MOST

Media **O**riented **S**ystems **T**ransport

Multimedia and Control Networking Technology

MOST DTCP test recommendation
Version 1.1
11/2005

Document Version 1.1-01

MOSTCO CONFIDENTIAL

See page 3 for the terms of disclosure

MOST® COOPERATION

Legal Notice

COPYRIGHT

LICENSE DISCLAIMER

Nothing on any MOST Cooperation Web Site, or in any MOST Cooperation document, shall be construed as conferring any license under any of the MOST Cooperation or its members or any third party's intellectual property rights, whether by estoppel, implication, or otherwise.

CONTENT AND LIABILITY DISCLAIMER

MOST Cooperation or its members shall not be responsible for any errors or omissions contained at any MOST Cooperation Web Site, or in any MOST Cooperation document, and reserves the right to make changes without notice. Accordingly, all MOST Cooperation and third party information is provided "AS IS". In addition, MOST Cooperation or its members are not responsible for the content of any other Web Site linked to any MOST Cooperation Web Site. Links are provided as Internet navigation tools only.

MOST COOPERATION AND ITS MEMBERS DISCLAIM ALL WARRANTIES WITH REGARD TO THE INFORMATION (INCLUDING ANY SOFTWARE) PROVIDED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

In no event shall MOST Cooperation or its members be liable for any damages whatsoever, and in particular MOST Cooperation or its members shall not be liable for special, indirect, consequential, or incidental damages, or damages for lost profits, loss of revenue, or loss of use, arising out of or related to any MOST Cooperation Web Site, any MOST Cooperation document, or the information contained in it, whether such damages arise in contract, negligence, tort, under statute, in equity, at law or otherwise.

FEEDBACK INFORMATION

Any information provided to MOST Cooperation in connection with any MOST Cooperation Web Site, or any MOST Cooperation document, shall be provided by the submitter and received by MOST Cooperation on a non-confidential basis. MOST Cooperation shall be free to use such information on an unrestricted basis.

TRADEMARKS

MOST Cooperation and its members prohibit the unauthorized use of any of their trademarks. MOST Cooperation specifically prohibits the use of the MOST Cooperation LOGO unless the use is approved by the Steering Committee of MOST Cooperation.

SUPPORT AND FURTHER INFORMATION

For more information on the MOST technology, please contact:

MOST Cooperation
Administration
P. O. Box 4327
D-76028 Karlsruhe
Germany

Tel: (+49) (0) 721 966 50 00
Fax:(+49) (0) 721 966 50 01
E-mail: contact@mostcooperation.com
Web: www.mostcooperation.com

MOST DTCP test recommendation

Bibliography

Number	Document
[1]	MOST Specification Framework
[2]	MOST Specification
[3]	MOST High Protocol Specification
[4]	MOST NetServices “Basic Layer”; User Manual and Specification
[5]	MOST NetServices “Application Socket”; User Manual and Specification
[6]	FOT Datasheet
[7]	MOST Transceiver Datasheet
[8]	MOST Function Catalogue
[9]	MOST Specification Of Physical Layer

Document History

Changes to MOST DTCP test recommendation

Change Ref.	Section	Changes
Draft 0.1	All	1st draft
Draft 0.2	All	- Removed experimental set-up 1 - Removed the content key update test. - Removed old style authentication. - Removed retry test. - Added remarks about optional / mandatory character of the tests in the core functions section. - Added a chapter: conditions to claim compliance (TBD.) - Added a compliance matrix - Added a table of figures - Added the generic isochronous packet transfer - Updated the chapter “Scope of the document”. - Updated the full authentication procedure. - Updated the restrictive authentication procedure - Updated the generic synchronous format test - Updated the error sections. - Updated the “Test output” chapter.
Draft 0.3	All	- Copied chapter “terms and abbreviations” from MOST Copy Protection Scheme specification. - Updated the scope chapter. - Updated the chapter “Conditions to claim compliance” - Removed the error case “decode error” - Removed speculative authentication.
Draft 0.6	All	- Adapted to MOST Co template - Corrected the authentication sequence test - Removed SourceInfo.Get and SourceInfo.Status from the authentication tests - Added cipher error test - Updated the error parameters array
Draft 0.7	All	- Added: DTCP_StartResult.Result in authentication flows. - Added: example for SRM update. - Added source error and sink error tests. - Added requirements for devices under test and a tester device - Added definition of test data for the packing and unpacking tests. - Updated the error chapter with comments on warnings. - Updated the error section of the “Core functions” chapter. - Updated the experimental set up. - Removed sink and source software and hardware errors. - Removed cipher error test. - Modified the packing / unpacking section. - Modified the test of error cases.
Draft 0.8	Data packing and unpacking	- Updated the example schematics
	Authentication sequences	- Added a warning message: separated the error cases “command rejected” and “wrong key”.
Draft 0.9	SRM sequence	- Made mandatory
	Data packing / unpacking	- Test of encryption frame size made optional (already part of DTLA specification tests). - Example test data patterns modified. - Test of data decoding by the sink made optional (for development only).

Change Ref.	Section	Changes
	Error behavior	- DTCP_Control.Error not tested anymore, - DTCP_Control.Response(rejected) tested instead - Added an optional test for DTCP_Status
	Authentication	- Added optional tests for DTCP_CipherStatus function - Made the cipher error tests optional
V1.0	Error behavior	- Two errors corrected in the sink error behavior sequence.
V1.1	Test description	- Added preliminary note for payloads bigger than 128bytes
	Error behavior	- Changed back the DTCP_Control.Result(Error) to DTCP_Control.Error(rejected).
V1.1-01	N/A	Changed document template.

Table of Contents

1 INTRODUCTION 9

1.1 Scope and Prerequisites 9
1.2 Conditions to Claim Compliance 9
1.3 Terms and Abbreviations 9
1.4 References Documents 10

2 CORE FUNCTIONS 10

2.1 Authentication 10
2.2 Data Packing and Unpacking 10
2.3 Error Cases 11

3 PROCEDURE OF MOST-DTCP COMPLIANCE TEST 11

3.1 Test Output / Error Codes 11
3.2 Experimental Set-up 13
3.2.1 Device under Test: Required Features 13
3.2.2 Test Equipment: Required Features 14
3.2.2.1 Tester Device 14
3.2.2.1.1 Required Features for Authentication Test 14
3.2.2.1.2 Required Features for Packing / Unpacking Test 14
3.2.2.1.3 Required Features for Error Test 14
3.2.2.2 Test Content 14
3.2.2.2.1 Synchronous Data 15
3.2.2.2.2 Packet Data 16
3.3 Test Descriptions 16
3.3.1 Authentication 17
3.3.1.1 Full Authentication 17
3.3.1.2 Restricted Authentication 20
3.3.1.3 Content Key Request 23
3.3.1.4 SRM Update 25
3.3.1.4.1 Example 27
3.3.2 Data Packing and Unpacking 28
3.3.2.1 Generic Synchronous Format 28
3.3.2.1.1 Example 30
3.3.2.2 Generic Packet Data Format 31
3.3.3 Error Cases 32
3.3.3.1 Error Behavior (Device under Test is a Source) 32
3.3.3.1.1 Example: Invalid Subfunction Call 33
3.3.3.2 Error Behavior (Device under Test is a Sink) 34

4 COMPLIANCE MATRIX 36

5 APPENDIX A: LIST OF FIGURES 37

6 APPENDIX B: LIST OF TABLES 38

This page is intentionally left blank.

1 Introduction

1.1 Scope and Prerequisites

This recommendation defines tests to be passed to ensure interoperability of MOST-DTCP enabled devices.

This document is not a DTCP compliance test. Each equipment checked versus this compliance document is expected to comply with DTLA's DTCP specification.

It is a prerequisite that the device under test complies with the MOST specification. Therefore the MOST channel allocation and connections procedures shown in the MOST-DTCP specification are not in the scope of this document.

1.2 Conditions to Claim Compliance

To claim compliance, a device or system has to:

- pass the "Authentication" test for at least one of the authentication procedures (full and/or restricted),
- pass the "Data packing and unpacking" test for each data type and transport type the device claims compliance with,
- behave according to the "Error cases" tests in case of error.

A compliance matrix document (example template in chapter 4) shall be used to report test results.

Note:
A device may comply with optional features from the MOST Content Protection Scheme specification if it passes the corresponding tests. In that case, the optional steps supported in the different test items have to be reported in the compliance matrix. However, actual implementation of these optional features in the final system should be subject to an agreement between the different parties involved.

1.3 Terms and Abbreviations

Controller: the device controlling the protocol
DTCP: Digital Transmission Content Protection
DUT: device under test
MOST: Media Oriented System Transport
Sink: target of a data transfer
Source: origin of a data transfer

1.4 References Documents

- MOST Content Protection Scheme Specification – DTCP Implementation, MOST Co., rev. 2.1
- MOST Stream Transmission Specification, MOST Co, rev. 1.1
- DTCP volume 1 supplement B: mapping DTCP to MOST, DTLA, rev. 1.2a
- Digital Transmission Content Protection Specification, 5C, rev. 1.3

2 Core Functions

2.1 Authentication

Test item	Remarks	Test procedure
Full authentication sequence	The SRM sequence test and the content key request test are included in this test. A device claiming compatibility must pass at least one of both authentication sequence tests.	Full authentication
Restricted authentication	The SRM sequence test and the content key request test are included in this test. A device claiming compatibility must pass at least one of both authentication sequence tests.	Restricted authentication
SRM update	Even if official SRM content was not released yet by the DTLA, the system under test needs to support the procedure anyway.	SRM update
Content key request	This test is called from the full authentication sequence. However, a device may request the content key independently from the authentication.	Content key request
Device status query	Tests the DTCP_CipherStatus method for state machine's status and exchange keys availability.	Full / Restricted authentication

Table 2-1: Authentication Test Functions

2.2 Data Packing and Unpacking

Test item	Remarks	Test procedure
Generic synchronous format	The test procedure describes a global method to test the correct transmission of synchronous data, according to the MOST Stream specification (chapter MOST-DTCP).	Generic synchronous format
Isochronous transfer	This test procedure describes a global method to test the correct transmission of streamed data according to the MOST stream specification.	Generic packet format

Table 2-2: Data Packing / Unpacking Test Functions

2.3 Error Cases

Test item	Remarks	Test procedure
Command rejection	None	Error behavior
Reception of an error message	None	Error behavior
Cipher error	Test the DTCP_CipherStatus routine for cipher status parameters.	Error behavior
Error status query	Tests the DTCP_Status method.	Error behavior

Table 2-3: Error Cases Test Functions

3 Procedure of MOST-DTCP Compliance Test

3.1 Test Output / Error Codes

One or more error codes represent the output of each test. This output shall be displayed using the following format:
"TestGroup.Test.Error(parameters)".

Elements of the output format:

- TestGroup is a number referring to the sub-chapter of the chapter 3.3 Test Descriptions
- Test is a number referring to the test item within the chapter 3.3
- Error is a number describing the type of error (see the error parameters array below in Table 3-1)
- The parameters list specifies the conditions of the error (for example if a parameter was wrong, what parameter exactly).

An error causes the termination of the test in progress.

In the case when the error is caused by a parameter which is not in the scope of this specification (DTLA time out for example or a rejected message) a warning will be issued. The behavior of the system in case of a warning condition is not defined. Warnings are provided as a mean for third party test labs to report such problems that are not covered by the specification but will most likely block any further testing.

x.x.Error	Meaning	Parameters	Comment
x.x.0	SUCCESS	None	
x.x.1	ERROR: Wrong address.	Test step	The address part of the MOST message is wrong.
x.x.2	ERROR: Wrong command.	Test step	The MOST-DTCP command seen on the bus is not the command expected according to the specification.
x.x.3	ERROR: Invalid parameter(s)	Test step, [wrong parameters]	One ore more parameters in the MOST-DTCP command body are wrong.
x.x.4	ERROR: SRM triggered without reason.	None	
x.x.5	ERROR: wrong packet header	Test step, [wrong parameters]	
x.x.6	ERROR: wrong info bytes	Test step, [info in, info out]	
x.x.7	ERROR: wrong packet size	Format	
x.x.8	ERROR: wrong encryption frame size	Stream type	
x.x.9	ERROR: wrong data out	Channel, pattern,[data in, data out]	
x.x.10	ERROR: wrong command not rejected	Command, [parameters list]	
x.x.11	ERROR: cipher error, wrong handling	None	
x.x.20	WARNING: DTLA time out.	Step	
x.x.21	WARNING: DTCP error, a wrong key was used	Step	In that case it is likely that the system will detect the error after sending the DTCP_Control.Result(response) message. The step to return is the step where the wrong key was first used by the DUT.
x.x.22	WARNING: DTCP error, the device rejects the command.	Step	A device used a command that is not recognized by the other side.

Table 3-1: Error Parameters Values

3.2 Experimental Set-up

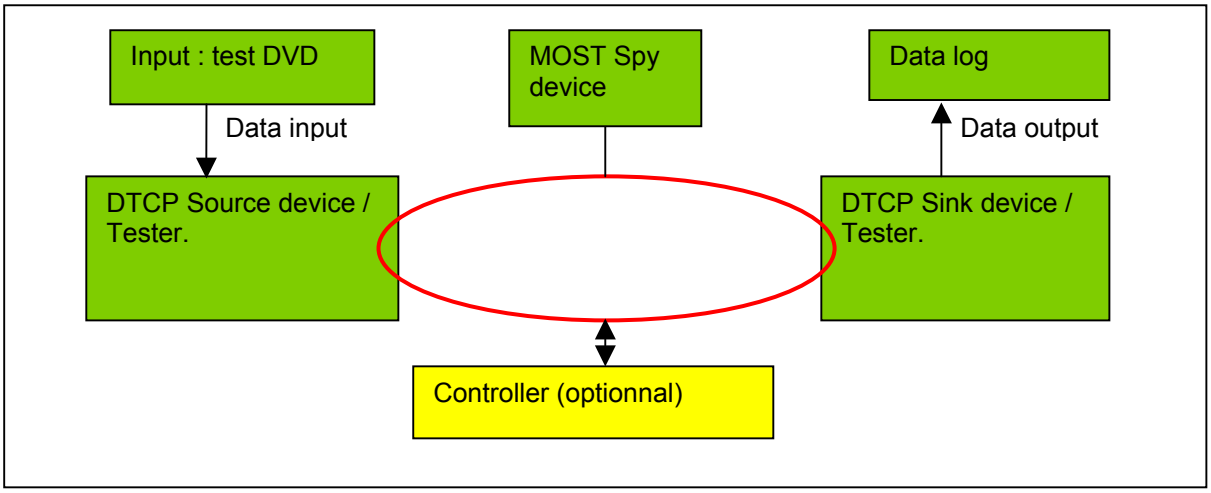

Figure 3-1: General Test Set-up

- An external controller has to be provided if necessary to trigger the AKE sequence and to allocate MOST channels.
- If the device under test is a source, the sink has to be a tester device.
- If the device under test is a sink, the source has to be a tester device.

3.2.1 Device under Test: Required Features

- For source devices: no peculiar requirement.
- For sink devices:

In order to implement the optional data packing / unpacking test, the sink devices under test have to provide access to decoded data. In that case, and in order not to break the DTLA rules (no unencrypted data accessible from outside the system), the sink devices under test have to be specially prepared and reserved for the test, providing an access to the decoded data. Moreover the test must run using the development key set. Therefore the tests for correct decoding of data in the sink are only suited for the development stage, and not required for test houses (optional).

3.2.2 Test Equipment: Required Features

3.2.2.1 Tester Device

A tester device is necessary for the tests, for example to simulate error conditions, or to test SRM version changes.

3.2.2.1.1 Required Features for Authentication Test

The tester device must provide a way to change its SRM version number.

3.2.2.1.2 Required Features for Packing / Unpacking Test

The testing equipment is able to pack and unpack synchronous data and isochronous data. The testing equipment logs data received and decoded, when used to simulate a sink. The testing equipment logs data output decoded by the sink, when used to simulate a source (Optional, for development only). The testing equipment compares received data versus sent data (Optional, for development only).

3.2.2.1.3 Required Features for Error Test

The tester can generate a MOST command with an invalid parameter (Example: command ID 0xFF). The tester can generate DTCP_Control.Error messages on command. The tester can generate DTCP_CipherStatus.CipherError messages on command.

3.2.2.2 Test Content

To run the tests for data packing and unpacking, test content has to be provided. To test a source (data packing), this content can be provided on a test DVD. To test a sink (data unpacking), this content is prepared and sent by the tester device (Optional, for development only).

3.2.2.2.1 Synchronous Data

3.2.2.2.1.1 Device under Test is a Source: Test Data Fed in the Source.

- ***Minimum requirement for the device***

Packing at least stereo audio with 16 bits per sample.

- ***Test of the packet length***

All channels set to value 0.

- ***Test of the encryption frame size (optional test)***

Following byte values are allocated in that order to the different channels to help differentiate the encryption frames on the bus.

Bytes 0 to 31: 0x00 (5 * encryption frame size)
Bytes 32 to 63: 0xFF (5 * encryption frame size)
Bytes 64 to 95: 0x00 (…)
etc.

Example*:*
Stereo (2 channels), 16 bits per sample (minimum requirement). This pattern leads to the appearance on the bus of alternating data "packets" of 5-encryption frame size length, form which the encryption frame size can be derivated.

	Header ch.	Info ch.	Data [0]	Data [1]	Data [2]	Data [3]
Frame n		Info[0]	0	0	0	0
Frame n + 1		Info[1]	0	0	0	0
…		…	…	…	…	…
Frame n + 7		Info[7]	0	0	0	0
Frame n + 8		Info[0]	0xFF	0xFF	0xFF	0xFF
Frame n + 9			0xFF	0xFF	0xFF	…

3.2.2.2.1.1.1 Device under Test is a Sink: Test Data Sent by the Testing Device (Optional Test).

The testing device generates data arranged in:

- Stereo (two channels), 16 bit per sample,
- 6 channels, 16 bit per sample
- 6 channels, 24 bits per sample.

For each channel, the whole range of possible samples must be tested. This means

- for 16 bit per sample formats: 65 536 values,
- for 24 bits per sample format: 16 777 216 values.

***Example**:*
Stereo (2 channels), 16 bits per sample (minimum requirement).

	Header ch.	Info ch.	Data [0]	Data [1]	Data [2]	Data [3]
Frame		Info[0]	0x00	0x00	0x00	0x00
Frame 1		Info[1]	0x00	0x01	0x00	0x01
Frame 2		…	0x00	0x02	0x00	0x02
Frame 3			0x00	0x03	0x00	0x03
Frame 4			0x00	0x04	0x00	0x04
…			…	…	…	…
Frame n – 1			0xFF	0xFE	0xFF	0xFE
Frame n			0xFF	0xFF	0xFF	0xFF

3.2.2.2.2 Packet Data

3.2.2.2.2.1 Device under Test is a Source: Test Data Fed in the Source.

- ***Minimum requirement for the device***

Packing at least stereo audio, 16 bits per sample.

- ***Test of the packet length***

Stream of “0” s.

- ***Test of the encryption frame size (optional test)***

Data streamed is all 0 for the first encryption frame, all 0xFF for the second, all 0 for the third, etc.

3.2.2.2.2.1.1 Device under Test is a Sink: Test Data Sent by the Testing Device (Optional Test).

The testing device generates a stream containing all values between 0x00000000 and 0xFFFFFFFF (2^32 values).

3.3 Test Descriptions

Preliminary note:

According to the public DTLA documents, the payload in a control packet is limited to 128 bytes. Bigger payloads will be fragmented in 128 bytes long packets that will then be sent consecutively. This leads to looping sequences of DTCP_StartControl.StartResult commands and DTCP_StartControl.Result answers until a complete payload is transmitted.

To simplify the reading, this document always summarizes these potential looping sequences with one single exchange “Command / Response”. The corresponding test steps are of course only successful if all the expected payload packets have been successfully transmitted.

3.3.1 Authentication

3.3.1.1 Full Authentication

Name of test	Full authentication
Reference documents	MOST Content protection scheme rev 1.1-09 DTCP supplement B: mapping DTCP to MOST DTCP specification volume 1 (v1.3), informational version
Reference to core function	Full authentication sequence SRM update Content key request
Motivation, Matter of interest	This test procedure checks the full authentication procedure. Test of SRM update and content key request are included.
Device type	DTCP source/sink
General / Prerequisite	The test shall be ran for each EMI mode (copy-never, copy once etc.) supported. The test shall be ran under following conditions: - Both SRM versions are same. - SRM version of the source is newer than SRM version of the sink - SMR version of the sink is newer than SRM version of the source

Test sequence/Test flow-chart		
Action		Result
Step 1 (Opt.)	Controller to the DUT: DTCP_CipherStatus.Get()	Go to next step.
Step 2 (Opt.)	DUT to the controller: DTCP_CipherStatus(…) Authentication state should be "Unauthenticated"	YES, go to next step Wrong address, err. 1.1.1 Wrong command, err. 1.1.2 Invalid parameter, err. 1.1.3.
Step 3	Controller to the sink: DTCP_StartProcess.StartResult	See step 2.
Step 4 (Opt)	Sink to the controller: DTCP_StartProcess.Processing[1]	YES, go to next step Wrong address, err. 1.1.1 Wrong command, err. 1.1.2 Invalid parameter, err. 1.1.3 Time out, warning 1.1.20
Step 5 (Opt.)	Controller to the DUT: DTCP_CipherStatus.Get()	Go to next step.
Step 6 (Opt.)	DUT to the controller: DTCP_CipherStatus(…) Authentication state should be "Full authentication"	See step 2.
Step 7	Sink to the source DTCP_Control.StartResult(CHALLENGE)	YES, go to next step Wrong address, err. 1.1.1 Wrong command, err. 1.1.2 Invalid parameter, err. 1.1.3 Time out, warning 1.1.20 Wrong key, warning 1.1.21

[1] Message appears every 100ms until the end of the sequence.

Step 8 (Opt)	Source to the sink (Optional): DTCP_Control.Processing	See step 4.
Step 9	Source to the sink: DTCP_Control.Result(response)	YES, go to next step Wrong address, err. 1.1.1 Wrong command, err. 1.1.2 Invalid parameter, err. 1.1.3 Time out, warning 1.1.20 Wrong key, warning 1.1.21 Rejected, warning 1.1.22
Step 10	Source to the sink: DTCP_Control.StartResult(CHALLENGE)	See step 7.
Step 11 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	See step 4.
Step 12	Sink to the source: DTCP_Control.Result(response)	See step 9.
Step 13	Source to the sink: DTCP_Control.StartResult(RESPONSE)	See step 7.
Step 14 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	See step 4.
Step 15	Sink to the source: DTCP_Control.Result(response)	See step 9.
Step 16	Sink to the source DTCP_Control.StartResult(RESPONSE)	See step 7.
Step 17 (Opt)	Source to the sink (Optional): DTCP_Control.Processing	See step 4.
Step 18	Source to the sink: DTCP_Control.Result(response)	See step 9.
Step 19	Source to the sink: DTCP_Control.StartResult(EXCHANGE_KEY)	See step 7.
Step 20 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	See step 4.
Step 21	Sink to the source: DTCP_Control.Result(response)	See step 9.
Step 22	Go to SRM test	SRM Successful, go to next step SRM Unsuccessful, see specific SRM error code.

Step 23	Sink to the controller: DTCP_StartProcess.Result	See step 4.
Step 24 (Opt., if DUT is a source)	Sink to the source: DTCP_CipherStatus.Get(…)	Go to next step.
Step 25 (Opt., if DUT is a source)	Source to the sink DTCP_CipherStatus.Status(…) Check available exchange key(s) and verify versus the expected status. Check authentication state: should be "Authenticated"	See step 2.
Step 26	All supported EMI modes successfully tested?	Yes, go to next step. No, go to step 1.
Step 27	All SRM cases successfully tested ?	Yes, go to next step. No, go to step 1.
Step 28	Go to Content Key Request test.	Content key request successful, end of test, DUT ok (opt. go to next step). Content key request unsuccessful: return specific error code.
Step 29 (Opt., if DUT is a source)	Sink to the source: DTCP_CipherStatus.Get(…)	Go to next step.
Step 30 (Opt., if DUT is a source)	Source to the sink DTCP_CipherStatus.Status(…) Check authentication state: should be "Authenticated"	YES, end of test DUT ok. Wrong address, err. 1.1.1 Wrong command, err. 1.1.2 Invalid parameter, err. 1.1.3.
Test results		
DUT ok 1.1.0	Success	
DUT error 1.1.1	Message sent to the wrong node. Return 1.1.1(step)	
DUT error 1.1.2	Wrong command Return 1.1.2(step)	
DUT error 1.1.3	One or more invalid parameters. Return 1.1.3(step,[parameter])	
DUT warning 1.1.20	Process timed out. Return 1.1.20(step)	
DUT warning 1.1.21	Wrong key. Return 1.1.21(step)	
DUT warning 1.1.22	Command rejection. Return 1.1.22(step)	

Table 3-2: Full Authentication Test Procedure

3.3.1.2 Restricted Authentication

Name of test	Restricted authentication
Reference documents	MOST Content protection scheme rev 1.1-09 DTCP supplement B: mapping DTCP to MOST DTCP specification volume 1 (v1.3), informational version
Reference to core function	Restricted authentication sequence. SRM update Content key request
Motivation, Matter of interest	This test procedure checks the restricted authentication procedure. SRM sequence and content key request tests are included.
Device type	DTCP source/sink
General / Prerequisite	The test should be ran for each EMI mode (copy-never, copy once etc.) supported. This test shall be ran under following conditions: - Both SRM versions are same. - SRM version of the source is newer than SRM version of the sink - SMR version of the sink is newer than SRM version of the source

Test sequence/Test flow-chart		
Action		Result
Step 1 (Opt.)	Controller to the DUT: DTCP_CipherStatus.Get()	Go to next step.
Step 2 (Opt.)	DUT to the controller: DTCP_CipherStatus(…) Authentication state should be "Unauthenticated"	YES, next step Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3
Step 3	Controller to the sink: DTCP_StartProcess.StartResult	See step 2.
Step 4 (Opt)	Sink to the controller: DTCP_StartProcess.Processing[2]	YES, next step Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3 Time out, warning 1.2.20
Step 5 (Opt.)	Controller to the DUT: DTCP_CipherStatus.Get()	Go to next step.
Step 6 (Opt.)	DUT to the controller: DTCP_CipherStatus(…) Authentication state should be "Restricted authentication"	See step 2.
Step 7	Sink to the source DTCP_Control.StartResult(CHALLENGE)	YES, next step Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3 Time out, warning 1.2.20 Wrong key, warning 1.2.21.
Step 8 (Opt)	Source to the sink (Optional): DTCP_Control.Processing	See step 4.

[2] This message appears every 100ms until the end of the sequence.

Step 9	Source to the sink: DTCP_Control.Result(response)	YES, next step Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3 Time out, warning 1.2.20 Wrong key, warning 1.2.21 Rejected, warning 1.2.22
Step 10	Source to the sink: DTCP_Control.StartResult(CHALLENGE)	See step 7.
Step 11 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	See step 4.
Step 12	Sink to the source: DTCP_Control.Result(response)	See step 9.
Step 13	Sink to the source DTCP_Control.StartResult(RESPONSE)	See step 7.
Step 14 (Opt)	Source to the sink (Optional): DTCP_Control.Processing	See step 4.
Step 15	Source to the sink: DTCP_Control.Result(response)	See step 9.
Step 16	Source to the sink: DTCP_Control.StartResult(EXCHANGE_KEY)	See step 7.
Step 17 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	See step 4.
Step 18	Sink to the source: DTCP_Control.Result(response)	See step 9.
Step 19	Go to SRM test	SRM Successful, go to next step SRM Unsuccessful, see specific SRM error code.
Step 20	Sink to the controller: DTCP_StartProcess.Result	See step 4.
Step 21 (Opt., if DUT is a source)	Sink to the source: DTCP_CipherStatus.Get(…)	Go to next step.
Step 22 (Opt., if DUT is a source)	Source to the sink DTCP_CipherStatus.Status(…) Check available exchange key(s) and verify versus the expected status. Check authentication state: Should be “Authenticated”	See step 2.

Step 23	All EMI modes successfully tested?	Yes, go to next step. No, go to step 1.
Step 24	All SRM cases successfully tested ?	Yes, go to next step. No, go to step 1.
Step 25	Go to Content Key Request test.	Content key request successful, end of test, DUT ok (opt. go to next step). Content key request unsuccessful: return specific error code.
Step 26 (Opt., if DUT is a source)	Sink to the source: DTCP_CipherStatus.Get(…)	Go to next step.
Step 27 (Opt., if DUT is a source)	Source to the sink DTCP_CipherStatus.Status(…) Check authentication state: Should be “Authenticated”	YES, end of test DUT ok. Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3

Test results	
DUT ok 1.2.0	Success
DUT error 1.2.1	Message sent to the wrong node. Return 1.2.1(step)
DUT error 1.2.2	Wrong command Return 1.2.2(step)
DUT error 1.2.3	One or more invalid parameters. Return 1.2.3(step,[parameter])
DUT warning 1.2.20	The answer was not sent in time. Return 1.2.20(step(s))
DUT warning 1.2.21	Wrong key. Return 1.2.21(step)
DUT warning 1.2.22	Command rejected. Return 1.2.22(step)

Table 3-3: Restricted Authentication Test Procedure

3.3.1.3 Content Key Request

Name of test	Content Key request
Reference documents	MOST Content protection scheme rev. 1.1-09 DTCP supplement B: mapping DTCP to MOST DTCP specification volume 1 (v1.3), informational version
Reference to core function	Content key request Full authentication
Motivation, Matter of interest	This test will usually not run independently from the authentication test. However, some devices might support that feature. Whether data can be correctly played back after this sequence is not in the scope of this test. Please check the data packing / unpacking section for that purpose.
Device type	DTCP source/sink
General / Prerequisite	There is already an exchange key available for the communication between source and sink.

Test sequence

Action		**Result**
Step 1 (Opt)	If an external controller is used. Controller to the source: DTCP_Info.Get	YES, go to next step. Wrong address, err. 1.3.1 Wrong command, err. 1.3.2 Invalid parameter, err. 1.3.3
Step 2 (Opt)	If an external controller is used. Source to the controller: DTCP_Info.Status	YES, go to next step Wrong address, err. 1.3.1 Wrong command, err. 1.3.2 Invalid parameter, err. 1.3.3 Time out, warning 1.3.20
Step 3	Controller to the sink DTCP_ContentKeyProcess.StartResult	See step 2.
Step 4 (Opt)	Sink to the controller: DTCP_ContentKeyProcess.Processing[3]	See step 2.
Step 5 (Opt., if DUT is a source)	Sink to the source: DTCP_CipherStatus.Get(…)	Go to next step.
Step 6 (Opt., if DUT is a source)	Source to the sink DTCP_CipherStatus.Status(…) Check authentication state: Should be “Send content channel key”	See step 2.
Step 7	Sink to the source DTCP_Control.StartResult(CONTENT_KEY_REQ)	YES, go to next step Wrong address, err. 1.3.1 Wrong command, err. 1.3.2 Invalid parameter, err. 1.3.3 Time out, warning 1.3.20 Wrong key, warning 1.3.21
Step 8 (Opt)	Source to the sink (Optional): DTCP_Control.Processing	See step 2.

[3] This message appears every 100ms until the end of the sequence.

Step 9	Source to the sink: DTCP_Control.Result(response)	YES, got to next step Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3 Time out, warning 1.2.20 Wrong key, warning 1.2.21 Rejected, warning 1.2.22
Step 10	Sink to the source: DTCP_ContentKeyProcess.Result	YES, end of test DUT Ok. Wrong address, err. 1.2.1 Wrong command, err. 1.2.2 Invalid parameter, err. 1.2.3 Time out, warning 1.2.20
Test results		
DUT ok 1.3.0	Success	
DUT error 1.3.1	Message sent to the wrong node. Return 1.3.1(step)	
DUT error 1.3.2	Wrong command. Return 1.3.2(step)	
DUT error 1.3.3	One or more invalid parameters. Return 1.3.3(step,[parameter])	
DUT warning 1.3.20	Process timed out. Return 1.3.20(step).	
DUT warning 1.3.21	Wrong key. Return 1.3.21(step).	
DUT warning 1.3.22	Command rejected. Return 1.3.22(step)	

Table 3-4: Content Key Request Test Procedure

3.3.1.4 SRM Update

Name of test	SRM update
Reference documents	MOST Content protection scheme rev. 1.1-09 DTCP specification volume 1 (v1.3), informational version
Reference to core function	SRM update
Motivation, Matter of interest	This test checks the correct behavior of the device under test during the SRM update.
Device type	Source / sink
General / prerequisite	This test shall run in the following different cases: - Both SRM versions are same. - SRM version of the source is newer than SRM version of the sink - SMR version of the sink is newer than SRM version of the source

Test sequence

Action		Result
Step 1	If SRM (DUT) newer than SRM(tester) go to step 2 Else go to step 5	NA
Step 2	Source to the sink: DTCP_Control.StartResult(SRM)	YES, got to next step Wrong address, err. 1.4.1 Wrong command, err. 1.4.2 Invalid parameter, err. 1.4.3 Time out, warning 1.4.20 Wrong key, warning 1.4.21.
Step 3 (Opt.)	Sink to the source (optional): DTCP_Control.Processing	YES, got to next step Wrong address, err. 1.4.1 Wrong command, err. 1.4.2 Invalid parameter, err. 1.4.3 Time out, warning 1.4.20
Step 4	Sink to the source: DTCP_Control.Result(response)	YES, DUT Ok, end of test. Wrong address, err. 1.4.1 Wrong command, err. 1.4.2 Invalid parameter, err. 1.4.3 Time out, warning 1.4.20 Wrong key, warning 1.4.21. Rejected, warning 1.4.22
Step 5	If SRM(sink) newer than SRM(source) go to step 6 Else go to step 9	NA
Step 6	Sink to the source: DTCP_Control.StartResult(SRM)	See step 2.
Step 7 (Opt.)	Source to the sink (optional): DTCP_Control.Processing	See step 3.
Step 8	Source to the sink: DTCP_Control.Result(response)	See step 4.
Step 9	Identical SRM versions: No SRM message.	YES, DUT OK, end of test. NO, wrong behavior, err. 1.4.5.

Test results	
DUT ok 1.4.0	Success
DUT error 1.4.1	Message sent to the wrong node. Return 1.4.1(step)
DUT error 1.4.2	Wrong command Return 1.4.2(step)
DUT error 1.4.3	One or more invalid parameters. Return 1.4.3(step,[parameter])
DUT error 1.4.5	SRM sequence triggered but the SRM version numbers are identical. Return 1.4.5.
DUT warning 1.4.20	The answer was not sent in time. Return 1.4.20(step)
DUT warning 1.4.21	Wrong key. Return 1.4.21(step)
DUT warning 1.4.22	Command rejected. Return 1.4.22(step)

Table 3-5: SRM Update Test Procedure

3.3.1.4.1 Example

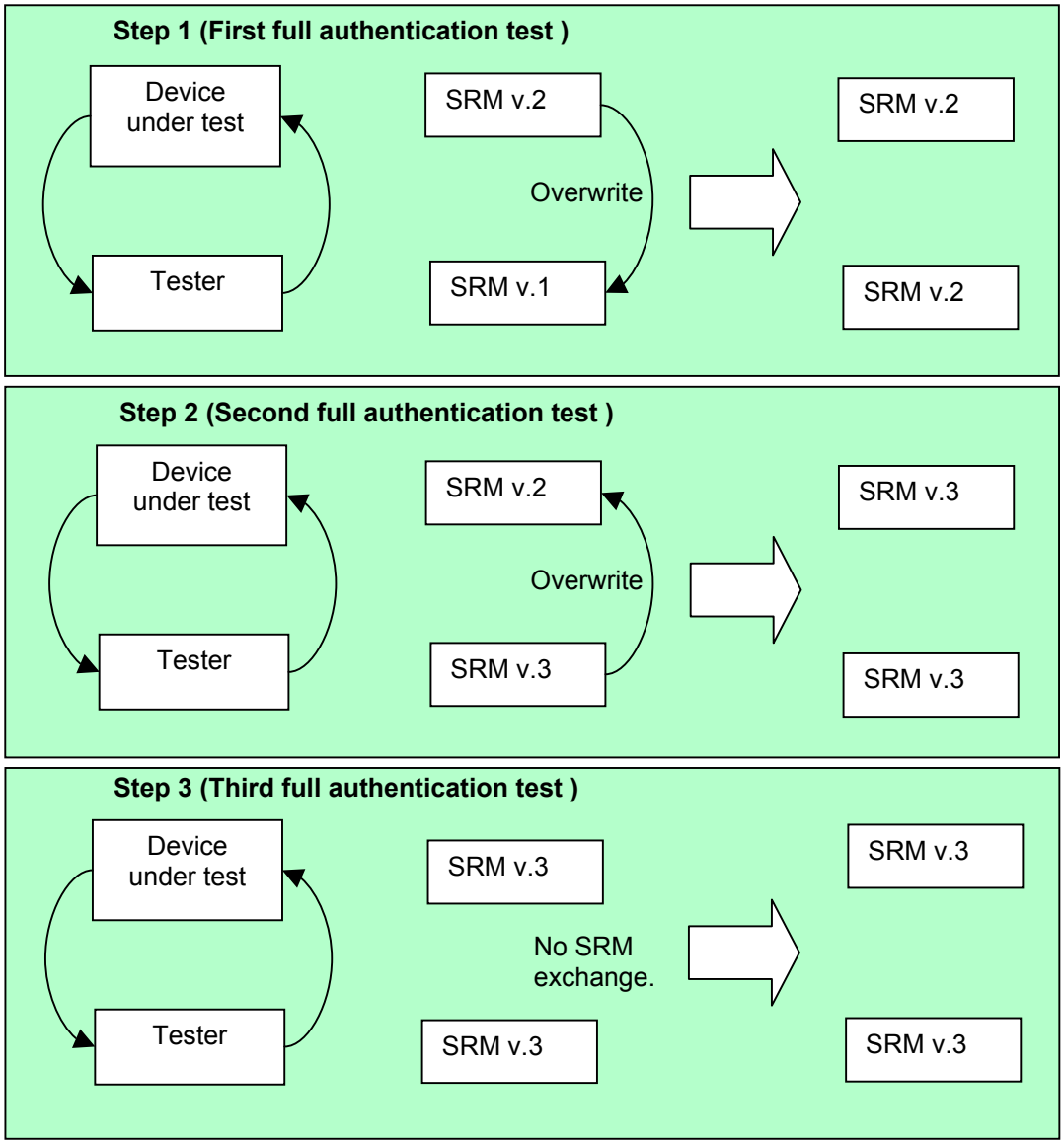

Figure 3-2: SRM Test Example

3.3.2 Data Packing and Unpacking

3.3.2.1 Generic Synchronous Format

Name of test	Generic synchronous format
Reference documents	MOST protection scheme rev 1.1-09 chapter 4.2 MOST stream transmission rev. 1.06 chapter 6.2 DTCP volume 1 supplement B.
Reference to core function	Generic synchronous format
Motivation, Matter of interest	The test is used to verify the correct format of MOST packets on the bus.
Device type	Source / Sink
General/prerequisite	**Test content:** see chapter „3.2.2.2 Test Content“ for a description of the data sent or received by the device under test. **Data channels:** the test is described for one channel. Each data channel used in the application shall be tested. **Test packet formats:** The format and content of the test packets is defined in the MOST stream transmission, chapter 6.2 and in the MOST protection scheme chapter 4.2. A device transporting a given content type must pass this test with the corresponding packet format. **Note:** the test for the sink data-unpacking is only possible with specially prepared development devices, therefore not suited for test houses, and consequently optional.

Test sequence		
Action		Result
Step 1	If the device under test is a source, go to step 2 else go to step 7.	Go to next step.
Step 2	A communication is initiated between source and sink.	Go to next step.
Step 3	Verification of packet size. Source encrypts test data.	Synchronous packets on the bus. Correct format: go to next step. Wrong header, err. 2.1.5 Wrong packet size, err. 2.1.7
Step 4 (Opt., if supported by the tester)	Verification of encryption frame size. Source encrypts test data.	Synchronous packets on the bus Correct format, go to next step. Wrong header, err. 2.1.9 Wrong encryption frame size, err. 2.1.8
Step 5	Verification of info bytes received by the tester.	Correct info bytes, end of test, DUT Ok. Wrong format of info bytes, err. 2.1.5
Step 6	All available formats tested?	Yes, end of test, DUT ok (opt. go to next step). No, set next format and go to step 2
Step 7 (Opt., for development only)	For each format supported by the device under test, tester transmits the test data.	Go to next step.

Step 8 (Opt., for development only)	Check data at sink's side	EMI interpreted correctly, Correct decoding, go to step 5 Error in data log, err. 2.1.9
Step 9 (Opt., for development only)	All available formats tested?	YES, end of test, DUT ok NO, set next format and go to step 7.
Test results		
DUT ok 2.1.0		Success.
DUT error 2.1.5		Wrong header. Return 2.1.5 (step,[parameter]).
DUT error 2.1.6		Wrong info bytes. Return 2.1.6 (step, [info in: info out]).
DUT error 2.1.7		Wrong packet size. Return 2.1.7 (format).
DUT error 2.1.8		Wrong encryption frame size. Return 2.1.8 (stream type)
DUT error 2.1.9		Wrong decoded data. Return 2.1.9 (channel, pattern, [data in: data out]).

Table 3-6: Generic Synchronous Packet Format

3.3.2.1.1 Example

3.3.2.1.1.1 Device under Test is a Source

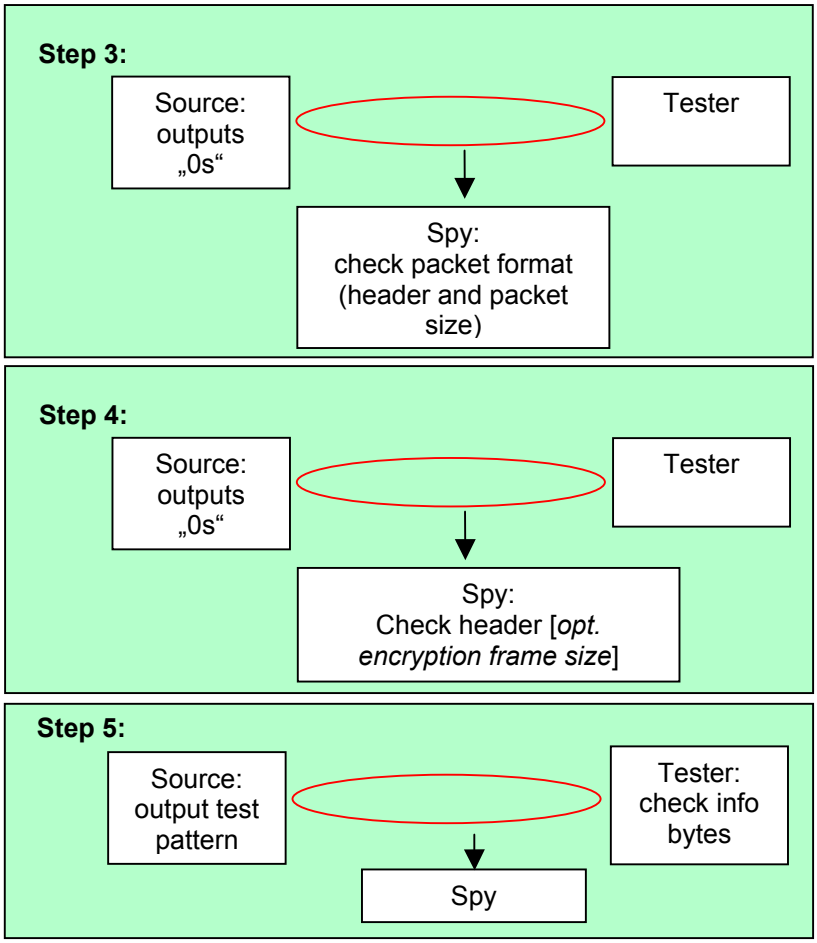

Figure 3-3: Source Testing by Packing

3.3.2.1.1.2 Device under Test is a Sink

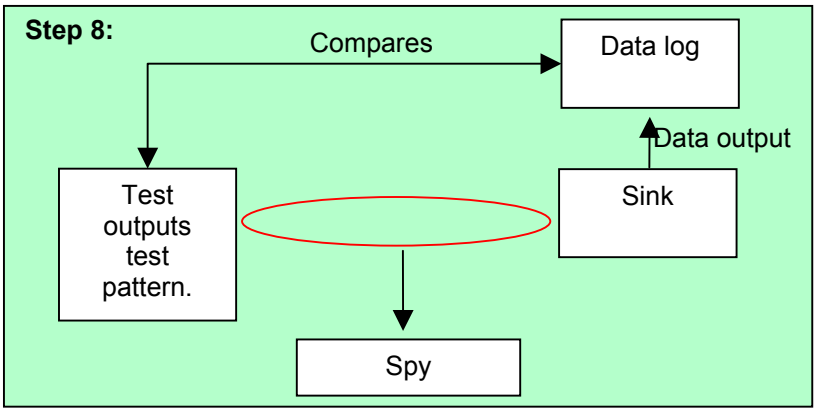

Figure 3-4: Sink Testing by Unpacking

3.3.2.2 Generic Packet Data Format

Name of test	Generic packet format	
Reference documents	MOST protection scheme rev 1.1-09 chapter 4.1 MOST stream transmission rev. 1.06 chapter 6.2 DTCP volume 1 supplement B.	
Reference to core function	Generic isochronous format	
Motivation, Matter of interest	The test is used to verify: - the correct format of MOST packets on the bus - the integrity of the complete data path (check output versus input).	
Device type	Source / Sink	
General/prerequisite	**Test content:** see chapter „3.2.2.2 Test Content“ for a description of the data sent or received by the device under test. **Data:** the data of the six test streams shall be correctly streamed on MOST to pass the test. **Test packet formats:** The format and content of the test packets is defined in the MOST stream transmission, chapter 6.2 and in the MOST protection scheme chapter 4.2. **Note:** the test for the sink data-unpacking is only possible with specially prepared development devices, therefore not suited for test houses, and consequently optional.	
Test sequence		
Action		Result
Step 1	If the device under test is a source, go to step 2 else go to step 6	Go to next step.
Step 2	A DTCP data transfer is started between source and sink.	Go to next step.
Step 3	Verification of packet size. Source encrypts test stream.	Isochronous frames on the bus. Correct format: go to next step. Wrong header, err. 2.2.5 Wrong packet size, err. 2.2.7
Step 4 (Opt., if supported by the tester)	Verification of encryption frame size. Source encrypts test stream.	Isochronous packets on the bus Correct format, go to next step. Wrong header, err. 2.2.9 Wrong encryption frame size, err. 2.2.8
Step 5	Verification of info bytes received by the tester.	Correct info bytes, end of test, DUT Ok. Wrong format of info bytes, err. 2.2.5.
Step 6 (Opt., for development only)	Tester transmits the test stream.	Go to next step.
Step 7 (Opt., for development only)	Check data at sink’s side	EMI interpreted correctly, correctly decoded data, end of test, DUT Ok. Error in data output, err. 2.2.9
Test results		
DUT ok 2.2.0		Success.

DUT error 2.2.5	Wrong header. Return 2.2.5 (step,[parameter]).
DUT error 2.2.6	Wrong info bytes. Return 2.2.6 (step, [info in: info out]).
DUT error 2.2.8	Wrong encryption frame size. Return 2.2..8 (stream type, stream data)
DUT error 2.2.9	Wrong decoded data. Return 2.2.9 (stream type, stream data, [data in: data out]).

Table 3-7: Generic Packet Format

3.3.3 Error Cases

3.3.3.1 Error Behavior (Device under Test is a Source)

Name of test	Error behavior
Reference documents	MOST protection scheme rev 1.1-09. DTCP volume 1 supplement B.
Reference to core function	Command rejection, Behavior upon reception of an error message.
Motivation, matter of interest	Test the behavior of the device under test in case of an error.
Device type	Source
General/prerequisite	Following cases are tested: - reaction of the source upon reception of an invalid command during authentication - reaction of the source upon reception of an error message from the sink. - cipher error in the sink.

Test sequence		
Action		Result
Step 1	Controller starts an authentication. Sink sends DTCP_Control.StartResult(invalid command)	NA
Step 2	Source reacts with DTCP_Control.Error(rejected)	YES, go to next step. No, err. 3.1.10 Wrong address, err. 3.1.1 Wrong command, err. 3.1.2 Invalid parameter, err. 3.1.3 Time out, warning 3.1.20
Step 3 (Opt.)	Controller to the source: DTCP_CipherStatus.Get()	Go to next step.
Step 4 (Opt.)	Source to the controller: DTCP_CipherStatus.Status(...) Check the status content. Should be “unauthenticated”.	Valid, go to next step Wrong address, err. 3.1.1 Wrong command, err. 3.1.2 Invalid parameter, err. 3.1.3 Time out, warning 3.1.20
Step 5	Controller starts an authentication. Source sends its challenge. Sink answers: DTCP_Control.Error(rejected)	NA
Step 6	Source breaks current authentication.	YES, go to next step. No, err. 3.1.10

Step 7 (Opt.)	Source and sink start complete authentication and start exchanging data. Sink sends to source: DTCP_CipherStatus.CipherError(DecodingError)	Go to next step.
Step 8 (Opt.)	Source stops operations	YES, end of test, DUT ok. No, err. 3.1.10.

Test results	
DUT ok 3.1.0	Success
DUT error 3.1.1	Message sent to the wrong node. Return 3.1.1(step)
DUT error 3.1.2	Wrong command. Return 3.1.2(step)
DUT error 3.1.3	One or more wrong parameters. Return 3.1.3(step,[parameter])
DUT error 3.1.10:	The device does not reject a command with wrong parameters. Return 3.1.10(command,[parameters list])
Warning 3.1.20:	The answer was not sent in time. Return 3.1.20(command).

Table 3-8: Error Behavior, Device under Test is a Source

3.3.3.1.1 Example: Invalid Subfunction Call

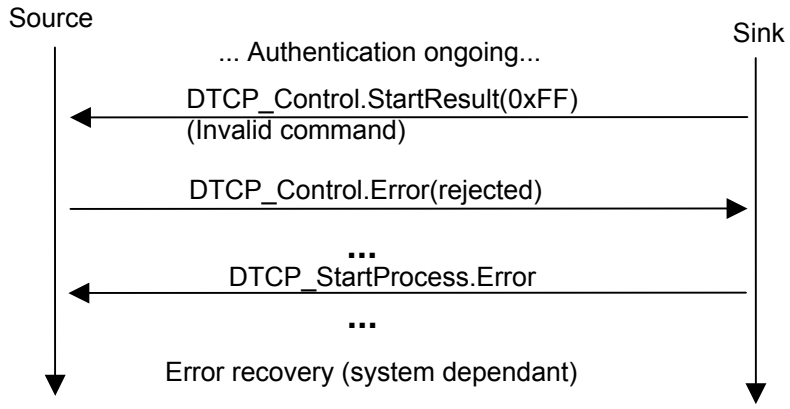

Figure 3-5: Invalid Subfunction Call

3.3.3.2 Error Behavior (Device under Test is a Sink)

Name of test	Error behavior
Reference documents	MOST protection scheme rev 1.1-09. DTCP volume 1 supplement B.
Reference to core function	Command rejection, behavior upon reception of an error message.
Motivation, matter of interest	Test the behavior of the device under test in case of an error.
Device type	Sink
General/prerequisite	Following cases are tested: - reaction of the sink upon reception of an invalid command during authentication - reaction of the sink upon reception of an error message from the source during authentication and content key exchange.

Test sequence		
Action		Result
Step 1	Controller starts an authentication. Source sends DTCP_Control.StartResult(invalid command)	NA
Step 2	Sink reacts with DTCP_Control.Error(rejected)	YES, go to next step. No, err. 3.2.10 Wrong address, err. 3.2.1 Wrong command, err. 3.2.2 Invalid parameter, err. 3.2.3 Time out, warning 3.2.20
Step 3	Sink to the controller: DTCP_StartProcess.Error	See step 2.
Step 4 (Opt.)	Controller to the sink: DTCP_CipherStatus.Get()	Go to next step.
Step 5 (Opt.)	Sink to the controller: DTCP_CiperhStatus.Status(…) Check the status content. Should be “unauthenticated”.	Valid, go to next step Wrong address, err. 3.1.1 Wrong command, err. 3.1.2 Invalid parameter, err. 3.1.3 Time out, warning 3.1.20
Step 6	Controller starts an authentication. Sink sends its challenge. Source answers: DTCP_Control.Error(rejected)	Go to next step.
Step 7	The sink breaks the current authentication.	Yes go to next step, No error 3.2.10
Step 8	Sink to the controller: DTCP_StartProcess.Error	See step 2.
Step 9	Controller starts an authentication. After completion, controller starts content key processing. Sink sends its key request command. Source answers DTCP_Control.Error	Go to next step.
Step 10	Sink to the controller: DTCP_ContentKeyProccess.Error	YES, go to next step,. Wrong address, err. 3.2.1 Wrong command, err. 3.2.2 Invalid parameter, err. 3.2.3 Time out, warning 3.2.20.

Step 11 (Opt.)	Source and sink start complete authentication and start exchanging data. Source sends to sink: DTCP_CipherStatus.CipherError(EncodingError)	Go to next step.
Step 12 (Opt.)	Sink to the controller: DTCP_StartProcess.Error.	YES, go to next step, Wrong address, err. 3.2.1 Wrong command, err. 3.2.2 Invalid parameter, err. 3.2.3 Time out, warning 3.2.20.
Step 13 (Opt.)	Sink stops operations	YES, end of test, DUT ok. No, err. 3.2.10.
Test results		
DUT ok 3.2.0		Success
DUT error 3.2.1		Message sent to the wrong node. Return 3.2.1(step)
DUT error 3.2.2		Wrong command. Return 3.2.2(step)
DUT error 3.2.3		One or more wrong parameters. Return 3.2.3(step,[parameter])
DUT error 3.2.10:		The device does not reject a command with wrong parameters. Return 3.2.10(command,[parameters list])
Warning 3.2.20:		The answer was not sent in time. Return 3.2.20(command).

Table 3-9: Error Behavior (Device under Test Is a Sink)

4 Compliance Matrix

Check boxes: ☑, ☐

Authentication tests. It is mandatory to pass at least one of full authentication and restricted authentication tests.		
Test name	Result (Optional steps supported, detailed: error(s) and warning(s))	Passed
Full authentication		☐
Restricted authentication		☐
SRM update		☐
Restricted authentication		☐

Data packing / unpacking. A test shall be passed for each supported format of data.			
Test name	Supported formats	Result (detailed: error(s) and warning(s))	Passed
Generic synchronous format			☐
			☐
Generic packet format			☐
			☐

Error behavior		
Test name	Result (detailed: error(s) and warning(s))	Passed
Error behavior		☐

Table 4-1: Compliance Matrix

5 Appendix A: List of Figures

Figure 3-1: General Test Set-up 13
Figure 3-2: SRM Test Example 27
Figure 3-3: Source Testing by Packing 30
Figure 3-4: Sink Testing by Unpacking 30
Figure 3-5: Invalid Subfunction Call 33

6 Appendix B: List of Tables

Table 2-1: Authentication Test Functions 10
Table 2-2: Data Packing / Unpacking Test Functions 10
Table 2-3: Error Cases Test Functions 11
Table 3-1: Error Parameters Values 12
Table 3-2: Full Authentication Test Procedure 19
Table 3-3: Restricted Authentication Test Procedure 22
Table 3-4: Content Key Request Test Procedure 24
Table 3-5: SRM Update Test Procedure 26
Table 3-6: Generic Synchronous Packet Format 29
Table 3-7: Generic Packet Format 32
Table 3-8: Error Behavior, Device under Test is a Source 33
Table 3-9: Error Behavior (Device under Test Is a Sink) 35
Table 4-1: Compliance Matrix 36

