



# Generic Audio/Video Distribution Profile (GAVD)

## Application Programming Interface Reference Manual

Profile Version:

Release: 5.1.0  
March 05, 2021



Bluetooth and the Bluetooth logos are trademarks owned by Bluetooth SIG, Inc., USA and licensed to Stonestreet One, LLC. Bluetopia®, Stonestreet One™, and the Stonestreet One logo are registered trademarks of Stonestreet One, LLC, Louisville, Kentucky, USA. All other trademarks are property of their respective owners.  
Copyright © 2000-2014 by Stonestreet One, LLC. All rights reserved.

## Table of Contents

<b>1. INTRODUCTION.....</b>	<b>4</b>
1.1 Scope .....	4
1.2 Applicable Documents .....	5
1.3 Acronyms and Abbreviations .....	6
<b>2. GAVD PROFILE PROGRAMMING INTERFACE .....</b>	<b>8</b>
2.1 <b>GAVD Profile Commands.....</b>	<b>8</b>
GAVD_Initialize.....	10
GAVD_Cleanup.....	11
GAVD_Connect_Request_Response.....	11
GAVD_Register_SDP_Record.....	12
GAVD_Connect.....	13
GAVD_Disconnect.....	14
GAVD_Discover_End_Points .....	14
GAVD_Get_End_Point_Capabilities .....	15
GAVD_Get_Configuration.....	16
GAVD_Register_End_Point.....	17
GAVD_Un_Register_End_Point.....	21
GAVD_Create_End_Point_Group .....	21
GAVD_Delete_End_Point_Group .....	23
GAVD_Connect_Remote_End_Point .....	23
GAVD_Open_Remote_End_Point.....	25
GAVD_Close_End_Point.....	29
GAVD_Set_Configuration_Response .....	30
GAVD_Start_Stream_Request .....	31
GAVD_Start_Stream_Response.....	32
GAVD_Suspend_Stream_Request .....	33
GAVD_Suspend_Stream_Response.....	34
GAVD_Reconfigure_Request .....	35
GAVD_Reconfigure_Response.....	37
GAVD_Security_Control_Request.....	38
GAVD_Security_Control_Response .....	39
GAVD_Abort_Stream_Request .....	40
GAVD_Data_Write .....	41
GAVD_Sender_Report_Data_Write .....	42
GAVD_Receiver_Report_Data_Write .....	44
GAVD_SDES_Report_Data_Write.....	45
GAVD_Recovery_Data_Write .....	46
GAVD_Get_Server_Connection_Mode .....	47
GAVD_Set_Server_Connection_Mode.....	48
GAVD_Register_Signalling_Connection_Status .....	48
GAVD_Un_Register_Signalling_Connection_Status .....	49
GAVD_Disconnect_Signalling_Connection .....	50
GAVD_Get_Data_Queueing_Parameters .....	51

GAVD_Set_Data_Queueing_Parameters .....	52
GAVD_Get_L2CAP_Channel_Info .....	53
GAVD_Get_End_Point_All_Capabilities .....	54
GAVD_Set_Delay_Report .....	55
GAVD_Set_Reject_Response_State: .....	55
<b>2.2 GAVD Profile Event Callback Prototypes.....</b>	<b>55</b>
GAVD_Event_Callback_t .....	56
<b>2.3 GAVD Profile Events.....</b>	<b>58</b>
etGAVD_Connect_Request_Indication .....	60
etGAVD_Connect_Confirmation .....	61
etGAVD_Disconnect_Indication .....	61
etGAVD_Discover_Confirmation .....	61
etGAVD_Get_Capabilities_Confirmation.....	63
etGAVD_Get_Configuration_Confirmation .....	64
etGAVD_Set_Configuration_Indication .....	66
etGAVD_Open_End_Point_Indication .....	66
etGAVD_Open_End_Point_Confirmation .....	67
etGAVD_Close_End_Point_Indication .....	68
etGAVD_Start_Indication .....	69
etGAVD_Start_Confirmation .....	69
etGAVD_Suspend_Indication .....	70
etGAVD_Suspend_Confirmation .....	70
etGAVD_Reconfigure_Indication .....	71
etGAVD_Reconfigure_Confirmation.....	72
etGAVD_Security_Control_Indication .....	73
etGAVD_Security_Control_Confirmation .....	73
etGAVD_Abort_Indication.....	74
etGAVD_Abort_Confirmation .....	74
etGAVD_Data_Indication .....	74
etGAVD_Sender_Report_Data_Indication .....	75
etGAVD_Receiver_Report_Data_Indication .....	76
etGAVD_SDES_Report_Data_Indication.....	77
etGAVD_Recovery_Data_Indication .....	78
etGAVD_Data_Channel_Empty_Indication .....	78
etGAVD_Report_Data_Channel_Empty_Indication.....	79
etGAVD_Recovery_Data_Channel_Empty_Indication .....	79
etGAVD_Multiplexed_Channel_Empty_Indication .....	79
etGAVD_Signalling_Connect_Indication .....	80
etGAVD_Signalling_Disconnect_Indication.....	80
etGAVD_Signalling_Channel_Idle_Indication.....	81
etGAVD_Signalling_Channel_Endpoint_Open_Indication.....	81
etGAVD_Signalling_Channel_Endpoint_Close_Indication.....	82
etGAVD_Delay_Report_Indication.....	83
etGAVD_Delay_Report_Confirmation .....	83
etGAVD_General_Reject_Indication .....	84
<b>3. FILE DISTRIBUTIONS.....</b>	<b>85</b>

# 1. Introduction

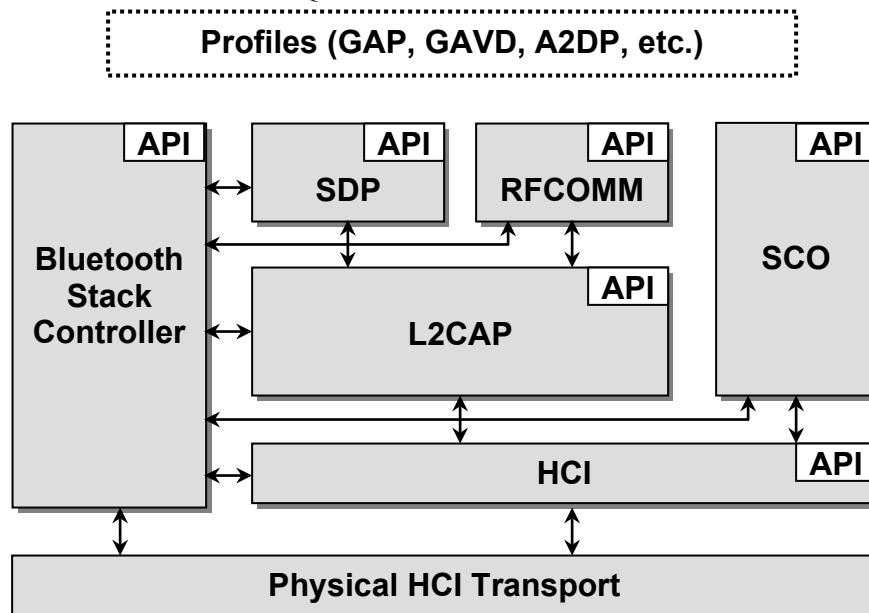
Bluetopia®, the Bluetooth Protocol Stack by Stonestreet One, provides a software architecture that encapsulates the upper functionality of the Bluetooth Protocol Stack. More specifically, this stack is a software solution that resides above the Physical HCI (Host Controller Interface) Transport Layer and extends through the L2CAP (Logical Link Control and Adaptation Protocol) and the SCO (Synchronous Connection-Oriented) Link layers. In addition to basic functionality at these layers, the Bluetooth Protocol Stack by Stonestreet One provides implementations of the Service Discovery Protocol (SDP), RFCOMM (the Radio Frequency serial COMMunications port emulator), and several of the Bluetooth Profiles. Program access to these layers, services, and profiles is handled via Application Programming Interface (API) calls.

This document focuses on the API reference that contains a description of all programming interfaces for the Bluetooth Generic Audio/Video Distribution Profile provided by Bluetopia. Chapter 2 contains a description of the programming interfaces for this profile. Chapter 3 contains the header file name list for the Bluetooth GAVD Profile library.

## 1.1 Scope

This reference manual provides information on the GAVD Profile API. This API is available on the full range of platforms supported by Stonestreet One:

- Windows
- Windows Mobile
- Windows CE
- Linux
- QNX
- Other Embedded OS



**Figure 1-1 The Stonestreet One Bluetooth Protocol Stack**

## 1.2 Applicable Documents

The following documents may be used for additional background and technical depth regarding the Bluetooth technology.

1. *Specification of the Bluetooth System, Volume 1, Core*, version 1.1, February 22, 2001.
2. *Specification of the Bluetooth System, Volume 2, Profiles*, version 1.1, February 22, 2001.
3. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 2.0 + EDR, November 4, 2004.
4. *Specification of the Bluetooth System, Volume 2, Core System Package*, version 2.0 + EDR, November 4, 2004.
5. *Specification of the Bluetooth System, Volume 3, Core System Package*, version 2.0 + EDR, November 4, 2004.
6. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 2.1+EDR, July 26, 2007.
7. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 2.1+EDR, July 26, 2007.
8. *Specification of the Bluetooth System, Volume 2, Core System Package [Controller Volume]*, version 2.1+EDR, July 26, 2007.
9. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 2.1+EDR, July 26, 2007.
10. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 2.1+EDR, July 26, 2007.
11. *Specification of the Bluetooth System, Bluetooth Core Specification Addendum 1*, June 26, 2008.
12. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 3.0+HS, April 21, 2009.
13. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 3.0+HS, April 21, 2009.
14. *Specification of the Bluetooth System, Volume 2, Core System Package [Controller Volume]*, version 3.0+HS, April 21, 2009.
15. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 3.0+HS, April 21, 2009.
16. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 3.0+HS, April 21, 2009.
17. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 3.0+HS, April 21, 2009.
18. *Specification of the Bluetooth System, Volume 0, Master Table of Contents & Compliance Requirements*, version 4.0, June 30, 2010.

19. *Specification of the Bluetooth System, Volume 1, Architecture and Terminology Overview*, version 4.0, June 30, 2010.
20. *Specification of the Bluetooth System, Volume 2, Core System Package [BR/EDR Controller Volume]*, version 4.0, June 30, 2010.
21. *Specification of the Bluetooth System, Volume 3, Core System Package [Host Volume]*, version 4.0, June 30, 2010.
22. *Specification of the Bluetooth System, Volume 4, Host Controller Interface [Transport Layer]*, version 4.0, June 30, 2010.
23. *Specification of the Bluetooth System, Volume 5, Core System Package [AMP Controller Volume]*, version 4.0, June 30, 2010.
24. *Specification of the Bluetooth System, Volume 6, Core System Package [Low Energy Controller Volume]*, version 4.0, June 30, 2010.
25. *Bluetooth Assigned Numbers*, version 1.1, February 22, 2001.
26. *Digital cellular telecommunications system (Phase 2+); Terminal Equipment to Mobile Station (TE-MS) multiplexer protocol (GSM 07.10)*, version 7.1.0, Release 1998; commonly referred to as: ETSI TS 07.10.
27. *Audio/Video Distribution Transport Protocol Specification*, version 1.0, May 22, 2003.
28. *Generic Audio/Video Distribution Profile*, version 1.0, May 22, 2002.
29. *Bluetopia® Protocol Stack, Application Programming Interface Reference Manual*, version 4.0.1, January 10, 2013.
30. *Audio/Video Distribution Transport Protocol Specification*, version 1.3, July 24, 2012.

Possible error returns are listed for each API function call. These are the *most likely* errors, but in fact programmers should allow for the possibility of any error listed in the BTerrors.h header file to occur as the value of a function return.

### 1.3 Acronyms and Abbreviations

Acronyms and abbreviations used in this document and other Bluetooth specifications are listed in the table below.

Term	Meaning
API	Application Programming Interface
AVDTP	Audio/Video Distribution Transport Protocol
BD_ADDR	Bluetooth Device Address
BR	Basic Rate
BT	Bluetooth
EDR	Enhanced Data Rate
GAVD	Generic Audio/Video Distribution

Term	Meaning
HS	High Speed
LE	Low Energy
LSB	Least Significant Bit
LSEID	Local Stream Endpoint Identifier
MSB	Most Significant Bit
RSEID	Remote Stream Endpoint Identifier
SDP	Service Discovery Protocol
SPP	Serial Port Protocol
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

## 2. GAVD Profile Programming Interface

The GAVD Profile programming interface defines the protocols and procedures to be used to implement audio/video distribution/transport capabilities. The GAVD Profile commands are listed in section 2.1, the event callback prototype is described in section 2.2, and the GAVD Profile events are itemized in section 2.3. The actual prototypes and constants outlined in this section can be found in the **GAVDAPI.H** header file in the Bluetopia distribution.

### 2.1 GAVD Profile Commands

The available GAVD Profile command functions are listed in the table below and are described in the text that follows.

Function	Description
GAVD_Initialize	Initializes the GAVD Profile and starts a local Stream Endpoint Manager.
GAVD_Cleanup	Removes the GAVD Profile from the system and shuts down the local Stream Endpoint Manager.
GAVD_Connect_Request_Response	Responds to individual request to connect to a local GAVD/AVDTP Server.
GAVD_Register_SDP_Record	Adds a generic GAVD Service Record to the SDP database.
GAVD_Un_Register_SDP_Record	Removes a generic GAVD Service Record from the SDP database.
GAVD_Create_End_Point_Group	Creates an Endpoint group.
GAVD_Delete_End_Point_Group	Deletes a previously created Endpoint group.
GAVD_Connect	Connects to a Stream Endpoint Manager on the specified remote device.
GAVD_Disconnect	Closes a connection to a Stream Endpoint Manager that was previously opened with the GAVD_Connect() function.
GAVD_Discover_End_Points	Sends a Discover request to a remote Stream Endpoint Manager.
GAVD_Get_End_Point_Capabilities	Sends a Get Capabilities request for a specified remote stream endpoint.
GAVD_Get_Configuration	Sends a Get Configuration request for a specified remote stream endpoint.
GAVD_Register_End_Point	Registers a local stream endpoint with a local Stream Endpoint Manager.



GAVD_Un_Register_End_Point	Unregisters a previously registered stream endpoint from a local Stream Endpoint Manager.
GAVD_Open_Remote_End_Point	Establishes a connection to a remote endpoint on a Remote Endpoint Stream Manager.
GAVD_Close_End_Point	Closes a connection to an endpoint that was previously opened via a call to GAVD_Register_End_Point() or GAVD_Open_Remote_End_Point().
GAVD_Set_Configuration_Response	Responds to a request from a remote GAVD device to set the configuration of a stream endpoint.
GAVD_Start_Stream_Request	Requests to start one or more streams on a remote GAVD device.
GAVD_Start_Stream_Response	Responds to a request from a remote GAVD device to start one or more streams.
GAVD_Suspend_Stream_Request	Suspends one or more streams on a remote GAVD device.
GAVD_Suspend_Stream_Response	Responds to a request from a remote GAVD device to suspend one or more streams.
GAVD_Reconfigure_Request	Requests the reconfiguration of an endpoint on a remote GAVD device.
GAVD_Reconfigure_Response	Responds to a request from a remote GAVD device to reconfigure a stream.
GAVD_Security_Control_Request	Requests the change of the security settings for a stream.
GAVD_Security_Control_Response	Responds to a request from a remote GAVD device to change the security settings for a stream.
GAVD_Abort_Stream_Request	Requests to abort one or more streams on a remote GAVD device.
GAVD_Data_Write	Sends Media data over a specified stream.
GAVD_Sender_Report_Data_Write	Sends Sender Report data over the specified stream.
GAVD_Receiver_Report_Data_Write	Sends Receiver Report data over the specified stream.
GAVD_SDES_Report_Data_Write	Sends SDES Report data over the specified stream.
GAVD_Recovery_Data_Write	Sends RTP FEC recovery data over a specified stream.
GAVD_Get_Server_Connection_Mode	Retrieves the current GAVD/AVDTP Server Connection Mode.

GAVD_Register_Signalling_Connection_Status	Register an event callback to be notified of various GAVD/AVDTP signaling channel operations.
GAVD_Un_Register_Signalling_Connection_Status	Un-registers a previously registered GAVD/AVDTP signaling channel event callback.
GAVD_Disconnect_Signalling_Connection	Force a disconnect of a specified GAVD/AVDTP signaling connection.
GAVD_Set_Server_Connection_Mode	Changes the current GAVD/AVDTP Server Connection Mode.
GAVD_Get_Data_Queueing_Parameters	Retrieves current GAVD/L2CAP data queueing mode
GAVD_Set_Data_Queueing_Parameters	Changes the current GAVD/L2CAP data queueing mode
GAVD_Get_L2CAP_Channel_Info	Querys L2CAP Information for a local stream end point.

## GAVD\_Initialize

This function initializes the GAVD Profile. This function must be called before any other GAVD Profile function can be called. This function can only be called once per Bluetooth Stack Instance.

### Prototype:

```
int BTPSAPI GAVD_Initialize(unsigned int BluetoothStackID)
```

### Parameters:

BluetoothStackID<sup>1</sup>      Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC\_Initialize().

### Return:

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

### Possible Events:

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Cleanup

This function is responsible for removing a GAVD Profile from the system and shutting down the local Stream Endpoint Manager.

### Note:

This function does NOT delete any SDP Service Record Handles (i.e., added via a call to the GAVD\_Register\_SDP\_Record() function).

### Prototype:

```
void BTPSAPI GAVD_Cleanup(unsigned int BluetoothStackID)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
-------------------------------	---------------------------------------------------------------------------------------------

### Return:

None.

### Possible Events:

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Connect\_Request\_Response

This function responds to an individual request to connect to a local GAVD/AVDTP Server.

### Prototype:

```
int BTPSAPI GAVD_Connect_Request_Response(unsigned int BluetoothStackID, BD_ADDR_t BD_ADDR, Boolean_t AcceptConnection)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
BD_ADDR	Address of the Bluetooth device that is attempting to connect to a local GAVD/AVDTP Server.
AcceptConnection	Boolean specifying if the pending connection request is to be accepted (TRUE).

### Return:

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED

BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

#### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

### GAVD\_Register\_SDP\_Record

This function adds a generic GAVD Service Record to the SDP database.

#### Notes:

1. The Service Record Handle that is returned from this function will remain in the SDP Record Database until it is deleted by calling the SDP\_Delete\_Service\_Record() function. A macro is provided to delete the Service Record from the SDP Database. This macro maps the GAVD\_Un\_Register\_SDP\_Record() to SDP\_Delete\_Service\_Record(), and is defined as follows:

```
GAVD_Un_Register_SDP_Record(__BluetoothStackID, __SDPRecordHandle)
(SDP_Delete_Service_Record(__BluetoothStackID, __SDPRecordHandle))
```

2. Any Protocol Information that is specified will be added in the protocol attribute after the default protocol list of L2CAP and AVDTP.
3. The Service Name is always added at Attribute ID 0x0100. A Language Base Attribute ID List is created that specifies that 0x0100 is UTF-8 Encoded, English Language.
4. At least one Service Class (UUID) must be specified in the SDP Service Record structure.
5. The ProtocolList and ProfileList members of the SDP Service Record structure are optional (specified as NULL). The Protocol List information must be a Data Element Sequence, and the information contained in this sequence is added AFTER the AVDTP and L2CAP Protocol information. The ProfileList must also be a Data Element Sequence, however, this information is added as-is (nothing is added other than this information).

#### Prototype:

```
int BTPSAPI GAVD_Register_SDP_Record(unsigned int BluetoothStackID,
    GAVD_SDP_Service_Record_t *SDPServiceRecord, char *ServiceName,
    DWord_t *SDPServiceRecordHandle)
```

#### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
SDPServiceRecord	Specifies additional SDP information to add to the record. This is defined by the following structure:

```
typedef struct
{
    unsigned int    NumberServiceClassUUID;
```

```

SDP_UUID_Entry_t *SDPUUIDEntries;
SDP_Data_Element_t *ProtocolList;
SDP_Data_Element_t *ProfileList;
} GAVD_SDP_Service_Record_t;

```

**ServiceName** Name to appear in the SDP Database for this service.

**SDPServiceRecordHandle** Returned handle to the SDP Database entry that may be used to remove the entry at a later time.

**Return:**

None.

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Connect**

This function is used to connect to a remote Stream Endpoint Manager on the specified remote device.

**Prototype:**

```

int BTPSAPI GAVD_Connect(unsigned int BluetoothStackID, BD_ADDR_t BD_ADDR,
    GAVD_Event_Callback_t GAVDEventCallback, unsigned long CallbackParameter)

```

**Parameters:**

**BluetoothStackID**<sup>1</sup> Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC\_Initialize().

**BD\_ADDR** Address of the Bluetooth device to connect with.

**GAVDEventCallback** Function that is called whenever events occur on this connection.

**CallbackParameter** A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each event callback.

**Return:**

Positive, non-zero if successful. If this function is successful, the return value will represent the GAVD Client ID that can be passed to all other functions that require it.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

**Possible Events:**

etGAVD\_Connect\_Confirmation

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Disconnect**

This function is used to close a connection to a remote Stream Endpoint Manager that was previously opened via the GAVD\_Connect() function. This function does NOT close the remote Stream Endpoint Manager, it ONLY closes this client's connection to the remote Stream Endpoint Manager IF and ONLY IF no stream endpoints were created. If stream endpoints were created, calling this routine will not close connection. All stream endpoints must be closed before calling this routine.

**Prototype:**

int BTPSAPI **GAVD\_Disconnect**(unsigned int BluetoothStackID, unsigned int GAVDID)

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDID	The GAVD Client ID. This is the value that was returned from the GAVD_Connect() function.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:****Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Discover\_End\_Points**

This function is used to send a Discover request to a remote Stream Endpoint Manager that was previously opened via the GAVD\_Connect() function.

**Prototype:**

```
int BTPSAPI GAVD_Discover_End_Points(unsigned int BluetoothStackID,  
    unsigned int GAVDID)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDID	The GAVD client ID. This is the value that was returned from the GAVD_Connect() function.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION  
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

```
etGAVD_Discover_Confirmation  
etGAVD_Disconnect_Indication
```

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Get\_End\_Point\_Capabilities**

This function is used to send a Get Capabilities request to a remote Stream Endpoint Manager to determine the supported capabilities of the specified remote stream endpoint.

**Prototype:**

```
int BTPSAPI GAVD_Get_End_Point_Capabilities(unsigned int BluetoothStackID, unsigned int  
    GAVDID, unsigned int RSEID)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDID	The GAVD client ID. This is the value that was returned from the GAVD_Connect() function.
RSEID	The remote endpoint ID. This value is one of the values returned in a Discover Confirmation event.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Get\_Capabilities\_Confirmation  
etGAVD\_Disconnect\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Get\_Configuration**

This function is used to send a Get Configuration request to a remote Stream Endpoint Manager to determine the current configuration of the specified remote stream endpoint.

**Prototype:**

int BTPSAPI **GAVD\_Get\_Configuration**(unsigned int BluetoothStackID,  
unsigned int GAVDID, unsigned int RSEID)

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDID	The GAVD client ID. This is the value that was returned from the GAVD_Connect() function.
RSEID	The remote endpoint ID. This value is one of the values that was returned in a Discover Confirmation event.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Get\_Configuration\_Confirmation



etGAVD\_Disconnect\_Indication

#### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Register\_End\_Point

This function is used to register a local stream endpoint with the local Stream Endpoint Manager. If this function is successful then an endpoint will be created that is ready to be discovered, configured, and connected to by remote AVDTP/GAVD clients.

#### Prototype:

```
int BTPSAPI GAVD_Register_End_Point(unsigned int BluetoothStackID,
    GAVD_Local_End_Point_Info_t *LocalEndPointInfo,
    GAVD_Event_Callback_t GAVDEventCallback, unsigned long CallbackParameter)
```

#### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LocalEndPointInfo	Data containing the configuration information for the local endpoint. This is defined by the following structure:

```
typedef struct
{
    GAVD_TSEP_t                TSEP;
    GAVD_Media_Type_t          MediaType;
    Word_t                     MediaInMTU;
    Word_t                     ReportingInMTU;
    Word_t                     RecoveryInMTU;
    unsigned int               NumberCapabilities;
    GAVD_Service_Capabilities_Info_t *CapabilitiesInfo;
} GAVD_Local_End_Point_Info_t;
```

where, TSEP defines whether the endpoint is a source or a sink using the following enumeration:

```
typedef enum
{
    tspSRC,
    tspSNK
} GAVD_TSEP_t;
```

MediaType identifies the media type as audio, video, or multimedia using the following enumeration:

```
typedef enum
{
```

```
mtAudio,  
mtVideo,  
mtMultimedia  
} GAVD_Media_Type_t;
```

MediaInMTU, ReportingInMTU, and RecoveryInMTU are the maximum allowable MTU sizes.

NumberCapabilities specifies how many service capability information elements are contained in the CapabilitiesInfo member.

CapabilitiesInfo defines the service capabilities that an endpoint supports using the following structure:

```
typedef struct  
{  
    GAVD_Service_Category_t ServiceCategory;  
    union  
    {  
        GAVD_Recovery_Info_Element_Data_t  
            GAVD_Recovery_Info_Element_Data;  
        GAVD_Content_Protection_Info_Element_Data_t  
            GAVD_Content_Protection_Info_Element_Data;  
        GAVD_Header_Compression_Info_Element_Data_t  
            GAVD_Header_Compression_Info_Element_Data;  
        GAVD_Multiplexing_Info_Element_Data_t  
            GAVD_Multiplexing_Info_Element_Data;  
        GAVD_Media_Codec_Info_Element_Data_t  
            GAVD_Media_Codec_Info_Element_Data;  
        GAVD_Raw_Info_Element_Data_t  
            GAVD_Raw_Info_Element_Data;  
    } InfoElement;  
} GAVD_Service_Capabilities_Info_t;
```

where, ServiceCategory is defined by the following enumeration:

```
typedef enum  
{  
    scNone,  
    scMediaTransport,  
    scReporting,  
    scRecovery,  
    scContentProtection,  
    scHeaderCompression,  
    scMultiplexing,  
    scMediaCodec,  
    scUnknown  
} GAVD_Service_Category_t;
```

GAVD\_Recovery\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    Byte_t    RecoveryType;
    Byte_t    MaxRecoveryWindowSize;
    Byte_t    MaxNumberMediaPackets;
} GAVD_Recovery_Info_Element_Data_t;
```

GAVD\_Content\_Protection\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    Word_t    ContentProtectionType;
    Byte_t    ContentProtectionTypeSpecificInfoLength;
    Byte_t    *ContentProtectionTypeSpecificInfo;
} GAVD_Content_Protection_Info_Element_Data_t;
```

GAVD\_Header\_Compression\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    Boolean_t MediaPacketHeaderCompression;
    Boolean_t RecoveryPacketHeaderCompression;
    Boolean_t BackChannelSupported;
} GAVD_Header_Compression_Info_Element_Data_t;
```

GAVD\_Multiplexing\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    unsigned int          MediaMuxLSEID;
    GAVD_Transport_Channel_Type_t MediaMuxChannel;
    Boolean_t             UseReportingChannel;
    unsigned int          ReportingMuxLSEID;
    GAVD_Transport_Channel_Type_t ReportingMuxChannel;
    Boolean_t             UseRecoveryChannel;
    unsigned int          RecoveryMuxLSEID;
    GAVD_Transport_Channel_Type_t RecoveryMuxChannel;
} GAVD_Multiplexing_Info_Element_Data_t;
```

where, MediaMuxChannel, ReportingMuxChannel, and RecoveryMuxChannel are defined by the following enumeration:

```
typedef enum
{
    trMedia,
    trReporting,
```

```

        trRecovery,
        trNone
    } GAVD_Transport_Channel_Type_t;

```

GAVD\_Media\_Codec\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    GAVD_Media_Type_t    MediaType;
    Byte_t               MediaCodecType;
    Byte_t               MediaCodecSpecificInfoLength;
    Byte_t               *MediaCodecSpecificInfo;
} GAVD_Media_Codec_Info_Element_Data_t;

```

where, MediaType is defined by the following enumeration:

```

typedef enum
{
    mtAudio,
    mtVideo,
    mtMultimedia
} GAVD_Media_Type_t;

```

GAVD\_Raw\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    Byte_t    RawDataLength;
    Byte_t    *RawData;
} GAVD_Raw_Info_Element_Data_t;

```

GAVDEventCallback	Function that is be called whenever events occur regarding the specified local endpoint.
CallbackParameter	A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each event callback.

### Return:

Positive, non-zero value if successful that is the LSEID that must be used to identify this Stream End Point in future calls.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

### Possible Events:

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## **GAVD\_Un\_Register\_End\_Point**

This function is used to unregister previously registered local stream endpoint from a local Stream Endpoint Manager. This endpoint was registered via the GAVD\_Register\_End\_Point() function.

### **Prototype:**

```
int BTPSAPI GAVD_Un_Register_End_Point(unsigned int BluetoothStackID,  
    unsigned int LSEID)
```

### **Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Endpoint ID of the local stream endpoint to unregister. This value was returned from a successful call to the GAVD_Register_End_Point() function.

### **Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION  
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

### **Possible Events:**

### **Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## **GAVD\_Create\_End\_Point\_Group**

This function is used to create an endpoint group within the local Stream Endpoint Manager. An endpoint group is used to give the illusion to remote devices that there is only a single endpoint present (the group LSEID), however, it is backed by all members in the group. This allows (the apparent) multiple connections to the same LSEID (from the individual remote devices), however, it is really tracked separately within the local Stream Endpoint Manager (with it's own LSEID).

### Notes:

All endpoints in the group must have been registered via the `GAVD_Register_End_Point()` function and all endpoints in the group **\*MUST\*** have identical capabilities.

Endpoint Groups can **ONLY** be created/deleted when there are no active GAVD/AVDTP signaling connections active.

**Prototype:**

```
int BTPSAPI GAVD_Create_End_Point_Group(unsigned int BluetoothStackID,  
    unsigned int GroupLSEID, unsigned int NumberLSEIDs, unsigned int *LSEIDList)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to <code>BSC_Initialize()</code> .
GroupLSEID	Endpoint identifier of the endpoint groups local stream endpoint. This value was returned from a successful call to the <code>GAVD_Register_End_Point()</code> function and this endpoint's capabilities (including SEID) will be the only endpoint shared with remote Stream Endpoint Managers (via the discovery process).
NumberLSEIDs	Specifies the number of additional endpoints that are to be included in the group (each individual endpoint will be specified in the LSEID list parameter).
LSEIDList	List of endpoints (not including the group endpoint) that are to be included in the group. The number of endpoints in the list is specified by the number of LSEIDs parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INSUFFICIENT_RESOURCES  
BTGAVD_ERROR_GROUP_LSEID_LIST_INVALID  
BTGAVD_ERROR_GROUP_LSEID_INVALID  
BTGAVD_ERROR_SIGNALLING_CHANNEL_CONNECTED  
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:****Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Delete\_End\_Point\_Group

This function is used to un-group a previously grouped set of endpoints. This function does not delete each individual endpoint from the local Stream Endpoint Manager, it only instructs the local Stream Endpoint Manager to no longer treat the endpoints in the specified group as belonging to an endpoint group.

### Notes:

Endpoint Groups can **ONLY** be created/deleted when there are no active GAVD/AVDTP signaling connections active.

### Prototype:

```
int BTPSAPI GAVD_Delete_End_Point_Group(unsigned int BluetoothStackID,  
    unsigned int GroupLSEID)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GroupLSEID	Group identifier of the local stream endpoint group. This is the same group identifier that was passed to the <b>GAVD_Create_Endpoint_Group()</b> function when the group was created.

### Return:

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_GROUP_LSEID_INVALID  
BTGAVD_ERROR_SIGNALLING_CHANNEL_CONNECTED  
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

### Possible Events:

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Connect\_Remote\_End\_Point

This function is used to establish a connection to a remote endpoint on a remote Endpoint Stream Manager. This function will establish a connection to the remote Endpoint Stream Manager if a connection is not already present and establish all required channels (Media, Reporting, Recovery) as needed. The local endpoint that is used is an Endpoint that was registered via the GAVD\_Register\_End\_Point() function.

Notes:

This function differs from the `GAVD_Open_Remote_End_Point()` function in that it does **NOT** create a new Endpoint in the endpoint database. This function will mark an existing, registered, Endpoint as in use, and will attempt to connect this already registered Stream Endpoint to the specified remote Endpoint.

**Prototype:**

```
int BTPSAPI GAVD_Connect_Remote_End_Point(unsigned int BluetoothStackID, unsigned int
    LSEID, BD_ADDR_t BD_ADDR, unsigned int RSEID,
    unsigned int NumberConfigurationCapabilities,
    GAVD_Service_Capabilities_Info_t ConfigurationCapabilities[], GAVD_Event_Callback_t
    GAVDEventCallback,
    unsigned long CallbackParameter)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to <code>BSC_Initialize()</code> .
LSEID	Identifier of the local stream endpoint to use as the local side of the endpoint connection. This value is returned from the <code>GAVD_Register_End_Point()</code> function.
BD_ADDR	Bluetooth address of the remote device.
RSEID	Identifier of the remote stream endpoint to open.
NumberConfigurationCapabilities	The number of configuration options.
ConfigurationCapabilities	The configurations options that will be used to configure the remote endpoint.
GAVDEventCallback	Function that is be called whenever events occur regarding the specified local endpoint.
CallbackParameter	A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each event callback.

**Return:**

Positive, non-zero value if successful that is the LSEID that must be used to identify the connection to the remote stream endpoint in all other endpoint related functions.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

```
etGAVD_Open_End_Point_Confirmation
```

**Notes:**



1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Open\_Remote\_End\_Point

This function is used to establish a connection to a remote endpoint on a remote Endpoint Stream Manager. This function will establish a connection to the remote Endpoint Stream Manager if a connection is not already present and establish all required channels (Media, Reporting, Recovery) as needed. The local endpoint that is registered will not be able to be discovered or connected to. This endpoint will only exist as long as there is a connection to the specified remote endpoint.

### Prototype:

```
int BTPSAPI GAVD_Open_Remote_End_Point(unsigned int BluetoothStackID, BD_ADDR_t
    BD_ADDR, unsigned int RSEID,
    GAVD_Local_End_Point_Info_t *LocalEndPointInfo,
    unsigned int NumberConfigurationCapabilities,
    GAVD_Service_Capabilities_Info_t ConfigurationCapabilities[], GAVD_Event_Callback_t
    GAVDEventCallback,
    unsigned long CallbackParameter)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
BD_ADDR	Bluetooth address of the remote device.
RSEID	Identifier of the remote stream endpoint to open.
LocalEndPointInfo	Contains the necessary information for the local end point that will connect to the remote end point that is being opened. This is defined by the following structure:

```
typedef struct
{
    GAVD_TSEP_t                TSEP;
    GAVD_Media_Type_t          MediaType;
    Word_t                     MediaInMTU;
    Word_t                     ReportingInMTU;
    Word_t                     RecoveryInMTU;
    unsigned int               NumberCapabilities;
    GAVD_Service_Capabilities_Info_t *CapabilitiesInfo;
} GAVD_Local_End_Point_Info_t;
```

where, TSEP defines whether the endpoint is a source or a sink using the following enumeration:

```
typedef enum
{
```

```
tspSRC,  
tspSNK  
} GAVD_TSEP_t;
```

MediaType identifies the media type as audio, video, or multimedia using the following enumeration:

```
typedef enum  
{  
    mtAudio,  
    mtVideo,  
    mtMultimedia  
} GAVD_Media_Type_t;
```

MediaInMTU, ReportingInMTU, and RecoveryInMTU are the maximum allowable MTU sizes.

NumberCapabilities specifies how many service capability information elements are contained in the CapabilitiesInfo member.

CapabilitiesInfo defines the service capabilities that an endpoint supports using the following structure:

```
typedef struct  
{  
    GAVD_Service_Category_t ServiceCategory;  
    union  
    {  
        GAVD_Recovery_Info_Element_Data_t  
            GAVD_Recovery_Info_Element_Data;  
        GAVD_Content_Protection_Info_Element_Data_t  
            GAVD_Content_Protection_Info_Element_Data;  
        GAVD_Header_Compression_Info_Element_Data_t  
            GAVD_Header_Compression_Info_Element_Data;  
        GAVD_Multiplexing_Info_Element_Data_t  
            GAVD_Multiplexing_Info_Element_Data;  
        GAVD_Media_Codec_Info_Element_Data_t  
            GAVD_Media_Codec_Info_Element_Data;  
        GAVD_Raw_Info_Element_Data_t  
            GAVD_Raw_Info_Element_Data;  
    } InfoElement;  
} GAVD_Service_Capabilities_Info_t;
```

where, ServiceCategory is defined by the following enumeration:

```
typedef enum  
{  
    scNone,  
    scMediaTransport,  
    scReporting,
```

```

    scRecovery,
    scContentProtection,
    scHeaderCompression,
    scMultiplexing,
    scMediaCodec,
    scUnknown
} GAVD_Service_Category_t;

```

GAVD\_Recovery\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    Byte_t    RecoveryType;
    Byte_t    MaxRecoveryWindowSize;
    Byte_t    MaxNumberMediaPackets;
} GAVD_Recovery_Info_Element_Data_t;

```

GAVD\_Content\_Protection\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    Word_t    ContentProtectionType;
    Byte_t    ContentProtectionTypeSpecificInfoLength;
    Byte_t    *ContentProtectionTypeSpecificInfo;
} GAVD_Content_Protection_Info_Element_Data_t;

```

GAVD\_Header\_Compression\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    Boolean_t    MediaPacketHeaderCompression;
    Boolean_t    RecoveryPacketHeaderCompression;
    Boolean_t    BackChannelSupported;
} GAVD_Header_Compression_Info_Element_Data_t;

```

GAVD\_Multiplexing\_Info\_Element\_Data is defined by the following structure:

```

typedef struct
{
    unsigned int    MediaMuxLSEID;
    GAVD_Transport_Channel_Type_t MediaMuxChannel;
    Boolean_t    UseReportingChannel;
    unsigned int    ReportingMuxLSEID;
    GAVD_Transport_Channel_Type_t ReportingMuxChannel;
    Boolean_t    UseRecoveryChannel;
    unsigned int    RecoveryMuxLSEID;
    GAVD_Transport_Channel_Type_t RecoveryMuxChannel;
} GAVD_Multiplexing_Info_Element_Data_t;

```

where, MediaMuxChannel, ReportingMuxChannel, and RecoveryMuxChannel are defined by the following enumeration:

```
typedef enum
{
    trMedia,
    trReporting,
    trRecovery,
    trNone
} GAVD_Transport_Channel_Type_t;
```

GAVD\_Media\_Codec\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    GAVD_Media_Type_t    MediaType;
    Byte_t               MediaCodecType;
    Byte_t               MediaCodecSpecificInfoLength;
    Byte_t               *MediaCodecSpecificInfo;
} GAVD_Media_Codec_Info_Element_Data_t;
```

where, MediaType is defined by the following enumeration:

```
typedef enum
{
    mtAudio,
    mtVideo,
    mtMultimedia
} GAVD_Media_Type_t;
```

GAVD\_Raw\_Info\_Element\_Data is defined by the following structure:

```
typedef struct
{
    Byte_t    RawDataLength;
    Byte_t    *RawData;
} GAVD_Raw_Info_Element_Data_t;
```

NumberConfigurationCapabilities	The number of configuration options.
ConfigurationCapabilities	The configurations options that will be used to configure the remote endpoint.
GAVDEventCallback	Function that is be called whenever events occur regarding the specified local endpoint.
CallbackParameter	A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each event callback.

**Return:**

Positive, non-zero value if successful that is the LSEID that must be used to identify the connection to the remote stream endpoint in all other endpoint related functions.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Open\_End\_Point\_Confirmation

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Close\_End\_Point**

This function is used to close a connection to an endpoint that was previously opened via a call to GAVD\_Register\_End\_Point() or GAVD\_Open\_Remote\_End\_Point().

Note:

Calling this function on a local endpoint that was registered via the GAVD\_Register\_End\_Point() function does NOT remove the endpoint from the system, it merely disconnects the currently connected client (i.e., the end point is still present in the system and can be discovered and connected to).

**Prototype:**

```
int BTPSAPI GAVD_Close_End_Point(unsigned int BluetoothStackID,  
    unsigned int LSEID)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Endpoint ID of the local stream endpoint to close. This value was returned from a successful call to the GAVD_Register_End_Point() or GAVD_Open_Remote_End_Point() function.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID

## BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:****Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Set\_Configuration\_Response**

This function is used to respond to a request from a remote GAVD device to set the configuration of a stream endpoint.

**Prototype:**

```
int BTPSAPI GAVD_Set_Configuration_Response(unsigned int BluetoothStackID, unsigned int
LSEID, GAVD_Service_Category_t FirstFailingServiceCategory,
unsigned int ErrorCode)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID that is being configured.
FirstFailingServiceCategory	This represents the first service category that was requested to be configured that was unable to be configured (scNone if configuration was successful). Defined by the following enumeration:

```
typedef enum
{
    scNone,
    scMediaTransport,
    scReporting,
    scRecovery,
    scContentProtection,
    scHeaderCompression,
    scMultiplexing,
    scMediaCodec,
    scUnknown
} GAVD_Service_Category_t;
```

ErrorCode	Error response code of the configuration response. This value is one of the following values:
-----------	-----------------------------------------------------------------------------------------------

```
GAVD_AVDTP_ERROR_SUCCESS
GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT
GAVD_AVDTP_ERROR_BAD_LENGTH
GAVD_AVDTP_ERROR_BAD_ACP_SEID
GAVD_AVDTP_ERROR_SEP_IN_USE
GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
```

```

GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

**Possible Events:****Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Start\_Stream\_Request**

This function is used to request to start one or more streams on the remote GAVD device.

**Prototype:**

```
int BTPSAPI GAVD_Start_Stream_Request(unsigned int BluetoothStackID,
    unsigned int NumberStreams, unsigned int LSEID[])
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
NumberStreams	Number of streams to start. This specifies the number of local stream endpoint identifiers that are present in the LSEID parameter array.
LSEID[]	List of local stream endpoint IDs to start. This parameter must point to (at least) the number of local stream endpoints specified by the NumberStreams parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_INSUFFICIENT\_RESOURCES  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Start\_Confirmation  
etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Start\_Stream\_Response**

This function is used to respond to a request from a remote GAVD device to start one or more streams.

Note:

This function can ONLY be called from within the context of the Stream Endpoint Event Callback function in response to an Endpoint Start Request. Failure to call this function in the event callback during a Stream Endpoint Start Request Event will cause the GAVD Profile to respond automatically (with an error response), which means the stream(s) will not have been started.

**Prototype:**

int BTPSAPI **GAVD\_Start\_Stream\_Response**(unsigned int BluetoothStackID,  
unsigned int LSEID, unsigned int ErrorCode)

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that this response is for.
ErrorCode	Error response code of the start stream response. This value is one of the following values:

GAVD\_AVDTP\_ERROR\_SUCCESS  
GAVD\_AVDTP\_ERROR\_BAD\_HEADER\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_LENGTH  
GAVD\_AVDTP\_ERROR\_BAD\_ACP\_SEID  
GAVD\_AVDTP\_ERROR\_SEP\_IN\_USE



```

GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Suspend\_Stream\_Request**

This function is used to request to suspend one or more streams on the remote GAVD device.

**Prototype:**

```
int BTPSAPI GAVD_Suspend_Stream_Request(unsigned int BluetoothStackID,
    unsigned int NumberStreams, unsigned int LSEID[])
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
NumberStreams	Number of streams to suspend. This specifies the number of local stream endpoint identifiers that are present in the LSEID parameter array.

LSEID[] List of local stream endpoint IDs to suspend. This parameter must point to (at least) the number of local stream endpoints specified by the NumberStreams parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
 BTGAVD\_ERROR\_INSUFFICIENT\_RESOURCES  
 BTGAVD\_ERROR\_NOT\_INITIALIZED  
 BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
 BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Suspend\_Confirmation  
 etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Suspend\_Stream\_Response**

This function is used to respond to a request from a remote GAVD device to suspend one or more streams.

Note:

This function can ONLY be called from within the context of the Stream Endpoint Event Callback function in response to an Endpoint Suspend Request. Failure to call this function in the event callback during a Stream Endpoint Suspend Request Event will cause the GAVD Profile to respond automatically (with an error response), which means the stream(s) will not have been suspended.

**Prototype:**

int BTPSAPI GAVD\_Suspend\_Stream\_Response(unsigned int BluetoothStackID, unsigned int LSEID, unsigned int ErrorCode)

**Parameters:**

BluetoothStackID<sup>1</sup> Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC\_Initialize().

LSEID Local stream endpoint ID of the endpoint that this response is for.

ErrorCode Error response code of the suspend stream response. This value is one of the following values:

GAVD\_AVDTP\_ERROR\_SUCCESS

GAVD\_AVDTP\_ERROR\_BAD\_HEADER\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_LENGTH  
GAVD\_AVDTP\_ERROR\_BAD\_ACP\_SEID  
GAVD\_AVDTP\_ERROR\_SEP\_IN\_USE  
GAVD\_AVDTP\_ERROR\_SEP\_NOT\_IN\_USE  
GAVD\_AVDTP\_ERROR\_BAD\_SERV\_CATEGORY  
GAVD\_AVDTP\_ERROR\_BAD\_PAYLOAD\_FORMAT  
GAVD\_AVDTP\_ERROR\_NOT\_SUPPORTED\_COMMAND  
GAVD\_AVDTP\_ERROR\_INVALID\_CAPABILITIES  
GAVD\_AVDTP\_ERROR\_BAD\_RECOVERY\_TYPE  
GAVD\_AVDTP\_ERROR\_BAD\_MEDIA\_TRANSPORT  
\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_RECOVERY\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_ROHC\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_CP\_FORMAT  
GAVD\_AVDTP\_ERROR\_BAD\_MULTIPLEXING\_FORMAT  
GAVD\_AVDTP\_ERROR\_UNSUPPORTED  
\_CONFIGURATION  
GAVD\_AVDTP\_ERROR\_BAD\_STATE  
GAVD\_AVDTP\_ERROR\_TIMEOUT

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Reconfigure\_Request**

This function is used to request the reconfiguration of an endpoint on a remote GAVD device.

**Note:**

The only service capabilities that are allowed to be reconfigured are Media Codec Capabilities and Content Protection Capabilities.

**Prototype:**

```
int BTPSAPI GAVD_Reconfigure_Request(unsigned int BluetoothStackID,
    unsigned int LSEID, unsigned int NumberServiceCapabilities,
    GAVD_Service_Capabilities_Info_t ServiceCapabilities[])
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected and is being reconfigured.
NumberServiceCapabilities	Number of service capabilities to reconfigure. This value specifies the number of Service Capabilities pointed to by the ServiceCapabilities parameter.
ServiceCapabilities[]	Array of service capabilities to reconfigure. This is defined by the following structure:

```
typedef struct
{
    GAVD_Service_Category_t ServiceCategory;
    union
    {
        GAVD_Recovery_Info_Element_Data_t
            GAVD_Recovery_Info_Element_Data;
        GAVD_Content_Protection_Info_Element_Data_t
            GAVD_Content_Protection_Info_Element_Data;
        GAVD_Header_Compression_Info_Element_Data_t
            GAVD_Header_Compression_Info_Element_Data;
        GAVD_Multiplexing_Info_Element_Data_t
            GAVD_Multiplexing_Info_Element_Data;
        GAVD_Media_Codec_Info_Element_Data_t
            GAVD_Media_Codec_Info_Element_Data;
        GAVD_Raw_Info_Element_Data_t
            GAVD_Raw_Info_Element_Data;
    } InfoElement;
} GAVD_Service_Capabilities_Info_t;
```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

etGAVD\_Reconfigure\_Confirmation

etGAVD\_Close\_End\_Point\_Indication

#### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Reconfigure\_Response

This function is used to respond to a request from a remote GAVD device to reconfigure a stream.

#### Prototype:

```
int BTPSAPI GAVD_Reconfigure_Response(unsigned int BluetoothStackID,
    unsigned int LSEID, GAVD_Service_Category_t FirstFailingServiceCategory,
    unsigned int ErrorCode)
```

#### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected to the stream that is being reconfigured.
FirstFailingServiceCategory	This represents the first service category that was requested to be reconfigured that was unable to be reconfigured (scNone if reconfiguration was successful). Defined by the following enumeration: <pre>typedef enum {     scNone,     scMediaTransport,     scReporting,     scRecovery,     scContentProtection,     scHeaderCompression,     scMultiplexing,     scMediaCodec,     scUnknown } GAVD_Service_Category_t;</pre>
ErrorCode	Error response code of the reconfigure response. This value is one of the following values: <pre>GAVD_AVDTP_ERROR_SUCCESS GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT GAVD_AVDTP_ERROR_BAD_LENGTH GAVD_AVDTP_ERROR_BAD_ACP_SEID GAVD_AVDTP_ERROR_SEP_IN_USE GAVD_AVDTP_ERROR_SEP_NOT_IN_USE</pre>

```

GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Security\_Control\_Request**

This function is used to request the change of the security settings for a stream.

**Prototype:**

```

int BTPSAPI GAVD_Security_Control_Request(unsigned int BluetoothStackID,
    unsigned int LSEID, unsigned int ContentProtectionDataLength,
    Byte_t *ContentProtectionData)

```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected to the stream.

ContentProtectionDataLenth	Length of the security data. This specifies the length (in bytes) of the Content Protection Data (this is the length of the buffer that is pointed to by the ContentProtectionData parameter).
ContentProtectionData	Pointer to the Content Protection Data. This is pointer to a buffer that contains the Content Protection Data (of length specified by the ContentProtectionDataLength parameter).

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

```
etGAVD_Security_Control_Confirmation
etGAVD_Close_End_Point_Indication
```

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Security\_Control\_Response**

This function is used to respond to a request from a remote GAVD device to change the security settings for a stream.

**Prototype:**

```
int BTPSAPI GAVD_Security_Control_Response(unsigned int BluetoothStackID, unsigned int
    LSEID, unsigned int ErrorCode, unsigned int ContentProtectionDataLength, Byte_t
    *ContentProtectionData)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected to the stream.
ErrorCode	Error response code of the security control response. This value is one of the following values: <pre>GAVD_AVDTP_ERROR_SUCCESS GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT GAVD_AVDTP_ERROR_BAD_LENGTH</pre>

GAVD\_AVDTP\_ERROR\_BAD\_ACP\_SEID  
 GAVD\_AVDTP\_ERROR\_SEP\_IN\_USE  
 GAVD\_AVDTP\_ERROR\_SEP\_NOT\_IN\_USE  
 GAVD\_AVDTP\_ERROR\_BAD\_SERV\_CATEGORY  
 GAVD\_AVDTP\_ERROR\_BAD\_PAYLOAD\_FORMAT  
 GAVD\_AVDTP\_ERROR\_NOT\_SUPPORTED\_COMMAND  
 GAVD\_AVDTP\_ERROR\_INVALID\_CAPABILITIES  
 GAVD\_AVDTP\_ERROR\_BAD\_RECOVERY\_TYPE  
 GAVD\_AVDTP\_ERROR\_BAD\_MEDIA\_TRANSPORT  
 \_FORMAT  
 GAVD\_AVDTP\_ERROR\_BAD\_RECOVERY\_FORMAT  
 GAVD\_AVDTP\_ERROR\_BAD\_ROHC\_FORMAT  
 GAVD\_AVDTP\_ERROR\_BAD\_CP\_FORMAT  
 GAVD\_AVDTP\_ERROR\_BAD\_MULTIPLEXING\_FORMAT  
 GAVD\_AVDTP\_ERROR\_UNSUPPORTED  
 \_CONFIGURATION  
 GAVD\_AVDTP\_ERROR\_BAD\_STATE  
 GAVD\_AVDTP\_ERROR\_TIMEOUT

**ContentProtectionDataLength** Length of the security data. This specifies the length (in bytes) of the Content Protection Data (this is the length of the buffer that is pointed to by the ContentProtectionData parameter).

**ContentProtectionData** Pointer to the Content Protection Data. This is pointer to a buffer that contains the Content Protection Data (of length specified by the ContentProtectionDataLength parameter).

### Return:

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_INVALID\_OPERATION  
 BTGAVD\_ERROR\_INSUFFICIENT\_RESOURCES  
 BTGAVD\_ERROR\_NOT\_INITIALIZED  
 BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
 BTGAVD\_ERROR\_INVALID\_PARAMETER

### Possible Events:

etGAVD\_Close\_End\_Point\_Indication

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Abort\_Stream\_Request

This function is used to request to abort one or more streams on the remote GAVD device.



**Prototype:**

```
int BTPSAPI GAVD_Abort_Stream_Request(unsigned int BluetoothStackID, unsigned int LSEID)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected to the stream.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION  
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

```
etGAVD_Abort_Confirmation  
etGAVD_Close_End_Point_Indication
```

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Data\_Write**

This function is used to send data over the specified (started) stream.

**Prototype:**

```
int BTPSAPI GAVD_Data_Write(unsigned int BluetoothStackID, unsigned int LSEID, Boolean_t Marker, Byte_t PayloadType, DWord_t TimeStamp, Word_t DataLength, Byte_t *DataBuffer)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is connected to the stream.
Marker	Specifies whether the marker bit (used by certain codec payload formats) is to be used with the data written.
PayloadType	Indicates the content type of the data.
TimeStamp	Holds time stamp information.

DataLength	Specifies the length of the payload. This parameter specifies the length (in bytes) of the payload data that is to be written.
DataBuffer	Points to the payload data. This parameter is a pointer to the payload data to be written to the specified stream endpoint. This pointer must point to (at least) the number of bytes specified by the DataLength parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_L2CAP_MTU_EXCEEDED
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Sender\_Report\_Data\_Write**

This function is used to send Sender Report data over the specified stream.

**Prototype:**

```
int BTPSAPI GAVD_Sender_Report_Data_Write(unsigned int BluetoothStackID, unsigned int
LSEID, GAVD_Sender_Info_t *SenderInfo,
unsigned int NumberReportBlocks, GAVD_Report_Block_t *ReportBlocks,
Word_t ExtensionDataLength, DWord_t *ExtensionData)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is sourcing the data.
SenderInfo	Points to the Sender Information in the Sender Report. This is defined by the following structure:

```
typedef struct
{
    DWord_t    NTPTimestampMSW;
    DWord_t    NTPTimestampLSW;
```

	<pre> DWord_t    RTPTimeStamp; DWord_t    PacketCount; DWord_t    OctetCount; } GAVD_Sender_Info_t; </pre>
NumberReportBlocks	Indicates the number of report blocks pointed to by the ReportBlocks parameter.
ReportBlocks	Points to the report blocks. This is defined by the following structure: <pre> typedef struct {     DWord_t    SourceID;     Byte_t     FractionLost;     DWord_t    TotalPacketsLost;     Word_t     HighestSeqNumReceivedCycleCount;     Word_t     HighestSeqNumReceived;     DWord_t    IntervalJitter;     DWord_t    LastReport;     DWord_t    DelaySinceLastReport; } GAVD_Report_Block_t; </pre>
ExtensionDataLength	Indicates how many 32-bit words are in the extension data.
ExtensionData	Points to the extension data. This parameter points to an array of 32-bit words. The number of 32-bit words pointed to by this parameter is given by the ExtensionDataLength parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```

BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_L2CAP_MTU_EXCEEDED
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER

```

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Receiver\_Report\_Data\_Write

This function is used to send Receiver Report data over the specified stream.

### Prototype:

```
int BTPSAPI GAVD_Receiver_Report_Data_Write(unsigned int BluetoothStackID, unsigned int
    LSEID, unsigned int NumberReportBlocks,
    GAVD_Report_Block_t *ReportBlocks, Word_t ExtensionDataLength,
    DWord_t *ExtensionData)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is sourcing the data.
NumberReportBlocks	Indicates the number of report blocks pointed to by the ReportBlocks parameter.
ReportBlocks	Points to the report blocks. This is defined by the following structure: <pre>typedef struct {     DWord_t    SourceID;     Byte_t     FractionLost;     DWord_t    TotalPacketsLost;     Word_t     HighestSeqNumReceivedCycleCount;     Word_t     HighestSeqNumReceived;     DWord_t    IntervalJitter;     DWord_t    LastReport;     DWord_t    DelaySinceLastReport; } GAVD_Report_Block_t;</pre>
ExtensionDataLength	Indicates how many 32-bit words are in the extension data.
ExtensionData	Points to the extension data. This parameter points to an array of 32-bit words. The number of 32-bit words pointed to by this parameter is given by the ExtensionDataLength parameter.

### Return:

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_L2CAP_MTU_EXCEEDED
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

### Possible Events:

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_SDES\_Report\_Data\_Write**

This function is used to send SDES Report data over the specified stream.

**Prototype:**

```
int BTPSAPI GAVD_SDES_Report_Data_Write(unsigned int BluetoothStackID,
    unsigned int LSEID, unsigned int NumberSDESChunks,
    GAVD_SDES_Chunk_t *SDESChunks)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is sourcing the data.
NumberSDESChunks	Indicates the number of SDES chunks pointed to by the SDESChunks parameter.
SDESChunks	Points to the SDES chunks. This is defined by the following structure:

```
typedef struct
{
    DWord_t          SourceID;
    unsigned int      NumberSDESItems;
    GAVD_SDES_Item_t *SDESItems;
} GAVD_SDES_Chunk_t;
```

where, SDESItems is defined by the following structure:

```
typedef struct
{
    Byte_t    ItemDescriptor;
    Byte_t    ItemLength;
    Byte_t    *ItemData;
} GAVD_SDES_Item_t;
```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_L2CAP_MTU_EXCEEDED
BTGAVD_ERROR_INSUFFICIENT_RESOURCES
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
```

## BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Recovery\_Data\_Write**

This function is used to send RTP FEC recovery data over the specified stream.

**Prototype:**

```
int BTPSAPI GAVD_Recovery_Data_Write(unsigned int BluetoothStackID,
    unsigned int LSEID, Boolean_t Marker, Byte_t PayloadType, DWord_t TimeStamp,
    GAVD_FEC_Block_t *FECBlock)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint that is sourcing the data.
Marker	Specifies the marker bit used by certain codec payload formats.
PayloadType	Indicates the content type of the data.
TimeStamp	Holds time stamp information.
FECBlock	Points to the structure with the information to send. This is defined by the following structure:

```
typedef struct
{
    Word_t      SequenceNumberBase;
    Word_t      LengthRecovery;
    Byte_t      PayloadTypeRecovery;
    DWord_t     Mask;
    DWord_t     TimeStampRecovery;
    unsigned int FECDataLength;
    Byte_t      *FECData;
} GAVD_FEC_Block_t;
```

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_INVALID_OPERATION
BTGAVD_ERROR_L2CAP_MTU_EXCEEDED
```

BTGAVD\_ERROR\_INSUFFICIENT\_RESOURCES  
BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Possible Events:**

etGAVD\_Close\_End\_Point\_Indication

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Get\_Server\_Connection\_Mode**

Retrieves the current GAVD/AVDTP Server Connection Mode. The default is gsmAutomaticAccept. This function is used for GAVD/AVDTP Servers which use Bluetooth Security Mode 2.

**Prototype:**

```
int BTPSAPI GAVD_Get_Server_Connection_Mode(unsigned int BluetoothStackID,  
      GAVD_Server_Connection_Mode_t *ServerConnectionMode)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
ServerConnectionMode	Pointer Server Connection Mode variable to put the current GAVD/AVDTP Server Connection Mode in. The possible values returned are:  gsmAutomaticAccept gsmAutomaticReject gsmManualAccept

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Set\_Server\_Connection\_Mode

Changes the current GAVD/AVDTP Server Connection Mode. The default is `gsmAutomaticAccept`. This function is used for GAVD/AVDTP Servers which use Bluetooth Security Mode 2.

### Prototype:

```
int BTPSAPI GAVD_Set_Server_Connection_Mode(unsigned int BluetoothStackID,  
      GAVD_Server_Connection_Mode_t ServerConnectionMode, GAVD_Event_Callback_t  
      GAVDEventCallback, unsigned long CallbackParameter)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to <code>BSC_Initialize()</code> .
ServerConnectionMode	Value to change the current GAVD/AVDTP Server Connection Mode to. If the server mode is anything other than <code>gsmManualAccept</code> then the final two parameters are ignored. The possible values are:  <code>gsmAutomaticAccept</code> <code>gsmAutomaticReject</code> <code>gsmManualAccept</code>
GAVDEventCallback	Event Callback that is used to receive notifications of a Bluetooth Connection Request.
CallbackParameter	Parameter to the callback function described above.

### Return:

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

### Notes:

1. The `BluetoothStackID` parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Register\_Signalling\_Connection\_Status

This function allows the ability to register to receive GAVD/AVDTP signaling channel status events. This allows applications the ability to determine when remote devices are connected and the overall state/operation of the signaling channel (i.e. when it is idle).

### Notes:

There can only be a single GAVD/AVDTP signaling channel status callback registered in the system.



**Prototype:**

```
int BTPSAPI GAVD_Register_Signalling_Connection_Status(unsigned int BluetoothStackID,  
    GAVD_Event_Callback_t GAVDEventCallback, unsigned long CallbackParameter)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDEventCallback	Function that is be called whenever events occur regarding the GAVD/AVDTP signaling channel occur.
CallbackParameter	A user-defined parameter (e.g., a tag value) that will be passed back to the user in the callback function with each event callback.

**Return:**

Positive, non-zero value if successful that represents the GAVD signaling channel callback ID. This parameter can be passed to the GAVD\_Un\_Register\_Signalling\_Connection\_Status() function to un-register the callback.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

**Possible Events:**

```
etGAVD_Signalling_Connect_Indication  
etGAVD_Signalling_Disconnect_Indication  
etGAVD_Signalling_Channel_Idle_Indication  
etGAVD_Signalling_Channel_Endpoint_Open_Indication  
etGAVD_Signalling_Channel_Endpoint_Close_Indication
```

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Un\_Register\_Signalling\_Connection\_Status**

This function allows the ability to un-register a previously registered GAVD/AVDTP signaling channel status callback.

Notes:

There can only be a single GAVD/AVDTP signaling channel status callback registered in the system.

**Prototype:**

```
int BTPSAPI GAVD_Un_Register_Signalling_Connection_Status(unsigned int BluetoothStackID,  
    unsigned int GAVDConnectionStatusID)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDConnectionStatusID	Event Callback identifier of the registered GAVD/AVDTP signaling channel status callback. This value is the value that was returned from a successful call to the GAVD_Register_Signalling_Connection_Status() function.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Disconnect\_Signalling\_Connection**

This function allows the ability to force a disconnection of a currently connected GAVD/AVDTP signaling channel connection (to a specific Bluetooth device). It is recommended that this function be used in conjunction with the GAVD/AVDTP signaling channel status events (registered via the GAVD\_Register\_Signalling\_Connection\_Status() function) to be aware of the current signaling channel status (i.e. idle versus active).

Notes:

Disconnecting an active (i.e. a signaling channel with open/active endpoints) signaling channel will cause all endpoints that are connected to the same Bluetooth device to be disconnected. For this reason it is strongly suggested that anyone who calls this API to track the state of the connection with the specified remote device.

**Prototype:**

```
int BTPSAPI GAVD_Disconnect_Signalling_Connection(unsigned int BluetoothStackID,  
          BD_ADDR_t BD_ADDR)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
BD_ADDR	Bluetooth device address of the connected remote Bluetooth device that is have the GAVD/AVDTP signaling channel disconnected.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_CONNECT_REQUEST_IN_PROGRESS
BTGAVD_ERROR_DISCONNECT_REQUEST_IN_PROGRESS
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Get\_Data\_Queueing\_Parameters**

Retrieves the current GAVD/L2CAP data queueing parameters. These parameters dictate how the data packets are queued into L2CAP (when calling the GAVD\_Data\_Write() function). This mechanism allows for the ability to implement a streaming type interface by limiting the number of packets that can be queued (simultaneously) in L2CAP. This is useful to keep L2CAP from infinitely queuing packets which can lead to stale data if there is an issue sending the data to the remote device (i.e. interference).

Notes:

This function sets the queuing parameters globally for GAVD. Setting the Queuing parameters for an individual stream endpoint is currently not supported. This is because multiple stream endpoints could be multiplexed over the same channel.

A value of zero for the QueueLimit member of the L2CAP queuing parameters means that there is no queuing active (i.e. all packets are queued, regardless of the queue depth).

It is recommended to ALWAYS use the L2CA\_QUEUEING\_FLAG\_DISCARD\_OLDEST flag when specifying queuing parameters. Although the threshold method can be used (by having the queue fail and waiting for a Data buffer empty indication), it is recommended to allow the data to always be queued and have the oldest data deleted from the queue and have the newest data placed at the end of the queue (i.e. always queue the data).

**Prototype:**

```
int BTPSAPI GAVD_Get_Data_Queueing_Parameters(unsigned int BluetoothStackID,
        L2CA_Queueing_Parameters_t *QueueingParameters)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
QueueingParameters	Pointer to a structure that will contain the currently configured Queuing Parameters that are currently used by GAVD. See the

L2CAP\_Enhanced\_Data\_Write() function (in Bluetopia Core API documentation) for more information on the values for this parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Set\_Data\_Queueing\_Parameters**

Sets the current GAVD/L2CAP data queueing parameters. These parameters dictate how the data packets are queued into L2CAP (when calling the GAVD\_Data\_Write() function). This mechanism allows for the ability to implement a streaming type interface by limiting the number of packets that can be queued (simultaneously) in L2CAP. This is useful to keep L2CAP from infinitely queuing packets which can lead to stale data if there is an issue sending the data to the remote device (i.e. interference).

Notes:

This function sets the queuing parameters globally for GAVD. Setting the Queuing parameters for an individual stream endpoint is currently not supported. This is because multiple stream endpoints could be multiplexed over the same channel.

A value of zero for the QueueLimit member of the L2CAP queuing parameters means that there is no queuing active (i.e. all packets are queued, regardless of the queue depth).

It is recommended to ALWAYS use the L2CA\_QUEUEING\_FLAG\_DISCARD\_OLDEST flag when specifying queuing parameters. Although the threshold method can be used (by having the queue fail and waiting for a Data buffer empty indication), it is recommended to allow the data to always be queued and have the oldest data deleted from the queue and have the newest data placed at the end of the queue (i.e. always queue the data).

**Prototype:**

```
int BTPSAPI GAVD_Set_Data_Queueing_Parameters(unsigned int BluetoothStackID,  
        L2CA_Queueing_Parameters_t *QueueingParameters)
```

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
-------------------------------	---------------------------------------------------------------------------------------------

**QueueingParameters**      Pointer to a structure that contains the new Queuing Parameters to set. See the `L2CAP_Enhanced_Data_Write()` function (in Bluetopia Core API documentation) for more information on the values for this parameter.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_NOT_INITIALIZED
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID
BTGAVD_ERROR_INVALID_PARAMETER
```

**Notes:**

1. The `BluetoothStackID` parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

**GAVD\_Get\_L2CAP\_Channel\_Info**

This function provides a mechanism of retrieving the the L2CAP information for a specified Local Stream End Point.

**Prototype:**

```
int BTPSAPI GAVD_Get_L2CAP_Channel_Info(unsigned int BluetoothStackID, unsigned int
    LSEID, GAVD_L2CAP_Information_t *L2CAPInformation)
```

**Parameters:**

<b>BluetoothStackID<sup>1</sup></b>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to <code>BSC_Initialize()</code> .
<b>LSEID</b>	Identifier for the local Stream End Point to query the L2CAP Information for.
<b>L2CAPInformation</b>	Pointer to a structure to return the L2CAP Information. This structure is defined as follows: <pre>typedef struct {     Word_t OutMTU;     Word_t InFlushTO;     Word_t OutFlushTO;     Word_t RemoteCID;     Word_t LocalCID; } GAVD_L2CAP_Information_t;</pre>

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## **GAVD\_Get\_End\_Point\_All\_Capabilities**

This function provides a mechanism for retrieving all end-point capabilities for a remote stream endpoint.

**Prototype:**

int BTPSAPI **GAVD\_Get\_End\_Point\_All\_Capabilities**(unsigned int BluetoothStackID, unsigned int GAVDID, unsigned int RSEID)

**Parameters:**

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
GAVDID	The GAVD Client ID. This is the value that was returned from the GAVD_Connect() function.
RSEID	The remote endpoint ID. Endpoint to query the capabilities for. This value is one of the values returned in a Discover Confirmation event.

**Return:**

Zero if successful.

An error code if negative; one of the following values:

BTGAVD\_ERROR\_NOT\_INITIALIZED  
BTGAVD\_ERROR\_INVALID\_OPERATION  
BTGAVD\_ERROR\_INVALID\_BLUETOOTH\_STACK\_ID  
BTGAVD\_ERROR\_INVALID\_PARAMETER

**Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Set\_Delay\_Report

This function provides a mechanism for setting the delay report indicating the time delay between a video and an audio source in 100 microsecond units. GAVD does not currently support video endpoints. This function is provided solely to provide required functionality for testing.

### Prototype:

```
int BTPSAPI GAVD_Set_Delay_Report(unsigned int BluetoothStackID, unsigned int LSEID,  
    Word_t Delay)
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize().
LSEID	Local stream endpoint ID of the endpoint to set the delay report for.
Delay	Reported delay, in 100 microsecond units, between a video frame and the audio that synchronizes to the beginning of that frame. May be positive or negative.

### Return:

Zero if successful.

An error code if negative; one of the following values:

```
BTGAVD_ERROR_NOT_INITIALIZED  
BTGAVD_ERROR_INVALID_OPERATION  
BTGAVD_ERROR_INVALID_BLUETOOTH_STACK_ID  
BTGAVD_ERROR_INVALID_PARAMETER
```

### Notes:

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## GAVD\_Set\_Reject\_Response\_State:

This function is provided for testing purposes only and is expected to be deprecated and replaced in a future revision.

## 2.2 GAVD Profile Event Callback Prototypes

The event callback functions mentioned in the GAVD Profile functions all accept the callback function described by the following prototype.

## GAVD\_Event\_Callback\_t

Prototype of callback function passed in to the GAVD\_Connect(), GAVD\_Register\_End\_Point(), and GAVD\_Open\_Remote\_End\_Point() functions. This callback function is dispatched whenever events destined for the specified connection/endpoint occur.

### Prototype:

```
void (BTPSAPI *GAVD_Event_Callback_t)(unsigned int BluetoothStackID,
    GAVD_Event_Data_t *GAVD_Event_Data, unsigned long CallbackParameter);
```

### Parameters:

BluetoothStackID <sup>1</sup>	Unique identifier assigned to this Bluetooth Protocol Stack via a call to BSC_Initialize()
GAVD_Event_Data	Data describing the event for which the callback function is called. This is defined by the following structure:

```
typedef struct
{
    GAVD_Event_Type_t    Event_Data_Type;
    Word_t               Event_Data_Size;
    union
    {
        GAVD_Connect_Request_Indication_Data_t
            *GAVD_Connect_Request_Indication_Data;
        GAVD_Connect_Confirmation_Data_t
            *GAVD_Connect_Confirmation_Data;
        GAVD_Disconnect_Indication_Data_t
            *GAVD_Disconnect_Indication_Data;
        GAVD_Discover_Confirmation_Data_t
            *GAVD_Discover_Confirmation_Data;
        GAVD_Get_Capabilities_Confirmation_Data_t
            *GAVD_Get_Capabilities_Confirmation_Data;
        GAVD_Get_Configuration_Confirmation_Data_t
            *GAVD_Get_Configuration_Confirmation_Data;
        GAVD_Set_Configuration_Indication_Data_t
            *GAVD_Set_Configuration_Indication_Data;
        GAVD_Open_End_Point_Indication_Data_t
            *GAVD_Open_End_Point_Indication_Data;
        GAVD_Open_End_Point_Confirmation_Data_t
            *GAVD_Open_End_Point_Confirmation_Data;
        GAVD_Close_End_Point_Indication_Data_t
            *GAVD_Close_End_Point_Indication_Data;
        GAVD_Start_Indication_Data_t
            *GAVD_Start_Indication_Data;
        GAVD_Start_Confirmation_Data_t
            *GAVD_Start_Confirmation_Data;
        GAVD_Suspend_Indication_Data_t
            *GAVD_Suspend_Indication_Data;
    }
};
```



```

    GAVD_Suspend_Confirmation_Data_t
        *GAVD_Suspend_Confirmation_Data;
    GAVD_Reconfigure_Indication_Data_t
        *GAVD_Reconfigure_Indication_Data;
    GAVD_Reconfigure_Confirmation_Data_t
        *GAVD_Reconfigure_Confirmation_Data;
    GAVD_Security_Control_Indication_Data_t
        *GAVD_Security_Control_Indication_Data;
    GAVD_Security_Control_Confirmation_Data_t
        *GAVD_Security_Control_Confirmation_Data;
    GAVD_Abort_Indication_Data_t
        *GAVD_Abort_Indication_Data;
    GAVD_Abort_Confirmation_Data_t
        *GAVD_Abort_Confirmation_Data;
    GAVD_Data_Indication_Data_t
        *GAVD_Data_Indication_Data;
    GAVD_Sender_Report_Data_Indication_Data_t
        *GAVD_Sender_Report_Data_Indication_Data;
    GAVD_Receiver_Report_Data_Indication_Data_t
        *GAVD_Receiver_Report_Data_Indication_Data;
    GAVD_SDES_Report_Data_Indication_Data_t
        *GAVD_SDES_Report_Data_Indication_Data;
    GAVD_Recovery_Data_Indication_Data_t
        *GAVD_Recovery_Data_Indication_Data;
    GAVD_Channel_Empty_Indication_Data_t
        *GAVD_Channel_Empty_Indication_Data;
    GAVD_Signalling_Connect_Indication_Data_t
        *GAVD_Signalling_Connect_Indication_Data;
    GAVD_Signalling_Disconnect_Indication_Data_t
        *GAVD_Signalling_Disconnect_Indication_Data;
    GAVD_Signalling_Channel_Idle_Indication_Data_t
        *GAVD_Signalling_Channel_Idle_Indication_Data;
    GAVD_Signalling_Channel_Endpoint_Open_Indication_Data_t
        *GAVD_Signalling_Channel_Endpoint_Open_Indication_
            Data;
    GAVD_Signalling_Channel_Endpoint_Close_Indication_Data_t
        *GAVD_Signalling_Channel_Endpoint_Close_Indication_
            Data;
    GAVD_Delay_Report_Indication_Data_t
        *GAVD_Delay_Report_Indication_Data;
    GAVD_Delay_Report_Confirmation_Data_t
        *GAVD_Delay_Report_Confirmation_Data;
    GAVD_General_Reject_Indication_Data_t
        *GAVD_General_Reject_Indication_Data;
} Event_Data;
} GAVD_Event_Data_t;

```

where, Event\_Data\_Type is one of the enumerations of the event types listed in the table in section 2.3, and each data structure in the union is described with its event in that section as well.

**CallbackParameter**      User-defined parameter (e.g., tag value) that was defined in the callback registration. This value is passed to the caller each time the event callback is dispatched.

**Return:****Notes:**

1. The BluetoothStackID parameter is not included in versions of Bluetopia that have been optimized to only control a single Bluetooth device, such as some embedded versions of Bluetopia. Please refer to the appropriate header file to determine if this parameter is part of the function call or not.

## 2.3 GAVD Profile Events

The possible GAVD Profile events from the Bluetooth stack are listed in the table below and are described in the text that follows:

Event	Description
etGAVD_Connect_Request_Indication	Dispatched when a remote device is requesting a connection to the local device.
etGAVD_Connect_Confirmation	Confirms that a GAVD remote Stream Endpoint Manager connect request has been responded to or has encountered an error.
etGAVD_Disconnect_Indication	Indicates that a local client has been disconnected from a remote Stream Endpoint Manager.
etGAVD_Discover_Confirmation	Confirms the completion of a Stream Discover command that was sent to a remote Stream Endpoint Manager.
etGAVD_Get_Capabilities_Confirmation	Confirms the completion of a Get Capabilities command that was sent to a remote Stream Endpoint Manager.
etGAVD_Get_Configuration_Confirmation	Confirms the completion of a Get Configuration command.
etGAVD_Set_Configuration_Indication	Indicates that a Set Configuration command has been received from a remote Stream Endpoint Manager.
etGAVD_Open_End_Point_Indication	Indicates that an Open Endpoint command has been received from a remote Stream Endpoint Manager.
etGAVD_Open_End_Point_Confirmation	Confirms the completion of an Open Endpoint command that was sent to a remote Stream Endpoint Manager.
etGAVD_Close_End_Point_Indication	Indicates that a Close Endpoint command has been

	received from a remote Stream Endpoint Manager.
etGAVD_Start_Indication	Indicates that a Start Stream command has been received from a remote Stream Endpoint Manager.
etGAVD_Start_Confirmation	Confirms that a Start Stream command sent to a remote Stream Endpoint Manager was received.
etGAVD_Suspend_Indication	Indicates that a Suspend Stream command has been received from a remote Stream Endpoint Manager.
etGAVD_Suspend_Confirmation	Confirms that a Suspend Stream command sent to a remote Stream Endpoint was received.
etGAVD_Reconfigure_Indication	Indicates that a Reconfigure Stream command has been received from a remote Stream Endpoint Manager.
etGAVD_Reconfigure_Confirmation	Confirms the completion of a Reconfigure Stream command that was sent to a remote Stream Endpoint Manager.
etGAVD_Security_Control_Indication	Indicates that a Security Control command has been received from a remote Stream Endpoint Manager.
etGAVD_Security_Control_Confirmation	Confirms the completion of a Security Control command that was sent to a remote Stream Endpoint Manager.
etGAVD_Abort_Indication	Indicates that an Abort Stream command has been received from a remote Stream Endpoint Manager.
etGAVD_Abort_Confirmation	Confirms the completion of an Abort Stream command that was sent to a remote Stream Endpoint Manager.
etGAVD_Data_Indication	Indicates that data has been received from a remote Endpoint.
etGAVD_Sender_Report_Data_Indication	Indicates that Sender Report data has been received from a remote Endpoint.
etGAVD_Receiver_Report_Data_Indication	Indicates that Receiver Report data has been received from a remote Endpoint.
etGAVD_SDES_Report_Data_Indication	Indicates that SDES Report data has been received from a remote Endpoint.
etGAVD_Recovery_Data_Indication	Indicates that Recovery data has been received from a remote Endpoint.
etGAVD_Data_Channel_Empty_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a Stream End Point no longer has any data queued to be sent on the Media Data Channel.

etGAVD_Report_Data_Channel_Empty_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a Stream End Point no longer has any data queued to be sent on the Report Data Channel.
etGAVD_Recovery_Data_Channel_Empty_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a Stream End Point no longer has any data queued to be sent on the Recovery Data Channel.
etGAVD_Multiplexed_Channel_Empty_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a Stream End Point no longer has any data queued to be sent on a Multiplexed Data Channel.
etGAVD_Signalling_Connect_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a GAVD/AVDTP signaling channel is connected.
etGAVD_Signalling_Disconnect_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a GAVD/AVDTP signaling channel is no longer connected.
etGAVD_Signalling_Channel_Idle_Indication	Dispatched by a GAVD/AVDTP entity to the local application when the GAVD/AVDTP signaling channel becomes idle (i.e. no endpoints are open).
etGAVD_Signalling_Channel_Endpoint_Open_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint is opened to/from a remote device.
etGAVD_Signalling_Channel_Endpoint_Close_Indication	Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint is no longer open to/from a remote device.
etGAVD_Delay_Report_Indication	Dispatched by GAVD/AVDTP when a delay report is received from a remote endpoint.
etGAVD_Delay_Report_Confirmation	Dispatched by GAVD/AVDTP when a remote delay report response is received from a remote endpoint.
etGAVD_General_Reject_Indication	Dispatched by GAVD/AVDTP when an AVDTP command message is rejected by the remote server.

### **etGAVD\_Connect\_Request\_Indication**

This event is dispatched when a remote device is requesting a connection to the local device. This event is only dispatched to servers in Manual Accept Mode. This event must be responded to with a call to `GAVD_Connect_Request_Response()` in order to accept or reject the pending connection request.

**Return Structure:**

```
typedef struct
{
    BD_ADDR_t    BD_ADDR;
} GAVD_Connect_Request_Indication_Data_t;
```

**Event Parameters:**

BD_ADDR	Bluetooth Address of the Remote Device that is attempting to connect.
---------	-----------------------------------------------------------------------

**etGAVD\_Connect\_Confirmation**

Confirms that a Connect request has been responded to or has encountered an error.

**Return Structure:**

```
typedef struct
{
    unsigned int  GAVDID;
    unsigned int  Status;
} GAVD_Connect_Confirmation_Data_t;
```

**Event Parameters:**

GAVDID	Identifier of the local GAVD client.
Status	Success or failure error code. This is one of the following values: GAVD_STATUS_SUCCESS GAVD_STATUS_CONNECTION_TIMEOUT GAVD_STATUS_CONNECTION_REFUSED GAVD_STATUS_UNKNOWN_ERROR

**etGAVD\_Disconnect\_Indication**

Indicates that a local client has been disconnected from a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    unsigned int  GAVDID;
    BD_ADDR_t    BD_ADDR;
} GAVD_Disconnect_Indication_Data_t;
```

**Event Parameters:**

GAVDID	Identifier of the local GAVD client.
BD_ADDR	Bluetooth Device address of the remote device that was disconnected.

**etGAVD\_Discover\_Confirmation**

Confirms the completion of a Stream Discover command that was sent to a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    unsigned int          GAVDID;
    unsigned int          ErrorCode;
    unsigned int          NumberRemoteEndpoints;
    GAVD_Remote_End_Point_Data_t *RemoteEndpoints;
} GAVD_Discover_Confirmation_Data_t;
```

**Event Parameters:**

GAVDID	Identifier of the local GAVD client.
ErrorCode	Error response code of the discover confirmation. This value is one of the following values: GAVD_AVDTP_ERROR_SUCCESS GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT GAVD_AVDTP_ERROR_BAD_LENGTH GAVD_AVDTP_ERROR_BAD_ACP_SEID GAVD_AVDTP_ERROR_SEP_IN_USE GAVD_AVDTP_ERROR_SEP_NOT_IN_USE GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND GAVD_AVDTP_ERROR_INVALID_CAPABILITIES GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT_FORMAT GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT GAVD_AVDTP_ERROR_BAD_CP_FORMAT GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT GAVD_AVDTP_ERROR_UNSUPPORTED_CONFIGURATION GAVD_AVDTP_ERROR_BAD_STATE GAVD_AVDTP_ERROR_TIMEOUT
NumberRemoteEndpoints	Number of remote endpoints that are contained in the array pointed to by the RemoteEndpoints parameter.
RemoteEndpoints	Pointer to an array of stream endpoints that are available on the remote GAVD device. This is defined by the following structure:

```
typedef struct
{
    unsigned int          RSEID;
    GAVD_TSEP_t          TSEP;
    GAVD_Media_Type_t    MediaType;
    Boolean_t            InUse;
} GAVD_Remote_End_Point_Data_t;
```

where;

RSEID is the identifier of the remote stream endpoint.

TSEP specifies the type (source or sink) of the stream endpoint.

MediaType specifies the media type of the stream endpoint, using the following enumeration:

```
typedef enum
{
    mtAudio,
    mtVideo,
    mtMultimedia
} GAVD_Media_Type_t;
```

InUse indicates whether the stream is currently in use or not.

### etGAVD\_Get\_Capabilities\_Confirmation

Confirms the completion of a Get Capabilities command that was sent to a remote Stream Endpoint Manager.

#### Return Structure:

```
typedef struct
{
    unsigned int          GAVDID;
    unsigned int          RSEID;
    unsigned int          ErrorCode;
    unsigned int          NumberServiceCapabilities;
    GAVD_Service_Capabilities_Info_t *ServiceCapabilities;
} GAVD_Get_Capabilities_Confirmation_Data_t;
```

#### Event Parameters:

GAVDID	Identifier of the local GAVD client.
RSEID	Identifier of the remote stream endpoint.
ErrorCode	Error response code of the get capabilities response. This value is one of the following values:

```
GAVD_AVDTP_ERROR_SUCCESS
GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT
GAVD_AVDTP_ERROR_BAD_LENGTH
GAVD_AVDTP_ERROR_BAD_ACP_SEID
GAVD_AVDTP_ERROR_SEP_IN_USE
GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT_FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
```

```

GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

**NumberServiceCapabilities** Number of service capabilities that are contained in the array pointed to by the ServiceCapabilities parameter.

**ServiceCapabilities** Pointer to an array of service capabilities. This is defined by the following structure:

```

typedef struct
{
    GAVD_Service_Category_t ServiceCategory;
    union
    {
        GAVD_Recovery_Info_Element_Data_t
            GAVD_Recovery_Info_Element_Data;
        GAVD_Content_Protection_Info_Element_Data_t
            GAVD_Content_Protection_Info_Element_Data;
        GAVD_Header_Compression_Info_Element_Data_t
            GAVD_Header_Compression_Info_Element_Data;
        GAVD_Multiplexing_Info_Element_Data_t
            GAVD_Multiplexing_Info_Element_Data;
        GAVD_Media_Codec_Info_Element_Data_t
            GAVD_Media_Codec_Info_Element_Data;
        GAVD_Raw_Info_Element_Data_t
            GAVD_Raw_Info_Element_Data;
    } InfoElement;
} GAVD_Service_Capabilities_Info_t;

```

## etGAVD\_Get\_Configuration\_Confirmation

Confirms the completion of a Get Configuration command.

### Return Structure:

```

typedef struct
{
    unsigned int          GAVDID;
    unsigned int          RSEID;
    unsigned int          ErrorCode;
    unsigned int          NumberServiceCapabilities;
    GAVD_Service_Capabilities_Info_t *ServiceCapabilities;
} GAVD_Get_Configuration_Confirmation_Data_t;

```

### Event Parameters:

**GAVDID** Identifier of the local GAVD client.

**RSEID** Identifier of the remote stream endpoint.



ErrorCode	<p>Error response code of the get configuration response. This value is one of the following values:</p> <p> GAVD_AVDTP_ERROR_SUCCESS  GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT  GAVD_AVDTP_ERROR_BAD_LENGTH  GAVD_AVDTP_ERROR_BAD_ACP_SEID  GAVD_AVDTP_ERROR_SEP_IN_USE  GAVD_AVDTP_ERROR_SEP_NOT_IN_USE  GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY  GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT  GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND  GAVD_AVDTP_ERROR_INVALID_CAPABILITIES  GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE  GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT_FORMAT  GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT  GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT  GAVD_AVDTP_ERROR_BAD_CP_FORMAT  GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT  GAVD_AVDTP_ERROR_UNSUPPORTED_CONFIGURATION  GAVD_AVDTP_ERROR_BAD_STATE  GAVD_AVDTP_ERROR_TIMEOUT </p>
NumberServiceCapabilities	Number of service capabilities that are contained in the array pointed to by the ServiceCapabilities parameter.
ServiceCapabilities	<p>Pointer to an array of service capabilities. This is defined by the following structure:</p> <pre> typedef struct {     GAVD_Service_Category_t ServiceCategory;     union     {         GAVD_Recovery_Info_Element_Data_t             GAVD_Recovery_Info_Element_Data;         GAVD_Content_Protection_Info_Element_Data_t             GAVD_Content_Protection_Info_Element_Data;         GAVD_Header_Compression_Info_Element_Data_t             GAVD_Header_Compression_Info_Element_Data;         GAVD_Multiplexing_Info_Element_Data_t             GAVD_Multiplexing_Info_Element_Data;         GAVD_Media_Codec_Info_Element_Data_t             GAVD_Media_Codec_Info_Element_Data;         GAVD_Raw_Info_Element_Data_t             GAVD_Raw_Info_Element_Data;     } InfoElement; } GAVD_Service_Capabilities_Info_t; </pre>

**etGAVD\_Set\_Configuration\_Indication**

Indicates that a Set Configuration command has been received from a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    BD_ADDR_t          BD_ADDR;
    unsigned int        LSEID;
    unsigned int        NumberServiceCapabilities;
    GAVD_Service_Capabilities_Info_t *ServiceCapabilities;
    unsigned int        RSEID;
} GAVD_Set_Configuration_Indication_Data_t;
```

**Event Parameters:**

BD_ADDR	Bluetooth address of the remote device.
LSEID	Identifier of the local stream endpoint.
NumberServiceCapabilities	Number of service capabilities that are contained in the array pointed to by the ServiceCapabilities parameter.
ServiceCapabilities	Pointer to an array of service capabilities. This is defined by the following structure:

```
typedef struct
{
    GAVD_Service_Category_t ServiceCategory;
    union
    {
        GAVD_Recovery_Info_Element_Data_t
            GAVD_Recovery_Info_Element_Data;
        GAVD_Content_Protection_Info_Element_Data_t
            GAVD_Content_Protection_Info_Element_Data;
        GAVD_Header_Compression_Info_Element_Data_t
            GAVD_Header_Compression_Info_Element_Data;
        GAVD_Multiplexing_Info_Element_Data_t
            GAVD_Multiplexing_Info_Element_Data;
        GAVD_Media_Codec_Info_Element_Data_t
            GAVD_Media_Codec_Info_Element_Data;
        GAVD_Raw_Info_Element_Data_t
            GAVD_Raw_Info_Element_Data;
    } InfoElement;
} GAVD_Service_Capabilities_Info_t;
```

RSEID	Identifier of the remote stream endpoint.
-------	-------------------------------------------

**etGAVD\_Open\_End\_Point\_Indication**

Indicates that an Open Endpoint command has been received from a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    unsigned int    LSEID;
    Word_t          MediaOutMTU;
    Word_t          ReportingOutMTU;
    Word_t          RecoveryOutMTU;
} GAVD_Open_End_Point_Indication_Data_t;
```

**Event Parameters:**

LSEID	Identifier of the local stream endpoint.
MediaOutMTU	Maximum output MTU size for data payload. This is the largest negotiated MTU size for the data channel.
ReportingOutMTU	Maximum output MTU size for reporting data payload. This is the largest negotiated MTU size for the reporting channel.
RecoveryOutMTU	Maximum output MTU size for recovery data payload. This is the largest negotiated MTU size for the recovery channel.

**etGAVD\_Open\_End\_Point\_Confirmation**

Confirms the completion of an Open Endpoint command that was sent to a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    unsigned int    LSEID;
    Word_t          MediaOutMTU;
    Word_t          ReportingOutMTU;
    Word_t          RecoveryOutMTU;
    unsigned int    ErrorCode;
    GAVD_Service_Category_t FirstFailingServiceCategory;
} GAVD_Open_End_Point_Confirmation_Data_t;
```

**Event Parameters:**

LSEID	Identifier of the local stream endpoint.
MediaOutMTU	Maximum output MTU size for data channel. This value will be used for the negotiation of the MTU of the data channel.
ReportingOutMTU	Maximum output MTU size for reporting data channel. This value will be used for the negotiation of the MTU of the reporting channel.
RecoveryOutMTU	Maximum output MTU size for recovery data channel. This value will be used for the negotiation of the MTU of the recovery channel.
ErrorCode	Error response code of the configuration confirmation. This value is one of the following values: GAVD_AVDTP_ERROR_SUCCESS

```

GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT
GAVD_AVDTP_ERROR_BAD_LENGTH
GAVD_AVDTP_ERROR_BAD_ACP_SEID
GAVD_AVDTP_ERROR_SEP_IN_USE
GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

**FirstFailingServiceCategory** This represents the first service category that was requested to be configured that was unable to be configured (scNone if configuration was successful). Defined by the following enumeration:

```

typedef enum
{
    scNone,
    scMediaTransport,
    scReporting,
    scRecovery,
    scContentProtection,
    scHeaderCompression,
    scMultiplexing,
    scMediaCodec,
    scUnknown
} GAVD_Service_Category_t;

```

## etGAVD\_Close\_End\_Point\_Indication

Indicates that a Close Endpoint command has been received from a remote Stream Endpoint Manager.

### Return Structure:

```

typedef struct
{
    unsigned int  LSEID;
} GAVD_Close_End_Point_Indication_Data_t;

```

### Event Parameters:

**LSEID** Identifier of the local stream endpoint.

**etGAVD\_Start\_Indication**

Indicates that a Start Stream command has been received from a remote Stream Endpoint Manager.

**Return Structure:**

```
typedef struct
{
    unsigned int  LSEID;
} GAVD_Start_Indication_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

**etGAVD\_Start\_Confirmation**

Confirms that a Start Stream command sent to a remote Stream Endpoint Manager was received.

**Return Structure:**

```
typedef struct
{
    unsigned int  LSEID;
    unsigned int  ErrorCode;
    unsigned int  FirstFailingLSEID;
} GAVD_Start_Confirmation_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

ErrorCode                                  Error response code of the start stream endpoint confirmation. This value is one of the following values:

```
GAVD_AVDTP_ERROR_SUCCESS
GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT
GAVD_AVDTP_ERROR_BAD_LENGTH
GAVD_AVDTP_ERROR_BAD_ACP_SEID
GAVD_AVDTP_ERROR_SEP_IN_USE
GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
```

GAVD\_AVDTP\_ERROR\_UNSUPPORTED  
 \_CONFIGURATION  
 GAVD\_AVDTP\_ERROR\_BAD\_STATE  
 GAVD\_AVDTP\_ERROR\_TIMEOUT

FirstFailingLSEID                      Identifier of the first local stream endpoint that failed to start.

### etGAVD\_Suspend\_Indication

Indicates that a Suspend Stream command has been received from a remote Stream Endpoint Manager.

#### Return Structure:

```
typedef struct
{
    unsigned int  LSEID;
} GAVD_Suspend_Indication_Data_t;
```

#### Event Parameters:

LSEID                                      Identifier of the local stream endpoint.

### etGAVD\_Suspend\_Confirmation

Confirms that a Suspend Stream command sent to a remote Stream Endpoint was received.

#### Return Structure:

```
typedef struct
{
    unsigned int  LSEID;
    unsigned int  ErrorCode;
    unsigned int  FirstFailingLSEID;
} GAVD_Suspend_Confirmation_Data_t;
```

#### Event Parameters:

LSEID                                      Identifier of the local stream endpoint.

ErrorCode                                  Error response code of the start stream endpoint confirmation. This value is one of the following values:

GAVD\_AVDTP\_ERROR\_SUCCESS  
 GAVD\_AVDTP\_ERROR\_BAD\_HEADER\_FORMAT  
 GAVD\_AVDTP\_ERROR\_BAD\_LENGTH  
 GAVD\_AVDTP\_ERROR\_BAD\_ACP\_SEID  
 GAVD\_AVDTP\_ERROR\_SEP\_IN\_USE  
 GAVD\_AVDTP\_ERROR\_SEP\_NOT\_IN\_USE  
 GAVD\_AVDTP\_ERROR\_BAD\_SERV\_CATEGORY  
 GAVD\_AVDTP\_ERROR\_BAD\_PAYLOAD\_FORMAT  
 GAVD\_AVDTP\_ERROR\_NOT\_SUPPORTED\_COMMAND  
 GAVD\_AVDTP\_ERROR\_INVALID\_CAPABILITIES  
 GAVD\_AVDTP\_ERROR\_BAD\_RECOVERY\_TYPE

```

GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

FirstFailingLSEID      Identifier of the first local stream endpoint that failed to suspend.

## etGAVD\_Reconfigure\_Indication

Indicates that a Reconfigure Stream command has been received from a remote Stream Endpoint Manager.

### Return Structure:

```

typedef struct
{
    unsigned int          LSEID;
    unsigned int          NumberServiceCapabilities;
    GAVD_Service_Capabilities_Info_t *ServiceCapabilities;
} GAVD_Reconfigure_Indication_Data_t;

```

### Event Parameters:

LSEID      Identifier of the local stream endpoint.

NumberServiceCapabilities      Number of service capabilities that are contained in the array pointed to by the ServiceCapabilities parameter.

ServiceCapabilities      Pointer to an array of service capabilities. This is defined by the following structure:

```

typedef struct
{
    GAVD_Service_Category_t ServiceCategory;
    union
    {
        GAVD_Recovery_Info_Element_Data_t
            GAVD_Recovery_Info_Element_Data;
        GAVD_Content_Protection_Info_Element_Data_t
            GAVD_Content_Protection_Info_Element_Data;
        GAVD_Header_Compression_Info_Element_Data_t
            GAVD_Header_Compression_Info_Element_Data;
        GAVD_Multiplexing_Info_Element_Data_t
            GAVD_Multiplexing_Info_Element_Data;
        GAVD_Media_Codec_Info_Element_Data_t
            GAVD_Media_Codec_Info_Element_Data;
        GAVD_Raw_Info_Element_Data_t
            GAVD_Raw_Info_Element_Data;
    }
}

```

```

    } InfoElement;
} GAVD_Service_Capabilities_Info_t;

```

## etGAVD\_Reconfigure\_Confirmation

Confirms the completion of a Reconfigure Stream command that was sent to a remote Stream Endpoint Manager.

### Return Structure:

```

typedef struct
{
    unsigned int          LSEID;
    unsigned int          ErrorCode;
    GAVD_Service_Category_t FirstFailingServiceCategory;
} GAVD_Reconfigure_Confirmation_Data_t;

```

### Event Parameters:

LSEID	Identifier of the local stream endpoint.
ErrorCode	Error response code of the reconfiguration confirmation. This value is one of the following values:

```

GAVD_AVDTP_ERROR_SUCCESS
GAVD_AVDTP_ERROR_BAD_HEADER_FORMAT
GAVD_AVDTP_ERROR_BAD_LENGTH
GAVD_AVDTP_ERROR_BAD_ACP_SEID
GAVD_AVDTP_ERROR_SEP_IN_USE
GAVD_AVDTP_ERROR_SEP_NOT_IN_USE
GAVD_AVDTP_ERROR_BAD_SERV_CATEGORY
GAVD_AVDTP_ERROR_BAD_PAYLOAD_FORMAT
GAVD_AVDTP_ERROR_NOT_SUPPORTED_COMMAND
GAVD_AVDTP_ERROR_INVALID_CAPABILITIES
GAVD_AVDTP_ERROR_BAD_RECOVERY_TYPE
GAVD_AVDTP_ERROR_BAD_MEDIA_TRANSPORT
    _FORMAT
GAVD_AVDTP_ERROR_BAD_RECOVERY_FORMAT
GAVD_AVDTP_ERROR_BAD_ROHC_FORMAT
GAVD_AVDTP_ERROR_BAD_CP_FORMAT
GAVD_AVDTP_ERROR_BAD_MULTIPLEXING_FORMAT
GAVD_AVDTP_ERROR_UNSUPPORTED
    _CONFIGURATION
GAVD_AVDTP_ERROR_BAD_STATE
GAVD_AVDTP_ERROR_TIMEOUT

```

FirstFailingServiceCategory	This represents the first service category that was requested to be reconfigured that was unable to be reconfigured (scNone if reconfiguration was successful). Defined by the following enumeration:
-----------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

```

typedef enum
{
    scNone,

```



```
    scMediaTransport,  
    scReporting,  
    scRecovery,  
    scContentProtection,  
    scHeaderCompression,  
    scMultiplexing,  
    scMediaCodec,  
    scUnknown  
} GAVD_Service_Category_t;
```

## **etGAVD\_Security\_Control\_Indication**

Indicates that a Security Control command has been received from a remote Stream Endpoint Manager.

### **Return Structure:**

```
typedef struct  
{  
    unsigned int    LSEID;  
    unsigned int    SecurityDataLength;  
    Byte_t          *SecurityData;  
} GAVD_Security_Control_Indication_Data_t;
```

### **Event Parameters:**

LSEID	Identifier of the local stream endpoint.
SecurityDataLength	Length of the security data. This value specifies the size (in bytes) of the Security Data information that is pointed to by the SecurityData member.
SecurityData	Pointer to the security control data.

## **etGAVD\_Security\_Control\_Confirmation**

Confirms the completion of a Security Control command that was sent to a remote Stream Endpoint Manager.

### **Return Structure:**

```
typedef struct  
{  
    unsigned int    LSEID;  
    unsigned int    ErrorCode;  
    unsigned int    SecurityDataLength;  
    Byte_t          *SecurityData;  
} GAVD_Security_Control_Confirmation_Data_t;
```

### **Event Parameters:**

LSEID	Identifier of the local stream endpoint.
ErrorCode	Status of the stream suspend attempt.

SecurityDataLength	Length of the security data. This value specifies the size (in bytes) of the Security Control Data information that is pointed to by the SecurityData member.
SecurityData	Pointer to the security control data.

### **etGAVD\_Abort\_Indication**

Indicates that an Abort Stream command has been received from a remote Stream Endpoint Manager.

#### **Return Structure:**

```
typedef struct
{
    unsigned int  LSEID;
} GAVD_Abort_Indication_Data_t;
```

#### **Event Parameters:**

LSEID	Identifier of the local stream endpoint.
-------	------------------------------------------

### **etGAVD\_Abort\_Confirmation**

Confirms the completion of an Abort Stream command that was sent to a remote Stream Endpoint Manager.

#### **Return Structure:**

```
typedef struct
{
    unsigned int  LSEID;
} GAVD_Abort_Confirmation_Data_t;
```

#### **Event Parameters:**

LSEID	Identifier of the local stream endpoint.
-------	------------------------------------------

### **etGAVD\_Data\_Indication**

Indicates that data has been received from a remote Endpoint.

**Return Structure:**

```
typedef struct
{
    unsigned int    LSEID;
    DWord_t        TimeStamp;
    Byte_t          PayloadType;
    Boolean_t        Marker;
    Word_t          SequenceNumber;
    unsigned int    DataLength;
    Byte_t          *DataBuffer;
} GAVD_Data_Indication_Data_t;
```

**Event Parameters:**

LSEID	Identifier of the local stream endpoint.
TimeStamp	Holds time stamp information.
PayloadType	Indicates the content type of the data.
Marker	Specifies whether the marker bit (used by certain codec payload formats) was specified or not.
SequenceNumber	Specifies the sequence number of the received data.
DataLength	Specifies the length of the received data. This value represents the size (in bytes) of the data that is pointed to by the DataBuffer member.
DataBuffer	Points to the received data.

**etGAVD\_Sender\_Report\_Data\_Indication**

Indicates that Sender Report data has been received from a remote Endpoint.

**Return Structure:**

```
typedef struct
{
    unsigned int    LSEID;
    GAVD_Sender_Info_t *SenderInfo;
    unsigned int    NumberReportBlocks;
    GAVD_Report_Block_t *ReportBlocks;
    unsigned int    ExtensionDataLength;
    DWord_t        *ExtensionData;
} GAVD_Sender_Report_Data_Indication_Data_t;
```

**Event Parameters:**

LSEID	Identifier of the local stream endpoint.
SenderInfo	Contains the RTCP Sender Information. This is defined by the following structure:

```
typedef struct
{
```

	<pre> DWord_t    NTPTimestampMSW; DWord_t    NTPTimestampLSW; DWord_t    RTPTimestamp; DWord_t    PacketCount; DWord_t    OctetCount; } GAVD_Sender_Info_t; </pre>
NumberReportBlocks	Indicates the number of report blocks pointed to by the ReportBlocks parameter.
ReportBlocks	Points to the report blocks. This is defined by the following structure: <pre> typedef struct {     DWord_t    SourceID;     Byte_t     FractionLost;     DWord_t    TotalPacketsLost;     Word_t     HighestSeqNumReceivedCycleCount;     Word_t     HighestSeqNumReceived;     DWord_t    IntervalJitter;     DWord_t    LastReport;     DWord_t    DelaySinceLastReport; } GAVD_Report_Block_t; </pre>
ExtensionDataLength	Indicates how many 32-bit words are in the extension data.
ExtensionData	Points to the extension data. This member points to an array of 32-bit words. The number of 32-bit words pointed to by this member is given by the ExtensionDataLength member.

### etGAVD\_Receiver\_Report\_Data\_Indication

Indicates that Receiver Report data has been received from a remote Endpoint.

#### Return Structure:

```

typedef struct
{
    unsigned int    LSEID;
    unsigned int    NumberReportBlocks;
    GAVD_Report_Block_t *ReportBlocks;
    unsigned int    ExtensionDataLength;
    DWord_t         *ExtensionData;
} GAVD_Receiver_Report_Data_Indication_Data_t;

```

#### Event Parameters:

LSEID	Identifier of the local stream endpoint.
NumberReportBlocks	Indicates the number of report blocks pointed to by the ReportBlocks parameter.
ReportBlocks	Points to the report blocks. This is defined by the following structure: <pre> typedef struct { </pre>

	DWord_t	SourceID;
	Byte_t	FractionLost;
	DWord_t	TotalPacketsLost;
	Word_t	HighestSeqNumReceivedCycleCount;
	Word_t	HighestSeqNumReceived;
	DWord_t	IntervalJitter;
	DWord_t	LastReport;
	DWord_t	DelaySinceLastReport;
	} GAVD_Report_Block_t;	
ExtensionDataLength	Indicates how many 32-bit words are in the extension data.	
ExtensionData	Points to the extension data. This member points to an array of 32-bit words. The number of 32-bit words pointed to by this member is given by the ExtensionDataLength member.	

### etGAVD\_SDES\_Report\_Data\_Indication

Indicates that SDES Report data has been received from a remote Endpoint.

#### Return Structure:

```
typedef struct
{
    unsigned int          LSEID;
    unsigned int          NumberSDESChunks;
    GAVD_SDES_Chunk_t    *SDESChunks;
} GAVD_SDES_Report_Data_Indication_Data_t;
```

#### Event Parameters:

LSEID	Identifier of the local stream endpoint.
NumberSDESChunks	Indicates the number of SDES chunks pointed to by the SDESChunks parameter.
SDESChunks	Points to the SDES chunks. This is defined by the following structure:

```
typedef struct
{
    DWord_t          SourceID;
    unsigned int      NumberSDESItems;
    GAVD_SDES_Item_t *SDESItems;
} GAVD_SDES_Chunk_t;
```

where, SDESItems is defined by the following structure:

```
typedef struct
{
    Byte_t    ItemDescriptor;
    Byte_t    ItemLength;
    Byte_t    *ItemData;
} GAVD_SDES_Item_t;
```

**etGAVD\_Recovery\_Data\_Indication**

Indicates that Recovery data has been received from a remote Endpoint.

**Return Structure:**

```
typedef struct
{
    unsigned int      LSEID;
    DWord_t          TimeStamp;
    Byte_t            PayloadType;
    Boolean_t          Marker;
    Word_t             SequenceNumber;
    GAVD_FEC_Block_t  *FECBlock;
} GAVD_Recovery_Data_Indication_Data_t;
```

**Event Parameters:**

LSEID	Identifier of the local stream endpoint.
TimeStamp	Holds time stamp information.
PayloadType	Indicates the content type of the data.
Marker	Specifies the marker bit used by certain codec payload formats.
SequenceNumber	Specifies the sequence number of the received data.
FECBlock	Points to the structure with the information to send. This is defined by the following structure:

```
typedef struct
{
    Word_t      SequenceNumberBase;
    Word_t      LengthRecovery;
    Byte_t      PayloadTypeRecovery;
    DWord_t     Mask;
    DWord_t     TimeStampRecovery;
    unsigned int FECDataLength;
    Byte_t      *FECData;
} GAVD_FEC_Block_t;
```

**etGAVD\_Data\_Channel\_Empty\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint no longer has any data queued to be sent on the Media Data Channel. This event is only dispatched when the GAVD\_Data\_Write() function returns the error code BTPS\_ERROR\_INSUFFICIENT\_RESOURCES.

**Return Structure:**

```
typedef struct
{
    unsigned int          LSEID;
} GAVD_Channel_Empty_Indication_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

**etGAVD\_Report\_Data\_Channel\_Empty\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint no longer has any data queued to be sent on the Reporter Data Channel. This event is only dispatched when either the GAVD\_Sender\_Report\_Data\_Write () or GAVD\_Receiver\_Report\_Data\_Write() functions returns the error code BTPS\_ERROR\_INSUFFICIENT\_RESOURCES.

**Return Structure:**

```
typedef struct
{
    unsigned int          LSEID;
} GAVD_Channel_Empty_Indication_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

**etGAVD\_Recovery\_Data\_Channel\_Empty\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint no longer has any data queued to be sent on the Recovery Data Channel. This event is only dispatched when the GAVD\_Recovery\_Data\_Write () function returns the error code BTPS\_ERROR\_INSUFFICIENT\_RESOURCES.

**Return Structure:**

```
typedef struct
{
    unsigned int          LSEID;
} GAVD_Channel_Empty_Indication_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

**etGAVD\_Multiplexed\_Channel\_Empty\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a stream endpoint no longer has any data queued to be sent on any of the multiplexed data channels. This event is only dispatched when one of the data write functions returns the error code BTPS\_ERROR\_INSUFFICIENT\_RESOURCES.

**Return Structure:**

```
typedef struct
{
    unsigned int          LSEID;
} GAVD_Channel_Empty_Indication_Data_t;
```

**Event Parameters:**

LSEID                                      Identifier of the local stream endpoint.

**etGAVD\_Signalling\_Connect\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a new GAVD/AVDTP signaling channel is connected (either initiated remotely or locally). This event is only dispatched to the event callback that was registered via the GAVD\_Register\_Signalling\_Connection\_Status() function.

**Return Structure:**

```
typedef struct
{
    unsigned int          GAVDConnectionStatusID;
    BD_ADDR_t            BD_ADDR;
} GAVD_Signalling_Connect_Indication_Data_t;
```

**Event Parameters:**

GAVDConnectionStatusID    Identifier of the GAVD/AVDTP connection status event callback. This is the same value that was returned from the GAVD\_Register\_Signalling\_Connection\_Status() function.

BD\_ADDR                    Bluetooth device address of the device that the signaling status is valid for.

**etGAVD\_Signalling\_Disconnect\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when a connected GAVD/AVDTP signaling channel is disconnected (either initiated remotely or locally). This event is only dispatched to the event callback that was registered via the GAVD\_Register\_Signalling\_Connection\_Status() function.



**Return Structure:**

```
typedef struct
{
    unsigned int    GAVDConnectionStatusID;
    BD_ADDR_t      BD_ADDR;
    Byte_t          Reason;
} GAVD_Signalling_Disconnect_Indication_Data_t;
```

**Event Parameters:**

GAVDConnectionStatusID	Identifier of the GAVD/AVDTP connection status event callback. This is the same value that was returned from the GAVD_Register_Signalling_Connection_Status() function.
BD_ADDR	Bluetooth device address of the device that the signaling status is valid for.
Reason	HCI disconnect reason for the disconnection. Please see the HCI error codes for possible values for this parameter.

**etGAVD\_Signalling\_Channel\_Idle\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application when there are no longer any open stream endpoints. This event is used in conjunction with the GAVD\_Close\_End\_Point() function to denote that the stream endpoint is fully closed. This event is only dispatched to the event callback that was registered via the GAVD\_Register\_Signalling\_Connection\_Status() function.

**Return Structure:**

```
typedef struct
{
    unsigned int    GAVDConnectionStatusID;
    BD_ADDR_t      BD_ADDR;
} GAVD_Signalling_Channel_Idle_Indication_Data_t;
```

**Event Parameters:**

GAVDConnectionStatusID	Identifier of the GAVD/AVDTP connection status event callback. This is the same value that was returned from the GAVD_Register_Signalling_Connection_Status() function.
BD_ADDR	Bluetooth device address of the device that has an idle signaling channel.

**etGAVD\_Signalling\_Channel\_Endpoint\_Open\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application whenever a new stream endpoint is opened. This event is dispatched regardless if the local or remote device initiated the open stream endpoint procedure. This event is only dispatched to the event callback that was registered via the GAVD\_Register\_Signalling\_Connection\_Status() function.

**Return Structure:**

```
typedef struct
{
    unsigned int    GAVDConnectionStatusID;
    BD_ADDR_t      BD_ADDR;
    unsigned int    LSEID;
    unsigned int    RSEID;
} GAVD_Signalling_Channel_Endpoint_Open_Indication_Data_t;
```

**Event Parameters:**

GAVDConnectionStatusID	Identifier of the GAVD/AVDTP connection status event callback. This is the same value that was returned from the GAVD_Register_Signalling_Connection_Status() function.
BD_ADDR	Bluetooth device address of the device that now has the specified opened stream endpoint.
LSEID	Local stream endpoint identifier of the newly opened endpoint.
RSEID	Remote stream endpoint identifier of the newly opened endpoint.

**etGAVD\_Signalling\_Channel\_Endpoint\_Close\_Indication**

Dispatched by a GAVD/AVDTP entity to the local application whenever an currently open stream endpoint is closed. This event is dispatched regardless if the local or remote device initiated the close stream endpoint procedure. This event is only dispatched to the event callback that was registered via the GAVD\_Register\_Signalling\_Connection\_Status() function.

**Return Structure:**

```
typedef struct
{
    unsigned int    GAVDConnectionStatusID;
    BD_ADDR_t      BD_ADDR;
    unsigned int    LSEID;
    unsigned int    RSEID;
} GAVD_Signalling_Channel_Endpoint_Open_Indication_Data_t;
```

**Event Parameters:**

GAVDConnectionStatusID	Identifier of the GAVD/AVDTP connection status event callback. This is the same value that was returned from the GAVD_Register_Signalling_Connection_Status() function.
BD_ADDR	Bluetooth device address of the device that now has the specified closed stream endpoint.
LSEID	Local stream endpoint identifier of the closed endpoint.
RSEID	Remote stream endpoint identifier of the closed endpoint.

## etGAVD\_Delay\_Report\_Indication

Dispatched by GAVD/AVDTP to the local application when a remote endpoint sends a delay report. This event is always dispatched from a remote endpoint operating in the SNK role to a local endpoint operating in the SRC role.

### Return Structure:

```
typedef struct
{
    BD_ADDR_t    BD_ADDR;
    unsigned int LSEID;
    Word_t       Delay;
} GAVD_Delay_Report_Indication_Data_t;
```

### Event Parameters:

BD_ADDR	Bluetooth device address of the device with the SNK role that sent the delay report.
LSEID	Local stream endpoint identifier of the endpoint in the SRC role receiving the delay report.
Delay	Delay report value in units of 100 microseconds.

## etGAVD\_Delay\_Report\_Confirmation

Dispatched by GAVD/AVDTP to the local application on receipt of a delay report by a remote endpoint. This event is always dispatched to the SNK role sending the delay report when an updated delay report is received by a remote endpoint in the SRC role.

### Return Structure:

```
typedef struct
{
    BD_ADDR_t    BD_ADDR;
    unsigned int LSEID;
    int           ErrorCode;
} GAVD_Delay_Report_Confirmation_Data_t;
```

### Event Parameters:

BD_ADDR	Bluetooth device address of the device with the SNK role that sent the delay report.
LSEID	Local stream endpoint identifier of the endpoint in the SRC role receiving the delay report.
ErrorCode	0 if the delay report was received by the remote endpoint, an error code otherwise. See AVDTP_SPEC_V13 8.20.6.2 for possible error codes.

**etGAVD\_General\_Reject\_Indication**

Dispatched by GAVD/AVDTP to the local application on receipt general reject response from the remote GAVD server or endpoint.

**Return Structure:**

```
typedef struct _tagGAVD_General_Reject_Indication_Data_t
{
    unsigned int LSEID;
    BD_ADDR_t    RemoteBD_ADDR;
    Byte_t       MessageID;
} GAVD_General_Reject_Indication_Data_t;
```

**Event Parameters:**

LSEID	Local stream endpoint identifier of the endpoint receiving the indication of a general-reject response.
RemoteBD_ADDR	Bluetooth device address of the remote device that sent the general reject response. General reject may occur before or during endpoint establishment, so no RSEID may be defined.
MessageID	MessageID of the message rejected by the remote GAVD/AVDTP server.

### 3. File Distributions

The header files that are distributed with the Bluetooth GAVD Profile Library are listed in the table below.

File	Contents/Description
GAVDAPI.h	Bluetooth GAVD Profile API definitions
SS1BTGAV.h	Bluetooth GAVD Profile Include file
GAVDType.h	Bluetooth GAVD Profile type definitions