Capabilities.

- Reference

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BV-26-C [Get All Capabilities response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BV-26-C [Get All Capabilities response – ACP]

Table 4.22: Get All Capabilities response – ACP test cases

- Test Procedure

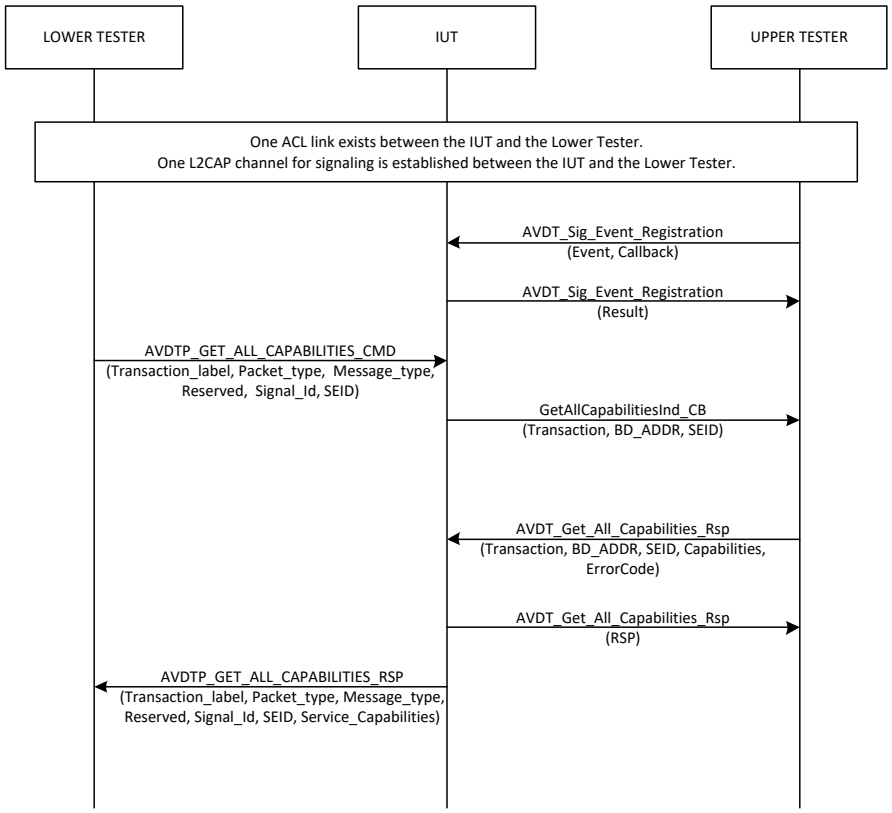

Figure 4.22: Get All Capabilities response – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_Capabilities_Ind and Callback = GetCapabilitiesInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and SEID = $SEID_{IUT}$ = any value.
4. The IUT sends an GetAllCapabilitiesInd_CB event with Transaction, BD_ADDR, and SEID set to valid values.
5. The Upper Tester sends an AVDT_Get_All_Capabilities_Rsp service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, SEID = $SEID_{IUT}$, Capabilities = valid capabilities values for each category that the addressed SEP supports, and ErrorCode = SUCCESS.
6. The IUT sends an AVDT_Get_All_Capabilities_Rsp response to the Upper Tester with valid RSP.
7. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Service_Capabilities set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive sent by the IUT to the Upper Tester is 0x0000 (Event successfully registered).

In Step 4, the IUT invokes the GetAllCapabilitiesInd_CB callback function with the following parameters:

- Transaction = $Transaction_{LowerTester}$
- BD_ADDR = $BD\_ADDR_{LowerTester}$
- SEID = $SEID_{IUT}$

In Step 6, the IUT sends the AVDT_Get_All_Capabilities_Rsp service primitive to the Upper Tester with:

- RSP = REQUEST_ACCEPTED

In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Message_type = 10 (response accept)
- Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
- SEID = $SEID_{IUT}$
- Service_Capabilities = the values provided with AVDT_Get_All_Capabilities_Rsp

4.3.1.21 Get All Capabilities response with RFD values – INT

- Test Purpose

  Verify that the IUT (INT) ignores capability values reserved for future definition (RFD) returned in the GetAllCapabilities response.

- Reference

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-28-C [Get All Capabilities response with RFD values – INT]
AVDTP/SNK/INT/SIG/SMG/BV-28-C [Get All Capabilities response with RFD values – INT]

Table 4.23: Get All Capabilities response with RFD values – INT test cases

- Test Procedure

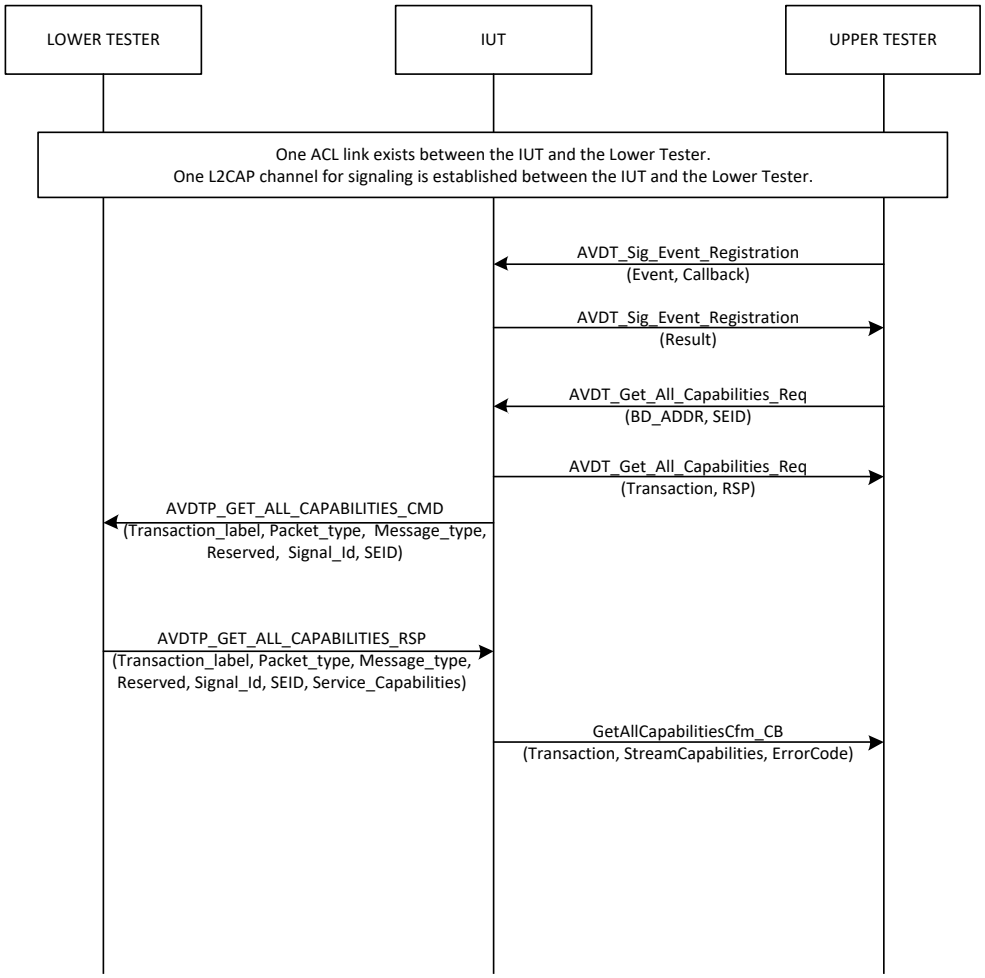

Figure 4.23: Get All Capabilities response with RFD values – INT MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Get_All_Capabilities_Cfm and Callback = GetAllCapabilitiesCfm_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Get_All_Capabilities_Req service primitive to the IUT with BD_ADDR = $BD\_ADDR_{LowerTester}$ and SEID = $SEID_{LowerTester}$ = any value.
4. The IUT sends an AVDT_Get_All_Capabilities_Req response to the Upper Tester with valid Transaction and RSP.
5. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
6. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the IUT with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Service_Capabilities = valid capabilities values for each of the following types: Media transport, reporting, recovery, video codec or audio codec or multimedia codec, content protection, header compression, multiplexing, delay reporting. In addition, at least one service

capabilities category value that is defined as RFD in Table 8.47, Section 8.21.1 in [5] must be included.

7. The IUT sends a GetAllCapabilitiesCfm_CB event to the Upper Tester with Transaction, StreamCapabilities, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the result of the AVDT_Sig_Event_Registration service primitive is 0x0000 (Event successfully registered).

  In Step 4, the IUT sends the AVDT_Get_All_Capabilities_Req service primitive to the Upper Tester with:

  - Transaction = $Transaction_{IUT}$ = any value
  - RSP = REQUEST_ACCEPTED

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
  - SEID = $SEID_{LowerTester}$

  In Step 7, the IUT invokes the GetCapabilitiesCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - Stream_capabilities = the values provided with AVDTP_GET_ALL_CAPABILITIES_RSP that do not have associated service
  - Category values that are defined as RFD in Table 8.47, Section 8.21.1 in [5]
  - ErrorCode = SUCCESS

4.3.1.22 Set Configuration command following a Get Capabilities response with All Defined Capabilities – INT

- Test Purpose

  Verify that the IUT (INT) is able to accept all defined capabilities in the GetCapabilities response returned by the Lower Tester.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The first SEP of the Lower Tester supports all the basic capabilities defined in AVDTP in its AVDTP_GET_CAPABILITIES_RSP.
  - The IUT has performed Stream End Point Discovery and has discovered the first SEP of the Lower Tester.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-31-C [Set Configuration command following a Get Capabilities response with All Defined Capabilities – INT]
AVDTP/SNK/INT/SIG/SMG/BV-31-C [Set Configuration command following a Get Capabilities response with All Defined Capabilities – INT]

Table 4.24: Set Configuration command following a Get Capabilities response with All Defined Capabilities – INT test cases

- Test Procedure

  1. The IUT performs the Get Capabilities procedure for the first SEP of the Lower Tester.
  2. The Lower Tester returns a valid set of capabilities. The set of capabilities returned by the Lower Tester are all the basic capabilities defined in AVDTP.
  3. The IUT then sends a Set Configuration command to the Lower Tester using the capabilities supported by the IUT.

- Expected Outcome

  Pass verdict

  The IUT sends an AVDTP_Set_Configuration_CMD command.

4.3.1.23 Detect Stream Discover command reject by ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of a Stream Discover command.

- Reference

  [3] 6.4, 6.6, 8.6, 13.1

  [5] 6.4, 6.6, 8.6, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-01-C [Detect Stream Discover command reject by ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-01-C [Detect Stream Discover command reject by ACP – INT]

Table 4.25: Detect Stream Discover command reject by ACP – INT test cases

- Test Procedure

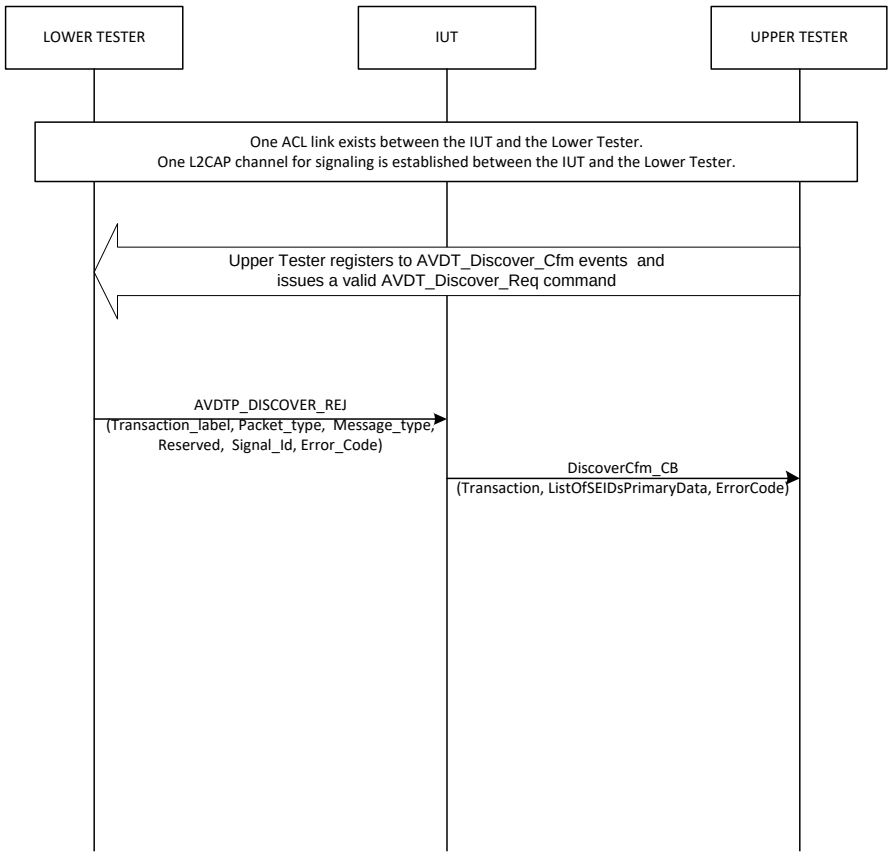

Figure 4.24: Detect Stream Discover command reject by ACP – INT MSC

1. The Upper Tester registers to AVDT_Discover_Cfm events with the IUT and issues a valid AVDT_Discover_Req command, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_DISCOVER_REJ command to the IUT with Transaction = $Transaction_{IUT}$ = value provided with the AVDTP_DISCOVER_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x01 (AVDTP_DISCOVER), and Error_Code = 0x01 (BAD_HEADER_FORMAT).
3. The IUT sends a DiscoverCfm_CB event to the Upper Tester with Transaction, ListOfSEIDsPrimaryData, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 3, the IUT invokes the DiscoverCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - Error_Code = the Error_Code provided with the AVDTP_DISCOVER_REJ message

4.3.1.24 Discard corrupted Stream Discover command – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to discard a corrupted Stream Discover command received by the INT.

- Reference

  [3] 6.4, 6.6, 8.6, 13.1

  [5] 6.4, 6.6, 8.6, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-02-C [Discard corrupted Stream Discover command – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-02-C [Discard corrupted Stream Discover command – ACP]

Table 4.26: Discard corrupted Stream Discover command – ACP test cases

- Test Procedure

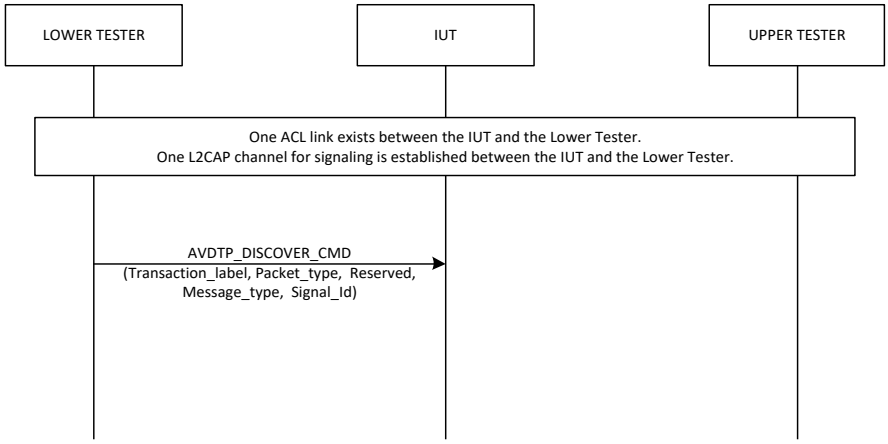

Figure 4.25: Discard corrupted Stream Discover command – ACP MSC

1. The Lower Tester sends an AVDTP_DISCOVER_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 01 (reserved), and Signal_Id = 0x01 (AVDTP_DISCOVER).

- Expected Outcome

  Pass verdict

  No message is sent to the upper layers and no response is generated.

4.3.1.25 Stream Discover Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Stream Discover command.

- Reference

  [3] 6.4, 6.6, 8.6, 13.1

  [5] 6.4, 6.6, 8.6, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Upper Tester registers to AVDT_Discover_Ind events.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-03-C [Stream Discover Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-03-C [Stream Discover Reject response – ACP]

Table 4.27: Stream Discover Reject response – ACP test cases

- Test Procedure

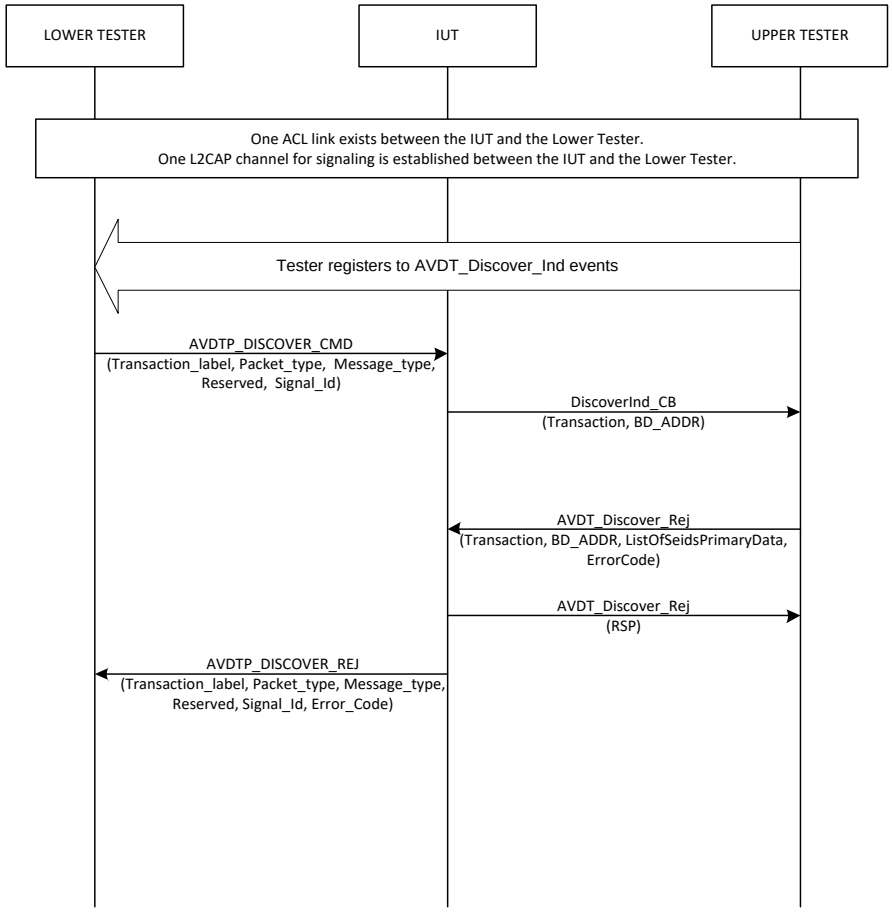

Figure 4.26: Stream Discover Reject response – ACP MSC

1. The Upper Tester registers to AVDT_Discover_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_DISCOVER_CMD command to the IUT with Transaction_label = $\text{Transaction}_{\text{LowerTester}}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x01 (AVDTP_DISCOVER).
3. The IUT sends a DiscoverInd_CB event to the Upper Tester with valid Transaction and BD_ADDR.

4. The Upper Tester sends an AVDT_Discover_Rej service primitive to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, BD_ADDR = $BD\_ADDR_{LowerTester}$, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT sends an AVDT_Discover_Rej response with a valid RSP.
6. The IUT sends an AVDTP_DISCOVER_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x01 (AVDTP_DISCOVER)
  - Error_Code = the value provided with AVDT_Discover_Rej

4.3.1.26 Detect Get Capabilities command reject by ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of a Get Capabilities command.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-04-C [Detect Get Capabilities command reject by ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-04-C [Detect Get Capabilities command reject by ACP – INT]

Table 4.28: Detect Get Capabilities command reject by ACP – INT test cases

- Test Procedure

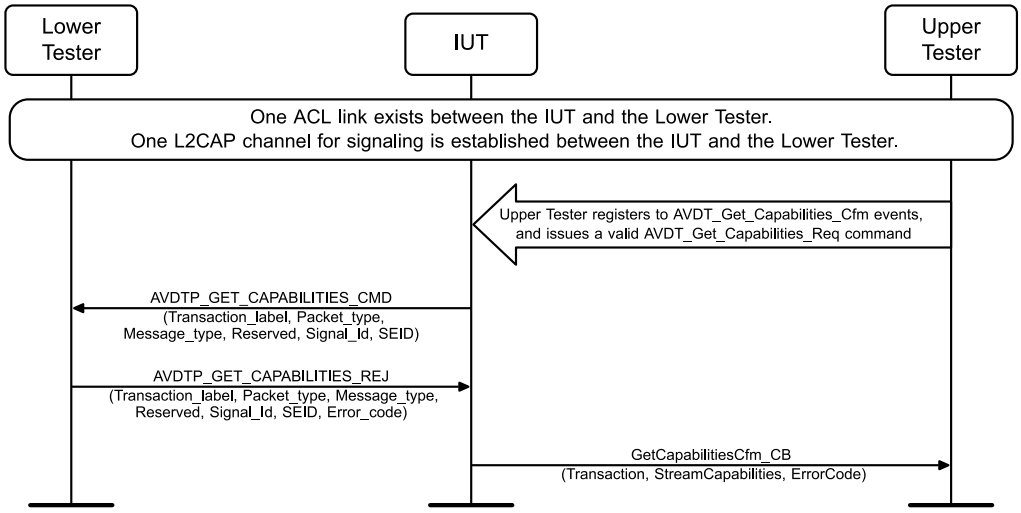

Figure 4.27: Detect Get Capabilities command reject by ACP – INT MSC

1. The Upper Tester registers to AVDT_Get_Capabilities_Cfm events with the IUT and issues a valid AVDT_Get_Capabilities_Req command for $SEID_{LowerTester}$, which the IUT acknowledges.
2. The IUT sends an AVDTP_GET_CAPABILITIES_REJ command with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
3. The Lower Tester sends an AVDTP_GET_CAPABILITIES_REJ response to the IUT with Transaction = $Transaction_{IUT}$ = value provided with the AVDTP_GET_CAPABILITIES_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Error_Code = 0x11 (BAD_LENGTH).
4. The IUT sends a GetCapabilitiesCfm_CB event to the Upper Tester with Transaction, StreamCapabilities, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the Get_CapabilitiesCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = the Error_Code provided with the AVDTP_GET_CAPABILITIES_REJ message

4.3.1.27 Invalid Get Capabilities Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Get Capabilities Reject response to the INT.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-05-C [Invalid Get Capabilities Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-05-C [Invalid Get Capabilities Reject response – ACP]

Table 4.29: Invalid Get Capabilities reject Response – ACP test cases

- Test Procedure

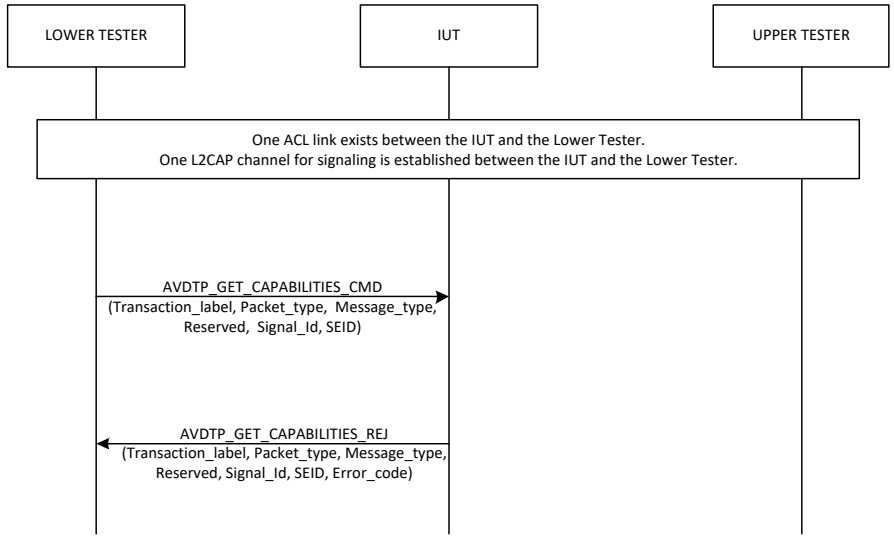

Figure 4.28: Invalid Get Capabilities Reject response – ACP MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES). The last byte meant to bear the ACP SEID + 2 reserved bits is omitted in order to have an **invalid message length**.
2. The IUT sends an AVDTP_GET_CAPABILITIES_REJ command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Error_code set to valid values.

- Expected Outcome

Pass verdict

In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Message_type = 11 (response reject)
- Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
- Error_code = 0x11 (BAD_LENGTH)

4.3.1.28 Get Capabilities Reject response – ACP

- Test Purpose

Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Get Capabilities command.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Upper Tester registers to AVDT_Get_Capabilities_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-06-C [Get Capabilities Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-06-C [Get Capabilities Reject response – ACP]

Table 4.30: Get Capabilities Reject response – ACP test cases

- Test Procedure

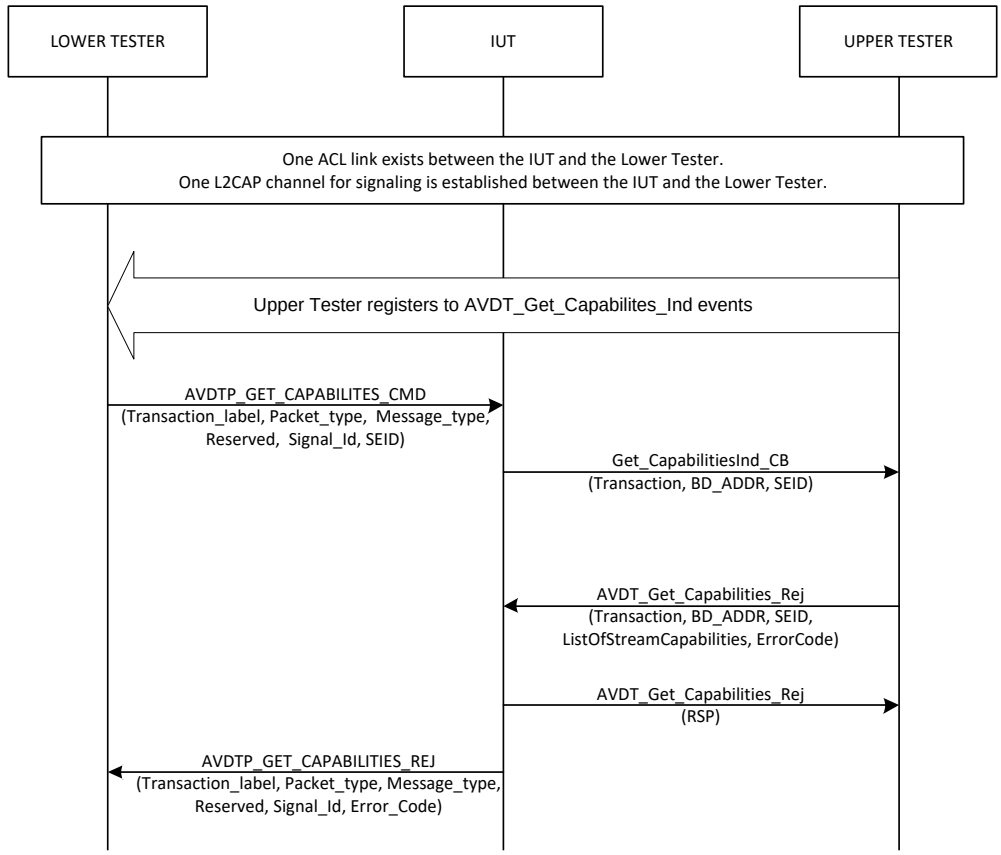

Figure 4.29: Get Capabilities Reject response – ACP MSC

1. The Upper Tester registers to AVDT_Get_Capabilities_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction_label = $\text{Transaction}_{\text{LowerTester}}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = SEID_{IUT}.

3. The IUT sends a Get_CapabilitiesInd_CB event to the Upper Tester with Transaction, BD_ADDR, and SEID set to valid values.
4. The Upper Tester sends an AVDT_Get_Capabilities_Rej service primitive to the IUT with Message_type = 00 (command) and Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES).
5. The IUT sends an AVDT_Get_Capabilities_Rej response to the Upper Tester with a valid RSP.
6. The IUT sends an AVDTP_GET_CAPABILITIES_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $\text{Transaction}_{\text{LowerTester}}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - Error_Code = the value provided with AVDT_Get_Capabilities_Rej

4.3.1.29 Detect Set Configuration command reject by ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of a Set Configuration command.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-07-C [Detect Set Configuration command reject by ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-07-C [Detect Set Configuration command reject by ACP – INT]

Table 4.31: Detect Set Configuration command reject by ACP – INT test cases

- Test Procedure

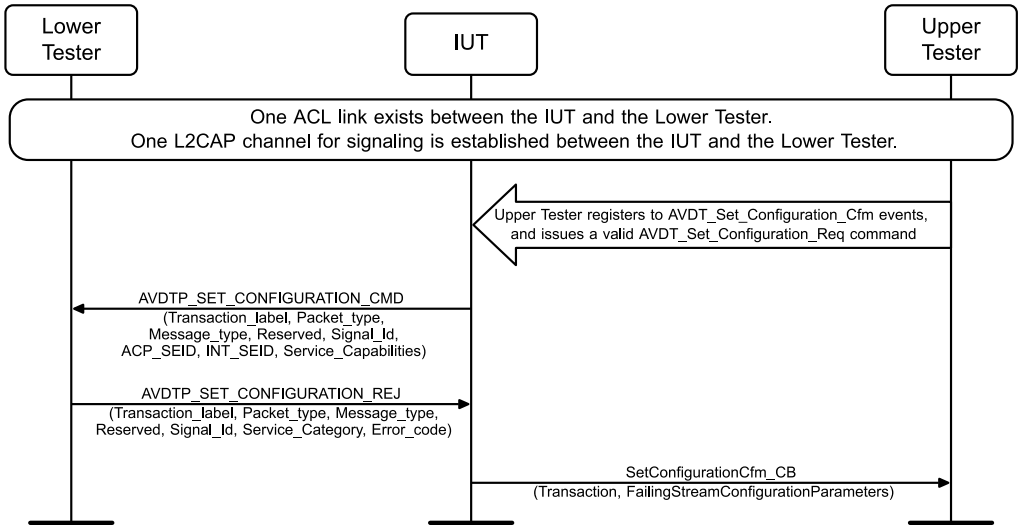

Figure 4.30: Detect Set Configuration command reject by ACP – INT MSC

1. The Upper Tester registers to AVDT_Set_Configuration_Cfm events with the IUT and issues a valid AVDT_Set_Configuration_Req command (see Section 4.3.1.5, Set Configuration command – INT), which the IUT acknowledges.
2. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities set to valid values.
3. The Lower Tester sends an AVDTP_ SET_CONFIGURATION_REJ response to the IUT with Transaction = $Transaction_{IUT}$ = value provided with the AVDTP_SET_CONFIGURATION_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), Service_Category = any of the service categories provided in AVDT_Set_Configuration_Req, and Error_Code = 0x13 (SEP_IN_USE).
4. The IUT sends a SetConfigurationCfm_CB event to the Upper Tester with Transaction and FailingStreamConfigurationParameters set to valid values.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the Set_ConfigurationCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - FailingStreamConfigurationParameters = All the configuration parameters

4.3.1.30 Set Configuration Reject for an already configured SEP – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the SEP is already configured.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured correctly $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-08-C [Set Configuration Reject for an already configured SEP – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-08-C [Set Configuration Reject for an already configured SEP – ACP]

Table 4.32: Set Configuration Reject for an already configured SEP – ACP test cases

- Test Procedure

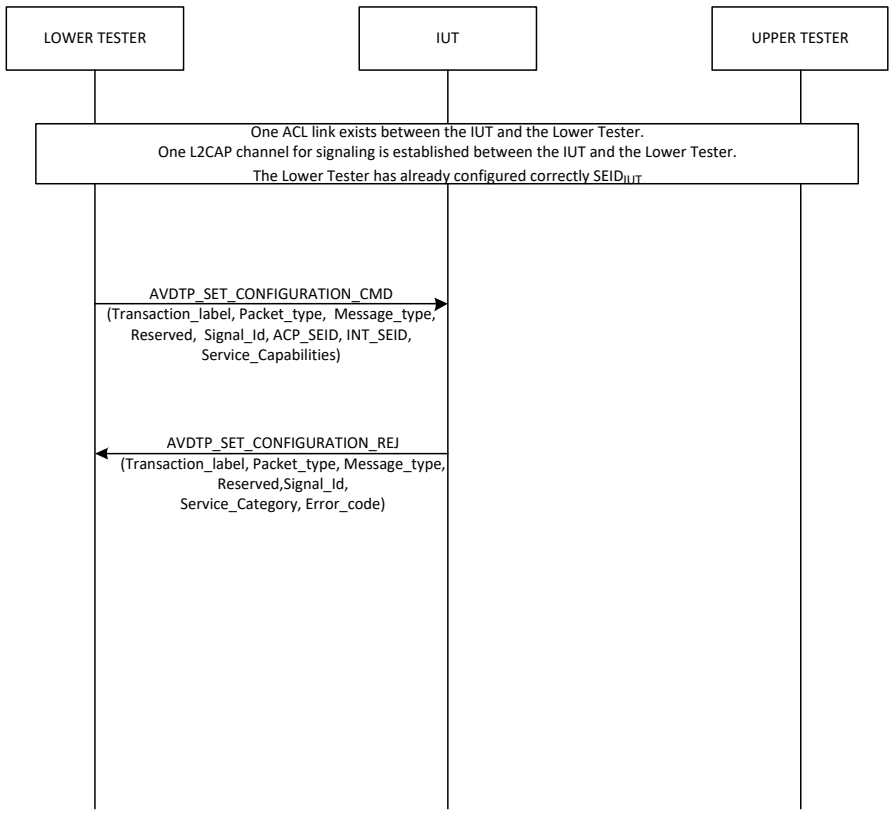

Figure 4.31: Set Configuration Reject for an already configured SEP – ACP MSC

1. The Lower Tester sends an AVDTP_ SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = the value of the IUT's SEP already configured, INT_SEID = any valid value, and Service_capabilities = any valid values.
2. The IUT sends an AVDTP_SET_CONFIGURATION_REJ response with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x13 (SEP_IN_USE)

4.3.1.31 Set Configuration Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Set Configuration command.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Upper Tester registers to AVDT_Set_Configuration_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-09-C [Set Configuration Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-09-C [Set Configuration Reject response – ACP]

Table 4.33: Set Configuration Reject response – ACP test cases

- Test Procedure

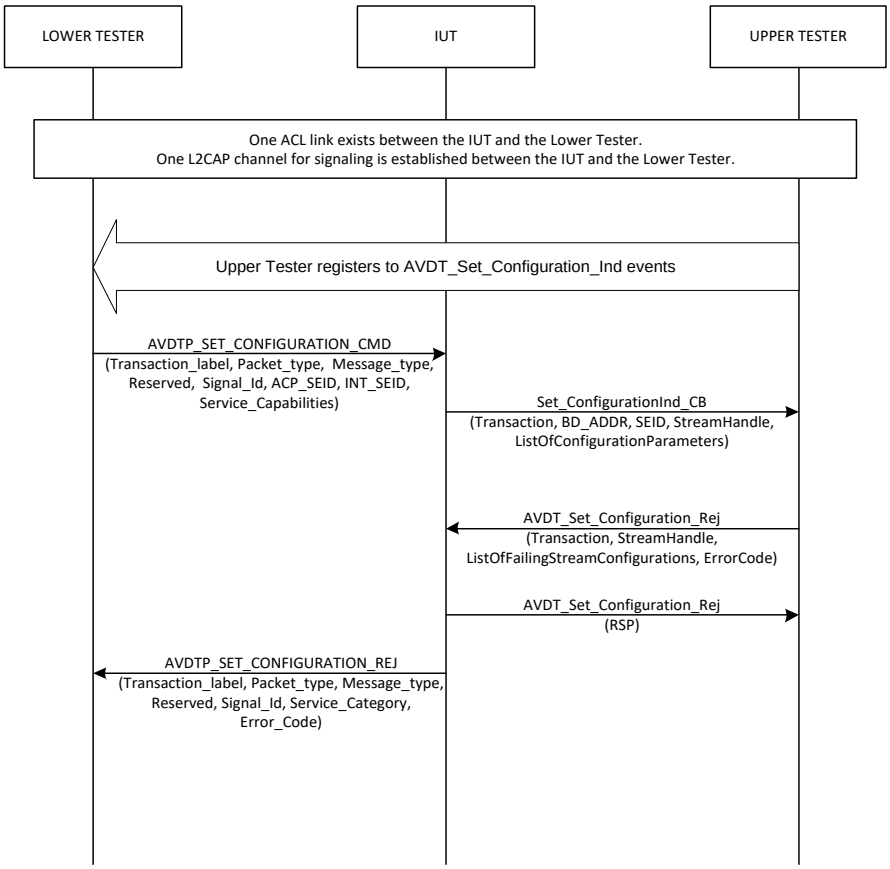

Figure 4.32: Set Configuration Reject response – ACP MSC

1. The Upper Tester registers to AVDT_Set_Configuration_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_ SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$, INT_SEID = any value, and Service_capabilities = valid values.
3. The IUT sends a Set_ConfigurationInd_CB event to the Upper Tester with Transaction, BD_ADDR, SEID, StreamHandle, and ListOfConfigurationParameters set to valid values.
4. The Upper Tester sends an AVDT_Set_Configuration_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle =$SH_{IUT}$ = the value provided by Set_ConfigurationInd_CB, SEID = $SEID_{IUT}$, ListOfFailingStreamConfigurations = the category provided in Service_capabilities of AVDTP_SET_CONFIGURATION_CMD, ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT sends an AVDT_Set_Configuration_Rej response to the Upper Tester with a valid RSP.
6. The IUT sends an AVDTP_SET_CONFIGURATION_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid value.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Service_Category = the value provided with AVDT_Set_Configuration_Rej
  - Error_Code = the value provided with AVDT_Set_Configuration_Rej

4.3.1.32 Detect rejection of a Get Configuration command by the ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of a Get Configuration command.

- Reference

  [3] 6.4, 6.9, 8.9, 13.1

  [5] 6.4, 6.10, 8.10, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured, see Section 4.3.1.5, Set Configuration command – INT.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-10-C [Detect rejection of a Get Configuration command by the ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-10-C [Detect rejection of a Get Configuration command by the ACP – INT]

Table 4.34: Detect rejection of a Get Configuration command by the ACP – INT test cases

- Test Procedure

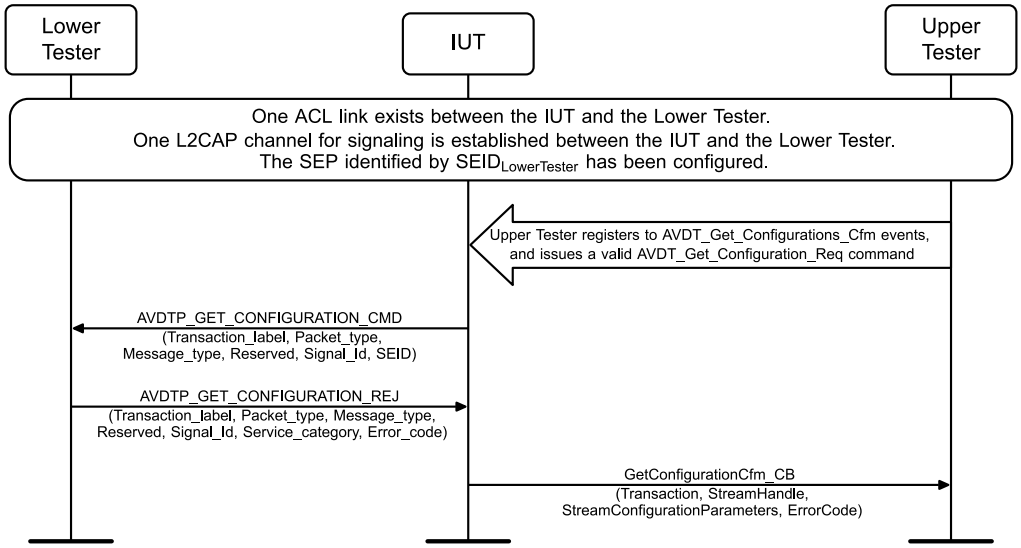

Figure 4.33: Detect rejection of a Get Configuration command by the ACP – INT MSC

1. The Upper Tester registers to AVDT_Get_Configuration_Cfm events with the IUT and issues a valid AVDT_Get_Configuration_Req command (see Section 4.3.1.7, Get Configuration command – INT), which the IUT acknowledges.
2. The IUT sends an AVDTP_GET_CONFIGURATION_CMD command with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
3. The Lower Tester sends an AVDTP_ SET_CONFIGURATION_REJ response value provided with the AVDTP_GET_CONFIGURATION_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION), and Error_Code = 0x12 (BAD_ACP_SEID).
4. The IUT sends a GetConfigurationCfm_CB even to the Upper Tester with Transaction, StreamHandle, StreamConfigurationParameters, and ErrorCode set to valid values.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the Get_ConfigurationCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - StreamHandle = SH_{IUT} = the value provided when the SEP was configured
  - StreamConfigurationParameters = the value provided with the AVDTP_GET_CONFIGURATION_REJ message
  - ErrorCode = the Error_Code provided with the AVDTP_GET_CONFIGURATION_REJ message

4.3.1.33 Invalid Get Configuration Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Get Configuration Reject response to the INT.

- Reference

  [3] 6.4, 6.9, 8.9, 13.1

[5] 6.4, 6.10, 8.10, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-11-C [Invalid Get Configuration Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-11-C [Invalid Get Configuration Reject response – ACP]

Table 4.35: Invalid Get Configuration Reject response – ACP test cases

- Test Procedure

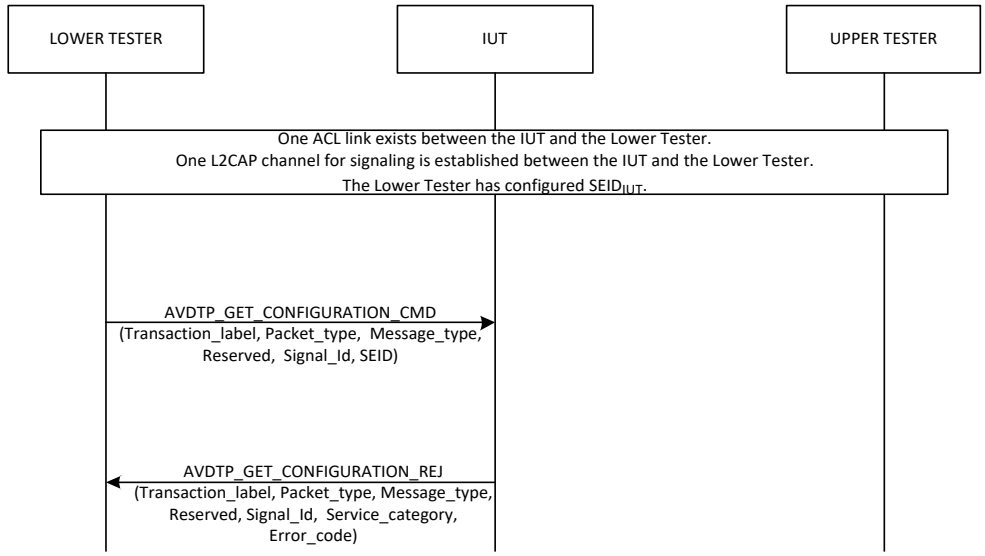

Figure 4.34: Invalid Get Configuration Reject response – ACP MSC

1. The Lower Tester sends an AVDTP_ GET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00. Message_type = 00 (command), Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION), and SEID = $SEID_{IUT}$ = an invalid value (SEID not existing on the IUT).
2. The IUT sends an AVDTP_GET_CONFIGURATION_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:
  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION)

- Service_Category = any value
- Error_code = 0x12 (BAD_ACP_SEID)

4.3.1.34 Get Configuration Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Get Configuration command.

- Reference

  [3] 6.4, 6.9, 8.9, 13.1

  [5] 6.4, 6.10, 8.10, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$.
  - The Upper Tester registers to AVDT_Get_Configuration_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-12-C [Get Configuration Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-12-C [Get Configuration Reject response – ACP]

Table 4.36: Get Configuration Reject response – ACP test cases

- Test Procedure

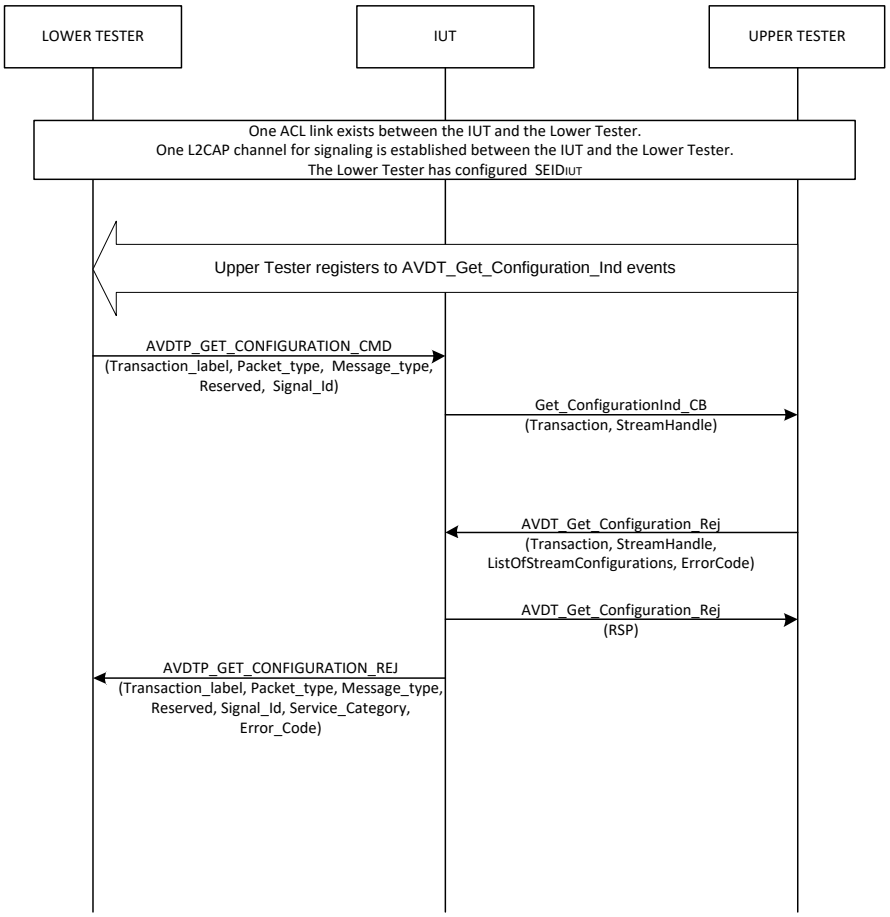

Figure 4.35: Get Configuration Reject response – ACP MSC

1. The Upper Tester registers to AVDT_Get_Configuration_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_ GET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION), and SEID = $SEID_{IUT}$.
3. The IUT sends the Get_ConfigurationInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
4. The Upper Tester sends an AVDT_Get_Configuration_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided when $SEID_{IUT}$ was configured, ListOfStreamConfigurations = any value, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT sends an AVDT_Get_Configuration_Rej response to the Upper Tester with valid RSP.
6. The IUT sends an AVDTP_GET_CONFIGURATION_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Upper Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x04 (AVDTP_GET_CONFIGURATION)
  - Error_Code = the value provided with AVDT_Get_Configuration_Rej

4.3.1.35 Detect rejection of the Reconfigure command by the ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of the Reconfigure command.

- Reference

  [3] 6.4, 6.14, 8.10, 13.1

  [5] 6.4, 6.15, 8.11, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured and is in the state given by SEP State in Table 4.37.

- Test Case Configuration

Test Case	SEP State
AVDTP/SRC/INT/SIG/SMG/BI-13-C [Detect rejection of the Reconfigure command by the ACP following Suspend – INT]	Configured, opened, started, and suspended
AVDTP/SNK/INT/SIG/SMG/BI-13-C [Detect rejection of the Reconfigure command by the ACP following Suspend – INT]	Configured, opened, started, and suspended
AVDTP/SRC/INT/SIG/SMG/BI-43-C [Detect rejection of the Reconfigure command by the ACP following Open – INT]	Configured, opened
AVDTP/SNK/INT/SIG/SMG/BI-43-C [Detect rejection of the Reconfigure command by the ACP following Open – INT]	Configured, opened

Table 4.37: Detect rejection of the Reconfigure command by the ACP – INT test cases

- Test Procedure

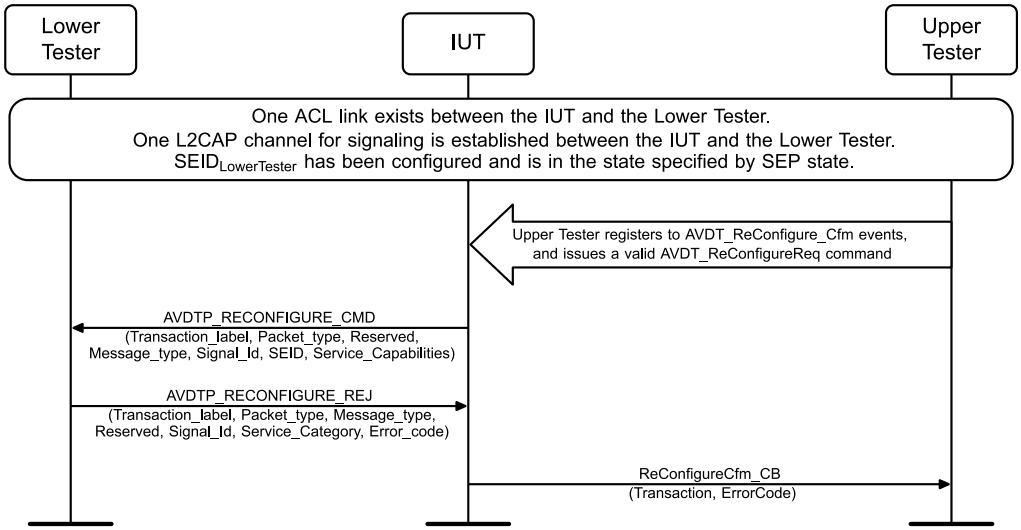

Figure 4.36: Detect rejection of the Reconfigure command by the ACP – INT MSC

1. The Upper Tester registers to AVDT_ReConfigure_Cfm events with the IUT and issues a valid AVDT_ReConfigure_Req command (see Section 4.3.1.9, Reconfigure command – INT), which the IUT acknowledges.
2. The IUT sends an AVDTP_RECONFIGURE_CMD command to the Lower Tester with Transaction_label, Packet_type, Reserved, Message_type, Signal_Id, SEID, and Service_Capabilities set to valid values.
3. The Lower Tester sends an AVDTP_ RECONFIGURE_REJ response to the IUT with Transaction_label = $Transaction_{IUT}$ = value provided with the AVDTP_RECONFIGURE_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x05 (AVDTP_RECONFIGURE), Service_Category = any of the service categories provided in AVDT_ReConfigure_Req, and Error_Code = 0X14 (SEP_NOT_IN_USE), 0X17 (BAD_SERV_CATEGORY).
4. The IUT sends a ReConfigureCfm_CB event to the Upper Tester with valid Transaction and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the ReConfigureCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = the Error_Code provided with the AVDTP_RECONFIGURE_REJ message

4.3.1.36 Reconfigure Reject response for invalid requests – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reconfigure Reject response to the INT.

- Reference

  [3] 6.4, 6.14, 8.10, 13.1

  [5] 6.4, 6.15, 8.11, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{IUT}$ has been configured and opened.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-14-C [Reconfigure Reject response for invalid requests – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-14-C [Reconfigure Reject response for invalid requests – ACP]

Table 4.38: Reconfigure Reject response for invalid requests – ACP test cases

- Test Procedure

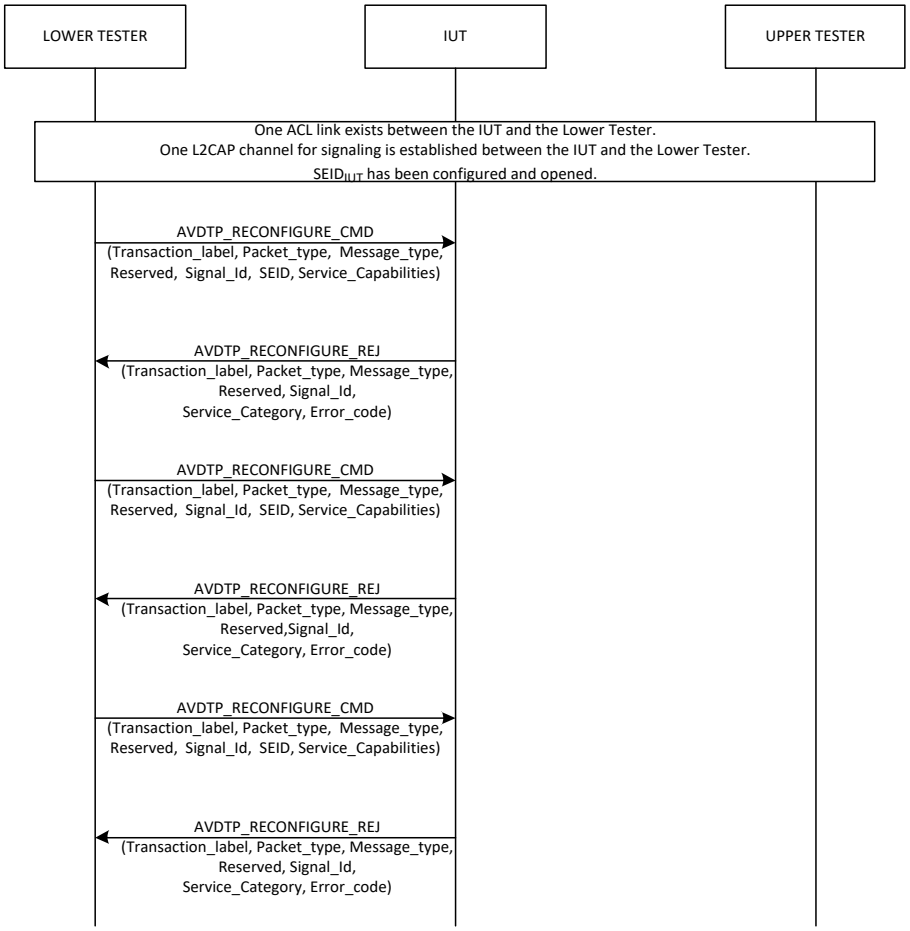

Figure 4.37: Reconfigure Reject response for invalid requests – ACP MSC

1. The Lower Tester sends an AVDTP_RECONFIGURE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x05 (AVDTP_RECONFIGURE), SEID = $SEID_{IUT}$, and Service_capabilities = Service category not defined in AVDTP (e.g., all the bits of the field defining the service category are set to 1).
2. The IUT sends an AVDTP_RECONFIGURE_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid values.

3. The Lower Tester sends an AVDTP_RECONFIGURE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x05 (AVDTP_RECONFIGURE), SEID = Invalid value (i.e., not corresponding to an already configured SEP), and Service_capabilities = valid media transport capabilities.
4. The IUT sends an AVDTP_RECONFIGURE_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid values.
5. The Lower Tester sends an AVDTP_RECONFIGURE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x05 (AVDTP_RECONFIGURE), SEID = $SEID_{IUT}$, and Service_capabilities = valid media transport capabilities.
6. The IUT sends an AVDTP_RECONFIGURE_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)
  - Service_Category = the value provided in AVDTP_RECONFIGURE_CMD
  - Error_code = 0x17 (BAD_SERV_CATEGORY)

  In Step 4, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)
  - Service_Category = 0
  - Error_code = 0x14 (SEP_NOT_IN_USE) or 0x12 (BAD_ACP_SEID)

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)
  - Service_Category = the value provided in AVDTP_RECONFIGURE_CMD
  - Error_code = 0x1A (INVALID_CAPABILITIES)

4.3.1.37 Reject Reconfigure request – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Reconfigure command.

- Reference

  [3] 6.4, 6.14, 8.10, 13.1

  [5] 6.4, 6.15, 8.11, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{IUT}$ has been configured and is in the state given by SEP State in Table 4.39.
  - The Upper Tester registers to AVDT_Reconfigure_Ind events.

- Test Case Configuration

Test Case	SEP state
AVDTP/SRC/ACP/SIG/SMG/BI-15-C [Reject Reconfigure request following Suspend – ACP]	Configured, opened, started, and suspended by the Lower Tester
AVDTP/SNK/ACP/SIG/SMG/BI-15-C [Reject Reconfigure request following Suspend – ACP]	Configured, opened, started, and suspended by the Lower Tester
AVDTP/SRC/ACP/SIG/SMG/BI-45-C [Reject Reconfigure request without Suspend – ACP]	Configured, opened by the Lower Tester
AVDTP/SNK/ACP/SIG/SMG/BI-45-C [Reject Reconfigure request without Suspend – ACP]	Configured, opened by the Lower Tester

Table 4.39: Reject Reconfigure Request – ACP test cases

- Test Procedure

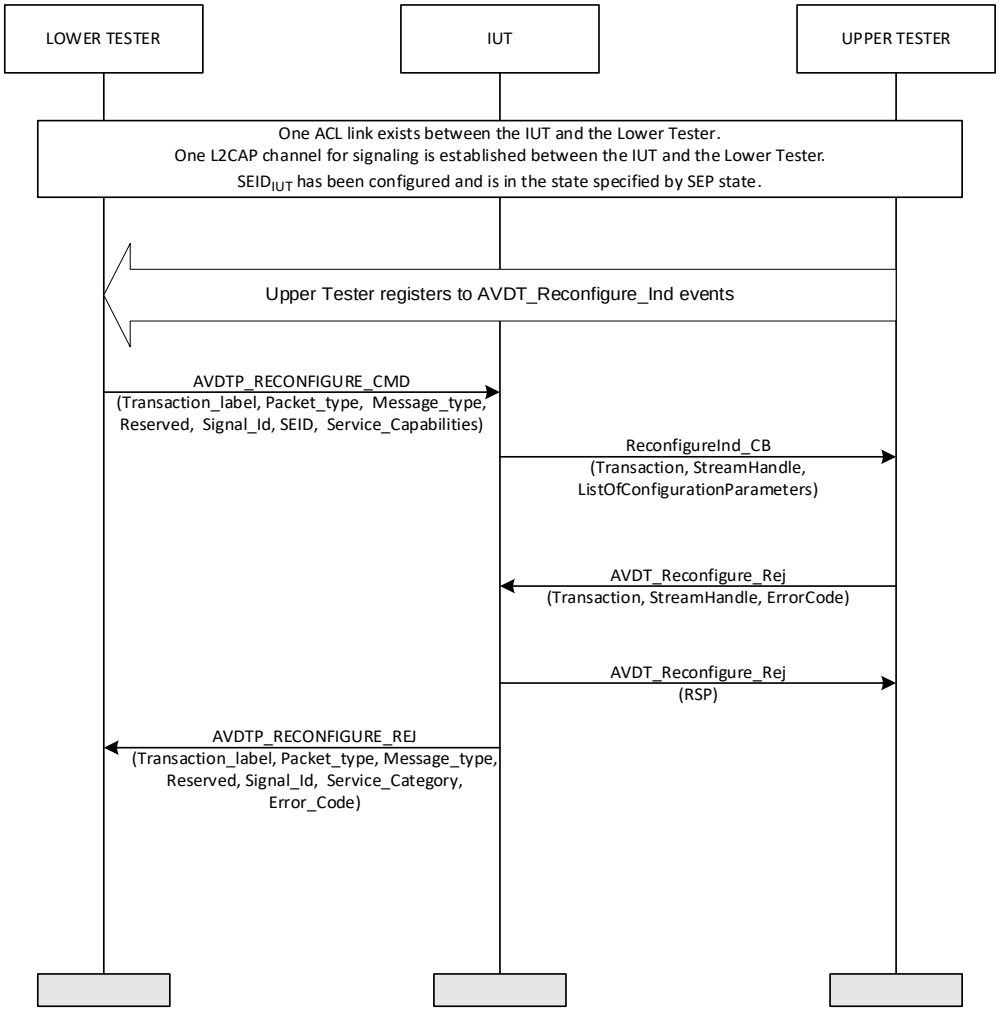

Figure 4.38: Reject Reconfigure request – ACP MSC

1. The Upper Tester registers to AVDT_Reconfigure_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_RECONFIGURE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x05 (AVDTP_RECONFIGURE), SEID = $SEID_{IUT}$, and Service_capabilities = valid values.
3. The IUT sends a ReconfigureInd_CB event to the Upper Tester with Transaction, StreamHandle, and ListOfConfigurationParameters set to valid values.
4. The Upper Tester sends an AVDT_Reconfigure_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided by ReconfigureInd_CB, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT sends an AVDT_Reconfigure_Rej response to the Upper Tester with a valid RSP.
6. The IUT sends an AVDTP_RECONFIGURE_REJ response to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, Service_Category, and Error_Code set to valid values.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x05 (AVDTP_RECONFIGURE)
  - Error_Code = the value provided with AVDT_Reconfigure_Rej

4.3.1.38 Detect the rejection of an Open command by ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of an Open Stream command.

- Reference

  [3] 6.4, 6.10, 8.11, 13.1

  [5] 6.4, 6.11, 8.12, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) with any valid parameters, see Section 4.3.1.5, Set Configuration command – INT.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-16-C [Detect the rejection of an Open command by ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-16-C [Detect the rejection of an Open command by ACP – INT]

Table 4.40: Detect the rejection of an Open command by ACP – INT test cases

- Test Procedure

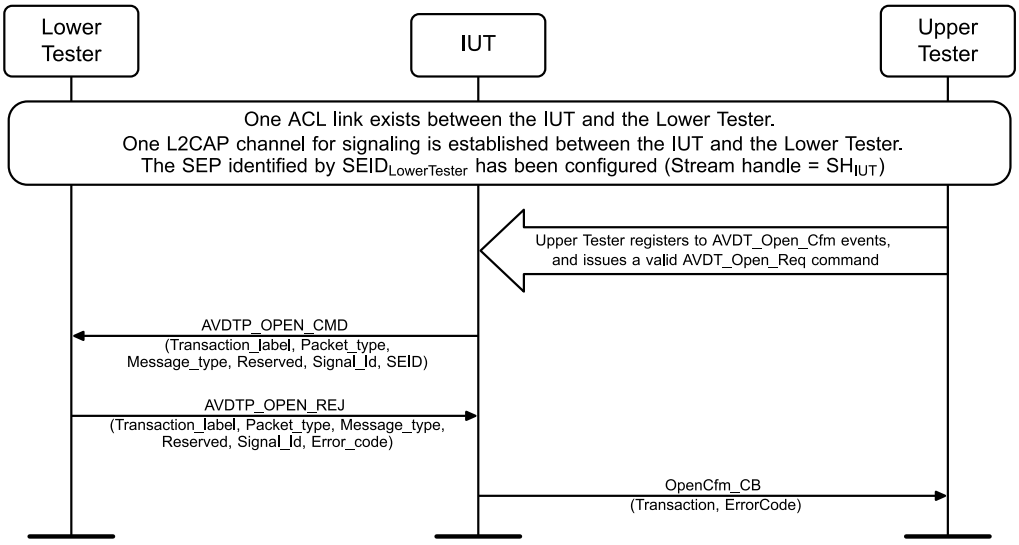

Figure 4.39: Detect the rejection of an Open command by ACP – INT MSC

1. The Upper Tester registers to AVDT_Open_Cfm events with the IUT and issues a valid AVDT_Open_Req command (see Section 4.3.1.11, Open command – INT), which the IUT acknowledges.
2. The IUT sends an AVDTP_OPEN_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
3. The Lower Tester sends an AVDTP_OPEN_REJ response to the IUT with Transaction_label = $Transaction_{IUT}$ = value provided with the AVDTP_OPEN _CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x06 (AVDTP_OPEN), and Error_Code = 0x31 (BAD_STATE).
4. The IUT sends an OpenCfm_CB event to the Upper Tester with valid Transaction and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the OpenCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = the Error_Code provided with the AVDTP_OPEN_REJ message

4.3.1.39 Invalid Open Stream Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue an Open Stream Reject response to the INT.

- Reference

  [3] 6.4, 6.10, 8.11, 13.1

  [5] 6.4, 6.11, 8.12, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not configured the SEP identified by $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-17-C [Invalid Open Stream Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-17-C [Invalid Open Stream Reject response – ACP]

Table 4.41: Invalid Open Stream Reject response – ACP test cases

- Test Procedure

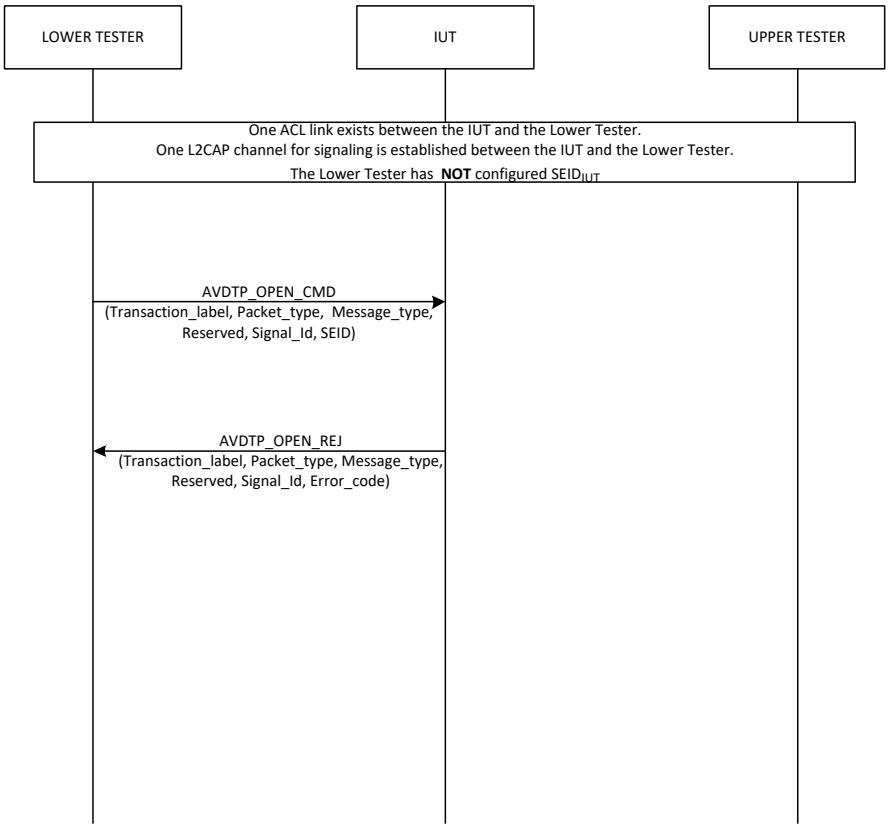

Figure 4.40: Invalid Open Stream Reject response – ACP MSC

1. The Lower Tester sends an AVDTP_OPEN_CMD command to the IUT with Transaction_label = Transaction$_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x06 (AVDTP_OPEN), and SEID = SEID$_{IUT}$ = the value used for configuring the SEP. The reserved bits following SEID are set to '00'.
2. The IUT responds to the Lower Tester with AVDTP_OPEN_REJ with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Upper Tester with the fields:

  - Transaction_label = Transaction$_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x06 (AVDTP_OPEN)
  - Error_code = 0x31 (BAD_STATE)

4.3.1.40 Reject Open Stream command response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to an Open Stream command.

- Reference

  [3] 6.4, 6.10, 8.11, 13.1

  [5] 6.4, 6.11, 8.12, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (Stream handle = SH_{IUT}).
  - The Upper Tester registers to AVDT_Open_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-18-C [Reject Open Stream command response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-18-C [Reject Open Stream command response – ACP]

Table 4.42: Reject Open Stream command response – ACP test cases

- Test Procedure

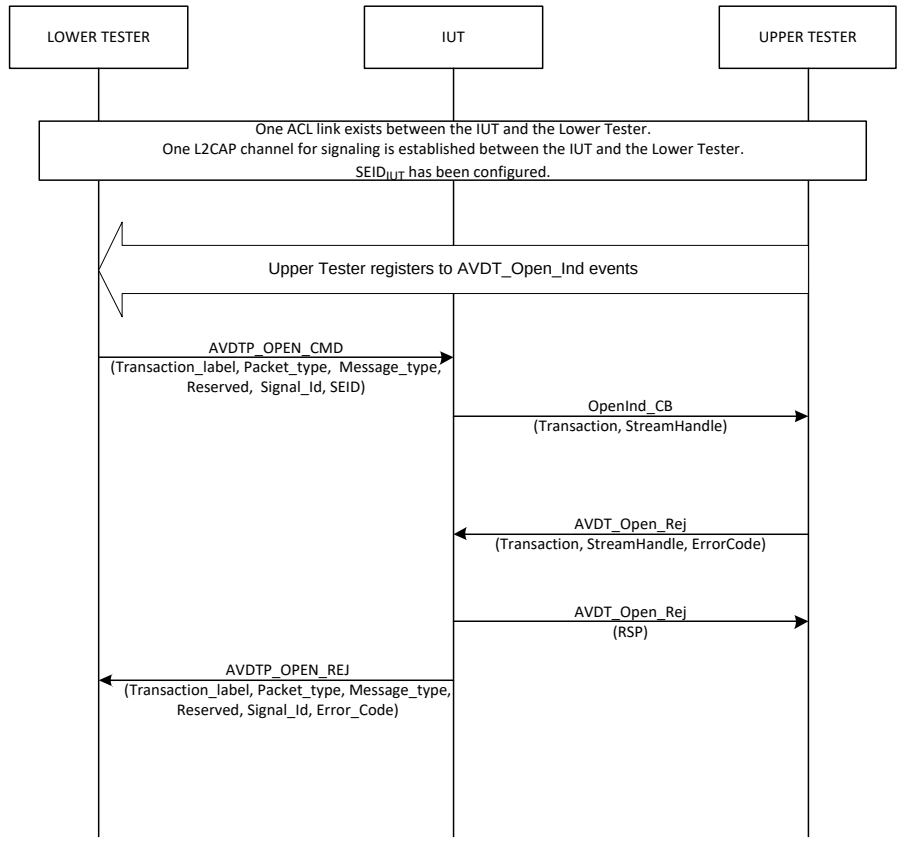

Figure 4.41: Reject Open Stream command response – ACP MSC

1. The Lower Tester sends an AVDTP_OPEN_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x06 (AVDTP_OPEN), and SEID = $SEID_{IUT}$.
2. The IUT sends an OpenInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
3. The Upper Tester sends an AVDT_Open_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided by OpenInd_CB, and ErrorCode = any value in the range 0xC0-0xFF.
4. The IUT responds to the Upper Tester with AVDT_Open_Rej with RSP.
5. The IUT sends an AVDTP_OPEN_REJ command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 5, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x06 (AVDTP_OPEN)
  - Error_Code = the value provided with AVDT_Open_Rej

AVDTP/SRC/INT/SIG/SMG/BI-19-C [Detect the rejection of the Start Stream command by the ACP – INT]

- Test Purpose

  Verify that the IUT (SRC, INT) is able to report the rejection by the remote device of the Start Stream command.

- Reference

  [3] 6.4, 6.11, 8.12, 13.1

  [5] 6.4, 6.12, 8.13, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) with valid parameters, see Section 4.3.1.5, Set Configuration command – INT.
  - A stream connection to the SEP identified by $SEID_{LowerTester}$ (Stream handle = SH_{IUT}) has been established, see Section 4.3.1.11, Open command – INT.

- Test Procedure

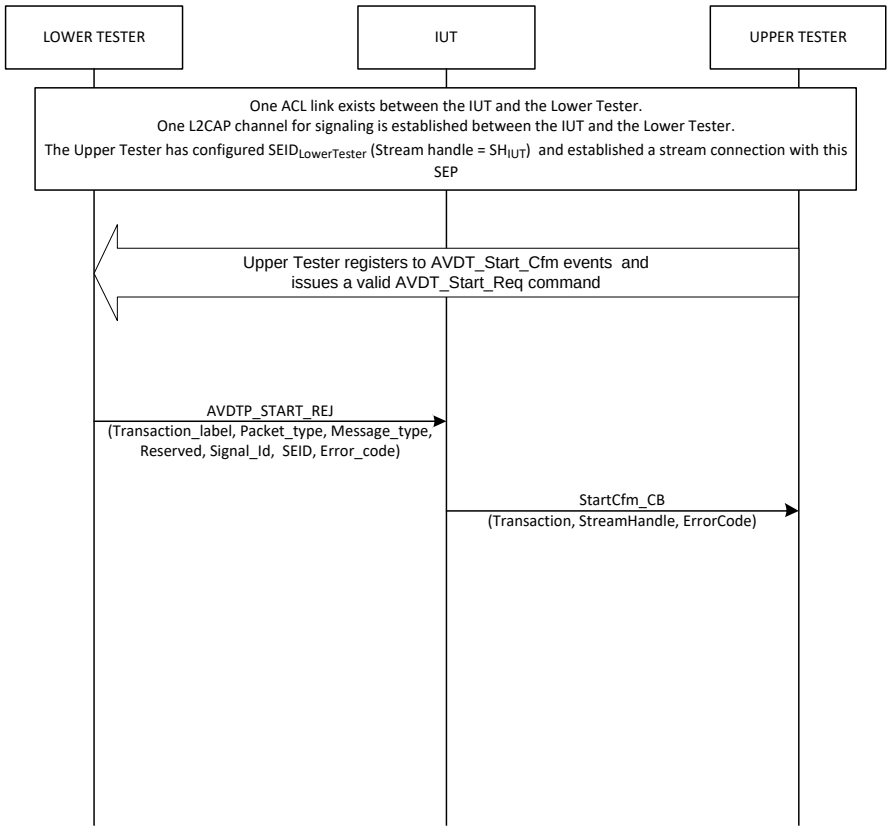

Figure 4.42: AVDTP/SRC/INT/SIG/SMG/BI-19-C

1. The Upper Tester registers to AVDT_Start_Cfm events with the IUT and issues a valid AVDT_Start_Req command (see AVDTP/SRC/INT/SIG/SMG/BV-17-C [Start a Streaming procedure – INT]), which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_START_REJ command to the IUT with Transaction_label = TransactionIUT = value provided with the AVDTP_START_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x07 (AVDTP_START), SEID = SEIDLowerTester, and Error_Code = 0x31 (BAD_STATE).
3. The IUT sends a StartCfm_CB event to the Upper Tester with Transaction, StreamHandle, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 3, the IUT invokes the StartCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - StreamHandle = SH_{IUT}
  - ErrorCode = the Error_Code provided with the AVDTP_START_REJ message

4.3.1.41 Start Stream Reject response when not in Open state – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Start Stream Reject response to the INT.

- Reference

  [3] 6.4, 6.11, 8.12, 13.1

  [5] 6.4, 6.12, 8.13, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
  - The SEP identified by $SEID_{IUT}$ is not in the Open state.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-20-C [Start Stream Reject response when not in Open state – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-20-C [Start Stream Reject response when not in Open state – ACP]

Table 4.43: Start Stream Reject response when not in Open state – ACP test cases

- Test Procedure

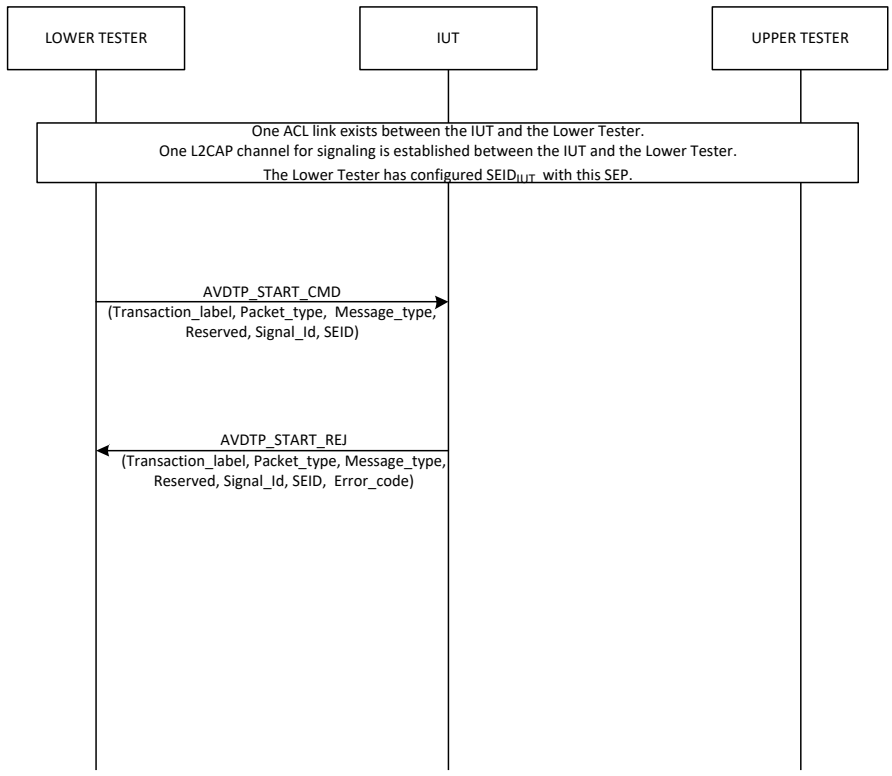

Figure 4.43: Start Stream Reject response when not in Open state – ACP MSC

1. The Lower Tester sends an AVDTP_START_CMD command with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x07 (AVDTP_START), and SEID = $SEID_{IUT}$ = the value used for configuring the SEP.
2. The IUT responds to the Lower Tester with AVDTP_START_REJ with Transaction_label, Packet_Type, Message_type, Reserved, Signal_Id, SEID, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0X07 (AVDTP_START)
  - SEID = $SEID_{IUT}$
  - Error_code = 0x31 (BAD_STATE)

4.3.1.42 Reject Start Stream command response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Start Stream command.

- Reference

  [3] 6.4, 6.11, 8.12, 13.1

  [5] 6.4, 6.12, 8.13, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (Stream handle = SH_{IUT}), see Section 4.3.1.6, Set Configuration response – ACP.
  - The Lower Tester has established a stream connection with the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.
  - The Upper Tester registers to AVDT_Start_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-21-C [Reject Start Stream command response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-21-C [Reject Start Stream command response – ACP]

Table 4.44: Reject Start Stream command response – ACP test cases

- Test Procedure

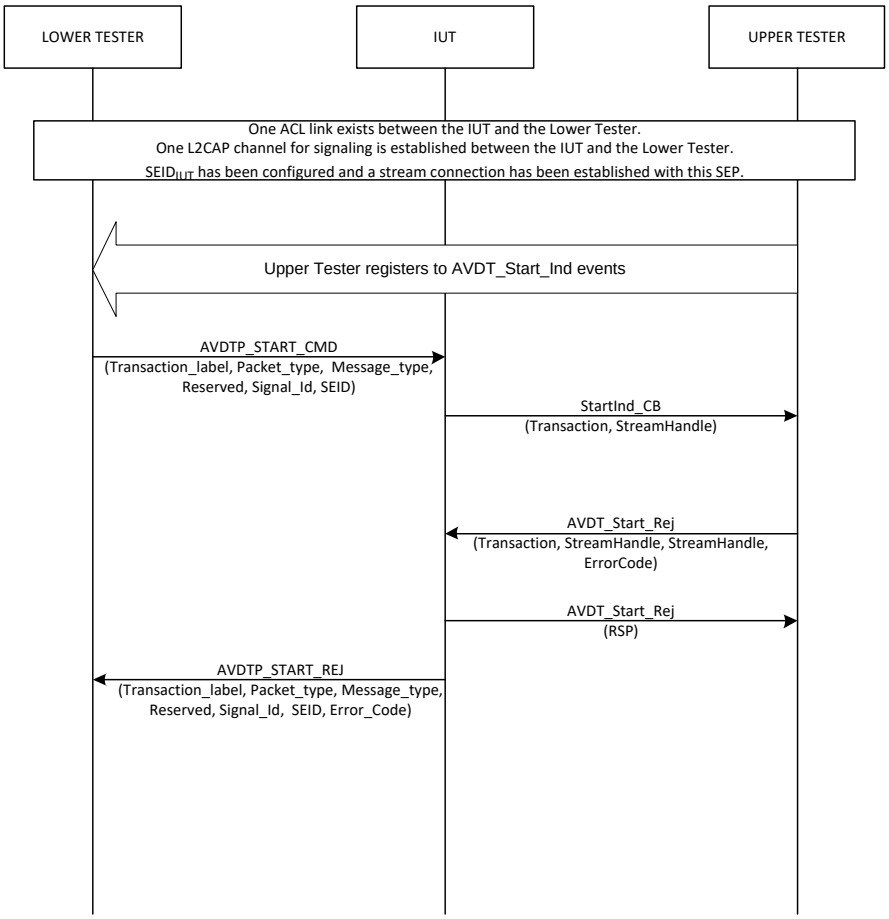

Figure 4.44: Reject Start Stream command response – ACP MSC

1. The Upper Tester registers to AVDT_Start_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_START_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x07 (AVDTP_START), and SEID = $SEID_{IUT}$.
3. The IUT sends a StartInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
4. The Upper Tester sends an AVDT_Start_Rej service primitive with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided by StartInd_CB, StreamHandle = SH_{IUT} = the value provided by StartInd_CB, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT responds to the Upper Tester with AVDT_Start_Rej with RSP.
6. The IUT sends an AVDTP_START_REJ command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)

- Signal_Id = 0x07 (AVDTP_START)
- SEID = $SEID_{IUT}$
- Error_Code = the value provided with AVDT_Start_Rej

4.3.1.43 Detect the rejection of the Close Stream command by the ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to report the rejection by the remote device of the Close Stream command.

- Reference

  [3] 6.4, 6.12, 8.13, 13.1

  [5] 6.4, 6.13, 8.14, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{LowerTester}$ has been configured (Stream handle = SH_{IUT}) with any valid parameters, see Section 4.3.1.7, Set Configuration command – INT.
  - A connection to the SEP identified by $SEID_{LowerTester}$ has been established, see Section 4.3.1.11, Open command – INT.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-22-C [Detect the rejection of the Close Stream command by the ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-22-C [Detect the rejection of the Close Stream command by the ACP – INT]

Table 4.45: Detect the rejection of the Close Stream command by the ACP – INT test cases

- Test Procedure

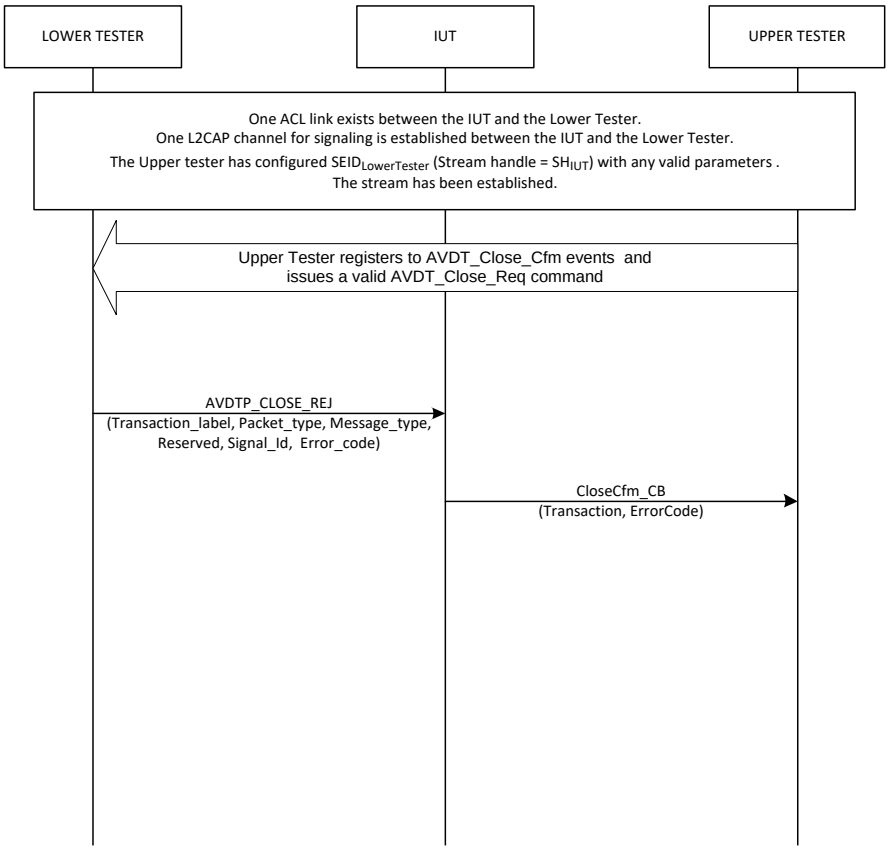

Figure 4.45: Detect the rejection of the Close Stream command by the ACP – INT MSC

1. The Upper Tester registers to AVDT_Close_Cfm events with the IUT and issues a valid AVDT_Close_Req command, see Section 4.3.1.14, Close command – INT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_CLOSE_REJ command to the IUT with Transaction_label = $Transaction_{IUT}$ = value provided with the AVDTP_CLOSE _CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x08 (AVDTP_CLOSE), and Error_Code = 0x31 (BAD_STATE).
3. The IUT sends a CloseCfm_CB event to the Upper Tester with Transaction and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 3, the IUT invokes the CloseCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = the Error_Code provided with the AVDTP_CLOSE_REJ message

4.3.1.44 Close Stream Reject response to invalid SEP – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Close Stream Reject response to the Lower Tester (INT).

- Reference

  [3] 6.4, 6.12, 8.13, 13.1

  [5] 6.4, 6.13, 8.14, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured and established the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-23-C [Close Stream Reject response to invalid SEP – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-23-C [Close Stream Reject response to invalid SEP – ACP]

Table 4.46: Close Stream Reject response to invalid SEP – ACP test cases

- Test Procedure

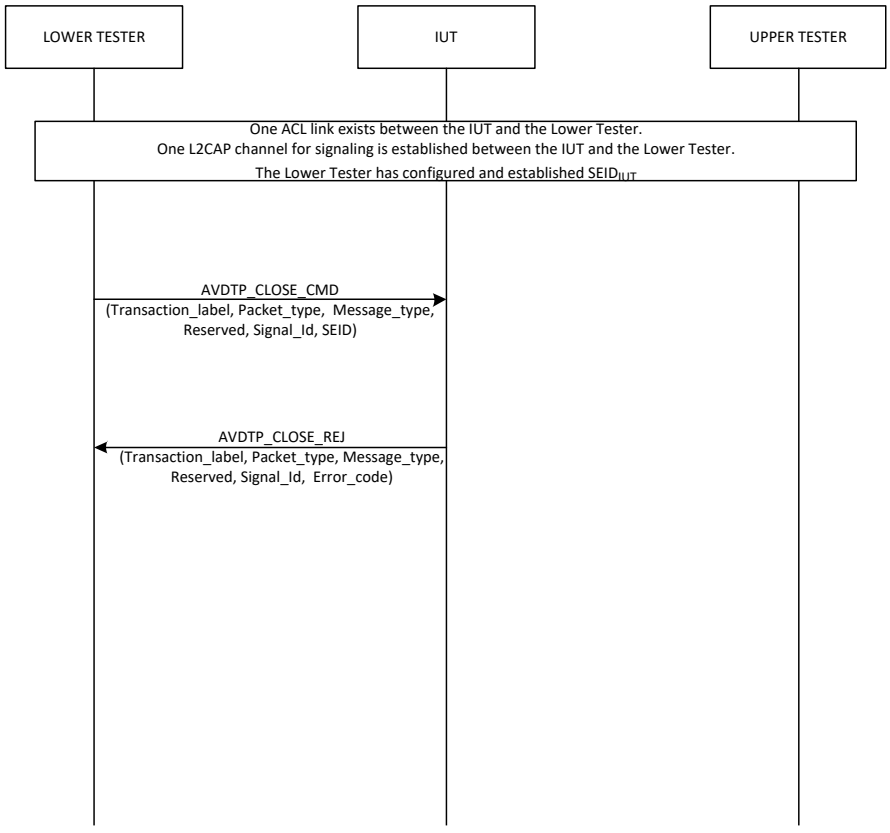

Figure 4.46: Close Stream Reject response to invalid SEP – ACP MSC

1. The Lower Tester sends an AVDTP_CLOSE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x08 (AVDTP_CLOSE), and SEID = $SEID_{IUT}$ = any value, except the one used for configuring and establishing the SEP.
2. The IUT responds to the Lower Tester with AVDTP_CLOSE_REJ with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x08 (AVDTP_CLOSE)
  - Error_code = 0x12 (BAD_ACP_SEID)

4.3.1.45 Reject Close Stream command response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Close Stream command.

- Reference

  [3] 6.4, 6.12, 8.13, 13.1

  [5] 6.4, 6.13, 8.14, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured and established the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.
  - The Upper Tester registers to AVDT_Close_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-24-C [Reject Close Stream command response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-24-C [Reject Close Stream command response – ACP]

Table 4.47: Reject Close Stream command response – ACP test cases

- Test Procedure

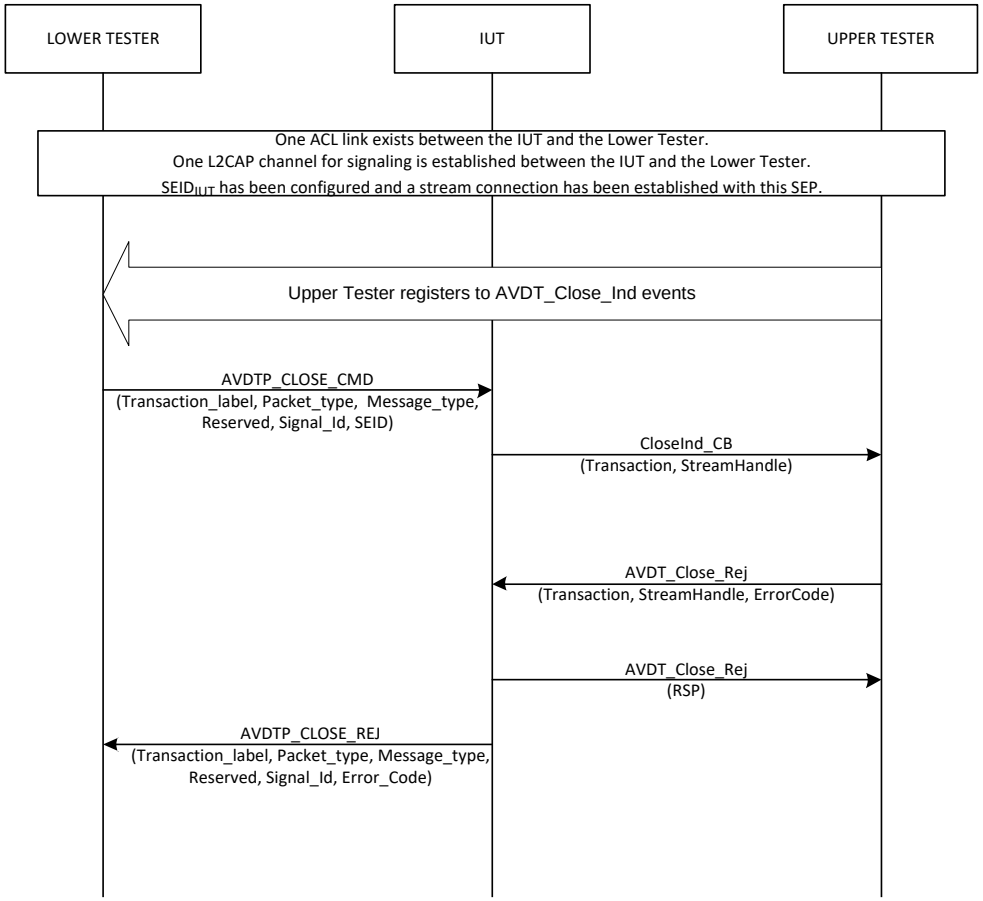

Figure 4.47: Reject Close Stream command response – ACP MSC

1. The Upper Tester registers to AVDT_Close_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_CLOSE_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x08 (AVDTP_CLOSE), and SEID = $SEID_{IUT}$.
3. The IUT sends a CloseInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
4. The Upper Tester sends an AVDT_Close_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided by CloseInd_CB, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT responds to the Upper Tester with AVDT_Close_Rej with RSP.
6. The IUT sends an AVDTP_CLOSE_REJ with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error Code.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)

- Signal_Id = 0x08 (AVDTP_CLOSE)
- Error_code = the value provided with AVDT_Close_Rej

AVDTP/SRC/INT/SIG/SMG/BI-25-C [Detect rejection of a Suspend Stream command by the ACP – INT]

- Test Purpose

  Verify that the IUT (SRC, INT) is able to report to the Upper Tester the rejection of a Suspend Stream command sent by the Lower Tester (ACP).

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel..
  - The Upper Tester has configured the SEP identified by $SEID_{LowerTester}$ (Stream handle = SH_{IUT}) with valid parameters, see Section 4.3.1.5, Set Configuration command – INT.
  - A stream connection to the SEP identified by $SEID_{LowerTester}$ has been established (Stream handle = SH_{IUT}) and started the stream, see AVDTP/SRC/INT/SIG/SMG/BV-17-C [Start a Streaming procedure – INT].

- Test Procedure

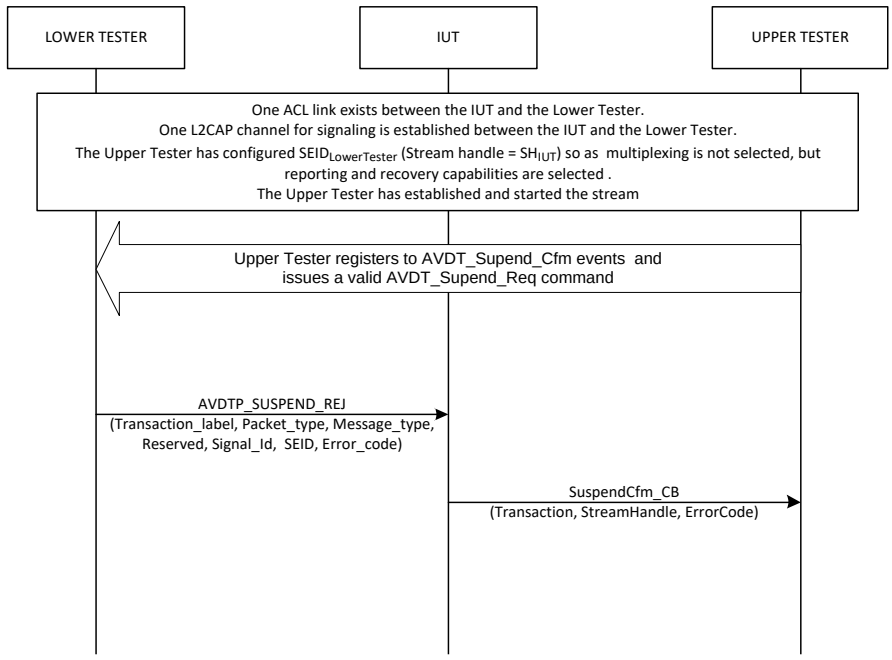

Figure 4.48: AVDTP/SRC/INT/SIG/SMG/BI-25-C MSC

1. The Upper Tester registers to AVDT_Suspend_Cfm events with the IUT and issues a valid AVDT_Suspend_Req command via AVDTP/SRC/INT/SIG/SMG/BV-21-C [Suspend command – INT], which the IUT acknowledges, or by other means.
2. The Lower Tester sends an AVDTP_SUSPEND_REJ command to the IUT with Transaction_label = $Transaction_{IUT}$ = value provided with the AVDTP_SUSPEND_CMD message, Packet_type = 00

(single packet), Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x09 (AVDTP_SUSPEND), SEID = $SEID_{LowerTester}$, and Error_Code = 0x31 (BAD_STATE).

3. The IUT sends a SuspendCfm_CB event to the Upper Tester with Transaction, StreamHandle, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 3, the IUT invokes the SuspendCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - StreamHandle = SH_{IUT}
  - ErrorCode = the Error_Code provided with the AVDTP_SUSPEND_REJ message

4.3.1.46 Reject Suspend Stream response when the stream is not in a correct state – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Suspend Stream Reject response to the Lower Tester (INT).

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
  - The Lower Tester has established a stream connection to the SEP identified by $SEID_{IUT}$, see Section 4.3.1.12, Open command – ACP.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-26-C [Reject Suspend Stream response when the stream is not in a correct state – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-26-C [Reject Suspend Stream response when the stream is not in a correct state – ACP]

Table 4.48: Reject Suspend Stream response when the stream is not in a correct state – ACP test cases

- Test Procedure

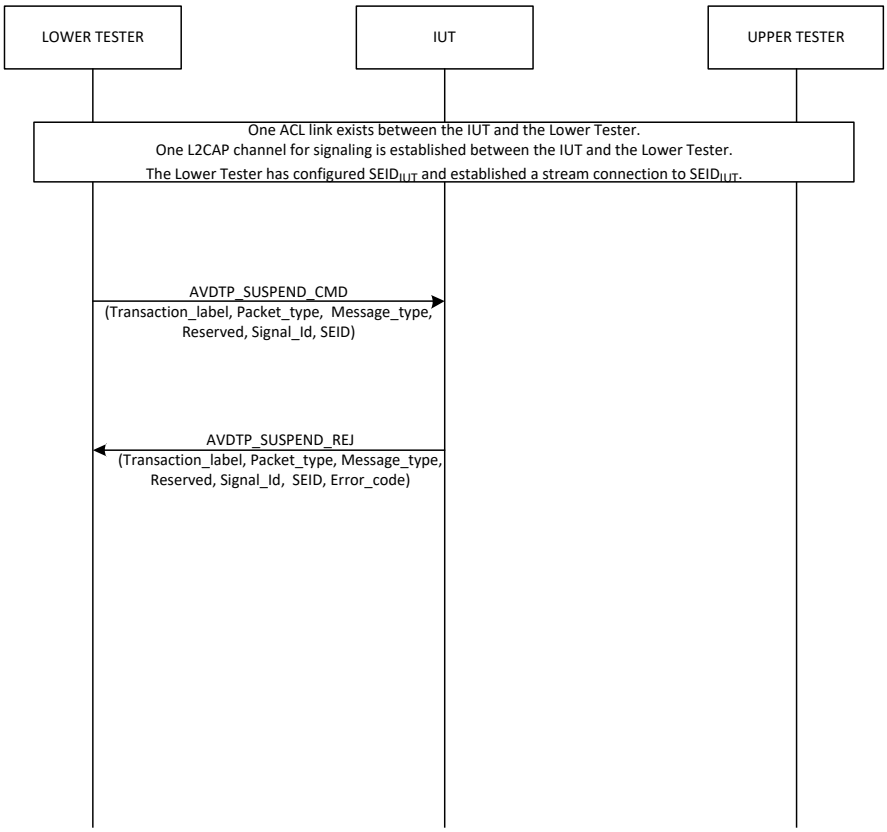

Figure 4.49: Reject Suspend Stream response when the stream is not in a correct state – ACP MSC

1. The Lower Tester sends an AVDTP_SUSPEND_CMD command to the IUT with Transaction_label = Transaction$_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x09 (AVDTP_SUSPEND), and SEID = SEID$_{IUT}$ = the value used for configuring the SEP.
2. The IUT responds to the Lower Tester with AVDTP_SUSPED_REJ with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = Transaction$_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x09 (AVDTP_SUSPEND)
  - SEID = SEID$_{IUT}$
  - Error_Code = 0x31 (BAD_STATE)

4.3.1.47 Reject Suspend Stream command response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Suspend Stream command from the Lower Tester (INT).

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$ (Stream handle = SH_{IUT}), see Section 4.3.1.8, Set Configuration response – ACP.
  - The Lower Tester has established a stream connection to the SEP identified by $SEID_{IUT}$ and started streaming, see Section 4.3.1.13, Receive Start command – ACP.
  - The Upper Tester registers to AVDT_Suspend_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-27-C [Reject Suspend Stream command response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-27-C [Reject Suspend Stream command response – ACP]

Table 4.49: Reject Suspend Stream command response – ACP test cases

- Test Procedure

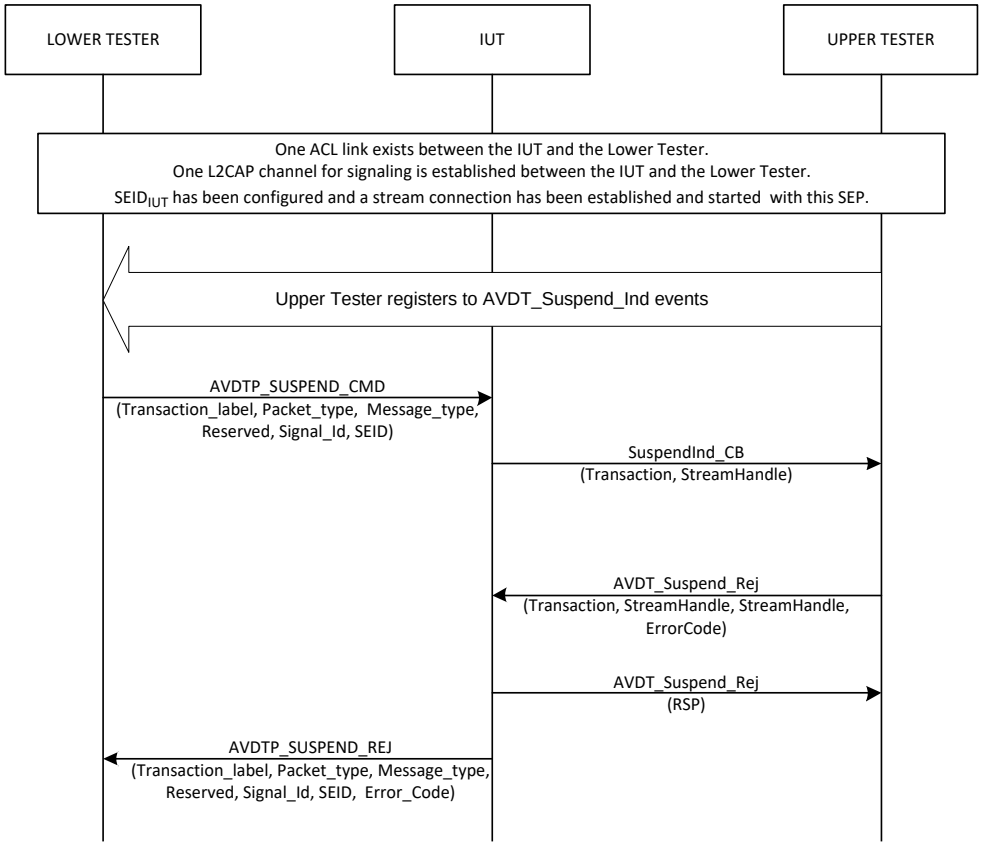

Figure 4.50: Reject Suspend Stream command response – ACP MSC

1. The Upper Tester registers to AVDT_Suspend_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_SUSPEND_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x09 (AVDTP_SUSPEND), and SEID = $SEID_{IUT}$.
3. The IUT sends a SuspendInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
4. The Upper Tester sends an AVDT_Suspect_Rej service primitive to the IUT with Transaction = $Transaction_{LowerTester}$, StreamHandle = SH_{IUT} = the value provided by SuspendInd_CB, StreamHandle = SH_{IUT} = the value provided by SuspendInd_CB, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT responds to the Upper Tester with AVDT_Suspend_Rej with RSP.
6. The IUT sends an AVDTP_SUSPEND_REJ command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)

- Signal_Id = 0x09 (AVDTP_SUSPEND)
- SEID = $SEID_{IUT}$
- Error_Code = the value provided with AVDT_Suspend_Rej

4.3.1.48 General Reject response – does not include signal ID – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a general Reject response to the Lower Tester (INT).

- Reference

  [3] 8.17, 13.1

  [5] 8.18, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-28-C [General Reject response – does not include signal ID – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-28-C [General Reject response – does not include signal ID – ACP]

Table 4.50: General Reject response – does not include signal ID test cases

- Test Procedure

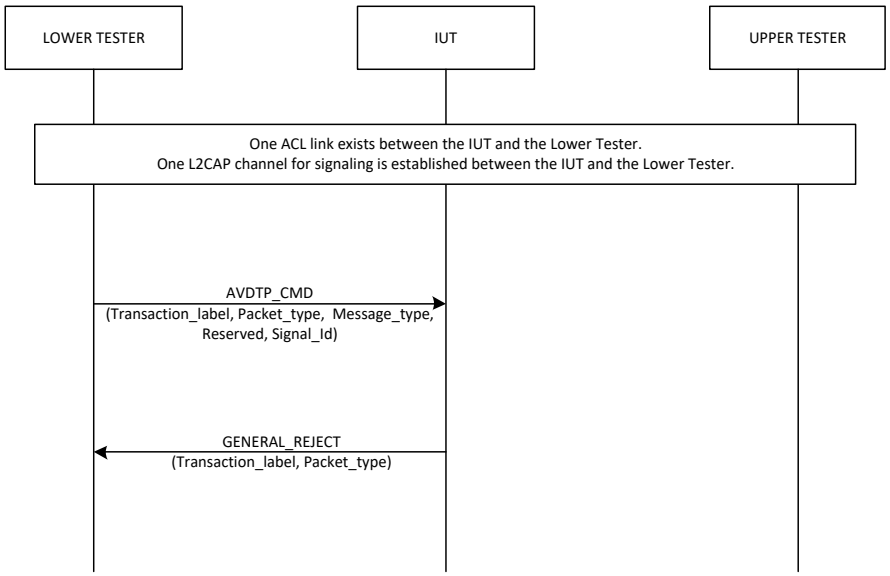

Figure 4.51: General Reject response – does not include signal ID – ACP MSC

1. The Lower Tester sends an AVDTP_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x3F (Undefined Signal ID).
2. The IUT responds to the Lower Tester with GENERAL_REJECT with valid Transaction_label and Packet_type.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 01 (general reject)
  - Other 10 bits = All set to '0'.

4.3.1.49 Refuse further streaming following Suspend – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to refuse further streaming after the Streaming procedure has been suspended.

- Reference

  [3] 6.4, 6.13, 8.14, 13.1

  [5] 6.4, 6.14, 8.15, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has configured the SEP identified by $SEID_{IUT}$, see Section 4.3.1.6, Set Configuration response – ACP.
  - The Lower Tester has established a stream connection to the SEP identified by $SEID_{IUT}$ and started the stream, see Section 4.3.1.13, Receive Start command – ACP.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-29-C [Refuse further streaming following Suspend – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-29-C [Refuse further streaming following Suspend – ACP]

Table 4.51: Refuse further streaming following Suspend – ACP test cases

- Test Procedure

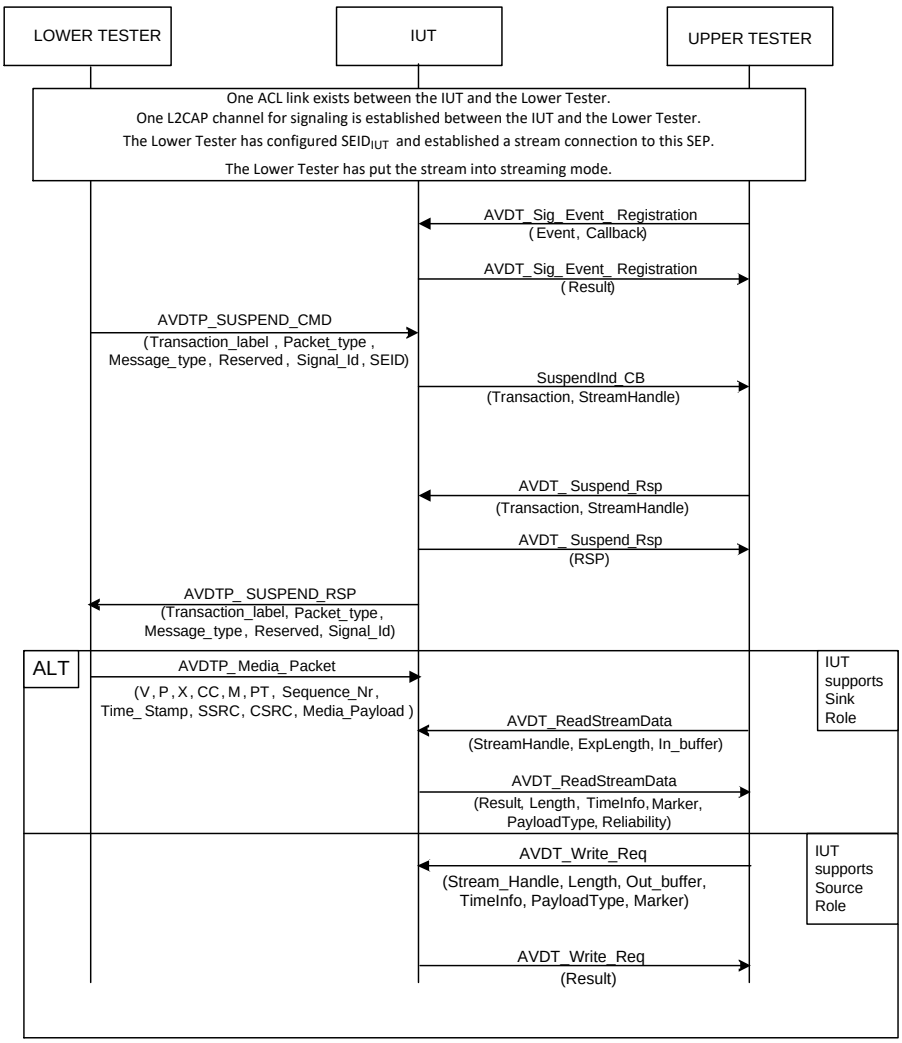

Figure 4.52: Refuse further streaming following Suspend – ACP MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration service primitive to the IUT with Event = AVDT_Suspend_Ind and Callback = SuspendInd_CB.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Lower Tester sends an AVDTP_SUSPEND_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x09 (AVDTP_SUSPEND), and SEID = $SEID_{IUT}$ = the value used for configuring the SEP.
4. The IUT sends a SuspendingInd_CB event to the Upper Tester with valid Transaction and StreamHandle.
5. The Upper Tester sends an AVDT_Suspend_Rsp service primitive with Transaction = $Transaction_{LowerTester}$ and Stream_Handle = SH_{IUT} = the value corresponding to $SEID_{IUT}$.
6. The IUT responds to the Upper Tester with AVDT_Suspend_Rsp with RSP.
7. The IUT sends an AVDTP_SUSPEND_RSP response to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.

8. Perform either alternative 8A or 8B depending on whether the IUT is in the Sink Role or the Source Role:

   Alternative 8A (The IUT supports the Sink Role):

   8A.1. The Lower Tester sends an AVDTP_Media_Packet to the IUT with valid V, P, X, CC, M, PT, Sequence_Nr, Time_Stamp, SSRC, CSRC, and Media_Payload.

   8A.2. The Upper Tester sends an AVDT_ReadStreamData service primitive to the IUT with Stream_Handle = SH_{IUT}, ExpLength = $L_{DataBuffer}$ (stream data frame buffer length) = Media_Payload expected length of AVDTP media frame, and InBuffer = $ADDR_{DataBuffer}$.

   8A.3. The IUT responds to the Upper Tester with AVDT_ReadStreamData with Result not 0 (unsuccessful data transfer).

   Alternative 8B (The IUT supports the Source Role):

   8B.1. The Upper Tester sends an AVDT_Write_Req service primitive to the IUT with Stream_Handle = SH_{IUT} = the value used for configuring and establishing a stream connection for $SEID_{LowerTester}$, Length = $L_{DataBuffer}$ (stream data frame buffer length) = Media_Payload of AVDTP media frame, OutBuffer = $ADDR_{DataBuffer}$, TimingInfo = Time_StampTester, and Marker = one of the possible values (1 or 0).

   8B.2. The IUT responds to the Upper Tester with AVDT_Write_Req with Result.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDT_Sig_Event_Registration service primitive to the Upper Tester with Result = 0x0000 (Event successfully registered).

  In Step 4, the IUT invokes the SuspendInd_CB callback function with the following parameters:

  - Transaction = $Transaction_{LowerTester}$
  - StreamHandle = SH_{IUT} = the value corresponding to $SEID_{IUT}$

  In Step 6, the IUT sends the AVDT_Suspend_Rsp service primitive to the Upper Tester with RSP = REQUEST_ACCEPTED.

  In Step 7, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x09 (AVDTP_SUSPEND)

  In Step 8A.3, the IUT sends the AVDT_ReadStreamData service primitive to the Upper Tester with the following parameter:

  - Result not 0 (unsuccessful data transfer)

  In alternative 8B, the IUT sends no media packet.

4.3.1.50 Ignore reserved capabilities in GetCapabilities response – INT

- Test Purpose

  Verify that the IUT (INT) is able to ignore a reserved capability in the GetCapabilities response.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEID is in the Idle state.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-30-C [Ignore reserved capabilities in GetCapabilities response – INT]
AVDTP/SNK/INT/SIG/SMG/BI-30-C [Ignore reserved capabilities in GetCapabilities response – INT]

Table 4.52: Ignore reserved capabilities in GetCapabilities response test cases

- Test Procedure

  1. The IUT performs Discover and GetCapabilities for the first SEP.
  2. The Lower Tester returns a valid set of capabilities with the addition of a capability with the value 0xEE and a length of 1 octet.
  3. The IUT sends a SetConfiguration command to the Lower Tester including all the capabilities except the 0xEE capability.

- Expected Outcome

  Pass verdict

  In Step 3, the IUT sends a SetConfiguration command that does not include the 0xEE capability.

4.3.1.51 General Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a general Reject response to the Lower Tester (INT).

- Reference

  [3] 8.17, 13.1

  [5] 8.18, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-38-C [General Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-38-C [General Reject response – ACP]

Table 4.53: General Reject response – ACP test cases

- Test Procedure

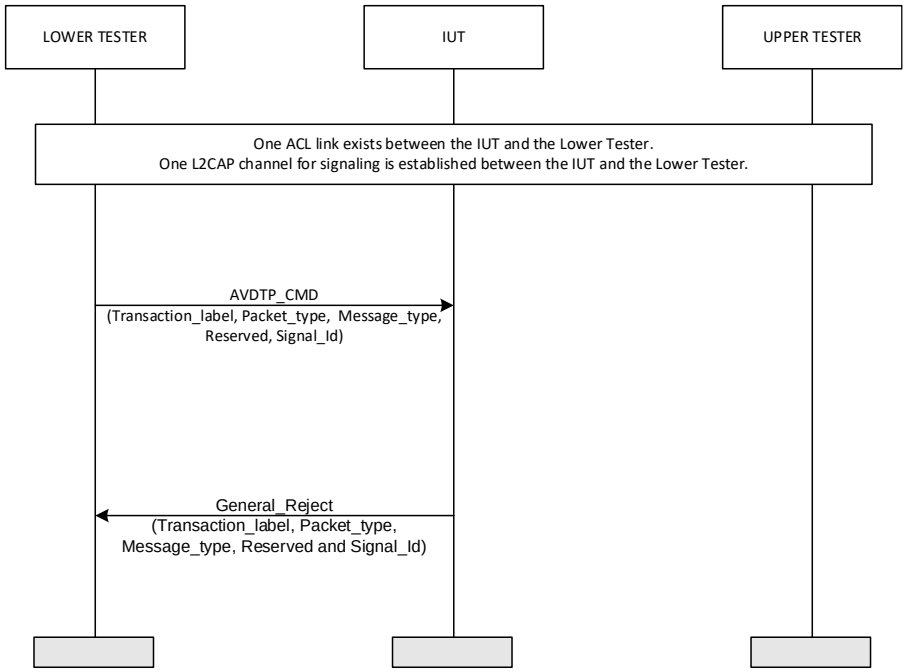

Figure 4.53: General Reject response – ACP MSC

1. The Lower Tester sends an AVCTP_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x3F (Undefined signal id).
2. The IUT responds to the Lower Tester with General_Reject with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.

- Expected Outcome

Pass verdict

In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

- Transaction_label = $Transaction_{LowerTester}$
- Packet_type = 00 (single packet)
- Message type = 01 (general reject)
- Bits 6 and 7 of Octet 1 are both set to ‘0’
- Other 6 bits = Set to the invalid signal identifier being rejected (0x3F)

4.3.1.52 Detect rejection of a Get All Capabilities command by the ACP – INT

- Test Purpose

  Verify that the IUT (INT) is able to detect the rejection by the ACP of a Get All Capabilities command.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-32-C [Detect rejection of a Get All Capabilities command by the ACP – INT]
AVDTP/SNK/INT/SIG/SMG/BI-32-C [Detect rejection of a Get All Capabilities command by the ACP – INT]

Table 4.54: Detect rejection of a Get All Capabilities command by the ACP – INT test cases

- Test Procedure

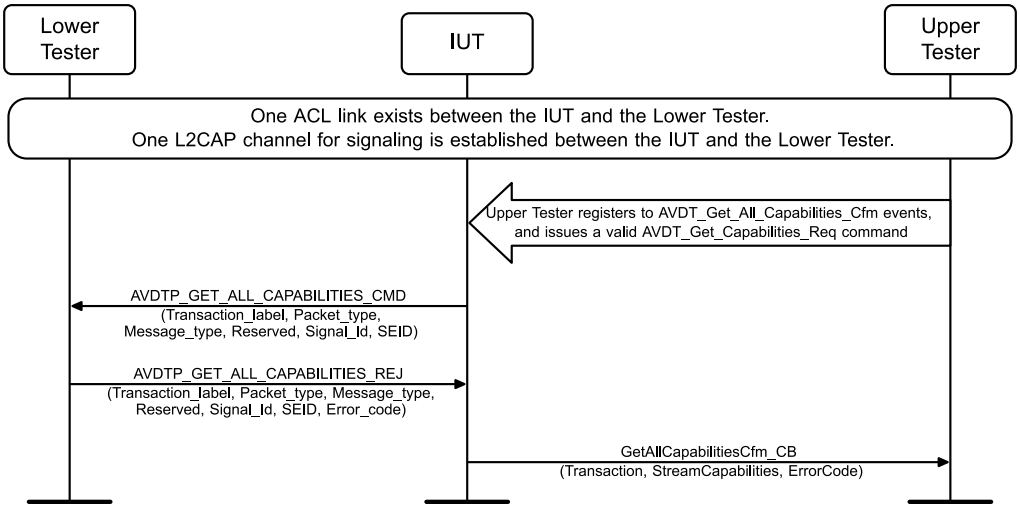

Figure 4.54: Detect rejection of a Get All Capabilities command by the ACP – INT MSC

1. The Upper Tester registers to AVDT_Get_All_Capabilities_Cfm events with the IUT and issues a valid AVDT_Get_All_Capabilities_Req command for $SEID_{LowerTester}$, which the IUT acknowledges.
2. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the Lower Tester with Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID set to valid values.
3. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_REJ rejection to the IUT with Transaction = $Transaction_{IUT}$ = value provided with the AVDTP_GET_ALL_CAPABILITIES_CMD message, Packet_type = 00 (single packet), Reserved = 00, Message_type = 11 (response reject), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Error_Code = 0x11 (BAD_LENGTH).
4. The IUT sends a GetAllCapabilitiesCfm_CB event to the Upper Tester with valid Transaction, StreamCapabilities, and ErrorCode.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT invokes the GetAllCapabilitiesCfm_CB callback function with the following parameters:

  - Transaction = $Transaction_{IUT}$
  - ErrorCode = the Error_Code provided with the AVDTP_GET_ALL_CAPABILITIES_REJ message

4.3.1.53 Invalid Get All Capabilities Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Get All Capabilities Reject response to the INT.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-33-C [Invalid Get All Capabilities Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-33-C [Invalid Get All Capabilities Reject response – ACP]

Table 4.55: Invalid Get All Capabilities Reject response – ACP test cases

- Test Procedure

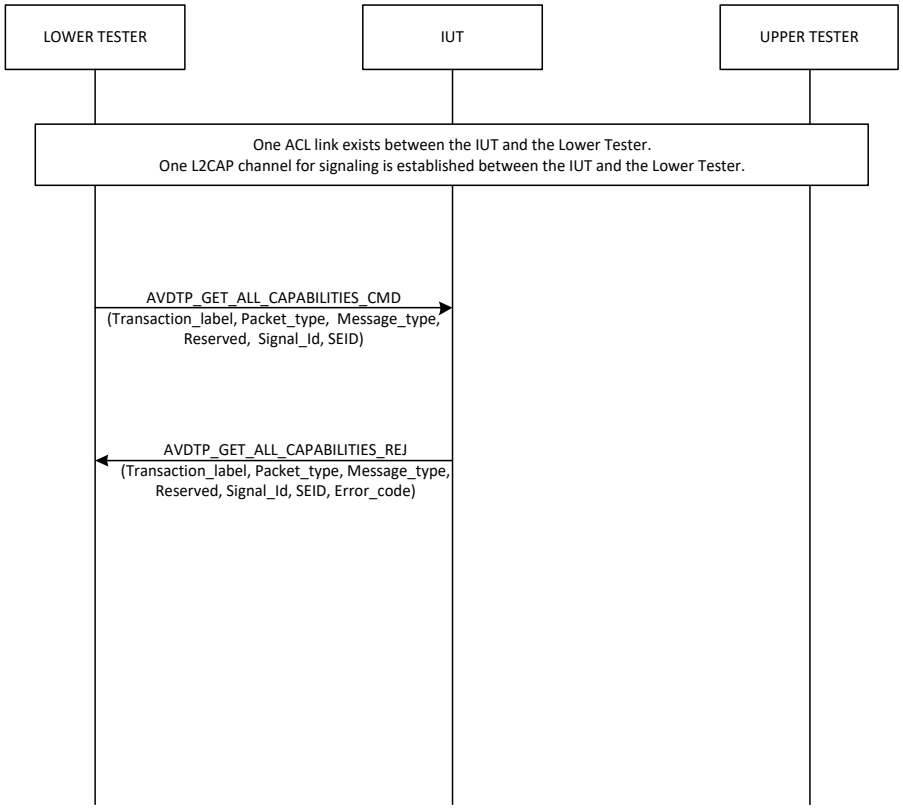

Figure 4.55: Invalid Get All Capabilities Reject response – ACP MSC

1. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), and Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES). The last byte meant to bear the ACP SEID + 2 reserved bits is omitted in order to have an invalid message length.
2. The IUT responds to the Lower Tester with AVDTP_GET_ALL_CAPABILITIES_REJ with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  Transaction_label = $Transaction_{LowerTester}$

  Message_type = 11 (response reject)

  Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)

  Error_code = 0x11 (BAD_LENGTH)

4.3.1.54 Get All Capabilities Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Reject response signal upon the Upper Tester's request in answer to a Get All Capabilities command.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.8, 8.8, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Upper Tester registers to AVDT_Get_All_Capabilities_Ind events.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SMG/BI-34-C [Get All Capabilities Reject response – ACP]
AVDTP/SNK/ACP/SIG/SMG/BI-34-C [Get All Capabilities Reject response – ACP]

Table 4.56: Get All Capabilities Reject response – ACP test cases

- Test Procedure

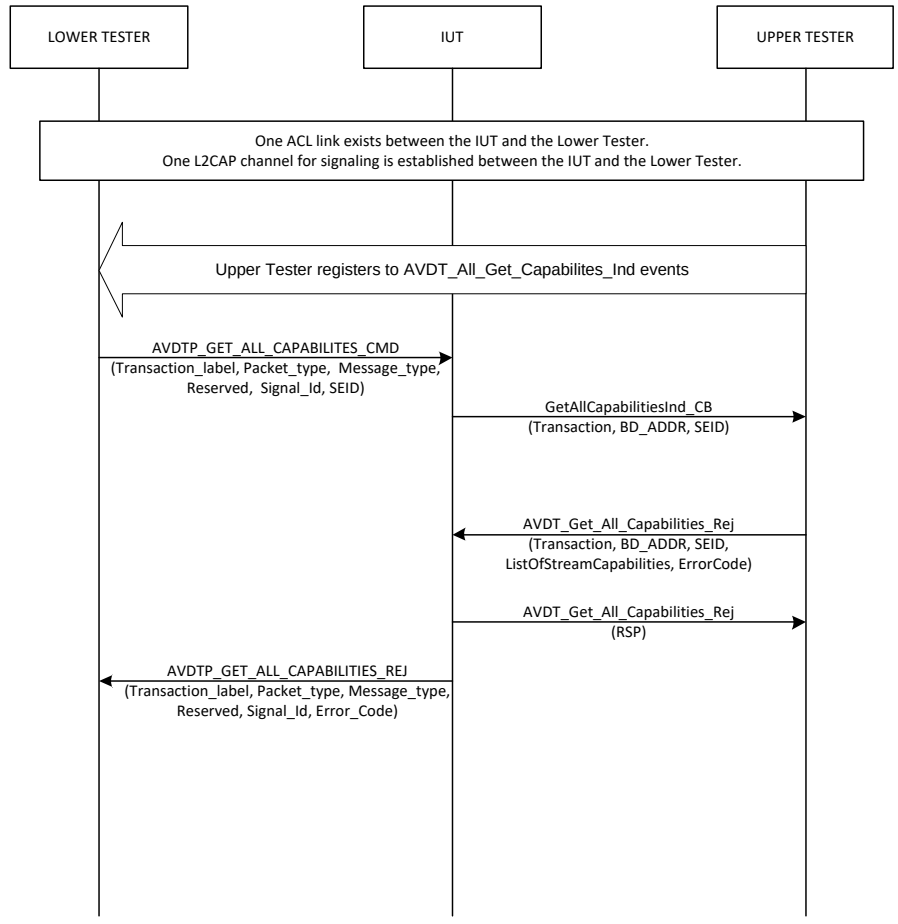

Figure 4.56: Get All Capabilities Reject response – ACP MSC

1. The Upper Tester registers to AVDT_Get_All_Capabilities_Ind events with the IUT, which the IUT acknowledges.
2. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and SEID = $SEID_{IUT}$.
3. The IUT sends a GetAllCapabilitiesInd_CB event to the Upper Tester with valid Transaction, BD_ADDR, and SEID.
4. The Upper Tester sends an AVDT_Get_All_Capabilities_Rej command to the IUT with Transaction = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, SEID = $SEID_{IUT}$, ListOfStreamCapabilities = any value, and ErrorCode = any value in the range 0xC0-0xFF.
5. The IUT responds to the Upper Tester with AVDT_Get_All_Capabilities_Rej with RSP.
6. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_REJ rejection to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and Error_Code.

- Expected Outcome

  Pass verdict

  In Step 6, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
  - Error_Code = the value provided with AVDT_Get_All_Capabilities_Rej

4.3.1.55 Accept all Defined Capabilities in a GetAllCapabilities response – INT

- Test Purpose

  Verify that the IUT (INT) is able to accept all defined capabilities in the GetAllCapabilities response.

- Reference

  [5] 8.21

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEID is in the Idle state.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BV-35-C [Accept all Defined Capabilities in a GetAllCapabilities response – INT]
AVDTP/SNK/INT/SIG/SMG/BV-35-C [Accept all Defined Capabilities in a GetAllCapabilities response – INT]

Table 4.57: Accept all Defined Capabilities in a GetAllCapabilities response – INT test cases

- Test Procedure

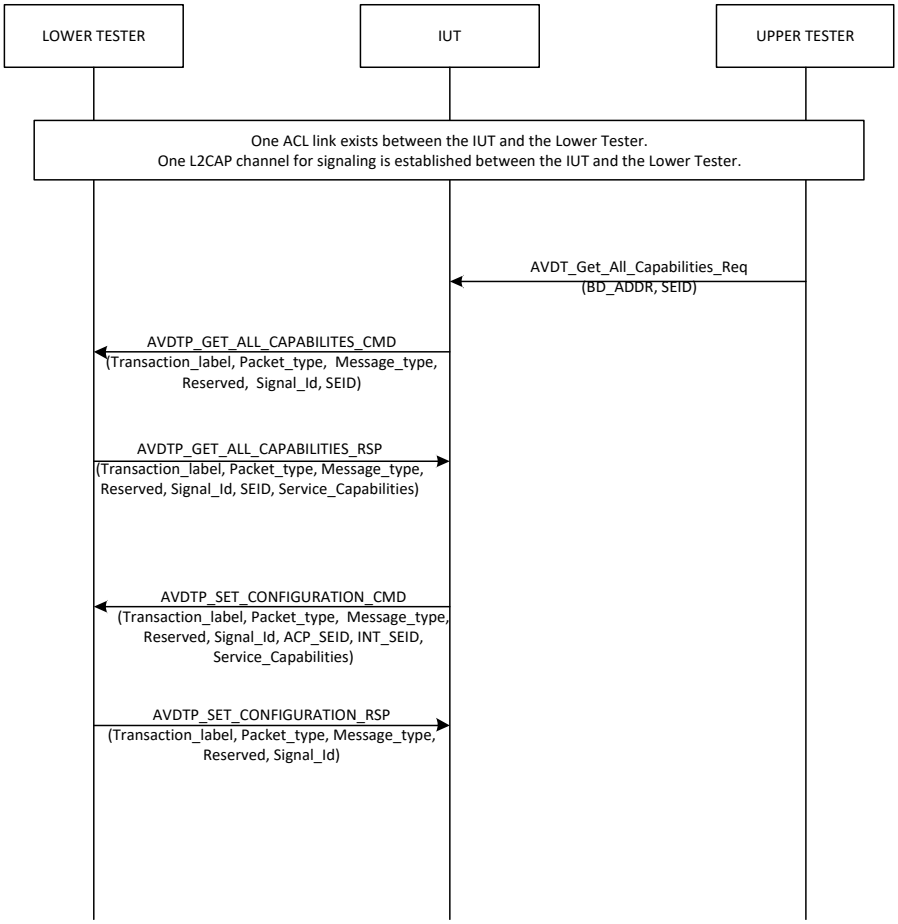

Figure 4.57: Accept all Defined Capabilities in a GetAllCapabilities response – INT MSC

1. The Upper Tester sends an AVDT_Get_All_Capabilities_Req command to the IUT with valid BD_ADDR and SEID.
2. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID.
3. The Lower Tester responds to the IUT with AVDTP_GET_ALL_CAPABILITIES_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Service_Capabilities, which are all capabilities defined in AVDTP.
4. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities, including all capabilities supported by the IUT.
5. The Lower Tester responds to the IUT with AVDTP_SET_CONFIGURATION_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with valid values.

4.3.1.56 Ignore reserved capabilities in the GetAllCapabilities response – INT

- Test Purpose

  Verify that the IUT (INT) is able to ignore a reserved capability in the GetAllCapabilities response.

- Reference

  [5] 8.19

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEID is in the Idle state.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SMG/BI-36-C [Ignore reserved capabilities in the GetAllCapabilities response – INT]
AVDTP/SNK/INT/SIG/SMG/BI-36-C [Ignore reserved capabilities in the GetAllCapabilities response – INT]

Table 4.58: Ignore reserved capabilities in the GetAllCapabilities response – INT test cases

- Test Procedure

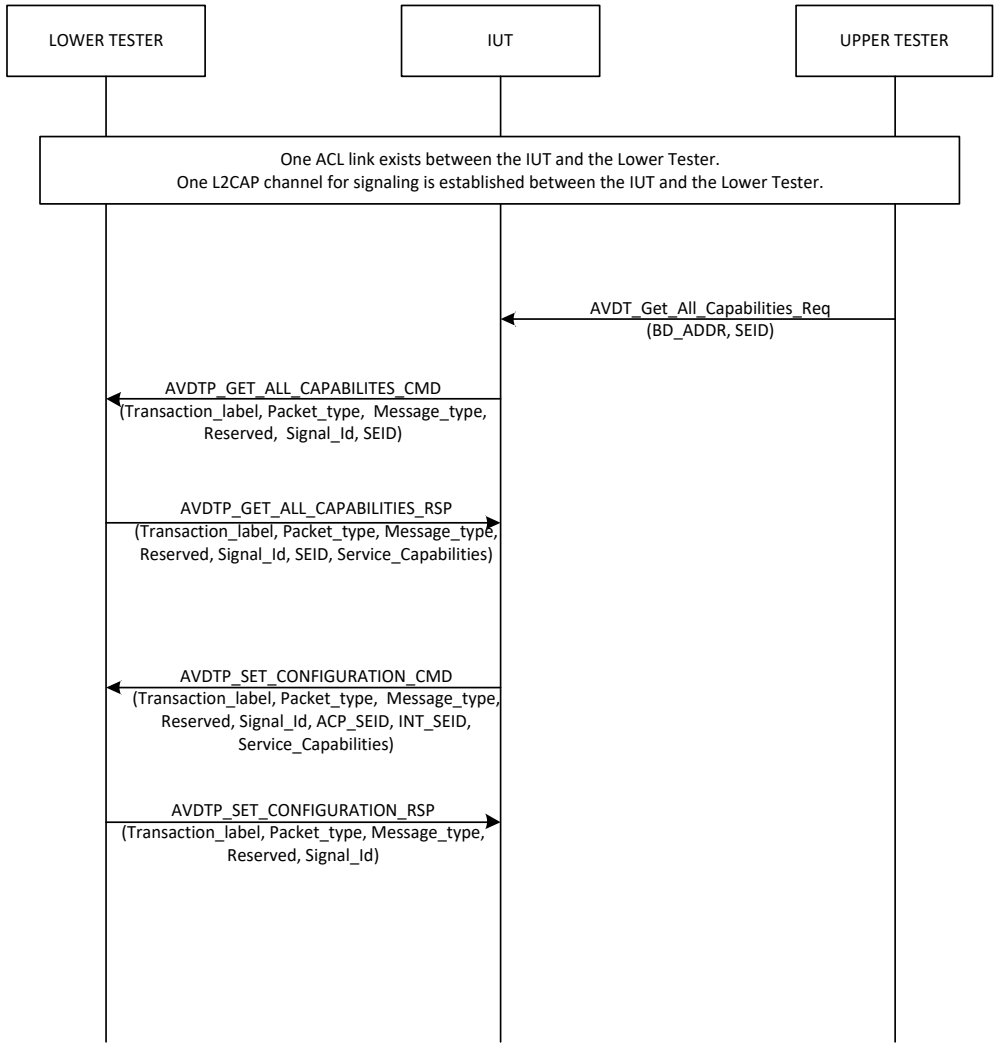

Figure 4.58: Ignore reserved capabilities in the GetAllCapabilities response – INT MSC

1. The Upper Tester sends an AVDT_Get_All_Capabilities_Req command to the IUT with valid BD_ADDR and SEID.
2. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID.
3. The Lower Tester responds to the IUT with AVDTP_GET_ALL_CAPABILITIES_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Service_Capabilities such that all capabilities are valid along with the addition of a capability with the value 0xEE and a length of 1 octet.
4. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities including all capabilities from Step 3 except the 0xEE capability.
5. The Lower Tester responds to the IUT with AVDTP_SET_CONFIGURATION_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.

- Expected Outcome

  Pass verdict

  In Step 4, the IUT sends an AVDTP_SET_CONFIGURATION_CMD command that does not include the 0xEE capability from Step 3.

4.3.2 Security Signaling Service

Verify that the following security Signaling procedures are implemented according to their specification in AVDTP:

- Content Security Control command

4.3.2.1 Content Security Control command – INT

- Test Purpose

  Verify that the IUT (INT) is able to issue a valid Content Security Control command and reports the replied confirmation.

- Reference

  [3] 6.4, 6.15, 8.16, 13.1

  [5] 6.4, 6.16, 8.17, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - A Stream with the SEP identified by $SEID_{IUT}$ has been configured and the corresponding Stream Handle SH_{IUT} has been handed to the Upper Tester.
  - The SecurityControlCfm_CB function is registered in the IUT as callback function for the AVDTP_SECURITY_CONTROL_RSP Message.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/SEC/BV-01-C [Content Security Control command – INT]
AVDTP/SNK/INT/SIG/SEC/BV-01-C [Content Security Control command – INT]

Table 4.59: Content Security Control command – INT test cases

- Test Procedure

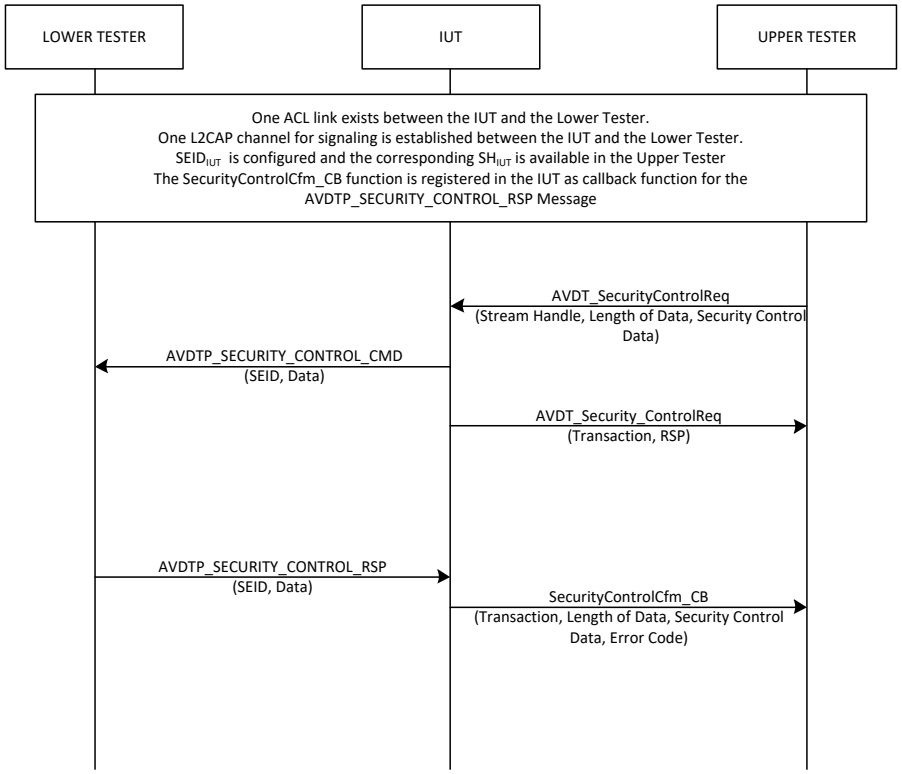

Figure 4.59: Content Security Control command – INT MSC

1. The Upper Tester sends an AVDT_SecurityControlReq to the IUT with Stream Handle = SH_{IUT}, Length of Data = $LENGTH_{Data\_Buffer}$, and Security Control Data = $ADDRESS_{Data\_Buffer}$.
2. The IUT sends an AVDTP_SECURITY_CONTROL_CMD command to the Lower Tester with valid SEID and Data.
3. The IUT sends an AVDT_SecurityControlReq to the Upper Tester with valid Transaction and RSP.
4. The Lower Tester sends an AVDTP_SECURITY_CONTROL_RSP response to the IUT with SEID = $SEID_{IUT}$ and Data = $DATA[\ ]_{LowerTester}$ (as assigned by the Lower Tester).
5. The IUT sends a SecurityControlCfm_CB event to the Upper Tester with valid Transaction, Length of Data, Security Control Data, and Error Code.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT issues an AVDTP_SECURITY_CONTROL_CMD message to the Lower Tester containing the data provided by the Upper Tester and the SEID corresponding to the Stream Handle SH_{IUT}.

  In Step 3, the IUT returns a valid Transaction label as a return value of the AVDT_SecurityControlReq service primitive and the RSP Output parameter has the value RSP = 0x0000 (Request accepted by the local entity – Service in progress).

  In Step 5, the IUT invokes the SecurityControlCfm_CB callback function in the Upper Tester with the data provided by the Lower Tester and the same Transaction Label returned to the Upper Tester from the AVDT_SecurityControlReq.

4.3.2.2 Content Security Control command – ACP

- Test Purpose

  Verify that the IUT (ACP) reports the reception of a valid Content Security Control command and replies the returned confirmation.

- Reference

  [3] 6.4, 6.15, 8.16, 13.1

  [5] 6.4, 6.16, 8.17, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The SEP identified by $SEID_{IUT}$ is configured and the corresponding SH_{IUT} is available in the Upper Tester.
  - The SecurityControlInd_CB function is registered in the IUT as callback function for the AVDTP_SECURITY_CONTROL_CMD Message.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SEC/BV-02-C [Content Security Control command – ACP]
AVDTP/SNK/ACP/SIG/SEC/BV-02-C [Content Security Control command – ACP]

Table 4.60: Content Security Control command – ACP test cases

- Test Procedure

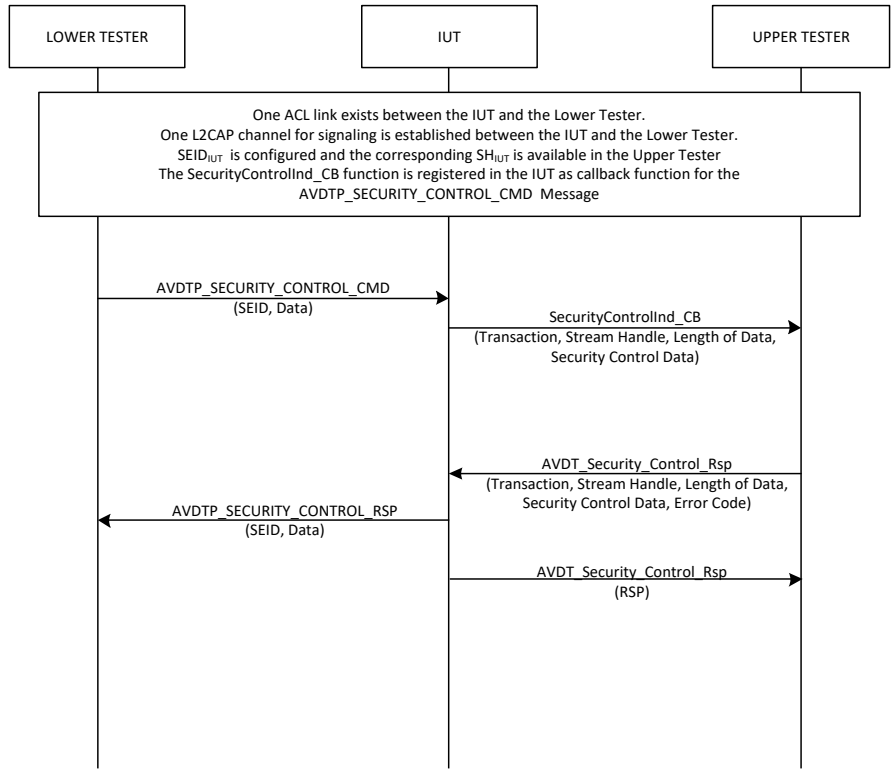

Figure 4.60: Content Security Control command – ACP MSC

1. The Lower Tester sends an AVDTP_SECURITY_CONTROL_CMD command to the IUT with SEID = $SEID_{IUT}$ and Data = DATA[ ]$_{LowerTester}$ (as assigned by the Lower Tester).
2. The IUT sends a SecurityControlInd_CB event to the Upper Tester with valid Transaction, Stream Handle, Length of Data, and Security Control Data.
3. The Upper Tester sends an AVDT_Security_Control_Rsp service primitive to the IUT with Transaction = Transaction label provided to the Upper Tester by the SecurityControlInd_CB callback function, Stream Handle = SH_{IUT}, Length of Data = $LENGTH_{Data\_Buffer}$, Security Control Data = $ADDRESS_{Data\_Buffer}$, and Error Code = 'No Error'.
4. The IUT sends an AVDTP_SECURITY_CONTROL_RSP response to the Lower Tester with valid SEID and Data.
5. The IUT sends an AVDT_Security_Control_Rsp response to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the SecurityControlInd_CB callback function with the transaction label and data provided by the Lower Tester. The Stream Handle is the one corresponding to $SEID_{IUT}$.

  In Step 4, the IUT sends an AVDTP_SECURITY_CONTROL_RSP to the Lower Tester, containing the transaction label and data provided by the Upper Tester. The SEID is the $SEID_{IUT}$ that corresponds to the Stream Handle provided by the Upper Tester.

4.3.2.3 Set Configuration Reject response – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the CP type is undefined or if the CP format is incorrect.

- Reference

  [3] 6.4, 6.15, 8.16, 13.1

  [5] 6.4, 6.16, 8.17, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not yet configured $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/SEC/BI-01-C [Set Configuration Reject response – ACP]
AVDTP/SNK/ACP/SIG/SEC/BI-01-C [Set Configuration Reject response – ACP]

Table 4.61: Set Configuration Reject response – ACP test cases

- Test Procedure

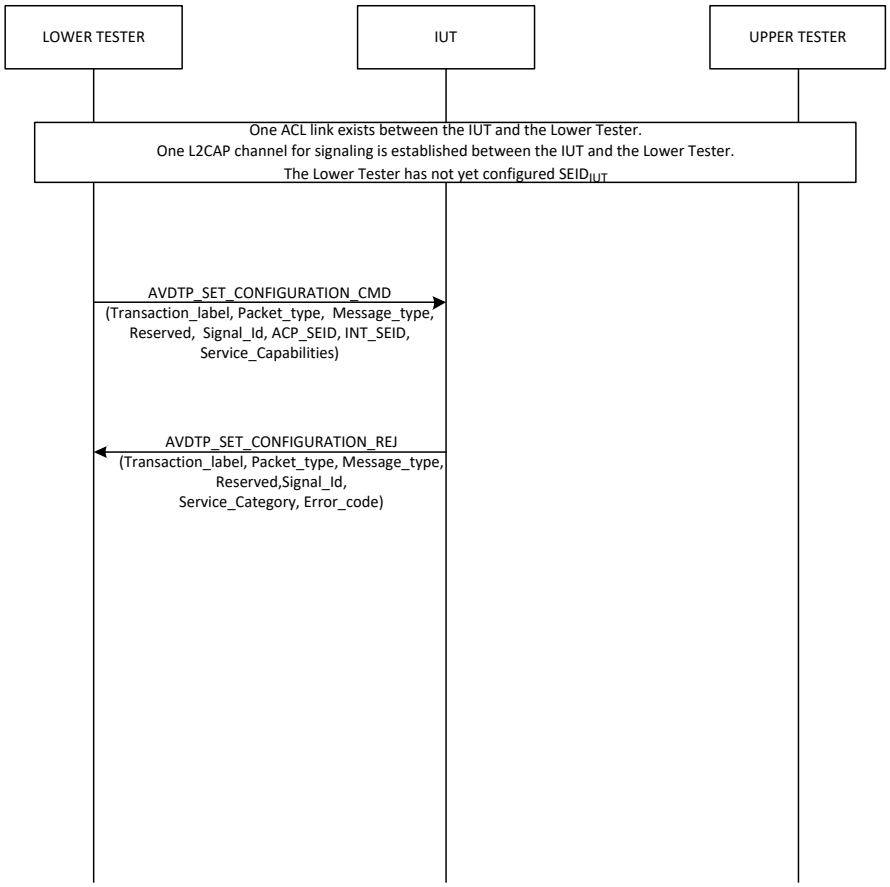

Figure 4.61: Set Configuration Reject response – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an incorrect format of the content protection service capability (e.g., LOSC = 0x01).
2. The IUT responds to the Lower Tester with AVDTP_SET_CONFIGURATION_REJ with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x27 (BAD_CP_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x27 (BAD_CP_FORMAT)

4.3.3 Signaling Message Fragmentation Service

Verify that the fragmentation of signaling messages is implemented according to its specification in AVDTP.

4.3.3.1 Fragmentation of signaling messages

- Test Purpose

  Verify that the IUT (ACP) fragments the signaling messages that cannot fit in a single L2CAP packet.

- Reference

  [3] 8.3

  [5] 8.3

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - On the Lower Tester side, the value of the MTU for the L2CAP channel ($MTU_{LowerTester}$) is configured to 48 bytes (lowest one).
  - The GetCapabilitiesInd_CB callback is registered in the IUT as a callback for the AVDTP_GET_CAPABILITIES_CMD message.
  - The IUT exposes one SEID = $SEID_{IUT}$ and the Lower Tester knows this ID (e.g., from a precedent Stream Discover Command).

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/SIG/FRA/BV-01-C [Fragmentation of signaling messages – ACP]
AVDTP/SNK/ACP/SIG/FRA/BV-01-C [Fragmentation of signaling messages – ACP]

Table 4.62: Fragmentation of signaling messages test cases

- Test Procedure

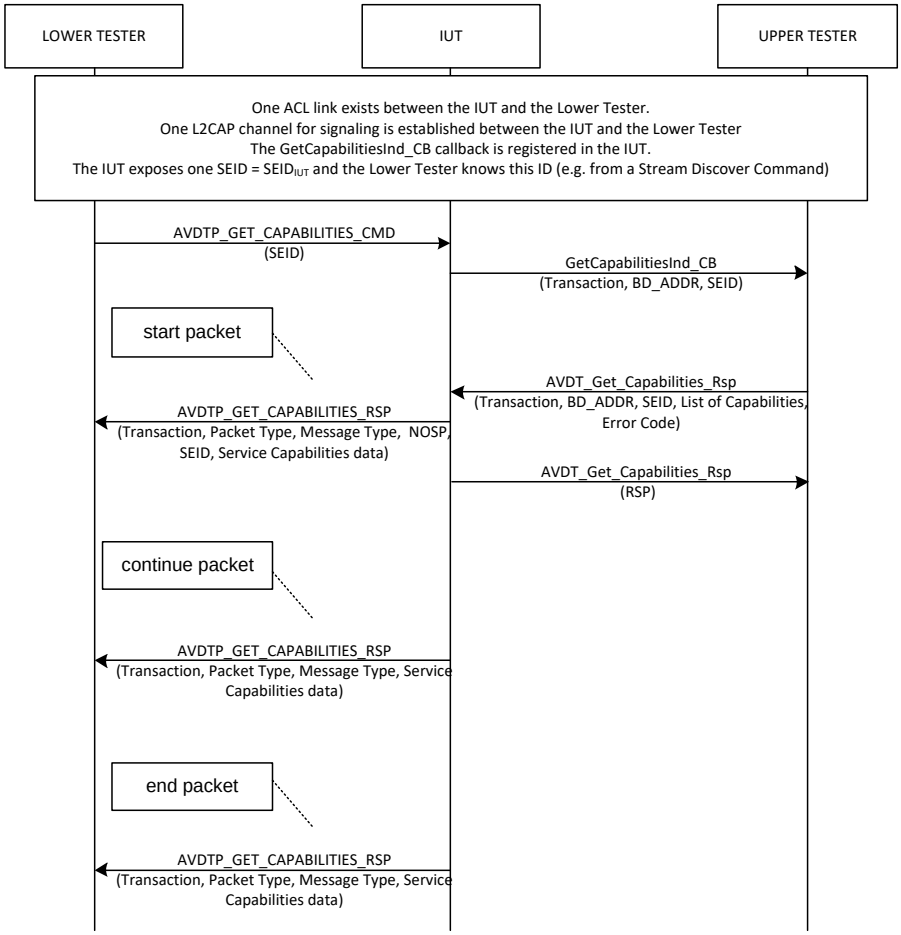

Figure 4.62: Fragmentation of signaling messages MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with SEID = $SEID_{IUT}$ (e.g., received through a precedent Stream Discover Command).
2. The IUT sends the GetCapabilitiesInd_CB event to the Upper Tester with valid Transaction, BD_ADDR, and SEID.
3. The Upper Tester sends an AVDT_Get_Capabilities_Rsp response to the IUT with Transaction = Transaction label received by the precedent GetCapabilitiesInd_CB callback function, BD_ADDR = $BD\_ADDR_{LowerTester}$, SEID = $SEID_{IUT}$, List of Capabilities = the Capabilities exposed by this SEP are chosen in a way that the size of the resulting response message exceeds the Lower Tester supported MTU in such a way that the message is fragmented in 3 consecutive L2CAP packets. This can be achieved at the IUT side by aggregating an appropriate list of such capabilities (for instance by enlarging the list of exposed content protection methods), and Error Code = 'No Error'.
4. The IUT sends an AVDTP_GET_CAPABILITIES_RSP start packet to the Lower Tester with valid Transaction, Packet Type, Message Type, NOSP, SEID, and Service Capabilities data.
5. The IUT sends an AVDT_Get_Capabilities_Rsp response to the Upper Tester with RSP.
6. The IUT sends an AVDTP_GET_CAPABILITIES_RSP continue packet to the Lower Tester with valid Transaction, Packet Type, Message Type, and Service Capabilities data.
7. The IUT sends an AVDTP_GET_CAPABILITIES_RSP end packet to the Lower Tester with valid Transaction, Packet Type, Message Type, and Service Capabilities data.

- Expected Outcome

  Pass verdict

  After reception of the AVDT_Get_Capabilities_Rsp service primitive from the Upper Tester, the IUT issues three consecutive packets to the Lower Tester:

  - One start packet, one continue packet and one end packet.
  - The transaction labels in the start, continue and end packets are equal.

  In Step 4, the IUT sends the AVDTP_GET_CAPABILITIES_RSP start packet with the specified values:

  - NOSP = 3
  - Packet type = 01 (start packet)
  - Message Type = 10 (response accept)

  In Step 6, the IUT sends the AVDTP_GET_CAPABILITIES_RSP continue packet with the specified values:

  - Packet type = 10 (continue packet)
  - Message Type = 10 (response accept)

  In Step 7, the IUT sends the AVDTP_GET_CAPABILITIES_RSP end packet with the specified values:

  - Packet type = 11 (end packet)
  - Message Type = 10 (response accept)

  After reassembly in the Lower Tester, the data in the Service Capabilities field matches the data that was sent by the Upper Tester.

4.3.3.2 Reassembling of signaling messages – INT

- Test Purpose

  Verify that the IUT (INT) reassembles the signaling messages that cannot fit in a single L2CAP packet.

- Reference

  [3] 8.3

  [5] 8.3

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The GetCapabilitiesCfm_CB callback is registered in the IUT as callback for the AVDTP_GET_CAPABILITIES_RSP message.
  - The Upper Tester knows the SEID exposed by the Lower Tester.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/SIG/FRA/BV-02-C [Reassembling of signaling messages – INT]
AVDTP/SNK/INT/SIG/FRA/BV-02-C [Reassembling of signaling messages – INT]

Table 4.63: Reassembling of signaling messages test cases

- Test Procedure

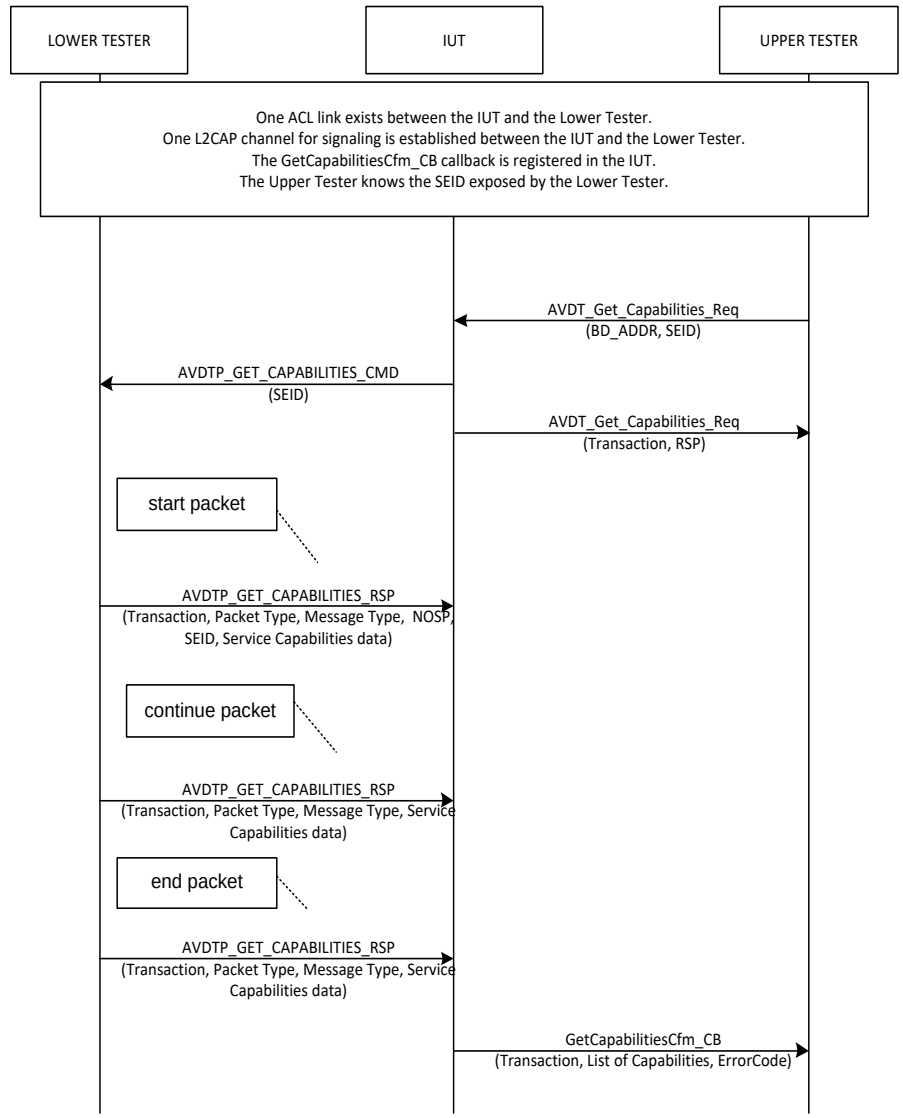

Figure 4.63: Reassembling of signaling messages – INT MSC

1. The Upper Tester sends an AVDT_GetCapabilitiesReq command to the IUT with BD_ADDR = $BD\_ADDR_{LowerTester}$ and SEID = $SEID_{LowerTester}$.
2. The IUT sends an AVDTP_GET_CAPABILITIES_CMD command to the Lower Tester with valid SEID.
3. The IUT sends an AVD_Get_Capabilities_Req response to the Upper Tester with valid Transaction and RSP.
4. The Lower Tester sends an AVDTP_GET_CAPABILITIES_RSP start packet to the IUT with Transaction = $Transaction_{AVDTP_GET_CAPABILITIES}$_CMD (as received by the Lower Tester), Packet

Type = 01 (start packet), Message Type = 10 (response accept), NOSP = 3 (Number of Signal packets), SEID = $SEID_{LowerTester}$, and Service Capabilities = the Capabilities exposed by this SEP are chosen in a way that the size requires three L2CAP packets, 3x48 bytes minus header sizes.

5. The Lower Tester sends an AVDTP_GET_CAPABILITIES_RSP continue packet to the IUT with Transaction = $Transaction_{AVDTP\_GET\_CAPABILITIES\_CMD}$ (see Step 4), Packet Type = 10 (continue packet), Service Capabilities = see Step 4.
6. The Lower Tester sends an AVDTP_GET_CAPABILITIES_RSP end packet to the IUT with Transaction = $Transaction_{AVDTP\_GET\_CAPABILITIES\_CMD}$ (see Step 4), Packet Type = 11 (end packet), Service Capabilities = see Step 4.
7. The IUT sends the GetCapabilitiesCfm_CB event to the Upper Tester with valid Transaction, List of Capabilities, and ErrorCode.

- Expected Outcome

  Pass verdict

  After reception of the three packets belonging to the AVDTP_GET_CAPABILITIES_RSP from the Lower Tester, the IUT invokes the GetCapabilitiesCfm_CB callback function in the Upper Tester.

  The List of Capabilities delivered by the GetCapabilitiesCfm_CB callback function matches the data provided by the Lower Tester in the three AVDTP_GET_CAPABILITIES_RSP packets.

  The Error Code delivered by the GetCapabilitiesCfm_CB callback function states, “No Error”.

4.3.4 Delay Reporting

Verify that the Stream Management Signaling procedure of delay reporting is implemented according to its specification in AVDTP.

AVDTP/SNK/ACP/SIG/SYN/BV-01-C [Synchronization capability offered]

- Test Purpose

  Verify that the IUT (SNK device) offers the delay reporting capability. This is required to enable a SRC to configure delay reporting.

- Reference

  [5] 8.21.9

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.

- Test Procedure

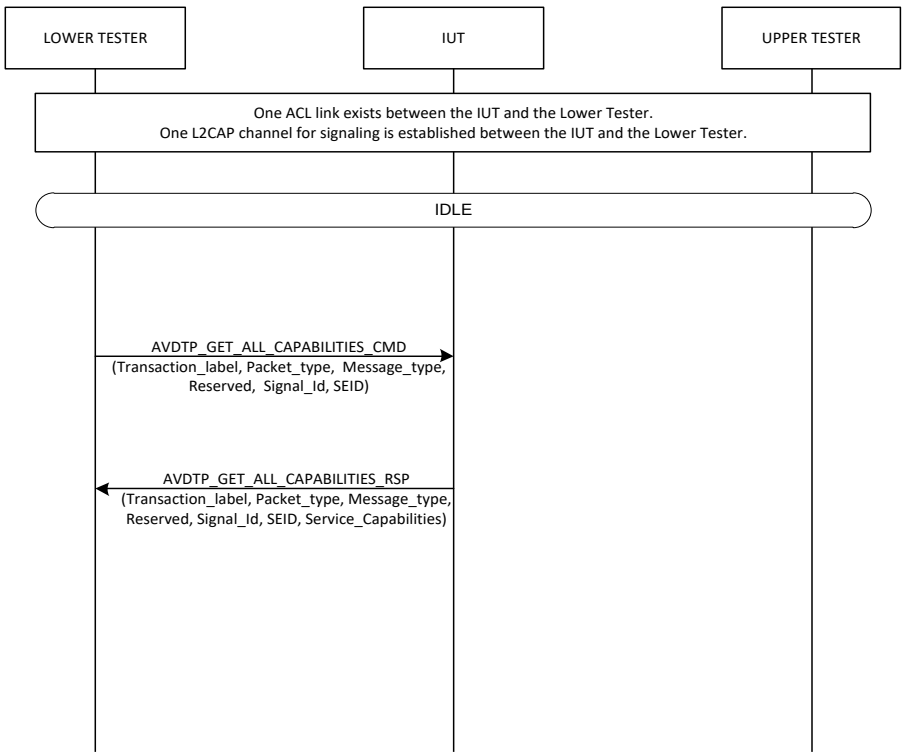

Figure 4.64: AVDTP/SNK/ACP/SIG/SYN/BV-01-C [Synchronization capability offered] MSC

1. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, and SEID.
2. The IUT responds to the Lower Tester with AVDTP_GET_ALL_CAPABILITIES_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, SEID, and Service_Capabilities.

- Expected Outcome

  Pass verdict

  The delay reporting capability is offered by the IUT and returned in the Get All Capabilities Response.

AVDTP/SNK/INT/SIG/SYN/BV-02-C [Sync capability configuration]

- Test Purpose

  Verify that the delay reporting capability is configured by a SNK device when it is offered by the SRC device. This allows a SRC device to detect and use the delay reporting capability of a SNK device when the SNK device is the INT of the start Streaming procedure.

- Reference

  [5] 8.21.9

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - Delay reporting capability offered by SRC (Lower Tester).

- Test Procedure

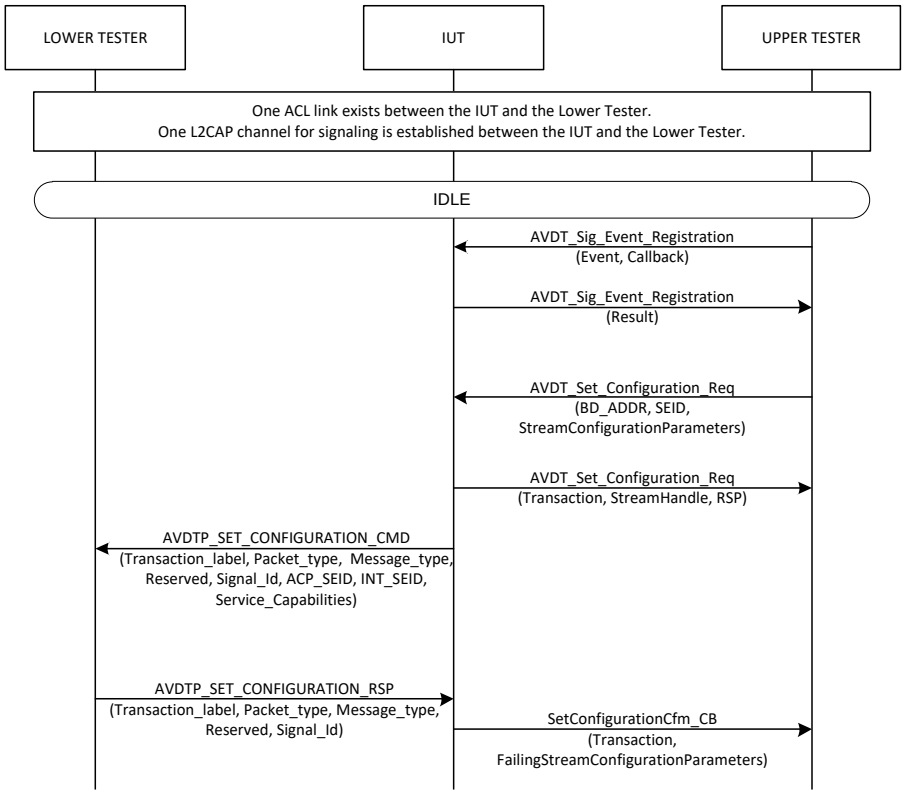

Figure 4.65: AVDTP/SNK/INT/SIG/SYN/BV-02-C [Sync capability configuration] MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration command to the IUT with valid Event and Callback.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Set_Configuration_Req command to the IUT with valid BD_ADDR, SEID, and StreamConfigurationParameters.
4. The IUT responds to the Upper Tester with AVDT_Set_Configuration_Req with valid Transaction, StreamHandle, and RSP.
5. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities.
6. The Lower Tester responds to the IUT with AVDTP_SET_CONFIGURATION_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.
7. The IUT sends the SetConfigurationCfm_CB event to the Upper Tester with valid Transaction and FailingStreamConfigurationParameters.

- Expected Outcome

  Pass verdict

  Delay reporting is configured by the IUT.

AVDTP/SNK/ACP/SIG/SYN/BV-03-C [Initial delay report (SNK is ACP)]

- Test Purpose

  Verify that an initial delay report is sent by a SNK (ACP) device when the Stream Configuration procedure is initiated by the SRC device (transition from Idle to Open state) and that the report conforms to the specification.

- Reference

  [5] 8.19.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEP discovered.
  - Capabilities of SEP discovered.

- Test Procedure

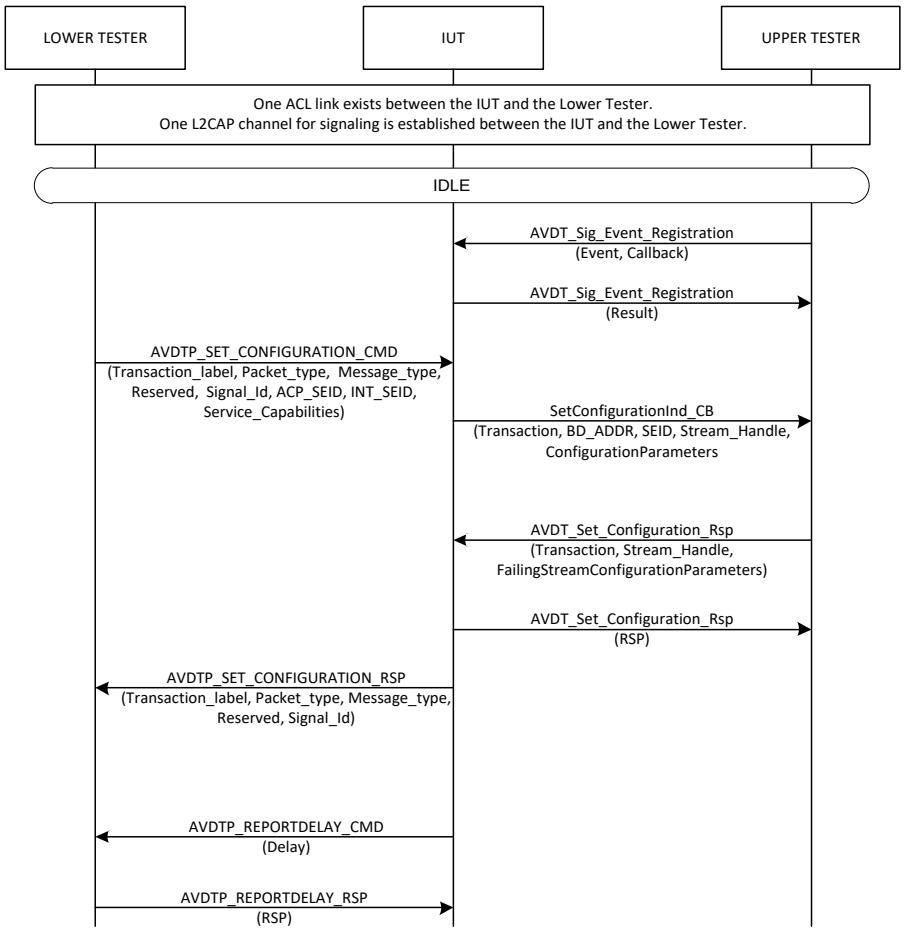

Figure 4.66: AVDTP/SNK/ACP/SIG/SYN/BV-03-C [Initial delay report (SNK is ACP)] MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration command to the IUT with valid Event and Callback.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with valid Result.

3. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD to the IUT with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities.
4. The IUT sends the SetConfigurationInd_CB event to the Upper Tester with valid Transaction, BD_ADDR, SEID, Stream_Handle, and ConfigurationParameters.
5. The Upper Tester sends an AVDT_Set_Configuration_Rsp response to the IUT with valid Transaction, Stream_Handle, and FailingStreamConfigurationParameters.
6. The IUT responds to the Upper Tester with AVDT_Set_Configuration_Rsp with RSP.
7. The IUT sends an AVDTP_SET_CONFIGURATION_RSP command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.
8. The IUT sends an AVDTP_REPORTDELAY_CMD command to the Lower Tester with valid Delay.
9. The Lower Tester responds to the IUT with AVDTP_REPORTDELAY_RSP with RSP.

- Expected Outcome

  Pass verdict

  In Step 8, the IUT sends the valid delay report: AVDTP_REPORTDELAY_CMD(Delay) directly after the AVDTP_SET_CONFIGURATION_RSP.

AVDTP/SNK/INT/SIG/SYN/BV-04-C [Initial delay report (SNK is INT)]

- Test Purpose

  Verify that an initial delay report is sent by a SNK (INT) device when the Stream Configuration procedure is initiated by the SNK device (transition from Idle to Open state) and that the report conforms to the specification.

- Reference

  [5] 8.19.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SEP discovered.
  - Capabilities of SEP discovered.

- Test Procedure

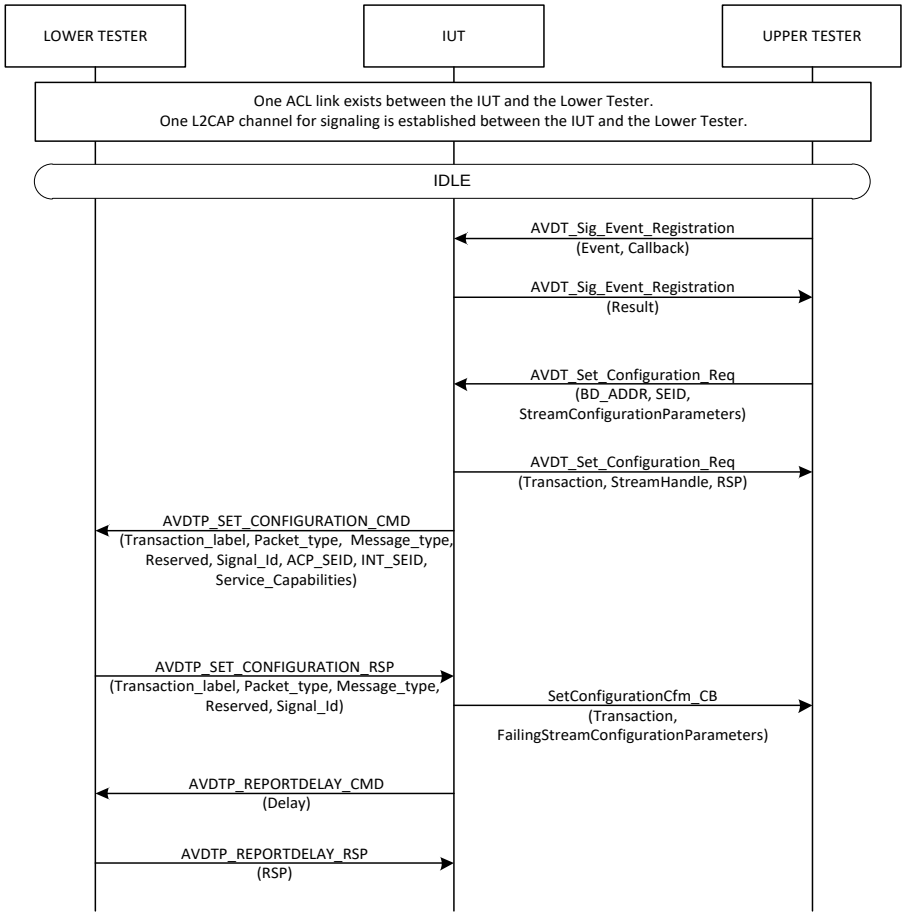

Figure 4.67: AVDTP/SNK/INT/SIG/SYN/BV-04-C [Initial delay report (SNK is INT)] MSC

1. The Upper Tester sends an AVDT_Sig_Event_Registration command to the IUT with valid Event and Callback.
2. The IUT responds to the Upper Tester with AVDT_Sig_Event_Registration with Result.
3. The Upper Tester sends an AVDT_Set_Configuration_Req command to the IUT with valid BD_ADDR, SEID, and StreamConfigurationParameters.
4. The IUT responds to the Upper Tester with AVDT_Set_Configuration_Req with valid Transaction, StreamHandle, and RSP.
5. The IUT sends an AVDTP_SET_CONFIGURATION_CMD command to the Lower Tester with valid Transaction_label, Packet_type, Message_type, Reserved, Signal_Id, ACP_SEID, INT_SEID, and Service_Capabilities.
6. The Lower Tester responds to the IUT with AVDTP_SET_CONFIGURATION_RSP with valid Transaction_label, Packet_type, Message_type, Reserved, and Signal_Id.
7. The IUT sends the SetConfigurationCfm_CB event to the Upper Tester with valid Transaction and FailingStreamConfigurationParameters.
8. The IUT sends an AVDTP_REPORTDELAY_CMD command to the Lower Tester with a valid Delay.
9. The Lower Tester responds to the IUT with AVDTP_REPORTDELAY_RSP with RSP.

- Expected Outcome

  Pass verdict

  In Step 8, the IUT sends the valid delay report: AVDTP_REPORTDELAY_CMD(Delay) directly after the AVDTP_SET_CONFIGURATION_RSP.

AVDTP/SRC/INT/SIG/SYN/BV-05-C [Delay report from Open state]

- Test Purpose

  Verify that a delay report received by a SRC IUT while in the Open state is properly accepted as valid.

- Reference

  [5] 8.19.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - SRC IUT in process of Initiating (INT) media connection.
  - Configuration of SEP includes the Delay Reporting Capability.
  - L2CAP channel for signaling and media is open (Open state).

- Test Procedure

  SNK sends AVDTP_REPORTDELAY_CMD(delay).

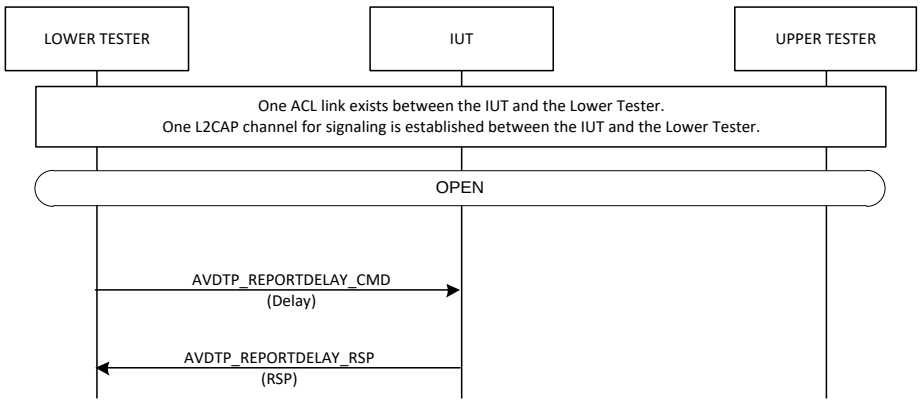

Figure 4.68: AVDTP/SRC/INT/SIG/SYN/BV-05-C [Delay report from Open state] MSC

1. The Lower Tester sends an AVDTP_REPORTDELAY_CMD command to the IUT with valid Delay.
2. The IUT responds to the Lower Tester with AVDTP_REPORTDELAY_RSP with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT responds to the delay report with AVDTP_REPORTDELAY_RSP(SUCCESS).

- Notes

  Some SRC devices will immediately send the AVDTP_START_CMD command upon successful transition to the Open state. If this occurs, the Lower Tester should send the AVDTP_REPORTDELAY_CMD before sending the sending the AVDTP_START_RSP.

AVDTP/SRC/ACP/SIG/SYN/BV-06-C [Delay report from Streaming state]

- Test Purpose

  Verify that a delay report received by a SRC IUT while in the Streaming state is properly accepted as valid.

- Reference

  [5] 8.19.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - Configuration of SEP includes the Delay Reporting Capability.
  - L2CAP channel for media is open and started (Streaming state).

- Test Procedure

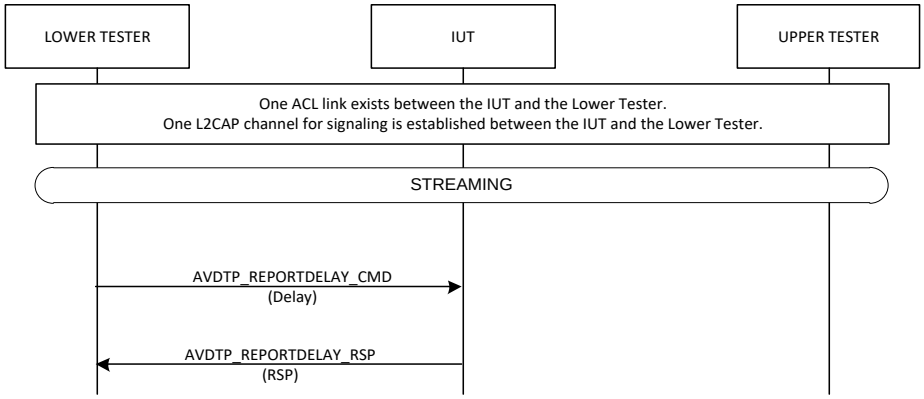

Figure 4.69: AVDTP/SRC/ACP/SIG/SYN/BV-06-C [Delay report from Streaming state] MSC

1. The Lower Tester sends an AVDTP_REPORTDELAY_CMD command to the IUT with valid Delay.
2. The IUT responds to the Lower Tester with AVDTP_REPORTDELAY_RSP with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT responds to the delay report with the AVDTP_REPORTDELAY_RSP (SUCCESS).

4.4 Transport

Verify the correct implementation of the Transport procedures.

4.4.1 Basic Transport Service

Verify that the basic Transport procedures are implemented according to their specifications in AVDTP.

AVDTP/SRC/INT/TRA/BTR/BV-01-C [Send streaming media data]

- Test Purpose

  Verify that the IUT (SRC) is able to send streaming media data on an established stream.

- Reference

  [3] 7.2, 13.2

  [5] 7.2, 13.2

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The IUT has discovered $SEID_{Tester}$. The service capabilities of $SEID_{Tester}$ has been queried and configured, see Section 4.3.1.1, Stream Discover command – INT. The SEP supports and has been configured as an SBC source codec. The stream connection has been established to $SEID_{Tester}$, see Section 4.3.1.7, Get Configuration command – INT.
  - One transport session has been opened.
  - The IUT is in the Open state.
  - The AVDT_Start_Cfm event callback function StartCfm_CB has been successfully registered.

- Test Procedure

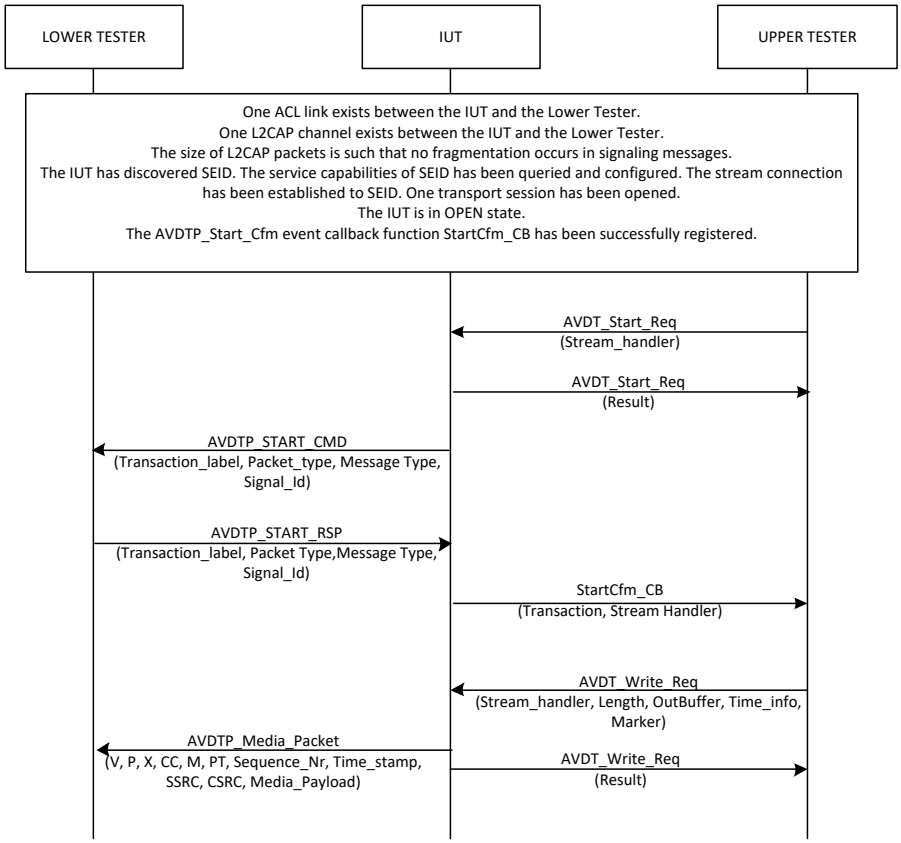

Figure 4.70: AVDTP/SRC/INT/TRA/BTR/BV-01-C [Send streaming media data] MSC

1. The Upper Tester sends an AVDT_Start_Req command to the IUT with Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration.
2. The IUT responds to the Upper Tester with AVDT_Start_Req with Transaction_label = $Transaction_{IUT}$ and RSP = 0 (request accepted).
3. The IUT sends an AVDTP_START_CMD command to the Lower Tester with Transaction_label = $Transaction_{IUT}$, Message_type = 00 (command), Signal_Id = 0x07 (AVDTP_START), NSEP = 1, and SEID = $SEID_{Tester}$.
4. The Lower Tester responds to the IUT with AVDTP_START_RSP with Transaction_label = $Transaction_{IUT}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 10 (response accept), and Signal_Id = 0x07 (AVDTP_START).
5. The IUT sends a StartCfm_CB event to the Upper Tester with Transaction_label = $Transaction_{IUT}$ and Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration.
6. The Upper Tester sends an AVDT_Write_Req command to the IUT with Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration, Length = L_{data_buffer} (stream data frame buffer length) = Media_Payload of AVDTP media frame, OutBuffer = $ADDR_{DataBuffer}$, TimingInfo = $Time\_Stamp_{Tester}$, Marker = 0 = one of the possible values (1 or 0), and Payload type = PT_{IUT} = value chosen according to the implementation in the IUT.
7. The IUT sends an AVDTP_Media_Packet to the Lower Tester with valid values.
8. The IUT sends an AVDT_Write_Req command to the Upper Tester with Result.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDT_Start_Req command to the Upper Tester with the following parameters:

  - Transaction_label = $Transaction_{IUT}$
  - RSP = 0 (request accepted)

  In Step 3, the IUT sends the AVDTP_START_CMD command to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{IUT}$
  - Message_type = 00 (command)
  - Signal_Id = 0x07 (AVDTP_START)
  - NSEP = 1
  - SEID = $SEID_{Tester}$

  In Step 5, the IUT invokes the StartCfm_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{IUT}$
  - Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration

  In Step 7, the IUT sends a media packet, see Section 7.2.1 in [3], [5] for Media Packet Format definitions.

  In Step 8, the IUT sends the AVDT_Write_Req command to the Upper Tester with Result.

AVDTP/SNK/ACP/TRA/BTR/BV-02-C [SNK receives streaming media data]

- Test Purpose

  Verify that the IUT (SNK) is able to receive streaming media data on an established stream.

- Reference

  [3] 7.2, 13.2

  [5] 7.2, 13.2

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered $SEID_{Tester}$. The service capabilities of $SEID_{Tester}$ has been queried and configured. The SEP supports and has been configured as an SBC source codec. The stream connection has been established to $SEID_{Tester}$. One transport session has been opened.
  - The IUT is in the Open state.
  - The AVDTP_Start_Ind event callback function StartInd_CB has been successfully registered.

- Test Procedure

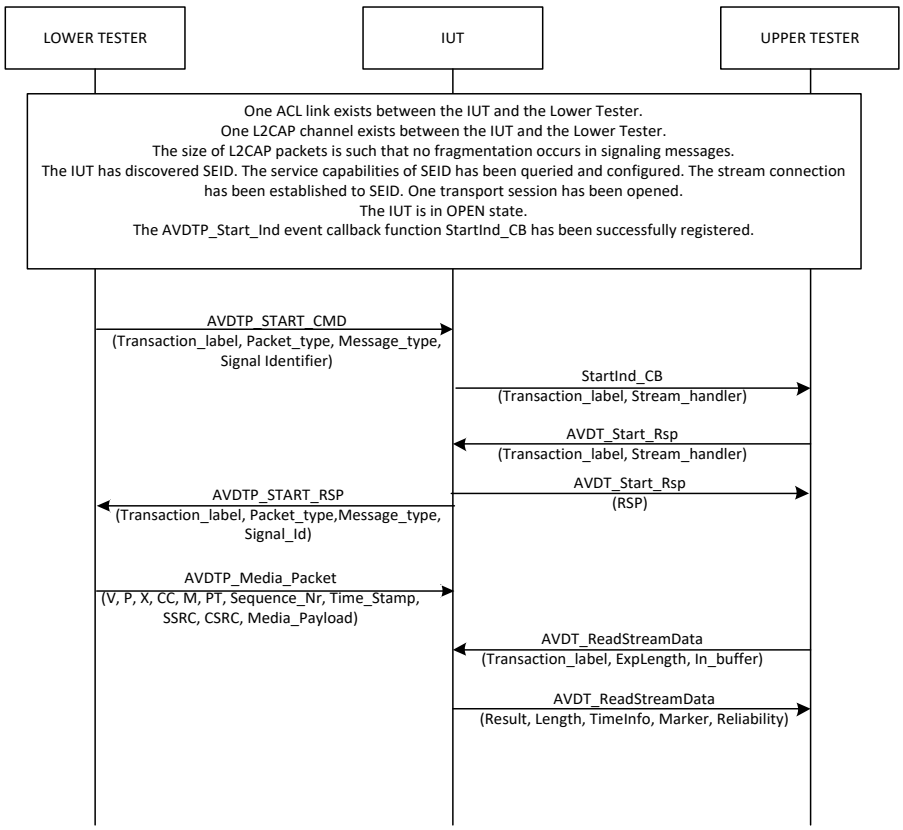

Figure 4.71: AVDTP/SNK/ACP/TRA/BTR/BV-02-C [SNK receives streaming media data] MSC

1. The Lower Tester sends an AVDTP_START_CMD command to the IUT with Transaction_label = $Transaction_{TestSystem}$, Packet_Type = 00 (single packet), Reserved = 00, Message_Type = 00 (command), Signal_Id = 0x07 (AVDTP_START), NSEP = 1, and SEID = $SEID_{Tester}$.
2. The IUT sends the StartInd_CB event to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration.
3. The Upper Tester sends an AVDT_Start_Rsp response to the IUT with Transaction_label = $Transaction_{LowerTester}$, Stream_Handle = SH_{IUT} = the value used for $SEID_{Tester}$ configuration, and SEID = $SEID_{Tester}$.
4. The IUT responds to the Upper Tester with AVDT_Start_Rsp with RSP.
5. The IUT sends an AVDTP_START_RSP to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_Type = 10 (response accept), and Signal_Id = 0x07 (AVDTP_START).
6. The Lower Tester sends an AVDTP_Media_Packet with V, P, X, CC, M, PT, Sequence_Nr, Time_Stamp, SSRC, CSRC, and Media_Payload set to valid values.
7. The Upper Tester sends an AVDT_ReadStreamData command to the IUT with Stream Handle = SH_{IUT}, ExpLength = $L_{DataBuffer}$ (stream data frame buffer length) = Media_Payload expected length of AVDTP media frame, and InBuffer = $ADDR_{DataBuffer}$.
8. The IUT responds to the Upper Tester with AVDT_ReadStreamData with Result = 0 (successful data transfer), Length = Media_Payload length of AVDTP media frame, Time_Info = $Time\_Stamp_{Tester}$, Marker = one of the possible values (1 or 0), Reliability = 0 (no errors), and Payload type = PT_{IUT} = value chosen according to the implementation in the IUT.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the Start_Ind_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = SH_{IUT} = the value used for $SEID_{Tester}$ configuration

  In Step 4, the IUT sends the AVDT_Start_Rsp command to the Upper Tester with RSP.

  In Step 5, the IUT sends the AVDTP_START_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_Type = 10 (response accept)
  - Signal_Id = 0x07 (AVDTP_START)

  In Step 8, the IUT sends the AVDT_ReadStreamData command to the Upper Tester with the following parameters:

  - Result = 0 (successful data transfer)
  - Length = Media_Payload length of AVDTP media frame
  - Time_Info = $Time\_Stamp_{Tester}$
  - Marker = one of the possible values (1 or 0)
  - Reliability = 0 (no errors)
  - Payload type = PT_{IUT} = value chosen according to the implementation in the IUT

4.4.1.1 Set Configuration Reject response following incorrect media transport format – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the format of the media transport is incorrect.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not yet configured $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/BTR/BI-01-C [Set Configuration Reject response following incorrect media transport format – ACP]
AVDTP/SNK/ACP/TRA/BTR/BI-01-C [Set Configuration Reject response following incorrect media transport format – ACP]

Table 4.64: Set Configuration Reject response following incorrect media transport format test cases

- Test Procedure

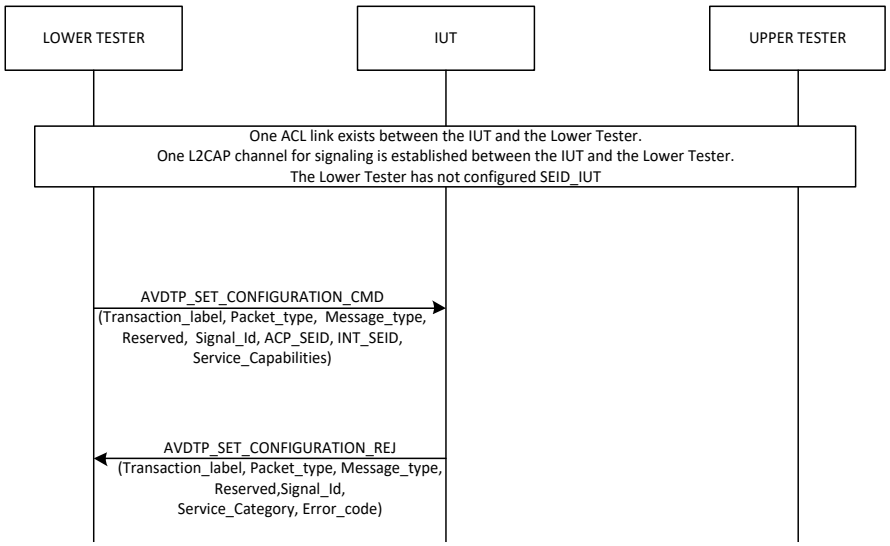

Figure 4.72: Set Configuration Reject response following incorrect media transport format – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an invalid format of the media transport capability (LOSC not equal to 0x00).
2. The IUT responds to the Lower Tester with AVDTP_SET_CONFIGURATION_REJ with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x23 (BAD_MEDIA_TRANSPORT_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x23 (BAD_MEDIA_TRANSPORT_FORMAT)

4.4.2 Reporting Service

Verify that an AVDTP implementation providing reporting features exposes the related capabilities and associates a reporting channel to each media transport session.

The compliance of AVDTP implementations with the RTCP specification is not subjected to testing.

4.4.2.1 Reporting Service – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes the Reporting Service indication with the SEP capabilities.

- Reference

  [3] 6.4, 6.7, 8.7, 13.1

  [5] 6.4, 6.7, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEID of the IUT.
  - The AVDTP_Get_Capabilities_Ind event callback function GetCapabilitiesInd_CB has been successfully registered.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/REP/BV-01-C [Reporting Service – ACP]
AVDTP/SNK/ACP/TRA/REP/BV-01-C [Reporting Service – ACP]

Table 4.65: Reporting Service test cases

- Test Procedure

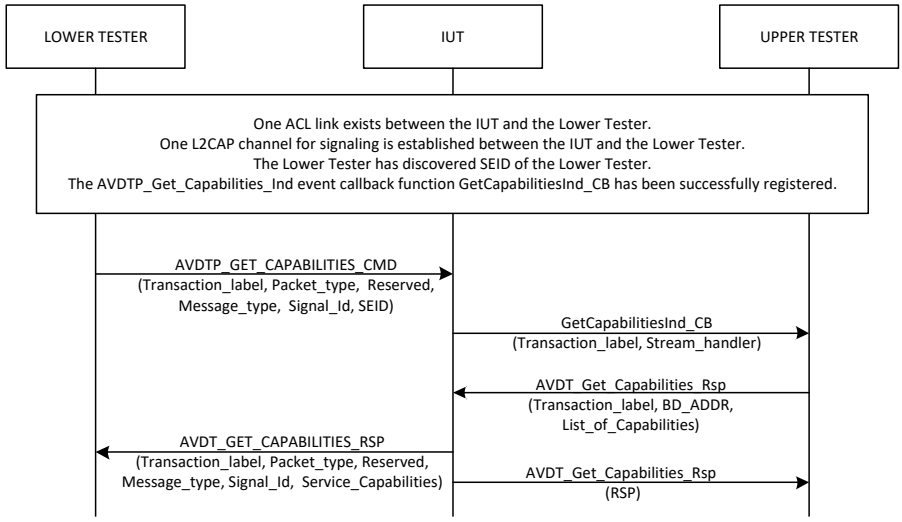

Figure 4.73: Reporting Service – ACP MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB event to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester sends an AVDT_Get_Capabilities_Rsp response to the IUT with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = list of valid capabilities of $SEID_{Tester}$.
4. The IUT sends an AVDTP_GET_CAPABILITIES_RSP to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Service_Capabilities = valid capabilities for supported services including reporting.
5. The IUT sends an AVDT_Get_Capabilities_Rsp to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the Get_Capabilities_Ind_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_CAPABILITIES_RSP to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - Service_Capabilities = Valid capabilities for supported services including reporting.

  The reporting capabilities:

  - Service_category = Reporting
  - LOSC = 0

  In Step 5, the IUT sends the AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.2.2 Reporting channel association – INT

- Test Purpose

  Verify that the IUT (either SRC or SNK) associates a reporting channel to each opened media transport session.

- Reference

  [3] 7.3, 13.1

  [5] 7.3, 13.1

- Initial Condition
  - One L2CAP channel for signaling is established between the IUT and the Lower Tester.
  - The Lower Tester has discovered two stream end points: SEID1 and SEID2. The service capabilities of both stream end points have been queried and configured.
  - The IUT is in the Configure state for both stream end points.
  - The AVDT_Open_Cfm event has been successfully registered.
  - The IUT is in the INT role.
- Test Case Configuration

Test Case
AVDTP/SRC/INT/TRA/REP/BV-02-C [Reporting channel association – INT]
AVDTP/SNK/INT/TRA/REP/BV-02-C [Reporting channel association – INT]

Table 4.66: Reporting channel association test cases

- Test Procedure

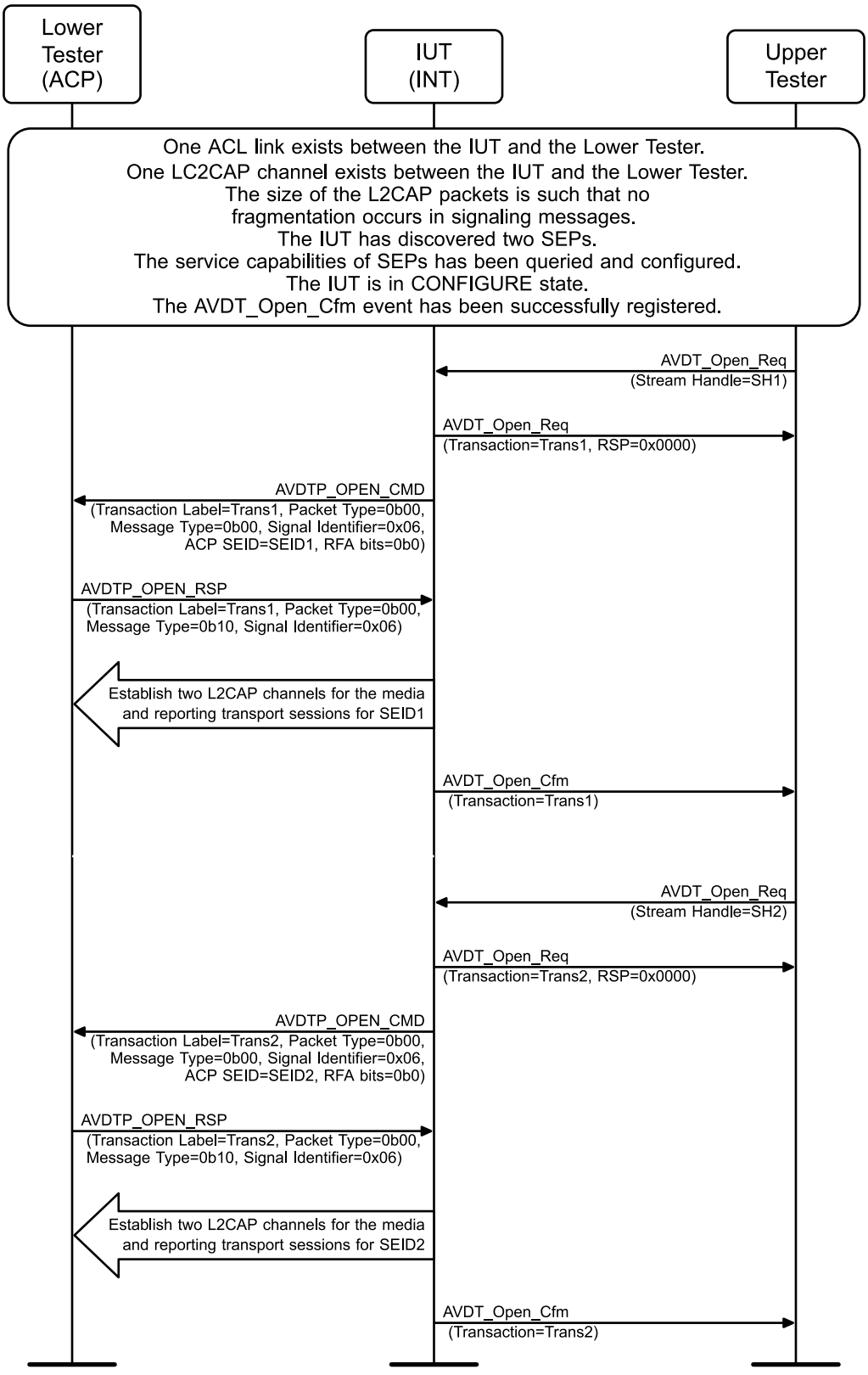

Figure 4.74: Reporting channel association – INT MSC

1. The Upper Tester sends AVDT_Open_Req to the IUT with the Stream Handle identified as SH1 in this procedure.
2. The IUT responds with an AVDT_Open_Req response with the RSP field set to 0x0000. The returned Transaction label is identified as Trans1 in this procedure.
3. The IUT sends AVDTP_OPEN_CMD to the Lower Tester with the Transaction Label set to Trans1. The Packet Type is set to 0b00, the Message Type is set to 0b00, the Signal Identifier is 0x06, the ACP SEID is SEID1, and all RFA bits are 0b0.
4. The Lower Tester sends AVDTP_OPEN_RSP with the Transaction Label set to Trans1. The Packet Type is set to 0b00, the Message Type is set to 0b10, and the Signal Identifier is set to 0x06.
5. The IUT establishes two L2CAP channels: one for the media transport session and one for the reporting transport session for SEID1.
6. The IUT sends the AVDT_Open_Cfm event to the Upper Tester with the Transaction label set to Trans1. No error code is present.
7. The Upper Tester sends AVDT_Open_Req to the IUT with the Stream Handle identified as SH2 in this procedure.
8. The IUT responds with an AVDT_Open_Req response with the RSP field set to 0x0000. The returned Transaction label is identified as Trans2 in this procedure.
9. The IUT sends AVDTP_OPEN_CMD to the Lower Tester with the Transaction Label set to Trans2. The Packet Type is set to 0b00, the Message Type is set to 0b00, the Signal Identifier is 0x06, the ACP SEID is SEID2, and all RFA bits are 0b0.
10. The Lower Tester sends AVDTP_OPEN_RSP with the Transaction Label set to Trans2. The Packet Type is set to 0b00, the Message Type is set to 0b10, and the Signal Identifier is set to 0x06.
11. The IUT establishes two L2CAP channels: one for the media transport session and one for the reporting transport session for SEID2.
12. The IUT sends the AVDT_Open_Cfm event to the Upper Tester with the Transaction label set to Trans2. No error code is present.

- Expected Outcome

  Pass verdict

  The IUT establishes two L2CAP channels: one for the media transport session and one for the reporting transport session for SEID1.

  The IUT establishes two L2CAP channels: one for the media transport session and one for the reporting transport session for SEID2.

4.4.2.3 Set Configuration Reject response with Bad Length or Bad Payload Format – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the report type is undefined or if the format of the reporting service capability is incorrect.

- Reference

  [3] 6.4, 6.8, 8.8, 13.1

  [5] 6.4, 6.9, 8.9, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not yet configured $SEID_{IUT}$.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/REP/BI-02-C [Set Configuration Reject response with Bad Length or Bad Payload Format – ACP]
AVDTP/SNK/ACP/TRA/REP/BI-02-C [Set Configuration Reject response with Bad Length or Bad Payload Format – ACP]

Table 4.67: Set Configuration Reject response with Bad Length or Bad Payload Format – ACP test cases

- Test Procedure

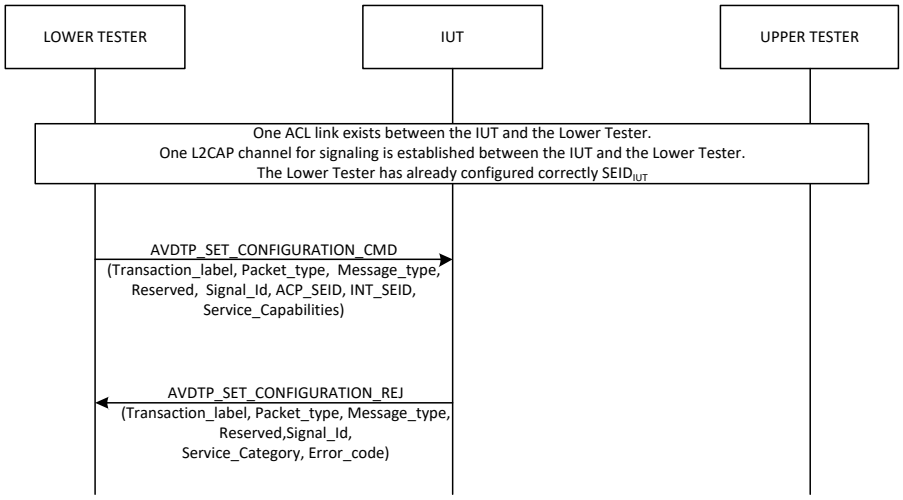

Figure 4.75: Set Configuration Reject response with Bad Length or Bad Payload Format – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an incorrect format of the reporting service capability (e.g., LOSC = 0x01).
2. The IUT responds to the Lower Tester with AVDTP_SET_CONFIGURATION_REJ with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x11 (BAD_LENGTH) or code 0x18 (BAD_PAYLOAD_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the following parameters:
  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)

- Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
- Error_code = 0x11 (BAD_LENGTH) or code 0x18 (BAD_PAYLOAD_FORMAT)

4.4.3 Recovery Service

Verify that an AVDTP implementation providing recovery features exposes the related capabilities.

The compliance of AVDTP implementations with the RFC2733 specification is not subjected to testing.

4.4.3.1 Recovery Service – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes the following parameters together with the SEP capabilities: Recovery type (RFC2733), maximum window size of a media packet sequence and maximum number of media packets that can be used to compute a recovery packet.

- Reference

  [3] 6.4, 6.7, 7.4, 8.7, 13.1

  [5] 6.4, 6.7, 7.4, 8.7, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SIED of the Upper Tester, $SEID_{Tester}$.
  - The AVDTP_Get_Capabilities_Ind event callback function GetCapabilitiesInd_CB has been successfully registered.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/REC/BV-01-C [Recovery Service – ACP]
AVDTP/SNK/ACP/TRA/REC/BV-01-C [Recovery Service – ACP]

Table 4.68: Recovery Service – ACP test cases

- Test Procedure

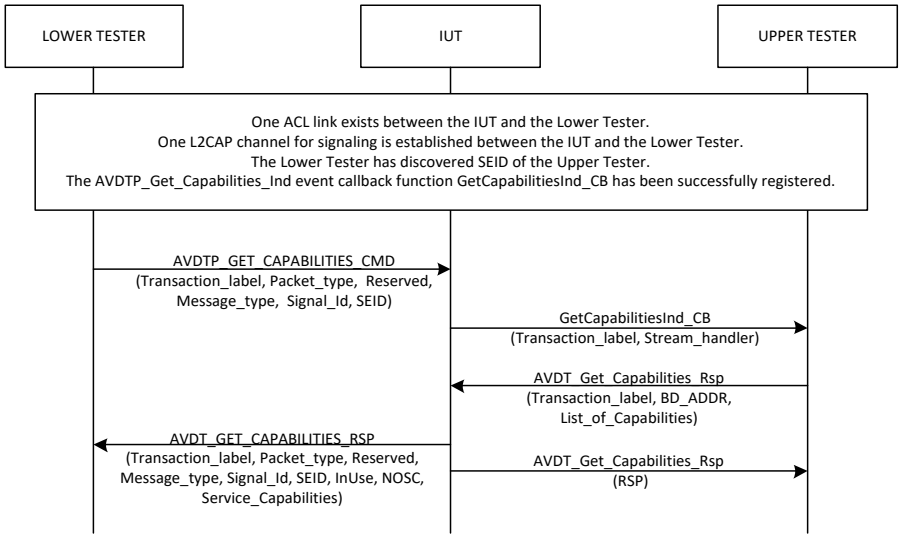

Figure 4.76: Recovery Service – ACP MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB callback function to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester sends an AVDT_Get_Capabilities_Rsp to the IUT with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, List_of_Capabilities = List of valid capabilities of $SEID_{Tester}$.
4. The IUT sends an AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Service_Capabilities = Valid capabilities for supported services including recovery.
5. The IUT sends an AVDT_Get_Capabilities_Rsp response to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - Service_Capabilities = Valid capabilities for supported services including recovery

The recovery capabilities:

- Service_category = Recovery
- LOSC = 3
- Recovery Type
- MRWS = Max_Recovery_Windows_Size
- MNMP = Max_Nr_of_Media_Packets_in_Parity_Code

In Step 5, the IUT sends the AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.3.2 Recovery Service when using GetAllCapabilities – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes the following parameters together with the SEP capabilities: Recovery type (RFC2733), maximum window size of a media packet sequence and maximum number of media packets that can be used to compute a recovery packet, when using the GetAllCapabilities command.

- Reference

  [3] 6.4, 6.8, 7.4, 8.8, 13.1

  [5] 6.4, 6.8, 7.4, 8.8, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEID of the Upper Tester, $SEID_{Tester}$.
  - The AVDTP_Get_Capabilities_Ind event callback function GetCapabilitiesInd_CB has been successfully registered.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/REC/BV-02-C [Recovery Service when using GetAllCapabilities – ACP]
AVDTP/SNK/ACP/TRA/REC/BV-02-C [Recovery Service when using GetAllCapabilities – ACP]

Table 4.69: Recovery Service when using GetAllCapabilities test cases

- Test Procedure

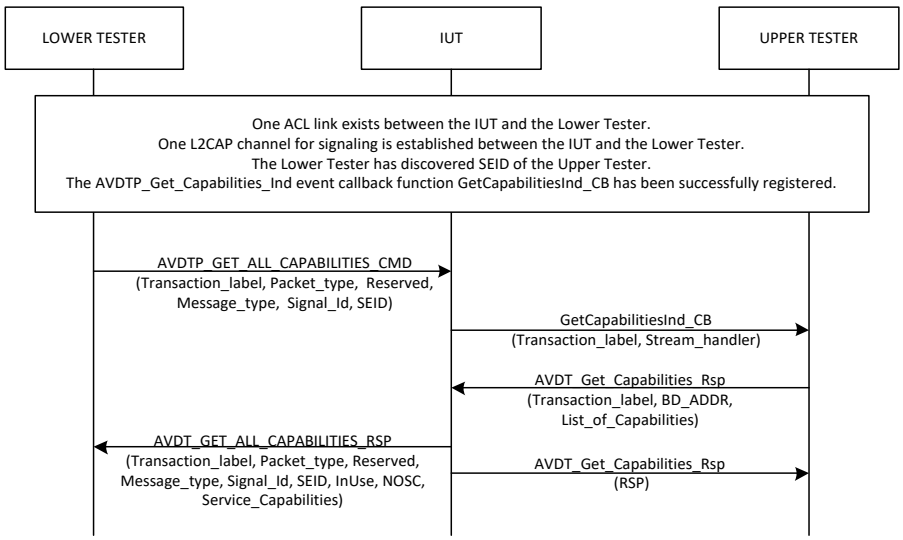

Figure 4.77: Recovery Service when using GetAllCapabilities – ACP MSC

1. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB callback function to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester sends an AVDT_GetCapabilities_Rsp command to the IUT with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = List of valid capabilities of $SEID_{Tester}$.
4. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Service_Capabilities = Valid capabilities for supported services including recovery.
5. The IUT sends an AVDT_Get_Capabilities_Rsp response to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
  - Service_Capabilities = Valid capabilities for supported services including recovery

The recovery capabilities:

- Service_category = Recovery
- LOSC = 3
- Recovery Type
- MRWS = Max_Recovery_Windows_Size
- MNMP = Max_Nr_of_Media_Packets_in_Parity_Code

In Step 5, the IUT sends the AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.3.3 Set Configuration Reject response with bad recovery format – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the recovery type is undefined or if the format of the recovery service capability is incorrect.

- Reference

  [3] 6.4, 6.7, 7.4, 8.7, 13.1

  [5] 6.4, 6.7, 7.4, 8.7, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not yet configured $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/REC/BI-01-C [Set Configuration Reject response with bad recovery format – ACP]
AVDTP/SNK/ACP/TRA/REC/BI-01-C [Set Configuration Reject response with bad recovery format – ACP]

Table 4.70: Set Configuration Reject response with bad recovery format – ACP test cases

- Test Procedure

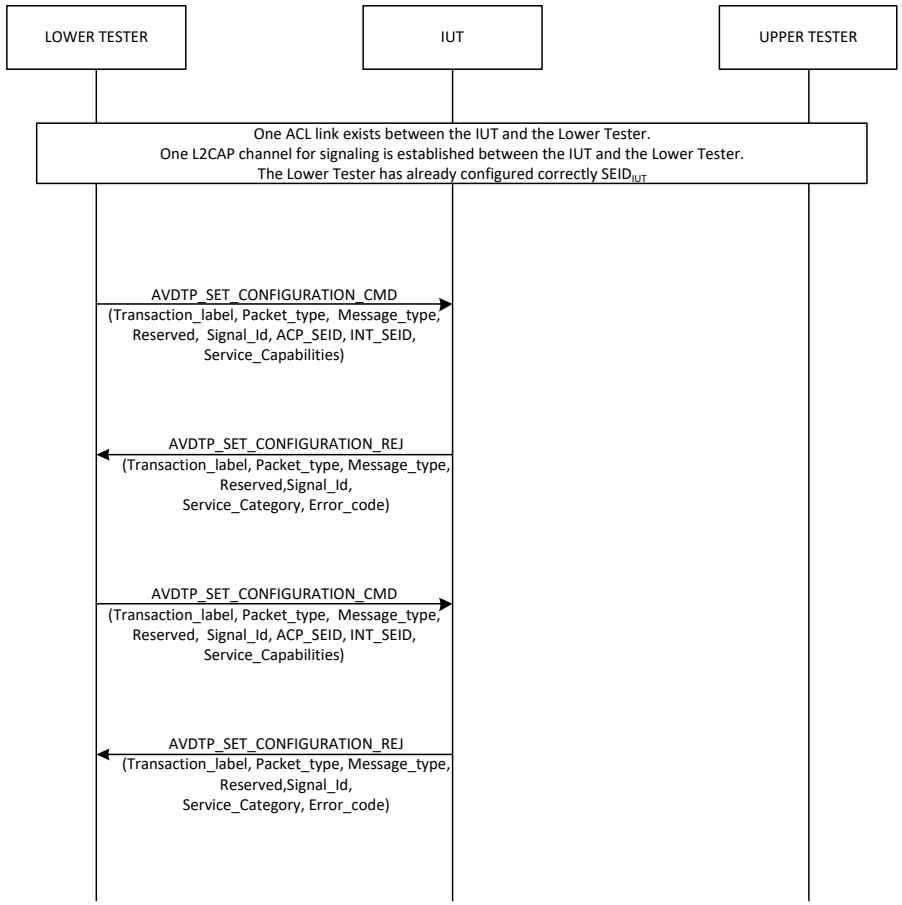

Figure 4.78: Set Configuration Reject response with bad recovery format – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including a recovery type value that is not defined in the AVDTP specification (e.g., 0xFF).
2. The IUT responds to the Lower Tester with an AVDTP_SET_CONFIGURATION_REJ response with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x22 (BAD_RECOVERY_TYPE).
3. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an incorrect format of the recovery service capability (e.g., LOSC = 0x01).
4. The IUT responds to the Lower Tester with an AVDTP_SET_CONFIGURATION_REJ response with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x25 (BAD_RECOVERY_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x22 (BAD_RECOVERY_TYPE)

  In Step 4, the IUT sends the AVDTP signaling message to the Lower Tester with the fields:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x25 (BAD_RECOVERY_FORMAT)

4.4.4 Multiplexing Service

Verify that the Multiplexing Service is implemented according to its specification in AVDTP.

AVDTP/SRC/INT/TRA/MUX/BV-01-C [Multiplexing Service]

- Test Purpose

  Verify that the IUT (SRC) is able to multiplex several transport sessions on a transport channel. The management of the application layer header is also checked.

- Reference

  [3] 7.5, 8.19.8

  [5] 7.5, 8.21.8

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The $SEID_{LowerTester1}$, and $SEID_{LowerTester2}$ stream end points have been discovered.
  - The capabilities have been queried and both SEPs are configured for basic transport capabilities and MUX service (with no Adaptation Layer Fragmentation) over one transport channel common to both transport sessions (one for each SEP).
  - Both SEPs are in the Streaming state.

- Test Procedure

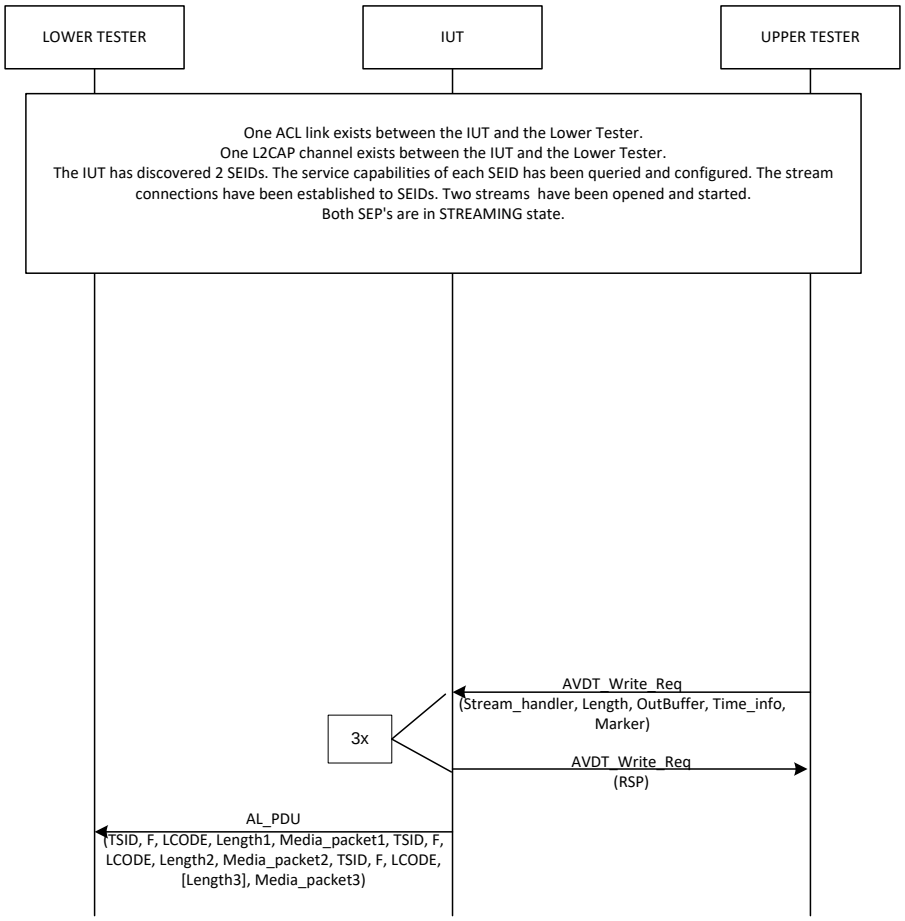

Figure 4.79: AVDTP/SRC/INT/TRA/MUX/BV-01-C [Multiplexing Service] MSC

Repeat Steps 1-2 for each round in Table 4.71:

1. The Upper Tester sends an AVDT_Write_Req command to the IUT with the parameters in Table 4.71.
2. The IUT responds to the Upper Tester with an AVDT_Write_Req with RSP.
3. The IUT sends an Adaptation Layer PDU packet with AL header 1: TSID = TSID1, F = 0 (unfragmented media packet), LCODE = 01, 10 or 11 (length field), and Length1 = length of mediapacket1 and Media packet1 = complete mediapacket, AL header 2: TSID = TSID2, F = 0 (unfragmented media packet), LCODE = 01, 10 or 11 (length field), and Length2 = length of mediapacket2 and Media packet2 = complete mediapacket2, and AL header 3: TSID = TSID1, F = 0 (unfragmented media packet), and LCODE = 00, 01, 10 or 11 (length field optional), and Length3 = length of mediapacket3 (if LCODE not 00) and Media packet3 = complete mediapacket3.

Round	Stream_Handler	Length	OutBuffer	Time_Info
1	SH1	$Length_{Out\_Buffer1}$	$ADDR_{Out\_Buffer1}$	$Time\_Stamp_{Tester1}$
2	SH2	$Length_{Out\_Buffer2}$	$ADDR_{Out\_Buffer2}$	$Time\_Stamp_{Tester2}$
3	SH1	$Length_{Out\_Buffer3}$	$ADDR_{Out\_Buffer3}$	$Time\_Stamp_{Tester3}$

Table 4.71: Multiplexing Service rounds

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDT_Write_Req command to the Upper Tester with the following parameters:

  - RSP = 0 (data frame accepted)

  In Step 3, the IUT sends the Adaptation Layer PDU packet to the Lower Tester with the following parameters:

  - AL header 1
    - TSID = TSID1
    - F = 0 (unfragmented media packet)
    - LCODE = 01, 10 or 11 (length field)
    - Length1 = length of mediapacket1
    - Media packet = complete mediapacket1
  - AL header 2
    - TSID = TSID2
    - F = 0 (unfragmented media packet)
    - LCODE = 01, 10 or 11 (length field)
    - Length2 = length of mediapacket2
    - Media packet = complete mediapacket2
  - AL header 3
    - TSID = TSID1
    - F = 0 (unfragmented media packet)
    - LCODE = 00, 01, 10 or 11 (length field optional)
    - Length3 = length of mediapacket3 (if LCODE not 00)
    - Media packet = complete mediapacket3

  Mediapacket1, Mediapacket2, and Mediapacket3 can be in any order, but Mediapacket1 is before Mediapacket3.

AVDTP/SNK/ACP/TRA/MUX/BV-02-C [SNK is able to demultiplex transport sessions]

- Test Purpose

  Verify that the IUT (SNK) is able to demultiplex several transport sessions from a transport channel. The management of the application layer header is also checked.

- Reference

  [3] 7.5, 8.19.8

  [5] 7.5, 8.21.8

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered $SEID_{Tester1}$ and $SEID_{Tester2}$ stream end points.
  - The capabilities have been queried and both SEPs are configured for basic transport capabilities and MUX service (with no Adaptation Layer Fragmentation) over one transport channel common to both transport sessions (one for each SEP).
  - Both SEPs are in the Streaming state.
- Test Procedure

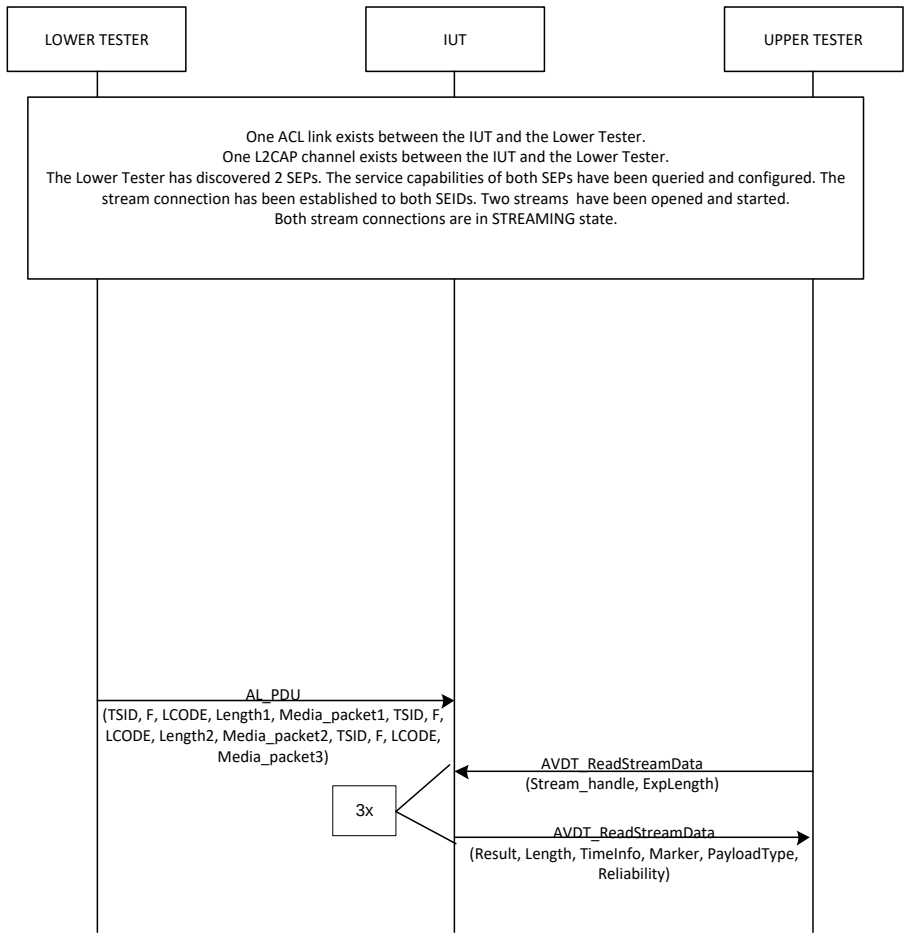

Figure 4.80: AVDTP/SNK/ACP/TRA/MUX/BV-02-C [SNK is able to demultiplex transport sessions] MSC

1. The Lower Tester sends an Adaptation Layer PDU packet with AL header 1: TSID = TSID1, F = 0 (unfragmented media packet), LCODE = 01, 10 or 11 (length field), and Length1 = length of mediapacket1 and Media packet1 = complete mediapacket; AL header 2: TSID = TSID2, F = 0 (unfragmented media packet), LCODE = 01, 10 or 11 (length field), and Length2 = length of mediapacket2 and Media packet2 = complete mediapacket; and AL header 3: TSID = TSID1, F = 0 (unfragmented media packet), and LCODE = 00 (no length field) and Media packet3 = complete mediapacket.

Repeat Steps 2–3 three times.

2. The Upper Tester sends an AVDT_ReadStreamData command with Stream_handler = twice stream handler of SEID1 and once stream handler of SEID2 and ExpLength.
3. The IUT responds to the Upper Tester with an AVDT_ReadStreamData with valid Result, Length, TimeInfo, Marker, PayloadType, and Reliability.

- Expected Outcome

  Pass verdict

  The IUT sends three consecutive AVDT_ReadStreamData output parameters that return Media_Packet1, Media_Packet2, Media_Packet3.

AVDTP/SRC/INT/TRA/MUX/BV-03-C [SRC fragments a transport packet]

- Test Purpose

  Verify that the IUT (SRC) is able to fragment a transport packet into several L2CAP packets. The management of the application layer header is also checked.

- Reference

  [3] 7.5, 8.19.8

  [5] 7.5, 8.21.8

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The $SEID_{LowerTester}$ stream end point has been discovered by the IUT.
  - The capabilities have been queried and the SEP is configured for basic transport capabilities and MUX service (with Adaptation Layer Fragmentation) over one transport channel.
  - The SEP is in the Open state.
  - The AVDT_Start_Cfm event callback function StartCfm_CB has been successfully registered for $SEID_{LowerTester}$.

- Test Procedure

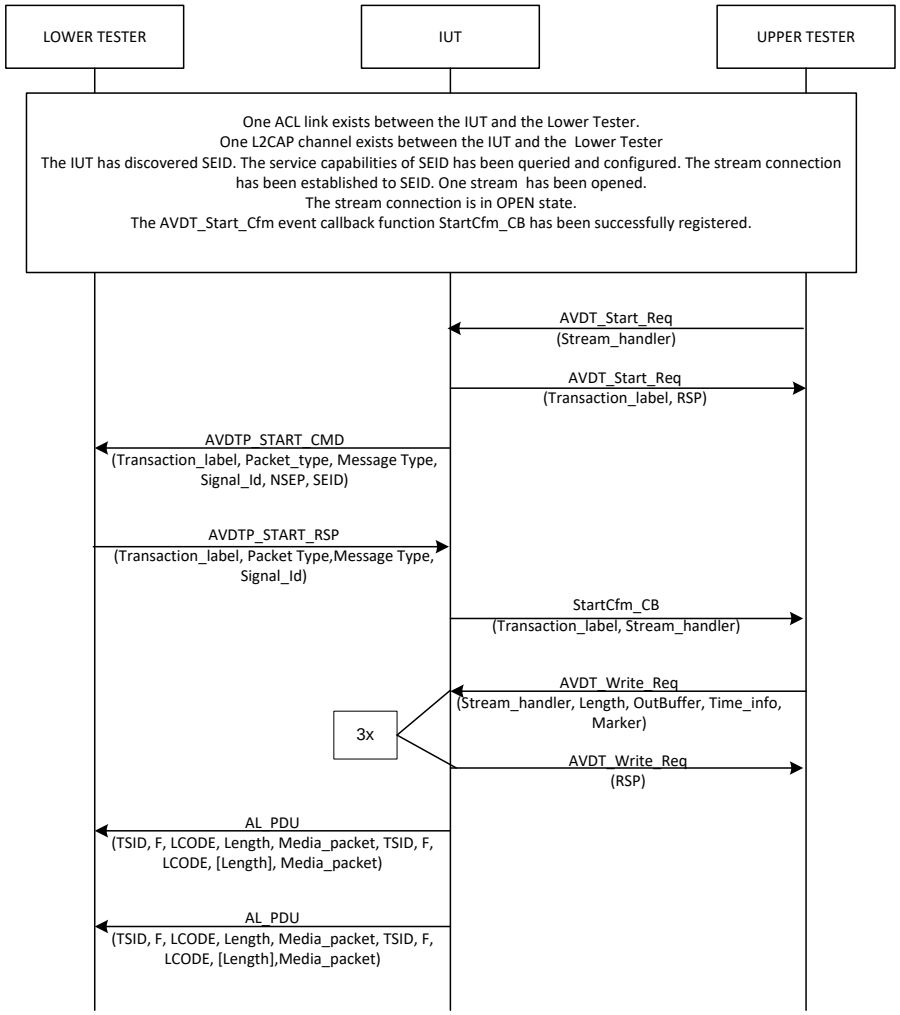

Figure 4.81: AVDTP/SRC/INT/TRA/MUX/BV-03-C [SRC fragments a transport packet] MSC

1. The Upper Tester sends an AVDT_Start_Req service primitive to the IUT with Stream_handler = $SH_{LowerTester}$.
2. The IUT responds with an AVDT_Start_Req response to the Upper Tester with AVDT_Start_Req with Transaction_label = $Transaction_{Test\_System}$ and RSP = 0.
3. The IUT sends an AVDTP_START_CMD command to the Lower Tester with Transaction_label = $Transaction_{Test\_System}$, Message_type = 0 (command), Signal_Id = (AVDTP_START), NSEP = 1, and SEID = $SEID_{Tester}$.
4. The Lower Tester responds to the IUT with an AVDTP_START_RSP response with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Message_type = 2 (response accepted), and Signal_Id = 0x07 (AVDTP_START).
5. The IUT sends a StartCfm_CB callback function to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = SH_{IUT}.

Repeat Steps 6–7 for each round in Table 4.72:

6. The Upper Tester sends an AVDT_Write_Req to the Upper Tester with Stream_Handler = SH_{IUT}, Length set to the value in Table 4.72, OutBuffer set to the value in Table 4.72, Time_Info = $Time\_Stamp_{Tester}$, and Market.
7. The IUT responds to the Upper Tester with AVDT_Write_Req with RSP.

8. The IUT sends an Adaptation Layer PDU packet with AL header 1: TSID = $TSID_{Media}$, F = 0 (unfragmented media packet), LCODE = 01, 10 or 11 (length field), and Length1 = length of mediapacket1 and Media packet = complete media packet and AL header 2: TSID = $TSID_{Recover}$, F = 0 (fragmented media packet), LCODE = 01 (16 bit length field), and Length = $Length_{MediaPacket}$ (complete media packet length) and Media packet = fragmented media packet (first part).
9. The IUT sends an Adaptation Layer PDU packet with AL header 1: TSID = $TSID_{Recover}$, F = 1 (fragmented media packet), LCODE = 01 (16 bit length field), and Length = SegmentLength (media packet segment length) and Media packet = fragmented media packet (second part) and AL header 2: TSID = $TSID_{Media}$, F = 0 (unfragmented media packet), LCODE = 00, 01, 10 or 11 (length field optional), and Length3 = length of mediapacket3 (if LCODE not 00) and Media packet = complete media packet.

Round	Length	OutBuffer
1	Length = $Length_{Out\_Buffer1}$	OutBuffer = $ADDR_{Out\_Buffer1}$ such that Out_Buffer1 fits into one AL_PDU.
2	Length = $Length_{Out\_Buffer2}$	OutBuffer = $ADDR_{Out\_Buffer2}$ such that Out_Buffer1 + Out_Buffer2 do not fit into one AL_PDU but fit into two AL_PDUs.
3	Length = $Length_{Out\_Buffer3}$	OutBuffer = $ADDR_{Out\_Buffer3}$ such that Out_Buffer1 + Out_Buffer2 + Out_Buffer3 fit exactly into two AL_PDUs (in order to flush the second AL_PDU).

Table 4.72: SRC fragments a transport packet rounds

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDT_Start_Req command to the Upper Tester with the following parameters:

  - Transaction_label = $Transaction_{Test\_System}$
  - RSP = 0

  In Step 3, the IUT sends the AVDTP_START_CMD command to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{Test\_System}$
  - Message_type = 0 (command)
  - Signal_Id = (AVDTP_START)
  - NSEP = 1
  - SEID = $SEID_{Tester}$

  In Step 5, the IUT invokes the StartCfm_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = SH_{IUT}

  In Step 7, the IUT sends the AVDT_Write_Req command to the Upper Tester with the following parameters:

  - Result = 0 (data frame accepted)

In Step 8, the IUT sends the Adaptation Layer PDU packet 1 to the Lower Tester with the following parameters:

- AL header 1
  - TSID = $TSID_{Media}$
  - F = 0 (unfragmented media packet)
  - LCODE = 01, 10 or 11 (length field)
  - Length1 = length of mediapacket1
  - Media packet = complete media packet
- AL header 2
  - TSID = $TSID_{Recover}$
  - F = 0 (fragmented media packet)
  - LCODE = 01 (16-bit length field)
  - Length = $Length_{MediaPacket}$ (complete media packet length)
  - Media packet = fragmented media packet (first part)

In Step 9, the IUT sends the Adaptation Layer PDU packet 2 to the Lower Tester with the following parameters:

- AL header 1
  - TSID = $TSID_{Recover}$
  - F = 1 (fragmented media packet)
  - LCODE = 01 (16 bit length field)
  - Length = SegmentLength (media packet segment length)
  - Media packet = fragmented media packet (second part)
- AL header 2
  - TSID = $TSID_{Media}$
  - F = 0 (unfragmented media packet)
  - LCODE = 00, 01, 10 or 11 (length field optional)
  - Length3 = length of mediapacket3 (if LCODE not 00)
  - Media packet = complete media packet

AVDTP/SNK/ACP/TRA/MUX/BV-04-C [SNK reconstructs a transport packet]

- Test Purpose

  Verify that the IUT (SNK) is able to reconstruct a transport packet fragmented into several L2CAP payloads. The management of the application layer header is also checked.

- Reference

  [3] 7.5, 8.19.8

  [5] 7.5, 8.21.8

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEP identified by $SEID_{Tester}$.
  - The capabilities have been queried and the SEP is configured for basic transport capabilities and MUX service (with Adaptation Layer Fragmentation) over one transport channel.
  - The SEP is in the Open state.
  - The AVDT_Start_Ind event callback function StartInd_CB has been successfully registered for $SEID_{Tester}$.
- Test Procedure

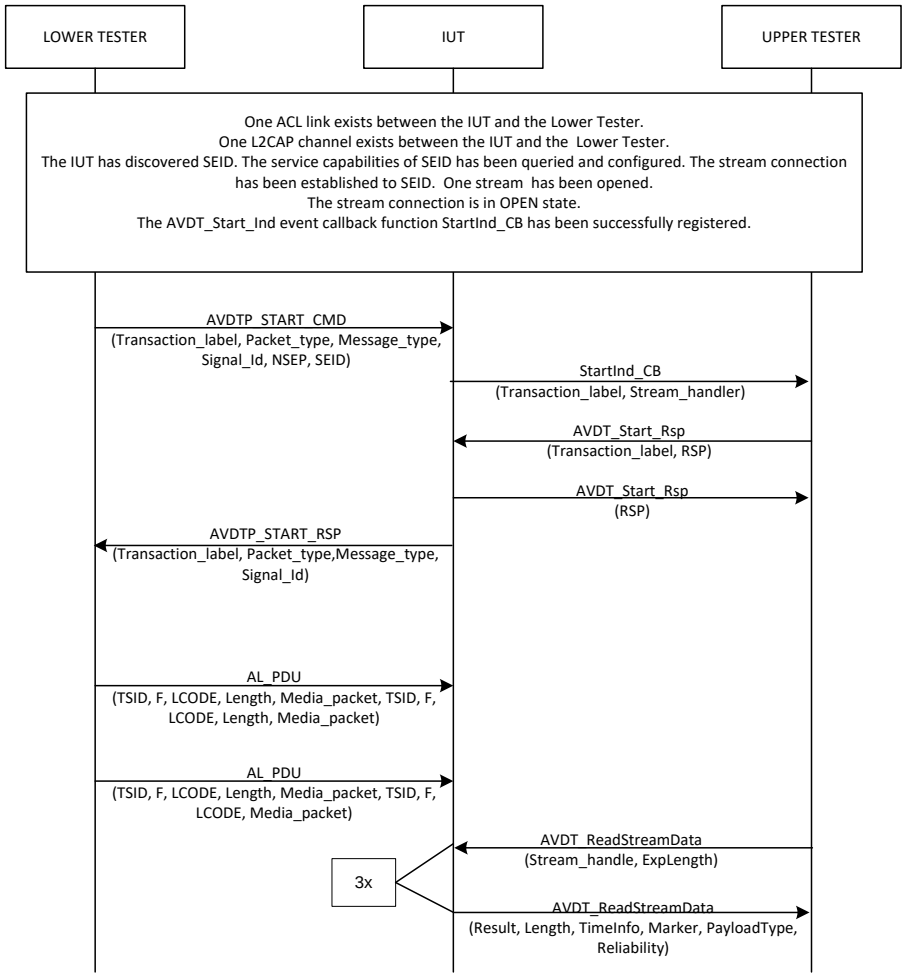

Figure 4.82: AVDTP/SNK/ACP/TRA/MUX/BV-04-C [SNK reconstructs a transport packet] MSC

1. The Lower Tester sends an AVDTP_START_CMD command to the IUT with Transaction_label = $Transaction_{Test\_System}$, Packet_type = 0 (single packet), Message_type = 0 (command), Signal_Id = 0x07 (AVDTP_START), NSEP = 1, and SEID = $SEID_{Tester}$.
2. The IUT sends a StartInd_CB callback function to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = SH_{IUT}.
3. The Upper Tester sends an AVDT_Start_Rsp service primitive to the IUT with Transaction_label and Stream_handler = $SH_{LowerTester}$.
4. The IUT responds to the Upper Tester with an AVDT_Start_Rsp event with RSP.

5. The IUT sends an AVDTP_START_RSP command to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 2 (response accepted), Signal_Id = 0x07 (AVDTP_START), NSEP = 1, and SEID = $SEID_{Tester}$.
6. The Lower Tester sends an Adaptation Layer PDU packet with AL header 1: TSID = $TSID_{Media}$, F = 0 (unfragmented media packet), 01, 10 or 11 (length field), and Length1 = length of mediapacket1 and Media packet = complete media packet and AL header 2: TSID = $TSID_{Recover}$, F = 0 (fragmented media packet), LCODE = 01 (length field coding), and Length = LengthMediaPacket (complete media packet length) and Media packet = fragmented media packet (first part).
7. The Lower Tester sends an Adaptation Layer PDU packet with AL header 1: TSID = $TSID_{Recover}$, F = 1 (fragmented media packet), LCODE = 01 (length field coding), and Length = SegmentLength (media packet segment length) and Media packet = fragmented media packet (second part) and AL header 2: TSID = $TSID_{Media}$, F = 0 (unfragmented media packet), and LCODE = 0 (no length field) and Media packet = complete media packet.

Repeat Steps 8–9 three times:

8. The Upper Tester sends an AVDT_ReadStreamData service primitive to the IUT with Stream_handler = $SH_{LowerTester}$ and ExpLength.
9. The IUT responds to the Upper Tester with an AVDT_ReadStreamData event with Result, Length, TimeInfo, Marker, PayloadType, and Reliability.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the StartInd_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = SH_{IUT}

  In Step 4, the IUT sends the AVDT_Start_Rsp event to the Upper Tester with RSP.

  In Step 5, the IUT sends the AVDTP_START_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 2 (response accepted)
  - Signal_Id = 0x07 (AVDTP_START)
  - NSEP = 1
  - SEID = $SEID_{Tester}$

  In Step 9, the IUT sends the AVDT_ReadStreamData event to the Upper Tester with Result, Length, TimeInfo, Marker, PayloadType, and Reliability.

4.4.4.1 GetCapabilities including Multiplexing Service capabilities – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes the following parameters together with the SEP capabilities: FRAG, Signal Header, TSID, TCID set according to the support claimed by the IUT.

- Reference

  [3] 6.4, 6.7, 7.5, 8.7, 8.21.8, 13.1

  [5] 6.4, 6.7, 7.5, 8.7, 8.21.8, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEID of the Upper Tester, $SEID_{Tester}$.
  - The AVDT_Get_Capabilities_Ind event callback function GetCapabilitiesInd_CB has been successfully registered.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/MUX/BV-05-C [GetCapabilities including Multiplexing Service capabilities – ACP]
AVDTP/SNK/ACP/TRA/MUX/BV-05-C [GetCapabilities including Multiplexing Service capabilities – ACP]

Table 4.73: GetCapabilities including Multiplexing Service capabilities – ACP test cases

- Test Procedure

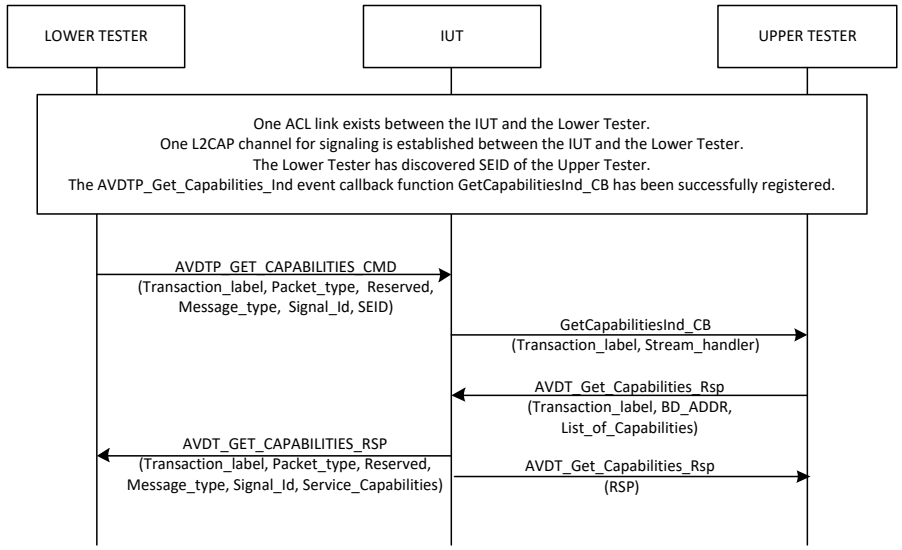

Figure 4.83: GetCapabilities including Multiplexing Service capabilities – ACP MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the Lower Tester with Transaction_label = $Transaction_{Test\_System}$, Packet_type = 00 (single packet), Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB event to the Upper Tester with valid Transaction_label and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester responds to the IUT with an AVDT_Get_Capabilities_Rsp service primitive with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = List of valid capabilities of $SEID_{Tester}$.
4. The IUT sends an AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Service_Capabilities = valid capabilities for supported services including multiplexing service capabilities.
5. The IUT sends an AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - Service_Capabilities = Valid capabilities for supported services including multiplexing service capabilities

  Multiplexing capabilities:

  - Service_category = Multiplexing mode
  - LOSC = $LOSC_{ExposedTransports}$ (e.g., if only the Media codec is exposed or configured then LOSC = 3, if both Media and Reporting or both Media and Recovery are exposed or configured then LOSC = 5, and if Media, Reporting, and Recovery are exposed or configured then LOSC = 7.)
  - FRAG = 1 (if fragmentation allowed, otherwise 0)
  - TSID = $TSID_{Media}$
  - TCID = $TCID_{Media}$
  - TSID = $TSID_{Reporting}$ (if Reporting is exposed or configured, otherwise not present)
  - TCID = $TCID_{Reporting}$ (if Reporting is exposed or configured, otherwise not present)
  - TSID = $TSID_{Recovery}$ (if Recovery is exposed or configured, otherwise not present)
  - TCID = $TCID_{Recovery}$ (if Recovery is exposed or configured, otherwise not present)

  In Step 5, the IUT sends the AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.4.2 Get All Capabilities including Multiplexing Service capabilities – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes the following parameters together with the SEP capabilities: FRAG, Signal Header, TSID, TCID set according to the support claimed by the IUT when using the Get All Capabilities command.

- Reference

  [3] 6.4, 6.7, 7.5, 8.7, 8.21.8, 13.1

  [5] 6.4, 6.7, 7.5, 8.7, 8.21.8, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel..
  - The Lower Tester has discovered the SEID of the Upper Tester, $SEID_{Tester}$.
  - The AVDT_Get_All_Capabilities_Ind event callback function GetAllCapabilitiesInd_CB has been successfully registered.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/MUX/BV-06-C [Get All Capabilities including Multiplexing Service capabilities – ACP]
AVDTP/SNK/ACP/TRA/MUX/BV-06-C [Get All Capabilities including Multiplexing Service capabilities – ACP]

Table 4.74: Get All Capabilities including Multiplexing Service capabilities – ACP test cases

- Test Procedure

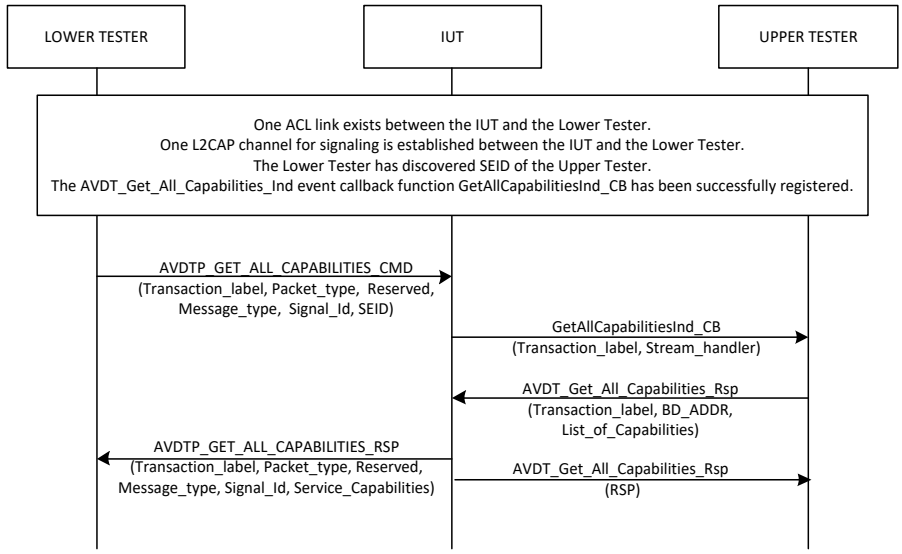

Figure 4.84: Get All Capabilities including Multiplexing Service capabilities – ACP MSC

1. The Lower Tester sends an AVDT_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{Test\_System}$, Packet_type = 00 (single packet), Message_type = 00 (command), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB callback function to the Upper Tester with a valid Transaction_label and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester responds to the IUT with an AVDT_Get_All_Capabilities_Rsp response with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = List of valid capabilities of $SEID_{Tester}$.
4. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Service_Capabilities = valid capabilities for supported services including multiplexing service capabilities.
5. The IUT sends an AVDT_Get_All_Capabilities_Rsp event to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_ALL_CAPABILITIES_RSP response with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)
  - Service_Capabilities = Valid capabilities for supported services including multiplexing service capabilities

  Multiplexing capabilities:

  - Service_category = Multiplexing mode
  - LOSC = $LOSC_{ExposedTransports}$ (e.g., if only the Media codec is exposed or configured then LOSC = 3, if both Media and Reporting or both Media and Recovery are exposed or configured then LOSC = 5, and if Media, Reporting, and Recovery are exposed or configured then LOSC = 7.)
  - FRAG = 1 (if fragmentation allowed, otherwise 0)
  - TSID = $TSID_{Media}$
  - TCID = $TCID_{Media}$
  - TSID = $TSID_{Reporting}$ (if Reporting is exposed or configured, otherwise not present)
  - TCID = $TCID_{Reporting}$ (if Reporting is exposed or configured, otherwise not present)
  - TSID = $TSID_{Recovery}$ (if Recovery is exposed or configured, otherwise not present)
  - TCID = $TCID_{Recovery}$ (if Recovery is exposed or configured, otherwise not present)

  In Step 5, the IUT sends the AVDT_Get_All_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.4.3 Set Configuration Reject response for incorrect format of Multiplexing Service capability – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the format of the Multiplexing Service capability is incorrect.

- Reference

  [3] 6.4, 6.8, 7.5, 8.8, 13.1

  [5] 6.4, 6.9, 7.5, 8.9, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel..
  - The Lower Tester has not yet configured $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/MUX/BI-01-C [Set Configuration Reject response for incorrect format of Multiplexing Service capability – ACP]
AVDTP/SNK/ACP/TRA/MUX/BI-01-C [Set Configuration Reject response for incorrect format of Multiplexing Service capability – ACP]

Table 4.75: Set Configuration Reject response for incorrect format of Multiplexing Service capability – ACP test cases

- Test Procedure

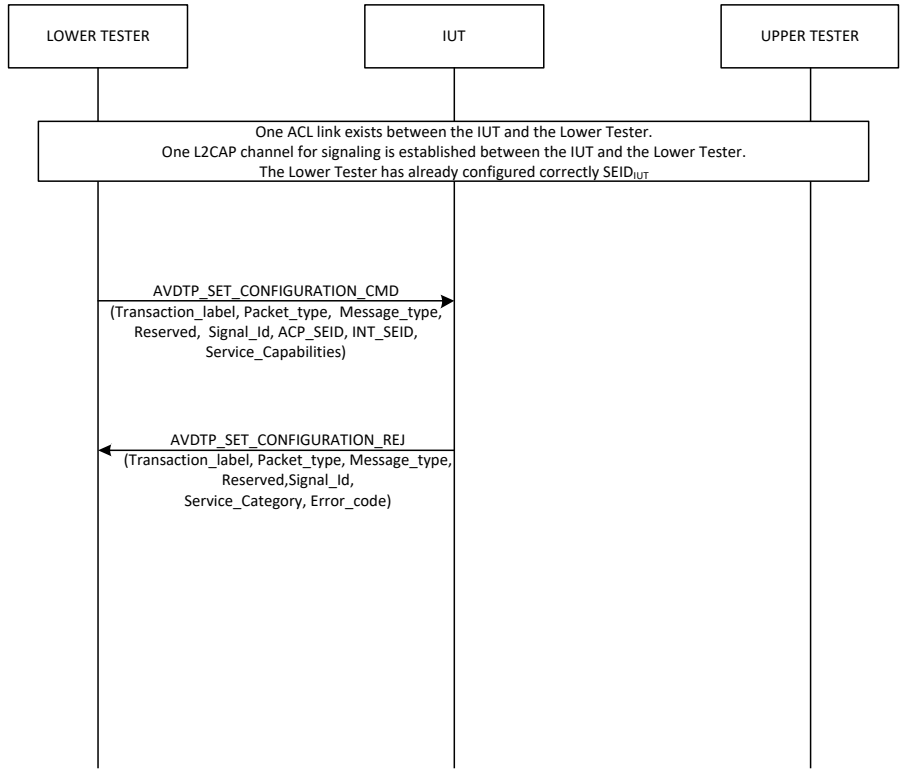

Figure 4.85: Set Configuration Reject response for incorrect format of Multiplexing Service capability – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{Lower\_Tester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an invalid format of the multiplexing service capability (LOSC equal to 0x00).
2. The IUT responds to the Lower Tester with an AVDTP_SET_CONFIGURATION_REJ command with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x28 (BAD_MULTIPLEXING_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x28 (BAD_MULTIPLEXING_FORMAT)

4.4.5 Robust Header Compression Service

Verify that an AVDTP implementation providing the ROHC features exposes the related capabilities.

According to Section 3.4 of [3] and [5], the compliance of AVDTP implementations with the ROHC specification is not subjected to testing.

4.4.5.1 Robust Header Compression Service – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes, together with the SEP capabilities, the following flags set according to the support claimed by the IUT: Basic, Recovery, BackCh.

- Reference

  [3] 6.4, 6.7, 7.6, 8.7, 13.1

  [5] 6.4, 6.7, 7.6, 8.7, 13.1

- Initial Condition

  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEID of the Upper Tester, $SEID_{Tester}$.
  - The AVDTP_Get_Capabilities_Ind event callback function GetCapabilitiesInd_CB has been successfully registered.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/RHC/BV-01-C [Robust Header Compression Service – ACP]
AVDTP/SNK/ACP/TRA/RHC/BV-01-C [Robust Header Compression Service – ACP]

Table 4.76: Robust Header Compression Service – ACP test cases

- Test Procedure

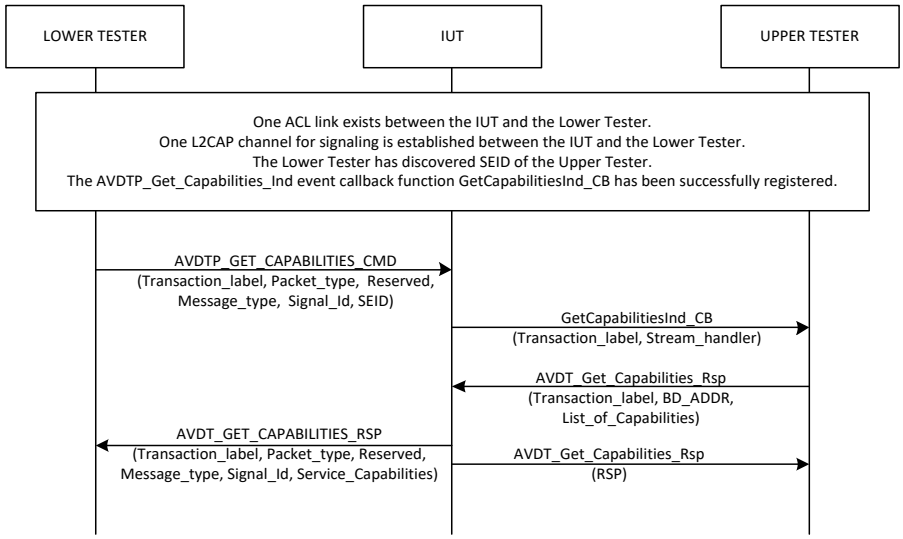

Figure 4.86: Robust Header Compression Service – ACP MSC

1. The Lower Tester sends an AVDTP_GET_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetCapabilitiesInd_CB event to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester responds to the IUT with an AVDT_Get_Capabilities_Rsp with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = List of all capabilities supported by the SEP.
4. The IUT sends an AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES), and Service_Capabilities = 2 valid capabilities including header compression.
5. The IUT sends an AVDT_Get_Capabilities_Rsp event to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetCapabilitiesInd_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_CAPABILITIES_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x02 (AVDTP_GET_CAPABILITIES)
  - Service_Capabilities = 2 valid capabilities including header compression.

- The header compression capabilities:
  - Service_category = Header_Compression
  - LOSC = 1
  - BachCh = 1 (if back channel should be used)
  - Basic = 1 (if service available for basic service)
  - Recovery = 1 (if service available for recovery service)

In Step 5, the IUT sends the AVDT_Get_All_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.5.2 GetAllCapabilities with Robust Header Compression Service – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) exposes, together with the SEP capabilities, the following flags set according to the support claimed by the IUT: Basic, Recovery, BackCh, when using the GetAllCapabilities command.

- Reference

  [3] 6.4, 6.8, 7.6, 8.8, 13.1

  [5] 6.4, 6.8, 7.6, 8.8, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has discovered the SEID of the Upper Tester, $SEID_{Tester}$.
  - The AVDTP_Get_All_Capabilities_Ind event callback function GetAllCapabilitiesInd_CB has been successfully registered.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/RHC/BV-02-C [GetAllCapabilities with Robust Header Compression Service – ACP]
AVDTP/SNK/ACP/TRA/RHC/BV-02-C [GetAllCapabilities with Robust Header Compression Service – ACP]

Table 4.77: GetAllCapabilities with Robust Header Compression Service – ACP test cases

- Test Procedure

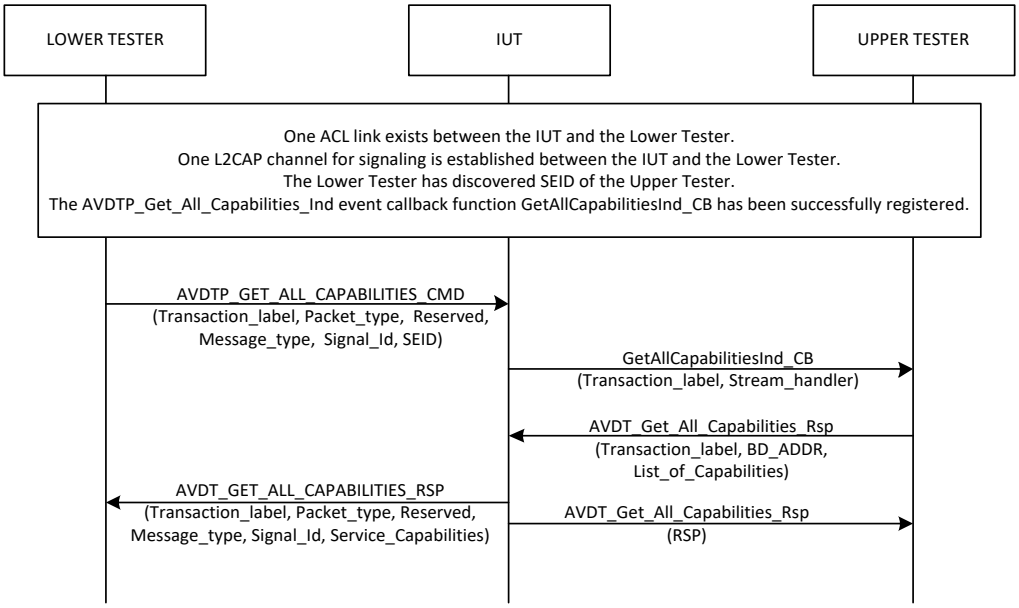

Figure 4.87: GetAllCapabilities with Robust Header Compression Service – ACP MSC

1. The Lower Tester sends an AVDTP_GET_ALL_CAPABILITIES_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and SEID = $SEID_{Tester}$.
2. The IUT sends a GetAllCapabilitiesInd_CB callback function to the Upper Tester with Transaction_label = $Transaction_{LowerTester}$ and Stream_handler = $SH_{LowerTester}$.
3. The Upper Tester responds to the IUT with an AVDT_Get_All_Capabilities_Rsp with Transaction_label = $Transaction_{LowerTester}$, BD_ADDR = $BD\_ADDR_{LowerTester}$, and List_of_Capabilities = List of all capabilities supported by the SEP.
4. The IUT sends an AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with Transaction_label = $Transaction_{LowerTester}$, Message_type = 10 (response accept), Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES), and Service_Capabilities = two valid capabilities including header compression.
5. The IUT sends an AVDT_Get_All_Capabilities_Rsp event to the Upper Tester with RSP.

- Expected Outcome

  Pass verdict

  In Step 2, the IUT invokes the GetAllCapabilitiesInd_CB callback function with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Stream_handler = $SH_{LowerTester}$

  In Step 4, the IUT sends the AVDTP_GET_ALL_CAPABILITIES_RSP response to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 10 (response accept)
  - Signal_Id = 0x0C (AVDTP_GET_ALL_CAPABILITIES)

- Service_Capabilities = Two valid capabilities including header compression
- The header compression capabilities:
  - Service_category = Header_Compression
  - LOSC = 1
  - BackCh = 1 (if back channel should be used)
  - Basic = 1 (if service available for basic service)
  - Recovery = 1 (if service available for recovery service)

In Step 5, the IUT sends the AVDT_Get_All_Capabilities_Rsp event to the Upper Tester with RSP.

4.4.5.3 Set Configuration Reject response for incorrect format of the ROHC Service capability – ACP

- Test Purpose

  Verify that the IUT (ACP) is able to issue a Set Configuration Reject response to the INT if the format of the ROHC Service capability is incorrect.

- Reference

  [3] 6.4, 6.8, 7.6, 8.8, 13.1

  [5] 6.4, 6.9, 7.6, 8.9, 13.1

- Initial Condition
  - Perform the preamble described in Section 4.2.1 to establish an L2CAP signaling channel.
  - The Lower Tester has not yet configured $SEID_{IUT}$.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/TRA/RHC/BI-01-C [Set Configuration Reject response for incorrect format of the ROHC Service capability – ACP]
AVDTP/SNK/ACP/TRA/RHC/BI-01-C [Set Configuration Reject response for incorrect format of the ROHC Service capability – ACP]

Table 4.78: Set Configuration Reject response for incorrect format of the ROHC Service capability – ACP test cases

- Test Procedure

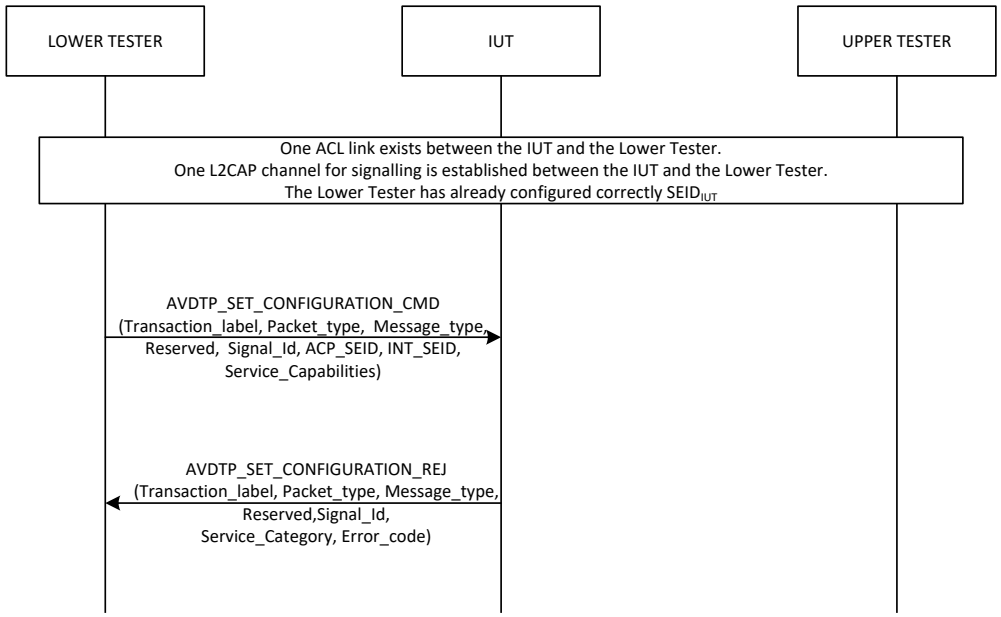

Figure 4.88: Set Configuration Reject response for incorrect format of the ROHC Service capability – ACP MSC

1. The Lower Tester sends an AVDTP_SET_CONFIGURATION_CMD command to the IUT with Transaction_label = $Transaction_{LowerTester}$ = any value, Packet_type = 00 (single packet), Reserved = 00, Message_type = 00 (command), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), ACP_SEID = $SEID_{IUT}$ = any value, INT_SEID = any valid value, and Service_capabilities = Service capabilities including an invalid format of the robust header compression service capability (LOSC equal to 0x00).
2. The IUT responds to the Lower Tester with an AVDTP_SET_CONFIGURATION_REJ response with Transaction_label = $Transaction_{LowerTester}$, Message_type = 11 (response reject), Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION), and Error_code = 0x26 (BAD_ROHC_FORMAT).

- Expected Outcome

  Pass verdict

  In Step 2, the IUT sends the AVDTP signaling message to the Lower Tester with the following parameters:

  - Transaction_label = $Transaction_{LowerTester}$
  - Message_type = 11 (response reject)
  - Signal_Id = 0x03 (AVDTP_SET_CONFIGURATION)
  - Error_code = 0x26 (BAD_ROHC_FORMAT)

4.5 L2CAP configuration

Verify the correct implementation of enhanced L2CAP channel modes for the signaling and Media Transport channels.

4.5.1 Backward compatibility with L2CAP Basic mode

Verify that AVDTP implementations remain backward compatible with earlier (legacy) devices.

While L2CAP Enhanced Retransmission and Streaming modes may be used to provide reliable Signaling and unreliable Media channels over unreliable connections, IUT must still be able to successfully connect to devices that support only L2CAP Basic mode connections for AVDTP.

4.5.1.1 Remote device only supports L2CAP Basic mode – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) as ACP will establish a signaling connection to a remote device that only supports L2CAP Basic mode.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is Idle with no L2CAP connections to the remote device.
  - The remote device is a device with L2CAP Basic mode support only.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/L2C/BM/BV-01-C [Remote device only supports L2CAP Basic mode – ACP]
AVDTP/SNK/ACP/L2C/BM/BV-01-C [Remote device only supports L2CAP Basic mode – ACP]

Table 4.79: Remote device only supports L2CAP Basic mode – ACP test cases

- Test Procedure

  1. The remote device initiates L2CAP (PSM=AVDTP) connection to IUT.
  2. The remote device configures L2CAP channel for Basic mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Signaling channel is successfully established.

4.5.1.2 Remote device only supports L2CAP Basic mode – INT

- Test Purpose

  Verify that the IUT (either SRC or SNK), as INT will establish a signaling connection to a remote device that only supports L2CAP Basic mode.

- Reference

  [5] 9.11

- Initial Condition
  - The Lower Tester is the remote device.
  - The IUT is Idle with no L2CAP connections to the remote device.
  - The remote device is a device with L2CAP Basic mode support only.
- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/BM/BV-02-C [Remote device only supports L2CAP Basic mode – INT]
AVDTP/SNK/INT/L2C/BM/BV-02-C [Remote device only supports L2CAP Basic mode – INT]

Table 4.80: Remote device only supports L2CAP Basic mode – INT test cases

- Test Procedure
  1. The IUT initiates L2CAP (PSM=AVDTP) connection to the remote device.
  2. The remote device configures L2CAP channel for Basic mode operation.
- Expected Outcome

  Pass verdict

  AVDTP Signaling channel is successfully established.

4.5.1.3 Remote device supports Retransmission mode but requires Basic mode L2CAP – INT

- Test Purpose

  Verify that the IUT (either SRC or SNK) as INT will establish a signaling connection to a remote device that supports Retransmission mode but requires a Basic mode L2CAP channel for AVDTP Signaling.
- Reference

  [5] 9.11
- Initial Condition
  - The Lower Tester is the remote device.
  - The IUT is Idle with no L2CAP connections to remote device.
  - The remote device supports L2CAP Enhanced Retransmission mode.
- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/BM/BV-03-C [Remote device supports Retransmission mode but requires Basic mode L2CAP – INT]
AVDTP/SNK/INT/L2C/BM/BV-03-C [Remote device supports Retransmission mode but requires Basic mode L2CAP – INT]

Table 4.81: Remote device supports Retransmission mode but requires Basic mode L2CAP – INT test cases

- Test Procedure

  1. The IUT initiates L2CAP (PSM=AVDTP) connection to the remote device.
  2. The remote device configures L2CAP channel for Basic mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Signaling channel is successfully established.

4.5.1.4 Media Transport channel when the remote device only supports Basic mode L2CAP – ACP

- Test Purpose

  Verify that the IUT (either SRC or SNK) as ACP is able to accept a Media Transport channel from a remote device that supports only a Basic mode L2CAP channel.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is connected to the remote device with an AVDTP Signaling L2CAP channel.
  - The SEP has been configured such as multiplexing is not selected, reporting is not selected and recovery is not selected, see Section 4.3.1.5, Set Configuration command – INT.
  - The remote device supports only L2CAP Basic mode.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/L2C/BM/BV-04-C [Media Transport channel when the remote device only supports Basic mode L2CAP – ACP]
AVDTP/SNK/ACP/L2C/BM/BV-04-C [Media Transport channel when the remote device only supports Basic mode L2CAP – ACP]

Table 4.82: Media Transport channel when the remote device only supports Basic mode L2CAP – ACP test cases

- Test Procedure

  1. The Lower Tester is the remote device.
  2. The remote device initiates the AVDTP_OPEN_CMD on the signaling channel to the remote device.
  3. The IUT responds with AVDTP_OPEN_RSP on signaling channel.
  4. The remote device initiates L2CAP Channel Creation (PSM=AVDTP).
  5. The remote device configures L2CAP channel for Basic mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Media Transport channel is successfully established.

4.5.1.5 Media Transport when remote device only supports Basic mode L2CAP – INT

- Test Purpose

  Verify that the IUT (SRC or SNK) as INT is able to establish a Media Transport connection to a remote device that supports only a Basic mode L2CAP channel.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is connected to the remote device with an AVDTP Signaling L2CAP channel.
  - The SEP has been configured such as multiplexing is not selected, reporting is not selected and recovery is not selected, see Section 4.3.1.5, Set Configuration command – INT.
  - The remote device supports only L2CAP Basic mode.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/BM/BV-05-C [Media Transport when remote device only supports Basic mode L2CAP – INT]
AVDTP/SNK/INT/L2C/BM/BV-05-C [Media Transport when remote device only supports Basic mode L2CAP – INT]

Table 4.83: Media Transport when remote device only supports Basic mode L2CAP – INT test cases

- Test Procedure

  1. The IUT initiates the AVDTP_OPEN_CMD on the signaling channel to the remote device.
  2. The remote device responds with AVDTP_OPEN_RSP on signaling channel.
  3. The IUT initiates L2CAP Channel Creation (PSM=AVDTP).
  4. The IUT Configures L2CAP channel for Basic mode operation.

- Expected Outcome

  Pass verdict

  AVDTP streaming channel is successfully established.

4.5.1.6 Media Transport when remote device supports Streaming but requires Basic mode L2CAP – INT

- Test Purpose

  Verify that the IUT (SRC or SNK) as INT is able to establish a Media Transport channel to a remote device that supports L2CAP Streaming mode but requires a Basic mode L2CAP channel.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is connected to the remote device with an AVDTP Signaling L2CAP channel.

- The SEP has been configured such as multiplexing is not selected, reporting is not selected and recovery is not selected (see Section 4.3.1.5, Set Configuration command – INT).
- The remote device supports L2CAP Streaming mode.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/BM/BV-06-C [Media Transport when remote device supports Streaming but requires Basic mode L2CAP – INT]
AVDTP/SNK/INT/L2C/BM/BV-06-C [Media Transport when remote device supports Streaming but requires Basic mode L2CAP – INT]

Table 4.84: Media Transport when remote device supports Streaming but requires Basic mode L2CAP – INT test cases

- Test Procedure

1. The IUT initiates the AVDTP_OPEN_CMD on the signaling channel to remote device.
2. The remote device responds with AVDTP_OPEN_RSP on signaling channel.
3. The IUT initiates L2CAP Channel Creation (PSM=AVDTP).
4. The remote device configures L2CAP channel for Basic mode operation.

- Expected Outcome

Pass verdict

AVDTP Media Transport channel is successfully established.

4.5.2 L2CAP Enhanced modes

Verify that an AVDTP implementation is able to successfully use L2CAP Enhanced Retransmission and Streaming mode channels.

The tests in this section only apply to IUTs that support and prefer the L2CAP Enhanced Retransmission mode for the signaling channel and L2CAP Streaming mode for the Media Transport channel.

4.5.2.1 Enhanced Retransmission mode for Signaling connection – INT

- Test Purpose

Verify that when both the IUT (as INT) and the remote device support Enhanced Retransmission mode for the Signaling connection, the IUT (SRC or SNK) will establish a Signaling connection to a remote device that requires the use of FCS to ensure channel reliability.

- Reference

[5] 9.11

- Initial Condition

- The Lower Tester is the remote device.
- The IUT is Idle with no L2CAP connections to the remote device.
- The remote device supports L2CAP Enhanced Retransmission mode.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/EM/BV-01-C [Enhanced Retransmission mode for Signaling connection – INT]
AVDTP/SNK/INT/L2C/EM/BV-01-C [Enhanced Retransmission mode for Signaling connection – INT]

Table 4.85: Enhanced Retransmission mode for Signaling connection – INT test cases

- Test Procedure

  1. The IUT initiates L2CAP (PSM=AVDTP) connection to remote device.
  2. The remote device configures L2CAP channel for Enhanced Retransmission mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Signaling channel is successfully established.

  AVDTP Signaling channel mode is Enhanced Retransmission mode with FCS enabled.

4.5.2.2 Enhanced Retransmission mode for Signaling connection – ACP

- Test Purpose

  Verify that when both the IUT (as ACP) and remote device prefer Enhanced Retransmission mode for Signaling connection, the IUT (SRC or SNK) will establish a Signaling connection to a remote device set to L2CAP Enhanced Retransmission mode with FCS.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is Idle with no L2CAP connections to remote device.
  - The remote device is a device with L2CAP Enhanced Retransmission mode preferred.

- Test Case Configuration

Test Case
AVDTP/SRC/ACP/L2C/EM/BV-02-C [Enhanced Retransmission mode for Signaling connection – ACP]
AVDTP/SNK/ACP/L2C/EM/BV-02-C [Enhanced Retransmission mode for Signaling connection – ACP]

Table 4.86: Enhanced Retransmission mode for Signaling connection – ACP test cases

- Test Procedure

  1. The remote device initiates L2CAP (PSM=AVDTP) connection to IUT.
  2. The remote device configures L2CAP channel for Enhanced Retransmission mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Signaling channel is successfully established in Enhanced Retransmission with FCS mode.

4.5.2.3 Media Transport channel with Streaming mode – INT

- Test Purpose

  Verify that when both IUT (as INT) and the remote device prefer L2CAP Streaming mode for Media Transport channel, the IUT (SRC or SNK) is able to establish a Media Transport channel set to L2CAP Streaming mode operation.

- Reference

  [5] 9.11

- Initial Condition

  - The Lower Tester is the remote device.
  - The IUT is connected to remote device with an AVDTP Signaling L2CAP channel.
  - The SEP has been configured such as multiplexing is not selected, reporting is not selected and recovery is not selected, see Section 4.3.1.5, Set Configuration command – INT.
  - The remote device is a device with L2CAP Streaming mode preferred for the Media Transport channel.

- Test Case Configuration

Test Case
AVDTP/SRC/INT/L2C/EM/BV-03-C [Media Transport channel with Streaming mode – INT]
AVDTP/SNK/INT/L2C/EM/BV-03-C [Media Transport channel with Streaming mode – INT]

Table 4.87: Media Transport channel with Streaming mode – INT test cases

- Test Procedure

  1. The IUT initiates the AVDTP_OPEN_CMD on the signaling channel to the remote device.
  2. The remote device responds with AVDTP_OPEN_RSP on signaling channel.
  3. The IUT initiates L2CAP Channel Creation (PSM=AVDTP).
  4. The IUT configures L2CAP channel for Streaming mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Media Transport channel is successfully established with L2CAP Streaming mode support.

4.5.2.4 Media Transport channel with Streaming mode – ACP

- Test Purpose

  Verify that when both IUT (as ACP) and remote device prefer L2CAP Streaming mode for Media Transport channel, the IUT (SRC or SNK) will establish a Media Transport channel set to L2CAP Streaming mode operation.

- Reference

  [5] 9.11

- Initial Condition
  - The Lower Tester is the remote device.
  - The IUT is connected to remote device with an AVDTP Signaling L2CAP channel.
  - The SEP has been configured such as multiplexing is not selected, reporting is not selected and recovery is not selected, see Section 4.3.1.5, Set Configuration command – INT.
  - The remote device is a device with L2CAP Streaming mode preferred for the Media Transport channel.
- Test Case Configuration

Test Case
AVDTP/SRC/ACP/L2C/EM/BV-04-C [Media Transport channel with Streaming mode – ACP]
AVDTP/SNK/ACP/L2C/EM/BV-04-C [Media Transport channel with Streaming mode – ACP]

Table 4.88: Media Transport channel with Streaming mode – ACP test cases

- Test Procedure

1. The remote device initiates the AVDTP_OPEN_CMD on the signaling channel to the IUT.
2. The IUT responds with AVDTP_OPEN_RSP on signaling channel.
3. The remote device initiates L2CAP Channel Creation (PSM=AVDTP).
4. The remote device configures L2CAP channel for Streaming mode operation.

- Expected Outcome

  Pass verdict

  AVDTP Media Transport channel is successfully established with L2CAP Streaming mode support.

5 Test case mapping

The Test Case Mapping Table (TCMT) maps test cases to specific requirements in the ICS. The IUT is tested in all roles for which support is declared in the ICS document.

The columns for the TCMT are defined as follows:

Item: Contains a logical expression based on specific entries from the associated ICS document. Contains a logical expression (using the operators AND, OR, NOT as needed) based on specific entries from the applicable ICS document(s). The entries are in the form of y/x references, where y corresponds to the table number and x corresponds to the feature number as defined in the ICS document for Audio/Video Distribution Transport Protocol (AVDTP) [4].

If a test case is mandatory within the respective layer, then the y/x reference is omitted.

Feature: A brief, informal description of the feature being tested.

Test Case(s): The applicable test case identifiers are required for Bluetooth Qualification if the corresponding y/x references defined in the Item column are supported. Further details about the function of the TCMT are elaborated in [6].

For the purpose and structure of the ICS/IXIT, refer to [6].

Item	Feature	Test Case(s)
AVDTP 4/1 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Stream Discover command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-05-C
AVDTP 4b/1 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Stream Discover command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-05-C
AVDTP 10/1 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Stream Discover Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-06-C
AVDTP 10b/1 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Stream Discover Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-06-C
AVDTP 4/2 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Get Capabilities Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-07-C
AVDTP 4b/2 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Get Capabilities Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-07-C

Item	Feature	Test Case(s)
AVDTP 10/2 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Get Capabilities Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-08-C
AVDTP 10b/2 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Get Capabilities Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-08-C
AVDTP 4/3 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Set Configuration Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-09-C
AVDTP 4b/3 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Set Configuration Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-09-C
AVDTP 10/3 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Set Configuration Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-10-C
AVDTP 10b/3 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Set Configuration Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-10-C
AVDTP 4/4 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Get Configuration Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-11-C
AVDTP 4b/4 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Get Configuration Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-11-C
AVDTP 10/4 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Get Configuration Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-12-C
AVDTP 10b/4 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Get Configuration Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-12-C
AVDTP 4/5 AND AVDTP 5/4	Reconfigure Command following Suspend – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-13-C

Item	Feature	Test Case(s)
AVDTP 4b/5 AND AVDTP 5b/4	Reconfigure Command following Suspend– INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-13-C
AVDTP 4/5	Reconfigure Command following Open – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-33-C
AVDTP 4b/5	Reconfigure Command following Open – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-33-C
AVDTP 10/5 AND AVDTP 11/4	Reconfigure Response following Suspend – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-14-C
AVDTP 10b/5 AND AVDTP 11b/4	Reconfigure Response following Suspend – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-14-C
AVDTP 10/5	Reconfigure Response following Open – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-34-C
AVDTP 10b/5	Reconfigure Response following Open – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-34-C
AVDTP 5/1 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Open Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-15-C
AVDTP 5b/1 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Open Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-15-C
AVDTP 11/1 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Open Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-16-C
AVDTP 11b/1 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Open Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-16-C
AVDTP 5/2 AND AVDTP 1/1 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Start a Streaming procedure – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-17-C
AVDTP 11/2 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Receive Start Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-18-C
AVDTP 11b/2 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Receive Start Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-18-C

Item	Feature	Test Case(s)
AVDTP 5/3 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Close Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-19-C
AVDTP 5b/3 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Close Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-19-C
AVDTP 11/3 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Close Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-20-C
AVDTP 11b/3 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Close Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-20-C
AVDTP 5/4 AND AVDTP 1/1 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Suspend Command – INT	AVDTP/SRC/INT/SIG/SMG/BV-21-C
AVDTP 11/4 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Suspend Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-22-C
AVDTP 11b/4 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Suspend Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-22-C
AVDTP 5/5 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Abort Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-23-C
AVDTP 5b/5 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Abort Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-23-C
AVDTP 11/5 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Accept Abort – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-24-C

Item	Feature	Test Case(s)
AVDTP 11b/5 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Accept Abort – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-24-C
AVDTP 4/6 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Get All Capabilities Command – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-25-C
AVDTP 4b/6 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Get All Capabilities Command – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-25-C
AVDTP 10/6 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Get All Capabilities Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BV-26-C
AVDTP 10b/6 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Get All Capabilities Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BV-26-C
AVDTP 4/6 AND (AVDTP 2/1 OR AVDTP 2/2 OR AVDTP 2/3 OR AVDTP 2/4)	Get All Capabilities Response with RFD values – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-28-C
AVDTP 4b/6 AND (AVDTP 2b/1 OR AVDTP 2b/2 OR AVDTP 2b/3 OR AVDTP 2b/4)	Get All Capabilities Response with RFD values – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-28-C
AVDTP 4/1 AND AVDTP 17/1	Detect Stream Discover Command Reject by ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-01-C
AVDTP 4b/1 AND AVDTP 17/1	Detect Stream Discover Command Reject by ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-01-C
AVDTP 1/1 AND AVDTP 10/1	Stream Discover Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-02-C
AVDTP 1/2 AND AVDTP 10b/1	Stream Discover Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-02-C
AVDTP 10/1 AND AVDTP 17/1	Stream Discover Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-03-C
AVDTP 10b/1 AND AVDTP 17/1	Stream Discover Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-03-C

Item	Feature	Test Case(s)
AVDTP 4/2 AND AVDTP 17/1	Detect Get Capabilities Command Reject by ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-04-C
AVDTP 4b/2 AND AVDTP 17/1	Detect Get Capabilities Command Reject by ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-04-C
AVDTP 10/2	Invalid Get Capabilities Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-05-C
AVDTP 10b/2	Invalid Get Capabilities Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-05-C
AVDTP 10/2 AND AVDTP 17/1	Get Capabilities Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-06-C
AVDTP 10b/2 AND AVDTP 17/1	Get Capabilities Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-06-C
AVDTP 4/3 AND AVDTP 17/1	Detect Set Configuration Command Reject by ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-07-C
AVDTP 4b/3 AND AVDTP 17/1	Detect Set Configuration Command Reject by ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-07-C
AVDTP 10/3	Set Configuration Reject for an Already Configured SEP – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-08-C
AVDTP 10b/3	Set Configuration Reject for an Already Configured SEP – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-08-C
AVDTP 10/3 AND AVDTP 17/1	Set Configuration Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-09-C
AVDTP 10b/3 AND AVDTP 17/1	Set Configuration Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-09-C
AVDTP 4/4 AND AVDTP 17/1	Detect Rejection of a Get Configuration Command by the ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-10-C
AVDTP 4b/4 AND AVDTP 17/1	Detect Rejection of a Get Configuration Command by the ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-10-C
AVDTP 10/4	Invalid Get Configuration Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-11-C
AVDTP 10b/4	Invalid Get Configuration Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-11-C
AVDTP 10/4 AND AVDTP 17/1	Get Configuration Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-12-C
AVDTP 10b/4 AND AVDTP 17/1	Get Configuration Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-12-C
AVDTP 4/5 AND AVDTP 5/4 AND AVDTP 17/1	Detect Rejection of the Reconfigure Command by the ACP following Suspend – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-13-C

Item	Feature	Test Case(s)
AVDTP 4b/5 AND AVDTP 5b/4 AND AVDTP 17/1	Detect Rejection of the Reconfigure Command by the ACP following Suspend – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-13-C
AVDTP 4/5 AND AVDTP 17/1	Detect the Rejection of the Reconfigure Command by the ACP following OPEN – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-43-C
AVDTP 4b/5 AND AVDTP 17/1	Detect the Rejection of the Reconfigure Command by the ACP folowing OPEN – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-43-C
AVDTP 10/5	Reconfigure Reject Response for Invalid Requests – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-14-C
AVDTP 10b/5	Reconfigure Reject Response for Invalid Requests – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-14-C
AVDTP 10/5 AND AVDTP 11/4 AND AVDTP 17/1	Reject Reconfigure Request following Suspend – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-15-C
AVDTP 10b/5 AND AVDTP 11b/4 AND AVDTP 17/1	Reject Reconfigure Request following Suspend – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-15-C
AVDTP 10/5 AND AVDTP 17/1	Reject Reconfigure Request following OPEN – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-45-C
AVDTP 10b/5 AND AVDTP 17/1	Reject Reconfigure Request following OPEN – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-45-C
AVDTP 5/1 AND AVDTP 17/1	Detect the Rejection of an Open Command by ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-16-C
AVDTP 5b/1 AND AVDTP 17/1	Detect the Rejection of an Open Command by ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-16-C
AVDTP 11/1	Invalid Open Stream Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-17-C
AVDTP 11b/1	Invalid Open Stream Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-17-C
AVDTP 11/1 AND AVDTP 17/1	Reject Open Stream Command Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-18-C
AVDTP 11b/1 AND AVDTP 17/1	Reject Open Stream Command Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-18-C
AVDTP 5/2 AND AVDTP 17/1	Detect the rejection of the Start Stream Command by the ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-19-C
AVDTP 11/2	Start Stream Reject Response when not in Open state – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-20-C
AVDTP 11b/2	Start Stream Reject Response when not in Open state – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-20-C
AVDTP 11/2 AND AVDTP 17/1	Reject Start Stream Command Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-21-C
AVDTP 11b/2 AND AVDTP 17/1	Reject Start Stream Command Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-21-C

Item	Feature	Test Case(s)
AVDTP 5/3 AND AVDTP 17/1	Detect the Rejection of the Close Stream Command by the ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-22-C
AVDTP 5b/3 AND AVDTP 17/1	Detect the Rejection of the Close Stream Command by the ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-22-C
AVDTP 11/3	Close Stream Reject Response to Invalid SEP – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-23-C
AVDTP 11b/3	Close Stream Reject Response to Invalid SEP – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-23-C
AVDTP 11/3 AND AVDTP 17/1	Reject Close Stream Command Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-24-C
AVDTP 11b/3 AND AVDTP 17/1	Reject Close Stream Command Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-24-C
AVDTP 5/4 AND AVDTP 17/1	Detect Rejection of a Suspend Stream Command by the ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-25-C
AVDTP 11/4	Reject Suspend Stream Response when the Stream is not in a correct state – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-26-C
AVDTP 11b/4	Reject Suspend Stream Response when the Stream is not in a correct state – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-26-C
AVDTP 11/4 AND AVDTP 17/1	Reject Suspend Stream Command Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-27-C
AVDTP 11b/4 AND AVDTP 17/1	Reject Suspend Stream Command Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-27-C
AVDTP 11/6 AND NOT AVDTP 16/3	General Reject Response – Does not include signal ID, ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-28-C
AVDTP 11b/6 AND NOT AVDTP 16/3	General Reject Response – Does not include signal ID, ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-28-C
AVDTP 11/6 AND AVDTP 16/3	General Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-38-C
AVDTP 11b/6 AND AVDTP 16/3	General Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-38-C
AVDTP 11/4 AND AVDTP 17/1 AND (AVDTP 8/1 OR AVDTP 8/2 OR AVDTP 8/3 OR AVDTP 8/4)	Refuse Further Streaming following Suspend – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-29-C
AVDTP 11b/4 AND AVDTP 17/1 AND (AVDTP 8b/1 OR AVDTP 8b/2 OR AVDTP 8b/3 OR AVDTP 8b/4)	Refuse Further Streaming following Suspend – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-29-C

Item	Feature	Test Case(s)
AVDTP 1/3 AND AVDTP 4/2 AND AVDTP 4/3	Ignore Reserved Capabilities in GetCapabilities Response – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-30-C
AVDTP 1/3 AND AVDTP 4b/2 AND AVDTP 4b/3	Ignore Reserved Capabilities in GetCapabilities Response – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-30-C
AVDTP 1/3 AND AVDTP 4/2 AND AVDTP 4/3	Set Configuration Command following a Get Capabilities Response with All Defined Capabilities – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-31-C
AVDTP 1/3 AND AVDTP 4b/2 AND AVDTP 4b/3	Set Configuration Command following a Get Capabilities Response with All Defined Capabilities – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-31-C
AVDTP 4/6 AND AVDTP 17/1	Detect Rejection of a Get All Capabilities Command by the ACP – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-32-C
AVDTP 4b/6 AND AVDTP 17/1	Detect Rejection of a Get All Capabilities Command by the ACP – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-32-C
AVDTP 10/6	Invalid Get All Capabilities Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-33-C
AVDTP 10b/6	Invalid Get All Capabilities Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-33-C
AVDTP 10/6 AND AVDTP 17/1	Get All Capabilities Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SMG/BI-34-C
AVDTP 10b/6 AND AVDTP 17/1	Get All Capabilities Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SMG/BI-34-C
AVDTP 1/3 AND AVDTP 4/3 AND AVDTP 4/6	Accept all Defined Capabilities in a GetAllCapabilities Response – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BV-35-C
AVDTP 1/3 AND AVDTP 4b/3 AND AVDTP 4b/6	Accept all Defined Capabilities in a GetAllCapabilities Response – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BV-35-C
AVDTP 1/3 AND AVDTP 4/3 AND AVDTP 4/6	Ignore Reserved Capabilities in the GetAllCapabilities Response – INT, SRC	AVDTP/SRC/INT/SIG/SMG/BI-36-C
AVDTP 1/3 AND AVDTP 4b/3 AND AVDTP 4b/6	Ignore Reserved Capabilities in the GetAllCapabilities Response – INT, SNK	AVDTP/SNK/INT/SIG/SMG/BI-36-C
AVDTP 6/1	Content Security Control Command – INT, SRC	AVDTP/SRC/INT/SIG/SEC/BV-01-C
AVDTP 6b/1	Content Security Control Command – INT, SNK	AVDTP/SNK/INT/SIG/SEC/BV-01-C
AVDTP 12/1	Content Security Control Command – ACP, SRC	AVDTP/SRC/ACP/SIG/SEC/BV-02-C
AVDTP 12b/1	Content Security Control Command – ACP, SNK	AVDTP/SNK/ACP/SIG/SEC/BV-02-C

Item	Feature	Test Case(s)
AVDTP 12/1	Set Configuration Reject Response – ACP, SRC	AVDTP/SRC/ACP/SIG/SEC/BI-01-C
AVDTP 12b/1	Set Configuration Reject Response – ACP, SNK	AVDTP/SNK/ACP/SIG/SEC/BI-01-C
AVDTP 13/1 AND AVDTP 17/1	Fragmentation of Signaling Messages – ACP, SRC	AVDTP/SRC/ACP/SIG/FRA/BV-01-C
AVDTP 13b/1 AND AVDTP 17/1	Fragmentation of Signaling Messages – ACP,SNK	AVDTP/SNK/ACP/SIG/FRA/BV-01-C
AVDTP 7/1 AND AVDTP 17/1	Reassembling of Signaling Messages – INT, SRC	AVDTP/SRC/INT/SIG/FRA/BV-02-C
AVDTP 7b/1 AND AVDTP 17/1	Reassembling of Signaling Messages – INT, SNK	AVDTP/SNK/INT/SIG/FRA/BV-02-C
AVDTP 14/1 AND AVDTP 5/2	Send Streaming Media Data – INT, SRC	AVDTP/SRC/INT/TRA/BTR/BV-01-C
AVDTP 15/1	SNK Receives Streaming Media Data – ACP, SNK	AVDTP/SNK/ACP/TRA/BTR/BV-02-C
AVDTP 14/1	Set Configuration Reject Response following Incorrect Media Transport Format – ACP, SRC	AVDTP/SRC/ACP/TRA/BTR/BI-01-C
AVDTP 15/1	Set Configuration Reject Response following Incorrect Media Transport Format – ACP, SNK	AVDTP/SNK/ACP/TRA/BTR/BI-01-C
AVDTP 14/2	Reporting Service – ACP, SRC	AVDTP/SRC/ACP/TRA/REP/BV-01-C
AVDTP 15/2	Reporting Service – ACP, SNK	AVDTP/SNK/ACP/TRA/REP/BV-01-C
AVDTP 14/2	Reporting Channel Association – INT, SRC	AVDTP/SRC/INT/TRA/REP/BV-02-C
AVDTP 15/2	Reporting Channel Association – INT, SNK	AVDTP/SNK/INT/TRA/REP/BV-02-C
AVDTP 14/2 AND AVDTP 16/1	Set Configuration Reject Response with Bad Length or Bad Payload Format – ACP, SRC	AVDTP/SRC/ACP/TRA/REP/BI-02-C
AVDTP 15/2 AND AVDTP 16/1	Set Configuration Reject Response with Bad Length or Bad Payload Format – ACP, SNK	AVDTP/SNK/ACP/TRA/REP/BI-02-C
AVDTP 14/3	Recovery Service – ACP, SRC	AVDTP/SRC/ACP/TRA/REC/BV-01-C
AVDTP 15/3	Recovery Service – ACP, SNK	AVDTP/SNK/ACP/TRA/REC/BV-01-C
AVDTP 14/3 AND AVDTP 10/6	Recovery Service when using GetAllCapabilities – ACP, SRC	AVDTP/SRC/ACP/TRA/REC/BV-02-C
AVDTP 15/3 AND AVDTP 10b/6	Recovery Service when using GetAllCapabilities – ACP, SNK	AVDTP/SNK/ACP/TRA/REC/BV-02-C
AVDTP 14/3	Set Configuration Reject Response with Bad Recovery Format – ACP, SRC	AVDTP/SRC/ACP/TRA/REC/BI-01-C
AVDTP 15/3	Set Configuration Reject Response with Bad Recovery Format – ACP, SNK	AVDTP/SNK/ACP/TRA/REC/BI-01-C

Item	Feature	Test Case(s)
AVDTP 14/4	Multiplexing Service – INT, SRC	AVDTP/SRC/INT/TRA/MUX/BV-01-C
AVDTP 15/4	SNK is able to Demultiplex Transport Sessions – ACP, SNK	AVDTP/SNK/ACP/TRA/MUX/BV-02-C
AVDTP 14/4	SRC Fragments a Transport Packet – INT, SRC	AVDTP/SRC/INT/TRA/MUX/BV-03-C
AVDTP 15/4	SNK Reconstructs a Transport Packet – ACP, SNK	AVDTP/SNK/ACP/TRA/MUX/BV-04-C
AVDTP 14/4	GetCapabilities including Multiplexing Service Capabilities – ACP, SRC	AVDTP/SRC/ACP/TRA/MUX/BV-05-C
AVDTP 15/4	GetCapabilities including Multiplexing Service Capabilities – ACP, SNK	AVDTP/SNK/ACP/TRA/MUX/BV-05-C
AVDTP 14/4 AND AVDTP 10/6	Get All Capabilities including Multiplexing Service Capabilities – ACP, SRC	AVDTP/SRC/ACP/TRA/MUX/BV-06-C
AVDTP 15/4 AND AVDTP 10b/6	Get All Capabilities including Multiplexing Service Capabilities – ACP, SNK	AVDTP/SNK/ACP/TRA/MUX/BV-06-C
AVDTP 14/4	Set Configuration Reject Response for Incorrect Format of Multiplexing Service Capability – ACP, SRC	AVDTP/SRC/ACP/TRA/MUX/BI-01-C
AVDTP 15/4	Set Configuration Reject Response for Incorrect Format of Multiplexing Service Capability – ACP, SNK	AVDTP/SNK/ACP/TRA/MUX/BI-01-C
AVDTP 14/5	Robust Header Compression Service – ACP, SRC	AVDTP/SRC/ACP/TRA/RHC/BV-01-C
AVDTP 15/5	Robust Header Compression Service – ACP, SNK	AVDTP/SNK/ACP/TRA/RHC/BV-01-C
AVDTP 14/5 AND AVDTP 10/6	GetAllCapabilities with Robust Header Compression Service – ACP, SRC	AVDTP/SRC/ACP/TRA/RHC/BV-02-C
AVDTP 15/5 AND AVDTP 10b/6	GetAllCapabilities with Robust Header Compression Service – ACP, SNK	AVDTP/SNK/ACP/TRA/RHC/BV-02-C
AVDTP 14/5	Set Configuration Reject Response for Incorrect Format of the ROHC Service Capability – ACP, SRC	AVDTP/SRC/ACP/TRA/RHC/BI-01-C
AVDTP 15/5	Set Configuration Reject Response for Incorrect Format of the ROHC Service Capability – ACP, SNK	AVDTP/SNK/ACP/TRA/RHC/BI-01-C
AVDTP 15/6	Synchronization Capability offered – ACP, SNK	AVDTP/SNK/ACP/SIG/SYN/BV-01-C
AVDTP 15/6 AND AVDTP 4b/3	Sync Capability configuration – INT, SNK	AVDTP/SNK/INT/SIG/SYN/BV-02-C

Item	Feature	Test Case(s)
AVDTP 15/6	Initial delay report (SNK is ACP) – ACP, SNK	AVDTP/SNK/ACP/SIG/SYN/BV-03-C
AVDTP 15/6 AND AVDTP 4b/3	Initial delay report (SNK is INT) – INT, SNK	AVDTP/SNK/INT/SIG/SYN/BV-04-C
AVDTP 14/6	Delay Report from Open state – INT, SRC	AVDTP/SRC/INT/SIG/SYN/BV-05-C
AVDTP 14/6	Delay Report from Streaming state – ACP, SRC	AVDTP/SRC/ACP/SIG/SYN/BV-06-C
AVDTP 1/1 AND AVDTP 1/4 AND AVDTP 18/1	Remote device only supports L2CAP Basic mode – ACP, SRC	AVDTP/SRC/ACP/L2C/BM/BV-01-C
AVDTP 1/2 AND AVDTP 1/4 AND AVDTP 18/1	Remote device only supports L2CAP Basic mode – ACP, SNK	AVDTP/SNK/ACP/L2C/BM/BV-01-C
AVDTP 1/1 AND AVDTP 1/3 AND AVDTP 18/1	Remote device and L2CAP Basic mode – INT, SRC	AVDTP/SRC/INT/L2C/BM/BV-02-C AVDTP/SRC/INT/L2C/BM/BV-03-C
AVDTP 1/2 AND AVDTP 1/3 AND AVDTP 18/1	Remote device and L2CAP Basic mode – INT, SNK	AVDTP/SNK/INT/L2C/BM/BV-02-C AVDTP/SNK/INT/L2C/BM/BV-03-C
AVDTP 1/1 AND AVDTP 1/4 AND AVDTP 18/2	Media Transport Channel when the remote device only supports Basic mode L2CAP – ACP, SRC	AVDTP/SRC/ACP/L2C/BM/BV-04-C
AVDTP 1/2 AND AVDTP 1/4 AND AVDTP 18/2	Media Transport Channel when the remote device only supports Basic mode L2CAP – ACP, SNK	AVDTP/SNK/ACP/L2C/BM/BV-04-C
AVDTP 1/1 AND AVDTP 1/3 AND AVDTP 18/2	Media Transport – remote device and L2CAP Basic mode – INT, SRC	AVDTP/SRC/INT/L2C/BM/BV-05-C AVDTP/SRC/INT/L2C/BM/BV-06-C
AVDTP 1/2 AND AVDTP 1/3 AND AVDTP 18/2	Media Transport – remote device and L2CAP Basic mode – INT, SNK	AVDTP/SNK/INT/L2C/BM/BV-05-C AVDTP/SNK/INT/L2C/BM/BV-06-C
AVDTP 1/1 AND AVDTP 1/3 AND AVDTP 18/1	Enhanced Retransmission mode for Signaling Connection – INT, SRC	AVDTP/SRC/INT/L2C/EM/BV-01-C
AVDTP 1/2 AND AVDTP 1/3 AND AVDTP 18/1	Enhanced Retransmission mode for Signaling Connection – INT, SNK	AVDTP/SNK/INT/L2C/EM/BV-01-C
AVDTP 1/1 AND AVDTP 1/4 AND AVDTP 18/1	Enhanced Retransmission mode for Signaling Connection – ACP, SRC	AVDTP/SRC/ACP/L2C/EM/BV-02-C
AVDTP 1/2 AND AVDTP 1/4 AND AVDTP 18/1	Enhanced Retransmission mode for Signaling Connection – ACP, SNK	AVDTP/SNK/ACP/L2C/EM/BV-02-C
AVDTP 1/1 AND AVDTP 1/3 AND AVDTP 18/2	Media Transport Channel with Streaming mode – INT, SRC	AVDTP/SRC/INT/L2C/EM/BV-03-C
AVDTP 1/2 AND AVDTP 1/3 AND AVDTP 18/2	Media Transport Channel with Streaming mode – INT, SNK	AVDTP/SNK/INT/L2C/EM/BV-03-C

Item	Feature	Test Case(s)
AVDTP 1/1 AND AVDTP 1/4 AND AVDTP 18/2	Media Transport Channel with Streaming mode – ACP, SRC	AVDTP/SRC/ACP/L2C/EM/BV-04-C
AVDTP 1/2 AND AVDTP 1/4 AND AVDTP 18/2	Media Transport Channel with Streaming mode – ACP, SNK	AVDTP/SNK/ACP/L2C/EM/BV-04-C

Table 5.1: Test case mapping

6 Revision history and acknowledgments

Revision History

Publication Number	Revision Number	Date	Comments
	1.00	2003/02/05	Release for Voting Draft
0	Version 1.0	2003/05/09	Update title and header
	Version 1.2 R00	2004/04/14	Updated Disclaimer and Copyright Notice. Clerical changes.
1	Version 1.2 R01	2004/04/19	Corrected internal cross reference hyperlinks
	1.2.2r1	2005/02/18	Editorial and format changes. Changed document numbering. Incorporated TSE 468 for TP/SIG/SMG/BI-02-C. Incorporated TSE 537 for TP/SIG/SMG/BI-01-C, TP/SIG/SMG/BI-04-C, TP/TRA/BTR/BI-01-C, TP/TRA/REC/BI-01-C, TP/TRA/RHC/BI-01-C, TP/SIG/SEC/BI/BI-01-C, TP/TRA/MUX/BI-01-C, TP/SIG/SMG/BI-07-C, TPTP/SIG/SMG/BI-10-C, TP/TRA/REP/BI-01-C, TP/SIG/SMG/BI-13-C, and TP/SIG/SMG/BI-26-C. Incorporated TSE 538 for MSCs. Incorporated TSE 539 for TCMT. Incorporated TSE 585 for TP/SIG/SMG/BI-28-C. Incorporated TSE 650 for TP/SIG/SMG/BI-01-C, TP/SIG/SMG/BI-04-C, TP/SIG/SMG/BI-06-C, TP/SIG/SMG/BI-07-C, TP/SIG/SMG/BI-09-C, TP/SIG/SMG/BI-10-C, TP/SIG/SMG/BI-12-C, TP/SIG/SMG/BI-13-C, TP/SIG/SMG/BI-15-C, TP/SIG/SMG/BI-16-C, TP/SIG/SMG/BI-18-C, TP/SIG/SMG/BI-19-C, TP/SIG/SMG/BI-21-C, TP/SIG/SMG/BI-22-C, TP/SIG/SMG/BI-27-C, TP/SIG/SMG/BI-25-C and TP/SIG/SMG/BI-27-C. Incorporated TSE 662 for Table 2 and TP/TRA/REP/BI-01-C. Incorporated TSE 696 for TP/SIG/SMG/BI-20-C. Incorporated TSE 718 for TP/SIG/SMG/BI-26-C Incorporated TSE 728 for TP/SIG/FRA/BV-01-C. Incorporated TSE 729 for TP/SIG/SMG/BI-06-C. Incorporated TSE 737 for TP/SIG/SMG/BV-07-C, TP/SIG/SMG/BV-10-C, TP/SIG/SMG/BV-11-C.
2	1.2.2	2005/03/01	Prepare for publication.
3	1.2.3	2005/03/02	Corrected cross-reference errors.
	1.2.4r1	2005/03/11	Changed the way TSE 662 was incorporated for ESR02 (SE 502) Errata Service Release to Specification Versions 1.1, 1.2, and Profiles which changed Table 2 and added new test case TP/TRA/REP/ESR02/BI-01-C. The TCMT was also changed to add entries for TP/TRA/REP/BI-01-C and TP/TRA/REP/ESR02/BI-01-C.

Publication Number	Revision Number	Date	Comments
4	1.2.4	2005/04/25	Prepare for publication
5	1.2.5	2005/08/31	Incorporate TSE: 749 for TP/SIG/SMG/BI-02-C, 763-:Removal of TP/SIG/SMG/BV{-01-04}-C 764TP/SIG/SMG/BV-{11,15,17, 19,21,22,23}-C TP/SIG/SMG/BI-{01,03,06,09,10,12,15,16,18}-C 765 TP/SIG/SMG/BI-29-C based on TP/SIG/SMG/BV-22-C with new test purpose 766 TP/SIG/SMG/BV-{11,15,17,19,21,23}-C TP/SIG/SMG/BI-{10,16,19,22,25}-C 767 TP/SIG/SMG/BI- {03,06,09,12,15,18,21,24,25,27}-C 768 TP/SIG/SEC/BV-01-C,TP/SIG/SEC/BV-02-C TP/SIG/FRA/BV-01-C,TP/SIG/FRA/BV-02-C 769 TP/SIG/SEC/BV-01-C TP/TRA/BTR/BV-01-C,TP/TRA/BTR/BV-02-C TP/TRA/REP/BV-01-C,TP/TRA/REP/BV-02-C TP/TRA/REC/BV-01-C,TP/TRA/MUX/BV-02-C TP/TRA/MUX/BV-03-C,TP/TRA/MUX/BV-04-C TP/TRA/MUX/BV-05-C,TP/TRA/RHC/BV-01-C 770 TP/SIG/SMG/BV-17-C, TP/SIG/SMG/BV-21-C TP/TRA/BTR/BV-01-C,TP/TRA/BTR/BV-02-C TP/TRA/MUX/BV-01-C,TP/TRA/MUX/BV-03-C 772 TP/SIG/SMG/BI-03-C,TP/SIG/SMG/BI-06-C TP/SIG/SMG/BI-09-C,TP/SIG/SMG/BI-12-C TP/SIG/SMG/BI-15-C,TP/SIG/SMG/BI-18-C TP/SIG/SMG/BI-21-C,TP/SIG/SMG/BI-27-C 773 Appendix A title 774 change Lower_Tester to LowerTester 775 Not implemented 776 Change to TCMT and ICS for 10-1 AND 17-1 777 Change to TCMT and ICS for 5-4 AND 17-1 778 TP/SIG/SMG/BI-20-C 779, change to TCMT and ICS for 7-1 AND 17-1 781 TP/SIG/SMG/BV-05-C, TP/SIG/SMG/BV-06-C 782 Added response msgs to MSCs: TP/SIG/SEC/BV-02-C, TP/SIG/FRA/BV-01-C TP/SIG/FRA/BV-02-C, TP/TRA/BTR/BV-01-C TP/TRA/BTR/BV-02-C, TP/TRA/REP/BV-01-C TP/TRA/REC/BV-01-C, TP/TRA/MUX/BV-05-C TP/TRA/RHC/BV-01-C 785 Edit service capabilities for TP/SIG/SMG/BV-{10|11|12}-C TSE 829 TP/TRA/REC-BV-01; TSE 830 for TP/SIG/SMG/BI-14-C; reformatting of TCMT table/TSE 764.

Publication Number	Revision Number	Date	Comments
6	1.2.6	2006/06/12	TSE 835 for TP/SIG/SMG/BV-18-C and TP/SIG/SMG/BI-29-C TSE 841 for TP/TRA/BTR/BV-01-C, TP/TRA/BTR/BV-02-C 919; Change TMCT items for TP/SIG/SMG/BV-17-C, TP/SIG/SMG/BV-21-C. 933: Change TP/SIG/SMG/BV-18-C
7	1.2.7	2006/01/08	Add Conformance Section TSE 1881: TP/SIG/SMG/BI-14-C: Change initial condition in test case and MSC. TSE 1882: TP/SIG/SMG/BI-20-C Fix MSC TSE 1911 TP/SIG/SMG/BV-07-C: Fix MSC References, Abbreviations and Definitions gathered under section 2. Removed “Uncertainties” updated TOC
8	1.2.8	2007/08/24	TSE 2070: TCMT: Add “AND 17/1” for TP/SIG/SMG/BI-01-C, TP/SIG/ SMG/BI-04-C, TP/SIG/BI-07-C, TP/SIG/ SMG/BI-10-C, TP/SIG/ SMG/BI-13-C, TP/SIG/ SMG/BI-16-C, TP/SIG SMG//BI-19-C, TP/SIG/ SMG/BI-22-C, TSE 2170: Change TCMT: TP/TRA/REP/BI-01-C, TP/TRA/REP/BV-01-C
9	1.2.9	2008/04/21	TSE 2166 TCMT for TP/SIG/SMG/BI-29-C TSE 2210: TP/TRA/REC/BV-01-C, TP/TRA/REP/ESR02/BI-01-C TSE 2309: TP/SIG/SMG/BV-17-C, TP/SIG/SMG/BV-18-C
	1.2.10r0	2008/07/23	TSE 2522: TP/SIG/SMG/BI-28-C: Pass verdict
10	1.2.10	2008/12/01	Prepare for publication.
	1.2.10	2009/12/01	Release for Synchronization Voting Draft
	1.2.11r0	2010/01/25	TSE 2676: New test case TP/SIG/SMG/BI-30-C TSE 2677: New test case TP/SIG/SMG/BI-31-C TSE 2834: TP/SIG/SMG/BI-28-C: rewritten TSE 2834: TP/SIG/SMG/ESR04/BI-28-C: add test case TSE 3042: TP/SIG/SMG/ESR02/BI-02-C: remove TC TSE 3048: TP/SIG/SMG/BI-14-C: Pass verdict TSE 3114: TP/SIG/SMG/BI-28-C: Fix MSC TSE 3888: TP/TRA/BTR/BI-01-C: Correct MCS
	1.2.11r1	2011/01/27	Input reviewer comments to TSE 2676: TP/SIG/SMG/BI-30-C: add 2nd Fail verdict; TSE 2834: TP/SIG/SMG/BI-28-C: removed _SUSPEND_ to match MSC label

Publication Number	Revision Number	Date	Comments
	1.2.11r2	2011/03/18	TSE 4234. New test cases: TP/SIG/SMG/ESR05/BV-13-C, TP/SIG/SMG/ESR05/BV-14-C, TP/SIG/SMG/ESR05/BI-13-C, TP/SIG/SMG/ESR05/BI-15-C
11	1.2.11	2011/07/21	Prepare for publication.
	1.3.0r0	2011/02/04	Prepare for publication
	1.3.0r1	2011/03/06	Update after AV F2F
	1.3.0r2	2011/07/08	Missing issues from AV F2F update
	1.3.0r3	2011/11/11	Merge to incorporate 2.1 + EDR Core Spec updates
	1.3.0r4	2012/01/09	Added changes from comments received from BTI initial review
	1.3.0r5	2012/04/17	Miscellaneous editorial updates Corrected Signal_Id value in TP/SIG/SMG/BV-25 Updated Test Condition and Expected Outcome of TP/SIG/SMG/BV-27-C Updated Test Condition and Expected Outcome of TP/SIG/SMG/BV-28-C
	1.3.0r6	2012/04/20	Updated input parameters for messages associated with TP/TRA/RHC/BV-01-C and TP/TRA/RHC/BV-02-C
	1.3.0r7	2012/06/05	Fixed the TCMT for TP/SIG/SMG/BI-35-C and TP/SIG/SMG/BI-35-C for support of get all capabilities command and consistently applied GetAllCapabilities in the test case test where it was incorrectly stated as GetCapabilities. Fixed the references for TP/SIG/SMG/BI-30-C and TP/SIG/SMG/BI-31-C. Per BTI review comments.
	1.3.0r8	2012/06/18	TSE 4882: Change TP/SIG/SMG/BI-31-C TC ID to become TP/SIG/SMG/BV-31-C with updated purpose and TP/SIG/SMG/BI-31-C will be phased out. Revised TCMT for new TC ID.
	1.3.0r9	2012/06/21	TSE 4885: Changed relevant test cases to remove non-fragmented AVDTP message condition and corresponding pass verdict. Affected test cases: TP/SIG/SMG/BV-05-C to TP/SIG/SMG/BV-28-C inclusive TP/SIG/SMG/BI-01-C to TP/SIG/SMG/BI-34-C inclusive TP/SIG/SEC/BV-01-C TP/SIG/SEC/BI-01-C TP/TRA/BTR/BV-01-C to TP/TRA/BTR/BV-02-C inclusive TP/TRA/BTR/BI-01-C TP/TRA/REP/BV-01-C to TP/TRA/REP/BV-02-C inclusive TP/TRA/REP/BI-01-C to TP/TRA/REP/ESR02/BI-01-C TP/TRA/REC/BI-01-C

Publication Number	Revision Number	Date	Comments
			TP/TRA/MUX/BV-01-C to TP/TRA/MUX/BV-06-C TP/TRA/MUX/BI-01-C TP/TRA/RHC/BI-01-C TP/L2C/BM/BV-04-C to TP/L2C/BM/BV-06-C inclusive TP/L2C/EM/BV-03-C to TP/L2C/EM/BV-04-C TSE 4886: Changed TP/TRA/MUX/BV-01-C and TP/TRA/MUX/BV-02-C to properly capture conditions of non-fragmentation of AL-PDUs.
	1.3.0r10	2012/06/24	Fixed Rev History for TSE 4885, added versions to front page, updated Conformance section for current text, and corrected accidental deletion in the initial condition of TP/SIG/SEC/BV-01-C.
	1.3.0r11	2012/07/15	Reordered subsection numbering for Section 2 to comply with published Test Specification template. Adjusted fail verdict and notes section formatting of test cases to comply with latest recommendations.
12	1.3.0	2012/07/24	Prepare for publication.
	1.3.1r00	2014/10/21	TSE 5961: Updated TCMT mapping for TP/SIG/SYN/BV-02-C and TP/SIG/SYN/BV-04-C.
	1.3.1r01	2014/11/23	BTI Review, Nerissa, Corrected TSE 5961 implementation.
13	1.3.1	2014/12/05	Prepare for TCRL 2014-2 publication
	1.3.2r00	2015/04/24	TSE 6344: Updated TCMT mapping for TP/SIG/SMG/BI-02-C TSE 6015: Updated TCMT & parameter in Pass verdict for TP/TRA/BTR/BV-01-C
	1.3.2r01	2015/06/03	Universal removal of “Fail verdict” statements equal to “Any of the pass verdicts fail to occur.” Addition of Fail verdict convention in section 3.2.5 Removal of Test Conditions that consist solely of “N/A”
	1.3.2r02	2015/06/15	Editorial revisions from BTI review. Revisions/Updates to Table 4.2 Error Code Mapping.
14	1.3.2	2015/07/14	Prepared for TCRL 2015-1 publication
	1.3.3r00	2015/10/01	TSE 6590: Corrected IUT and Tester roles for TP/SIG/SMG/BI-25-C. TSE 6530: Removed test TP/SIG/SMG/BV-27-C. TSE 6540: Added title to TP/SIG/SMG/BI-28-C and corrected signal ID value in Test Condition. TSE 6552: Corrected LOSC values in TP/TRA/MUX/BV-05-C and TP/TRA/MUX/BV-06-C and added full TSID/TCID information.
15	1.3.3	2015/12/22	Prepared for TCRL 2015-2 publication
	1.3.4r00	2016/02/29	TSE 6696: Removed the discover requirement from test condition of test case TP/SIG/SMG/BV-31-C.

Publication Number	Revision Number	Date	Comments
	1.3.4r01	2016/03/30	TSE 6654: Updated heading title, Reference, Initial Condition, Test Procedure, Test Condition, and Pass verdict for test cases TP/SIG/SMG/BV-17-C, TP/SIG/SMG/BV-18-C, TP/SIG/SMG/BI-29-C, TP/TRA/BTR/BV-01-C, TP/TRA/BTR/BV-02-C.
	1.3.4r02	2016/05/19	Test Spec Template Conversion. Added test case names for those tests that did not already have them. Changed all remaining instances of “Test System” to “Lower Tester” to comply with current TSTO conventions. Section consolidation in section 2. Removal of section 3.2 (found to be redundant) Test Purposes for TP/L2C/BM/BV-01-C, to -06-C rewritten to provide clarity of the respective purposes
16	1.3.4	2016/07/13	Prepared for TCRL 2016-1 publication.
	1.3.5r00	2017/03/28	TSE 8908: Updated to current template. Test Case ID conversion. Updated Test Case Mapping for the ICS changes in TSE 8372.
	1.3.5r01	2017/06/22	TSE 9483: Corrected Test Case Mapping for SIG/FRA/BV-01-C, SIG/FRA/BV-02-C, TRA/REC/BV-02-C, TRA/MUX/BV-06-C, TRA/RHC/BV-02-C. Updated Section 4.1 to account for new Test Case ID conventions.
17	1.3.5	2017/07/03	Approved by BTI. Prepared for TCRL 2017-1 publication.
	1.3.6r00–r02	2019-04-08 – 2019-06-25	TSE 11211 (rating 2): Updated to the latest template version and made editorial revisions. Updated TCMT references to Table 0 following ICS changes to split the version tables per role.
18	1.3.6	2019-07-28	Approved by BTI. Prepared for TCRL 2019-1 publication.
	1.3.6 edition 2r00	2020-01-22	TSE 13362 (rating 1): Rearranged Acronyms table to align acronyms with the correct definitions, and added an entry for SMG.
	1.3.6 edition 2r01	2020-02-12	Restored errantly deleted “TRA” row to acronyms table.
	1.3.6 edition 2r02	2020-05-12	Integration review feedback.
	1.3.6 edition 2	2020-06-01	Performed minor formatting and template updates, rolled back document numbering to reflect an edition release, and accepted all tracked changes. Approved by BTI on 2020-06-01. Prepared for edition 2 publication.
	p19r00–r03	2021-07-27 – 2021-12-07	TSE 16837 (rating 2): Updated the test purpose, initial condition, and MSC for the section containing TCs AVDTP/SRC/INT/SIG/SMG/BV-19-C and AVDTP/SNK/INT/SIG/SMG/BV-19-C. TSE 16916 (rating 2): Removed test cases AVDTP/SRC/ACP/TRA/REP/BI-01-C and

Publication Number	Revision Number	Date	Comments
			AVDTP/SNK/ACP/TRA/REP/BI-01-C. Fixed the arrow in the MSC of AVDTP/SRC and AVDTP/SNK/ACP/SIG/SMG/BI-38-C to be in the direction of the Lower Tester from the IUT. Renamed the following test cases: AVDTP/SRC/ACP/SIG/SMG/ESR04/BI-28-C to AVDTP/SRC/ACP/SIG/SMG/BI-38-C, AVDTP/SNK/ACP/SIG/SMG/ESR04/BI-28-C to AVDTP/SNK/ACP/SIG/SMG/BI-38-C, AVDTP/SRC/ACP/TRA/REP/ESR02/BI-01-C to AVDTP/SRC/ACP/TRA/REP/BI-02-C, AVDTP/SNK/ACP/TRA/REP/ESR02/BI-01-C to AVDTP/SNK/ACP/TRA/REP/BI-02-C, AVDTP/SNK/ACP/SIG/SMG/ESR05/BI-15-C to AVDTP/SNK/ACP/SIG/SMG/BI-45-C, AVDTP/SRC/ACP/SIG/SMG/ESR05/BI-15-C to AVDTP/SRC/ACP/SIG/SMG/BI-45-C, AVDTP/SNK/ACP/SIG/SMG/ESR05/BV-14-C to AVDTP/SNK/ACP/SIG/SMG/BV-34-C, AVDTP/SRC/ACP/SIG/SMG/ESR05/BV-14-C to AVDTP/SRC/ACP/SIG/SMG/BV-34-C, AVDTP/SNK/INT/SIG/SMG/ESR05/BI-13-C to AVDTP/SNK/INT/SIG/SMG/BI-43-C, AVDTP/SRC/INT/SIG/SMG/ESR05/BI-13-C to AVDTP/SRC/INT/SIG/SMG/BI-43-C, AVDTP/SNK/INT/SIG/SMG/ESR05/BV-13-C to AVDTP/SNK/INT/SIG/SMG/BV-33-C, and AVDTP/SRC/INT/SIG/SMG/ESR05/BV-13-C to AVDTP/SRC/INT/SIG/SMG/BV-33-C. Converted the following test cases into table driven tests with their previous ESR05 variant tests, added a test case configuration table with an SEP State, and referenced this table in the combined initial conditions and the initial conditions in the MSCs and updated the MSC captions: AVDTP/SNK/ACP/SIG/SMG/BI-15-C, AVDTP/SRC/ACP/SIG/SMG/BI-15-C, AVDTP/SNK/ACP/SIG/SMG/BV-14-C, AVDTP/SRC/ACP/SIG/SMG/BV-14-C, AVDTP/SNK/INT/SIG/SMG/BI-13-C, AVDTP/SRC/INT/SIG/SMG/BI-13-C, AVDTP/SNK/INT/SIG/SMG/BV-13-C, and AVDTP/SRC/INT/SIG/SMG/BV-13-C. Updated the TCMT to include AVDTP ICS for the Suspend feature for the following tests: AVDTP/SNK/ACP/SIG/SMG/BI-15-C, AVDTP/SRC/ACP/SIG/SMG/BI-15-C, AVDTP/SNK/ACP/SIG/SMG/BV-14-C, AVDTP/SRC/ACP/SIG/SMG/BV-14-C, AVDTP/SNK/INT/SIG/SMG/BI-13-C, AVDTP/SRC/INT/SIG/SMG/BI-13-C, AVDTP/SNK/INT/SIG/SMG/BV-13-C, and AVDTP/SRC/INT/SIG/SMG/BV-13-C. Removed unnecessary items in Signaling Message Format ICS tables from the TCMT of AVDTP/SRC/INT/SIG/SMG/BV-13-C, AVDTP/SNK/INT/SIG/SMG/BV-13-C,

Publication Number	Revision Number	Date	Comments
			AVDTP/SRC/ACP/SIG/SMG/BV-14-C, AVDTP/SNK/ACP/SIG/SMG/BV-14-C, and their ESR05 variants. Performed template-related formatting fixes. Updated Scope, Conformance, Pass/Fail verdict conventions, and the introduction text before the TCMT to align with the template. Updated copyright page to align with v2 of the DNMD.
19	p19	2022-01-25	Approved by BTI on 2021-12-19. Prepared for TCRL 2021-2 publication.
	p20r00	2022-09-01	TSE 20353 (rating 2): Corrected the pass verdict for AVDTP/SRC/ACP/SIG/SMG/BI-28-C and AVDTP/SNK/ACP/SIG/SMG/BI-28-C.
20	p20	2023-02-07	Approved by BTI on 2022-12-19. Prepared for TCRL 2022-2 publication.
	p20ed2r00	2023-02-13	TSE 22296 (rating 1): Added “NSEP” and “NSEP_det” to the abbreviations list. Editorials to align the document with the latest TS template.
	p20 edition 2	2023-03-14	Approved by BTI on 2023-03-13. Prepared for edition 2 publication.
	p21r00–r04	2023-08-30 – 2024-04-24	TSE 23947 (rating 2): Updated the TCMT entries for AVDTP/SRC/ACP/SIG/SMG/BI-02-C and AVDTP/SNK/ACP/SIG/SMG/BI-02-C. TSE 24605 (rating 2): Updated the TCMT items for AVDTP/SRC/ACP/SIG/SMG/BI-02-C and AVDTP/SNK/ACP/SIG/SMG/BI-02-C to address deprecated version. TSE 22578 (rating 1): Updated the Test Suite Structure (TSS) section to align the test environment text with the general test strategy section from TSTO. TSE 24485 (rating 4): Updated the initial condition, MSC, test procedure, and pass verdict and deleted the test condition for AVDTP/SRC/INT/TRA/REP/BV-02-C and AVDTP/SNK/INT/TRA/REP/BV-02-C. Updated the document to align with the latest standards.
21	p21	2024-07-01	Approved by BTI on 2024-05-22. Prepared for TCRL 2024-1 publication.
	p22r00–r02	2024-10-24 – 2024-11-04	TSE 24566 (rating 1): Added a setup preamble section. Updated all test cases with references to the preamble, active language, and test procedure step numbering.
22	p22	2025-02-18	Approved by BTI on 2024-12-25. Prepared for TCRL 2025-1 publication.

Publication Number	Revision Number	Date	Comments
	p23r00	2025-02-26	TSE 26591 (rating 2): Updated the MSC for test cases AVDTP/SRC/INT/SIG/SMG/BV-05-C and AVDTP/SNK/INT/SIG/SMG/BV-05-C. Updated the MSCs, test procedures, and expected outcomes for AVDTP/SRC/INT/SIG/SMG/BI-04-C, -07-C, -10-C, -13-C, -16-C, -32-C, and -43-C and AVDTP/SNK/INT/SIG/SMG/BI-04-C, -07-C, -10-C, -13-C, -16-C, -32-C, and -43-C. Renamed test case AVDTP/SRC/INT/SIG/SMG/BI-35-C to AVDTP/SRC/INT/SIG/SMG/BV-35-C and AVDTP/SNK/INT/SIG/SMG/BI-35-C to AVDTP/SNK/INT/SIG/SMG/BV-35-C, and updated the TCMT accordingly. Updated the test procedure for AVDTP/SNK/ACP/TRA/BTR/BV-02-C.
23	p23	2025-07-08	Approved by BTI on 2025-05-30. Prepared for TCRL pkg100 publication.

Acknowledgments

Name	Company
Dejan Berec	Bluetooth SIG, Inc.
Tharon Hall	Bluetooth SIG, Inc.
Alicia Courtney	Broadcom
Ash Kapur	Broadcom
Jiny Bradshaw	CSR
Allan Madsen	CSR
David Trainor	CSR
Akira Miyajima	Denso
Morgan Lindqvist	Ericsson
Fisseha Mekuria	Ericsson
Yuan Quinton	Marvell
Michinori Masuda	Matsushita Electric Industrial
Tsuyoshi Okada	Matsushita Electric Industrial
Thomas Karlsson	Mecel
Stephen Raxter	National Analysis Center
Janne Hamalainen	Nokia
Thierry Wœlfflé	Parrot
Christian Bouffioux	Philips
Henk Koopmans	Philips
Emmanuel Mellery	Philips
Piotr Polak	Philips
Scott Walsh	Plantronics
Brian Gix	Qualcomm
John Larkin	Qualcomm
Magnus Sommansson	Qualcomm

Name	Company
Wilhelm Hagg	Sony
Atsushi Ichise	Sony
Harumi Kawamura	Sony
Rüdiger Mosig	Sony
Hiroyasu Noguchi	Sony
Masahiko Seki	Sony
Siân James	Symbian
Ichiro Tomoda	Toshiba