TLS and DTLS Test Suite User Guide

Version 1.00

For use with TLS and DTLS Test Suite versions 1.12 and above

Date: 25-Jul-2017 18:21

Table of Contents

1 System Overview

1.1 Introduction

This test suite is a set of reference code for exercising HCC Embedded's DTLS and TLS software. These tests are designed to:

- Provide reference code.
- Provide a basic test system to ensure the system works correctly.
- Measure the performance of the TCP/IP stack.
- Measure the resource utilization of the TCP/IP stack.

1.2 Packages and Documents

Packages

The table below lists the packages that you need in order to use this module:

Package	Description
hcc_base_doc	This contains the two guides that will help you get started.
ip_tls	The TLS and DTLS module.
ip_tls_test	The test package described in this document.

Documents

For an overview of the HCC TLS/DTLS software, see Product Information on the main HCC website.

Readers should note the points in the HCC Documentation Guidelines on the HCC documentation website.

HCC Firmware Quick Start Guide

This document describes how to install packages provided by HCC to the target development environment. Also follow the *Quick Start Guide* when HCC provides package updates.

HCC Source Tree Guide

This document describes the HCC source tree. It gives an overview of the system to make clear the logic behind its organization.

HCC TLS and DTLS User Guide

This document describes the TLS and DTLS package itself.

HCC TLS and DTLS Test Suite User Guide

This is this document.

1.3 Change History

This is the first release of this manual, so there are no previous versions of this document.

For the history of changes made to the package code itself, see History: ip_tls_test.

2 Source File List

The following sections describe all the source code files included in the system. These files follow the HCC Embedded standard source tree system, described in the *HCC Source Tree Guide*. All references to file pathnames refer to locations within this standard source tree, not within the package you initially receive.

> **Note:** Do not modify any files except the configuration file.

2.1 API Header File

The file **src/api/api_tls_test.h** should be included by any application using the system. This is the only file that should be included by an application using this module. For details of these API functions, see Application Programming Interface.

2.2 Configuration File

The file **src/config/config_tls_test.h** contains all the configurable parameters of the system. You can configure the parameters as required. For detailed explanation of these options, see Configuration Options.

2.3 Test System File

The system file is **src/ip/stack/tls/test/tls_test_tcp.c**. This is described in Running Tests.

2.4 Version File

The file **src/version/ver_tls_test.h** contains the version number of this module. This version number is checked by all modules that use this module to ensure system consistency over upgrades.

3 Configuration Options

Set the system configuration options in the file **src/config/config_ip_tls_test.h**. This section lists the available options and their default values.

TLS_TEST_STACK_SIZE

The test stack size. The default is 2048.

TLS_TEST_PORT

The connection port for the client and the listen port for the server. The default is 4433.

TLS_TEST_LOCAL_PORT

The port of the local socket used for the DTLS client using the Socket interface. The default is 5000.

TLS_TEST_SVR_IP_V4_ADDR

The server IPv4 address. The default is `{ 10u, 1u, 3u, 240u }`.

TLS_TEST_SVR_IP_V6_ADDR

The server IPv6 address. The default is:

```
{ 0xfeu, 0x80u, 0x00u, 0x00u, 0x00u, 0x00u, 0x00u, 0x00u,\

0x69u, 0x8eu, 0xe8u, 0x1eu, 0x67u, 0x87u, 0x31u, 0x24u }
```

TLS_TEST_PEERNAME

The peer name. The default is "HCC demo".

TLS_TEST_CLIENT_VRFY

Set this if you want the system to try to verify the client. The default is FALSE.

TLS_TEST_MSG_SEND_MAX

The number of messages sent by the client. The default is 1.

TLS_TEST_TIMER_PERIOD

The timer period. The default is 100.

TLS_TEST_CONNECTION_TIMEOUT

The timeout for connection (in *100ms). The default is 79.

TLS_TEST_RECONNECTION_TIME

The time before reconnection (in *100ms). The default is 30.

TLS_TEST_IP_V6_EN

Set this to 1 to use IPv6 for the TLS connection. The default is 0.

TLS_TEST_PRINTF_EN

Keep the default of 1 to enable debug printf.

TLS_TEST_PRINT_DATA

Keep the default of 1 to enable printing of receive and send data.

TLS_SOCKET_RCV_BUF_SIZE

The size of the receive data buffer. The default is 4.

TLS_TEST_DATA_ST_IDX

The starting index of received data. The default is '0'.

TLS_TEST_DATA_END_IDX

The ending index of received data. The default is '9'.

TLS_TEST_DATA_RST

The characters that reset data. The default is `{'\r', '\n'}`.

TLS_TEST_CLIENT

Keep this at the default of 1 to test the client connection. Set it to 0 to run a server test.

TLS_TEST_TCP

Keep this at the default of 0 to test the Socket interface. Set it to 1 to test the TCP interface.

DTLS_TEST_EN

Keep this at the default of 0 to disable DTLS tests. To enable these, set it to 1.

DTLS_TEST_UDP

Keep this at the default of 0 to test the Socket interface. Set it to 1 to test the UDP interface.

TLS_SOCKET_SELECT

Set this to 1 to use **tls_select_socket()** to get information about received TLS data. The default of 0 means it uses **tls_poll_socket()** instead.

TLS_TEST_USE_STD_SOCKET

Set this to 1 when using standard sockets. The default is 0.

4 Running Tests

There are four types of test:

- Test client connection.
- Test server connection.
- DTLS tests.
- Tests for UDP interface or socket interface.

Set the relevant options in the configuration file before running any test type.

5 Application Programming Interface

This section documents the Application Programming Interface (API).

The functions are the following:

Function	Description
tls_test_init()	Initializes the test module and allocates the required resources.
tls_test_reg_callback()	Registers a callback function to be called when a test ends.

5.1 tls_test_init

Use this function to initialize the test module and allocate the resources required.

Format

```
t_tls_test_ret tls_test_init ( void )
```

Arguments

Argument
None.

Return Values

Return value	Description
TLS_OK	Successful execution.
Else	See Error Codes.

5.2 tls_test_reg_callback

Use this function to register a callback function to be called when a test ends.

Format

```
t_tls_test_ret tls_test_reg_callback ( t_tls_test_cb p_cb_fn )
```

Arguments

Argument	Description	Type
p_cb_fn	A pointer to the callback function.	t_tls_test_cb

Return Values

Return value	Description
TLS_OK	Successful execution.
Else	See Error Codes.

5.3 t_tls_test_cb

The **t_tls_test_cb** definition specifies the format of the test state callback function.

Format

```
typedef void ( * t_tls_test_cb ) (
    uint8_t    state,
    uint16_t   param )
```

Arguments

Argument	Description	Type
state	The test state.	uint8_t
param	The parameter passed to the callback.	uint16_t

5.4 Error Codes

The following tables show the meaning of the return codes. The test suite may not produce all of these errors.

Error code	Value	Meaning
TLS_INVALID_CONN_TICKET	0xFFFF	Invalid value for TLS session ticket.
DTLS_INVALID_UDP_HDL	0xFFFF	Invalid value for DTLS UDP connection handler.

Error code	Value	Meaning
TLS_OK	0	Successful execution.
TLS_SET	1	Flag is set.
TLS_NOT_SET	2	Flag is not set.
TLS_FOUND	3	Specified object was found.
TLS_SIGN_VERIFY	5	Signature should be verified.
TLS_WAIT	6	Function not executed, should be repeated.
TLS_INIT_ERR	7	Initialization error.
TLS_NOT_FOUND_ERR	8	The specified object was not found.
TLS_IO_ERR	9	IO operation error.
TLS_TYPE_ERR	10	The certificate type is incorrect.
TLS_FULL_ERR	11	Array is full, no free slot found.
TLS_NULL_PTR_ERR	12	One of the parameters was a NULL pointer.
TLS_HS_ERR	13	Handshake protocol error.
TLS_PARAM_ERR	14	Invalid parameter error.
TLS_MEM_ERR	15	Memory allocation error.
TLS_VERIFY_ERR	16	Signature verification error.
TLS_CERTIFICATE_ERR	17	A certificate is invalid.
TLS_ENCRYPTION_MODULE_ERR	18	Error returned by Embedded Encryption Manager.
TLS_HEARTBEAT_TIMEOUT_ERR	19	Heartbeat message timeout error.
TLS_DROP_MESSAGE	20	Drop current message silently.
TLS_TIMEOUT_ERR	21	Operation timed out.
TLS_DTLS_NEW_CONN	22	New connection found.
TLS_DTLS_DROP_FRMSG	23	Drop DTLS message fragment.
TLS_DTLS_CONNECT_ERR	24	Server is still connected. Cannot proceed with function.

5.5 Test States

The test states are as follows:

Name	Value	Description
TLS_TEST_ST_CONNECTED	1	Test connected.
TLS_TEST_ST_HANDSHAKE_DONE	2	TLS handshake succeeded.
TLS_TEST_ST_CONN_ACCEPT	3	TLS connection accepted.
TLS_TEST_ST_DATA_RECV	4	Test data received.
TLS_TEST_ST_DISCONNECTED	5	Test disconnected.

6 Integration

6.1 OS Abstraction Layer

All HCC modules use the OS Abstraction Layer (OAL) that allows the module to run seamlessly with a wide variety of RTOSes, or without an RTOS.

The test suite uses the following OAL components:

OAL Resource	Number Required
Tasks	1
Mutexes	0
Events	1

6.2 Utilities

The TCP code creates and uses a single timer in the **hcc_timer** module.

6.3 PSP Porting

The Platform Support Package (PSP) is designed to hold all platform-specific functionality, either because it relies on specific features of a target system, or because this provides the most efficient or flexible solution for the developer.

The module makes use of the following standard PSP functions:

Function	Package	Element	Description
psp_memcpy()	psp_base	psp_string	Copies a block of memory. The result is a binary copy of the data.
psp_memset()	psp_base	psp_string	Sets the specified area of memory to the defined value.