Prepared (also subject responsible if other)
ETH/XZR Gábor Szalai +36 1 4397591

No.
198 17-CNL 113 320 Uen

Approved	Checked	Date	Rev	Reference
ETH/XZRC (Zsolt Szego)		2012-09-07	L	GASK2

Telnet Test Port for TTCN-3 Toolset with TITAN, User Guide

Contents

1 Introduction

1.1 Revision history

Date	Rev	Characteristics	Prepared
2004-09-02	PA1	First draft version	EGBOTAT
2004-10-21	A	Final version after review	EGBOTAT
2005-01-11	PB1	Improvements	ETHACI
2005-02-22	PB2	Added new cfg file parameters	ETHACI
2005-03-03	B	Final version after review	ETHACI
2005-05-20	PC1	New parameters, options support	EGBOTAT
2005-05-23	C	Final version after review	EGBOTAT
2005-09-26	PD1	Changes to support server mode operation	EGERGFT
2006-01-23	PD2	Final version after review	EGERGFT
2006-02-08	PD3	Change mandatory parameter to optional (MTTSM00009323)	EGERGFT
2006-04-10	D	Final version after review	EGERGFT
2007-01-16	PE1	Change server login prompts to Test Port parameters (MTTSM00009850)	EGERGFT
2007-01-25	PF1	Updated for TITAN R7	ETHGASZ
2009-12-08	PG1	Extending mapping result handling, detection of client disconnection, failsafe sending, filtering client linefeeds and attaching server prompts	ETIBBEN
2011-01-24	PH1	Updated to use the new API	ETHGASZ
2011-04-18	PJ1	Document modification	EDVIJUH
2012-01-13	PK1	Configurable logging implemented	EISTSND
2012-03-01	PK2	CR CR_TR00019177	ETHGASZ
2012-03-13	PK3	CR_TR00019179	EZOLMED
2012-08-08	PL1	HQ21878	ETHGASZ

1.2 About this Document

1.2.1 How to Read this Document

This is the User Guide for the Telnet test port. The Telnet test port is developed for the TTCN-3 Toolset with TITAN. This document should be read together with Product Revision Information [4] and Function Specification [5].

1.2.2 Presumed Knowledge

The knowledge of the TITAN TTCN-3 Test Executor [2] and the TTCN-3 language [1] is essential.

1.2.3 References

[1] ETSI ES 201 873-1 v3.1.1 (2005-06)
The Testing and Test Control Notation version 3. Part 1: Core Language

[2] 2/198 17-CRL 113 200 Uen
Programmer's Technical Reference for TITAN TTCN-3 Test Executor

[3] 1/1531-CRL 113 200 Uen
TITAN Installation Guide

[4] 109 21-CNL 113 320-7
Telnet Test Port for TTCN-3 Toolset with TITAN, Product Revision Information

[5] 155 17-CNL 113 320
Telnet Test Port for TTCN-3 Toolset with TITAN, Function Specification

[6] RFC 854
Telnet protocol specification

[7] RFC 857
Telnet echo option

[8] RFC 1073
Telnet Window Size Option

[9] RFC 1091
Telnet Terminal-Type Option

1.2.4 Abbreviations

ASP	Abstract Service Primitive
IUT	Implementation Under Test
SUT	System Under Test
TTCN-3	Testing and Test Control Notation version 3
TCP	Transmission Control Protocol
RTE	RunTime Environment

1.2.5 Terminology

No specific terminology is used.

1.3 System Requirements

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

- TITAN TTCN-3 Test Executor version R8A (1.8.pl0) or higher installed. For installation guide see [3]. Please note: This version of the test port is not compatible with TITAN releases earlier than R8A.

- Any operating system supported by TITAN, while only tested on Solaris.

1.4 Fundamental Concepts

The test port establishes connection between the TTCN-3 test executor and SUT or accepts connection from the SUT, and transmits/receives messages.

2 The Test Port

2.1 Overview

The Telnet test port is an adaptation between the TITAN RTE and the SUT/IUT using standard telnet protocol. The telnet protocol is described in [6]. Overview of a test system using Telnet test port can be seen on Figure 1.

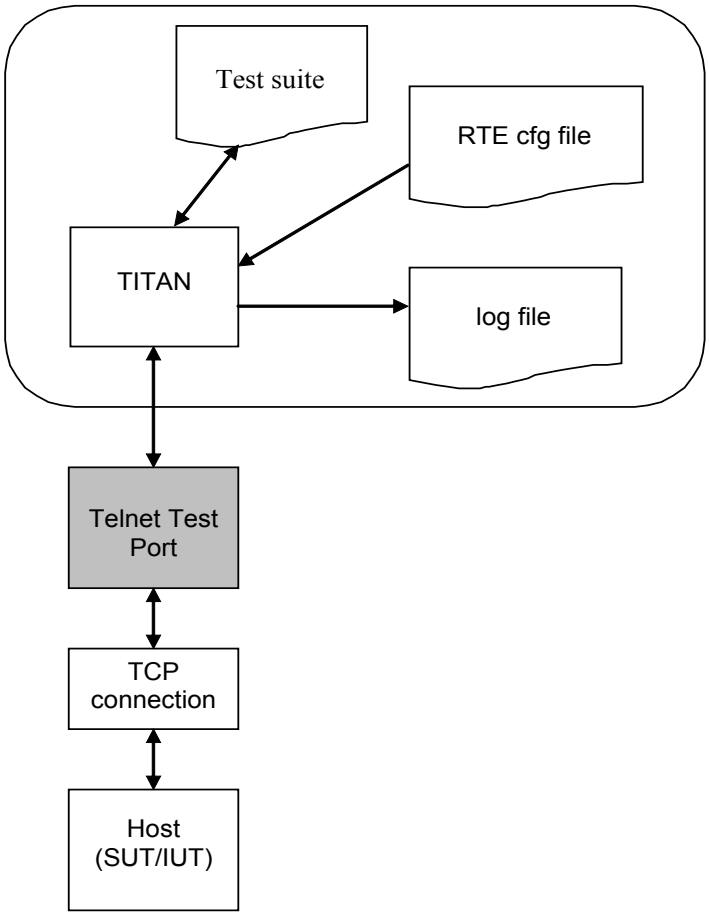

Figure 1: Overview of the Telnet Test Port

In client mode operation the Telnet test port works as a telnet terminal. It can send and receive charstring values.
The test port does telnet login automatically on executing the TTCN-3 map operation, unless the test port parameter CTRL_LOGIN_SKIPPED is set to “yes”. (This parameter is case-insensitive.)

In server mode operation the Telnet Test Port works as a telnet server. It can receive commands, in charstring format, from the SUT and it can send the answers received from the test suite back to the SUT. The Test Port automatically sends the prompt with the answer set in Test Port parameter 'CTRL_SERVER_PROMPT' (see 2.3.1).
The Test Port, on executing the TTCN-3 map operation, is waiting for connections on port set in CTRL_PORTNUM and does telnet login automatically. The connection to the client can be closed with the appropriate ASP (see [5]). (NOTE: the Test Port will never close the connection automatically to the client.)

2.2 Installation

Since the Telnet 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 Telnet test port. For more details on the installation of TTCN-3 Test Executor see the relevant section of [3].

The following is needed to use the Telnet test port:

- Copy the source files (*TELNETasp_PT.cc, TELNETasp_PT.hh*) and the definition module (*TELNETasp_PortType.ttcn*) into the directory of the test suite or create symbolic links to them.
- Import the module *TELNETasp_PortType* (TTCN-3 language, defines the test port) to the module(s) where the Telnet test port is used.
- Generate *Makefile* and compile the test suite according to [2].
- Add the required test port parameters to the runtime configuration file as described in section 2.3.1 of this document.

Do not modify the files supplied otherwise we will not be able to give technical support.

2.3 Configuration

The executable test program behaviour is determined via the run-time configuration file. This is a simple text file, which contains various sections (e.g. [TESTPORT_PARAMETERS]) after each other. The usual suffix of configuration files is ".cfg". For further information on the configuration file see [2].

2.3.1 Telnet test port parameters in the test port configuration file

General parameter:

debug

Set to "YES" if you need to debug the test port, otherwise set to "NO".

MAP_RECV_TIMEOUT

The receive timeout during map operation in milliseconds. The default value is 10000 (10 sec). Negative value disables the timeout.

Parameters used by the Telnet test port in client mode operation:

Type	Name	Description
charstring	CTRL_HOSTNAME	Name of the host (IUT)
integer	CTRL_PORTNUM	TCP port number of the host (IUT)
charstring	CTRL_USERNAME	Username used for telnet login
charstring	CTRL_PASSWORD	Password used for telnet login
charstring	CTRL_DOMAIN	Domain used for telnet login
charstring	CTRL_READMODE	When to pass an incoming message to ttcn
charstring	PROMPTx	Prompt string
charstring	REGEX_PROMPTx	Prompt containing wildcards
charstring	CTRL_LOGIN_SKIPPED	User authentication is skipped
charstring	CTRL_TERMINAL_TYPE	Telnet terminal type
charstring	CTRL_ECHO	Telnet echo option
charstring	CTRL_CRLF	Determines if CR should be sent before line feed.
Integer	CTRL_WINDOW_WIDTH	Width for telnet window size option
integer	CTRL_WINDOW_HEIGHT	Height for telnet window size option
charstring	CTRL_DETECT_SERVER_DISCONNETED	Setting test port behaviour when the server terminates the connection

charstring	CTRL_DETECT_CONNECTION_ ESTABLISHMENT_RESULT	Setting test port behaviour regarding the handling the result of a connection attempt
charstring	CTRL_CLIENT_CLEANUP_LINEFEED	Enabling the filtering of linefeeds directly preceding prompts from incoming messages
charstring	empty_echo	Enabling send empty message if the command doesn't have output
charstring	raw_regex_prompt<id>	regcomp based prompt

The parameter 'CTRL_HOSTNAME' is the hostname of the remote workstation to which the test port will connect. This parameter is mandatory.

The parameter 'CTRL_PORTNUM' is the port number of the remote host (specified by CTRL_HOSTNAME) that the test port tries to connect to. The value "7" means the test port will attempt to connect through telnet echo login. In this case the login procedure is skipped. This parameter is mandatory.

The CTRL_USERNAME and CTRL_PASSWORD parameters are mandatory and are used to authenticate the user on the remote machine.

The parameter 'CTRL_DOMAIN' is only used if the host is asking for the domain in the login procedure. This parameter is optional, but a dynamic testcase error occurs if the host requests the domain and this parameter is not set.

The parameter 'CTRL_READMODE' controls when to pass incoming messages to ttcn. The possible values are:

- "buffered"
- "unbuffered"

If it is set to "buffered" then the test port waits until the prompt arrives and every message received before the prompt is passed to ttcn in one message. If it is set to "unbuffered" then the test port passes data received between two new line sequences (or until the prompt) to ttcn.
The default value is "buffered".

The parameters PROMPT and REGEX_PROMPT are of type charstring. PROMPT is used to specify the exact prompt strings that are used on the remote machine. REGEX_PROMPT serves the same functionality but it can contain wildcards. Multiple PROMPT and REGEX_PROMPT parameters can be given in the configuration file, but each has to have a number (ID) concatenated to the parameter name (for an example see chapter 5.1). Specifying two prompts (either normal or with wildcards) with the same ID will result in the latter overriding the former. At least one PROMPT or REGEX_PROMPT parameter must be provided that is not an empty string.

The CTRL_LOGIN_SKIPPED parameter is used when user authentication (username and password) is not required. The possible values are “yes” and “no”. The default value is “no”. The values are case-insensitive.

The parameter CTRL_TERMINAL_TYPE specifies the telnet terminal-type option that is described [9].

The parameter CTRL_ECHO sets telnet echo option (see [7]). Possible values are “yes” for enabling echo and “no” for disabling it. Echo is disabled by default. The values are case-insensitive. The telnet echo option (if enabled) is sent to the server regardless of the filter settings.

The term "echo option" is used for the Telnet echo option used during the negotiation of the set up of a Telnet session as defined in [7].

Parameter CTRL_CRLF specifies whether to send CR before LF (as new line) after sending a command. It might be needed to enable this option when connecting to a host running Windows. Possible values are the same as for the parameter CTRL_ECHO.

The parameters CTRL_WINDOW_WIDTH and CTRL_WINDOW_HEIGHT specify the (initial) telnet window size option (see [8]).

The CTRL_DETECT_SERVER_DISCONNETED parameter determines the behaviour of the test port if the test port detects that the server terminates the communication. If this parameter is set to “yes”, the test port indicates the connection termination by passing an integer value 0 to the test suite. After this the test port has to be unmapped and mapped again if the user wants to use the test port in the actual test case further. If this value is set to “no”, dynamic test case error occurs when the server terminates the connection. The default value is “no”. The values are case-insensitive.

The CTRL_DETECT_CONNECTION_ESTABLISHMENT_RESULT parameter determines whether the test port should provide feedback to the user in case of a connection attempt. If this parameter is set to “yes” the test port indicates the result of the connection by passing an integer value 2 to the test suite if the attempt succeeded and 0 if failed. Thus the user can decide whether to attempt another mapping or discard further communication and neither does a failed connection attempt necessarily result in a dynamic test case error. If this value is set to “no”, dynamic test case error occurs after a failed connection attempt. The default value is “no”. The values are case-insensitive.

The table bellow summarizes the responses to a connection attempt. CDSD and CDCEF stand for CTRL_DETECT_SERVER_DISCONNETED and CTRL_ DETECT_CONNECTION_ESTABLISHMENT_RESULT, respectively. The prefix ¬ stands for a value “no”, omitting the prefix implies the opposite. A dash in each cell separates the successful and failed connection attempts; the integer above the dash means a success, whereas the integer below indicates a failure. Finally a “D” stands for a dynamic test case error and a hyphen means no response.

	CDCER	¬ CDCER
CDSD	2 / 0	- / D
¬CDSD	2 / D	- / D

The CTRL_CLIENT_CLEANUP_LINEFEED parameter determines whether the test port in client mode should filter out linefeeds directly preceding the prompt from the incoming messages received by the client. If this parameter is set to “yes”, the test port will filter out all such linefeeds. If this value is set to “no” then the messages remains unchanged. The default value is “yes”. The values are case-insensitive.
Note: only linefeeds between the real message and the prompt will be filtered. The mechanism will only search for linefeeds until the first non-linefeed character.

The empty_echo parameter determines whether the test port sends an empty charstring if the issued command has no printout. If the parameter is set to “yes” the test port sends an empty charstring to the test case before the prompt. If the value is set to “no” the test port do not send the empty charstring before the prompt.

The raw_regex_prompt<prompt_id> is used for specify the prompt string used on the remote host as regcomp supported POSIX regexp. It should have at least two subexpressions. The second subexpression selects the prompt. The regexp should match the entire buffer as it received.

Parameters used by the Telnet Test Port in server mode operation:

Type	Name	Description
integer	CTRL_PORTNUM	TCP port number to listen for incoming connections(IUT)
charstring	CTRL_USERNAME_CLIENT	Username used for telnet login
charstring	CTRL_PASSWORD_CLIENT	Password used for telnet login
charstring	CTRL_LOGIN_SKIPPED	User authentication is skipped
charstring	CTRL_MODE	Use the Test Port in client or server mode operation
charstring	CTRL_SERVER_PROMPT	Prompt string
charstring	CTRL_LOGINNAME_PROMPT	Prompt string
charstring	CTRL_PASSWORD_PROMPT	Prompt string
charstring	CTRL_SERVER_ATTACH_PROMPT	Enabling attaching of prompts to outgoing messages
charstring	CTRL_CLIENT_SERVER_DISCONNETED	Setting test port behaviour when the client terminates the connection
charstring	CTRL_SERVER_FAILSAFE_SENDING	Setting test port behaviour when a message sending fails

The parameter ‘CTRL_PORTNUM’ is the port number of the local host on which the Test Port is listening for new connection from the SUT. This parameter is mandatory.

The Test Port accepts connection from a client which sent a username and password set in the ‘CTRL_USERNAME_CLIENT’ and ‘CTRL_PASSWORD_CLIENT’ parameters.

The ‘CTRL_LOGIN_SKIPPED’ parameter is used when user authentication (username and password) is not required. The possible values are “yes” and “no”. The default value is “no”. The values are case-insensitive.

The ‘CTRL_MODE’ parameter is to choose during run time between client and server mode of operation. Acceptable values are: “client” or “server” Default value is “client”.

The parameter CTRL_SERVER_PROMPT is of type charstring and here must be specified the exact prompt string. The Test Port will send this prompt to the client after every message. One CTRL_SERVER_PROMPT parameter must be provided.

The CTRL_LOGINNAME_PROMPT parameter is the charstring to send to prompt the client for the login name. The default value is: ‘login: ’

The CTRL_PASSWORD_PROMPT parameter is the charstring to send to prompt the client for the password. The default value is: ‘password: ’

The CTRL_DETECT_CLIENT_DISCONNECTED parameter determines the behaviour of the test port if the test port detects that the client terminates the communication. If this parameter is set to “yes”, the test port indicates the connection termination by passing an integer value 3 to the test suite. If this value is set to “no”, the user of the test port in server mode will not receive any notification, and a dynamic test case error might occur (depending on the CTRL_SERVER_FAILSAFE_SENDING parameter) when a new message is attempted to be sent. The default value is “no”. The values are case-insensitive.

The CTRL_SERVER_ATTACH_PROMPT parameter determines whether the test port in server mode should attach the server prompt to every outgoing messages sent by the server. If this parameter is set to “yes”, the test port will attach the prompt to every such message. If this value is set to “no”, the message remains unchanged. The default value is “yes”. The values are case-insensitive.
Please note: enabling the parameter and setting the prompt to a “” string will result in passing a linefeed to the user. This is especially important when it is sent to the client on a connection attempt.

The CTRL_SERVER_FAILSAFE_SENDING parameter determines whether the test port in server mode should return and error or a warning in case a message sending fails. If this parameter is set to “yes”, the test port will return a warning after such sending attempts. If this value is set to “no”, , an error is returned. The default value is “no”. The values are case-insensitive.

The “empty_echo” parameter determines whether the test port sends an empty charstring if the issued command has no printout. If the parameter is set to “yes” the test port sends an empty charstring to the test case before the prompt. If the value is set to “no” the test port do not send the empty charstring before the prompt.

3 Error messages

TCP send failed. (<reason>)

Send operation failed.

Error accepting connection <reason>

Occurs only in server mode operation. The Test Port couldn't accept connection from a client, i.e. an accept() operation failed.

Error closing file descriptor <file_descriptor>. (<reason>)

The Test Port couldn't close <file_descriptor>, i.e. a close() operation failed.

Error listening on port <port_number>. (<reason>)

Occurs only in server mode operation. The Test Port couldn't listen on port <port_number>, i.e. a listen() operation failed.

Error reading from file descriptor <file_descriptor>. (<reason>)

The Test Port couldn't read from a file descriptor, i.e. a recv() operation failed.

Error binding socket to port <port_number>. (<reason>)

Occurs only in server mode operation. The Test Port couldn't bind a socket to port <port_number>, i.e. a bind() operation failed.

Missing mandatory parameter: <parameter_name>

The mandatory parameter <parameter_name> was not set in the configuration file.

Missing mandatory parameter: at least one PROMPT parameter must be provided

PROMPT parameter should be given as PROMPT<number> := "value".

Error converting string in "…" in parameter name "PROMPT…" to number.

REGEX_PROMPT parameter should be given as REGEX_PROMPT<number> := "value".

Error converting string in "…" in parameter name "REGEX_PROMPT…" to number.

PROMPT parameter must contain at least one character

REGEX_PROMPT parameter must contain at least one character

Unable to connect to <ctrl_hostname> as <ctrl_username>

The specified ctrl_username/ctrl_password pair is not valid on ctrl_hostname.

Invalid value for: <parameter>

The value of the parameter is different from the allowed ones.

Socket creation failed

There was an error creating the socket.

Unable to resolve hostname: <my_hostname>

The host name <my_hostname> cannot be resolved.

setsockopt(SO_REUSEADDR) failed

There was an error setting the socket option SO_REUSEADDR.

Error connecting <my_hostname>

There was an error connecting the host.

Error connecting <my_hostname>. Address already in use.

This error means an address is already in use. Probably a kernel error.

***** Socket error or the server closed the connection.**

The server closed the connection or an error occurred while waiting for the host to accept the login.

Unsupported port number!

The specified port number is not supported. Supported ports are 23 and 7.

*****Socket error or the server closed the connection (in Event_Handler).**

The server closed the connection.

Missing parameter CTRL_DOMAIN

The parameter CTRL_DOMAIN was not set but the host was requesting it.

Not enough memory!

A memory allocation error occurred.

Error sending window size: not connected.

There was an attempt to send the window size from TTCN-3, but the test port is not connected to the host.

Prompt parameter with wildcards shouldn’t start with “*”: “*…”

Prompt parameter with wildcards shouldn’t end with “*”: “…*”

Width of window size should be greater than 0 and less than 65536.

Height of window size should be greater than 0 and less than 65536.

Cannot convert pattern "…" to POSIX-equivalent.

Login incorrect!

Occurs only in server mode operation. The client tried to connect with an invalid username or password. Before this message the username and password sent by the client will be logged.

4 Warning messages

Dropping partial message.

This message is printed in case of unmap operation if the receive buffer is not empty.

Duplicated prompt string '<prompt_value>'

Using prompt '<prompt_value1>' that is a substring of prompt '<prompt_value2>' might cause problems.

Server refused window size negotiation.

Unable to decode Terminal Type sub-negotiation.

Send operation failed: the port is disconnected.

Send operation failed: the client has not logged in.

user_map() was called to a mapped port

Mode is not defined. Test Port will operate in client mode operation.

TCP send failed. (<reason>)

Send operation failed due to a disconnected client.

5 Examples

5.1 Configuration file

Using the client:

```
[LOGGING]
LogFile := "Telnet.log"

[EXECUTE]
Telnet.control

[TESTPORT_PARAMETERS]
system.telnet.CTRL_HOSTNAME := "my_hostname"
system.telnet.CTRL_PORTNUM := "23"
```

```
system.telnet.CTRL_USERNAME := “egbotat”
system.telnet.CTRL_PASSWORD := “abcd”
system.telnet.CTRL_READMODE := “buffered”
system.telnet.CTRL_LOGIN_SKIPPED := “no”
system.telnet.PROMPT1 := “Enter command: ”
system.telnet.PROMPT2 := “egbotat ~> ”
system.telnet.REGEX_PROMPT3 := “egbotat?*[>#]?”
system.telnet.CTRL_DETECT_SERVER_DISCONNECTED := “yes”
system.telnet.CTRL_DETECT_CONNECTION_ESTABLISHMENT_
RESULT := “yes”
system.telnet. CTRL_CLIENT_CLEANUP_LINEFEED := “no”
system.telnet.CTRL_TERMINAL_TYPE := “xterm”
system.telnet.CTRL_ECHO := “no”
system.telnet.CTRL_CRLF := “yes”
system.telnet.CTRL_WINDOW_WITDH := “80”
system.telnet.CTRL_WINDOW_HEIGHT := “24”
system.T_Client_PCO.RAW_REGEX_PROMPT1 := "^(.*)(ezolm
>)(.*)$"
```

Using the server:

```
[LOGGING]
LogFile := “Telnet.log”

[EXECUTE]
Telnet.control

[TESTPORT_PARAMETERS]
system.telnet.CTRL_PORTNUM := “23”
system.telnet.CTRL_LOGIN_SKIPPED := “no”
system.telnet.CTRL_USERNAME_CLIENT := “egergft”
system.telnet.CTRL_PASSWORD_CLIENT := “asdfg”
system.telnet.CTRL_SERVER_PROMPT := “Enter command: ”.
system.telnet.CTRL_LOGINNAME_PROMPT := “The LoginName: ”
system.telnet.CTRL_PASSWORD_PROMPT := “The Password: ”
system.telnet.CTRL_MODE := “server”
system.telnet.CTRL_SERVER_ATTACH_PROMPT := “yes”
system.telnet.CTRL_DETECT_SERVER_DISCONNECTED := “yes”
system.telnet.CTRL_SERVER_FAILSAFE_SENDING := “yes”
```