

DTCP Volume 1
Supplement E
Mapping DTCP to IP
(Informational Version)

Hitachi, Ltd.

Intel Corporation

Matsushita Electric Industrial Co., Ltd.

Sony Corporation

Toshiba Corporation

Revision 1.2

June 15, 2007

Preface

Notice

THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Hitachi, Intel, MEI, Sony, and Toshiba (collectively, the "5C") disclaim all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Some portions of this document, identified as "Draft" are in an intermediate draft form and are subject to change without notice. Adopters and other users of this Specification are cautioned these portions are preliminary, and that products based on it may not be interoperable with the final version or subsequent versions thereof.

Copyright © 1997 - 2007 by Hitachi, Ltd., Intel Corporation, Matsushita Electric Industrial Co., Ltd., Sony Corporation, and Toshiba Corporation (collectively, the "5C"). Third-party brands and names are the property of their respective owners.

Intellectual Property

Implementation of this specification requires a license from the Digital Transmission Licensing Administrator.

Contact Information

Feedback on this specification should be addressed to dtla-comment@dtcp.com.

The Digital Transmission Licensing Administrator can be contacted at dtla-manager@dtcp.com.

The URL for the Digital Transmission Licensing Administrator web site is: <http://www.dtcp.com>.

Table of Contents

PREFACE	2
Notice	2
Intellectual Property	2
Contact Information	2
V1SE.1 Introduction	8
V1SE.1.1 Related Documents	8
V1SE.1.2 Terms and Abbreviations	8
V1SE.2 Modifications to 4.2.3.2 Extended Format Fields (Optional Components of the Device Certificate)	9
V1SE.3 Modifications to Chapter 5 Restricted Authentication	9
V1SE.4 Modifications to Chapter 6 Content Channel Management Protection	9
V1SE.4.1 Modifications to 6.2.1 Exchange Keys	9
V1SE.4.2 Modifications to 6.2.2.2 K_C for AES-128	9
V1SE.4.2.1 Modifications to 6.2.2.2.1 AES-128 Related Key and Constant Sizes	10
V1SE.4.3 Modifications to 6.3.1 Establishing Exchange Keys	10
V1SE.4.4 Modifications to 6.3.2 Establishing Content Keys	10
V1SE.4.5 Modifications to 6.3.3 Odd/Even Bit	10
V1SE.4.6 Modifications to 6.4.1 Embedded CCI	11
V1SE.4.7 PCP-UR	11
V1SE.4.8 Modifications to 6.4.2 Encryption Mode Indicator (EMI)	12
V1SE.4.9 Modifications to 6.4.3 Relationship between Embedded CCI and EMI	12
V1SE.4.10 Modification to 6.4.4.1 Format-cognizant source function	13
V1SE.4.11 Modification to 6.4.4.2 Format-non-cognizant source function	13
V1SE.4.12 Modifications to 6.4.4.3 Format-cognizant recording function	14
V1SE.4.13 Modifications to 6.4.4.4 Format-cognizant sink function	14
V1SE.4.14 Modification to 6.4.4.5 Format-non-cognizant recording function	15
V1SE.4.15 Modification to 6.4.4.6 Format-non-cognizant sink function	15
V1SE.4.16 Modifications to 6.4.5.1 Embedded CCI for audio transmission	16
V1SE.4.17 Modifications to 6.4.5.3 Audio-format-cognizant source function	16
V1SE.4.18 Modifications to 6.4.5.5 Audio-format-cognizant recording function	16
V1SE.4.19 Modifications to 6.4.5.6 Audio-format cognizant sink function	16

V1SE.4.20 Modifications to 6.4.5.8 Audio-Format-non-cognizant sink function	16
V1SE.4.21 Modifications to 6.6.1 Baseline Cipher	17
V1SE.4.22 Modifications to 6.6.2.1 AES-128 Cipher	17
V1SE.4.23 Modification to 6.6.3 Content Encryption Formats	18
V1SE.4.23.1 N _c field	18
V1SE.4.23.2 PCP-UR field	19
V1SE.4.23.3 PCP-UR capable source devices	20
V1SE.4.23.4 PCP-UR capable sink devices	22
V1SE.4.24 Modifications to 6.7.1 Move Function	23
V1SE.5 Modifications to Chapter 8 (AV/C Digital Interface Command Set Extensions)	24
V1SE.5.1 Modifications to 8.1 Introduction	24
V1SE.5.2 Modifications to 8.3.1 AKE Control Command	24
V1SE.5.3 Modification to 8.3.2 AKE Status Command	25
V1SE.5.3.1 Modifications to AKE status command status field	25
V1SE.5.4 Modifications to 8.3.3	26
V1SE.5.4.1 AKE_ID dependent field	26
V1SE.5.4.2 Modifications to Authentication selection	26
V1SE.5.4.3 Modification to Exchange_key values	26
V1SE.5.5 Modifications to AKE Subfunctions	27
V1SE.5.6 Modifications to 8.4 Bus Reset Behavior	27
V1SE.6 Modifications to Appendix A (Additional Rules for Audio Applications)	28
V1SE.6.1 Modification to A.1 AM824 audio	28
V1SE.6.1.1 Modification to A.1.1 Type 1: IEC 60958 Conformant Audio	28
V1SE.6.1.2 Modification to A.1.2 Type 2: DVD-Audio	28
V1SE.6.1.3 Modification to A.1.3 Type 3: Super Audio CD	28
V1SE.6.2 Modification to A.2 MPEG Audio	28
V1SE.7 Modification to Appendix B (DTCP_Descriptor for MPEG Transport Stream)	28
V1SE.7.1 Modification to B.1 DTCP_descriptor	28
V1SE.7.2 Modification to B.2 DTCP_descriptor syntax	29
V1SE.7.2.1 Modification to B.2.1 private_data_byte Definitions:	30
V1SE.7.3 Modification to B.3 Rules for the Usage of the DTCP_descriptor	30
V1SE.7.3.1 Modification to B.3.1 Transmission of a partial MPEG-TS	30
V1SE.7.3.2 Modification to B.3.3.Treatment of the DTCP_descriptor by the sink device	31
V1SE.8 Additional Requirements	32

V1SE.8.1 Authentication Capability Constraint	32
V1SE.8.2 Internet Datagram Header Time To Live (TTL) Constraint	32
V1SE.8.3 802.11 Constraint	32
V1SE.8.4 DTCP-IP Move Protocol	32
V1SE.8.4.1 Move RTT-AKE	32
V1SE.8.4.1.1 Establishing Move Exchange Key	33
V1SE.8.4.2 Move Transmission	34
V1SE.8.4.3 Move Commitment	35
V1SE.8.4.3.1 Resumption of Move Commitment	36
V1SE.8.4.4 Cancel of Move transaction	38
V1SE.8.5 Additional Localization via RTT	38
V1SE.8.5.1 Protected RTT Protocol	38
V1SE.8.5.2 RTT-AKE	40
V1SE.8.5.3 Background RTT Check	41
V1SE.8.6 Content Key Confirmation	42
V1SE.9 Additional Commands and Sequences	43
V1SE.10 Recommendations	44
V1SE.10.1 Recommended MIME type for DTCP protected content	44
V1SE.10.2 Identification of DTCP Sockets	44
V1SE.10.2.1 URI Recommended Format	44
V1SE.10.2.2 HTTP response / request	44
V1SE.10.3 Header Field Definition for HTTP	45
V1SE.10.3.1 Range.dtcp.com	45
V1SE.10.3.2 Content-Range.dtcp.com	45
V1SE.10.4 BLKMove.dtcp.com	45
V1SE.10.5 Definition for UPnP AV CDS Property	45
V1SE.10.5.1 DTCP.COM_FLAGS param	45
V1SE.10.5.2 res@dtcp:uploadInfo	45

Figures

Figure 1 Protected Content Packet Format	18
Figure 2 N_c with PCP-UR and SN_c	19
Figure 3 PCP-UR Format	19
Figure 4 DTCP-IP Control Packet Format	24
Figure 5 Status Packet Format	25
Figure 6 Move RTT-AKE Protocol Flow	33
Figure 7 Move Commitment Protocol Flow	35
Figure 8 Resume procedure for sink device	37
Figure 9 Resume procedure for source device when MV_FINALIZE is received	37
Figure 10 Resume procedure for source device when MV_COMPLETE is received	37
Figure 11 RTT Protocol Diagram	39
Figure 12 AKE-RTT Informative Flow Diagrams	41
Figure 13 Background RTT Check Informative Flow Diagram	42
Figure 14 Content Key Confirmation Procedure	43

Tables

Table 1 Length of Keys and Constants (Content Channel Management)	10
Table 2 E-EMI Mode and E-EMI Description	12
Table 3 Relationship between E-EMI and Embedded CCI	12
Table 4 Format-Cognizant Source Function CCI handling	13
Table 5 Format-Non-Cognizant Source Function CCI handling	13
Table 6 Format-cognizant recording function CCI handling	14
Table 7 Format-cognizant sink function CCI handling	14
Table 8 Format-non-cognizant recording function CCI handling	15
Table 9 Audio Embedded CCI Values	16
Table 10 Audio-format cognizant source function CCI handling	16
Table 11 Audio-format-cognizant recording function CCI handling	16
Table 12 Audio-format-cognizant sink function CCI handling	16
Table 13 UR Mode values	19
Table 14 Content Type values	19
Table 15 E-EMI Mode and CCI mapping for Audiovisual content	21
Table 16 E-EMI Mode and CCI mapping for Type 1 Audio content	21
Table 17 AKE Status Command Status Field	25
Table 18 AKE_procedure values	26
Table 19 Authentication selection	26
Table 20 Exchange_key values	26
Table 21 Syntax of private_data_type for DTCP_audio_descriptor	29
Table 22 Descriptor_ID	30
Table 23 DTCP_CCI_audio	30
Table 24 Audio_type	30

Volume 1 Supplement E DTCP Mapping to IP

V1SE.1 Introduction

This supplement describes the mapping of DTCP onto Internet Protocol (IP). All aspects of IEEE 1394 DTCP functionally are preserved except those described in Appendix D of Volume 1 which do not apply to this mapping and this supplement only details DTCP-IP specific changes or additions.

V1SE.1.1 Related Documents

This specification shall be used in conjunction with the following publications. When the publications are superseded by an approved revision, the revision shall apply.

- Digital Transmission Content Protection Specification Volume 1 and Volume 2
- RFC768 User Datagram Protocol
- RFC791 Internet Protocol
- RFC793 Transmission Control Protocol
- RFC1945 Hypertext Transfer Protocol – HTTP/1.0
- RFC2616 Hypertext Transfer Protocol – HTTP/1.1
- RFC1889 RTP: A Transport Protocol for Real-Time Applications
- UPnP ContentDirectory:2, ContentDirectory:2 Service Template Version 1.01, UPnP Forum, May 31, 2006

V1SE.1.2 Terms and Abbreviations

DTCP-IP	DTCP volume 1 Supplement E
DTCP Socket	Means the Socket used for AKE commands
E-EMI	Extended Encryption Mode Indicator
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
PCP	Protected Content Packet
RTP	Real-time Transport Protocol
RTT	Round Trip Time
Socket	Means IP-address concatenated with port number [e.g. <host>: <port>]
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
PCP-UR	Protected Content Packet – Usage Rule

V1SE.2 Modifications to 4.2.3.2 Extended Format Fields (Optional Components of the Device Certificate)

For IP, the optional content channel cipher for AES-128 is not used.

V1SE.3 Modifications to Chapter 5 Restricted Authentication

Restricted authentication is not permitted for DTCP-IP transports.

V1SE.4 Modifications to Chapter 6 Content Channel Management Protection

V1SE.4.1 Modifications to 6.2.1 Exchange Keys

DTCP-IP requires only a single exchange key for all defined E-EMI.

V1SE.4.2 Modifications to 6.2.2.2 K_C for AES-128

The Content Key (K_C) is used as the key for the content encryption engine. K_C is computed from the three values shown below:

- Exchange Key K_X where only a single exchange key is used for all E-EMIs to protect the content.
- Seed for content channel N_C generated by the source device which is sent in plain text to all sink devices.
- Constant value C_{A0} , C_{B1} , C_{B0} , C_{C1} , C_{C0} , or C_{D0} which corresponds to an E-EMI value in the packet header.

The Content Key is generated as follows:

$$K_C = J\text{-AES}(K_X, f[\text{E-EMI}], N_C) \quad \text{Where:}$$

$$f[\text{E-EMI}] \{$$

$$f[\text{E-EMI}] = C_{A0} \text{ when E-EMI = Mode A0}$$

$$f[\text{E-EMI}] = C_{B1} \text{ when E-EMI = Mode B1}$$

$$f[\text{E-EMI}] = C_{B0} \text{ when E-EMI = Mode B0}$$

$$f[\text{E-EMI}] = C_{C1} \text{ when E-EMI = Mode C1}$$

$$f[\text{E-EMI}] = C_{C0} \text{ when E-EMI = Mode C0}$$

$$f[\text{E-EMI}] = C_{D0} \text{ when E-EMI = Mode D0}$$

$$\}$$

C_{A0} , C_{B1} , C_{B0} , C_{C1} , C_{C0} , and C_{D0} are universal secret constants assigned by the DTLA. The values for these constants are specified in DTCP Specification available under license from DTLA.

Additional rules for AES-128 Cipher are described in the DTCP Specification available under license from the DTLA.

V1SE.4.2.1 Modifications to 6.2.2.2.1 AES-128 Related Key and Constant Sizes

Followings are the lengths of the keys and constants described above:

Key or Constant	Size (bits)
Exchange Key (K_X)	96
Scrambled Exchange Key (K_{SX})	96
Constants ($C_{A0}, C_{B1}, C_{B0}, C_{C1}, C_{C0}, C_{D0}$)	96
Content Key for AES-128 Baseline Cipher (K_C)	128
Seed for Content Channel (N_C)	64

Table 1 Length of Keys and Constants (Content Channel Management)

V1SE.4.3 Modifications to 6.3.1 Establishing Exchange Keys

It is mandatory that source devices expire an Exchange Key within 2 hours after all content transmission using PCP(s) has ceased.

It is mandatory that sink devices expire an Exchange Key within 2 hours of continuous non-use of that Exchange Key for decryption.

Source and sink devices must expire their Exchange Keys when they detect themselves being disconnected from all mediums. For wireless mediums this means when device detects that it is not connected to an access point or it is not directly connected to another device.

Source devices can not change or expire Exchange key during content transmission using PCP(s).

V1SE.4.4 Modifications to 6.3.2 Establishing Content Keys

This section replaces section 6.3.2 and describes the mechanism for establishing the Content Keys (K_C) used to encrypt/decrypt content being sent over DTCP-IP.

Source devices that do not support PCP-UR generate N_C as follows:

- For RTP transfers, source device generates a 64 bit random number as an initial value for N_C using RNG_F . N_C is updated periodically by incrementing it by $1 \bmod 2^{64}$ while at least on RTP transmission with PCP is in progress regardless of the value of E-EMI. The same value of N_C shall be used for all RTP simultaneous transmissions. The minimum period for update of the N_C is defined as 30 seconds, and the maximum period is defined as 120 seconds.
- For HTTP transfers, source devices generate a 64 bit random number as an initial value of N_C for the initial TCP connection using RNG_F . The initial N_C for subsequent TCP connections must be different (another random number may be generated). If a HTTP response / request has more than 128 MB of content, N_C shall be updated every 128MB. N_C is updated by incrementing it by $1 \bmod 2^{64}$. When plural HTTP responses / request are transmitted using the same TCP connection, N_C for subsequent HTTP response / request shall be updated from the latest N_C for the TCP connection.

Source devices that do support PCP-UR understand that N_C consists of two fields; a 16 bit PCP-UR field and a 48 bit SN_C nonce, where SN_C is handled in manner similar to the 64 bit N_C nonce except that the initial value of SN_C consist of a zero followed by a 47 bit random number and is updated by incrementing it by $1 \bmod 2^{48}$.

V1SE.4.5 Modifications to 6.3.3 Odd/Even Bit

The Odd/Even Bit is not used in DTCP-IP as N_C value is sent with each PCP.

V1SE.4.6 Modifications to 6.4.1 Embedded CCI

Embedded CCI is carried as part of the content stream. Many content formats including MPEG have fields allocated for carrying the CCI associated with the stream. The definition and format of CCI is specific to each content format. Information used to recognize the content format should be embedded within the content.

In the following sections, Embedded CCI is interpreted to one of four states Copy Never (CN), Copy One Generation (COG), No More Copies (NMC) or Copy Freely. Copy Freely has two variations; Copy freely with EPN asserted (CF/EPN) and Copy freely with EPN unasserted (CF).

Since the rules for recording differ based on content type, COG is identified as either Copy One Generation for audiovisual content (COG-AV) or Copy One Generation for audio content (COG-Audio) in the following sections.

V1SE.4.7 PCP-UR

PCP-UR is used as a common way to carry usage rule such as APS and ICT in the PCP header. The format of PCP-UR is described in section V1SE.4.23.1.

PCP-UR may be used in two cases. If PCP-UR is used for content which has Embedded CCI, sink functions which do not recognize the Embedded CCI (Format-non-cognizant sink and recording function) can use information in the PCP-UR along with E-EMI.

If PCP-UR is used for content which has no Embedded CCI, sink devices can regard the PCP-UR along with E-EMI as the Embedded CCI. For this type of content, sink functions and recording functions which recognize E-EMI and PCP-UR behave as Format-cognizant functions.

V1SE.4.8 Modifications to 6.4.2 Encryption Mode Indicator (EMI)

E-EMI Mode	E-EMI Value	Description
Mode A0	1100 ₂	Copy-never (CN)
Mode B1	1010 ₂	Copy-one-generation (COG) [Format-cognizant recording only]
Mode B0	1000 ₂	Copy-one-generation (COG) [Format-non-cognizant recording permitted]
Mode C1	0110 ₂	Move [Audiovisual]
Mode C0	0100 ₂	No-more-copies (NMC)
Mode D0	0010 ₂	Copy-free with EPN asserted (CF/EPN)
N.A.	0000 ₂	Copy-free (CF)
	---- ₂	All other values reserved

Table 2 E-EMI Mode and E-EMI Description

V1SE.4.9 Modifications to 6.4.3 Relationship between Embedded CCI and EMI

E-EMI	Embedded CCI					
	CF	CF/EPN	NMC	COG-AV	COG-Audio	CN
Mode A0 (CN)	Allowed	Allowed	Allowed ¹	Allowed	Allowed	Allowed
Mode B1 (Format cognizant only recordable)	Allowed	Allowed	Prohibited	Allowed	Allowed	Prohibited
Mode B0 (Format non-cognizant recordable)	Allowed	Allowed	Prohibited	Allowed	Prohibited	Prohibited
Mode C0 (NMC)	Allowed	Allowed	Allowed	Allowed	Allowed	Prohibited
Mode D0 (CF/EPN)	Allowed	Allowed	Prohibited	Prohibited	Prohibited	Prohibited
N.A.	Allowed	Prohibited	Prohibited	Prohibited	Prohibited	Prohibited

Table 3 Relationship between E-EMI and Embedded CCI

¹ Not typically used.

V1SE.4.10 Modification to 6.4.4.1 Format-cognizant source function

Embedded CCI of programs					E-EMI
CF	CF/EPN	NMC	COG-AV	CN	
Don't care	Don't care	* ²	Don't care	Present	Mode A0
Don't care	Don't care	Cannot be present	Present	Cannot be present	Mode B1
Don't care	Don't care	Cannot be present	Present	Cannot be present	Mode B0
Don't care	Don't care	Present	Cannot be present ³	Cannot be present	Mode C0
Don't care	Present	Cannot be present	Cannot be present	Cannot be present	Mode D0
Present	Cannot be present	Cannot be present	Cannot be present	Cannot be present	N.A.
Other combinations					Transmission Prohibited

Table 4 Format-Cognizant Source Function CCI handling

V1SE.4.11 Modification to 6.4.4.2 Format-non-cognizant source function

E-EMI or recorded CCI ⁴ of source content	E-EMI used for transmission
Copy Never	Mode A0
COG: Format cognizant only recordable	Mode B1
COG: Format non-cognizant recordable	Mode B0
No-more-copies	Mode C0
EPN asserted Copy Free	Mode D0
Copy-Free	N.A.

Table 5 Format-Non-Cognizant Source Function CCI handling

² Don't care, but not typically used.

³ This combination is allowed for format-non-cognizant source function, but is not permitted for format-cognizant source function.

⁴ Recorded CCI is copy control information that is not embedded in the content program and does not require knowledge of the content format to extract.

V1SE.4.12 Modifications to 6.4.4.3 Format-cognizant recording function

E-EMI	Embedded CCI for each program				
	CF	CF/EPN	NMC	COG-AV	CN
Mode A0	Recordable	Recordable	Do not record	*5	Do not record
Mode B1	Recordable	Recordable	Discard entire content stream ⁶	*5	Discard entire content stream ⁶
Mode B0	Recordable	Recordable	Discard entire content stream ⁶	*5	Discard entire content stream ⁶
Mode C0	Recordable	Recordable	Do not record	Do not record	Discard entire content stream ⁶
Mode D0	Recordable	Recordable	Discard entire content stream ⁶	Discard entire content stream ⁶	Discard entire content stream ⁶

Table 6 Format-cognizant recording function CCI handling

V1SE.4.13 Modifications to 6.4.4.4 Format-cognizant sink function

E-EMI	Embedded CCI for each program				
	CF	CF/EPN	NMC	COG-AV	CN
Mode A0	Available for processing	Available for processing	Available for processing ¹	Available for processing	Available for processing
Mode B1	Available for processing	Available for processing	Discard entire content stream ⁷	Available for processing	Discard entire content stream ⁷
Mode B0	Available for processing	Available for processing	Discard entire content stream ⁷	Available for processing	Discard entire content stream ⁷
Mode C0	Available for processing	Available for processing	Available for processing	Available for processing ⁸	Discard entire content stream ⁷
Mode D0	Available for processing	Available for processing	Discard entire content stream ⁷	Discard entire content stream ⁷	Discard entire content stream ⁷

Table 7 Format-cognizant sink function CCI handling

⁵ If the recording function supports recording a CCI value of No-more-copies then the CCI value of No-more-copies shall be recorded with the program. Otherwise the CCI of Copy-never shall be recorded with the program.

⁶ If the function detects this CCI combination among the programs it is recording, the entire content stream is discarded.

⁷ If the function detects this CCI combination among the programs, the entire content stream is discarded.

⁸ If the device has a rule for handling No-more-copies, this program shall be handled according to the rule. Otherwise the program shall be handled as Copy Never.

V1SE.4.14 Modification to 6.4.4.5 Format-non-cognizant recording function

E-EMI of the received stream	Recorded CCI ⁹ to be written onto user recordable media
Mode A0	Stream cannot be recorded
Mode B1	Stream cannot be recorded
Mode B0	No-more-copies
Mode C0	Stream cannot be recorded
Mode D0	EPN asserted Copy Free

Table 8 Format-non-cognizant recording function CCI handling

V1SE.4.15 Modification to 6.4.4.6 Format-non-cognizant sink function

Only bridge and rendering functions are allowed for this function unless the sink function is capable of processing the DTCP_descriptor or PCP-UR.

⁹ Recorded CCI is copy control information that is not embedded in the content program and does not require knowledge of the content format to extract.

V1SE.4.16 Modifications to 6.4.5.1 Embedded CCI for audio transmission

Value and Abbreviation	Meaning
11	Not defined
10 (COG-audio)	Copy-permitted-per-type
01 (NMC)	No-more-copies
00 (CF)	Copy-free

Table 9 Audio Embedded CCI Values

V1SE.4.17 Modifications to 6.4.5.3 Audio-format-cognizant source function

Embedded CCI of programs			E-EMI
CF	NMC	COG-audio	
Type specific ¹⁰			Mode A0
Don't care	Cannot be present	Present	Mode B1
Don't care	Present	Don't care	Mode C0
Present	Cannot be present	Cannot be present	N.A.

Table 10 Audio-format cognizant source function CCI handling

V1SE.4.18 Modifications to 6.4.5.5 Audio-format-cognizant recording function

E-EMI	Embedded CCI of Program		
	CF	NMC	COG-audio
Mode A0	Recordable	Do not record	Recordable ¹¹
Mode B1	Recordable	Discard entire content stream ¹²	Recordable ¹¹
Mode C0	Recordable	Do not record	Recordable ¹¹

Table 11 Audio-format-cognizant recording function CCI handling

V1SE.4.19 Modifications to 6.4.5.6 Audio-format cognizant sink function

E-EMI	Embedded CCI of program		
	CF	NMC	COG-audio
Mode A0	Available for processing	Available for processing	Available for processing
Mode B1	Available for processing	Discard entire content stream ¹²	Available for processing
Mode C0	Available for processing	Available for processing	Available for processing

Table 12 Audio-format-cognizant sink function CCI handling

V1SE.4.20 Modifications to 6.4.5.8 Audio-Format-non-cognizant sink function

Only bridge and rendering functions are allowed for this function unless the sink function is capable of processing the DTCP_audio_descriptor or PCP-UR.

¹⁰ Usage is specified for each Audio type in Appendix A.

¹¹ The CCI value of No-more-copies shall be recorded with the program. Additional rules for recording are specified by each audio application in Appendix A.

¹² If the function detects this CCI combination among the programs it is recording the entire content stream is discarded.

V1SE.4.21 Modifications to 6.6.1 Baseline Cipher

For IP, the baseline cipher is AES-128 using the Cipher Block Chaining (CBC). AES-128 is described in FIPS 197 dated November 26, 2001 and the CBC mode is described in NIST SP 800-38A 2001 Edition.

V1SE.4.22 Modifications to 6.6.2.1 AES-128 Cipher

For AES-128, Cipher Block Chaining (CBC) is used. AES-128 is described in FIPS 197 dated November 26, 2001 and the CBC mode is described in NIST SP800-38A 2001 Edition. Additional rules for AES-128 Cipher are described in the DTCP specification available under license from DTLA.

V1SE.4.23 Modification to 6.6.3 Content Encryption Formats

DTCP encrypted content is sent via Protected Content Packets (PCP) where the format of the PCP is described in the following figure.

	msb						lsb
Header[0]	reserved (zero)		C_A	E-EMI			
Header[1]	exchange_key_label						
Header[2]	N _c (64 bits)						
Header[3]							
Header[4]							
Header[5]							
Header[6]							
Header[7]							
Header[8]							
Header[9]							
Header[10]	Byte length of content denoted as CL (32 bits)						
Header[11]							
Header[12]							
Header[13]							
EC[0]	Content affixed with 0 to 15 bytes of padding						
EC[1]							
EC[2]							
-							
-							
-							
-							
EC[N-1]							

Figure 1 Protected Content Packet Format

Header [0]: C_A means cipher_algorithm where a value of 0₂ denotes AES-128 and the value 1₂ denotes optional cipher. E-EMI is as defined in section V1SE.4.7

Header [1]: Contains exchange_key_label which is described in the DTCP Specification available under license from DTLA.

Header [2..9]: Contains N_c as described in section V1SE.4.2.1.

Header [10..13]: Denotes byte length of content and does not include any padding bytes, where CL is less than or equal to 128 MB.

EC [0..N-1]: Represents encrypted frame and there is no EC when CL is zero otherwise it is a multiple of 16 Bytes in length where $N = (\text{Int}((\text{CL}-1)/16)+1)*16$ where padding length is equal to N-CL and Int(X) means maximum integer less than or equal to X. The value of each padding Byte is 00₁₆.

For RTP transfers, each RTP payload is encapsulated by a single PCP.

For HTTP transfers, responses / requests may contain 1 or more PCPs.

V1SE.4.23.1 N_c field

Source devices that do not support PCP-UR treat N_c as a 64 bit nonce and source devices that do support PCP-UR understand that N_c consists of two fields; a 16 bit PCP-UR field and a 48 bit SN_c nonce as shown in Figure 2.

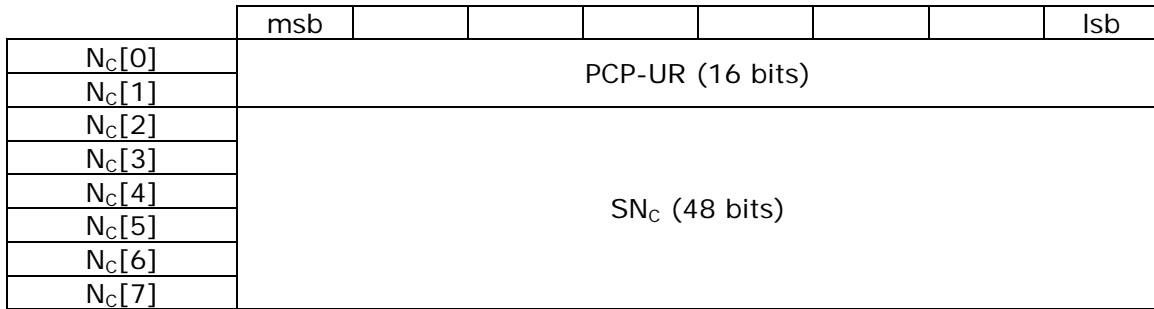


Figure 2 N_c with PCP-UR and SN_c

Source device may support PCP-UR but if a source device supports PCP-UR it shall always transmit content with the N_c with the PCP-UR field and 48 bit SN_c nonce.

V1SE.4.23.2 PCP-UR field

The following figure shows the format of PCP-UR field:

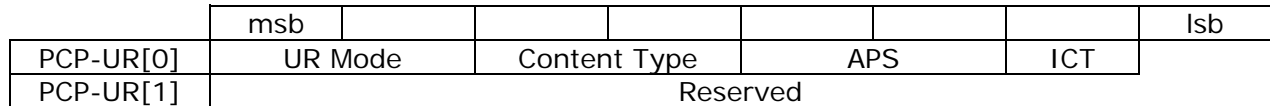


Figure 3 PCP-UR Format

UR Mode field indicates how the PCP-UR is interpreted. Source devices should not change the value of UR Mode in the middle of a content transmission.

UR Mode	Meaning
00 ₂	No information
01 ₂	Content stream has Embedded CCI. PCP-UR has the same information as the Embedded CCI
10 ₂	Content stream has no valid Embedded CCI. PCP-UR and E-EMI are regarded as the Embedded CCI
11 ₂	Reserved

Table 13 UR Mode values

Content Type field indicates the type of content. Source devices should not change the value of Content Type in the middle of a content transmission. When Content Type field has value of 01₂, following APS and ICT fields are unavailable.

Content Type	Meaning
00 ₂	Audiovisual
01 ₂	Type 1 Audio
10 ₂	Reserved
11 ₂	Reserved

Table 14 Content Type values

APS field contains analog copy protection information as described in section B.2.1 of Volume 1 of the Specification.

ICT field contains Image_Constraint_Token as described in section B.2.1 of Volume 1 of the Specification. When a source device sends multiplexed content, most restrictive value shall be set to this field.

Reserved field is the area for future extension. Source devices shall set to zero. Sink devices shall use value of Reserved field to calculate K_c in order that they can accommodate any future changes.

V1SE.4.23.3 PCP-UR capable source devices

PCP-UR capable source devices shall always transmit content using the N_c that consists of the PCP-UR field and 48 bit SN_c nonce.

Source devices must provide PCP-UR when transmitting content that does not have APS and ICT associated to that content in the embedded CCI.

Source devices that support PCP-UR shall support CAPABILITY_EXCHANGE subfunction and shall set PCP-UR as follows.

- Source device shall set zero to the UR Mode field (UR Mode 00_2) and subsequent PCP-UR fields when it transmits the following content:
 - MPEG-TS content.
 - Type 2 Audio content and Type 3 Audio content.
 - Content consists of multiple substream which may have different states for Content Type and APS fields.
 - Content received using DTCP without PCP-UR and the source device cannot recognize Embedded CCI corresponds to APS and ICT.
- Source device may use UR Mode 00_2 or UR Mode 01_2 when it transmits content stream with Embedded CCI that contains CCI, APS and ICT information associated to that content but UR Mode 01_2 is recommended.
 - When UR Mode 00_2 is used, the source device shall set Content Type, APS and ICT fields to zero.
 - When UR Mode 01_2 is used, the source device shall set the value of Content Type field according to the types of content and:
 - ✧ When value of Content Type field is 00_2 it will set APS and ICT fields equivalent to those values in Embedded CCI.
 - ✧ When value of Content Type field is 01_2 , the source device shall set APS and ICT fields to zero.
- Source device shall set 10_2 to the UR Mode field when it transmits content stream without Embedded CCI which corresponds to CCI, APS and ICT associated to that content or with invalid value of such Embedded CCI. In this case, the source device shall set the value of Content Type field according to the types of content. The source device shall also set APS and ICT fields equivalent to the information associated to the content.

- When UR Mode is 10₂, source device shall set E-EMI based on CCI of transmitting content as follows:

Content Type 00₂ case:

E-EMI Mode	CCI
Mode A0	Copy-never (CN)
Mode B1	Copy-one-generation (COG) [Format-cognizant recording only]
Mode B0	Copy-one-generation [Format-non-cognizant recording permitted]
Mode C0	No-more-copies (NMC)
Mode D0	Copy-free with EPN asserted (CF/EPN)
N.A.	Copy-free (CF)

Table 15 E-EMI Mode and CCI mapping for Audiovisual content

In case of Move, Mode C1 of E-EMI is used.

Content Type 01₂ case:

Any content format using CCI¹³ equivalent to SCMS can be transmitted as Type 1 Audio with UR Mode 10₂.

E-EMI Mode	CCI
Mode A0	N.A.
Mode B1	Copy-one-generation (COG) [Format-cognizant recording only]
Mode B0	N.A.
Mode C0	No-more-copies (NMC)
Mode D0	N.A.
N.A.	Copy-free (CF)

Table 16 E-EMI Mode and CCI mapping for Type 1 Audio content

- Source device shall set zero to the APS and ICT fields when Content Type is 01₂.

¹³ Content format without ASE-CCI can be transmitted.

V1SE.4.23.4 PCP-UR capable sink devices

PCP-UR capable sink devices are required to confirm that the source device is PCP-UR capable by using the CAPABILITY_EXCHANGE subfunction. Sink devices can use PCP-UR only when content accompanied by the PCP-UR is encrypted by the source device which supports PCP-UR.

PCP-UR capable sink devices shall treat PCP-UR based on the value of UR Mode as follows.

UR Mode 00₂:

- Sink device shall ignore fields in PCP-UR subsequent to the UR Mode field.

UR Mode 01₂:

- If Embedded CCI is recognized, the Embedded CCI shall be used instead of PCP-UR. (Considered to be Format-cognizant sink functions and Format-cognizant recording functions.)
- If Embedded CCI is not recognized, the sink device behave as Format-non-cognizant sink functions or Format-non-cognizant recording functions and may use PCP-UR along with E-EMI to control its behavior. If a content consists of multiple substreams, all the substreams are regarded as they have the same CCI with regard to the information in PCP-UR and E-EMI.
- If sink device detects value of 10₂ and 11₂ for Content Type field, it shall ignore the subsequent fields in the PCP-UR field.

UR Mode 10₂:

- Sink devices may regard the PCP-UR and E-EMI as the Embedded CCI of the content and shall disregard any embedded CCI or alternative Embedded CCI. In this case, the Sink devices behave as Format-cognizant sink functions or Format-cognizant recording functions. If a content consists of multiple substreams, all the substreams will have the same CCI.
- Sink devices may determine CCI of content from E-EMI based on the mapping shown in V1SE.4.23.3.
- If sink device detects value of 10₂ and 11₂ for Content Type field, it shall ignore the subsequent fields in PCP-UR field and behaves as Format-non-cognizant function.

UR Mode 11₂:

- Sink device shall behave in the same way as when UR Mode is 00₂.

V1SE.4.24 Modifications to 6.7.1 Move Function

This supplement defines a Move function in addition to the one described in section 6.7.1 where content with Embedded CCI of No-more-copies content may not be remarked as Copy-one-generation but instead be transmitted as No-more-copies using Mode C1 of E-EMI for IP transport of DTCP protected content and Recording functions may record the received content without remarking embedded CCI. E-EMI Mode B1 shall be used for Move-mode when source function uses Move function described in section 6.7.1. For clarity, the move function shall be used between a single source and a single sink function.

Section V1SE.8.4 defines a protocol for transaction based Move function using Mode C1 of E-EMI, which uses Exchange key dedicated for Move.

V1SE.5 Modifications to Chapter 8 (AV/C Digital Interface Command Set Extensions)

V1SE.5.1 Modifications to 8.1 Introduction

DTCP-IP uses TCP port to send/receive DTCP control packets, status command packets, and response packets. DTCP Socket identification of source device is described in section V1SE.10.2. Devices shall wait at least one second for a response to a command before timing out.

V1SE.5.2 Modifications to 8.3.1 AKE Control Command

This section maps the AKE control command specified in Section 8.3.1 to the DTCP-IP Control Packet Format. Except as otherwise noted, the AKE control command sub fields used with IP have the same values and functions as detailed in Chapter 8.

	msb							lsb
Type[0]	0	0	0	0	0	0	0	1
Length[0]	(msb) Byte Length of Control and AKE_Info Fields (N+8)							(lsb)
Length[1]								
Control[0]	reserved (zero)				ctype/response			
Control[1]	Category = 0000 ₂ (AKE)				AKE_ID = 0000 ₂			
Control[2]	subfunction							
Control[3]	AKE_procedure							
Control[4]	exchange_key							
Control[5]	subfunction_dependent							
Control[6]	AKE_label							
Control[7]	number(option)				status			
AKE_Info[0..N-1]	AKE_Info							

Figure 4 DTCP-IP Control Packet Format

- Type, Length, and Control byte 0 are used to map DTCP to IP. Where the Type field identifies version 1 AKE control packet.
- ctype/response has the same values as referenced in chapter 8 of DTCP specification and specified by the AV/C Digital Interface Command.
- Control bytes 1..7 are identical to operand bytes 0..6 as specified in section 8.3.1, except for four most significant bits of Control byte 7 which is not used in IP.
- The AKE_Info field is identical to the data field specified in section 8.3.1.
- The AKE_label and source Socket of each control command should be checked to ensure that it is from the appropriate controller.
- Unless otherwise noted in the description of each subfunction, if a given command frame includes a data field, the corresponding response frame does not have a data field.

V1SE.5.3 Modification to 8.3.2 AKE Status Command

This section maps the AKE status command specified in Section 8.3.2 to the DTCP-IP Status Packet Format. Except as otherwise noted, the AKE status command sub fields used with IP have the same values and functions as detailed in Chapter 8.

	msb							lsb
Type[0]	0	0	0	0	0	0	0	1
Length[0]	(msb) Byte length of Control							
Length[1]								(lsb)
Control[0]	reserved (Zero)				ctype/response			
Control[1]	category = 0000 ₂ (AKE)				AKE_ID = 0000 ₂			
Control[2]	subfunction							
Control[3]	AKE_procedure							
Control[4]	exchange_key							
Control[5]	subfunction_dependent							
Control[6]	AKE_label = FF ₁₆							
Control[7]	number = F ₁₆				status			

Figure 5 Status Packet Format

- Type, Length, and Control byte 0 are used to map DTCP to IP. Where the Type field identifies version 1 AKE control packet.
- ctype has the same values as referenced in Chapter 8 of DTCP specification and specified by the AV/C Digital Interface Command Set.
- Control bytes 1..7 are identical to operand bytes 0..6 as specified in Section 8.3.2.

V1SE.5.3.1 Modifications to AKE status command status field

Value	Status	Response code
0000 ₂	No error	STABLE
0001 ₂	Support for no more authentication procedures is currently available	STABLE
0111 ₂	Any other error	STABLE
1111 ₂	No information ¹⁴	REJECTED

Table 17 AKE Status Command Status Field

¹⁴ It is recommended that implementers not use the “No information” response.

V1SE.5.4 Modifications to 8.3.3

V1SE.5.4.1 AKE_ID dependent field

DTCP-IP implementations only require a single exchange key, specifically Bit 3 of exchange_key field will be used for transporting all DTCP Protected content over IP for all defined E-EMI.

For DTCP-IP both Source and Sink shall support only Full Authentication.

Therefore Restricted Authentication procedure (rest_auth) and Enhanced Restricted Authentication procedure (en_rest_auth) are prohibited. Extended Full Authentication procedure (ex_full_auth) is optional¹⁵ and not used to handle Bit 3 of Exchange_key field.

Bit	AKE_procedure
0 (lsb)	Prohibited
1	Prohibited
2	Full Authentication procedure (full_auth)
3	Extended Full Authentication procedure ¹⁶ (ex_full_auth, optional) ¹⁷
4 – 7 (msb)	Reserved for future extension and shall be zero

Table 18 AKE_procedure values

V1SE.5.4.2 Modifications to Authentication selection

Source supported authentication Procedures	Sink supported authentication procedures	
	Full_auth	Full_auth and Ex_full_auth
Full_auth	Full Authentication	Full Authentication
Full_auth and Ex_full_auth	Full Authentication	Extended Full Authentication

Table 19 Authentication selection

V1SE.5.4.3 Modification to Exchange_key values

DTCP-IP uses a single exchange key.

Bit	Exchange_key
0 (lsb)	Prohibited
1	Prohibited
2	Prohibited
3	Exchange key for AES-128
4 – 7 (msb)	Reserved for future extension and shall be zero

Table 20 Exchange_key values

¹⁵ Features of this specification that are labeled as “optional” describe capabilities whose usage has not yet been established by DTLA.

¹⁶ Devices that support extended device certificates use the Extended Full Authentication procedure described in this chapter.

¹⁷ Features of this specification that are labeled as “optional” describe capabilities whose usage has not yet been established by the 5C.

V1SE.5.5 Modifications to AKE Subfunctions

Subfunction modified for DTCP-IP are described in the DTCP specification available under license from the DTLA.

V1SE.5.6 Modifications to 8.4 Bus Reset Behavior

If TCP connection is broken during authentication procedure, both source and sink devices shall immediately stop authentication procedure.

V1SE.6 Modifications to Appendix A (Additional Rules for Audio Applications)

V1SE.6.1 Modification to A.1 AM824 audio

Rules described in sections A.1.1, A.1.2, and A.1.2.3 are not limited to AM824 and Mode A is regarded as Mode A0 for DTCP-IP.

V1SE.6.1.1 Modification to A.1.1 Type 1: IEC 60958 Conformant Audio

Any content format with ASE-CCI equivalent to SCMS shall be regarded as Type 1 Audio.

V1SE.6.1.2 Modification to A.1.2 Type 2: DVD-Audio

Any content format containing DVD-Audio content and having ASE-CCI as described in Section A.1.2.2 shall be regarded as Type 2 Audio.

V1SE.6.1.3 Modification to A.1.3 Type 3: Super Audio CD

Any content format containing Super Audio CD content and having ASE-CCI equivalent to that described in Section A.1.3.2 shall be regarded as Type 3 Audio.

V1SE.6.2 Modification to A.2 MPEG Audio

Audio transmission via MPEG transport stream is permitted. Note that MPEG audio with ASE-CCI equivalent to SCMS is also Type 1 audio.

V1SE.7 Modification to Appendix B (DTCP_Descriptor for MPEG Transport Stream)

V1SE.7.1 Modification to B.1 DTCP_descriptor

As no standardized method for carrying Embedded CCI in the MPEG-TS is currently available, the DTLA has established the DTCP_descriptor and DTCP_audio_descriptor to provide a uniform data field to carry Embedded CCI in the MPEG-TS. When MPEG-TS format audiovisual content is protected by DTCP, the DTCP_descriptor shall be used to deliver Embedded CCI information to sink devices. DTCP_audio_descriptor is defined for audio transmission which uses Type 1 Audio specified in Section V1SE.6.1.1.

V1SE.7.2 Modification to B.2 DTCP_descriptor syntax

DTCP_audio_descriptor is defined for audio transmission in addition to DTCP_descriptor defined in Section B.2. The first bit value of Private_data_type is used to distinguish DTCP_descriptor and DTCP_audio_descriptor.

In case of audio transmission, the following syntax is used, and DTCP_descriptor is referred to as DTCP_audio_descriptor.

The DTCP_audio_descriptor has the same syntax as DTCP_descriptor except for private_data_byte field. The definition of the private_data_byte field of the DTCP_audio_descriptor is as follows:

<u>Syntax</u>	<u>Size(bits)</u>	<u>Formats</u>
Private_data_type{		
Descriptor_ID	1	bslbf
Reserved	5	bslbf
DTCP_CCI_audio	2	bslbf
Audio_Type	3	bslbf
Reserved	5	bslbf
}		

Table 21 Syntax of private_data_type for DTCP_audio_descriptor

V1SE.7.2.1 Modification to B.2.1 private_data_byte Definitions:

Definition for the following fields is added for DTCP_audio_descriptor.

Descriptor_ID

This field indicates the kinds of descriptor.

Descriptor_ID	Meaning
0 ₂	DTCP_audio_descriptor
1 ₂	DTCP_descriptor

Table 22 Descriptor_ID**DTCP_CCI_audio**

This field indicates the embedded CCI states for the transmission of Type 1 audio content.

DTCP_CCI_audio	Meaning
00 ₂	Copy-free
01 ₂	No-more-copies
10 ₂	Copy-permitted-per-type
11 ₂	Not defined

Table 23 DTCP_CCI_audio**Audio_type**

This field indicates the Audio type.

Audio_type	Meaning
000 ₂	Type 1
001 ₂ ..111 ₂	Reserved for future extension

Table 24 Audio_type**V1SE.7.3 Modification to B.3 Rules for the Usage of the DTCP_descriptor****V1SE.7.3.1 Modification to B.3.1 Transmission of a partial MPEG-TS**

For the audio transmission following rules are applied.

When a partial MPEG-TS that includes one or more programs is transmitted using DTCP, Audio-Format-cognizant source function shall insert the DTCP_audio_descriptor into the PMT¹⁸ of each program for which ASE-CCI of Type 1 Audio is used and the ASE-CCI is not Copy-free. When the DTCP_audio_descriptor is inserted, it shall only be applied to the PMT.

An Audio-Format-cognizant source function shall set the DTCP_CCI_audio bits according to the ASE-CCI of Type 1 Audio provided for each program within the MPEG-TS. The DTCP_audio_descriptor shall be inserted into the program_info loop of the relevant PMT.

Additionally, if any of the Elementary Streams within a program are assigned specific ASE-CCI values of Type 1 Audio, Audio-format-cognizant source function shall set the DTCP_CCI_audio bits according to ASE-CCI of Type 1 Audio. The DTCP_audio_descriptor shall be inserted into the ES_info loop of the relevant PMT for the Elementary Stream.

When Audio related content that is required to be treated as audiovisual content is transmitted as a part of Audio program, Audio-Format-cognizant source function, according to the upstream license,

¹⁸ as described in the definition of ISO/IEC 13818-1

may insert DTCP_descriptor of the audio related contents to related ES_info loop in the Audio program.

V1SE.7.3.2 Modification to B.3.3.Treatment of the DTCP_descriptor by the sink device

This section replaces Section B.3.3 and describes the treatment of the DTCP_descriptor and DTCP_audio_descriptor when received by a sink device. When the function of the sink device is format cognizant and receives recognizable Embedded CCI other than the DTCP_descriptor and DTCP_audio_descriptor within an MPEG-TS, the alternative Embedded CCI shall take precedence over the information contained within the DTCP_descriptor or DTCP_audio_descriptor. Furthermore, the DTCP_descriptor and DTCP_audio_descriptor are only valid when they are inserted into the PMT. If a DTCP_descriptor or DTCP_audio_descriptor is found in another location, it shall be ignored.

When the only Embedded CCI detected is the DTCP_descriptor or DTCP_audio_descriptor, the DTCP_descriptor shall be regarded as the Embedded CCI described in Sections V1SE.4.11 and V1SE.4.12 except as otherwise noted, and the DTCP_audio_descriptor shall be regarded as the Embedded CCI described in Sections V1SE.4.18 , and interpreted as follows:

- If a DTCP_descriptor or DTCP_audio_descriptor is found in an ES_info loop of the PMT, the Embedded CCI value contained in the descriptor should only be used as the CCI for the specific ES for which the DTCP_descriptor or DTCP_audio_descriptor is associated.
- When the only Embedded CCI detected in an ES_info loop of an Audio program is DTCP_descriptor, the DTCP_descriptor shall be regarded as the Embedded CCI described in only Section V1SE.4.12.
- If a DTCP_descriptor and DTCP_audio_descriptor is not found in the ES_info loop for a specific ES, but is instead found in the program_info loop, the Embedded CCI values contained within the DTCP_descriptor or DTCP_audio_descriptor shall be used as the CCI for that ES.
- A program in a stream shall be regarded as Copy-free if the stream contains multiple programs and none of Embedded CCI, DTCP_descriptor and DTCP_audio_descriptor is detected in the program and a DTCP_descriptor or DTCP_audio_descriptor is detected in another program on the same stream.

V1SE.8 Additional Requirements

V1SE.8.1 Authentication Capability Constraint

For DTCP-IP both source and sink devices shall only use Full Authentication.

V1SE.8.2 Internet Datagram Header Time To Live (TTL) Constraint

TTL is described in RFC791 and the following requirements only apply to IP datagrams that transport DTCP AKE commands. Transmitting devices shall set TTL value of such transmitted IP datagrams to a value no greater than 3 and correspondingly receiving devices shall discard such received IP datagrams which have a TTL value greater than 3.

V1SE.8.3 802.11 Constraint

DTCP devices with integrated 802.11 must ensure that either WEP or other such equivalent protection mechanism (e.g. WPA or WPA2) is engaged prior to exchanging DTCP AKE commands and protected content via such an network interface. For interoperability purposes devices must have at least WEP capabilities. Please note that this requirement to use WEP may be amended to require use of successor technologies as designated by DTLA.

V1SE.8.4 DTCP-IP Move Protocol

This section specifies a transaction based Move protocol¹⁹ for a Move function using Mode C1 of E-EMI that uses a move specific Exchange key for each Move transaction. The transaction based Move protocol results in either the content being completely moved to the sink device (Success case) otherwise the content remain useable in the source with no usable content in the sink device (Cancel case). Source and sink devices that support the transaction based Move protocol shall support the requirements specified in this section.

The Move protocol consists of three parts; Move RTT-AKE, Move Transmission and Move Commitment. Each transaction based on the Move protocol (Move transaction) begins with Move request from a sink device and completes when the Move Commitment process completes or any one of these processes are canceled or aborted.

An unique Exchange key (K_{XM}) is generated specifically for each Move transaction during Move RTT-AKE. K_{XM} is used to calculate the Content key (K_C) used to encrypt the moved content. Content received by the sink device remains unusable until the successful completion of the Move Commitment phase of the Move transaction. Where upon successful completion of the Move Commitment phase the moved content in the source device is made unusable and the moved content in the sink device is made useable.

Both source and sink devices can cancel a Move transaction anytime before starting the Move Commitment process.

V1SE.8.4.1 Move RTT-AKE

Source devices generate an Exchange key (K_{XM}) specifically for the Move transaction and to calculate the Content key (K_C) used to encrypt the content to be moved during the Move transaction.

¹⁹ Without using this Move protocol, move of content based on Exchange key (K_X) may be performed as specified in V1SE.4.24.

Move RTT-AKE is used to exchange K_{XM} and associated protocol flow is shown in following figure.

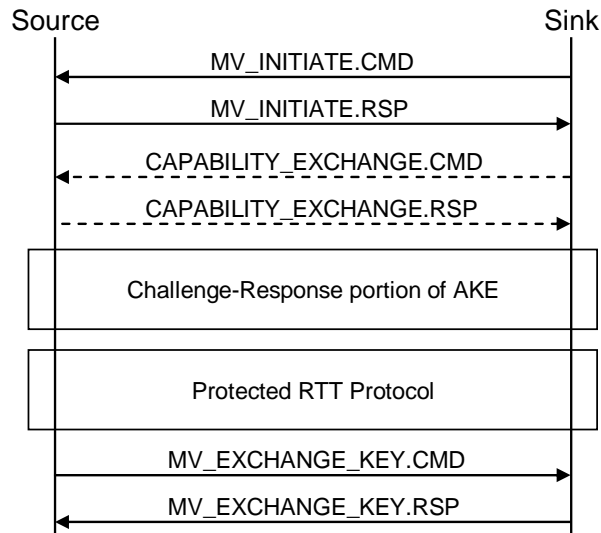


Figure 6 Move RTT-AKE Protocol Flow

1. Sink device initiates the Move RTT-AKE protocol by sending MV_INITIATE command. If source device can perform the DTCP-IP Move protocol, the source device returns response as accepted.
2. If sink device needs to exchange capabilities, the sink device may send CAPABILITY_EXCHANGE command at this point.
3. Challenge-Response portion of AKE and Protected RTT protocol (see section V1SE.8.5.1) are executed subsequently to share Authentication key for Move (HK_{AUTH}). In the Challenge-Response portion of AKE, source device performs the Sink counting specified in Appendix C of Volume 1 specification. Source device may skip Protected RTT Protocol when sink device is on its RTT Registry as specified in V1SE.8.5.2.
4. Source device generates Move Exchange key (K_{XM}) and send it to the sink device. (See the following section for detail)

V1SE.8.4.1.1 Establishing Move Exchange Key

Source device establishes the Move Exchange key (K_{XM}) and sends it to sink device with the following procedure:

1. The source device shall assign a random value for the Move Exchange key (K_{XM}) (using RNG_F) being established. The source device assigns K_{XM_label} to this K_{XM} .
2. The source device then scrambles the key K_{XM} using HK_{AUTH} (calculated using K_{AUTH}) resulting in K_{SXM} according to the function described in the DTCP Specification available under license from the DTLA.
3. The source device sends K_{SXM} and K_{XM_label} to the sink device.
4. The sink device descrambles the K_{SXM} using HK'_{AUTH} (calculated using K'_{AUTH}) to determine the shared K_{XM} according to the function described in the DTCP Specification available under license from the DTLA.

Source devices use the value of K_{XM_label} to identify corresponding Move transaction in Move Transmission and Move Commitment processes. Source devices shall not use the value of K_{XM_label} assigned to the Move transaction(s) that have not yet completed.

Source and sink devices shall manage K_{XM} and K_{XM_label} as follows:

- K_{XM} shall be managed independent of K_X in terms of generation and expiration. K_{XM_label} may have the same value as $exchange_key_label$.
- K_{XM} and K_{XM_label} can only be used in the corresponding Move transaction and shall not be used for other purposes.
- K_{XM} and K_{XM_label} shall be expired when the corresponding Move transaction completes regardless of result.
- It is mandatory that the source device expires a K_{XM} within 2 hours after Move Transmission using the K_{XM} has ceased.
- It is mandatory that the sink device expires a K_{XM} within 2 hours of continuous non-use of that K_{XM} for decryption.
- Source and sink devices must expire their K_{XM} when they detect themselves being disconnected from all mediums. For wireless mediums this means when device detects that it is not connected to an access point or it is not directly connected to another device.
- When K_{XM} is expired the K_{XM_label} shall also be expired except when the K_{XM_label} is stored for resumption of Move Commitment. (See section V1SE.8.4.3.1)

Note that source device shall not reset Sink Counter when K_{XM} is expired except for the case the source device shares neither Exchange key nor Move Exchange key other than the K_{XM} with any sink device.

V1SE.8.4.2 Move Transmission

Move Transmission process starts upon the completion of Move RTT-AKE and is the part of Move transaction where moved content is encrypted using Content key K_C calculated using K_{XM} instead of K_X and using Mode C1 (Move Audiovisual) of E-EMI in Move Transmission. (See section V1SE.4.2) The source device shall set the value of K_{XM_label} to $exchange_key_label$ field in PCP.

Source device shall not encrypt the same part²⁰ of content more than once using K_{XM} during a Move transaction. Source device shall prevent content from plural transmission for move.

Sink device shall keep the content received during Move Transmission unusable until successful completion of the Move Commitment process except for the use of the receiving content as if it has Mode C0 of E-EMI.

When HTTP is used for the Move Transmission, source device and sink device must not initiate another HTTP transfer²¹ for the Move Transmission before completing an HTTP transfer for the Move Transmission in a single Move transaction. Refer section V1SE.10.4 for recommended HTTP header field.

In the content key confirmation procedure during Move Transmission, K_{XM} shall be used instead of K_X to calculate MAC value by both source device and sink device. (See section V1SE.8.6: Content Key Confirmation) Source device shall manage the value of N_C in conjunction with the value of K_{XM_label} . (Note that there is only one N_C value for a K_{XM_label} at a time.) Source device shall compare received $N_C T$ with N_C corresponding to received K_{XM_label} .

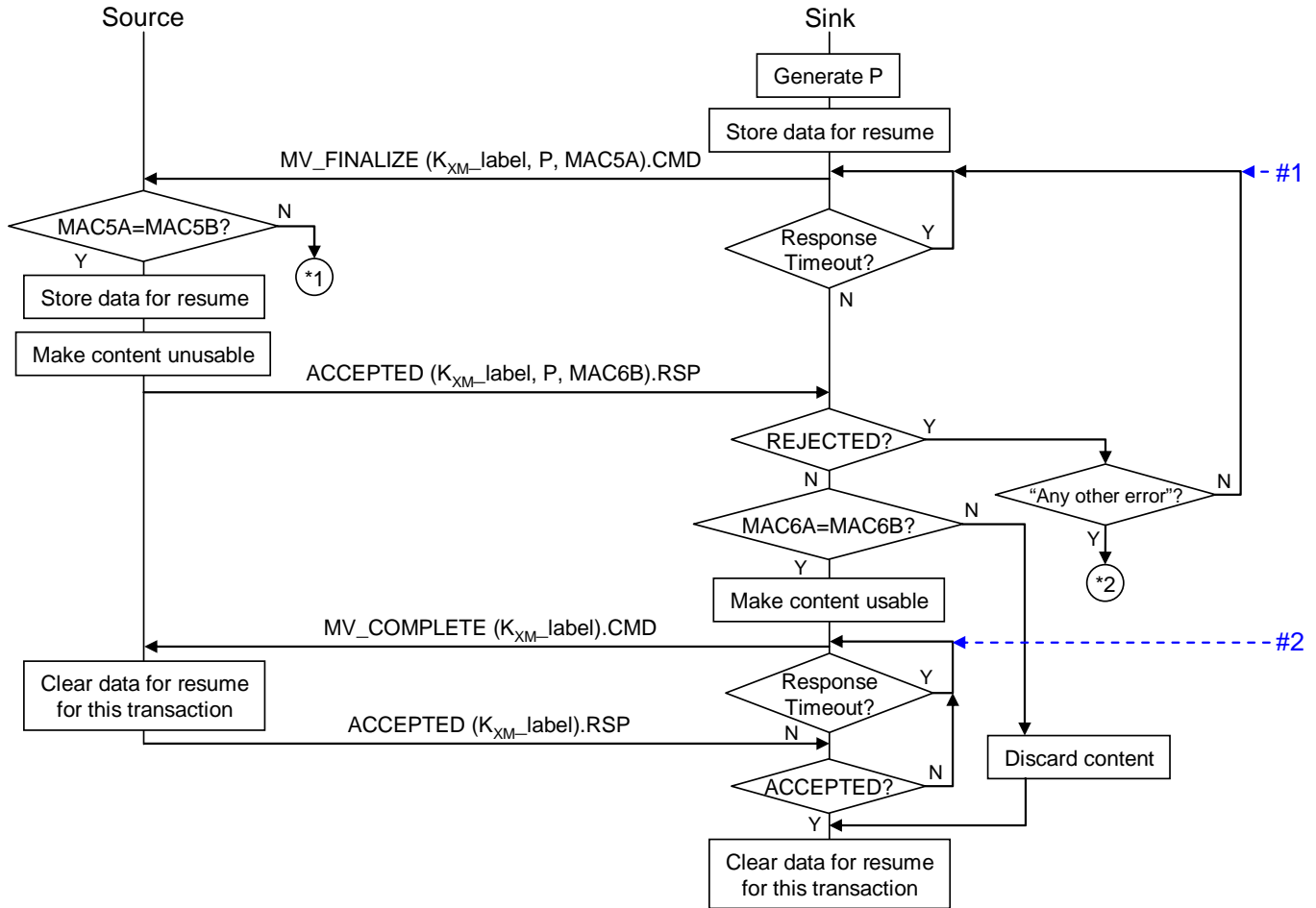
²⁰ The content may be retransmitted in transport protocol (ex. TCP).

²¹ Source devices may not be capable of supporting Move transaction via multiple HTTP transfers in a single Move transaction.

V1SE.8.4.3 Move Commitment

Sink devices initiate the Move Commitment process when Move Transmission has completed.

Sink device can make received content usable only upon the successful completion of the Move Commitment process. The following figure depicts the Move Commitment protocol flow.



*1: Source device is recommended to return REJECTED.RSP with "Any other error" status and keep waiting MV_FINALIZE.CMD. However, it may cancel the Move transaction if content has not yet been made unusable, then it should return REJECTED.RSP with "Any other error" and clear resume-data for this transaction (if stored).

*2: Sink device is recommended to resend MV_FINALIZE.CMD after reconfirming IP address of source device with which K_{XM} has been exchanged. However, it aborts the Move Commitment process if result is the same. When it aborts, it should clear resume-data for this transaction.

Figure 7 Move Commitment Protocol Flow

SHA-1 is used to construct following MAC values that are exchanged during Move Commitment protocol to ensure that the source device and the sink device which share K_{XM}.

- MAC5A = MAC5B = [SHA-1(MJ+P)]_{msb80}
- MAC6A = MAC6B = [SHA-1(MJ+P)]_{lsb80}

Where MJ is 160 bits and equal to SHA-1(K_{XM} || K_{XM}), and K_{XM} corresponds to K_{XM}_label in the MV_FINALIZE command. P is 64 bits random number (generated by RNG_F). "+" used in the above formula means mod 2¹⁶⁰ addition.

Source device computes MAC5B and compares it to MAC5A when MV_FINALIZE command is received. If not equal, the source device returns REJECTED response with "Any other error" status, else if

equal, it shall make content transmitted in Move Transmission unusable and returns ACCEPTED response to the sink device.

Sink device computes MAC6A and compares it to MAC6B when ACCEPTED response is received. If not equal, the sink device completes Move transaction and discards received content, else if equal, it makes content received in Move Transmission usable and sends MV_COMPLETE command to the source device.

When the sink device detects timeout before receiving ACCEPTED response to MV_FINALIZE command, it should resend the MV_FINALIZE command unless REJECTED response with "Any other error" status is received from the source device with which K_{XM} was exchanged.

Source device completes Move transaction after sending ACCEPTED response when MV_COMPLETE command is received. Sink device completes Move transaction when the ACCEPTED response is received.

When sink device detects timeout before receiving ACCEPTED response to MV_COMPLETE command, it should resend the MV_COMPLETE command not to leave data for the Move Commitment process in sink device (and source device).

V1SE.8.4.3.1 Resumption of Move Commitment

There is a brief period in the Move Commitment process where Moved content is marked unusable in both the source and sink device such that if an interruption (e.g. loss of TCP connection) were to occur at this point in the process it would result in loss of moved content. To avoid this, it is recommended that both source and sink device store²² required data²³ to complete Move Commitment protocol into NVRAM and perform the following resume procedure. The data is stored at the beginning and cleared at the end of Move Commitment protocol as shown in V1SE.8.4.3.

In case of broken AKE TCP connection, the TCP connection must first be reestablished between the affected source and sink device. When sink device cannot get DTCP socket without notification from source device (e.g. content-push type Moves), the source device should transmit HTTP POST request²⁴ with DTCP socket in the POST header to the sink device.

The sink device should execute the procedure shown below after communication with the source device is reestablished. Where #1 and #2 are the entry points specified in Figure 7.

²² At least the device should keep the stored data while the device is power-on.

²³ For example, parameters required in Move Commitment and information to discover device and moved content. Note that to keep this information unchanged is essential for resume of Move Commitment (e.g. UPnP AV CDS Object ID).

²⁴ To the same destination as Move Transmission without message-body.

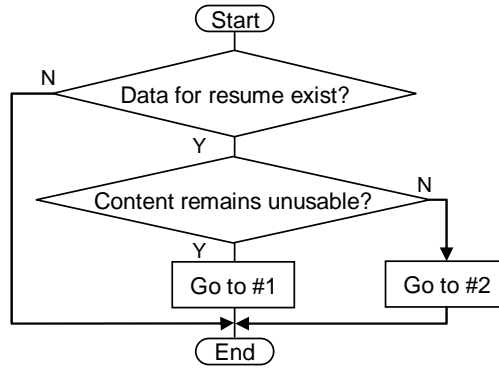
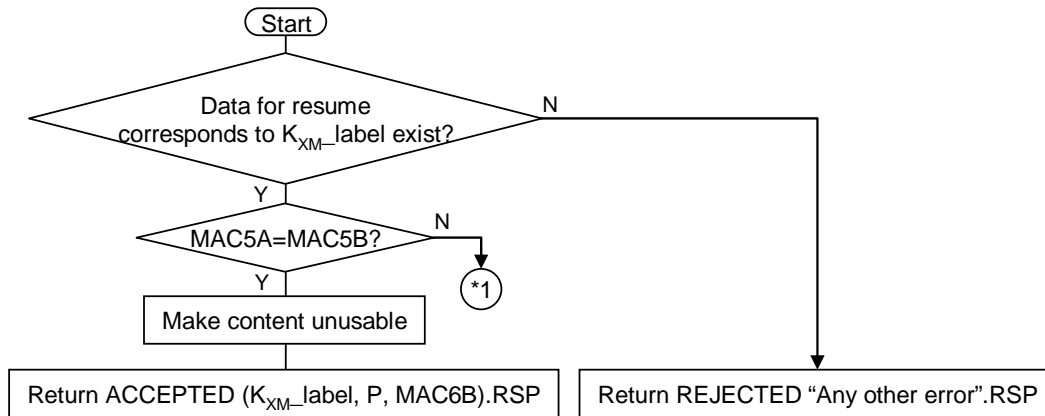


Figure 8 Resume procedure for sink device

The source device should execute the procedure shown below based on the K_{XM_label} specified in MV_FINALIZE or MV_COMPLETE command when one of these commands is received.



*1: Source device is recommended to return REJECTED.RSP with "Any other error" status and keep waiting MV_FINALIZE.CMD. However, it may cancel the Move transaction if content has not yet been made unusable, then it should return REJECTED.RSP with "Any other error" and clear resume-data for this transaction (if stored).

Figure 9 Resume procedure for source device when MV_FINALIZE is received

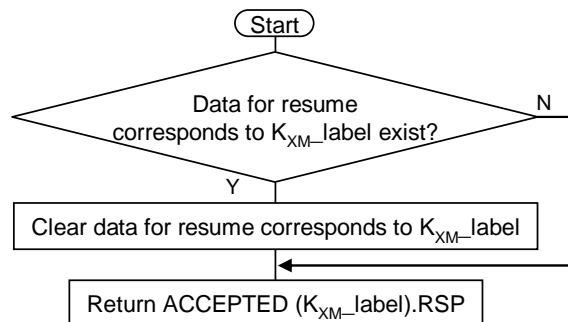


Figure 10 Resume procedure for source device when MV_COMPLETE is received

The source device should return ACCEPTED response to MV_COMPLETE command even when it has already cleared data for resume.

V1SE.8.4.4 Cancel of Move transaction

Source device can cancel Move transaction without disabling its content before issuing the first ACCEPTED response to MV_FINALIZE command. Sink device can cancel Move transaction as if it has received no content before issuing the first MV_FINALIZE command.

Sink devices which cancel a Move transaction shall discard content received in Move Transmission in the Move transaction.

When in the Move RTT-AKE process, the device desiring to cancel the Move transaction should send AKE_CANCEL command.

When in the Move Transmission process, the device desiring to cancel the Move transaction should send MV_CANCEL command. It is recommended that source and sink devices maintain the AKE TCP connection until completion of the MV_CANCEL command from source device.

When in the Move Commitment process, source device should return REJECTED response with "Any other error" status to MV_FINALIZE command when it cancels Move transaction. Source device shall not return REJECTED response with "Any other error" status to MV_FINALIZE command if it has already issued ACCEPTED response for MV_FINALIZE command for the Move transaction. Source and sink devices shall clear stored data for resume corresponds to the Move transaction being canceled.

V1SE.8.5 Additional Localization via RTT

Source and sink devices must implement Additional Localization as specified in this section.

Source devices with Additional Localization (AL) when conducting an AKE with a Sink device with AL must perform a RTT test if the sink device's Device ID is not on the source device's RTT registry.

Source devices will add a Sink device's Device ID to the Source device's RTT registry, will set the content transmission counter for the sink device to 40 hours, and will provide an exchange key only if the source device measures a RTT value of 7 milliseconds or less during RTT test.

Source devices when transmitting content will update content transmission counters of all RTT registered sink devices and are required to remove the Device ID of a sink device from the RTT registry after counting 40 hours of content transmission.

Background RTT testing is not a required capability. If background RTT testing is supported, the source device will add the sink device's Device ID to the RTT registry if not registered and set content transmission counter to 40 hours only if the source device measures a RTT value of 7 milliseconds or less during RTT test.

V1SE.8.5.1 Protected RTT Protocol

DTCP-IP's protected RTT protocol is described in Figure 11 and is used in RTT-AKE and Background RTT check procedures. The RTT protocol is executed after the Challenge-Response portion of the AKE is completed. SHA-1 is used to construct following messages that are exchanged during RTT testing protocol to ensure that source and sink which completed Challenge-Response portion of AKE are only ones involved in RTT testing.

- $MAC1A = MAC1B = [SHA-1(MK+N)]_{msb80}$
- $MAC2A = MAC2B = [SHA-1(MK+N)]_{lsb80}$
- $OKMSG = [SHA-1(MK+N+1)]_{msb80}$

Where MK is 160 bits and equal to $SHA-1(Kauth||Kauth)$, N is 16 bit number that ranges from 0 to 1023, and "+" used in RTT Protocol means mod 2^{160} addition.

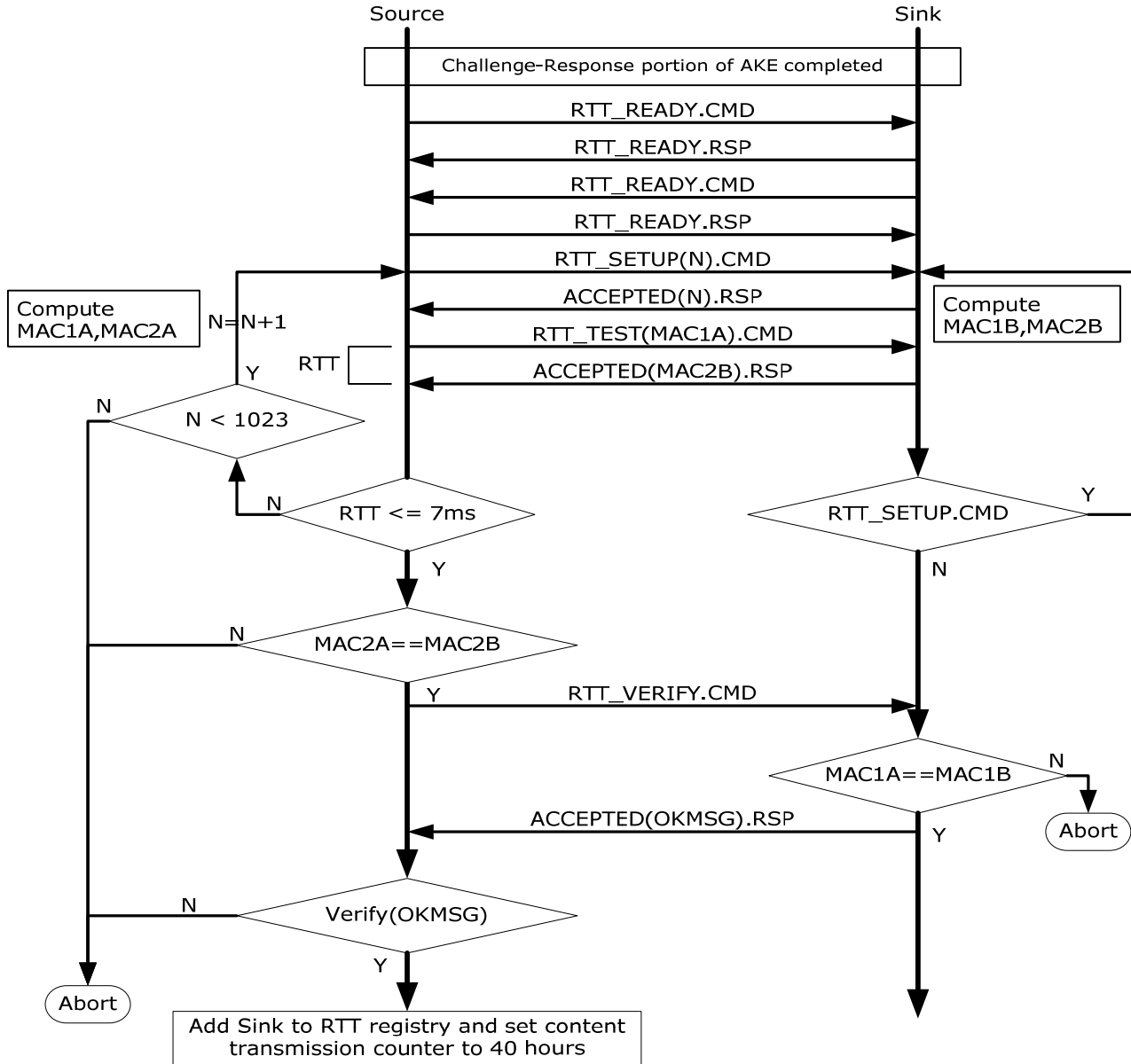


Figure 11 RTT Protocol Diagram

The RTT_READY command is used to indicate that authentication computation is complete and that source and sink devices are ready to execute the RTT test procedure.

The RTT procedure begins by first establishing value of N using the RTT_SETUP command. N is initially set to zero and can range from 0 to 1023 as maximum permitted RTT trials per AKE is 1024.

After preparation of MAC values corresponding to N, source device will then measure RTT which is the time interval starting after source transmits RTT_TEST command and terminates upon reception of RTT_TEST accepted response.

If the RTT is greater than 7 milliseconds and the value of N is less than 1023 the source will repeat RTT procedure by incrementing N by 1 and reissue RTT_SETUP and RTT_TEST commands.

If the measured RTT is less than or equal to 7 milliseconds:

The source device compares most recently computed MAC2A to most recently received MAC2B and if not equal the source device aborts RTT procedure else if equal it sends RTT_VERIFY command to sink device.

The sink device will after receipt of RTT_VERIFY command compare the most recently received MAC1A and most recently computed MAC1B and if not equal aborts RTT procedure else if equal it will send OKMSG in RTT_VERIFY accepted response.

The source device will verify OKMSG and if it is not correct the source device aborts RTT procedure else it will add sink device's Device ID to RTT registry and set content transmission counter to 40 hours.

If RTT procedure is aborted the source shall not provide an exchange key.

V1SE.8.5.2 RTT-AKE

The RTT-AKE procedure starts exactly the same as normal AKE but source and sink devices that have DTCP certificates with AL flag set to one must check AL flag value of other device and if the AL flag value is also set to one then:

The sink device after completing Challenge-Response portion of AKE will wait and the sink device will abort if it receives any other command than the RTT_READY command, EXCHANGE_KEY command, or AKE_CANCEL command.

The source device then examines the RTT registry and if the sink device's Device ID is on its RTT registry, the source device proceeds to exchange key portion of AKE otherwise the source device initiates a RTT test procedure and if during test it obtains a RTT measurement of 7 milliseconds or less it will add the sink device's Device ID to its RTT registry, set content transmission counter to 40 hours, and then proceed to exchange key portion of AKE.

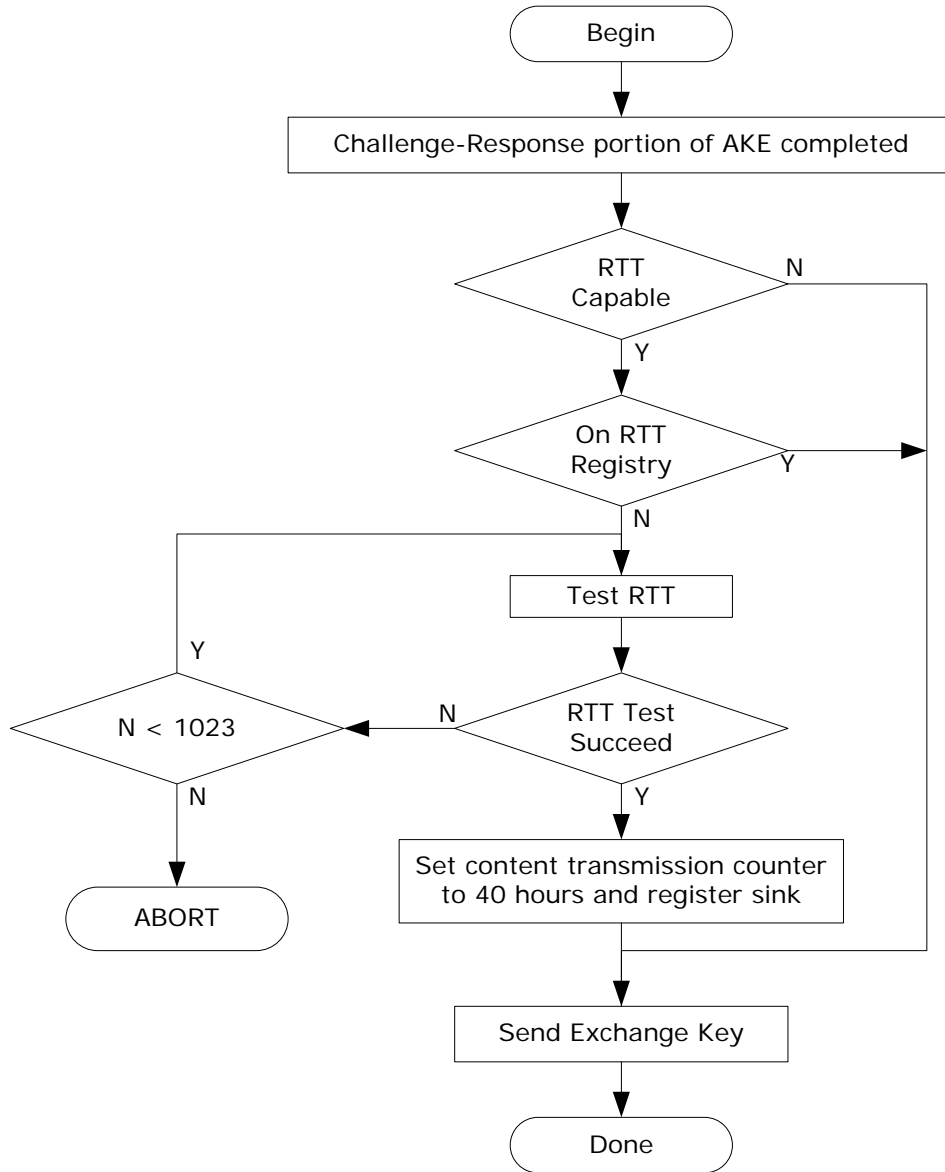


Figure 12 AKE-RTT Informative Flow Diagrams

V1SE.8.5.3 Background RTT Check

The Background RTT check procedure permits either the source or sink device to initiate an RTT background check which is only used to add sink device to source device's RTT registry if not on RTT registry or if already on the source device's RTT registry set the count transmission counter to 40 hours. In case of Background RTT check source devices shall not transmit an exchange key.

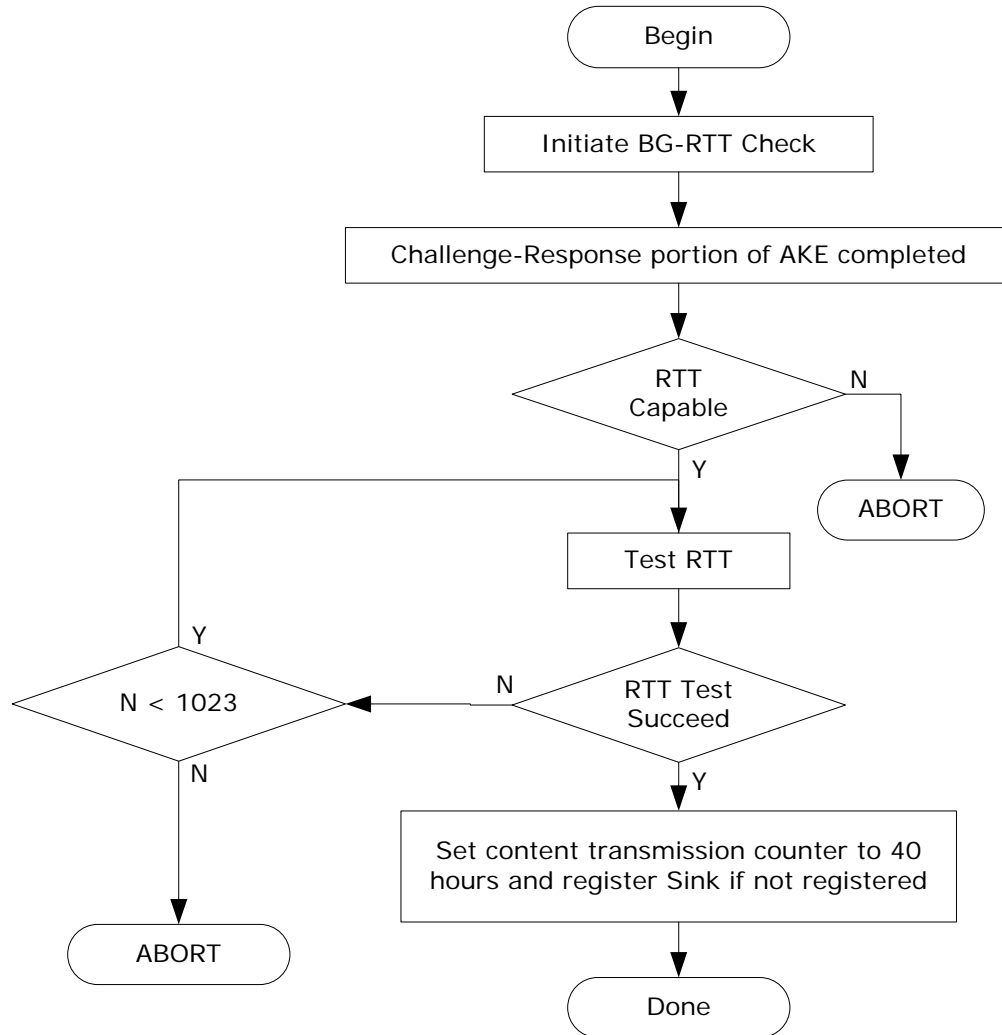


Figure 13 Background RTT Check Informative Flow Diagram

V1SE.8.6 Content Key Confirmation

For interoperability the content key confirmation function is limited to only those source and sink devices whose AL flag has a value of one. The sink device uses the CONT_KEY_CONF subfunction to confirm that the content key via the associated N_c is current.

Sink devices must monitor and confirm the N_c value of the most recently received PCP containing encrypted content for each content stream and then periodically reconfirm subsequent N_c (s) at least every 2 minutes. Periodic confirmation of N_c can be avoided if after initial confirmation the sink monitors and confirms that subsequent N_c values are monotonically increasing contiguous values.

Sink devices which confirmed that the source device of receiving content supports PCP-UR may use SN_c as a substitute for N_c .

Per content stream, sink devices after an initial non-confirmation of a N_c have one minute to repeatedly attempt to confirm a subsequent N_c value before they must terminate decryption for that content stream.

Sink devices may restart decryption upon confirmation of any N_c after a N_c non-confirmation event.

The content key confirmation procedure requires the sink device to send the N_c value under test (N_{cT}) to the source device. Upon receipt the source device checks the received N_{cT} against its current N_c values and if any are within the range N_{cT} to $N_{cT}+5$ then it confirms that N_{cT} is valid. Note that source devices which support PCP-UR shall use only the least significant 48 bits of both N_c and N_{cT} for this check since upper 16 bits are used for PCP-UR. The confirmation procedure is depicted in following figure.

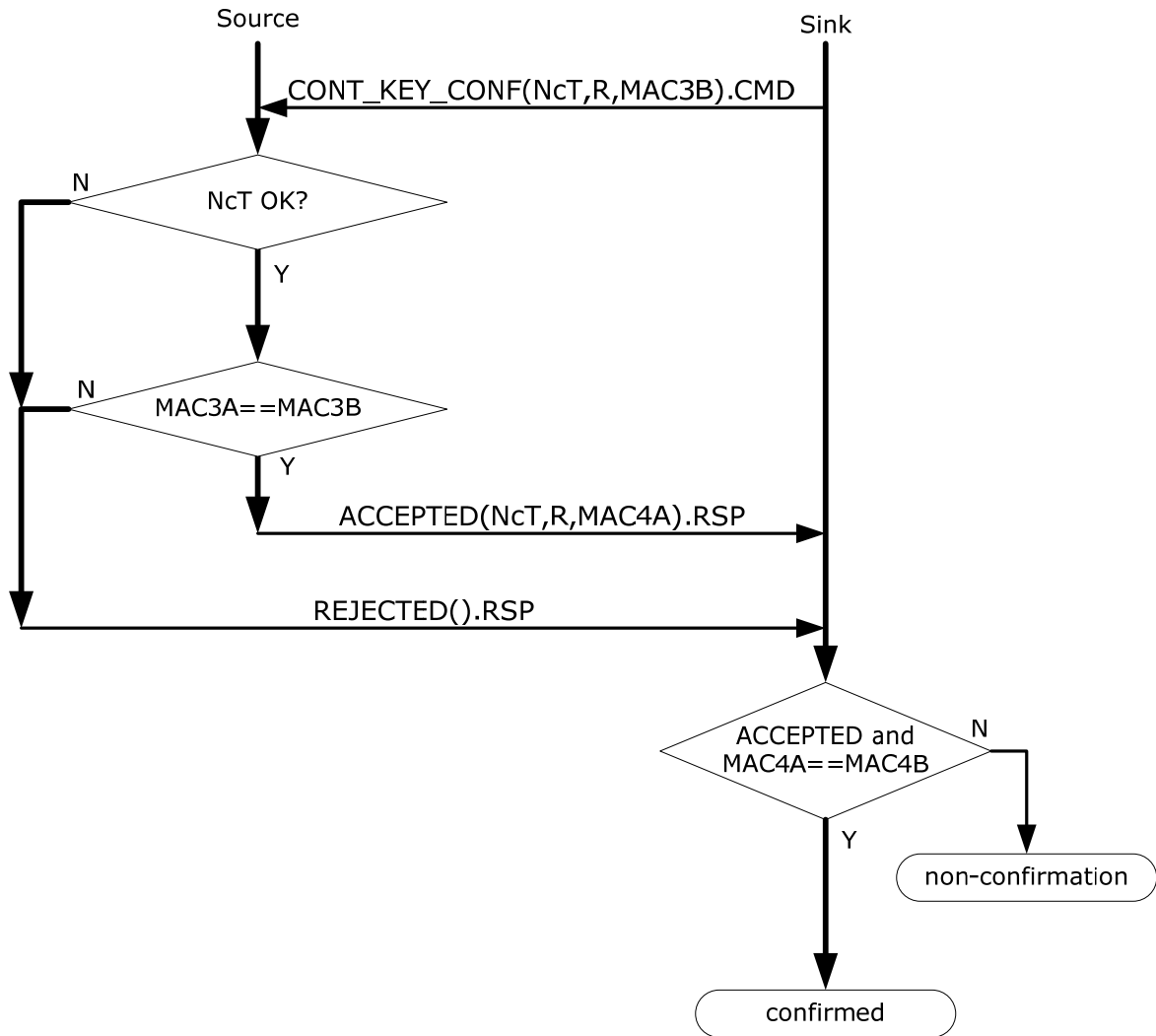


Figure 14 Content Key Confirmation Procedure

Where:

$$MX = \text{SHA-1}(Kx || Kx),$$

R is 64 bits, its initial value is a random number and is incremented by $1 \bmod 2^{64}$ for subsequent trials.

$$\text{MAC3A} = \text{MAC3B} = [\text{SHA-1}(MX + N_{cT} + R)]_{\text{msb80}}$$

$$\text{MAC4A} = \text{MAC4B} = [\text{SHA-1}(MX + N_{cT} + R)]_{\text{lsb80}}$$

"+" used in the above formulas means $\bmod 2^{160}$ addition

V1SE.9 Additional Commands and Sequences

These additions defined for DTCP-IP are described in the DTCP specification available under license from the DTLA.

V1SE.10 Recommendations

V1SE.10.1 Recommended MIME type for DTCP protected content

DTCP application media type is as follows:

application/x-dtcp1; CONTENTFORMAT=<mimetype>

Where **CONTENTFORMAT**, is the standard content media type that is protected by DTCP.

In addition, information identifying DTCP Socket may be included as follows:

**application/x-dtcp1; DTCP1HOST=<host>; DTCP1PORT=<port>;
CONTENTFORMAT=<mimetype>**

Refer to V1SE.10.2.1 for description of **DTCP1HOST** and **DTCP1PORT**.

Content type of HTTP response / request is set to DTCP application media type.

V1SE.10.2 Identification of DTCP Sockets

DTCP uses a TCP port to support various command and control protocols (i.e. AKE, Exchange Keys, SRM,,,) and either TCP or UDP for content transport. This section details recommend practices for identifying DTCP Sockets.

V1SE.10.2.1 URI Recommended Format

This following information is inserted into the query string portion of URI and is used to communicate the source's content and DTCP Socket to the sink. The source obtains the sink's DTCP Socket when the sink establishes a TCP connection to the source.

<service>://<host>:<port>/<path>/<FileName>.<FileExtention>?CONTENTPROTECTIONTYPE=DTCP1&DTCP1HOST=<host>&DTCP1PORT=<port>

Where:

CONTENTPROTECTIONTYPE, is set to "DTCP1" where 1 represents a DTCP-IP version number that can be incremented in future as the needed.

DTCP1HOST specifies the IP address and **DTCP1PORT** specifies the port number of the DTCP Socket of the source device.

V1SE.10.2.2 HTTP response / request

Content type of HTTP response /request²⁵ is set to DTCP application media type as follows:

**Content-Type: application/x-dtcp1 ; DTCP1HOST=<host> ; DTCP1PORT=<port> ;
CONTENTFORMAT=<mimetype>**

²⁵ For example, HTTP POST request with "Expect: 100-continue" header.

V1SE.10.3 Header Field Definition for HTTP

The following header fields are defined for HTTP transfers.

V1SE.10.3.1 Range.dtcp.com

The Range.dtcp.com header is used in the same manner as the RANGE header defined in RFC 2616 except that range specification applies to the content before DTCP processing.

V1SE.10.3.2 Content-Range.dtcp.com

The Content-Range.dtcp.com header is used in the same manner as the CONTENT-RANGE header defined in RFC 2616 except that range specification applies to the content before DTCP processing.

V1SE.10.4 BLKMove.dtcp.com

The BLKMove.dtcp.com header is used to specify which K_{XM} is used in the Move Transmission process specified in V1SE.8.4.2. K_{XM_label} is a parameter of this header as follows:

BLKMove.dtcp.com: < K_{XM_label} >

< K_{XM_label} > is denoted in hexadecimal 2 digits.

V1SE.10.5 Definition for UPnP AV CDS²⁶ Property

The following is defined for properties in UPnP AV CDS.

V1SE.10.5.1 DTCP.COM_FLAGS param

The DTCP.COM_FLAGS param is used in the 4th field of res@protocolInfo property to show static attribute of content regarding DTCP transmission. The DTCP.COM_FLAGS param is 32 bits field, and bit definition is as follows:

- Bit 31: DTCP Movable
- Bit 30: Move protocol specified in V1SE.8.4 is supported
- Bit 29-0: Reserved (zero)

Bit 31 is set to one if associated content can be moved using DTCP. Bit 30 is also set to one if the content can be moved based on the Move protocol in V1SE.8.4. When only bit 31 is set to one, the Move protocol²⁷ in V1SE.8.4 cannot be used. Reserved bits are set to zero. Devices refer to the reserved bits ignore the value.

The 32 bits value of DTCP.COM_FLAGS param is denoted in hexadecimal 8 digits.

V1SE.10.5.2 res@dtcp:uploadInfo

The res@dtcp:uploadInfo property is used to show how the content is uploaded using DTCP. The res@dtcp:uploadInfo property is 32 bits field, and bit definition is as follows:

²⁶ Refer to UPnP ContentDirectory:2 document.

²⁷ Without using this Move protocol, move of content based on Exchange key (K_X) may be performed as specified in V1SE.4.24.

- Bit 31: Content will be moved using DTCP Move
- Bit 30: Move protocol specified in V1SE.8.4 will be used
- Bit 29-0: Reserved (zero)

Bit 31 is set to one if associated content will be moved using DTCP. Bit 30 is also set to one if the move will be executed based on the Move protocol in V1SE.8.4. When only bit 31 is set to one, the Move protocol²⁷ in V1SE.8.4 is not used.. Reserved bits are set to zero. Devices refer to the reserved bits ignore the value.

The 32 bits value of res@dtcp:uploadInfo is denoted in hexadecimal 8 digits.

The definition of XML namespace whose prefix is "dtcp:" is "urn:schemas-dtcp-com:metadata-1-0/".