SCTPasp Test Port for TTCN-3 Toolset with TITAN, User's Guide

Zoltán Medve

Version 198 17-CNL 113 469, Rev. H, 2013-09-17

Table of Contents

About This Document . . . 1
How to Read This Document . . . 1
Prerequisite Knowledge . . . 1
System Requirements . . . 1
Fundamental Concepts . . . 1
Overview . . . 2
Installation . . . 2
Configuration . . . 2
SCTPasp Test Port Parameters in the RTE Configuration File . . . 3
Start Procedure . . . 8
TTCN-3 Test Executor . . . 8
Stop Procedure . . . 9
TTCN-3 Test Executor . . . 9
Using the test port in TTCN3 . . . 9
Abstract service primitives . . . 9
Incoming/Outgoing ASPs . . . 9
Incoming ASPs . . . 10
Outgoing ASPs . . . 12
Client Mode . . . 16
Server mode . . . 17
Reconnect mode . . . 17
Normal mode . . . 17
Error Messages . . . 17
Warning Messages . . . 18
Examples . . . 19
Configuration file . . . 19
Abbreviations . . . 21
References . . . 22

About This Document

How to Read This Document

This is the User's Guide for the SCTPasp test port. The SCTPasp test port is developed for the TTCN-3 Toolset with TITAN. This document is intended to be read together with Function Specification [5].

Prerequisite Knowledge

The knowledge of the TITAN TTCN-3 Test Executor [1] and the TTCN-3 language [1] is essential. Basic knowledge of the SCTP protocol is valuable when reading this document.

System Requirements

In order to operate the SCTPasp test port the following system requirements must be satisfied:

- Platform: Solaris 10 or Suse Linux 9.1 and above.
- TITAN TTCN-3 Test Executor R7A (1.7.pl0) or higher installed. For installation guide see [4].

NOTE	This version of the test port is not compatible with TITAN releases earlier than R7A. The usage of TITAN releases earlier than R8A is not recommended because this version of the test port is prepared to handle the big integer numbers which feature is introduced in TITAN R8A. The usage of TITAN releases earlier than R8A can result a dynamic test case error.

On SUSE Linux 9.1/9.2 the following SCTP Linux Kernel implementation packages (or higher version) should be installed:

```
lksctp-tools-1.0.1-2.i586.rpm
lksctp-tools-devel-1.0.1-2.i586.rpm
```

The test port is able to determine the version of the installed lksctp tool, so the compilation flags: `LKSCTP_1_0_9` and `LKSCTP_1_0_7` no longer needed.

Fundamental Concepts

The test port establishes SCTP connection between the TTCN-3 test executor and the SUT. The test port transmits and receives SCTP messages between the TITAN RTE and the SUT.

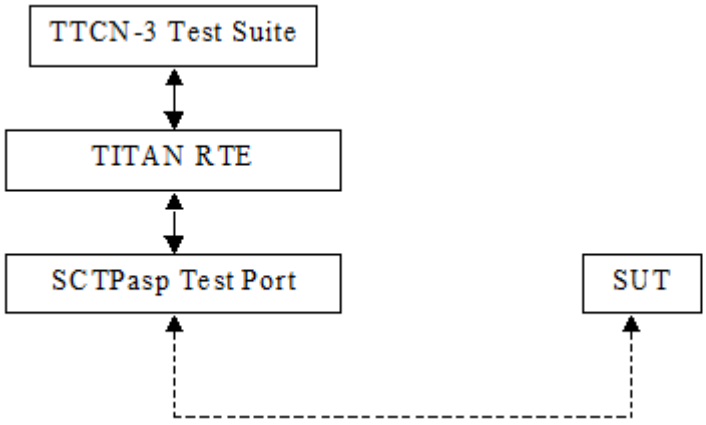
The SCTPasp test port has many ASPs to control the behavior of the test port and to provide information about internal events. For details see Abstract Service Primitives

Overview

The SCTP test port offers SCTP primitives to the test suite in TTCN-3 format. The TTCN-3 definition of the ASPs can be found in a separate TTCN-3 module. This module must be imported into the test suite.

The test port translates the SCTP ASPs to SCTP packets (and vice versa) between the TITAN RTE and the SUT.

See the overview of the test system below:



Installation

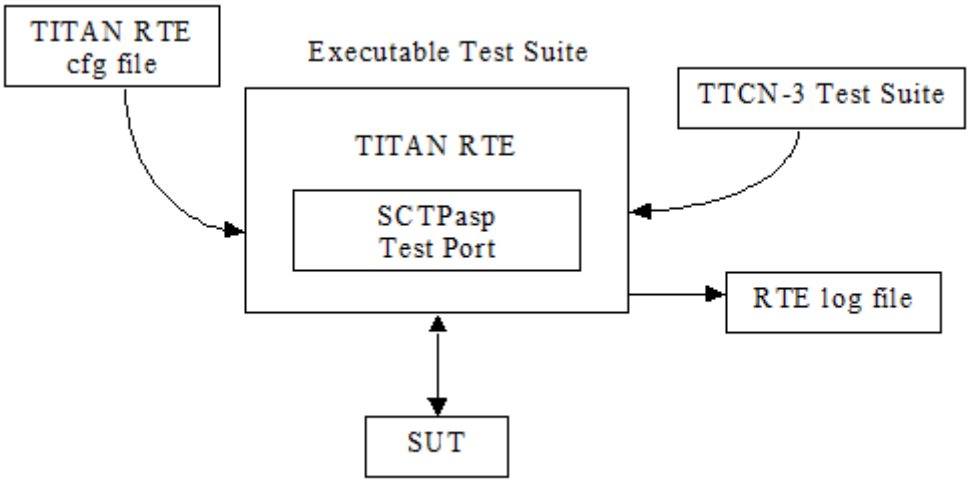
Since the SCTPasp test port is used as a part of the TTCN-3 test environment this requires TTCN-3 Test Executor to be installed before any operation of the SCTP test port. For more details on the installation of TTCN-3 Test Executor see the TITAN Installation Guide [4].

NOTE	The test port files shall be added to the project or to the *Makefile*. On SUSE Linux the rpms mentioned in 1.3 shall be installed.

Configuration

The executable test program behavior is determined via the RTE configuration file. This is a simple text file, which contains various sections (e.g. `[TESTPORT_PARAMETERS]`) after each other. The usual suffix of the RTE configuration file is *.cfg*. For further information about the configuration file see [1].

See the overview of the configuration process below:



SCTPasp Test Port Parameters in the RTE Configuration File

In the `[TESTPORT_PARAMETERS]` section you can specify parameters that are passed to the test ports. Each parameter definition consists of a component name, a port name, a parameter name and a parameter value. The component name can be either an identifier or a component reference (integer) value. The port and parameter names are identifiers while the parameter value must always be a charstring (with quotation marks). Instead of component name or port name (or both of them) the asterisk ("*") sign can be used, which means "all components" or "all ports of the component". More information about the RTE configuration file can be found in [1].

In the `[TESTPORT_PARAMETERS]` section the following parameters can be set for the SCTPasp test port. If the corresponding parameter is mandatory an (M), if it is optional an (O), if it does not affect the test port – an (X) and if it is conditional a © is shown after its name. The first letter refers to simple mode, the second letter refers to the normal mode:

- `simple_mode (M, O)`

  The parameter is optional, and can be used to enable simple mode. This parameter has the highest priority. If it is given overrides parallel reconnect and `server_mode` settings. Available values: `"yes"`/`"no"`.

  The default value is `"no"`.

- `reconnect (C, X)`
  - Simple mode

    The parameter is optional in client mode and forbidden in server mode (server mode and reconnect mode are mutually exclusive). It can be used to enable reconnect mode. Available values: `"yes"`/`"no"`.

    The default value is `"no"`.

  - Normal mode

It does not affect the test port.

- `reconnect_max_attempts (O, X)`
  - Simple mode

    The parameter is optional, and can be used to specify the maximum number of attempts to restore the SCTP connection in reconnect mode.

    The default value is `"6"`.

    The time interval, between two subsequent connection attempts, is increasing logarithmically (multiplied by 2). The initial time interval is one second. Allowed values: positive integers.

  - Normal mode

    It does not affect the test port.

- `server_mode (C, X)`
  - Simple mode

    The parameter is conditional (server mode and reconnect mode are mutually exclusive), and can be used to specify whether the test port shall act as a server or a client. If the value is `"yes"`, the test port will act as a server. If the value is `"no"`, the test port will act as a client. Available values: `"yes"`/`"no"`.

    The default value is `"no"`.

  - Normal mode

    It does not affect the test port.

- `debug (O, O)`
  - Simple mode / Normal mode

    The parameter is optional, and can be used to enable debug logging. Available values: `"yes"` /`"no"`.

    The default value is `"no"`.

- `server_backlog (O, O)`
  - Simple mode

    The parameter can be used to specify the number of allowed pending (queued) connection requests on the port the server listens. It is optional in server mode and not used in client mode.

    The default value is `"1"`.

    Allowed values: positive integers.

  - Normal mode

In this case the parameters affect all servers handled by the test port.

- `local_IP_address (O, X)`
  - Simple mode

    The parameter can be used to specify the local IP address the SCTP sockets bind to. Its presence is optional.

    The default value is `INADDR_ANY`.

    Allowed values: valid IPv4 addresses given with DNS name or dot notation.

NOTE	Using a machine with multiple interfaces an arbitrary IP address will be chosen to bind to when this parameter is not specified. To avoid this error prone situation it is recommended to set this parameter in this case.

  - Normal mode

    It does not affect the test port.

- `local_port (C, X)`
  - Simple mode

    The parameter can be used to specify the port the SCTP sockets bind to. It is mandatory in server mode and optional in client mode. There is no default value. Allowed values: `0-65535`.

  - Normal mode

    It does not affect the test port.

- `peer_IP_address (C, O)`
  - Simple mode

    The parameter can be used to specify the server's IP address. It is not used in server mode. It is mandatory in reconnect mode optional otherwise. There is no default value. Allowed values: valid IPv4 addresses given with DNS name or dot notation.

  - Normal mode

    It is used in the connect ASPs when peer hostname is omitted.

- `peer_port (C, O)`
  - Simple mode

    The parameter can be used to specify the peer's listening port. It is not used in server mode. It is mandatory in reconnect mode optional otherwise. There is no default value. Allowed values: `0-65535`.

  - Normal mode

    It is used in the connect ASPs when peer port number is omitted.

- `sinit_num_ostreams (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to determine the number of outbound streams the application wishes to be able to send to.

    The default value is `"64"`.

    Allowed values: positive integers.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sinit_max_instreams (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to determine the maximum number of inbound streams the application is prepared to support.

    The default value is `"64"`.

    Allowed values: positive integers.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sinit_max_attempts (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to specify how many attempts the SCTP endpoint should make at resending the INIT.

    The default value is `"0"`.

    Allowed values: positive integers.

NOTE	The default value of `"0"` indicates to use the endpoint's default value. Alteration is not recommended unless you know what you are doing.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sinit_max_init_timeo (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to determine the largest Time-Out or RTO value (in milliseconds) to use in attempting an INIT.

    The default value is `"0"`.

Allowed values: positive integers.

> **NOTE** The default value of `"0"` indicates to use the endpoint's default value. Alteration is not recommended unless you know what you are doing.

- Normal mode

  It applies to the test port globally (all client and server sockets).

- `sctp_association_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_ASSOC_CHANGE` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sctp_address_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_PEER_ADDR_CHANGE` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sctp_send_failure_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_SEND_FAILED` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sctp_peer_error_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_REMOTE_ERROR` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

- Normal mode

  It applies to the test port globally (all client and server sockets).

- `sctp_shutdown_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_SHUTDOWN_EVENT` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sctp_partial_delivery_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_PARTIAL_DELIVERY_EVENT` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

- `sctp_adaption_layer_event (0, 0)`
  - Simple mode

    The parameter is optional, and can be used to enable `ASP_SCTP_ADAPTION_INDICATION` ASPs. Available values: `"enabled"`/`"disabled"`.

    The default value is `"enabled"`.

  - Normal mode

    It applies to the test port globally (all client and server sockets).

Start Procedure

TTCN-3 Test Executor

Before running the executable test suite the TTCN-3 modules and C++ codes should be compiled and linked into an executable program. This process can be automated using the make utility. The *Makefile* generation process is described in [1].

NOTE	The C++ implementation files *SCTPasp_PT.hh* and *SCTPasp_PT.cc* of the test port should be included in the *Makefile*.

If the executable test suite is ready, run it giving the RTE configuration file as argument in your terminal:

```
Home> ExecutabletestSuite RTEConfigurationFile.cfg
```

For more information, see [1].

Stop Procedure

TTCN-3 Test Executor

The test port should stop automatically after it finished the execution of all test cases. It closes down the SCTP socket towards the SUT and terminates.

The execution of the test suite can be stopped at any time by pressing `<Cntr>-c`. It shuts down the socket and terminates.

Using the test port in TTCN3

Abstract service primitives

Incoming/Outgoing ASPs

`ASP_SCTP`

This ASP is used to send and receive user data. It has four fields:

- `client_id`:
  It specifies the client the message is to be sent to. This field should be set to `"OMIT"` in client mode and it is mandatory in server mode and normal mode. Breaking these rules will cause a TTCN error. In received `ASP_SCTP` messages the field will contain the id of the peer endpoint.
- `sinfo_stream`:
  It specifies the stream number the message is to be sent to. Each association has at least one outbound stream. For further details about streams, see [7].
- `sinfo_ppid`:
  It specifies information about the upper protocol layer.

NOTE	This information is passed opaquely by the SCTP stack from one end to the other.

- `data`:
  User data stored in unstructured octetstring.

Incoming ASPs

ASP_SCTP_ASSOC_CHANGE

This ASP indicates an `sctp_assoc_change` notification. This notification is generated when the status of an association has changed: it has been opened or closed.

It has two fields:

- `client_id`:
  It specifies the association identified by the participating client.
- `sac_state`:
  It indicates what kind of event has happened to the association. The most important ones are `SCTP_COMM_UP` and `SCTP_COMM_LOST`. The former indicates that a new association is now ready and data may be exchanged with this peer. The latter indicates that the association has failed. For more information, see [8].

ASP_SCTP_PEER_ADDR_CHANGE

This ASP indicates an `sctp_peer_addr_change` notification. This notification is generated when an address that is part of an existing association has experienced a change of state (for example, a failure or return to service of the reachability of an endpoint via a specific transport address).

It has two fields:

- `client_id`:
  It specifies the association identified by the participating client.
- `spc_state`:
  It indicates what kind of event has happened to an address that is part of an existing association. The most important ones are `SCTP_ADDR_AVAILABLE` and `SCTP_ADDR_UNREACHABLE`. The former indicates that this address is now reachable. The latter indicates that the address specified can no longer be reached. Any data sent to this address is rerouted to an alternate until this address becomes reachable. For more information, see [7].

NOTE	The test port currently does not support multihoming. This means that one address is available per association.

ASP_SCTP_SEND_FAILED

This ASP indicates an **`sctp_send_failed`** notification. This notification is generated when a message could not be sent to the remote endpoint.

It has one field:

- `client_id`:

It specifies the association identified by the participating client.

ASP_SCTP_REMOTE_ERROR

This ASP indicates an **sctp_remote_error** notification. This notification is generated when an operational error has been received from the remote peer.

It has one field:

- client_id:
  It specifies the association identified by the participating client.

ASP_SCTP_SHUTDOWN_EVENT

This ASP indicates an sctp_shutdown_event notification. This notification is generated when the peer endpoint has been shut down.

It has one field:

- client_id:
  It specifies the association identified by the participating client.

ASP_SCTP_PARTIAL_DELIVERY_EVENT

This ASP indicates an sctp_partial_delivery_event notification. It is used to tell a receiver that the partial delivery has been aborted. This may indicate the association is about to be aborted.

It has one field:

- client_id:
  It specifies the association identified by the participating client.

ASP_SCTP_ADAPTION_INDICATION

This ASP indicates an sctp_adaption_indication notification. It holds the peer's indicated adaption layer.

It has one field:

- client_id:
  It specifies the association identified by the participating client.

ASP_SCTP_Connected

This ASP is used to indicate that a new client is connected to one of our server sockets in normal mode. It has five fields:

- client_id:
  It specifies the association identified by the participating client.
- local_hostname:
  It specifies the local host name the remote client connected to.

- `local_portnumber`:
  It specifies the local port the remote client connected to.
- `peer_hostname`:
  It specifies the host name of the remote client.
- `peer_portnumber`:
  It specifies the port number of the remote client.

ASP_SCTP_SENDMSG_ERROR

This ASP is used to indicate a send message error by echoing back the `ASP_SCTP` being failed to send. It has four fields:

- `client_id`:
  It specifies the client the message is to be sent to.
- `sinfo_stream`:
  It specifies the stream number the message is to be sent to.
- `sinfo_ppid`:
  It specifies information about the upper protocol layer.
- `data`:
  It user data stored in unstructured octetstring.

ASP_SCTP_RESULT

This ASP is used to indicate the status of action started by the user. It is generated after `ASP_SCTP_Connect`, `ASP_SCTP_ConnectFrom` and `ASP_SCTP_SetSocketOptions`. Reporting server listening socket opening result is optional, and can be activated with the `SCTP_REPORT_LISTEN_RESULT` C++ pre-processor flag:

```
CPPFLAGS = -D$(PLATFORM) -I$(TTCN3_DIR)/include -I$(SCTP_DIR)/include
–DSCTP_REPORT_LISTEN_RESULT
```

The ASP has three fields:

- `client_id`:
  It specifies the association identified by the participating client.
- `error_status`:
  It specifies if there was an error during the execution. If the operation is successful it is set to `"0"`, otherwise it is set to `"1"`.
- `error_message`:
  It holds the textual information about the error caused by the user started operation. This field is optional. It will be omitted if the operation is successful.

Outgoing ASPs

ASP_SCTP_Connect

This ASP is used in client mode to initiate a new connection. You should not use it in server mode otherwise you will get a TTCN error. It has two fields:

- `peer_hostname`:
  It specifies the host name of the SCTP server. This field is optional. It may be omitted when the corresponding test port parameter has been already specified in the configuration file. If this field is omitted and the corresponding test port parameter is not specified in the configuration file, TTCN error will be generated.
- `peer_portnumber`:
  It specifies the port number of the SCTP server. This field is optional. It may be omitted when the corresponding test port parameter has been already specified in the configuration file. If this field is omitted and the corresponding test port parameter is not specified in the configuration file, TTCN error will be generated.

NOTE	In normal mode `ASP_SCTP_Connect` returns immediately and `ASP_SCTP_RESULT` will indicate the result of the operation. This may take some time if the remote end does not answer. In simple mode `ASP_SCTP_Connect` blocks until the end of the connect operation.

ASP_SCTP_ConnectFrom

This ASP is used in normal mode to initiate a new connection when the local host name and port number should be defined. In simple mode it has no affect. It has four fields:

- `local_hostname`:
  It specifies the local IP address the SCTP socket binds to. This field is optional. If omitted it takes the value of the corresponding test port parameter. If there is no such parameter it will be assigned to the default value (`INADDR_ANY`).
- `local_portnumber`:
  It specifies the local port number the SCTP socket binds to.
- `peer_hostname`:
  It specifies the host name of the SCTP server. This field is optional. It may be omitted when the corresponding test port parameter has been already specified in the configuration file. If this field is omitted and the corresponding test port parameter is not specified in the configuration file, TTCN error will be generated.
- `peer_portnumber`:
  It specifies the port number of the SCTP server. This field is optional. It may be omitted when the corresponding test port parameter has been already specified in the configuration file. If this field is omitted and the corresponding test port parameter is not specified in the configuration file, TTCN error will be generated.

NOTE	`ASP_SCTP_ConnectFrom` returns immediately and `ASP_SCTP_RESULT` will indicate the result of the operation. This may take some time if the remote end does not answer.

ASP_SCTP_Listen

This ASP is used in normal mode to create a new server socket. In simple mode it has no affect. It has two fields:

- `local_hostname`:
  It specifies the local IP address the SCTP socket binds to. This field is optional. If omitted it takes the value of the corresponding test port parameter. If there is no such parameter it will be assigned to the default value (`INADDR_ANY`).
- `local_portnumber`:
  It specifies the local port number the SCTP socket binds to.

NOTE	To activate reporting the result of the listen operation, see section `ASP_SCTP_RESULT`

ASP_SCTP_SetSocketOptions

This ASP is defined as a union and can be applied to the setting of four different groups of socket options.

- `SCTP_INIT`

  It has four fields:

  - `sinit_num_ostreams`
  - `sinit_max_instreams`
  - `sinit_max_attempts`,
  - `sinit_max_init_timeo`

  They have the same semantics as the corresponding test port parameters described in section SCTPasp Test Port Parameters in the RTE Configuration File

- `SCTP_EVENTS`

  It has eight fields:

  - `sctp_data_io_event`
  - `sctp_association_event`
  - `sctp_address_event`
  - `sctp_send_failure_event`
  - `sctp_peer_error_event`
  - `sctp_shutdown_event`
  - `sctp_partial_delivery_event`
  - `sctp_adaption_layer_event`

  They have the same semantics as the corresponding test port parameters described in section SCTPasp Test Port Parameters in the RTE Configuration File.

- `SO_LINGER`

This option is used to perform the SCTP ABORT primitive. To enable the option set `l_onoff` to `"1"`. If the `l_linger` value is set to `"0"`, sending `ASP_SCTP_Close` is the same as the ABORT primitive. If the value is set to a negative value you will get a warning message. If the value is set to a positive value, the `close`() operation can be blocked for at most `l_linger` milliseconds. If the graceful shutdown phase does not finish during this period, `close`() will return but the graceful shutdown phase continues in the system.

It has two fields:

- `l_onoff`:
  Setting option on or off.
- `l_linger`:
  Setting linger time.

- `SCTP_RTOINFO`

This option is used to set the retransmission timeout (RTO) parameters on a per-socket basis. It has four fields:

- `client_id`:
  It specifies the association identified by the participating client.
- `srto_initial`:
  It specifies the initial RTO value in milliseconds.
- `srto_max`:
  It specifies the maximum RTO value in milliseconds.
- `srto_min`:
  It specifies the minimum RTO value in milliseconds.

NOTE	`SCTP_EVENTS` options apply to the test port globally (all client and server sockets). In normal mode `SCTP_INIT` and `SO_LINGER` socket options only apply to the latest socket created by `ASP_SCTP_Connect`, `ASP_SCTP_ConnectFrom` and `ASP_SCTP_Listen`.

ASP_SCTP_Close

This ASP is used to close SCTP connections. It has one field:

- `client_id`:
  It specifies the association identified by the participating client to be closed.
  - Simple mode:

    This field should be set to `"OMIT"` in client mode otherwise a TTCN error will be generated. If you omit it in server mode all client connections will be closed.

  - Normal mode:

    If you omit the `client_id` all client and server sockets will be closed.

Client Mode

In client mode the ASPs should be used in the following sequence (optional steps are placed in brackets; "*" means `0-many`; "+" means `1-many`; "?" means `0-1`):

- `ASP_SCTP_Connect`

  Example template:

  ```
  template ASP_SCTP_Connect t_ASP_SCTP_Connect :=
  {
    peer_hostname :=  localhost,
    peer_portnumber := 6017
  }
  ```

- `ASP_SCTP_SetSocketOptions`

  Example template:

  ```
  template ASP_SCTP_SetSocketOptions t_ASP_SCTP_EVENTS :=
  {
    Sctp_events :=
    {
      sctp_data_io_event := true,
      sctp_association_event := true,
      sctp_address_event := false,
      sctp_send_failure_event := false,
      sctp_peer_error_event := false,
      sctp_shutdown_event := false,
      sctp_partial_delivery_event := false,
      sctp_adaption_layer_event := false
    }
  }
  ```

- `ASP_SCTP`

  Example template:

  ```
  template ASP_SCTP t_ASP_SCTP :=
  {
    client_id := omit,
    sinfo_stream := 0,
    sinfo_ppid := 0,
    data := 'FFF000'O
  }
  ```

- `ASP_SCTP_Close`

Example template:

```
template ASP_SCTP_Close t_ASP_SCTP_Close :=
{
   client_id := omit
}
```

In client mode `client_id` should be set to `"OMIT"`!

NOTE	In client mode the connection should be initiated manually by sending out `ASP_SCTP_Connect`.

Server mode

In server mode the following ASPs can be used in arbitrary sequences: `ASP_SCTP_SetSocketOptions`, `ASP_SCTP`, `ASP_SCTP_Close`. Using `ASP_SCTP_Connect` will result in a TTCN error.

Reconnect mode

There is a special reconnect mode when the test port is used as a client. In reconnect mode the client automatically connect to an arbitrary server. If the connection fails a reconnection procedure will be initiated. This procedure will block the RTE, it is strongly recommended not to use the test port in reconnect mode.

In reconnect mode only `ASP_SCTP` should be used.

Normal mode

In normal mode the test port can handle many client and server socket at the same time. This can be achieved by consecutive usage of `ASP_SCTP_Connect`, `ASP_SCTP_ConnectFrom` and `ASP_SCTP_Listen`. The several SCTP associations can be differentiated by their `client_ids`. The first sources of the `client_id` are ASP_SCTP_RESULT, which returns after a client socket attempts to connect to a server socket, and `ASP_SCTP_Connected`, which is got when a server socket accepts a new client connection. `ASP_SCTP_Conneced` contains information about the remote host name and port of the client too.

Error Messages

The error messages have the following general form:

```
Dynamic test case error: <error text>
```

Error messages are written into the log file. In the log file a time stamp is also given before the message text.

The list of the possible error messages is shown below. Note that this list contains the error messages produced by the test port. The error messages coming from the TITAN are not shown.

`set_parameter(): Invalid parameter value: %s for parameter %s. Only yes and no can be used!`

`set_parameter(): Invalid parameter value: %s for parameter %s. It should be positive integer!`

`set_parameter(): Invalid parameter value: %s for parameter %s. It should be enabled or disabled!`

`Event handler: accept error (server mode)!`

`Fcntl() error!`

`user_map(): server mode and reconnect mode are mutually exclusive!`

`user_map(): in server mode local_port must be defined!`

`Listen error!`

`ASP_SCTP_CONNECT is not allowed in server mode!`

`Peer IP address should be defined!`

`Peer port should be defined!`

`ASP_SCTP_CONNECT called during active connection.`

`Setsocketoptions error: UNBOUND value!`

`In NORMAL mode the client_id field of ASP_SCTP should be set to a valid value and not to omit!`

`In client mode the client_id field of ASP_SCTP_Close should be set to OMIT!`

`In server mode the client_id field of ASP_SCTP should be set to a valid value and not to omit!`

`In client mode the client_id field of ASP_SCTP should be set to OMIT!`

`Bad client id! %d`

`Forced reconnect failed! Remote end is unreachable!`

`map_delete_item: index out of range (0-%d): %d`

`Socket error: cannot create socket!`

`Bind error!`

`Gethostbyname error!`

`Gethostbyname error! h→h_addr is NULL!`

Warning Messages

`%s: unknown & unhandled parameter: %s`

`Connect error!`

`Setsockopt error!`

```
Sendmsg error! Strerror=%s

Unknown notification type!
```

Examples

Configuration file

An example RTE configuration file is shown below:

```
#ModuleName.SampleParameter := SampleValue
[TESTPORT_PARAMETERS]
system.SCTP_SimpleClientPort.simple_mode := "yes"
system.SCTP_SimpleClientPort.reconnect := "no"
system.SCTP_SimpleClientPort.reconnect_max_attempts := "10"
system.SCTP_SimpleClientPort.server_mode := "no"
system.SCTP_SimpleClientPort.debug := "yes"
system.SCTP_SimpleClientPort.server_backlog := "1"
system.SCTP_SimpleClientPort.peer_IP_address := "127.0.0.1"
system.SCTP_SimpleClientPort.peer_port := "6017"
system.SCTP_SimpleClientPort.sinit_num_ostreams := "64"
system.SCTP_SimpleClientPort.sinit_max_instreams := "64"
system.SCTP_SimpleClientPort.sinit_max_attempts := "0"
system.SCTP_SimpleClientPort.sinit_max_init_timeo := "0"
system.SCTP_SimpleClientPort.sctp_association_event := "enabled"
system.SCTP_SimpleClientPort.sctp_address_event := "enabled"
system.SCTP_SimpleClientPort.sctp_send_failure_event := "enabled"
system.SCTP_SimpleClientPort.sctp_peer_error_event := "enabled"
system.SCTP_SimpleClientPort.sctp_shutdown_event := "enabled"
system.SCTP_SimpleClientPort.sctp_partial_delivery_event := "enabled"
system.SCTP_SimpleClientPort.sctp_adaption_layer_event := "enabled"

system.SCTP_SimpleServerPort.simple_mode := "yes"
system.SCTP_SimpleServerPort.reconnect := "no"
system.SCTP_SimpleServerPort.reconnect_max_attempts := "10"
system.SCTP_SimpleServerPort.server_mode := "yes"
system.SCTP_SimpleServerPort.debug := "yes"
system.SCTP_SimpleServerPort.server_backlog := "1"
system.SCTP_SimpleServerPort.local_IP_address := "0.0.0.0"
system.SCTP_SimpleServerPort.local_port := "6017"
system.SCTP_SimpleServerPort.peer_IP_address := "127.0.0.1"
system.SCTP_SimpleServerPort.sinit_num_ostreams := "64"
system.SCTP_SimpleServerPort.sinit_max_instreams := "64"
system.SCTP_SimpleServerPort.sinit_max_attempts := "0"
system.SCTP_SimpleServerPort.sinit_max_init_timeo := "0"
system.SCTP_SimpleServerPort.sctp_association_event := "enabled"
system.SCTP_SimpleServerPort.sctp_address_event := "enabled"
system.SCTP_SimpleServerPort.sctp_send_failure_event := "enabled"
system.SCTP_SimpleServerPort.sctp_peer_error_event := "enabled"
```

```
system.SCTP_SimpleServerPort.sctp_shutdown_event := "enabled"
system.SCTP_SimpleServerPort.sctp_partial_delivery_event := "enabled"
system.SCTP_SimpleServerPort.sctp_adaption_layer_event := "enabled"


system.SCTP_ClientPort.simple_mode := "no"
system.SCTP_ClientPort.reconnect := "no"
system.SCTP_ClientPort.reconnect_max_attempts := "10"
system.SCTP_ClientPort.server_mode := "no"
system.SCTP_ClientPort.debug := "yes"
system.SCTP_ClientPort.server_backlog := "1"
system.SCTP_ClientPort.peer_IP_address := "127.0.0.1"
system.SCTP_ClientPort.peer_port := "6017"
system.SCTP_ClientPort.sinit_num_ostreams := "64"
system.SCTP_ClientPort.sinit_max_instreams := "64"
system.SCTP_ClientPort.sinit_max_attempts := "0"
system.SCTP_ClientPort.sinit_max_init_timeo := "0"
system.SCTP_ClientPort.sctp_association_event := "enabled"
system.SCTP_ClientPort.sctp_address_event := "enabled"
system.SCTP_ClientPort.sctp_send_failure_event := "enabled"
system.SCTP_ClientPort.sctp_peer_error_event := "enabled"
system.SCTP_ClientPort.sctp_shutdown_event := "enabled"
system.SCTP_ClientPort.sctp_partial_delivery_event := "enabled"
system.SCTP_ClientPort.sctp_adaption_layer_event := "enabled"

system.SCTP_ServerPort.simple_mode := "no"
system.SCTP_ServerPort.reconnect := "no"
system.SCTP_ServerPort.reconnect_max_attempts := "10"
system.SCTP_ServerPort.server_mode := "yes"
system.SCTP_ServerPort.debug := "yes"
system.SCTP_ServerPort.server_backlog := "1"
system.SCTP_ServerPort.local_IP_address := "0.0.0.0"
system.SCTP_ServerPort.local_port := "6017"
system.SCTP_ServerPort.peer_IP_address := "127.0.0.1"
system.SCTP_ServerPort.sinit_num_ostreams := "64"
system.SCTP_ServerPort.sinit_max_instreams := "64"
system.SCTP_ServerPort.sinit_max_attempts := "0"
system.SCTP_ServerPort.sinit_max_init_timeo := "0"
system.SCTP_ServerPort.sctp_association_event := "enabled"
system.SCTP_ServerPort.sctp_address_event := "enabled"
system.SCTP_ServerPort.sctp_send_failure_event := "enabled"
system.SCTP_ServerPort.sctp_peer_error_event := "enabled"
system.SCTP_ServerPort.sctp_shutdown_event := "enabled"
system.SCTP_ServerPort.sctp_partial_delivery_event := "enabled"
system.SCTP_ServerPort.sctp_adaption_layer_event := "enabled"

#ComponentID.PortName.ParameterName := "ParameterValue"
[EXTERNAL_COMMANDS]

#BeginControlPart := "begin_control_part_command"
#EndControlPart := "end_control_part_command"
```

```
#BeginTestCase := "begin_testcase_command"
#EndTestCase := "end_testcase_command"
[LOGGING]
FileMask := LOG_ALL | TTCN_MATCHING | TTCN_DEBUG
ConsoleMask := TTCN_ERROR | TTCN_WARNING | TTCN_ACTION | TTCN_TESTCASE |
TTCN_STATISTICS | TTCN_DEBUG
SourceInfoFormat := Single

#FileMask := LOG_ALL | TTCN_MATCHING | TTCN_DEBUG
#ConsoleMask := LOG_ALL | TTCN_MATCHING | TTCN_DEBUG
#TimeStampFormat := DateTime
#LogEventTypes := Yes
#LogSourceInfo := Yes
[GROUPS]

#Group := host1, host2, host3
[COMPONENTS]

#ComponentName := Group
[MAIN_CONTROLLER]
TCPPort := 9999
NumHCs := 1

[EXECUTE]
SCTPasp_regressiontest_Testcases.control

//saved by GUI
```

Abbreviations

ASP

Abstract Service Primitive

RTE

Run-Time Environment

SCTP

Stream Control Transmission Protocol Terminology

SUT

System Under Test

TTCN-3

Testing and Test Control Notation version 3

References

[1] ETSI ES 201 873-1 v3.2.1 (2007-02)
The Testing and Test Control Notation version 3; Part 1: Core Language

[2] User Guide for TITAN TTCN-3 Test Executor

[3] Programmer's Technical Reference for TITAN TTCN-3 Test Executor

[4] Installation Guide for TITAN TTCN-3 Test Executor

[5] SCTPasp Test Port for TTCN-3 Toolset with TITAN, Function Specification

[6] Socket API Extensions for Stream Control Transmission Protocol (SCTP)
https://tools.ietf.org/html/draft-ietf-tsvwg-sctpsocket-10

[7] RFC 2960 (2000)
Stream Control Transmission Protocol