Content Protection for eXtended Media Specification

SD Memory Card Book Common Part

Intel Corporation

International Business Machines Corporation

Panasonic Corporation

Toshiba Corporation

Revision 0.85 Preliminary Release

September 27, 2010

This page is intentionally left blank.

Preface

Notice

Intellectual Property

Implementation of this specification requires a license from the 4C Entity, LLC.

Contact Information

Please address inquiries, feedback, and licensing requests to the 4C Entity, LLC:

- Licensing inquiries and requests should be addressed to 4C-Services@4CEntity.com.
- Feedback on this specification should be addressed to 4C-Services@4CEntity.com.

The URL for the 4C Entity, LLC web site is http://www.4CEntity.com.

This page is intentionally left blank.

Table of Contents

List of Figures

This page is intentionally left blank.

List of Tables

This page is intentionally left blank.

Chapter 1 Introduction

1. Introduction

1.1 Purpose and Scope

The *Content Protection for eXtended Media Specification* (CPXM) defines a robust and renewable method for protecting content stored on a number of physical media types. CPXM technology is an enhanced version of Content Protection Removable Media (CPRM) technology. The specification is organized into several “books”. The *Introduction and Common Cryptographic Elements* book provides a brief overview of CPXM, and defines cryptographic procedures that are common among its different uses. The *SD Memory Card Book* specifies additional details for using CPXM technology to protect content stored on the SD Memory Card, and on other implementations of protected storage with an interface and security system equivalent to that of the SD Memory Card. Note that such other implementations must not provide any external interface to the memory other than one that adheres to the protocols described in this specification.

The *SD Memory Card Book* consists of the following parts, under the general title *CPXM Specification SD Memory Card Book:*

- *Common Part,*
- *SD-Application Specific Parts*

This document is the *Common Part* of the *SD Memory Card Book*, and describes aspects of CPXM that are common to each SD-Application. Other *SD-Application Specific Parts* of the *SD Memory Card Book* describe additional details specific to each SD-Application. When there is a discrepancy between an application-independent book and an application-specific book then the application-specific book takes precedence.

The use of this specification and access to the intellectual property and cryptographic materials required to implement it will be the subject of a license. A license authority referred to as the 4C Entity, LLC is responsible for establishing and administering the content protection system based in part on this specification.

This document is an abridged version and for evaluation purpose only.

1.2 Document Organization

This document is organized as follows:

- Chapter 1 provides an introduction.
- Chapter 2 lists abbreviations and acronyms used in this document.
- Chapter 3 describes the common CPXM protection mechanisms that are used to protect each SD-Application’s content stored on SD Memory Card media.

1.3 References

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

4C Entity, LLC, *CPXM License Agreement* (Unpublished)

4C Entity, LLC, *CPXM Specification: Introduction and Common Cryptographic Elements*

4C Entity, LLC, *Content Protection System Architecture White Paper, Revision 0.81*

SD Group, *SD Memory Card Specifications, Part 3: Security Specification, Version 3.10*

1.4 Future Directions

This document describes aspects of CPXM that are common to each SD-Application. In future revisions, additional common CPXM elements may also be described.

1.5 Notation

Except where specifically noted otherwise, this document uses the same notations and conventions for numerical values, operations, and bit/byte ordering as described in the *Introduction and Common Cryptographic Elements* book of this specification.

Chapter 2
Abbreviations and Acronyms

2. Alphabetical List of Abbreviations and Acronyms

The following abbreviations and acronyms are used in this document:

4C	4 Companies (IBM, Intel, Panasonic, and Toshiba)
K_{app}	Application Key
AKE	Authentication and Key Exchange
K_{auth}	Authentication Key
CBC	Cipher Block Chaining
CCI	Copy Control Information
CPRM	Content Protection for Recordable Media
CPXM	Content Protection for eXtended Media
ECB	Electronic Codebook
FAT	File Allocation Table
ID	Identifier
KCD	Key Conversion Data
LLC	Limited Liability Company
lsb	Least Significant Bit
ID_{media}	Media Identifier
K_m^0	Media Key or Media Key Base
K_m^{-1}	Media Key Precursor
MKB	Media Key Block
K_{mu}	Media Unique Key
msb	Most Significant Bit
PC	Personal Computer
SD	Secure Digital
K_t	Title Key
TKURE	Title Key and Usage Rule Entry
TBD	To Be Determined
UR	Usage Rules
XOR	Exclusive-OR

This page is intentionally left blank.

Chapter 3 Common CPXM Elements for the SD Memory Card

3. Common CPXM Elements for the SD Memory Card

3.1 Introduction

This chapter specifies details for using elements of CPXM technology that are common to the protection of each SD-Application content stored on SD Memory Card media. The formats for each SD-Application and the SD Memory Card are licensable from the SD Association, which also publishes specifications describing them in detail (see the corresponding references in Section 1.3). This chapter assumes that the reader is familiar with these formats, as defined in their corresponding specifications.

CPXM has two device types: 1) Media Devices such as the SD Memory Card detailed in this specification, and 2) Host Devices which consists of the higher function device which uses the Media Device.

It is anticipated that CPXM technology may also be applied to other SD formats under future extensions to this specification, as authorized by the 4C Entity, LLC.

3.2 Host Devices

Each CPXM compliant Host Device (e.g. recording, playback, viewing, source, and destination device) for the SD Memory Card shall follow the protocols for accessing Host Devices described in this specification.

3.2.1 Device Key Set

Each Host Device is given a set of Host Device Key Set, consists of (a) a Device Node and (b) a set of 325 Device Keys (K_d) and the UV Descriptor (UV) associated with each Device Key, {Device Node, {K_{d_0}, UV_0}, {K_{d_1}, UV_1}, ..., {K_{d_324}, UV_{324}}}. A Host Device shall treat its Host Device Keys as Highly Confidential as defined in the CPXM License Agreement. The confidentiality of Device Node and UV Descriptors are vendor specific. There are different sets for each application group defined by the SD Association. The actual keys are provided by the 4C Entity, LLC, to adopters of CPXM and are used for processing the MKB to calculate the Media Key Precursor (K_m^{-1}) and the Media Key (K_m^0), as described in the *Introduction and Common Cryptographic Elements* book of this specification. The Host Device supporting installation of the MKB shall have two Device Key Sets; that is one for the Placeholder MKB, and the other one for the Fixed MKB or the Proprietary MKB, in order to perform MKB Update properly. See Section 3.3.1.2 for the definition of the types of MKB. The Host Device Key Set may be either unique per device, or shared by multiple devices. The CPXM License Agreement describes the details and requirements associated with these two alternatives.

3.2.2 Media Unique Key

A Host Device uses the Media Unique Key when encrypting a Title Key or decrypting an encrypted Title Key. A Media Unique Key is calculated with the Media Key Precursor (K_m^{-1}) and the Media Identifier using the AES One-way Function. A Media Unique Key (K_{mu}) shall be calculated as:

$$K_{mu} = AES\_G(K_m^{-1}, ID_{media}).$$

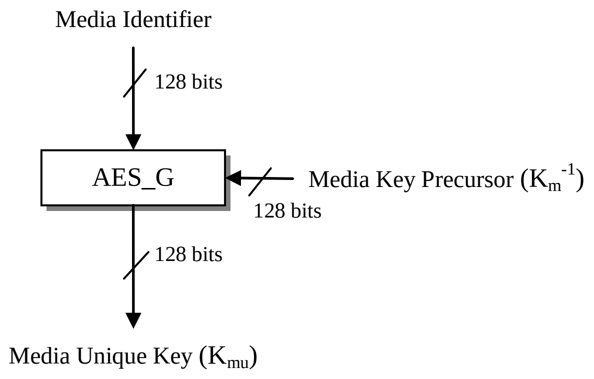

Figure 3-1 – Calculation of Media Unique Key

3.2.3 Media Key Block (MKB)

CPXM compliant Host Device shall retain the most recent Media Key Block (MKB) which it encounters and has verified in non-volatile memory storage, if the device supports MKB Update. A device shall protect the integrity of the MKB, so that MKB shall not be modified, altered, deleted or replaced in any form.

When persistently storing an MKB, the device shall have at least 128K (131,072) bytes of non-volatile memory for that purpose. The Host Device shall update the MKB in its storage if it receives a MKB from an SD Memory Card where the Version Number of the received MKB is greater than the Version Number of the MKB currently stored, and the received MKB is small enough to fit in the Host Device's non-volatile storage.

Note: If an MKB received by the Host Device with content from a remote server has a Version Number greater than the Version Number of the MKB stored on the SD Memory Card, the Host Device shall update the MKB in the SD Memory Card even if the MKB is larger than the capacity of the non-volatile storage in the Host Device.

3.3 SD Memory Card (Media Device) CPXM Components

This section describes the logical location and format of the CPXM components, when stored on the SD Memory Card. Figure 3-2 depicts the logical locations of CPXM Components on the SD Memory Card. These logical locations are used, unless explicitly noted otherwise in an application specific part of *CPXM Specification SD Memory Card Book*.

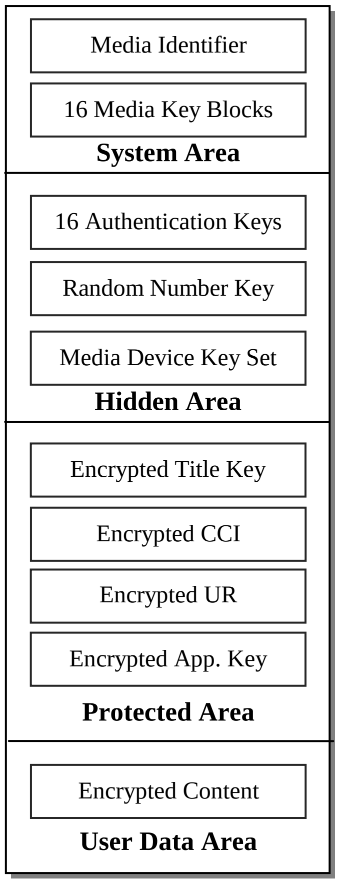

Figure 3-2 – SD Memory Card

- A Media Identifier (ID_{media}) and sixteen Media Key Blocks are pre-recorded in the System Area.
- One Media Device Key Set, sixteen Authentication Keys and a Random Number Key are pre-recorded in the Hidden Area.
- Encrypted Title Keys, Encrypted CCI (Copy Control Information), Encrypted UR (Usage Rules) and Encrypted Application Keys are recorded in the Protected Area. The actual number of Title Keys and associated CCI and/or Usage Rules is a result of how many titles have been recorded.
- Encrypted Content is recorded in the User Data Area

3.3.1 System Area

Each CPXM Compliant SD Memory Card shall contain a System Area, which is a read only area for the Host Device. The System Area contains the Media Identifier (ID_{media}) and the sixteen Media Key Blocks (MKBs).

3.3.1.1 Media Identifier

Each CPXM Compliant SD Memory Card shall contain a 128-bit Media Identifier (ID_{media}), which is placed in the System Area by the SD Memory Card manufacturer. The Media Identifier includes the Device Node (DN) of the Media Device Key Set. The Media Identifier logical format is shown in Table 3-1.

Table 3-1 – Media Identifier Format for SD Memory Card

Byte \ Bit	7	6	5	4	3	2	1	0
0	Manufacturer ID							
1	will be defined by 4C Entity, LLC							
2								
3								
4								
5								
6								
7								
8	Reserved (00_{16})							
9	Reserved (00_{16})							
10	Reserved (00_{16})							
11	Device Node Number of Media Device Key Set							
12								
13								
14								
15								

4C Entity, LLC assigns each licensee a unique 1-byte value as the Manufacture ID field, and 4C Entity, LLC assigns each media a unique Device Node Number of Media Device Key Set. The reserved field shall be filled with 00_{16} for this revision of the specification. The Device Node Number of the Media Device Key Set is used in the MKB Update process (Section 3.9, step 2).

3.3.1.2 Media Key Block (MKB)

Each CPXM Compliant SD Memory Card shall have sixteen MKB slots and these slots shall be occupied with sixteen distinct Media Key Blocks (MKBs) for CPXM in order to support multiple applications. There are three types of MKBs: Fixed, Proprietary, and Placeholder.

A Fixed MKB is an MKB for an application specified in the SD Card Association. A Proprietary MKB is an MKB for a proprietary purpose of an application that will be defined in the future. A Placeholder MKB is a special MKB that can be replaced with either a Fixed MKB or a Proprietary MKB after the shipment of the card. The Placeholder MKB shall not be used for protecting any content; that is the SD Memory Card shall not accept reads from or writes to the Protected Area using the Placeholder MKB.

Once the MKB slot is occupied with either a Fixed MKB or a Proprietary MKB, the SD Memory Card shall prohibit replacing an MKB with the MKB corresponding to another Application ID that is different from the one in the MKB slot. Also, the SD Memory Card shall not install an MKB with the same Application ID that already exists in another MKB slot.

Table 3-2 – Types of MKB and Application ID

Type of MKB	Application ID
Fixed MKB	0000h ~ 000Fh
Proprietary MKB	0010h ~ FFFEh
Placeholder MKB	FFFFh

Each MKB has a two-byte Application ID field in the Type and Version Record. Application ID from 0000h to 000Fh are reserved for Fixed MKBs. Application ID from 0010h to FFFEh are reserved for Proprietary MKBs. Application ID FFFFh is the specific number for a Placeholder MKB.

The first eight slots shall be occupied with the Fixed MKBs, the MKB slot #0 has MKB with Application ID 0000h, the MKB slot #1 has MKB with Application ID 0001h and so on. The next eight slots are defined in the SDA document "*SD Memory Card Specifications, Part 3: Security Specification*".

Each MKB is provided by 4C Entity, LLC. The Fixed MKB and the Placeholder MKB are pre-recorded in the System Area by the SD Memory Card manufacturer. The Fixed MKB and the Proprietary MKB may be installed to the System Area of the SD Memory Card by the CPXM compliant Host Devices. The maximum size of each MKB for CPXM is 1 Megabyte (1,048,576).

3.3.2 Hidden Area

Each CPXM Compliant SD Memory Card shall contain a Hidden Area, which is a write-protected area that is accessible only to the SD Memory Card itself and contains the Media Device Key Sets and the Authentication Keys. Media Device Key Set consists of (a) a Device Node and (b) a set of 325 Media Device Keys (K_d) and (c) the UV Descriptor (UV) associated with each Device Key, {Device Node, {K_{d_0}, UV_0}, {K_{d_1}, UV_1}, ..., {K_{d_324}, UV_{324}}}. A Media Device shall treat its Media Device Keys as Highly Confidential as defined in the CPXM License Agreement. The confidentiality of Device Node and UV Descriptors are vendor specific. The Hidden Area may also contain a Random Number Key, which is used in the random number generation process.

3.3.2.1 Authentication Key

Each CPXM Compliant SD Memory Card shall contain sixteen distinct Authentication Keys (K_{auth}) corresponding to the sixteen distinct Media Key Blocks. Each K_{auth} is a 128-bit value and is pre-computed by the manufacturer based on each Media Key (K_m) and the Media Identifier using the AES One-way Function. The Media Key K_m^0 is defined in Section 3.2.4 of the *Introduction and Common Cryptographic Elements* book of the CPXM Specification. The Media Key for an MKB is assigned by 4C Entity, LLC, and is unique for each MKB, and consequently unique for each application group defined by the SD Association. Figure 3-3 shows the procedure for Authentication Key calculation. As shown in Figure 3-3, each Authentication Key (K_{auth}) is calculated as

$$K_{auth} = AES\_G\ (K_m^0, ID_{media})$$

The AES One-way Function (AES_G) is described in the *Introduction and Common Cryptographic Elements* book of the CPXM Specification.

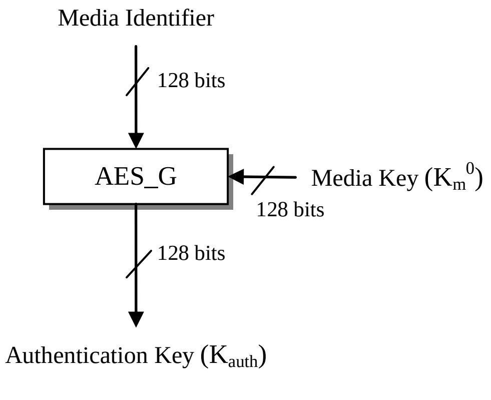

Figure 3-3 – Calculation of Authentication Key

3.3.3 Protected Area

Each CPXM Compliant SD Memory Card shall contain a Protected Area, which is a read/write area that is accessible only after successful explicit mutual authentication. The Protected Area contains the Encrypted Title Keys, the Application Keys and may also contain the Encrypted CCI (Copy Control Information), and/or Encrypted UR (Usage Rules).

3.3.3.1 Encrypted Title Key, CCI (Copy Control Information) and UR (Usage Rules)

In the SD Memory Card, each piece of content to be protected shall be encrypted by a unique Title Key.

Some SD-Applications define CCI (Copy Control Information). When CCI is defined, the Title Key and CCI of the content are concatenated and encrypted together by an Application Key, which is unique for each MKB. The Encrypted Title Keys and CCI are stored as a file in the Protected Area.

Other SD-Applications also define UR (Usage Rules). When UR is defined, the encryption of UR is defined by each application specific part of the *CPXM Specification SD Memory Card Book*.

The file format of the Protected Area and the detailed format of the Encrypted Title Keys, CCI and/or UR are specific for each application, and are described in each application specific part of the *CPXM Specification SD Memory Card Book.*

3.3.3.2 Application Key

In the SD Memory Card, there may be Application Keys stored in the Protected Area of an SD Memory Card. Each Application Key is encrypted by a Media Unique Key associated with each MKB. The Encrypted Application File is stored as a file in the Protected Area.

There are two Application Key files associated with each MKB. One is the even Application Key file and the other is the odd Application Key file. The file name convention for the Application Key files is defined in Section 3.7. The Host Device determines which Application Key file is the active one. When the Host Device retrieves an MKB from the SD Memory Card, the SD Memory Card returns it with the number of times that the MKB has been updated. If the number is even (e.g. 0, 2, 4 ...) then the even Application Key is the active one and the odd Application Key is the inactive one. Otherwise, if the number is odd (e.g. 1, 3, 5 …) then the odd Application Key is the active one and the even Application Key is the inactive one. For the purposes of this calculation, zero shall be treated as an even number.

The Encrypted Application Key logical file format is shown in Table 3-3.

Table 3-3 – Encrypted Application Key File Format

Bit / Byte	7	6	5	4	3	2	1	0
0	File Type: AA_{16}							
1	File Length: 000014_{16}							
2								
3								
4	Encrypted Application Key (K_{app})							
…								
19								

3.3.4 User Data Area

The SD Memory Card shall contain a User Data Area, which is a user-accessible read/write area that is used to store encrypted content. The User Data Area may also contain other user data. When the SD Memory Card is connected to a PC, the user data area typically looks like a normal disk.

3.3.4.1 Encrypted content

In the SD Memory Card, each piece of content to be protected shall be encrypted with a unique Title Key, and stored as an encrypted file in the User Data Area. The file system of the User Data Area is typically an ordinary FAT file system (ISO/IEC 9293-compliant FAT file system). The directory structure and file names of the encrypted content are defined in each application specific part of the *CPXM Specification SD Memory Card Book*.

3.4 Content Encryption and Decryption Protocol

Figure 3-4 illustrates a basic process for content encryption and decryption on the SD Memory Card. This figure shows the case where UR is defined and assuming that CCI is included in UR. The details are described in each application specific part of the *CPXM Specification SD Memory Card Book*.

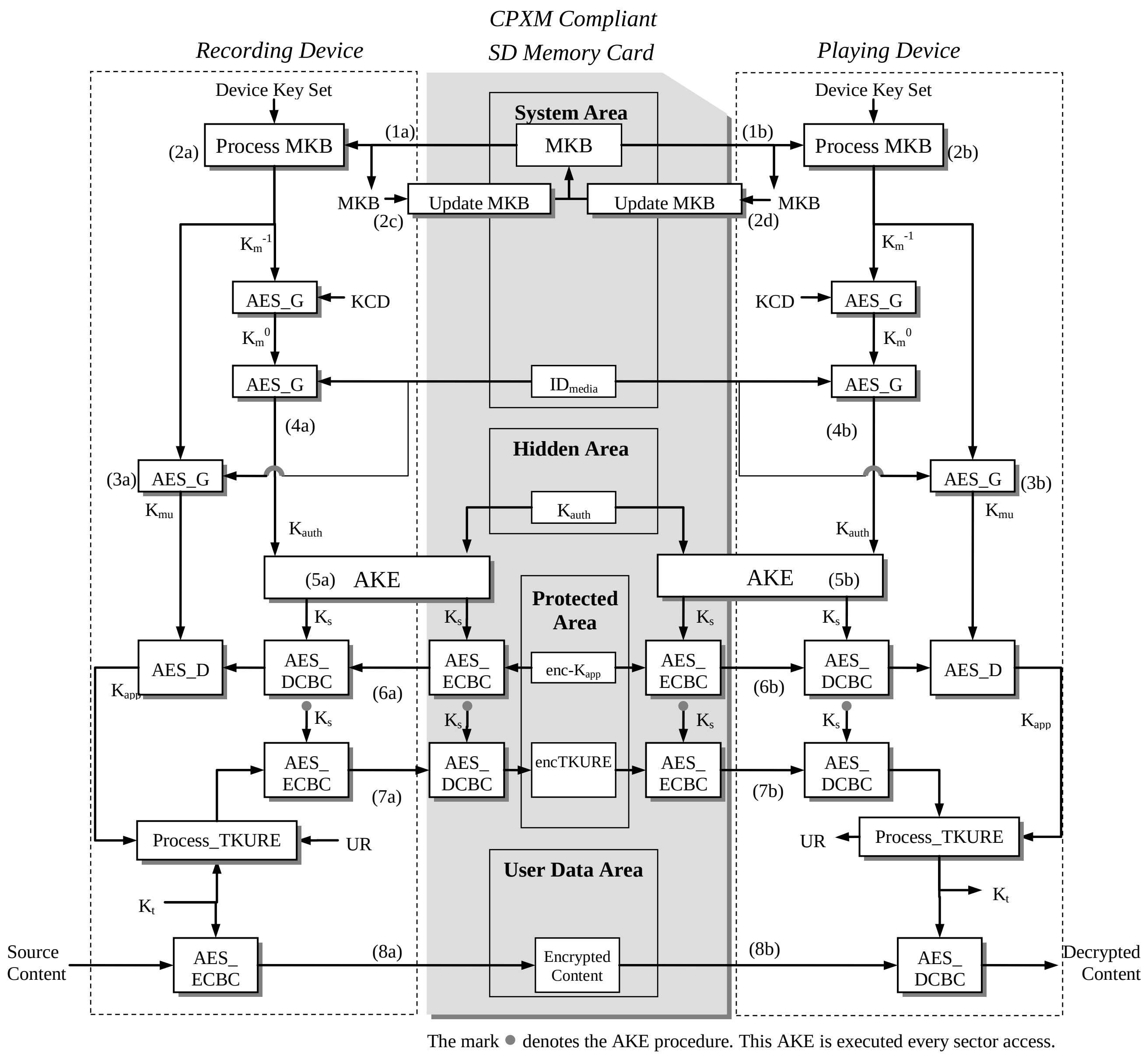

Figure 3-4 – Content Encryption and Decryption on SD Memory Card

The SD Memory Card and the Host Device (Recording Device / Playback Device) authenticate each other, and encrypt or decrypt content as follows:

- (1) The Host Device retrieves information from the SD Memory Card

(1a, 1b) The Host Device retrieves the MKB_{media} appropriate for the application, and the Media Identifier (ID_{media}) from the SD Memory Card.

- (2) The Host Device calculates the Media Key Precursor (K_m^{-1}) and Media Key K_m^0

(2a, 2b) The Host Device calculates the Media Key Precursor (K_m^{-1}) and Media Key (K_m^0) from the MKB_{media} (see chapter 3 in the *Introduction and Common Cryptographic Elements* book of this specification) using Process_MKB. Note that in Process_MKB, the Host Devices shall verify the resulting Media Key (using the Type and Version Record and the Verification Data in the Verify Media Key Record in the MKB_{media}) to confirm that the Version Number in the MKB_{media} is correct. The Version Number in the following steps shall be the same number used in the MKB.

(2c, 2d) The Host Devices execute these steps only if the Host Device supports the MKB Update. The Host Device compares the Version Number of the MKB of the Host Device (MKB_{host}) and the MKB in the SD Memory Card(MKB_{media}). If the Version Number of the Host Device's MKB (MKB_{host}) is equal to the one in the MKB_{media} retrieved from the SD Memory Card (MKB_{media}), then go to step 3. Otherwise, if the Version Number of MKB_{host} is lower than MKB_{media}, replace MKB_{host} with MKB_{media}, as the latest MKB, but only if the all of following verifications succeed;

1) verify the Media Key K_m^0 with the Verify Media Key Records in the MKB_{media}, to ensure that K_m^0; and the Version Number of the MKB_{media} are appropriate (see Section 3.2.5.2 in the *Introduction and Common Cryptographic Elements* book of this specification).
2) Check that the Version Number of the receiving MKB_{media} is greater than that of the stored MKB_{host}, by decrypting the Verify Media Key Record in the MKB_{media} using K_m^0.
3) For the MKB_{media}, verify the Check Data for MKB field in the End of Media Key Block Record in the MKB_{media}.

If the any of above verifications fail, continue to step 3, using the Media Key Precursor (K_m^{-1}) which was calculated at step 2a or step 2b. Otherwise, if the Version Number of MKB_{host} is greater than the Version Number of the MKB_{media}, the host performs MKB Update process (see Section 3.9 in this part of the book) shall be performed to update the MKB in the SD Memory Card and the process is restarted from step 1.

- (3) The Host Device calculates the Media Unique Key (K_{mu}) using the MKB_{media}

(3a, 3b) The Host Device calculates the Media Unique Key (K_{mu}) with the Media Key Precursor (K_m^{-1}) and the Media Identifier (ID_{media})

$K_{mu} = AES\_G(K_m^{-1}, ID_{media})$

- (4) The Host Devices calculates the Authentication Key (K_{auth}) using the MKB_{media}

(4a, 4b) The Host Devices calculate the Authentication Key (K_{auth}) with the Media Key (K_m^0) and the Media Identifier (ID_{media})

$K_{auth} = AES\_G(K_m^0, ID_{media})$

- (5) AKE Process

(5a, 5b) If the AKE Process succeeds, the Session Key (K_s), which is randomly generated in each AKE Process, is shared between the Host Device and the SD Memory Card. (The detail of AKE Process is shown in Section 3.4.1)

Note that, because a new Session Key is established for each new access by the Host Device, in the following step, the Session Key may not be the same among the steps (e.g. step 6 and step 7, however, the Session Key shall be shared between the Host Device and the SD Memory Card and shall be the same within each step.

- (6) The Host Device retrieves the active Application Key (K_{app})

(6a, 6b) The Encrypted Application Key (enc-K_{app}) is encrypted by the SD Memory Card using the Session Key (K_s), which is shared at step 5, using AES_ECBC described in the *Introduction and Common Cryptographic Elements* book of this specification. The doubly-encrypted Application Key (K_{app}) is sent to the Host Device. In the Host Device, it is decrypted by the Session Key (K_s), which is shared at step 5, using AES_DCBC, and that result (Encrypted Application Key) is decrypted using the Media Unique Key (K_{mu}) with AES_D described in the *Introduction and Common Cryptographic Elements* book of this specification.

In step 6a, if the associated Application Key is not stored in the Protected Area of the SD Memory Card, the Host Device shall generate and record an Application Key into the SD Memory Card by the following steps. First, the Host Device shall generate a random Application Key by the method described in Section 2.4 of the *Introduction and Common Cryptographic Elements* book of this specification. Then the Host Device encrypts the Application Key (K_{app}) using the Media Unique Key (K_{mu}) with AES_E, and then further encrypts the Encrypted Application Key with the Session Key (K_s) generated in the AKE process, using AES_ECBC. The doubly-encrypted Application Key is sent to the SD Memory Card. In the SD Memory Card, it is decrypted with the Session Key (K_s) using AES_DCBC, and that result (Encrypted Application Key) is stored in the Protected Area.

- (7a) The Host Device encrypts the Title Key and the Usage Rules

When the content is encrypted, the Recording Device generates a random a 128-bit Title Key by the method described in Section 2.4 of the *Introduction and Common Cryptographic Elements* book of this specification. The Recording Device generates an Encrypted TKURE with the Title Key (K_t), the Usage Rules (UR) and the Application Key (K_{app}).

Then, the Encrypted TKURE are further encrypted with the Session Key (K_s) generated in the AKE Process again, using AES_ECBC. The double encrypted TKURE is sent to the SD Memory Card, where it is decrypted with the Session Key (K_s) using AES_DCBC. This Session key is shared at the AKE Process. The result of this computation, the encrypted TKURE is stored in the Protected Area.

- (7b) The Host Device decrypts the doubly-encrypted Title Key and the doubly-encrypted Usage Rules

The Encrypted TKURE are again encrypted by the SD Memory Card with the Session Key (K_s) generated in the AKE Process again, using AES_ECBC. The double encrypted TKURE is sent to the Playback Device, where it is decrypted with the Session Key (K_s) using AES_DCBC. This Session key is shared at the AKE Process. The result of this computation, the encrypted TKURE is processed using the Application Key (K_{app}) and provides a decrypted Title Key (K_t) and decrypted Usage Rules (UR).

- (8a) The Host Device encrypts the content

The Recording Device shall protect each piece of content by encrypting it with the Title Key (K_t) using AES_ECBC. Note that if the content delivered to the Recording Device is already encrypted in this way the Recording Device does not perform this encryption step. The Recording Device then sends the content to the SD Memory Card, and stores the content in the User Data Area.

- (8b) The Host Device decrypts the encrypted content

The Playback Device uses the Title Key (K_t) decrypted at step (7b) to decrypt the encrypted content using AES_DCBC.

3.4.1 Authentication and Key Exchange (AKE)

The SD Memory Card supports explicit mutual authentication with a Host Device (Recording Device and Playback Device) for protection against save-restore attacks and man-in-the-middle attacks.

When a device accesses data stored in the Protected Area, an SD Memory Card and the Host Device authenticate each other by a challenge-response protocol. When this authentication is successful, they share a secure common session key. This protocol is called Authentication and Key Exchange (AKE). The session key is used for encrypting and decrypting the protected data on the bus between the Host Device and the SD Memory Card. As shown in Figure 3-5, the AKE protocol also protects the argument field (32 bits) of the security commands, which are used to access the Protected Area as described in chapter 3 of *SD Memory Card Specifications -Part 3 Security specification*.

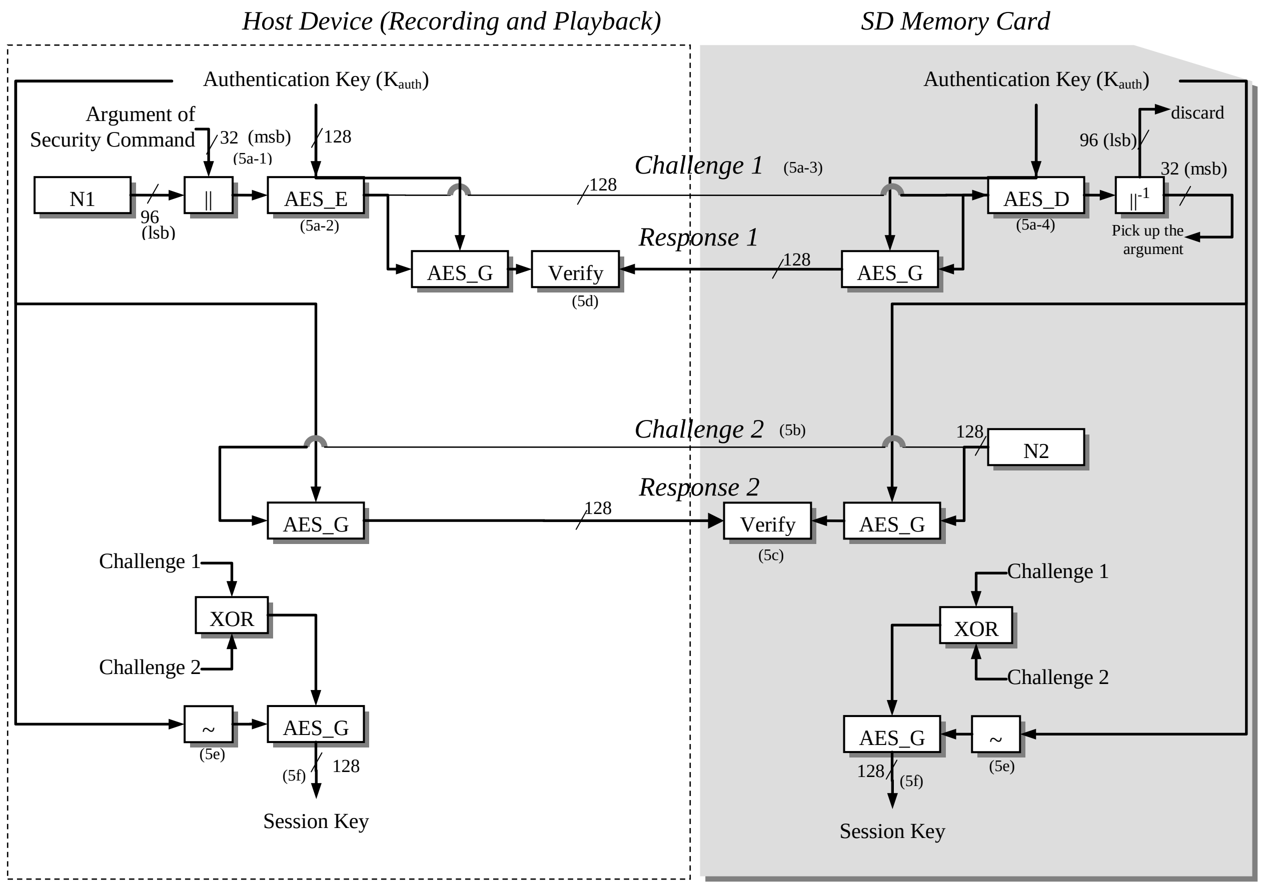

Figure 3-5 – Details of Authentication and Key Exchange (AKE) on SD Memory Card

The AKE procedure is the following:

(5a-1) In the Host Device (Recording Device / Playback Device), the 32-bit argument of the security command, which is described in the *SD Memory Card Specifications – Part 3 Security Specification*, is extended to 128 bits by concatenating a 96-bit nonce N1, generated by a method described in Section 2.4 of the *Introduction and Common Cryptographic Elements* book of this specification, to it.

(5a-2) The resulting 128 bits are encrypted with K_{auth} (Authentication Key) using AES_E.

$$\text{Challenge1} = \text{AES\_E}(K_{auth}, (\text{the 32-bit argument} \parallel \text{N1}))$$

(5a-3) The Host Device sends the cipher-text (128 bits) as Challenge1 to the SD Memory Card.

(5a-4) In the SD Memory Card, the received data is decrypted with K_{auth} using AES_D. The decrypted argument (the most significant 32 bits) is remembered securely and is used in the subsequent security command. The least significant 96 bits of the decrypted data are discarded.

(5b) SD Memory Card generates a nonce N2 (128 bits) generated by a method described in Section 3.4.1.1 and sends N2 as a Challenge2 to the Host Device.

Challenge2 = N2

(5c) The Host Device calculates Response2 = AES_G(K_{auth}, Challenge2) and returns it to the SD Memory Card. The SD Memory Card calculates Vresponse2 =AES_G(K_{auth}, Challenge2), and checks if Vresponse2 is equal to Response2. If they match, then go to step 5d. Otherwise, this whole AKE procedure shall be aborted.

(5d) The SD Memory Card calculates Response1 = AES_G(K_{auth}, Challenge1), and returns it to the Host Device. The Host Device calculates Vresponse1 = AES_G (K_{auth}, Challenge1), and checks if Vresponse1 is equal to Response1. If they match, then go to step 5e. Otherwise, this AKE procedure shall be aborted.

(5e) The bitwise complement of Authentication Key ($\sim K_{auth}$) is generated in both the SD Memory Card and the Host Device.

(5f) A Session Key (K_s) is calculated as

$$K_s = \text{AES\_G}\ (\sim K_{auth}, \text{Challenge1} \oplus \text{Challenge2})$$

in both the SD Memory Card and the Host Device. The Challenge and the Response values and the Session Key (K_s) are 128 bits long.

When this authentication procedure is successful, the data field of the security command is encrypted using the Session Key (K_s) with AES_ECBC.

The SD Memory Card shall enforce the order of steps in the protocol, by ignoring any commands received out of order. The SD Memory Card shall abort the process after step 5c if the Host Device's response is not correct.

3.4.1.1 Random Number Generation

The SD Memory Card uses a unique random number generator for the challenges in this protocol. Note that with this protocol, even if the challenges can be predicted, the Session Key cannot. Therefore, a random number generator used solely for this protocol does not need to produce an unpredictable sequence, like the generator specified in the *Introduction and Common Cryptographic Elements* book of this specification does. Its only requirements are: 1) the seed cannot be set from outside the card, and 2) the numbers come from a long sequence (approximately 2^{128}).

3.5 Accessing the Protected Area

In the case of the SD Memory Card, AKE is used to control access to the Protected Area.

Three operations are possible on the Protected Area:

- Write a protected data item, for example the Encrypted Application Key, Encrypted Title Key, CCI and/or UR. (part of recording content)
- Read a protected data item (part of playback)
- Delete a protected data item (part of removing content).

Each time one of these operations is performed, a successful AKE must first take place. The resulting Session Key is used to encrypt the data field of the security command, which is described in the *SD Memory Card Specifications – Part 3 Security Specification*. The data field is encrypted with AES in CBC mode. The argument field, while not encrypted, is protected against tampering, because it forms part of the challenge in the AKE protocol.

Note that the actual security commands used are block-oriented; they deal with units of 512-byte sectors. Thus, to write a small item, it might be necessary to first read the data block to make the small change. These straightforward details are omitted from this specification.

As explained in chapter 3 of *SD Memory Card Security Specifications – Part 3 Security Specification*, sectors in the protected area may be written in either “mode 0” or “mode 1”. Mode 0 sectors can be read by any application that can perform a successful AKE. Mode 1 sectors can only be read successfully by the same application that wrote them – or at least by one that uses the same MKB. In other words, when a mode 1 sector is written, the card records the MKB slot number used during the preceding AKE. If the sector is read using an AKE with a different MKB, an error occurs. File system metadata sectors shall be written in mode 0. File data sectors shall be written in mode 1.

3.5.1 Secure Write Process to the Protected Area

Figure 3-6 shows the protocol flow for the Host Device writing an Encrypted Title Key and Usage Rules (TKURE) to the Protected Area using an SD Memory Card command, which is described in the *SD Memory Card Specifications – Part 3 Security Specification*.

This figure shows the case where UR is defined. In the case where CCI are defined, in steps 5-1 ~ 5-3 of Figure 3-6, UR should be replace with CCI, unless explicitly noted otherwise in an application specific part of the *CPXM Specification SD Memory Card Book*.

A Host Device (e.g. Recording, Playback Device) may securely remember the Media Unique Key value and the Authentication Key value calculated during the first AKE, in order to avoid the "Calculation of Media Unique Key Process (step 1 of Figure 3-6)" in the subsequent AKE procedures with the same SD Memory Card.

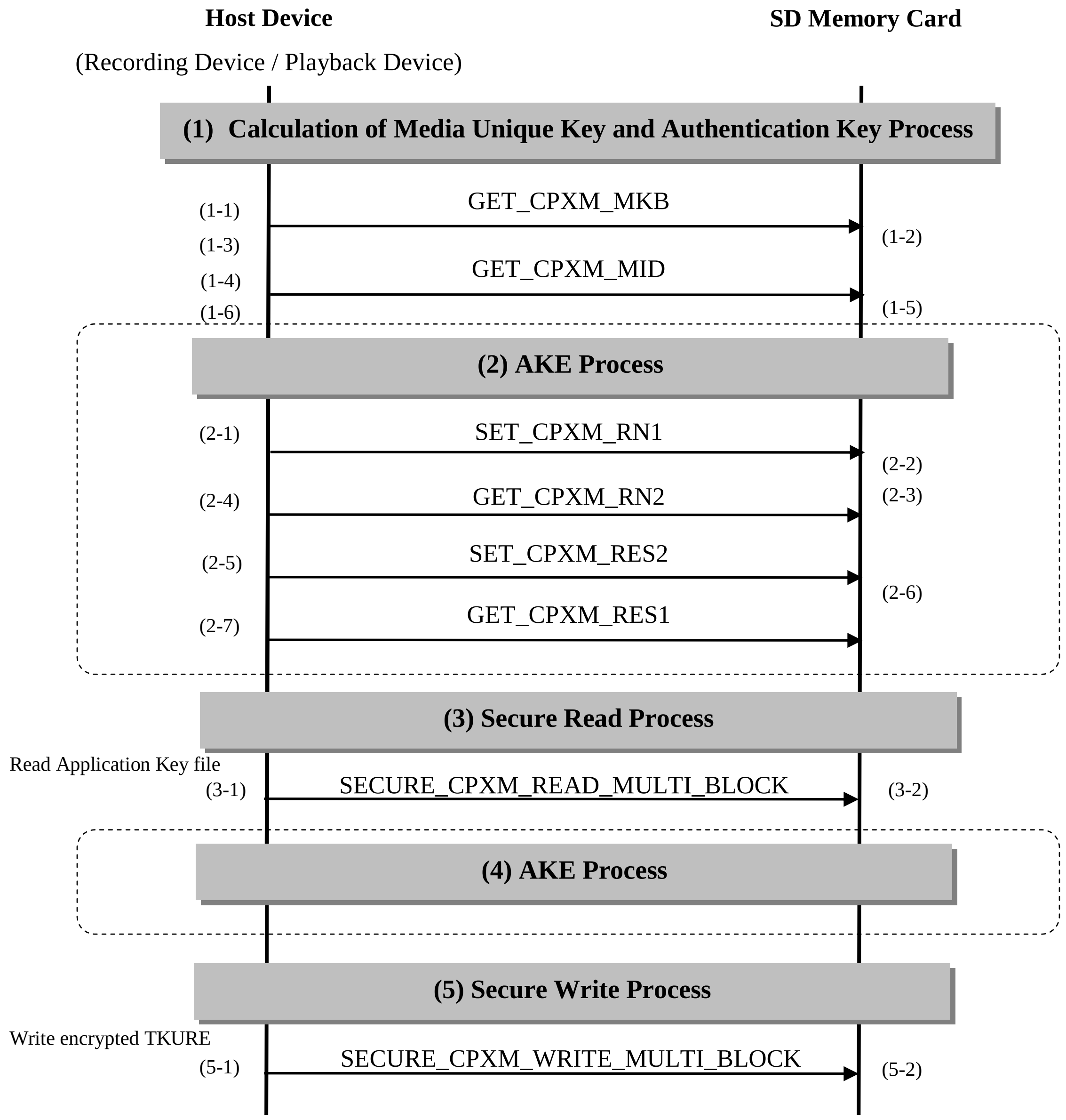

Figure 3-6 – Protocol Flow of "Secure Write Process" to the Protected Area

(1) The Host Device calculates the Media Unique Key and Authentication Key

(1-1): The Host Device sends a request to retrieve an MKB from the SD Memory Card with "GET_CPXM_MKB" command.

(1-2): The SD Memory Card returns the requested MKB.

(1-3): The Host Device calculates a Media Key from the MKB.

(1-4): The Host Device sends a request to retrieve the Media Identifier from the SD Memory Card.

(1-5): The SD Memory Card returns the Media Identifier.

(1-6): The Host Device calculates the Media Unique Key and the Authentication Key with the Media Key and the Media Identifier.

(2) AKE Process between the Host Device and the SD Memory Card

(2-1): The Host Device generates a nonce N1 and concatenates the argument for the subsequent security command to the nonce N1, and encrypts it with the Authentication Key as a Challenge1. The Host Device sends the Challenge1 with the "SET_CPXM_RN1" command. The subsequent security command denotes either secured read or secure write to the Protected Area (i.e. "SECURE_CPXM_READ_MULTI_BLOCK" or "SECURE_CPXM_WRITE_MULTI_BLOCK").

(2-2): The SD Memory Card decrypts the Challenge1 (RN1) to get the argument (32 bits) of the subsequent security command.

(2-3): The Host Device retrieves Challenge2 (RN2) by sending a command "GET_CPXM_RN2".

(2-4): The SD Memory Card generates a nonce value Challenge2 (RN2) and returns it to the Host Device.

(2-5): The Host Device calculates Response2 (RES2) = AES_G(Authentication Key, Challenge2) and sends it with "SET_CPXM_RES2" command.

(2-6): The SD Memory Card calculates Vresponse2 = AES_G(Authentication Key, Challenge2) and verifies it with received Response2. If the verification fails, stop this AKE process. Otherwise, the SD Memory Card calculates a Session Key = AES_G(bitwise compliment of the Authentication Key, Challenge1 $\oplus$ Challenge2).

(2-7): The Host Device calculates Vresponse1 = AES_G(Authentication Key, Challenge1) and verifies it with received Response1. If the verification fails, stop this AKE process. Otherwise, the Host Device calculates a Session Key = AES_G(bitwise compliment of the Authentication Key, Challenge1 $\oplus$ Challenge2).

(3) Secure Read Process

The Host Device and the SD Memory Card carry out the Authentication and Key Exchange described above, and share a session key.

(3-1): The Host Device reads Encrypted Application Key file with a "SECURE_CPXM_READ_MULTI_BLOCK" command.

(3-2): The SD Memory Card encrypts the Encrypted Application Key file by AES_ECBC using the Session Key which is the data part of the "SECURE_CPXM_READ_MULTI_BLOCK" command and returns it to the Host Device.

(3-3): The Host Device decrypts the data part of the "SECURE_CPXM_READ_MULTI_BLOCK" command with the Session Key using AES_DCBC, and receives the Application Key file. The Host Device decrypts the Encrypted Application Key in the Application Key file with the Media Unique Key using AES_D.

(4) AKE Process between the Host Device and the SD Memory Card

The detail is the same as steps 2-1 ~ 2-7 of Figure 3-6. Here, it is supposed to generate another session key as the Session Key 2.

(5) Secure Write Process

The Host Device and the SD Memory Card carry out the Authentication and Key Exchange described above, and share a session key.

(5-1): The Host Device encrypts the Title Key and the UR with the Application Key using AES_E. The Host Device writes the encrypted Title Key and the encrypted UR with the "SECURE_CPXM_WRITE_MULTI_BLOCK" command, where the data field of the command is encrypted by AES_ECBC with the Session Key 2.

(5-2): The SD Memory Card decrypts the data part of the "SECURE_CPXM_WRITE_MULTI_BLOCK" command with the Session Key 2 using AES_DCBC, and stores the encrypted Title Key and the encrypted UR into the Protected Area.

3.5.2 Secure Read Process with the Protected Area

Figure 3-7 shows the protocol flow for the Host Device reading an Encrypted Title Key and Usage Rules from the Protected Area using SD Memory Card command, which is described in chapter 3 of the *SD Memory Card Specifications – Part 3 Security Specification*.

This figure shows the case where UR is defined. In the case where CCI is defined, in steps 9-1 ~ 9-3 of Figure 3-7, UR is replace with CCI, unless explicitly noted otherwise in an application specific part of *CPXM Specification SD Memory Card Book*.

In Figure 3-7, it is supposed that a Host Device securely holds the Media Unique Key value and Authentication Key value before the AKE process. If not, it is necessary to execute the “Calculation of Media Unique Key Process (step 1 of Figure 3-6)” before the AKE process.

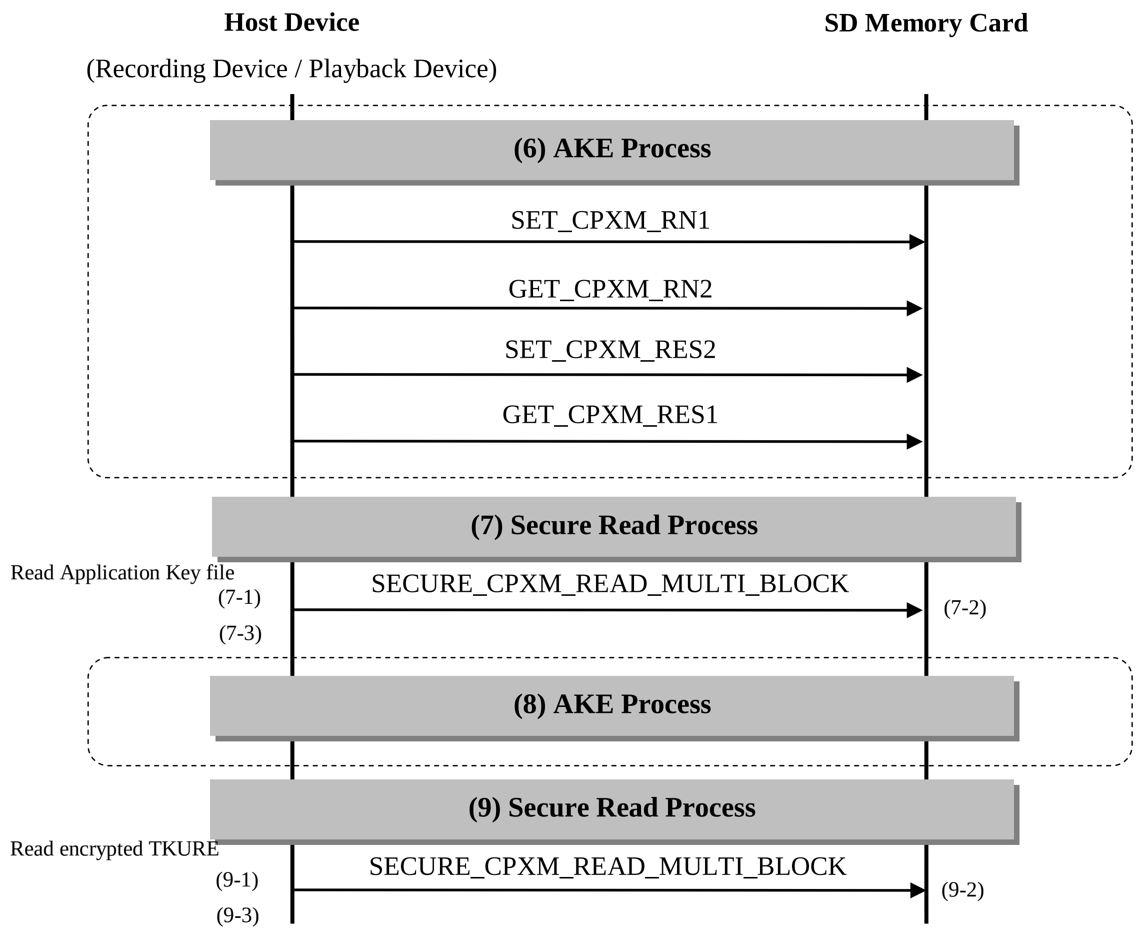

Figure 3-7 – Protocol Flow of “Secure Read Process” from the Protected Area

(6) AKE Process between the Host Device and the SD Memory Card

The detail is the same as steps 2-1 ~ 2-7 of Figure 3-6. Here, it is supposed to generate another session key as the Session Key 3.

(7) Secure Read Process

The Host Device and the SD Memory Card carry out the Authentication and Key Exchange described above, and share a session key, then the Host Device obtains the Application Key. The detail is the same as steps 3-1 ~ 3-3 of Figure 3-6.

(8) AKE Process between the Host Device and the SD Memory Card

The detail is the same as steps 2-1 ~ 2-7 of Figure 3-6. Here, it is supposed to generate another session key as the Session Key 4.

(9) Secure Read Process

(9-1): The Host Device reads the encrypted Title Key and the encrypted UR with a command "SECURE_CPXM_READ_MULTI_BLOCK" command.

(9-2): The SD Memory Card doubly encrypts the Encrypted Title Key and doubly encrypts the CCI by AES_ECBC using the Session Key 4, which are the data parts of the "SECURE_CPXM_READ_MULTI_BLOCK" command and returns it to the Host Device.

(9-3): The Host Device decrypts the data part of the "SECURE_CPXM_READ_MULTI_BLOCK" command with the Session Key 4 using AES_DCBC, and receives the encrypted Title Key and encrypted CCI. The Host Device decrypts the encrypted Title Key and encrypted UR with the Application Key using AES_D.

3.5.3 Secure Title Key Delete Process

An accessing Host Device may delete an Encrypted Title Key from the Protected Area by linking together the write and read operations described above. First a write operation overwrites the Encrypted Title Key with a selected value, and then a read operation reads the value to confirm that the overwriting was successful. Figure 3-8 shows the protocol flow for the Host Device deleting an Encrypted Title Key in the Protected Area.

This figure shows the case where UR is defined. In the case where CCI is defined, in step 11 of Figure 3-8, UR is replace with CCI, unless explicitly noted otherwise in an application specific part of *CPXM Specification SD Memory Card Book*.

In Figure 3-8, it is supposed that a Host Device securely holds the Media Unique Key value and Authentication Key value before the AKE process. If not, it is necessary to execute the “Calculation of Media Unique Key Process (step 1 of Figure 3-6)” before the AKE process.

Note: the SD Memory Card has a “Secure Erase” command. This command is a low-level writing command to improve performance of writes in the Protected Area. It is *not* a substitute for this Secure Title Key Delete protocol.

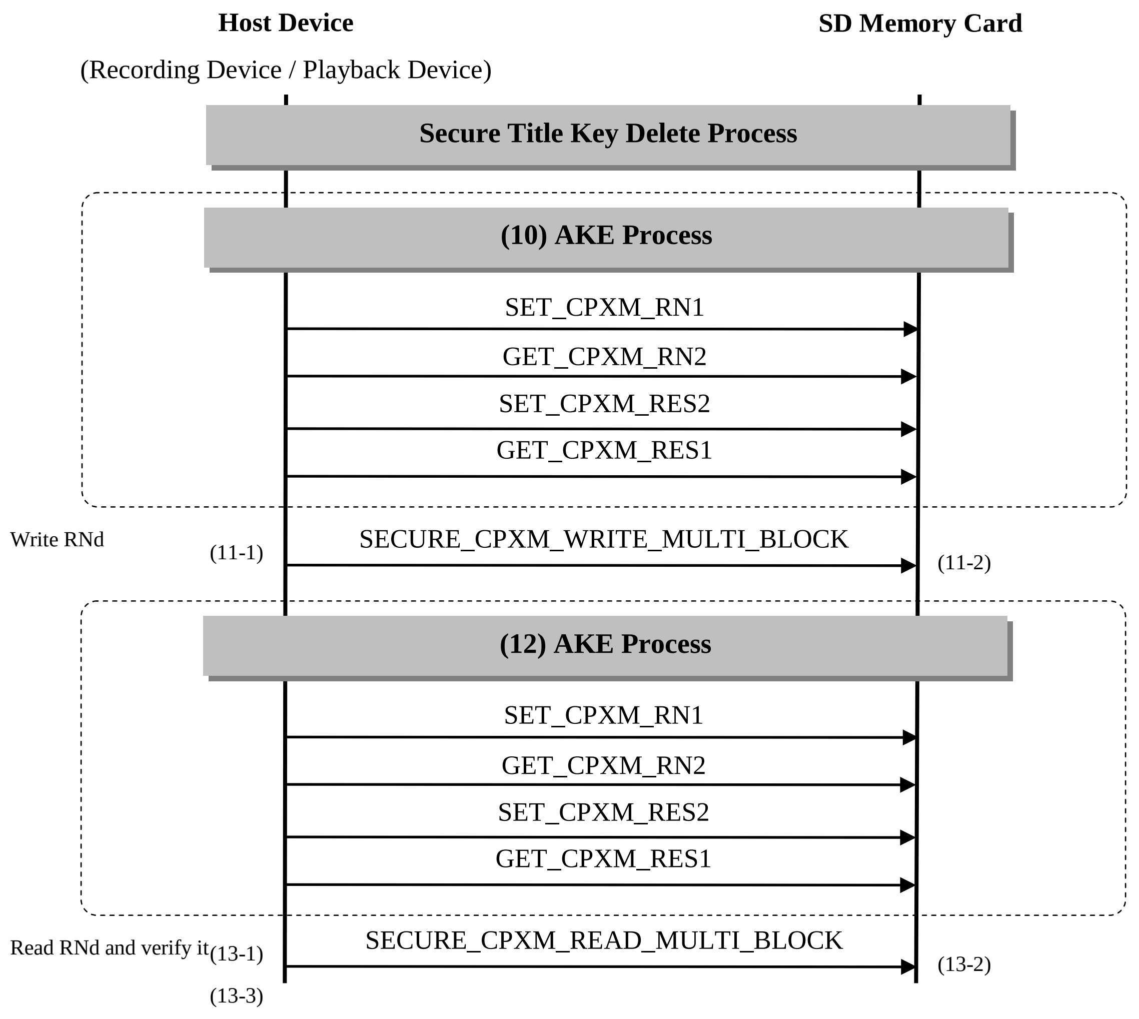

Figure 3-8 – Protocol Flow of “Secure Title Key Delete Process” to the Protected Area

(10) AKE Process between the Host Device and the SD Memory Card

The details are the same as those described in steps 2-1 to 2-7 of Figure 3-6. Here, it is supposed to generate another session key as the Session Key 5.

(11) Overwrite Process

(11-1): The Host Device overwrites the field of the Encrypted Title Key with a random number for delete (RNd) by the "SECURE_CPXM_WRITE_MULTI_BLOCK" command. Here, the data, including the RNd, to be written by the "SECURE_CPXM_WRITE_MULTI_BLOCK" command is encrypted by AES_ECBC with the Session Key 5. The Host Device shall hold the RNd securely for later use. Note that the mode of the "*SECURE_CPXM_WRITE_MULTI_BLOCK" command* shall be set "mode 1". Here, regarding the "mode" of the "*SECURE_CPXM_WRITE_MULTI_BLOCK"* command, refer to the *SD Memory Card Specifications – Part 3 Security Specification*.

(11-2): The SD Memory Card receives the encrypted data including the RNd and decrypts it using AES_DCBC with the Session Key 3, and then stores it in the Protected Area.

(12) AKE Process between the Host Device and the SD Memory Card

The detail is the same as steps 2-1 ~ 2-7 of Figure 3-6. Here, it is supposed to generate another session key as the Session Key 6.

(13) Verification Process

(13-1): The Host Device sends the SECURE_CPXM_READ_MULTI_BLOCK" command to read the data including RNd which is the same sector written at step 11.

(13-2): The SD Memory Card fetches the data from the Protected Area and encrypts the data including the RNd with AES_ECBC using the Session Key 6 and returns it to the Host Device.

(13-3): The Host Device receives the data including the RNd and checks the RNd is equal to the RNd securely held in the step 11-1. Here, before checking, the data including the RNd to be read with the "SECURE_READ_MULTI_BLOCK" command is decrypted by AES_DCBC using the Session Key 6.

If deletion of the Title Key is not verified, the Host Device must assume the deletion has not occurred.

The actual Title Key usually resides within a single sector. However, it might cross sector boundaries. In this case, all corresponding sectors must follow the above protocol.

3.6 Content Encryption and Decryption Format

3.6.1 General Principle

The content to be protected shall be encrypted. When the content is more than 128 bits, the content is encrypted using AES cipher in CBC mode, as described in the *Introduction and Common Cryptographic Elements* book of this Specification. If the last block is incomplete (a residual block, i.e. the length of the last block is less than 128 bits) it is left unencrypted (usually, each SD-Application does not contain critical data in the last few bytes). If the entire content is less than 128 bits, the data is unencrypted. A detailed encryption and decryption format of each object (e.g. AAC, MP3, JPEG, MPEG etc.) is described in each application specific part of *CPXM Specification SD Memory Card Book*.

3.7 File Format of the Protected Area

This section shows the file format of the Protected Area. The physical allocation of the file system of the Protected Area is described in the *SD Memory Card Specifications –Part 3 Security Specification*.

The directory and file configuration in the Protected Area are as follows, unless explicitly noted otherwise in an application specific part of *CPXM Specification SD Memory Card Book*:

- In the Protected Area, a directory is assigned by each application. The directory name is "XXX" (Here, "XXX" = the name of SD-Application which is defined by the SD Association)
- Within each directory, the file stored Encrypted Title Key using CPXM technology is named "YYY.KYX" (Here, "YYY" is assigned by each SD-Application).
- In the Protected Area, a file stored Encrypted Application Key associated with each MKB is named “APP_nn_x.KYX” (Here, “nn” is associated with the MKB slot number in decimal (00 to 15), and x is a character denoting “1” is odd and “2” is even.).

For example, as shown in Figure 3-9, the directory name is SD_APPLI and the file stored Encrypted Title Key is APPLInnn.KYX (Here, “nnn” is decimal number).

The detail of directory and file configuration in the Protected Area of each SD-Application is described in each application specific part of *CPXM Specification SD Memory Card Book*.

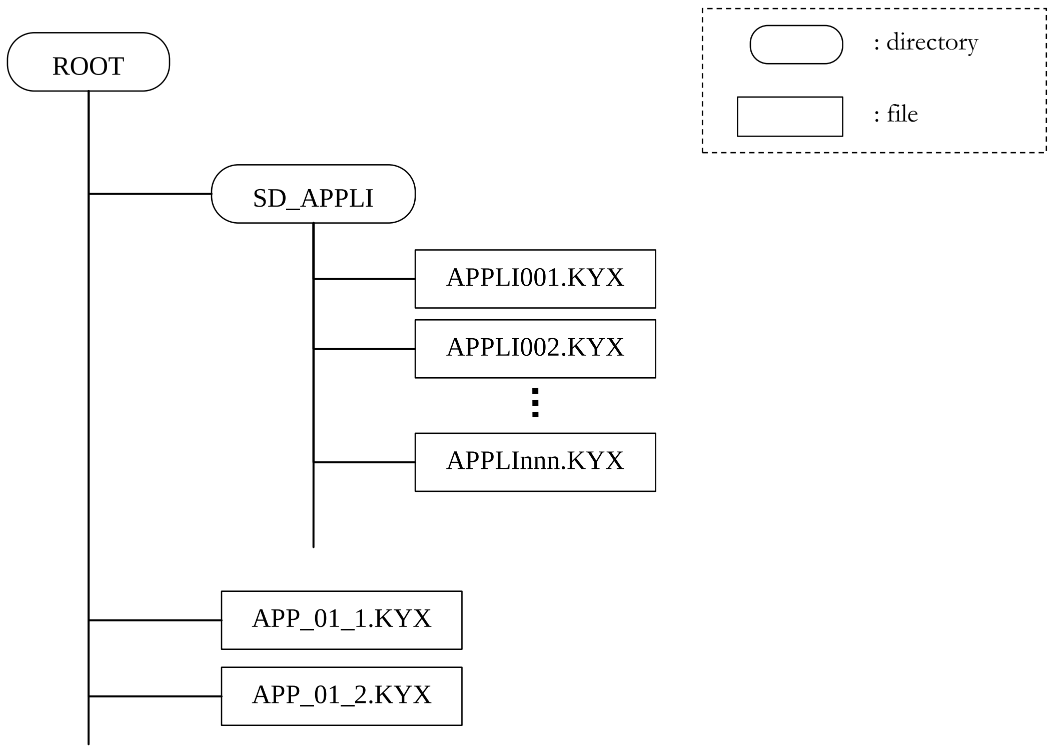

Figure 3-9 – Directory and File Configuration

3.8 Process

Each SD-Application specific process, (e.g. recording, move, copy, playback, viewing etc.), is described in the application specific part of the *CPXM Specification SD Memory Card Book*.

3.9 MKB Update

This section describes the MKB update procedure. The devices that support any recording functions, such as recording new content, move content or copy content from a host device to an SD Memory Card shall support this MKB Update feature. The devices that support only reading functions, such as playback content, move content or copy content from an SD Memory Card to a host device may support this feature. SD Memory Cards shall support this feature. This MKB update procedure also supports field installation of the MKB to an unassigned MKB slot, i.e. replacing a Placeholder MKB with either a Fixed MKB or a Proprietary MKB. The term "updating Host Device" is defined in this section and referred as a Host Device that supports the MKB Update feature.

Figure 3-10 shows the protocol flow for updating an MKB into the SD Memory Card.

In the SD Memory Card up to sixteen MKBs are pre-stored. All of the sixteen MKBs are updatable.

In the following steps, if the updating Host Device aborts the process at any point, the Host Device shall not record any new content to the SD Memory Card.

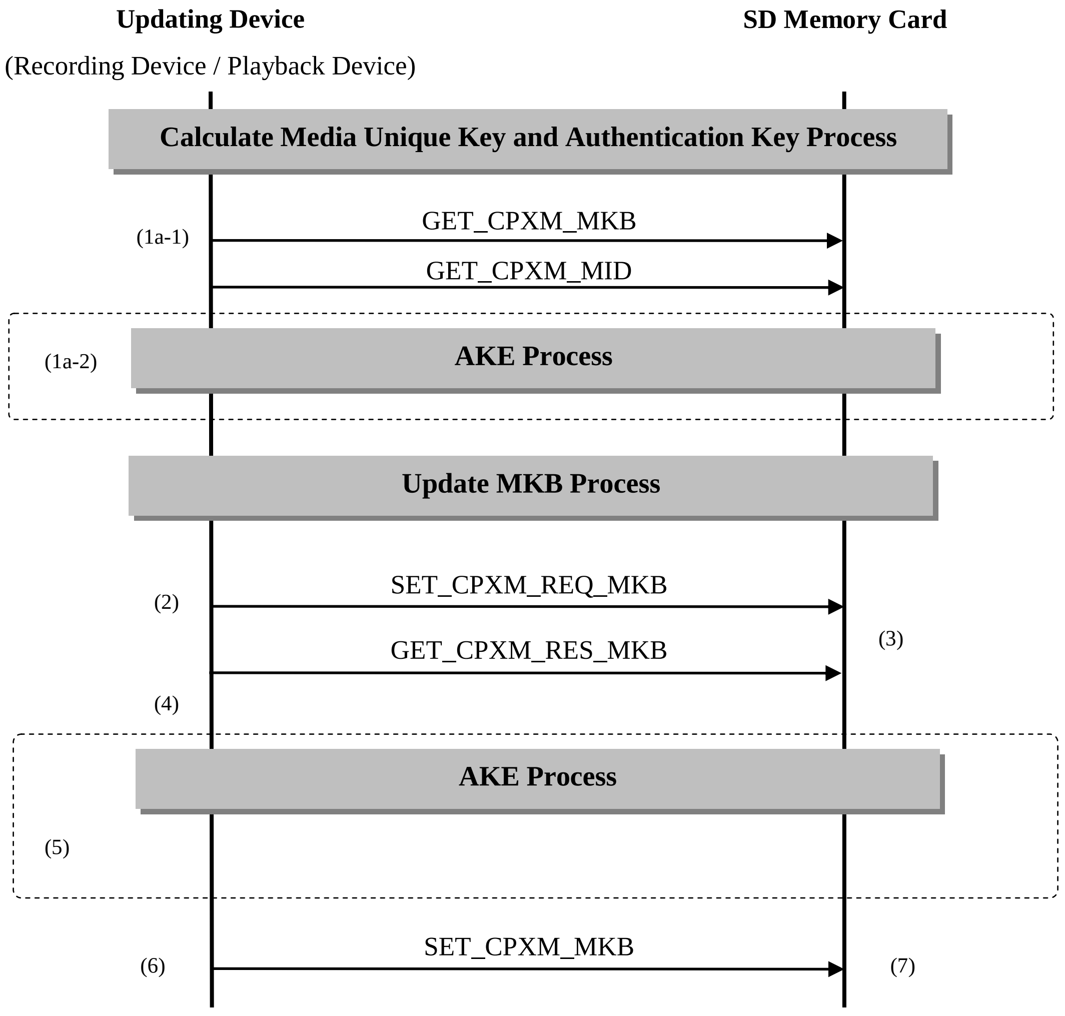

Figure 3-10 – Protocol Flow of "MKB Update Process"

(1) AKE using the MKB which is to be updated in the SD Memory Card (MKB_{media}) between the updating Host Device and the SD Memory Card

(1-1): The Updating Device retrieves an MKB_{media} from the SD Memory Card with the "GET_CPXM_MKB" command, and it calculates a Media Key (K_m^0) from the MKB_{media} with a Host Device Key (K_{d_h}). The Updating Device retrieves a Media Identifier from the SD Memory Card with the "GET_CPXM_MID" command and it calculates the Authentication Key (K_{auth}) with the Media Key (K_m^0) and the Media Identifier (ID_{media}). The Updating Device may skip this step 1-1 if this process has been performed already in the previous step.

(1-2): AKE using "K_{auth}". If this authentication step succeeds, the same Session Key (K_s) is generated at both the updating Host Device and the SD Memory Card.

(2) The updating Host Device sends a Media Key Base Request to the SD Memory Card

(2-1): The updating Host Device finds the 16-byte entry of Media Key Data in the Media Key Data Record and the 6-byte entry of the UV Descriptor in the Explicit Subset-Difference for the Media Key Record from the MKB in the updating Host Device (MKB_{host}) corresponding to the Device Node included in the ID_{media} of the SD Memory Card. If the updating Host Device cannot find the Media Key Data or the UV Descriptor, then this MKB update process shall be aborted.

(2-2): The updating Host Device concatenates the 16-byte Type and Version Record, the 16-byte Media Key Data, the 6-byte UV Descriptor in the MKB_{host} 3-byte MKB Length of the MKB_{host} in byte, and the 7-byte verification data and to generate a 48-byte Media Key Base Request (K_m^0-Request).

K_m^0-Request = Type and Version Record || Media Key Data || UV Descriptor || MKB Length || $FEEDDEADBEEF4C_{16}$

(2-3): The updating Host Device encrypts the K_m^0-Request using AES_ECBC with the Session Key generated at step (1) and sends the encrypted data to the SD Memory Card using the "SET_CPXM_REQ_MKB" command.

(2-4): The updating Host Device calculates the Expected Media Key Base Response (Expected-K_m^0-Response) with the Media Key and the Media Key Data, and stores it securely for later use. The updating Host Device calculates K_m^0 from the MKB_{host} and its own Host Device Key Sets.

Expected-K_m^0-Response = AES_G(K_m^0, Media Key Data)

(3) The SD Memory Card calculates the Media Key Base Response (K_m^0-Response) and sends it back to the updating Host Device

(3-1): The SD Memory Card decrypts the received encrypted K_m^0-Request using AES_DCBC with the K_s generated at step (1). The SD Memory Card shall verify the correctness of the decrypted data by observing the following condition:

[AES_DCBC(K_s, encrypted K_m^0-Request)]$_{lsb\_56}$ == $FEEDDEADBEEF4C_{16}$

If the verify succeeds, go to step (3-2), otherwise, this MKB update process shall be aborted.

(3-2): The SD Memory Card executes MKB process with the Media Key Data, Media Device Key Sets and the Version Number, and calculates a candidate Media Key (candidate K_m^0), and stores it securely for later use.

(3-3): The SD Memory Card calculates an Authentication Key (K_{auth}) with the candidate K_m^0 and the Media Identifier and stores it securely as a candidate K_{auth} in the SD Memory Card.

(3-4): The SD Memory Card calculates the 16-byte K_m^0-Response with the candidate K_m^0 and the entry of the MKB_{host}.

K_m^0-Response = AES_G(the candidate K_m^0, Media Key Data)

(4) The updating Host Device checks Media Key Base Response from the SD Memory Card

(4-1): The updating Host Device retrieves the K_m^0-Response using the "GET_CPXM_RES_MKB" command. The SD Memory Card sends the K_m^0-Response back to the updating Host Device securely using AES_ECBC with the K_s generated at step (1).

(4-2): The updating Host Device receives the encrypted K_m^0-Response and decrypts it using AES_DCBC with the K_s generated at step (1). The updating Host Device checks whether the K_m^0-Response is exactly the same as the Expected-K_m^0-Response calculated and securely stored at step (2). If they match, go to step (5), otherwise, this whole process shall be aborted.

(5) The updating Host Device records an Application Key file in the Protected Area of the SD Memory Card. When the updating Host Device is to install either a Fixed MKB or a Proprietary MKB to the MKB slot, this step shall be skipped.

(5-1): The updating Host Device executes AKE Process using the MKB in the SD Memory Card (MKB_{media}) between the updating Host Device and the SD Memory Card and generates a new K_s as required before reading and writing data from/to the Protected Area of an SD Memory Card.

(5-2): The updating Host Device securely reads the active Application Key file from the Protected Area with the "SECURE_CPXM_READ_MULTI_BLOCK" command and decrypts the Encrypted Application Key (K_{app}) with the K_{mu} derived from the MKB_{media} and the ID_{media}.

(5-3): The updating Host Device encrypts the K_{app} with the K_{mu} derived from the MKB_{host} and the ID_{media}, and securely stores the Application Key file with the "SECURE_CPXM_WRITE_MULTI_BLOCK" command as an inactive one into the Protected Area of the SD Memory Card.

For example, if the SD Memory Card returns the MKB with '2' as the number of time that the MKB has been updated in step 1-1, then the even Application Key file is the active one and the odd Application Key file is the inactive one. The Updating Device securely reads the even Application Key file and decrypts the Application Key with the K_{mu} derived from the MKB_{media} and the ID_{media}. The Updating Device re-encrypts the Application Key with the K_{mu} derived from the MKB_{host} and the ID_{media} and securely writes the odd Application Key file as described in the above step 5-2 and 5-3.

(6) The updating Host Device sends the MKB in the updating Host Device (MKB_{host}) to the SD Memory Card

The updating Host Device sends the MKB_{host} to the SD Memory Card by using the "SET_CPXM_MKB" command.

(7) The SD Memory Card switches the MKB and K_{auth} to the new ones

After the reception of the MKB_{host}, the SD Memory Card shall conduct following verifications;

a) When the updating Host Device is to install either a Fixed MKB or a Proprietary MKB to the MKB slot

1) Verify the candidate K_m^0 with the Verify Media Key Record in the MKB_{host} to ensure that the candidate K_m^0 and the Version Number of the MKB_{host} are appropriate (see Section 3.2.5.2 in the *Introduction and Common Cryptographic Elements* book of this specification).

2) For the MKB_{host}, verify the Check Data for MKB field in the End of Media Key Block Record in the MKB_{host}.

3) Check whether the Application ID in the Type and Version Record of the MKB_{host} is for either a Fixed MKB or a Proprietary MKB.

4) Check whether the Application ID in the Type and Version Record of the MKB_{host} does not exist in any of the other MKB slots.

b) When the updating Host Device is to update a Fixed MKB or a Proprietary MKB already installed in the SD Memory Card

1) Verify the candidate K_m^0 with the Verify Media Key Record in the MKB_{host} to ensure that the candidate K_m^0; and the Version Number of the MKB_{host} are appropriate (see Section 3.2.5.2 in the *Introduction and Common Cryptographic Elements* book of this specification).

2) Check whether the Version Number of the receiving MKB_{host} is greater than that of the stored MKB_{media} by decrypting the Verify Media Key Record in the MKB_{host} using the candidate K_m^0.

3) For the MKB_{host}, verify the Check Data for MKB field in the End of Media Key Block Record in the MKB_{host}.

4) Check whether the Application ID in the Type and Version Record of the MKB_{host} is identical with the Application ID in the MKB_{media}.

If any of the above verifications fail, this whole Update MKB Process shall be aborted. Otherwise, if the verifications all succeed, the SD Memory Card stores the MKB_{host} into itself and replaces the MKB_{media} with it. And also K_{auth} is replaced with the candidate K_{auth} which was securely stored at step (5). Additionally, the SD Memory Card shall increment the number of times that the MKB has been updated. The SD Memory Card shall discard the candidate K_m^0.