NAME

ng_l2cap - Netgraph node type that implements Bluetooth Logical Link Control and Adaptation Protocol (L2CAP)

SYNOPSIS

#include <sys/types.h>
#include <netgraph/bluetooth/include/ng_hci.h>
#include <netgraph/bluetooth/include/ng_l2cap.h>

DESCRIPTION

The **l2cap** node type is a Netgraph node type that implements Bluetooth Logical Link Control and Adaptation Protocol as per chapter D of the Bluetooth Specification Book v1.1.

L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions. L2CAP permits higher level protocols and applications to transmit and receive L2CAP data packets up to 64 kilobytes in length.

L2CAP Assumptions

1. The ACL link between two units is set up. The Baseband provides orderly delivery of data packets, although there might be individual packet corruption and duplicates. No more than one ACL link exists between any two devices.

2. The Baseband always provides the impression of full-duplex communication channels. This does not imply that all L2CAP communications are bi-directional. Multicasts and unidirectional traffic (e.g., video) do not require duplex channels.

3. L2CAP provides a reliable channel using the mechanisms available at the Baseband layer. The Baseband always performs data integrity checks when requested and resends data until it has been successfully acknowledged or a timeout occurs. As acknowledgements may be lost, timeouts may occur even after the data has been successfully sent.

L2CAP GENERAL OPERATION

The Logical Link Control and Adaptation Protocol (L2CAP) is based around the concept of "channels". Each channel is bound to a single protocol in a many-to-one fashion. Multiple channels can be bound to the same protocol, but a channel cannot be bound to multiple protocols. Each L2CAP packet received on a channel is directed to the appropriate higher level protocol.

Each one of the end-points of an L2CAP channel is referred to by a channel identifier. Channel identifiers (CIDs) are local names representing a logical channel end-point on the device. Identifiers

from 0x0001 to 0x003F are reserved for specific L2CAP functions. The null identifier (0x0000) is defined as an illegal identifier and must never be used as a destination end-point. All L2CAP signalling commands are sent to CID 0x0001. CID 0x0002 is reserved for group-oriented channel. The same CID must not be reused as a local L2CAP channel endpoint for multiple simultaneous L2CAP channels between a local device and some remote device.

CID assignment is relative to a particular device and a device can assign CIDs independently from other devices. Thus, even if the same CID value has been assigned to (remote) channel endpoints by several remote devices connected to a single local device, the local device can still uniquely associate each remote CID with a different device.

Channel Operational States

NG_L2CAP_CLOSED

In this state, there is no channel associated with this CID. This is the only state when a link level connection (Baseband) may not exist. Link disconnection forces all other states into the NG_L2CAP_CLOSED state.

NG_L2CAP_W4_L2CAP_CON_RSP

In this state, the CID represents a local end-point and an L2CAP Connect Request message has been sent referencing this endpoint and it is now waiting for the corresponding L2CAP Connect Response message.

NG_L2CAP_W4_L2CA_CON_RSP

In this state, the remote end-point exists and an L2CAP Connect Request has been received by the local L2CAP entity. An L2CA Connect Indication has been sent to the upper layer and the part of the local L2CAP entity processing the received L2CAP Connect Request waits for the corresponding response. The response may require a security check to be performed.

NG_L2CAP_CONFIG

In this state, the connection has been established but both sides are still negotiating the channel parameters. The Configuration state may also be entered when the channel parameters are being renegotiated. Prior to entering the NG_L2CAP_CONFIG state, all outgoing data traffic is suspended since the traffic parameters of the data traffic are to be renegotiated. Incoming data traffic is accepted until the remote channel endpoint has entered the NG_L2CAP_CONFIG state. In the NG_L2CAP_CONFIG state, both sides will issue L2CAP Configuration Request messages if only defaults are being used, a null message will be sent. If a large amount of parameters need to be negotiated, multiple messages will be sent to avoid any MTU limitations and negotiate incrementally. Moving from the NG_L2CAP_CONFIG state to the NG_L2CAP_OPEN state requires both sides to be ready. An L2CAP entity is ready when it has received a positive response to its final request and it has positively responded to the final request from the remote device.

NG_L2CAP_OPEN
In this state, the connection has been established and configured, and data flow may proceed.

NG_L2CAP_W4_L2CAP_DISCON_RSP
In this state, the connection is shutting down and an L2CAP Disconnect Request message has been sent. This state is now waiting for the corresponding response.

NG_L2CAP_W4_L2CA_DISCON_RSP
In this state, the connection on the remote endpoint is shutting down and an L2CAP Disconnect Request message has been received. An L2CA Disconnect Indication has been sent to the upper layer to notify the owner of the CID that the remote endpoint is being closed. This state is now waiting for the corresponding response from the upper layer before responding to the remote endpoint.

Protocol Multiplexing

L2CAP supports protocol multiplexing because the Baseband Protocol does not support any "type" field identifying the higher layer protocol being multiplexed above it. L2CAP is able to distinguish between upper layer protocols such as the Service Discovery Protocol, RFCOMM and Telephony Control.

Segmentation and Reassembly

The data packets defined by the Baseband Protocol are limited in size. Large L2CAP packets must be segmented into multiple smaller Baseband packets prior to their transmission over the air. Similarly, multiple received Baseband packets may be reassembled into a single larger L2CAP packet.

Quality of Service

The L2CAP connection establishment process allows the exchange of information regarding the quality of service (QoS) expected between two Bluetooth units.

Groups

The Baseband Protocol supports the concept of a piconet, a group of devices synchronously hopping together using the same clock. The L2CAP group abstraction permits implementations to efficiently map protocol groups on to piconets.

The following features are outside the scope of L2CAP responsibilities:

- L2CAP does not transport audio designated for SCO links.

- L2CAP does not enforce a reliable channel or ensure data integrity, that is, L2CAP performs no retransmissions or checksum calculations.

- L2CAP does not support a reliable multicast channel.

- L2CAP does not support the concept of a global group name.

HOOKS

This node type supports the following hooks:

hci Bluetooth Host Controller Interface downstream hook.

l2c Upper layer protocol upstream hook. Usually the Bluetooth L2CAP socket layer is connected to the hook.

ctl Control hook. Usually the Bluetooth raw L2CAP sockets layer is connected to the hook.

INTERFACE TO THE UPPER LAYER PROTOCOLS (L2CA CONTROL MESSAGES)

Bluetooth specification says that L2CA request must block until response is ready. L2CAP node uses *token* field from Netgraph message header to match L2CA request and response. The upper layer protocol must populate *token*. L2CAP node will queue request and start processing. Later, when response is ready or timeout has occurred, L2CAP node will create new Netgraph message, set *token* and NFG_RESP flag and send message to the upper layer. Note that L2CA indication messages will not populate *token* and will not set NGF_RESP flag. There is no reason for this, because they are just notifications and do not require acknowledgment.

NGM_L2CAP_L2CA_CON

Requests the creation of a channel representing a logical connection to a physical address. Input parameters are the target protocol (PSM) and remote device's 48-bit address (BD_ADDR). Output parameters are the local CID (LCID) allocated by the local L2CAP entity, and Result of the request. If Result indicates a pending notification, the Status value may contain more information of what processing is delaying the establishment of the connection.

NGM_L2CAP_L2CA_CON_IND

This message includes the parameters for the address of the remote device that issued the connection request, the local CID representing the channel being requested, the Identifier contained in the request, and the PSM value the request is targeting.

NGM_L2CAP_L2CA_CON_RSP

Issues a response to a connection request event indication. Input parameters are the remote device's 48-bit address, Identifier sent in the request, local CID, the Response code, and the Status attached to the Response code. The output parameter is the Result of the service request. This primitive must be called no more than once after receiving the indication.

NGM_L2CAP_L2CA_CFG

Requests the initial configuration (or reconfiguration) of a channel to a new set of channel parameters. Input parameters are the local CID endpoint, new incoming receivable MTU (InMTU), new outgoing flow spec-ification, and flush and link timeouts. Output parameters are the Result, accepted incoming MTU (InMTU), the remote side's flow requests, and flush and link timeouts.

NGM_L2CAP_L2CA_CFG_IND

This message includes the parameters indicating the local CID of the channel the request has been sent to, the outgoing MTU size (maximum packet that can be sent across the channel) and the flowspec describing the characteristics of the incoming data. All other channel parameters are set to their default values if not provided by the remote device.

NGM_L2CAP_L2CA_CFG_RSP

Issues a response to a configuration request event indication. Input parameters include the local CID of the endpoint being configured, outgoing transmit MTU (which may be equal or less to the OutMTU parameter in the configuration indication event) and the accepted flowspec for incoming traffic. The output parameter is the Result value.

NGM_L2CAP_L2CA_QOS_IND

This message includes the parameter indicating the address of the remote Bluetooth device where the QoS contract has been violated.

NGM_L2CAP_L2CA_DISCON

Requests the disconnection of the channel. Input parameter is the CID representing the local channel endpoint. Output parameter is Result. Result is zero if an L2CAP Disconnect Response is received, otherwise a non-zero value is returned. Once disconnection has been requested, no process will be able to successfully read or write from the CID.

NGM_L2CAP_L2CA_DISCON_IND

This message includes the parameter indicating the local CID the request has been sent to.

NGM_L2CAP_L2CA_WRITE

Response to transfer of data request. Actual data must be received from appropriate upstream hook and must be prepended with header defined as follows.

```
/* L2CA data packet header */
typedef struct {
    uint32_t token;  /* token to use in L2CAP_L2CA_WRITE */
    uint16_t length; /* length of the data */
    uint16_t lcid;   /* local channel ID */
```

```
} __attribute__ ((packed)) ng_l2cap_l2ca_hdr_t;
```

The output parameters are Result and Length of data written.

NGM_L2CAP_L2CA_GRP_CREATE
: Requests the creation of a CID to represent a logical connection to multiple devices. Input parameter is the PSM value that the outgoing connectionless traffic is labelled with, and the filter used for incoming traffic. Output parameter is the CID representing the local endpoint. On creation, the group is empty but incoming traffic destined for the PSM value is readable. *This request has not been implemented.*

NGM_L2CAP_L2CA_GRP_CLOSE
: The use of this message closes down a Group. *This request has not been implemented.*

NGM_L2CAP_L2CA_GRP_ADD_MEMBER
: Requests the addition of a member to a group. The input parameter includes the CID representing the group and the BD_ADDR of the group member to be added. The output parameter Result confirms the success or failure of the request. *This request has not been implemented.*

NGM_L2CAP_L2CA_GRP_REM_MEMBER
: Requests the removal of a member from a group. The input parameters include the CID representing the group and BD_ADDR of the group member to be removed. The output parameter Result confirms the success or failure of the request. *This request has not been implemented.*

NGM_L2CAP_L2CA_GRP_MEMBERSHIP
: Requests a report of the members of a group. The input parameter CID represents the group being queried. The output parameter Result confirms the success or failure of the operation. If the Result is successful, BD_ADDR_Lst is a list of the Bluetooth addresses of the N members of the group. *This request has not been implemented.*

NGM_L2CAP_L2CA_PING
: Initiates an L2CA Echo Request message and the reception of the corresponding L2CAP Echo Response message. The input parameters are remote Bluetooth device BD_ADDR, Echo Data and Length of the echo data. The output parameters are Result, Echo Data and Length of the echo data.

NGM_L2CAP_L2CA_GET_INFO
: Initiates an L2CA Information Request message and the reception of the corresponding L2CAP Info Response message. The input parameters are remote Bluetooth device BD_ADDR and Information Type. The output parameters are Result, Information Data and Size of the information data.

NGM_L2CAP_L2CA_ENABLE_CLT
Request to disable (enable) the reception of connectionless packets. The input parameter is the PSM value indicating service that should be blocked (unblocked) and Enable flag.

NETGRAPH CONTROL MESSAGES

This node type supports the generic control messages, plus the following:

NGM_L2CAP_NODE_GET_FLAGS
Returns current state for the node.

NGM_L2CAP_NODE_GET_DEBUG
Returns an integer containing the current debug level for the node.

NGM_L2CAP_NODE_SET_DEBUG
This command takes an integer argument and sets current debug level for the node.

NGM_L2CAP_NODE_GET_CON_LIST
Returns list of active baseband connections (i.e., ACL links).

NGM_L2CAP_NODE_GET_CHAN_LIST
Returns list of active L2CAP channels.

NGM_L2CAP_NODE_GET_AUTO_DISCON_TIMO
Returns an integer containing the current value of the auto disconnect timeout (in sec).

NGM_L2CAP_NODE_SET_AUTO_DISCON_TIMO
This command accepts an integer and sets the value of the auto disconnect timeout (in sec). The special value of 0 (zero) disables auto disconnect timeout.

SHUTDOWN

This node shuts down upon receipt of an NGM_SHUTDOWN control message, or when all hooks have been disconnected.

SEE ALSO

netgraph(4), l2control(8), l2ping(8), ngctl(8)

HISTORY

The **l2cap** node type was implemented in FreeBSD 5.0.

AUTHORS

Maksim Yevmenkin <*m_evmenkin@yahoo.com*>

BUGS

Most likely. Please report if found.