DRM Content Format – MPEG-2 TS Profile: MDCF

Approved Version 1.0 – 05 Jul 2011

Open Mobile Alliance

OMA-TS-SCE-MDCF-V1_0-20110705-A

Contents

Figures

Tables

1. Scope

Open Mobile Alliance (OMA) specifications are the result of continuous work to define industry-wide interoperable mechanisms for developing applications and services that are deployed over wireless communication networks.

The scope of OMA "Digital Rights Management" (DRM) is to enable the distribution and consumption of digital content in a controlled manner. The content is distributed and consumed on authenticated Devices per the usage rights expressed by the content owners. OMA DRM work addresses the various technical aspects of this system by providing appropriate specifications for content formats, protocols, and a rights expression language.

A number of DRM specifications have already been defined within the OMA. The latest approved release of the OMA DRM enabler ([OMADRM20], including [DRMDRM20], [DRMDCF20], [DRMREL20]), is referred to within this document as "OMA DRM 2.0".

The scope for this specification is to extend the OMA DRM specification to allow for distribution of OMA DRM protected content and associated metadata contained in an MPEG-2 transport stream. For this purpose this specification defines a specific DRM Content Format profile, called MDCF. Release 2 of the "*Digital Rights Management*" specification [DRM-v2] specifies the DCF and PDCF formats which are both derived from the ISO File Format. The MDCF format is not derived from the ISO File Format, but instead based on MPEG2 Transport Streams. The delivery of MDCF compliant MPEG-2 transport streams is beyond the scope of the MDCF specification; however, the MDCF format is defined so that an MDCF compliant MPEG-2 transport stream can be delivered conveniently via commonly deployed broadcast methods, such as via a Conditional Access Terminal.

2. References

2.1 Normative References

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt

[RFC4234] "Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell. October 2005, URL:http://www.ietf.org/rfc/rfc4234.txt

[SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[ISO/IEC 13818-1] ISO/IEC 13818-1, Information Technology – Generic Coding of moving pictures and associated audio information: Systems

[DRM-DCF-v2.1] "DRM Content Format, DRM 2.0.1", Open Mobile Alliance™, OMA-TS-DRM_DCF-V2_0_1-20080226-A, URL:http://www.openmobilealliance.org/

"DRM Content Format, DRM 2.1", Open Mobile Alliance™, OMA-TS-DRM_DCF-V2_1-20070724-C, URL:http://www.openmobilealliance.org/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version x.y, Open Mobile Alliance™, OMA-ORG-Dictionary-Vx_y, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Broadcast Program	A logical portion of a Broadcast Service with a distinct start and end time. In the case the Broadcast Program is not free-to-air, it can be offered individually for purchase, such as “Pay-Per-View”, or as part of a parent service (e.g. subscription service). A Broadcast Program may for example represent a movie, news show or soccer game.
Broadcast Service	A digital broadcast service delivered in an MPEG-2 transport stream consisting of a concatenation of Broadcast Programs, as defined in an MPEG-2 Program Map Table (PMT).
Interaction Channel	A bi-directional channel used to engage in communication protocols (such as DRM v2 ROAP) with other entities. The Interactive Channel can for example be used to request a Rights Object from a Rights Issuer.

3.3 Abbreviations

OMA	Open Mobile Alliance
MDCF	MPEG-2 Transport Stream DRM Content Format
STB	A Set Top Box. A device capable of receiving digital broadcast services contained in an MPEG-2 transport stream that may be delivered over cable, satellite, terrestrial, IP or any other medium. To access the digital broadcast services, a Set Top Box may or may not use a Conditional Access System. A STB may or may not be OMA DRM compliant.
CAS	Conditional Access System

4. Introduction

ISO/IEC International Standard 13818-1 ([ISO/IEC 13818-1]) , a.k.a MPEG2-System, specifies generic methods for multimedia multiplexing, synchronization and time base recovery. The Transport Stream (TS) specified therein is used as the stream format in many systems for digital television broadcast in the Consumer Electronics domain.

This specification defines a specific DRM Content Format profile that enables Content Providers to easily enhance their existing and deployed digital television services by allowing OMA DRM terminals with MPEG-2 TS support to access broadcast content. It enables OMA DRM terminals to request Rights Objects for access of broadcast Media Objects that, for example, was transported to the home using an [ISO/IEC 13818-1] based content delivery system and then distributed to an OMA DRM terminal using local connectivity. The figure below depicts a typical environment for delivery of broadcast content to the home using a Conditional Access (CA) terminal.

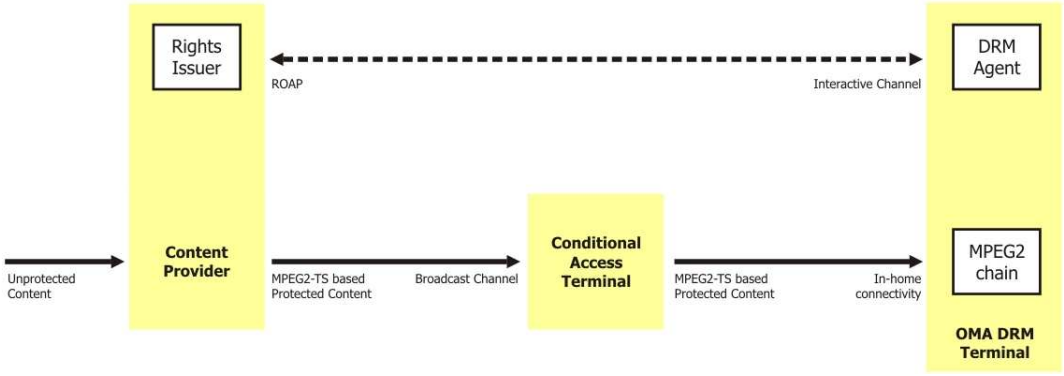

Figure 1: Typical environment for delivery of broadcast content to the home

This specification addresses the format of the broadcast content for access by an OMA DRM terminal with MPEG-2 TS support, as well as protocols for communication between the OMA DRM terminal and the Rights Issuer needed to access the broadcast content. The broadcast content delivery system and the Rights Issuer are beyond the scope of this specification. Hence this specification does neither require the presence nor the absence of a CA terminal. However, when a CA terminal is present, this specification allows that by means simulcrypt mechanisms beyond the scope of the specification the same broadcast content can be accessed in CA terminals (without OMA DRM support) and in OMA DRM terminals with MPEG-2 TS support conforming to this specification.

The specific DRM Content Format profile defined in this specification to support OMA DRM protected (broadcast) content contained in MPEG-2 transport streams is called the MPEG-2 transport stream DCF: MDCF. An MDCF is an MPEG-2 transport stream consisting of concatenated MPEG-2 transport packets that contain the (protected) broadcast content and the associated metadata needed to access the broadcast content. It should be noted that each MPEG-2 transport streams for broadcast is typically self-contained so as to allow for random access without out-of-band information. Thereby an MDCF can be considered as a file that typically carries all information needed to access the contained broadcast content.

When broadcast content is intended to be delivered to OMA DRM terminals, the Content Provider embeds the OMA specified information that is needed to create an MDCF file into the MPEG-2 TS prior to transport over the broadcast channel. The MDCF files may be broadcast directly to OMA DRM terminals or via CA terminals. When delivered via a CA-terminal, which is not necessarily OMA DRM conformant, then the CA terminal can create an MDCF file simply by storing the data received over the broadcast-channel into a file, without any modifications to the data. These MDCF files may then be distributed to OMA DRM terminals. To access the broadcast content contained in an MDCF, an OMA DRM Terminal needs to retrieve the associated Right Objects using OMA specified information in the MDCF.

To avoid the need for broadcasting content in multiple formats, the MDCF format re-uses commonly deployed methods for encryption of broadcast content. Various methods are defined by standardization bodies for broadcast applications, such as ARIB in Japan, ATSC in the USA and DVB in Europe. As a consequence, MDCF does not specify the encryption method, but instead signals which encryption method is used. This has the important advantage that an OMA DRM Terminal may use

commonly available hardware and software solutions for demultiplexing, decrypting and decoding of MPEG-2 transport streams.

4.1 Version 1.0

The MDCF 1.0 Enabler specifies the use of OMA DRM content protection in MPEG-2 transport streams and the access of such content by means of an OMA DRM Rights Object. In MDCF 1.0 it is assumed that Rights Objects to access OMA DRM protected content in an MPEG-2 transport stream are delivered over an Interactive Channel; the carriage of such ROs in an MPEG-2 transport stream is beyond the scope of MDCF 1.0.

5. The MPEG-2 transport stream Profile of DCF (MDCF)

[ISO/IEC 13818-1] does not specify mechanisms for content or service protection. Instead it allows for (partial) encryption of the TS and it supports (simultaneous) implementation of content protection mechanisms through generic system data stream definitions. This infrastructure defined in [ISO/IEC 13818-1] is used by this specification as a basis for an OMA defined key management schema, potentially in addition to non-OMA CA systems used by CA terminals. This specification defines data structures for embedding into an MDCF as Entitlement Control Messages (see next sections). These data structures serve two purposes:

- To transport the information needed by an OMA DRM terminal to request a Rights Object that is associated to the broadcast content, typically a program. This information corresponds to the information carried in the Common Headers as defined by OMA DRM 2.0 – DCF [OMA DCF].
- To transport the currently used content encryption key. This message is similar to the STKM as defined by OMA BCAST.

Although envisioned and enabled in [ISO/IEC 13818-1], this specification does not specify data structures to enable insertion of OMA DRM Rights Objects in the TS stream itself. Instead this specification envisions the Rights Objects to be retrieved via the interactive channel.

5.1 The MPEG-2 transport stream structure (Informative)

The Transport Stream defined in [ISO/IEC 13818-1] is a multiplex of packetized audio, video and other data, associated with (potentially) a number of simultaneously running programs and services. It consists of 188 bytes long Transport Stream packets, all of which have header with a Packet Identifier (PID). Typically the payload of all packets with the same PID constitutes a single elementary video-stream, audio-stream or data stream.

The data contained in TS packets with a PID of 0x00 constitutes the Program Association Table (PAT) - a directory of all currently embedded programs and services in the TS. Per program the PAT contains the PID value of the TS-packets that make up the Program Map Table (PMT) for that program. The PMT contains more information on a certain program, including (if applicable) content protection related information.

[ISO/IEC 13818-1] supports protection of content, for broadcast typically by means of Conditional Access (CA) systems, by allowing the encryption of the payload of TS-packets (see section 5.2). To increase the level of security, the encryption keys may be changed frequently. [ISO/IEC 13818-1] enables the signalling of TS-packet encryption and changes in encryption keys and it enables a Conditional Access system to embed “private” data into the TS to communicate key material to terminals.

For example, a broadcast may use “odd keys” and “even keys” to encrypt the content, with indication on transport packet level which key is used. Each key is used during a certain period. During the period that the "odd key" is used for decryption in the terminal, the next "even key" is conveyed in the transport stream, and when this period is elapsed, the transport packets indicate that the "even key" is to be used for decryption. Now the next "odd key" is conveyed in the transport stream to ensure its availability once the MPEG-2 TS packets indicate that the next "odd key" must be used. Etc. In this way the keys are provided in a timely manner, prior to the usage of the next key.

[ISO/IEC 13818-1] specifies two locations for CA systems to embed CA related information in the TS. Firstly there may be the Conditional Access Table (CAT), contained in TS-packets with a PID value of 0x01. This table contains, for each CA system that is enabled to provide access to the TS, a Conditional Access descriptors that contains the PID of the TS-packets that contain the Entitlement Management Messages (EMMs) for that CA system. The EMMs are intended to transport long-term key material, comparable to OMA DRM Rights Objects and OMA BCAST BCROs. Secondly there are Conditional Access descriptors located in the PMT of a program, containing the PID of the TS-packets that contain the Entitlement Control Messages (ECMs) for that program and a certain CA system. The ECMs are intended to transport short-term key material that allow for frequently changing content encryption keys, comparable to the OMA BCAST STKMs.

There may be CA-descriptors for more than one Conditional Access system, all of which may then be able to provide access to the set of video, audio and data streams that constitute the program or service. This is usually referred to as “simulcrypt”.

If such simulcrypt is applied in a broadcast using key switching, then both CA systems must ensure that the keys are provided in a timely manner.

As Conditional Access systems are beyond the scope of MPEG-2 TS, [ISO/IEC 13818-1] only describes an overall structure for the EMMs and ECMs, leaving the specification of further details for encryption methods and key management to standardization bodies for broadcast systems and / or to proprietary systems.

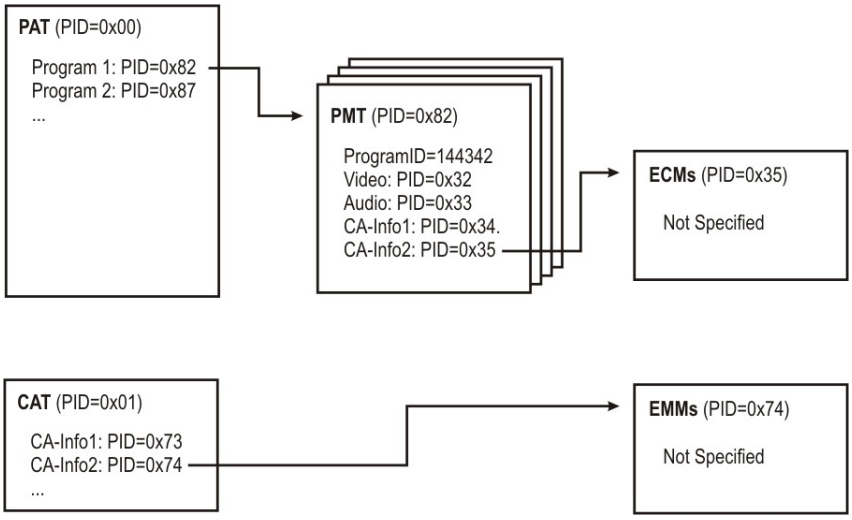

Figure 2: Example of the use Program Association Table (PAT), Program Map Table (PMT) and Conditional Access Table (CAT) to signal carriage of ECMs and EMMs

ISO/IEC 13818-1 specifies which audiovisual streams are included within a program, but no further distinction is made about the content contained in a program. As a consequence, a program in an MPEG-2 TS may represent a broadcast concatenation of an episode of a comedy series, a sport event, a news show, a late night movie, etc. To identify the comedy episode, the sport event, the news show and the movie within an MPEG-2 TS defined program, requires tools not specified in ISO/IEC 13818-1.

5.2 MPEG-2 transport stream scrambling

ISO/IEC 13818-1 enables two ways to protect a stream: “Transport Stream level scrambling” and “PES level scrambling”. Independent of TS or PES level descrambling, the descrambler shall use content keys for descrambling scrambled (TS or PES) packets and the keys may be changed regularly.

5.2.1 Transport Stream level scrambling

To protect network broadcasts the broadcaster may "scramble" (a.k.a. encrypt) the transport stream (a.k.a. TS). In this case, scrambling takes place after multiplexing the payload of the transport packet. The receiving Device will have to "descramble" (a.k.a. decrypt) the TS so the audio and/or video and/or data parts can be consumed. This is typically done with a piece of hardware called the "descrambler". The descrambler is controlled by the Transport Stream Control (a.k.a. TSC) bits in the TS packet; see **Table 1 – Definition of transport_scrambling_control bits**. Note however that only the value ‘00’ is specified by ISO/IEC 13818-1 and that the semantics of the other values is left to standardization bodies for broadcast applications, such as ARIB, ATSC and DVB. As an example, in the following sub-clause Transport Stream level scrambling in DVB applications is described.

Table 1 – Definition of transport_scrambling_control bits

Transport Stream Control bits	Description
00	No descrambling.
01	User defined.
10	User defined.
11	User defined.

5.2.1.1 Transport Stream level scrambling in DVB applications (Informative)

The DVB descrambler knows when it has to descramble or not by looking at the Transport Stream Control (a.k.a. TSC) bits in the TS packet as defined in Table 1. Limitations to TS level scrambling will adhere to ISO/IEC 13818-1.

Table 2 – Definition of transport_scrambling_control bits in DVB

Transport Stream Control bits	Description
00	No descrambling.
01	Reserved.
10	Scrambling by the EVEN content key.
11	Scrambling by the ODD content key.

Within DVB applications, there are two possible descrambler implementations: single key and dual key descramblers.

- Single key descramblers have one register to store a descrambler key. To change the key they have to overwrite the key in the register. This cannot be done when the descrambler is in the middle of descrambling a packet. Therefore, the broadcaster usually alternates the scrambled transmission by transmitting packets in the clear a short interval (mostly 1 second). The descrambler’s register is then updated with the new key. After the clear interval expires, the broadcaster sends TS packet that are scrambled with the new key. This cycle can repeat itself endlessly.

- Dual key descramblers have two registers so they can store two keys: the first register can contain the key the descrambler is currently using and the second register can be updated with a new key for the next keying period. To distinguish the registers they are identified as the odd and even key register. The TSC bit in the TS packet indicates if the descrambler needs to use the key in the odd or even key register in order to descramble the TS packet and flips to corresponding register when necessary.

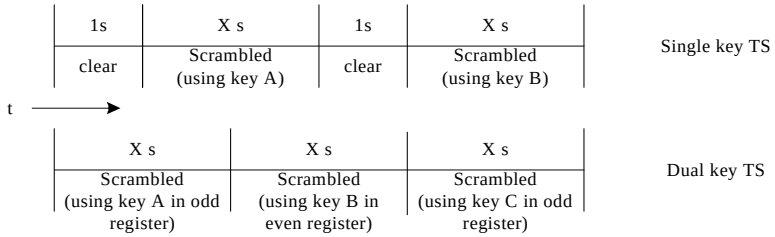

Figure 3 – Single key vs. dual key TS over time

The delivery of keys is beyond the scope of ISO/IEC 13818-1. For the MDCF, these keys are delivered in STKM-ECMs (see 5.4.1). The STKM signals if the delivered traffic key is "odd" or "even". Because of the aforementioned behaviour, Table 3 – Descrambling possibility matrix, applies.

Table 3 – Descrambling possibility matrix

	Single key descrambler	**Dual key descrambler**
Single key TS	De facto	Possible [a)]
Dual key TS	Impossible [b)]	De facto
[a)] TSC bits in single key TS will be stationary odd or even. Two solutions possible: 1) detect TSC status and put key in correct register, or 2) put the existing key in both the odd and even register at the same time. [b)] There is one second in the clear in dual key TS to change the key.		

5.2.2 PES level scrambling

Instead of scrambling all the content at the TS level, one or more of the Packetised Elementary Streams (a.k.a. PES) may be scrambled. In this case, scrambling generally takes place at the source, before multiplexing. The descrambler is controlled by the 2 bit PES scrambling control field in the PES packet header; see **Table 4 – Definition of PES_scrambling_control field bits**. Note however that only the value '00' is specified by ISO/IEC 13818-1 and that the semantics of the other values is left to standardization bodies for broadcast applications, such as ARIB, ATSC and DVB. As an example, in the following sub-clause PES level scrambling in DVB applications is described..

Table 4 – Definition of PES_scrambling_control field bits

PES_scrambling control field	**Description**
00	No descrambling.
01	User defined.
10	User defined.
11	User defined.

5.2.2.1 PES level scrambling in DVB applications

When PES level scrambling is used in DVB applications, the descrambler knows when it has to descramble or not by looking at the 2 bit PES scrambling control field in the PES packet header as defined in **Table 5 – Definition of pes_scrambling_control field bits**. Limitations to PES level scrambling will adhere to ISO/IEC 13818-1.

Table 5 – Definition of pes_scrambling_control field bits

PES_scrambling control field	Description
00	No descrambling.
01	No descrambling.
10	Scrambling by the EVEN content key.
11	Scrambling by the ODD content key.

5.3 CA Descriptor usage in MDCF

In this section it is defined how to use the CA descriptor, as specified in ISO/IEC 13818-1, within an MDCF. The presence of the MDCF CA descriptor signals the transport of MDCF compliant ECMs. **Table 6 – MDCF CA Descriptor** defines the format for the CA descriptor for MDCF. The first fields in the MDCF CA descriptor are specified in ISO/IEC 13818-1, while the fields after the CA_PID field are MDCF specific. The MDCF CA descriptor is embedded into the PMT of each program in an MDCF that is accessible through OMA DRM.

Table 6 – MDCF CA Descriptor

Syntax	No. of bits	Mnemonic
MDCF_CA_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
reserved	3	bslbf

CA_PID
if (descriptor_length>4) {
for (i=0; I<N; i++){
CA_descriptor_message()
}
}
}
	13	uimsbf

descriptor_tag, MPEG has defined the value of 9 for the CA-descriptor.

descriptor_length, the number of bytes in this descriptor immediately following the descriptor_length field,.

CA_system_ID, the value of this field in the MDCF CA descriptor is 0x00000004 as assigned by DVB; see http://www.dvb.org/products_registration/dvb_identifiers/index.xml#

CA_PID, the PID on which the sections with MDCF ECMs can be found

CA_descriptor_message(), see section 5.3.1.

5.3.1 CA descriptor message

The CA descriptor message is a generic mechanism to extend MDCF CA descriptors. It consists of an identifier of the message, followed by the length of the message and message data bytes; see **Table 7 – CA_descriptor_message**. An MDCF CA descriptor may contain zero or more CA descriptor messages.

Table 7 – CA_descriptor_message

Syntax	No. of bits	Mnemonic
CA_descriptor_message() {		
CA_descriptor_message_ID	8	uimsbf
message_data_length	8	uimsbf
for (i=0; i< message_data_lengh); i++){		
message_data_byte	8	bslbf
}		
}		

CA_descriptor_message_ID, the CA_descriptor_message_ID field identifies a specific CA_descriptor_message; the values in the inclusive range 0x00 – 0x1F are reserved for future use by OMA; the other values are available for private use by applications using MDCF and are assigned via OMNA registration. See also **Table 8 – CA_descriptor_message_ID values**.

Table 8 – CA_descriptor_message_ID values

Value	Description
0x00 – 0x1F	Reserved for future use by OMA
0x20 – 0xFF	Available for private use by MDCF applications; values are assigned via OMNA registration

message_data_length, the message_data_length field specifies the number of message_data_ bytes immediately following this field.

message_data_byte, the coding and semantics of message_data_bytes depend on the encoded value of the preceding CA_descriptor_message_ID field. For CA_descriptor_message_ID values in the inclusive range 0x00 – 0x1F, the message_data_bytes are reserved for future use by OMA. For CA_descriptor_message_ID values in the inclusive range 0x20 – 0xFF, the coding and semantics of message_data_bytes is specific to the MDCF application signalled by the value of the preceding CA_descriptor_message _ID field.

5.4 Usage of Private Sections in MDCF

In the following sub-sections ECM messages for usage in MDCF are specified. [ISO/IEC 13818-1] requires ECMs to be formatted as so-called private sections. A stream of such private sections may be used to transport a number of logical datastructures, called tables, with a certain frequency. Each ECM contributes to a certain table, as identified by the table_id field in the ECM. Each table may require a number of subsequent ECM's. A table may be repeated at appropriate intervals. **Table 9 – MDCF ECM tables** indicates the ECM-tables defined for the MDCF, with an example of a repetition rate that may be appropriate. Note (1) that two values are assigned for STKM ECMs, so as to allow for switching Traffic Encryption Keys in a manner that can be conveniently identified by a table-id filter, and (2) that the Table_id values assigned in **Table 9** uniquely identify the associated sections within the TS packets identified by the value of CA_PID in the MDCF CA descriptor.

Table 9 – MDCF ECM tables

Table_id	Table description	Example of a repetition rate (Hz)
0x80	Short Term Key Message ECM	10
0x81	Short Term Key Message ECM	10
0x82	ContentID ECM	10
0x83	Rights URL ECM	0.1
0x84	Textual headers ECM	0.1
0x85	Extended headers ECM	0.1

5.4.1 Short Term Key Message ECM

The OMA DRM Key management used in the MDCF is based on the Key Management of [BCAST] and extended for use in the context of MPEG2-TS. The Short-Term-Key-Message (STKM) defined in [BCAST] extended for the traffic_protection_protocol of the MDCF is formatted as in **Table 12 – MDCF_STKM_section** and **Table 13 – Format of STKM for MDCF**.

In the STKM syntax and semantics, so-called constants are used, whereby each constant is associated with a certain value of a parameter, For example, the odd_even_flag can take value 0 and 1; the constant TKM_FLAG_FALSE corresponds with the value 0, while the constant TKM_FLAG_TRUE corresponds with the value 1. Two types of constants are used; constants defined by [BCAST] are depicted in **Table 10 – BCAST constants**, and constants that are specific for the STKM for MDCF, are depicted in **Table 11 – MDCF constants** .

Table 10 – BCAST constants

Name	Value of associated parameter indicated by this constant
TKM_ALGO_IPSEC	0
TKM_ALGO_SRTP	1
TKM_ALGO_ISMACRYP	2
TKM_ALGO_DCF	3
TKM_FLAG_FALSE	0
TKM_FLAG_TRUE	1

Table 11 – MDCF constants

Name	Value of associated parameter indicated by this constant
TKM_ALGO_MPEG2_TS_CRYP	10
TKM_FLAG_EVEN	0
TKM_FLAG_ODD	1

An ECM that contributes to the STKM-table MUST be formatted as depicted in **Table 12 – MDCF_STKM_section**.

Table 12 – MDCF_STKM_section

Syntax	No. of bits	Mnemonic	Value
MDCF_STKM_section() {			
table_id	8	uimsbf	0x80
section_syntax_indicator	1	bslbf	0
OMA_reserved	1	bslbf	1
MPEG2_reserved	2	bslbf	
section_length	12	uimsbf	
short_term_key_message()			
}			

table_id, identifies the section as STKM-ECM. See **Table 9 – MDCF ECM tables**.

section_syntax_indicator, set to 0 to signal the use of the short section header ([ISO/IEC 13818-1] section 2.4.4.11).

OMA_reserved, bit reserved for future use by OMA.

MPEG2_reserved, bits reserved by [ISO/IEC 13818-1].

section_length, the number of bytes that follow the section_length field up to the end of the section.

short_term_key_message, the Short-Term-Key-Message for MDCF defined in **Table 13 – Format of STKM for MDCF**.

Table 13 – Format of STKM for MDCF

Short_Term_Key_Message_Description	Length	Type
short_term_key_message() {		
selectors_and_flags {		
protocol_version	4	uimsbf
protection_after_reception	2	uimsbf
reserved_for_future_use	1	bslbf
access_criteria_flag	1	uimsbf
traffic_protection_protocol	3	uimsbf
traffic_authentication_flag	1	uimsbf
next_traffic_key_flag	1	uimsbf
timestamp_flag	1	uimsbf
program_flag	1	uimsbf
service_flag	1	uimsbf
}		
if (traffic_protection_protocol == TKM_ALGO_MPEG2_TS_CRYPT) {		
content_key_index	4	uimsbf
odd_even_flag	1	bslbf
cipher mode	3	uimsbf
reserved_for_future_use	8	bslbf
if (cipher_mode == 0x1 && next_traffic_key_flag == KSM_FLAG_TRUE) {		
initial_vector_length	8	uimsbf
next_initial_vector	8 * initial_vector_length	bslbf
}		
}		
encrypted_traffic_key_material_length	8	uimsbf
encrypted_traffic_key_material	8*encrypted_traffic_key_material_length	bslbf
if (next_traffic_key_flag == TKM_FLAG_TRUE) {		
next_encrypted_traffic_key_material	8*encrypted_traffic_key_material_length	bslbf
}		
reserved_for_future_use	4	bslbf
traffic_key_lifetime	4	uimsbf
if (timestamp_flag == TKM_FLAG_TRUE) {		
timestamp	40	mjdutc
}		
if (access_criteria_flag == TKM_FLAG_TRUE) {		
reserved_for_future_use	8	bslbf
number_of_access_criteria_descriptors	8	uimsbf
access_criteria_descriptor_loop() {		
access_criteria_descriptor()		
}		
}		
if (program_flag == TKM_FLAG_TRUE) {		
program_selectors_and_flags {		
reserved_for_future_use	7	bslbf
permissions_flag	1	uimsbf
}		
if (permissions_flag == TKM_FLAG_TRUE) {		
permissions_category	8	uimsbf
}		
if (service_flag == TKM_FLAG_TRUE) {		
encrypted_PEK	128	bslbf
}		
program_CID_extension	32	uimsbf
program_MAC	96	bslbf
}		

if (service_flag == TKM_FLAG_TRUE) {		
service_CID_extension	32	uimsbf
service_MAC	96	bslbf
}		
}		

The semantics of many fields in the data structure in **Table 13 – Format of STKM for MDCF** are defined in the Service and Content Protection TS (SvcCntProtection) in [BCAST].

protocol_version –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

protection_after_reception –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

access_criteria_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

traffic_protection_protocol –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

traffic_authentication_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

next_traffic_key_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

timestamp_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

program_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

service_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

content_key_index –identifies the type of cipher and Traffic Encryption Key (TEK) used to scramble the MDCF. **Table 14 – Content_key_index options** defines the options.

Table 14 – Content_key_index options

Content_key_index value	Description	Comment
0x0	DVB-CSA key of 64 bit length.	
0x1	DES key of 56 bit length.	
0x2	3DES key of 168 bit length.	
0x3	3DES key of 112 bit length.	
0x4	3DES key of 56 bit length.	
0x5	AES key of 128 bit length.	
0x6	M2 key of 64 bit length.	Multi 2 for Japan.
0x7 – 0xF	Reserved for future use.	

odd_even_flag – indicates if the odd or even register is used for the TEK. The odd_even_flag can take the following values (see also **Table 11 – MDCF constants**):

KSM_FLAG_ODD = the receiving Device should insert the TEK into the odd register of the descrambler.
KSM_FLAG_EVEN = the receiving Device should insert the TEK into the even register of the descrambler.

cipher_mode – indicates the mode in which the cipher indicated by the field content_key_index is used, assuming the traffic_protection_protocol field has the value KSM_ALGO_MPEG2_TS_CRYPT. **Table 15 – cipher_mode options** defines the options.

Table 15 – cipher_mode options

cipher_mode value	Description	Comment
0x0	ECB	For DES, 3DES or AES.
0x1	CBC	For DES, 3DES or AES.
0x2	CSA	For DVB CSA.
0x3 – 0x7	Reserved for future use.	-

initial_vector_length – is the length in bytes of the Initial Vector.

next_initial_vector – is the Initial Vector for the next traffic_key_lifetime period that is used when the DES, 3DES or AES ciphers are used in CBC mode, as is indicated by the cipher_mode field. The value shall be a random number matching the size indicated by the field content_key_index (see **Table 13 – Format of STKM for MDCF**).

encrypted_traffic_key_material_length –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

encrypted_traffic_key_material –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

next_encrypted_traffic_key_material –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

traffic_key_lifetime –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

timestamp –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

number_of_access_criteria_descriptors –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

access_criteria_descriptor() –see the STKM common attributes in section 7 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

permissions_flag –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

permissions_category –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

encrypted_PEK –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

program_CID_extension –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

program_MAC –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

service_CID_extension –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

service_MAC –see the coding and semantics of STKM attributes in section 5.5.1 of the Service and Content Protection TS (SvcCntProtection) in [BCAST]

As defined in [BCAST], the encrypted_traffic_key_material is the key material currently used for encryption and optional authentication of the MDCF, encrypted using AES-128-CBC, with fixed IV 0, and with 0 padding in the last block, if needed. In the case of a MDCF, the encrypted_traffic_key_material is decrypted with keys from an RO (See section 5.5.1). After decryption (and discarding any padding), the Traffic Encryption Key (TEK) is obtained in a way that depends on the type of

TEK used. This is indicated by the content_key_index_value. **Table 16: Supported Chiphers** defines how to derive the TEK from the traffic_key_material.

Table 16: Supported Chiphers

Content_key_index value	TEK Description	Traffic key material	Obtain decrypted traffic key material
0x0	DVB-CSA key of 64 bit length.	64 bits input are padded to 128 bits according to [FIPS PUB 197 (2001)]	First 8 bytes
0x1	DES key of 56 bit length.	56 bits input are padded to 128 bits according to [FIPS PUB 197 (2001)]	First 7 bytes
0x2	3DES key of 168 bit length.	168 bits input are padded to 256 bits according to [FIPS PUB 197 (2001)]	First 21 bytes
0x3	3DES key of 112 bit length.	112 bits input are padded to 128 bits according to [FIPS PUB 197 (2001)]	First 14 bytes
0x4	3DES key of 56 bit length	56 bits input are padded to 128 bits according to [FIPS PUB 197 (2001)]	First 7 bytes
0x5	AES key of 128 bit length	128 bits input are according to [FIPS PUB 197 (2001)]	First 16 bytes
0x6	M2 key of 64 bit length	64 bits input are padded to 128 bits according to [FIPS PUB 197 (2001)]	First 8 bytes
0x7-0xF	Reserved for future use		

5.4.2 Content ID ECM

The Content ID ECM contains part of the information needed by a DRM Agent to request a Rights Object for the MDCF. See section 5.5.1. The format of a Content ID ECM is provided in **Table 17 – MDCF ContentID section**...

Table 17 – MDCF ContentID section

Syntax	No. of bits	Mnemonic	Value
MDCF_ContentID_section() {			
table_id	8	uimsbf	0x82
section_syntax_indicator	1	bslbf	0
OMA_reserved	1	bslbf	
MPEG2_reserved	2	bslbf	
section_length	12	uimsbf	
for (i=0; i<N; i++){			
CID_message()			
}			
}			

table_id, identifies the section as ContentID-ECM. See **Table 9 – MDCF ECM tables**.

section_syntax_indicator, set to 0 to signal the use of the short section header ([ISO/IEC 13818-1], section 2.4.4.11).

OMA_reserved, bit reserved for future use by OMA.

MPEG2_reserved, bits reserved by [ISO/IEC 13818-1].

section_length, the number of bytes that follow the section_length field up to the end of the section.

CID_message(), a message containing identification information on the content as needed to request a Rights Object. The general structure of a CID message is provided in section 5.4.2.1. To retrieve a Rights Object, the MDCF_ContentID_section MUST contain a CID_message with the baseCID and a CID_message with the socID. It is recommended that the contained CID messages are smaller than (184-3 (header)-1(section offset indicator) = 180 bytes so that the section can fit into one TS packet.

5.4.2.1 CID message

The CID message is a mechanism to include content identification information in a Content ID ECM. Each message consists of a message identifier, followed by the length of the message and message data bytes; see **Table 18 – CID_message**.

Table 18 – CID_message

Syntax	No. of bits	Mnemonic
CID_message() {		
CID_message_ID	8	uimsbf
message_data_length	8	uimsbf
for (i=0; i< message_data_lengh); i++){		
message_data_byte	8	bslbf
}		
}		

CID_message_ID, the CID_message_ID field identifies a specific CID message; the values in the inclusive range 0x00 – 0x1F are for use by OMA; the other values are available for private use by applications using MDCF and are assigned via OMNA registration. See also **Table 19 – CID_message_ID values**.

Table 19 – CID_message_ID values

Value	Description
0x00	baseCID
0x01	socID
0x002– 0x1F	Reserved for future use by OMA
0x20 – 0xFF	Available for private use by MDCF applications; values are assigned via OMNA registration

message_data_length, the message_data_length field specifies the number of message_data_ bytes immediately following this field.

message_data_byte, the coding and semantics of message_data_bytes depend on the encoded value of the preceding CID_message_ID field. For CID_message_ID values in the inclusive range 0x00 – 0x1F, the message_data_bytes are defined by OMA; see the following sub-sections. For CID_message_ID values in the inclusive range 0x20 – 0xFF, the coding and semantics of message_data_bytes is specific to the MDCF application, signalled by the value of the preceding CID_message_ID field.

5.4.2.1.1 baseCID message

A baseCID message is identified by the CID_message_ID value assigned to baseCID messages in **Table 19 – CID_message_ID values**. In a baseCID message, the message_data_bytes are **baseCID_bytes**, representing the service base content identifier for this content. Each MDCF_ContentID_section MUST contain a baseCID message.

5.4.2.1.2 socID message

A socID message is identified by the CID_message_ID value assigned to socID messages in **Table 19 – CID_message_ID values**. In a socID message, the message_data_bytes are **socID_bytes**, representing the service operation center identifier for this content. Each MDCF_ContentID_section MUST contain a socID message.

NOTE The socID is set to 'b' for OMA BCAST content in the OMA BCAST specification [OMA-BCAST]. In DVB-SPP, the socID is variable and can be obtained from the ESG. Even if the SocID is set to a fixed value in a specification, the socID message SHALL always be present in each MDCF_ContentID_section.

5.4.3 Rights URL ECM

The Rights URL ECM contains part of the information needed by a DRM Agent to request a Rights Object for the MDCF. See section 5.5.1 and **Table 20 – MDCF Rights URL section**.

Table 20 – MDCF Rights URL section

Syntax	No. of bits	Mnemonic	Value
MDCF_Rights_URL_section() {			
table_id	8	uimsbf	0x83
section_syntax_indicator	1	bslbf	0
OMA_reserved	1	bslbf	
MPEG2_reserved	2	bslbf	
section_length	12	uimsbf	
for (i=0; i<N; i++){			
Rights_URL_message()			
}			
}			

table_id, identifies the section as RI URL ECM. See **Table 9 – MDCF ECM tables**.

section_syntax_indicator, set to 0 to signal the use of the short section header ([ISO/IEC 13818-1] section 2.4.4.11).

OMA_reserved, bit reserved for future use by OMA.

MPEG2_reserved, bits reserved by [ISO/IEC 13818-1].

section_length, the number of bytes that follow the section_length field up to the end of the section.

Rights_URL_message(), one or more messages containing information on Rights Issuer URLs, as needed to request a Rights Object. The general structure of a RI_URL message is provided in section 5.4.3.1. It is recommended to have Rights URL ECMs smaller than (184-3 (header)-1(section offset indicator) = 180 bytes so that the section can fit into one TS packet.

5.4.3.1 Rights_URL message

The Rights_URL_message is a mechanism to include URLs in a Rights URL ECM. Each message consists of a message identifier, followed by the length of the message and message data bytes; see **Table 21 – Rights_URL_message**.

Table 21 – Rights_URL_message

Syntax	No. of bits	Mnemonic
Rights_URL_message() {		
rights_URL_message_ID	8	uimsbf
message_data_length	8	uimsbf
for (i=0; i< message_data_lengh); i++){		
message_data_byte	8	bslbf
}		
}		

rights_URL_message_ID, the rights_URL_message_ID field identifies a specific rights_URL_message; the values in the inclusive range 0x00 – 0x1F are reserved for future use by OMA; the other values are available for private use by applications using MDCF and are assigned via OMNA registration. See **Table 22 – Rights_URL_message_ID values**.

Table 22 – Rights_URL_message_ID values

Value	Description
0x00	RightsIssuerURL
0x01	SilentRightsURL
0x02	PreviewRightsURL
0x03 – 0x1F	Reserved for future use by OMA
0x20 – 0xFF	Available for private use by MDCF applications; values are assigned via OMNA registration

message_data_length, the message_data_length field specifies the number of message_data_ bytes immediately following this field.

message_data_byte, the coding and semantics of message_data_bytes depend on the encoded value of the preceding rights_URL_message_ID field. For rights_URL_message_ID values in the inclusive range 0x00 – 0x1F the message_data_bytes are defined by OMA; see the following sub-sections.. For rights_URL_message_ID values in the inclusive range 0x20 – 0xFF, the coding and semantics of message_data_bytes is specific to the MDCF application, signalled by the value of the preceding rights_URL_message _ID field.

5.4.3.1.1 RightsIssuerURL message

A RightsIssuerURL message is identified by the rights_URL_message_ID field value assigned to RightsIssuerURL messages in **Table 22 – Rights_URL_message_ID values**. In a RightsIssuerURL message, the message_data_bytes are **RightsIssuerURL_bytes**, representing a Rights Issuer URL for this content as specified by [DRM-DCF-v2.1].

5.4.3.1.2 SilentRightsURL message

A SilentRightsURL message is identified by the rights_URL_message_ID field value assigned to SilentRightsURL messages in **Table 22 – Rights_URL_message_ID values**. In a SilentRightsURL message, the message_data_bytes are **SilentRightsURL_bytes**, representing a Silent Rights URL for this content as specified by [DRM-DCF-v2.1].

5.4.3.1.3 PreviewRightsURL message

A PreviewRightsURL message is identified by the rights_URL_message_ID field value assigned to PreviewRightsURL messages in **Table 22 – Rights_URL_message_ID values**. In a PreviewRightsURL message, the message_data_bytes are **PreviewRightsURL_bytes**, representing a Preview Rights URL for this content as specified by [DRM-DCF-v2.1].

5.4.4 Textual headers ECM

Table 23 – MDCF Textual header section

Syntax	No. of bits	Mnemonic	Value
MDCF_textual_header_section() {			
table_id	8	uimsbf	0x84
section_syntax_indicator	1	bslbf	1
OMA_reserved	1	bslbf	
MPEG2_reserved	2	bslbf	
private_section_length	12	uimsbf	
table_id_extension	16	uimsbf	
MPEG2_reserved	2	bslbf	
version_number	5	uimsbf	
current_next_indicator	1	bslbf	
section_number	8	uimsbf	
last_section_number	8	uimsbf	
for (i = 0; i < private_section_length-9;.i) {			
textual_header_byte	8	bslbf	
}			
CRC_32	32	rpchof	
}			
}			

table_id, identifies the section as Textual Header ECM. See **Table 9 – MDCF ECM tables**.

section_syntax_indicator, set to 1 to signal the use of the long section header ([ISO/IEC 13818-1] section 2.4.4.11).

OMA_reserved, bit reserved for future use by OMA.

MPEG2_reserved, bits reserved by [ISO/IEC 13818-1].

section_length, the number of bytes that follow the section_length field up to the end of the section.

version_number, the version_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

section_number, the section_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

last_section_number, the last_section_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

textual_header_byte, the textual header information. Each textual header section will carry one textual header as specified in [DRM-DCF-v2.1].

5.4.5 Extended headers ECM

Table 24 – MDCF Extended header section

Syntax	No. of bits	Mnemonic	Value
MDCF_extended_header_section() {			
table_id	8	uimsbf	0x85
section_syntax_indicator	1	bslbf	1
OMA_reserved	1	bslbf	
MPEG2_reserved	2	bslbf	
private_section_length	12	uimsbf	
table_id_extension	16	uimsbf	
MPEG2_reserved	2	bslbf	
version_number	5	uimsbf	
current_next_indicator	1	bslbf	
section_number	8	uimsbf	
last_section_number	8	uimsbf	
for (i = 0; i < private_section_length-9;.i) {			
extended_header_byte	8	bslbf	
}			
CRC_32	32	rpchof	
}			
}			

table_id, identifies the section as Extended Headers ECM. See **Table 9 – MDCF ECM tables**.

section_syntax_indicator, set to 1 to signal the use of the long section header ([ISO/IEC 13818-1] section 2.4.4.11).

OMA_reserved, bit reserved for future use by OMA.

MPEG2_reserved, bits reserved by [ISO/IEC 13818-1].

section_length, the number of bytes that follow the section_length field up to the end of the section.

version_number, the version_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

section_number, the section_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

last_section_number, the last_section_number of the table ([ISO/IEC 13818-1] section 2.4.4.11).

extended_header_byte, the extended header information. Each extended header section will carry one extended header as specified in [DRM-DCF-v2.1].

5.5 Accessing the MDCF

To access the MDCF, the OMA DRM Terminal must perform the following steps:

1. Parse the MPEG-2 transport stream to identify whether it is MDCF compliant by means of the MDCF CA descriptor.
2. If the MPEG-2 transport stream is MDCF compliant, then parse the MDCF to retrieve the MDCF related ECM's for each program.
3. If the Rights Object for accessing the MDCF is not available, then request this Rights Objects from the Rights Issuer.
4. Process the Rights Objects and ECM's to descramble the MPEG-2 transport stream.

The first two steps in this process involve retrieving and processing the PAT, PMT, MDCF CA Descriptor and retrieving the MDCF related ECMs . This process is specified in [ISO/IEC 13818-1] (also see section 5.1). The other steps are described in this section.

As specified in [BCAST], the program_flag and the service_flag in the STKM-ECM signal which of the possible business models the Rights Issuer provides for the stream of which the MDCF is a portion. These flags also determine the processing that is required to access the stream.

If the program_flag in the short_term_key_message () in the STKM ECM (see **Table 13 – Format of STKM for MDCF**) is set to TRUE, then access to this particular program is provided on a per-program basis. The OMA DRM terminal may be able to retrieve a Rights Object for this specific program. This process is specified in section 5.5.1.1

If the service_flag in the short_term_key_message () in the STKM ECM (see **Table 13 – Format of STKM for MDCF**) is set to TRUE, then access to this particular program is provided as part of a service. The OMA DRM terminal may be able to retrieve a Rights Object for the service, which provides access to this program and all other programs that belong to the service. This process is specified in section 5.5.1.2..

Please note that both flags may be TRUE, indicating that the Rights Issuer supports both business models.

5.5.1 Requesting a Rights Object for content in an MDCF

5.5.1.1 Requesting a Rights Object for a program

The OMA DRM Terminal may use the Silent Header URL that may be transported in Textual Headers ECMs or in a RightsURL ECM to initiate the request of a Rights Object. To request a ProgramRO for (part of) a MDCF the OMA DRM Terminal MUST first append the Program_CID to the Silent Header URL.

A program contained in the MDCF is uniquely identified by the,Program_CID, see [BCAST]. The Program_CID consists of the socID, followed by the baseCID, followed by the program_CID extension, and is constructed as follows:

Program_CID = “cid:” || socID || "#P" || baseCID || "@" || HEX(program_CID_extension)

The socCID is a character string taken from the socID message in the ContentID ECM (see Section 5.4.2), the baseCID is a character string taken from the baseCID message in the ContentID ECM (see Section 5.4.2), program_CID_extension is taken from an STKM belonging to the Program and HEX() is a function defined in [OMA-BCAST]. The resulting Program_CID shall be a globally unique URI and MUST comply to the requirements for ContentIDs specified in [DRM-DCF-v2.1].

Alternatively the OMA DRM terminal may use a URL contained in the Rights URL ECM (see section 5.4.3) to initiate a browsing session with the Rights Issuer to purchase a Rights Object as defined in [OMA DRM]

5.5.1.2 Requesting a Rights Object for a service

The OMA DRM Terminal may use the Silent Header URL that may be transported in Textual Headers ECMs or in a RightsURL ECM to initiate the request of a Rights Object. To request a serviceRO for (part of) an MDCF the OMA DRM Terminal MUST first append the Service_CID to the Silent Header URL.

The service contained in the MDCF is uniquely identified by the,Service_CID see [BCAST]. The Service_CID consists of the socID, followed by the baseCID, followed by the service_CID extension, and is constructed as follows:

Service_CID ::= “cid:” || socID || "#S" || baseCID || "@" || HEX(service_CID_extension)

The socCID is taken from the socID message in the ContentID ECM (see Section 5.4.2), the baseCID is taken from the baseCID message in the ContentID ECM (see Section 5.4.2), service_CID_extension is taken from an STKM belonging to the Service and HEX() is a function defined in [OMA-BCAST]. The resulting Service_CID shall be a globally unique URI and MUST comply to the requirements for ContentIDs specified in [DRM-DCF-v2.1].

Alternatively the OMA DRM terminal may use a URL contained in the Rights URL ECM (see section 5.4.3) to initiate a browsing session with the Rights Issuer to purchase a Rights Object as defined in [OMA DRM]

5.5.2 Deriving the Traffic Encryption Key

From the RO associated to the (selected program of the) MDCF, the DRM Agent must retrieve the CEK and if available the MAC-key. If present in the RO (see [BCAST]), the MAC key must be used to authenticate the STKM-ECMs as specified in [BCAST].

Using the CEK, the OMA DRM Agent is able to decrypt the Traffic Encryption Key (TEK), from the information in the STKM-ECM. As specified in [BCAST], the program_flag and the service_flag in the STKM-ECM signal which of the possible business models is used for the stream from which the MDCF is a portion. These flags determine the required processing to retrieve the TEK from the **encrypted_traffic_key_material**. The required processing is described in **Table 25 – Key handling**.

Table 25 – Key handling

service_flag value	program_flag value	Processing
FALSE	FALSE	N/A
TRUE	FALSE	Use CEK from the RO to decrypt the encrypted_traffic_key_material.
FALSE	TRUE	Use CEK from the RO to decrypt the encrypted_traffic_key_material.
TRUE	TRUE	Use CEK from the RO to decrypt the encrypted_PEK. Use PEK to decrypt the encrypted_traffic_key_material.

The resulting TEK can be used to decrypt the encrypted packets of the MDCF.

As indicated in **Table 15 – cipher_mode options**, several ciphers can be used to encrypt the MDCF. Irrespective of the cipher used the odd_even_flag shall be in line with the bits of the transport_scrambling_control bits (see subclause 5.2.1) or the pes_scrambling_control bits (see subclause 0). Using **Table 3 – Descrambling possibility matrix**, the device will decide how to handle encrypted_traffic_key_material and next_encrypted_traffic_key_material in odd and/or even registers. The list below specifies some cipher dependent functionality.

- All ciphers

  The odd_even_flag in the STKM-ECM will indicate if the encrypted_traffic_key_material field will contain a content key for the odd or even register. The use of odd and even is explained in subclauses 5.2.1 and 0. Shortly before the traffic key lifetime expires, the STKM-ECM will include the field next_encrypted_traffic_key_material to allow the device to prepare for the key change.

  NOTE Although odd/even functionality is in practice not used for other ciphers than DVB-CSA, this document supports this functionality for all ciphers.

- DES, 3DES, AES

  DES, 3DES and AES can be used in ECB or CBC mode. In ECB mode, there is no IV. In CBC mode, the STKM-ECM shall include an IV. Switching from one key and IV to another is done when the traffic key lifetime expires. Shortly before the traffic key lifetime expires, the STKM-ECM will include the field next_encrypted_traffic_key_material and the field next_initialisation_vector to allow the device to prepare for the key change. The termination block handling is specified in ANSI/SCTE 52:2003 and shall be applied when the last content block to be encrypted is smaller than the cipher block size.

5.6 Exchanging an MDCF between OMA DRM terminals

An MDCF may be exchanged between OMA DRM Terminals. As described in previous sections, an MDCF must be parsed to determine its contents and a Rights Object must be requested to access the MDCF. The Rights Object that is received may be a domain Rights Object, which may be usable on other OMA DRM Terminals. This section specifies means that enable efficient exchange between OMA DRM terminals of the information extracted from the MDCF and the Rights Objects retrieved from an RI for the MDCF. This is achieved by creating (P)DCF files for the MDCF file. In this way not all OMA DRM Terminals that want to gain access the MDCF are required to perform all of the processing described in previous sections.

Before creating (P)DCF's for an MDCF, an OMA DRM Terminal may split-up a single large MDCF into a number of separate MDCF files, for example to locate each program in the MDCF into a separate file.

An OMA DRM Terminal MAY associate a (P)DCF with an MDCF in one of two ways. When using a DCF, the MDCF is either embedded into the DCF as a Content Object, or the MDCF file is referenced from the DCF using a Content Location Header. A PDCF may be used in applications that require synchronization of other data with the data in the MDCF. These other data are e.g. timed meta-data that is received in parallel to the MDCF or generated during recording. In this case the MDCF is an ISO File Format hint track in the PDCF. The standard mechanisms of the ISO File Format allow an embedded track in the PDCF or referencing the MDCF file.

When using a DCF, it is RECOMMENDED that an OMA DRM Terminal creates a separate DCF-file for each program in the MDCF, irrespective whether the part of the MDCF that contains that program is embedded into the DCF-file itself as a Content Object or stored as a separate file that is referenced from the DCF. An OMA DRM Terminal MAY also create a multi-part DCF for an MDCF, such that each part of the multi-part DCF is associated with one program in the MDCF.

It is also RECOMMENDED that all valid Rights Objects that were received for that program of the MDCF are embedded into the Mutable DRM Info Box of the (P)DCF-file.

5.6.1 ContentID field in the DCF header

The ContentID field can either be filled with a Service_CID or a Program_CID, as specified in previous sections. If available, the ContentID field SHOULD be filled with a Program_CID.

5.6.2 Content-Location header in the DCF header

The Content-Location, indicating which part of the MDCF to access for referenced content, is specified as follows.

```
ContentLocation = "Content-Location" ":" content-uri
content-uri = token | ( token || "#" || start_byte || end )
end = ("-" || end_byte) | ( "+" || n_bytes)
start_byte = *digit
end_byte = *digit | "end"
n_bytes = *digit
```

Where

token MUST be a file name, relative to the location of the DCF, or, if token is the empty string, the Content-Location header refers to the MDCF that is contained in the DCF file itself,

start_byte is the index of the byte in the MDCF to access the referenced content (program or service); the index of the first byte of the MDCF has value 0,

end_byte is either the string “end”, indicating the last byte of the file, or a number consisting of one or more digits, indicating the index of a byte in the MDCF after which no more bytes of the referenced content (program or service) are contained in the MDCF

n_bytes is the number of bytes in the MDCF immediately following start_byte, after which no more bytes of the referenced content (program or service) are contained in the MDCF.

5.6.3 EncryptionMethod field in the DCF header

In this specification, we specify a new value for Algorithm-id for use in the EncryptionMethod field in a DCF header.

Algorithm-id: MPEG-2_transport_stream_encryption

Value: for MPEG-2_transport_stream_encryption, the value 0x04 is assigned by OMNA

Semantics: the content is an MPEG Transport Stream which is protected as specified in Section 5.4.1

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-TS-SCE_MDCF-V1_0-20110705-A	05 Jul 2011	Status changed to Approved by TP: OMA-TP-2011-0233-INP_SCE_V1_0_ERP_for_Final_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for Client

Item	Function	Reference	Requirement
SCE-MDCF-DRMAGENT-C-001-O	Descrambling of TS	5.2	
SCE-MDCF-DRMAGENT-C-002-O	CA descriptor usage in MDCF	5.3	
SCE-MDCF-DRMAGENT-C-003-O	Usage of private sections in MDCF	5.4	
SCE-MDCF-DRMAGENT-C-004-O	Accessing the MDCF	5.5	
SCE-MDCF-DRMAGENT-C-005-O	Requesting an RO for content in an MDCF	5.5.1	
SCE-MDCF-DRMAGENT-C-006-O	Deriving the Traffic Encryption Key	5.5.2	
SCE-MDCF-DRMAGENT-C-007-O	Exchanging MDCF between OMA DRM terminals	5.6	