This module implements the WebSocket client functionality. WebSocket is a communication protocol providing full duplex, asynchronous communication between the connected endpoints. WebSocket is designed to leverage the currently existing web infrastructure. It is designed to work over HTTP ports 80 and 443 as well as to support HTTP proxies and intermediaries. WebSocket is fully compatible with Web (HTTP) servers and uses HTTP upgrade header to switch from HTTP to WebSocket protocol. WebSocket is standardized by IETF as RFC6455.

# Features and Limitations

Following are the list of salient features of this Websocket implementation:

1. Confirms to RFC6455
2. Both the server IP address and the host name (Domain Name)
3. Supports secured connection using SSL/TLS

# Header file/s

Components/websocket/inc/websock.h.

# Data Structure Definitions

## websock\_config\_t

This data structure is used to pass the parameters while opening a websocket connection with the remote server using websock\_open API.

|  |  |
| --- | --- |
| ***hostname*** | Host name or the IP address of the server |
| ***port*** | Port |
| ***uri*** | Websocket uri to connect to |
| ***secured*** | Secured websocket |
| ***ssl\_config*** | SSL connection configurations |
| ***time\_out*** | TCP connection timeout |
| ***hndshk\_hdrs*** | The headers to be set in the WebSocket Handshake request. The format is: "key:val". Use this to pass headers that are not set implicitly. Refer [websock\_open()](#_websock_open) for the list of headers set implicitly |
| ***num\_hndshk\_hdrs*** | Number of headers passed using hndshk\_hdrs |

Table : websock\_config\_t - parameters

## websock\_msg\_hdr\_t

This data structure is used to pass information about the data received from the server.

|  |  |
| --- | --- |
| ***fin*** | End of message |
| ***opcode*** | Opcode of the message frame.  If < 0, this structure contents are invalid. This can happen if the header is not fully received in the current call to websock\_recv().  This is not an error, keep calling websock\_recv(), unless websock\_recv() itself returns < 0 |
| ***payload\_len*** | Total length of the payload of the fragment being received |

Table : websock\_msg\_hdr\_t - parameters

# API Reference

## websock\_open

### Overview

This function opens a TCP socket to the server and performs the initial HTTP-based handshake to upgrade to the websocket protocol. Both secure (SSL) and non-secure websocket connections are supported.

Headers set implicitly during connection handshake:

1. |Host| header field whose value contains /host/ plus optionally ":" followed by /port/ (when not using the default port).
2. |Upgrade| header field whose value MUST include the "websocket" keyword.
3. |Connection| header field whose value MUST include the "Upgrade" token.
4. |Sec-websocket-Key|. The value of this header field MUST be a nonce consisting of a randomly selected 16-byte value that has been base64-encoded (see section 4 of [RFC4648]). The nonce MUST be selected randomly for each connection.
5. |Sec-websocket-Version|. The value of this header field MUST be 13.

The |Host| |Upgrade|, |Connection|, |Sec -Websocket-key| and |Sec-websocket-Version| headers are implicitly set by the websock\_Open() during the handshake. Any other headers to be used in connection upgrade request must be set by the caller.

### Definition

|  |
| --- |
| websock\_handle\_t  websock\_open(websock\_config\_t \*ws\_cfg) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *cfg* | Pointer to properly initialized configuration DS of type websock\_config\_t |

Table : websock\_open - parameters

### Return

Success: Pointer to websocket connection handle

Error: NULL

## websock\_close

### Overview

This function is used to close the connection.

### Definition

|  |
| --- |
| void  websock\_close(websock\_handle\_t h) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by websock\_open |

Table : websock\_close - parameters

### Return

None.

## websock\_send\_text

### Overview

This function is used to send “text” data over websocket connection.

### Definition

|  |
| --- |
| int  websock\_send\_text(websock\_handle\_t handle, char \*payload, int len) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by websock\_open |
| *payload* | Message payload |
| *len* | Payload length |

Table : websock\_send\_text - parameters

### Return

Success: 0

Error: -1

## websock\_send\_binary

### Overview

This function is used to send “binary” data over websocket connection.

### Definition

|  |
| --- |
| int  websock\_send\_binary(websock\_handle\_t handle, char \*payload, int len) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by websock\_open |
| *payload* | Message payload |
| *len* | Payload length |

Table : websock\_send\_binary - parameters

### Return

Success: 0

Error: -1

## websock\_recv

### Overview

This function receives websocket messages. This internally handles websocket close and ping messages. This is a blocking call, and blocks for the data for the specified timeout.

**Note**: This API needs to be kept calling in loop in a separate thread context.

### Definition

|  |
| --- |
| int  websock\_recv(websock\_handle\_t handle, websock\_msg\_hdr\_t \*msg\_hdr,  char \*buf, int \*len, int timeout) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *handle* | Handle returned by websock\_open |
| *msg\_hdr* | Pointer to structure of type websock\_msg\_hdr\_t used to pass on the websocket message header information to the calling application |
| *buf* | Pointer to buffer to copy payload data |
| *len* | Max buf length |
| *timeout* | Receive timeout |

Table : websock\_recv - parameters

### Return

Success: Received bytes. >=0

Error: -1

# Application Example

For the example code, refer: *examples/websocket* application.