loRaSwarmProtocol.py Reference

loRaSwarmProtcol.py class provides static helper functions for parsing and creating LoRaSwarm packets.

Parsing

When a packet is received as an array of Bytes, it can be parsed using the parsing methods. First, the packet must be parsed as a LoRaSwarm packet using the parse_packet_header function. This returns a tuple of a), A LoRaSwarm Protocol Header named tuple and b), the remaining packet Bytes with the header stripped off. The LoRaSwarm Protocol Header returned can be then used to determine how to parse the remaining bytes. The remaining Bytes are then passed to the correct parsing function and will return the respective named tuple for that packet type.

Named tuple definitions for each packet type are provided below:

Returned by: parse_packet_header.

LoRaSwarm Protocol Header	
nodeId	ID of node who sent the packet.
status1	Status Byte 1
status2	Status Byte 2
packetType	Byte determines the type of packet this header is wrapping.

Returned by: parse_heartbeat_packet.

Heartbeat Packet	
coordLat	GPS latitude in Decimal Degrees
coordLon	GPS longitude in Decimal Degrees
neighbours	Array of *NeighboursListElements*

Returned as elements of the 'neighbours' array field in parse_heartbeat_packet and parse_ext_application_packet.

NeighbourListElement	
neighbourId	Neighbour ID
snr	SNR in dB. Range r, $-5 \leq r \leq -20$
received	Bound for number of packets received.

Returned by: parse_application_packet.

Application Packet	
sender	ID of node from whom the packet originated
sequence	Sequence number of packet
forwardHorizon	Forward Horizon of packet
forwardCount	Current forward count
payloadLength	Length of following payload
payload	Array of bytes containing the payload

Returned by: parse_ext_application_packet.

Extended Application Packet	
neighbours	ID of node from whom the packet originated
applicationPacket	Application Packet named tuple containing the wrapped Application Packet

Returned by: parse_control_packet.

Control Packet	
sequence	Sequence number of control packet
payloadLength	Length of payload in Bytes
payload	Array of Bytes containing the payload

Creating Packets

In order to create a LoRaSwarm packet the header bytes must first be calculated. This is done by calling the function 'get_header_byte' which returns a bytearray. This bytearray of header Bytes must then be passed to the desired get_<packet type>_packet function as the first argument. The bytearray returned by the get_<packet type>_packet function is correctly formatted and ready to be transmitted over LoRa.

Function arguments are detailed below:

get_header_byte(nodeId, status1, status2)	
nodeId	ID of the node transmitting this packet
status1	Status Byte 1 of sending node
status2	Status Byte 2 of sending node

get_heartbeat_packet(header_bytes, coord_lat, coord_lon, neighbours)	
header_bytes	bytebuffer containing header Bytes of LoRaSwarm packet to be sent
coord_lat	GPS latitude in Decimal Degrees
coord_lon	GPS longitude in Decimal Degrees
neighbours	Array of Neighbour List Elements. (Note however that the 'received' parameter is given as an exact value of the number of received packets rather than a bound.

get_application_packet(header_bytes, sender, sequence, forward_horizon, forward_count, payload)	
header_bytes	bytebuffer containing header Bytes of LoRaSwarm packet to be sent
sender	ID of node from whom application packet originated.
sequence	Sequence ID of the packet
forward_horizon	Maximum number of times the packet should be forwarded.
forward_count	The number of times the packet will have been forwarded when a receiver receives this packet.

payload	A byte array containing the payload of the packet.

get_ext_app_packet(header_bytes, neighbours, sender, sequence, forward_horizon, forward_count, payload)	
header_bytes	bytebuffer containing header Bytes of LoRaSwarm packet to be sent
neighbours	Array of Neighbour List Elements. (Note however that the 'received' parameter is given as an exact value of the number of received packets rather than a bound.
sender	ID of node from whom application packet originated.
sequence	Sequence ID of the packet
forward_horizon	Maximum number of times the packet should be forwarded.
forward_count	The number of times the packet will have been forwarded when a receiver receives this packet.
payload	A byte array containing the payload of the packet.

get_control_packet(header_bytes, sequence, payload)	
sequence	Sequence ID of the packet
payload	A byte array containing the payload of the packet.