Advanced Access Content System (AACS)

Blu-ray Disc Pre-recorded Book

Intel Corporation

International Business Machines Corporation

Matsushita Electric Industrial Co., Ltd.

Microsoft Corporation

Sony Corporation

Toshiba Corporation

The Walt Disney Company

Warner Bros.

Revision 0.921

June 06, 2008

This page is intentionally left blank.

Preface

Notice

THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Intel Corporation, International Business Machines Corporation, Matsushita Electric Industrial Co., Ltd, Microsoft Corporation, Sony Corporation, Toshiba Corporation, The Walt Disney Company and Warner Bros. disclaim all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

This document is subject to change under applicable license provisions.

Intellectual Property

Implementation of this specification requires a license from AACS LA LLC.

Contact Information

Please address inquiries, feedback, and licensing requests to AACS LA LLC:

- Licensing inquiries and requests should be addressed to licensing@aacsla.com.
- Feedback on this specification should be addressed to comment@aacsla.com.

The URL for the AACS LA LLC web site is http://www.aacsla.com.

This page is intentionally left blank.

Table of Contents

List of Figures

This page is intentionally left blank.

List of Tables

Chapter 1
Introduction

1.1 Purpose and Scope

The Advanced Access Content System (AACS) specification defines an advanced, robust and renewable method for protecting audiovisual entertainment content, including high-definition content. The specification is organized into several "books". The *Introduction and Common Cryptographic Elements* book defines cryptographic procedures that are common among the various defined uses of the protection system. The *Pre-recorded Video Book* defines common details for using the system to protect audiovisual content distributed on any kind of pre-recorded (read-only) storage media. This document (the *Blu-ray Disc Pre-recorded Book*) specifies additional details for using the system to protect audiovisual content distributed on pre-recorded Blu-ray Disc Read-Only Media.

When there is a discrepancy between a format-independent book and this book then this 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 AACS LA is responsible for establishing and administering the content protection system based in part on this specification.

Note: In this specification the words "BD Pre-recorded Disc" means Blu-ray Disc Read-Only Media (BD-ROM).

1.2 Overview

In the Blu-ray Disc Pre-recorded Book, the following described procedures are required to protect AACS pre-recorded video content.

- Content Revocation
- Content Encryption and Decryption
- Uses of On-line Connections
- Managed Copy
- Sequence Keys

This document is provided as a detailed description of procedures and data structures that are specific for the use of the AACS technology on Blu-ray Disc Read-Only Media.

1.3 Organization of this Document

This document is organized as follows:

- Chapter 1 provides an introduction and overview.
- Chapter 2 describes Blu-Ray Disc specific procedures related to the revocation of pre-recorded video.
- Chapter 3 describes Blu-Ray Disc specific procedures for the production (encryption) and off-line playback (decryption) of AACS video content on pre-recorded Blu-Ray Read Only Media.
- Chapter 4 describes Blu-Ray Disc specific procedures for the use of content with network transactions.
- Chapter 5 describes Blu-ray Disc specific procedure for the Managed Copy of Pre-recorded Content.

- Chapter 6 describes Blu-ray Disc specific procedure for Sequence Keys.
- Chapter 7 describes clarifications for AACS unencrypted contents.

1.4 Reference

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

AACS LA, Introduction and Common Cryptographic Elements, Revision 0.91

AACS LA, Pre-recorded Video Book, Revision 0.92

Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 1: Basic Format Specifications, version 1.3

Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 2: File System Specifications, version 1.2

Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0

ROM-Mark Outline, version 1.0

KCD-Mark Outline, version 1.0

Digital Transmission Licensing Administrator, Digital Transmission Content Protection Specification Volume 1 Revision 1.4

1.5 Document History

This document version 0.921 supersedes version 0.92 dated December 5, 2007. It contains editorial improvements and various clarifications since the 0.92 version. The major points are:

- A clarification of the order of precedence among the AACS Specification books in Chapter 1
- Clarifications of the coverage by the Content Certificate or the Content Hash Table in Chapter 2
- Clarifications to the Enhanced Title and on-line functions in Chapter 3 and Chapter 4
- Addition of an example of Movie Object for two SKBs in Chapter 6
- Notification of an additional obligation to the player with regard to the on-line function in Annex D.1

1.6 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.

1.7 Terminology

Aligned Unit: An Aligned unit consists of a series of 32 Source Packets.

Block Key: A Block Key is a key to encrypt and decrypt each Aligned unit.

CPS Unit: A CPS Unit is a group of titles, to which the same title key has been assigned.

CPS Unit Key: A CPS Unit Key is a Blu-ray Disc synonym for the Title Key.

CPS Unit Usage file: A CPS Unit Usage file is a Blu-ray Disc synonym for the Title Usage file.

ECC Cluster: An ECC Cluster consists of a series of 32 Physical Sectors.

Hash Unit: A Hash Unit consists of a series of 96 Logical Sectors.

Hash Value: A Hash Value is data, which has been calculated from a byte sequence in a Hash Unit.

Logical Sector: A Logical Sector is a data field in a logical volume. All Logical Sectors in a logical volume shall have the same size.

Reserved: The term “Reserved”, when used to define the syntax of the data structure, indicates that the field may be used for future extensions. All the bits of reserved field in the syntax of data structure shall be set to 0_2. The term “Reserved”, when used to define the meaning of values, indicates that the reserved values may be used for future extensions. The reserved values shall never be used in this version.

Segment Key: A Segment Key is a Blu-ray Disc synonym for the Title Key for SK segment portion.

Source Packet: A Source Packet consists of a Source Packet header and a subsequent MPEG-2 transport packet.

1.8 Abbreviation and Acronyms

BD	Blu-ray Disc
BDMV	Blu-ray Disc Movie
BD-ROM	Blu-ray Disc Read-Only Media
CCI	Copy Control Information
CHT	Content Hash Table
CPS	Content Protection System
ECC	Error Correction Code
MPEG	Moving Picture Experts Group
RMF	ROM-Mark Flag
RMIVI	ROM_Mark_IV_Indicator
VFS	Virtual File System

1.9 About Blu-ray Disc Read-Only Media and ROM-Mark

Blu-ray Disc Read–Only Media has two types of physical media. In this document “BD9” and “BD25” are used to identify these two types of physical media with the following definition.

BD9: Physical media based on ECMA-267 with capacity of 4.7 or 8.5 gigabytes.
BD25: Physical media with capacity of 25.0 or 27.0 gigabytes in one Layer.

ROM-Mark is the method to record the Volume ID data for both BD9 and BD25.

This page is intentionally left blank.

Chapter 2
Details for Content Revocation

2. Introduction

Content revocation requires the Content Certificate that is specified in Chapter 2 of the *Pre-recorded Video Book* of this specification. This chapter describes additional details of content revocation that are specific to the BDMV format.

As described in the *Pre-recorded Video Book*, every hash units of the AV contents in the BDMV format on the disc is hashed, and this hashed value is included in the Content Hash Table. Every part of the Content Hash Table, that corresponds to an AV content file, is then hashed, and this hashed value is included in the unsigned Content Certificate as a Content Hash Table Digest. This unsigned Content Certificate is finally signed by the AACS LA, and this becomes the Content Certificate.

A disc may contain both encrypted contents and unencrypted contents. The Content Certificate, however, shall cover all the Clip AV stream files under “\BDMV\STREAM” directory on the disc, whether they are encrypted or not.

2.1 Content Certificate

In parallel with the “\BDMV” directory, a single Content Certificate shall be stored per physical layer in the “\AACS” directory and in the “\AACS\DUPLICATE” directory. The single-layer disc has a single file named “Content000.cer”, while the dual-layer disc has two files named “Content000.cer” for Layer 0 and “Content001.cer” for Layer 1. Note that the Content000.cer and the Content001.cer are stored on Layer 0 and Layer 1 respectively.

The data format of the Content Certificate is defined in Table 2-1.

Table 2-1 – Content Certificate for BD Pre-recorded Disc

Bit Byte	7	6	5	4	3	2	1	0
0	Certificate Type: 00_{16}							
1	(reserved)							
2 … 5	Total_Number_of_HashUnits							
6	Total_Number_of_Layers							
7	Layer_Number							
8 … 11	Number_of_HashUnits							
12 13	Number_of_Digests							

14 15	Applicant ID
16	(msb) CCSS ID (lsb) Sequence Number 1
17	Sequence Number 1 (msb)
18	Timestamp
19	(lsb) Sequence Number 2
20 21	Minimum CRL Version
22 23	(reserved)
24 25	Length_Format_Specific_Section
26 : 45	Hash_Value_of_MC_Manifest_File
46 : 65	Hash_Value_of_BDJ_Root_Cert
66 67	Num_of_CPS_Unit
68 … 87	Hash_Value_of_CPS_Unit_Usage_File#1
	…
68+(J-1)*20 .. 87+(J-1)*20	Hash_Value_of_CPS_Unit_Usage_File#J
K *(see note below)* : K+7	Content Hash Table Digest #1
…	…
K + (N-1)*8 … K+7 + (N-1)*8	Content Hash Table Digest #N
K+8+(N-1)*8 : K+47+(N-1)*8	Signature Data

*Note: K = 88+(J-1)*20*

Details of each field are defined in the *Pre-recorded Video Book* of this specification with the following exceptions:

- A 4-byte Total_Number_of_HashUnits field indicates the total number of Hash Unites on the disc.
- A 1-byte Total Number of Layers field indicates the total number of layers on the disc.
- A 1-byte Layer_Number field indicates the layer of the disc for which this Content Certificate is created. This filed shall be 0 for “Content 000.cer”, and 1 for “Content001.cer”.
- A 4-byte Number_of_HashUnits field indicates the number of Hash Units on the layer for which this Content Certificate is created.
- A 2-byte Number_of_Digests field indicates the number of Clip AV stream files that have a file size equal to or more than 96 Logical Sectors on the layer for which this Content Certificate is created.
- A 2-byte Applicant ID assigned by AACS LA.
- A 4-byte Content Sequence Number consists of 6-bit Content Certificate Signing Server ID (CCSS ID), 15-bit Timestamp, and 11-bit Sequence Number that is a concatenation of 4-bit Sequence Number 1 and 7-bit Sequence Number 2, and is assigned by AACS LA to uniquely identify the Certified Content amongst that content provider’s content. The combination of the Applicant ID and the Content Sequence Number is referred to as the *Content Certificate ID.* In other words, the Content Certificate ID is a 6-byte number. Timestamp indicates the date when a Content Certificate is signed, and contains a value for the elapsed days from 1st January 2008 with the value 0 representing 1st January 2008. Timestamp values predating 2 February 2008 are reserved, and shall not be used as a timestamp.
- A 2-byte Minimum CRL Version value, assigned by the AACS LA to indicate the minimum Content Revocation List Version number that must accompany the Certified Content.
- A 2-byte Length_Format_Specific_Section that specifies the length of the subsequent Format_Specific_Section. The Format Specific Section for BD includes the subsequent Hash_Value_of_MC_Manifest_File, Hash_Value_of_BDJ_Root_Cert, Num_of_CPS_Unit, and a sequence of Hash_Value_of_CPS_Unit_Usage_Files.
- A 20-byte Hash_Value_of_MC_Manifest_File contains the hash value for the Managed Copy Manifest File as defined in Section 5.3.
- A 20-byte Hash_Value_of_BDJ_Root_Cert contains the hash value for the BD-J Root Certificate as defined in Section 2.3.2.4.
- A 2-byte Num_of_CPS_Unit fields indicates the number of CPS Units on the disc.
- A series of 20-byte Hash_Value_of_CPS_Unit_Usage_Files contains the hash value for the CPS Unit Usage File as defined in Section 2.3.2.2.

2.2 Content Revocation List

In parallel with the “\BDMV” directory, the Content Revocation List (CRL) “ContentRevocation.lst” shall be stored in the “\AACS” directory and in the “\AACS\DUPLICATE” directory.

The data format for the Content Revocation List is defined in Table 2-2 of the *Pre-recorded Video Book* of this specification. Note that in the dual-layer case, the player shall check at least one Certificate ID and if the ID is revoked, the access to any layer of such a disc shall be aborted.

CRL data shall be recorded from the first byte of the file, and the null (00_{16}) padding may be attached after the CRL data in the file for the authoring and the mastering purpose.

2.3 Content Hash Table

2.3.1 Data Structure for Content Hash Table

For each physical layer of BD-ROM, the Content Hash Table (CHT) shall be stored in the “\AACS” directory and in the “\AACS\DUPLICATE” directory. The single-layer disc has a single file named “ContentHash000.tbl”, while the dual-layer disc has two files named “ContentHash000.tbl” for Layer 0 and “ContentHash001.tbl” for Layer 1. Note that the ContentHash000.tbl and the ContentHash001.tbl are stored on Layer 0 and Layer 1 respectively.

The Content Hash Table shall contain an 8-bytes hash value for each hash unit of the Clip AV stream files under “\BDMV\STREAM” directory in the corresponding layer. Detail of the hash calculations are defined in Section 2.3.2 of this specification. Each Clip AV stream file is sequentially divided into hash units from head to tail, and the size of each hash unit is 96 Logical Sectors. Note that the tail portion of each Clip AV stream file, which size is less than 96 Logical Sectors, is omitted from storing of its hash value. If the file size of Clip AV stream file is exactly the multiple of 96 Logical Sectors, there is no tail portion to be omitted from storing. If a Clip AV stream is divided in two and recorded on the both layer, the extents size of each Clip AV stream file on the Layer 0 shall be exactly the multiple of 96 Logical Sectors, and the extents of each Clip AV stream file on the Layer 1 shall be logically recorded after the extents of the corresponding Clip AV stream on the Layer 0. Note that the size of CHT is zero byte if there is no Clip AV stream that have a file equal to or more than 96 Logical Sectors on the corresponding layer.

Table 2-2 shows the data structure for Content Hash Table.

Table 2-2 Data Format for Content Hash Table

Syntax	No. of bits	Mnemonics
Content Hash Table {		
for(I=0 ; I < Number_of_Digests ; I++) {		
Starting_HU_Num#I	32	uimsbf
Clip_Num#I	32	uimsbf
HU_Offset_in_Clip#I	32	uimsbf
}		
for(I=0 ; I < Number_of_HashUnits ; I++){		
Hash_Value#I	64	bslbf
}		
}		

Starting_HU_Num#I (4 bytes) indicates the position in hash units of the first hash value of Clip AV stream file #I that have a file size equal to or more than 96 Logical Sectors in the hash value part in this table. This number is starting from zero.

(Note) In case of dual-layer disc, Starting_HU_Num#0 in the ContentHash001.tbl is equal to the Number_of_HashUnits on Layer 0. Refer to the example in Figure 2-2.

Clip_Num#I (4 bytes) indicates 5-digit number included in the file name of Clip AV stream file #I that have a file size equal to or more than 96 Logical Sectors. This value is stored in the ascending order of the 5-digit number included in the file name of the corresponding Clip AV stream file.

HU_Offset_in_Clip#I (4 bytes) indicates the offset in hash units from the top of Clip AV stream file #I that have a file size equal to or more than 96 Logical Sectors. This offset is starting from zero. The hash value at the Starting_HU_Num#I corresponds to the AV data at this offset in the Clip AV stream file #I.

Hash_Value#I (8 bytes) contains the hash value calculated from the hash unit #I in the layer corresponding to this Content Hash Table. These Hash_Value#I shall be listed in the ascending order of the 5-digit number included in the file name of the corresponding Clip AV stream file, and in the ascending order of the logical position in the Clip AV stream file.

Number_of_Digests is defined in Table 2-1, and indicates the number of Clip AV stream files in the layer corresponding to this Content Hash Table.

Number_of_HashUnits is defined in Table 2-1, and indicates the number of hash units in the layer corresponding to this Content Hash Table.

Content Hash Table Digest #J defined in Table 2-1 is the digest of the concatenation of the hash values from the Starting_Hash_Unit_Num#I to Starting_Hash_Unit_Num#(I+1) – 1.

Figure 2-1 shows the example of the relation between Content Certificate and Content Hash Tables.

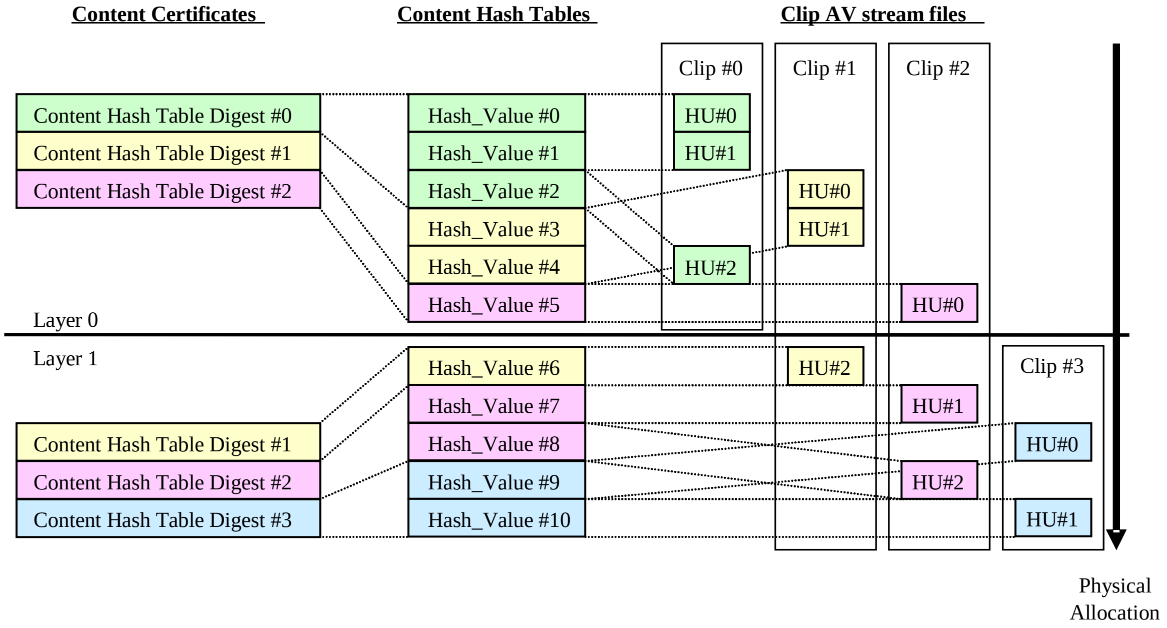


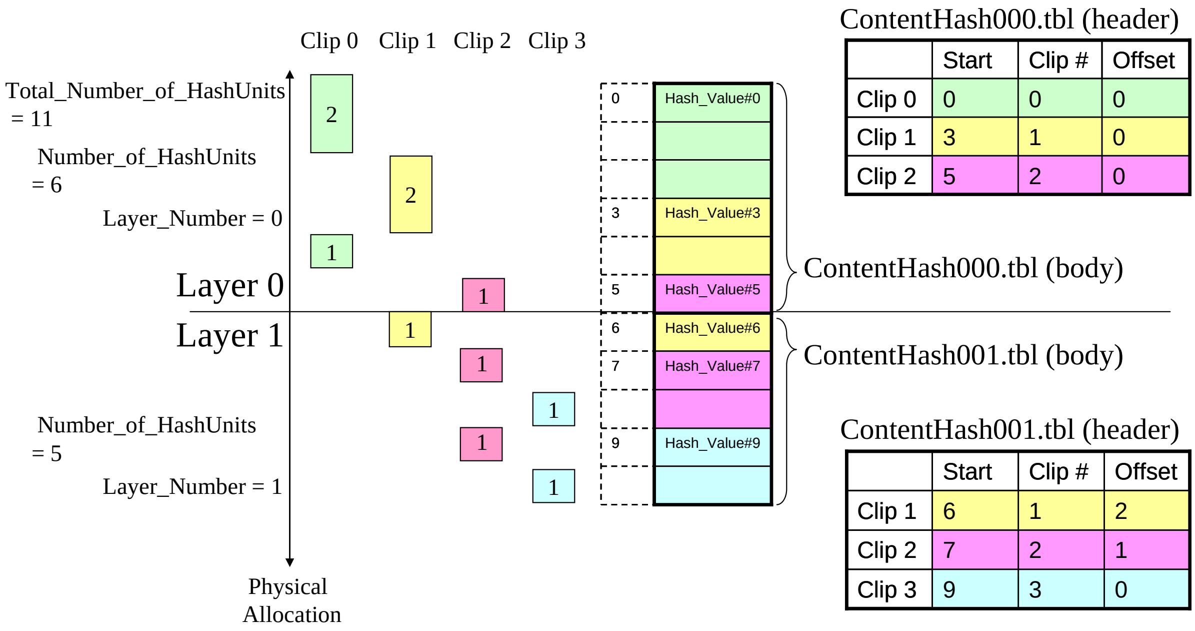
Figure 2-1 Example of the relation between Content Hash Table Digest and Hash Value

In this case, there is one Content Certificate for each layer, one Content Hash Table for each layer, and four Clip AV stream files that have a file size equal to or more than 96 Logical Sectors. The whole part of Clip AV stream file #0 is recorded on Layer 0, and the whole part of Clip AV stream file #3 is recorded on Layer 1. Each Clip AV stream file #1 and #2 are recorded separately on both Layer 0 and 1. From a physical allocation point of view, each Clip AV stream file is fragmented and the file extents of different Clip AV stream files are recorded alternately.

In this case, Content Hash Table for Layer 0 includes Hash_Values for Hash Units of Clip AV stream file #0, #1 and #2. Content Hash Table for Layer 1 includes Hash_Values for Hash Units of Clip AV stream file #1, #2 and #3. Note that Hash_Values for Hash Unit #0 and #1 for Clip AV stream file #1 and Hash Unit #0 for Clip AV stream file #2 are included only in the Content Hash Table for Layer 0.

To calculate the Content Hash Table Digest of each layer, only the Hash_Values in the same layer are used. For example, to calculate the Content Hash Table Digest #1 for Layer 0 in Figure 2-1, Hash_Value #3 and #4 in the Content Hash Table for Layer 0 are used. Hash_Value #0 in the Content Hash Table for Layer 1 is not used.

Figure 2-2 shows the example of the Content Hash Table syntax defined in Table 2-2.



ContentHash000.tbl (header)

	Start	Clip #	Offset
Clip 0	0	0	0
Clip 1	3	1	0
Clip 2	5	2	0

ContentHash001.tbl (header)

	Start	Clip #	Offset
Clip 1	6	1	2
Clip 2	7	2	1
Clip 3	9	3	0

Figure 2-2 Example of the Content Hash Table syntax

2.3.2 Hash Calculation

2.3.2.1 Clip AV stream

A hash value for each hash unit of Clip AV stream is calculated using the SHA-1 hashing function as defined in the below equation. If the data is encrypted, the encrypted data itself is used as an input to the hashing function, so that the player needs not to decrypt the data before calculating a hash value. The stored hash value is the least significant 64 bits of the result for SHA-1 hashing function.

$$\text{Hash\_Value} = [\text{SHA-1}(\text{Hash\_Unit})]_{\text{lsb\_64}}$$

Where SHA-1 is the SHA hashing function as defined in *Introduction and Common Cryptographic Elements* book of this specification.

2.3.2.2 Usage Rule

A hash value for each CPS Unit Usage File is also calculated using the SHA-1 hashing function as defined in the below equation.

Hash_Value_of_CPU_Unit_Usage_File = SHA-1(CPS Unit Usage File)

Hash_Value_of_CPU_Unite_Usage_File is used to verify the integrity of the CPS Unit Usage File.

2.3.2.3 Managed Copy Manifest File

A hash value for the Managed Copy Manifest File is also calculated using the SHA-1 hashing function as defined in the below equation.

Hash_Value_of_MC_Manifest_File = SHA-1(Managed Copy Manifest File)

Hash_Value_of_MC_Manifest_File is used to verify the integrity of the Managed Copy Manifest File. If Managed Copy Manifest File is not recorded on the BD-ROM, Hash_Value_of_MC_Manifest_File shall be set to all zero.

2.3.2.4 BD-J Root Certificate

A hash value for the BD-J Root Certificate (\CERTIFICATE\app.discroot.crt) for application authentication is also calculated using the SHA-1 hashing function as defined in the below equation.

Hash_Value_of_BDJ_Root_Cert = SHA-1(BD-J Root Certificate)

Application Authentication Data is used to verify the integrity of the Application. For the application authentication, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0*. Hash_Value_of_BDJ_Root_Cert is stored in the Content Certificate as defined in Table 2-1.

If BD-J Root Certificate is not recorded on the BD-ROM, Hash_Value_of_BDJ_Root_Cert shall be set to all zero.

2.3.3 Verifying Content Certificate

The licensed product to play back a BDMV shall verify the content certificate as defined in Section 2.6 of the *Pre-recorded Video Book* of this specification. This subsection provides the additional detail for BDMV format.

2.3.3.1 Clip AV stream

If the license product selects type a) as defined in procedure 1 of Section 2.6 of the *Pre-recorded Video Book* of this specification, seven Hash Units shall be randomly selected from the all Hash Units recorded on the BD-ROM.

If the license product selects type b) as defined in procedure 1 of Section 2.6 of the *Pre-recorded Video Book* of this specification, the Hash Unit, which is firstly read from the BD-ROM for each Title, shall be verified. During the playback of each Title, at least 1% of Hash Units recorded on the BD-ROM shall be randomly selected and verified.

As an authoring guideline, it is strongly recommended to prepare at least 3-second non-media-access segment within the first 300 second of title play back. Non-media-access segment is the segment where a player needs not to access any data on the media. Still picture presentation with pause is one example of non-access segment.

2.3.3.2 Usage Rule

The licensed product shall verify the Hash_Value_of_CPS_Unit_Usage_File for a CPS Unit to be played back.

2.3.3.3 Managed Copy Manifest File

If the licensed product uses (reads) the Managed Copy Manifest File for the purpose of Managed Copy, it shall verify the Hash_Value_of_MC_Manifest_File for a BD-ROM with the Managed Copy Manifest File.

2.3.3.4 BD-J Root Certificate

The licensed product shall verify the Hash_Value_of_BDJ_Root_Cert for a BD-ROM with a BD-J Root Certificate.

Chapter 3
Details for Content Encryption and Decryption

3. Introduction

The general approach for encryption and decryption of pre-recorded video content protected by AACS is specified in Chapter 3 of *Pre-recorded Video Book* of this specification. This chapter describes additional details of that approach that are specific to the use of AACS encryption with BD-ROM disc and Application Format.

3.1 Media Key Block

Each BD-ROM disc that contains content encrypted by AACS [using a CPS Unit Key that is provided in the AACS directory] shall include two Read-Only Media Key Blocks (MKB). The MKB “MKB_RO.inf” shall be stored in the “\AACS” directory and in the “\AACS\DUPLICATE” directory.

MKB data shall be recorded from the first byte of the file, and the null (00_{16}) padding may be attached after the MKB data in the file for the authoring and the mastering purpose.

(Note) The Read/Write MKB is mandatory for BD-ROM disc. The Read/Write MKB “MKB_RW.inf” for recorder shall be stored in the “\AACS” directory and in the “\AACS\DUPLICATE” directory.

3.2 Control Data Zone of BD9 Media

Control Data Zone of AACS-protected BD9 media is defined as shown in Figure 3-1.

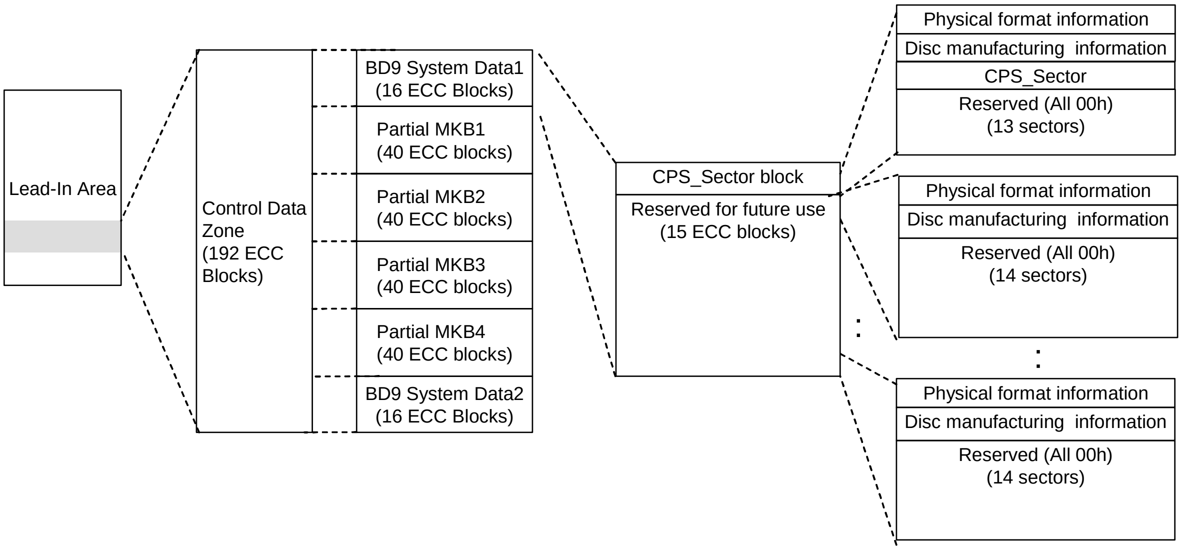

Lead-In Area
Control Data Zone (192 ECC Blocks)
BD9 System Data1 (16 ECC Blocks)
Partial MKB1 (40 ECC blocks)
Partial MKB2 (40 ECC blocks)
Partial MKB3 (40 ECC blocks)
Partial MKB4 (40 ECC blocks)
BD9 System Data2 (16 ECC Blocks)
CPS_Sector block
Reserved for future use (15 ECC blocks)
Physical format information
Disc manufacturing information
CPS_Sector
Reserved (All 00h) (13 sectors)
Physical format information
Disc manufacturing information
Reserved (All 00h) (14 sectors)
Physical format information
Disc manufacturing information
Reserved (All 00h) (14 sectors)

Figure 3-1 Control Data Zone of AACS-protected BD9 Media

Control Data Zone is divided into six areas: BD9 System Data1 (16ECC blocks), Partial MKB1 (40ECC blocks), Partial MKB2 (40ECC blocks), Partial MKB3 (40ECC blocks), Partial MKB4 (40ECC blocks), and BD9 System Data2 (16ECC blocks). BD9 System Data1 and BD9 System Data2 shall have the same data to duplicate the contents of BD9 System Data. Partial MKB1, Partial MKB2, Partial MKB3, and Partial MKB4 shall have the same data to record Partial Media Key Block 4 times.

As defined in ECMA 267 format specification, "Physical format information" and "Disc manufacturing information" are recorded respectively in the first sector and the second sector of all ECC blocks in BD9 Control Data Zone.

Both BD9 System Data1 and BD9 System Data2 consist of 16 ECC blocks. The first ECC block of both BD9 System Data1 and BD9 System Data2 has CPS_Sector at its third sector, and other sectors in this ECC block are reserved. The second ECC block to the last ECC block of Both BD9 System Data1 and BD9 System Data2 are reserved for future use, and has non-specified 14 sectors in each ECC block. The content of CPS_Sector is defined in 3.3.1.

Partial MKB1, Partial MKB2, Partial MKB3, and Partial MKB4 consist of 40 ECC blocks. The data structure of Partial MKB1, Partial MKB2, Partial MKB3, and Partial MKB4 is defined in 3.4.2.

3.3 Volume Identifier

For purpose of encryption and decryption of the content, the Volume Identifier (ID_v) is combined with the Media Key (K_m) to produce the Volume Unique Key (K_{vu}) as follows:

$$K_{vu} = \text{AES-G}(K_m, ID_v)$$

The Volume Identifier shall be stored in a manner that cannot be duplicated by consumer recorders. For BD–ROM, the Volume Identifier shall be stored in the ROM-Mark of the BD-ROM disc. For the details of the ROM-Mark, refer to *ROM-Mark Outline, version 1.0.*

Table 3-1 shows the data format for the Volume Identifier that is stored in the payload of the ROM-Mark.

Table 3-1 Data Format for Volume Identifier

Bit Byte	7	6	5	4	3	2	1	0
0	(msb) Volume Identifier (lsb)							
:								
15								

3.3.1 CPS_Sector

For BD25 Media, the last sector in the first Physical Cluster of each Info Fragment in the PIC zone (Permanent Information & Control Data zone) is reserved as a CPS_Sector.

The other sectors in the first Physical Cluster of each Info Fragment are reserved for storing Disc Information and other information. For the details of the PIC zone, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 1: Basic Format Specifications, version 1.3.*

For BD9 Media, the first ECC block in the Control Data Zone in the Lead-In is used to record CPS_Sector. The data structure of CPS_Sector is the same as BD25 case. The data structure inBD9 Control Data Zone is defined in section 3.2.

The ROM-Mark Flag (RMF) and the ROM_Mark_IV_Indicator (RMIVI) are stored in the top of the CPS_Sector. Table 3-2 shows the data format for CPS_Sector.

Table 3-2 Data Format for CPS_Sector

Bit **Byte**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	RMF	RMIVI			(reserved)			
1	(reserved)							
:								
2047								

The ROM-Mark Flag indicates whether a ROM-Mark is stored on the disc or not. **Table 3-3** defines the meaning of ROM-Mark Flag.

Table 3-3 ROM-Mark Flag

ROM-Mark Flag	**Meaning**
0_2	No ROM-Mark is stored on the disc
1_2	A ROM-Mark is stored on the disc

The ROM_Mark_IV_Indicator indicates which value of ROM_Mark_IV is used for the ROM-Mark detection. Table 3-4 defines the value and meaning of this field. This field shall be set to 000_2.

Table 3-4 ROM_Mark_IV_Indicator

ROM_Mark_IV_Indicator	**Meaning**
000_2	The ROM-Mark detector IV values shall be used as ROM_Mark_IV
001_2 - 101_2	Reserved for BD-CPS
other	Reserved

3.4 Partial Media Key Block for Host Revocation List

The Host Revocation List is stored as “Partial Media Key Block” in the Lead-in area of disc. Partial Media Key Block consists of “Type and Version Record” and “Host Revocation List Record”.

This section defines the structure of Partial Media Key Block and other requirement for Partial Media Key Block recording on BD-ROM Media.

Table 3-5 shows the data format for the Partial Media Key Block.

The Partial Media Key Block shall be stored as 64KB units with zero padding.

(Note 1) The maximum size of reserved area for Partial Media Key Block on BD-ROM Media is one megabyte.

Table 3-5 Partial Media Key Block Format

Bit **Byte**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	Type and Version Record							
…								
11								
12	Host Revocation List Record							
13								
14								
…								
X								

The BD drive is required to store the Partial Media Key Block in its non-volatile memory. The Host Revocation List Record is required to be stored in the non-volatile memory of the drive consists of the data being signed for the first signature block including the Signature for Block 1. The details of the Type and Version Record and the Host Revocation List Record are defined in Section 3.2.5 of the *Introduction and Common Cryptographic Elements* book of this specification.

(Note 2) For the BD Prerecorded Disc, the drive shall handle the disc as AACS compliant media, if the Partial Media Key Block is recorded on the BD-ROM.

The behavior for drive is as follows:

- In case that the drive cannot verify and read the Partial Media Key Block on the media for some reason, the drive shall read the Partial Media Key Block stored in non-volatile memory of the drive and use it for the authentication process.

3.4.1 Partial Media Key Block for Host Revocation List for BD25 Media

For BD25 Media, the Partial Media Key Block shall be stored in the PIC zone in Inner Zone 0 of the BD-ROM disc. Note that the PIC zone (Permanent Information & Control Data Zone) shall consist of 5 repetitions of a PIC Info Fragment. The Partial Media Key Block shall be written 5 times and shall begin from cluster 1, i.e. AUN 00B9220h, 00BFC20h, 00C6620h, 00CD020h, 00D3A20h. In case of multiple-layer disc, each PIC zone shall have Partial Media Key Block with the same way as single layer. The details of the PIC are described in *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 1: Basic Format*

Specifications, version 1.3.

3.4.2 Partial Media Key Block for Host Revocation List for BD9 Media

For BD9 Media, the Partial Media Key Block shall be stored in Control Data Zone of BD9 Media Lead-In area.

Figure 3-2 describes the structure of BD9 Lead-In and the recording method of Partial Media Key Block. The Partial Media Key Block shall be written 4 times in Partial MKB1~ Partial MKB4 area respectively. Partial MKB1~ Partial MKB4 area begin at ECC block number 17, 57, 97, 137. Each ECC block has 14 sectors that can be used to store the Partial Media Key Block information. All unused sectors shall be filled with 00h.

The details of the Lead-In area of BD9 Media are described in *ECMA-267 Format*.

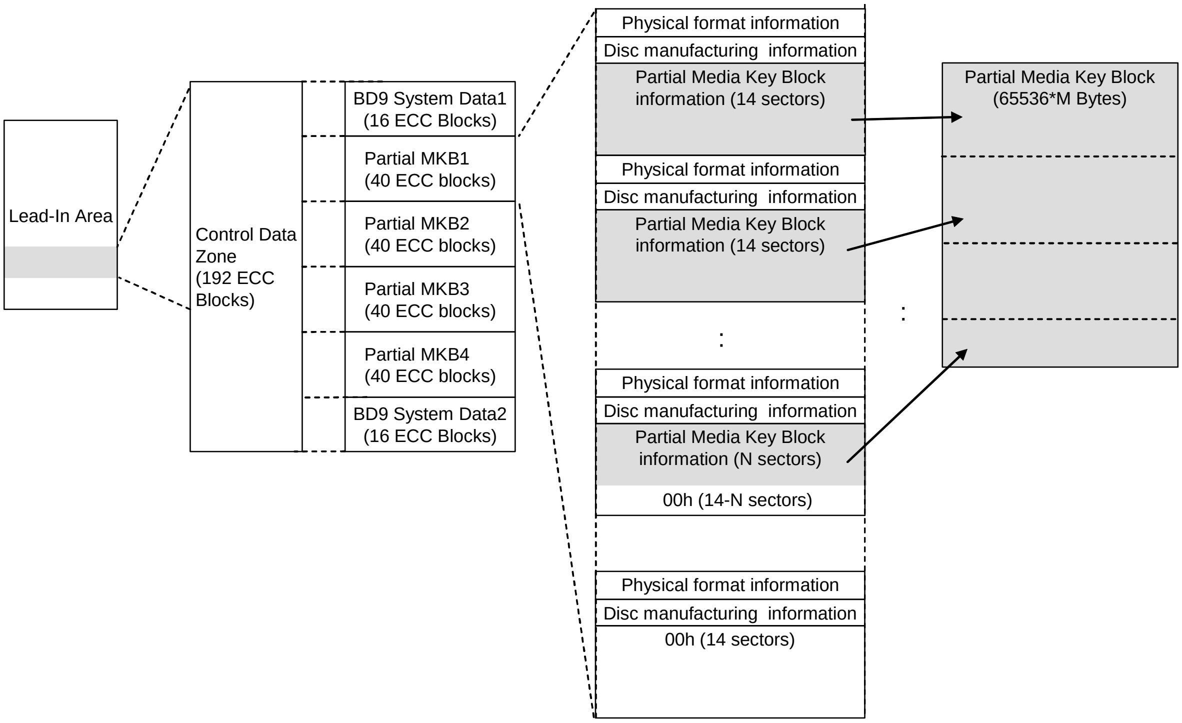

Lead-In Area
Control Data Zone (192 ECC Blocks)
BD9 System Data1 (16 ECC Blocks)
Partial MKB1 (40 ECC blocks)
Partial MKB2 (40 ECC blocks)
Partial MKB3 (40 ECC blocks)
Partial MKB4 (40 ECC blocks)
BD9 System Data2 (16 ECC Blocks)
Physical format information
Disc manufacturing information
Partial Media Key Block information (14 sectors)
Physical format information
Disc manufacturing information
Partial Media Key Block information (14 sectors)
:
Physical format information
Disc manufacturing information
Partial Media Key Block information (N sectors)
00h (14-N sectors)
Physical format information
Disc manufacturing information
00h (14 sectors)
:
Partial Media Key Block (65536*M Bytes)

Figure 3-2 Partial Media Key Block recording in AACS-protected BD9 Media

3.5 CPR_MAI in Content Provider Information Sectors of BD9 Media

Table 3-6 describes the data format of CPR_MAI (6bytes) in Content Provider Information of BD9 Media.

CPR_MAI_Byte1 (=10_{16}) indicates that the media is AACS-protected BD9 Media.

Table 3-6 Data Format for CPR_MAI in Content Provider Information of BD9 Media

Bit Byte	7	6	5	4	3	2	1	0
0	CPR_MAI_Byte1 = 10_{16}							
1	CPR_MAI_Byte2 = 00_{16}							
2	(reserved)							
:								
5								

3.6 Pre-Recorded Media Serial Number

For purpose of using On-line Connections, the Pre-Recorded Media Serial Number is defined and is used for generating the MAC of PMSN. The Pre-Recorded Media Serial Number is optional for BD-ROM disc. For BD-ROM, the Pre-Recorded Media Serial Number shall be stored in the BCA record of the BD-ROM disc.

Player shall use 128bits value in a Data Unit as a Pre-recorded Media Serial Number, where the first 8 bits of the value is set to 00000100_2.

Table 3-7 shows the data format for the Pre-Recorded Media Serial Number that is stored in BCA.

Table 3-7 Data Format for BCA Record for Pre-Recorded Media Serial Number

Bit Byte	7	6	5	4	3	2	1	0
0	Content Code = 000001_2						Data Unit sequence number = 00_2	
1	Applicant ID							
2								
3	(msb)							
:	Unique Value							
15								(lsb)

Each device shall use (from the Content Code to the Unique Value) a 128-bit Pre-recorded Media Serial Number.

Content Code field (6 bits) indicates the application identifier, and shall be set to 000001_2.

Data Unit sequences number field (2 bits) indicates the data unit sequence number, and shall be set to 00_2 for Pre-recorded Media Serial Number.

Applicant ID (16 bits) shall contain the applicant identifier assigned to each replicator by the AACS LA.

Unique Value field (104 bits) shall be assigned a unique value for each disc by each replicator.

3.7 Bus Encryption Flag

The Bus Encryption Flag (BEF) is used to indicate whether the sector data shall be encrypted in the interface bus between the PC Drive and the PC Host or not. If the BEF is set to 1b, the corresponding sector shall be encrypted in the interface bus in the manner that is to be later specified.

For BD25 Media, the Bus Encryption Flag shall be stored in the User Control Data associated with the corresponding sector.

Table 3-8 shows the data format for the Bus Encryption Flag (1 bit) which is recorded in User Control Data of BD ROM disc.

Table 3-8 Data Format for Bus Encryption Flag in User Control Data

Bit Byte	7	6	5	4	3	2	1	0
0	BEF	(reserved)						
1	(reserved)							
2								
:								
17								

For BD9 Media, the Bus Encryption Flag shall be stored in CPR Sector Header associated with the corresponding sector.

Table 3-9 shows the data format for the Bus Encryption Flag (1 bit) which is recorded in CPR_MAI in Data Area.

Table 3-9 Data Format for Bus Encryption Flag in Sector Header

Bit Byte	7	6	5	4	3	2	1	0
0	BEF	(reserved)						
1	(reserved)							
2								
:								
5								

3.8 Key Conversion Data

Note that for certain classes of devices, processing of the Media Key Block will result in a *Media Key Precursor* K_{mp} instead of a Media Key. These classes of devices are defined in the AACS license. After they calculate the Media Key Precursor, they must combine it with *Key Conversion Data* (KCD), to obtain the actual Media Key using the following process:

For certain classes of devices, the Key Conversion Data (KCD) is combined with the Media Key Precursor (K_{mp}) to produce the Media Key (K_m) as follows:

$$K_m = \text{AES-G}(K_{mp}, \text{KCD})$$

The Key Conversion Data shall be stored in a manner that cannot be read by open platform drive. For BD–ROM, the Key Conversion Data shall be stored in the KCD-Mark of the BD-ROM disc. For the details of the KCD-Mark, refer to *KCD-Mark Outline, version 1.0*.

Table 3-10 shows the data format for the Key Conversion Data that is stored in the payload of the KCD-Mark.

Table 3-10   Data Format for Key Conversion Data

Bit **Byte**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	(msb) Key Conversion Data (lsb)							
:								
15								

3.9 CPS Unit Key File and CPS Usage File

3.9.1 Application Format Structure

Figure 3-3 describes a simplified diagram of the BD-ROM application format.

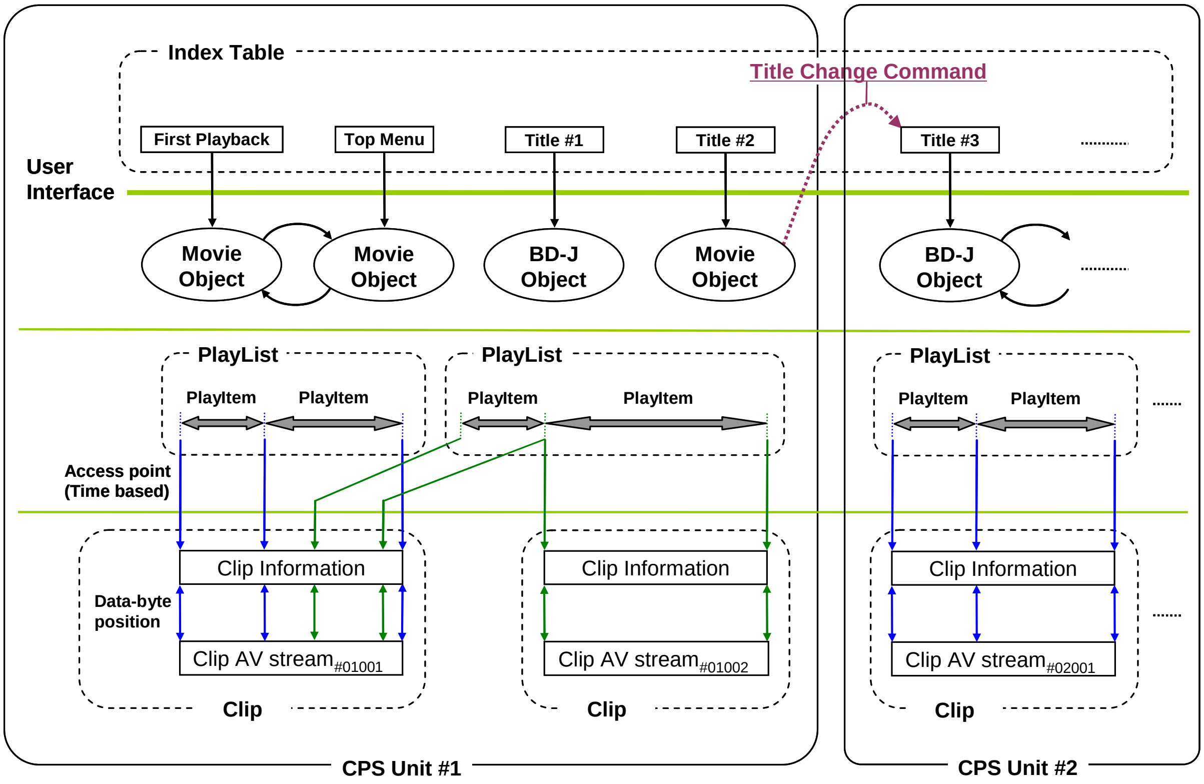


Figure 3-3 Application Format Structure and CPS Unit

This application format has four layers for managing AV stream files: those are Index Table, Movie Object, PlayList and Clip.

3.9.1.1 Clip

Each pair of an AV stream file and its attribute is considered to be one object. A Clip is an object consisting of a Clip AV stream file and its corresponding Clip information file. A Clip AV stream file stores data, which is basically an MPEG-2 transport stream defined in a structure conforming to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0*. The Clip Information file stores the time stamps of the access point into the corresponding AV stream file. The Player reads the Clip Information to find out the position where it should begin to read the data from the AV stream file.

3.9.1.2 PlayList

A PlayList is a collection of playing intervals in the Clips. One such playing interval is called a PlayItem and consists of a pair of pointers called: IN-point and OUT-point. This pair points to positions on a time axis of the Clip. Therefore, a PlayList is a collection of PlayItems. Here the IN-point means a start point of a playing interval, and the OUT-point means an end point of the playing interval.

3.9.1.3 Movie Object

A Movie Object consists of an executable navigation command program. This enables "dynamic scenario description". Movie Objects are a layer above PlayLists. A navigation command in a Movie Object can launch a PlayList playback or a Move Object can call another Movie Object so that a set of Movie Objects can manage playback of PlayLists in accordance with user's interaction and preferences.

3.9.1.4 BD-J Object

A BD-J Object consists of a table of BD-J applications and indicates a set of BD-J Applications. This also enables dynamic scenario description and interactive contents playback by use of the Java programming environment. BD-J Objects are at the same layer of Movie Object, and selected per title basis. BD-J Applications in BD-J Object provides on-line functionality not only for the corresponding Title but also for the whole BD-ROM disc.

3.9.1.5 Index Table

Index Table is top-level information of the application format. This table contains entry points for all Titles, First Playback, and Top Menu. The Player references this table whenever a Title, First Playback, or Menu executing operation needs to be performed.

3.9.1.6 First Playback

First Playback may be optionally defined in the Index Table and points to a Movie Object or a BD-J Object, which is played automatically when the disc is loaded. When the disc is loaded, the player refers to the entry of "First Playback" and obtains the corresponding Movie Object or BD-J Object. First Playback Movie Object / BD-J Object is an optional function. A disc may or may not contain First Playback Movie Object / BD-J Object.

3.9.1.7 Top Menu

Top Menu may be optionally defined in the Index Table and points to a Movie Object or a BD-J Object. This is called by a user operation such as a "MenuCall". A Movie Object indexed by Top Menu executes a PlayList whose PlayItem links a Clip having Button Objects. Each Button Object branches off to another Movie Object as a child Menu. Top Menu Movie Object is an optional function. A disc may or may not contain Top Menu Movie Object.

3.9.1.8 Title

Title is a logical unit for the user to recognize one playback group. The group may be one linear playback block or it may be a non-linear playback block with branching points. Each Title has a title_number. title_number values are defined in ascending order, starting from one. All the values of title_number, no more than the total number of titles, shall be defined at least once on a disc.

3.9.2 CPS Unit

A CPS Unit is a group of a First Playback, a Top Menu, and/or Titles, which are encrypted by using the same Unit Key (Kcu). Each CPS Unit has its corresponding CPS Unit Usage file. Each CPS Unit has a CPS_Unit_number. CPS_Unit_number values are defined in ascending order, starting from one. So, the maximum value of CPS_Unit_number shall be the same as the number of CPS Units that are assigned to First Playback, Top Menu, and/or Titles. And All CPS_Unit_number from one up to the maximum CPS_Unit_number shall be used at least once.

All AV stream files that are referred to by First Playback are included in the same CPS Unit, i.e. they are encrypted by using the same Unit Key. All AV stream files that are referred to by Top Menu are included in the same CPS Unit, i.e. they are encrypted by using the same Unit Key. All AV stream files that are referred to by one Title are included in the same CPS Unit, i.e. they are encrypted by using the same Unit Key. If First Playback, Top Menu and/or a Title share one or more Clips, they shall be included in the same CPS Unit, i.e. the same Unit Key shall be assigned to First Playback, Top Menu and/or the Title. If multiple Titles share one or more Clips, these Titles shall be included in the same CPS Unit, i.e. the same Unit Key shall be assigned to these Titles. First Playback may or may not be included in the same CPS Unit with Top Menu, a Title, and/or Titles. Top Menu may or may not be included in the same CPS Unit with one or more Titles.

For example in Figure 3-3, since a First Playback, a Top Menu, and two Titles commonly refer to the same Clip AV stream$_{\#01001}$, they belong to the same CPS Unit #1. Both Clip AV stream$_{\#01001}$ and Clip AV stream$_{\#01002}$ shall be encrypted by using the same key Kcu1.

To achieve higher security and future flexibility, different keys shall be assigned to different CPS Units. For example, Figure 3-3 shows different keys, Kcu1 and Kcu2, that are assigned to CPS Unit #1 and CPS Unit #2. In this case, the switching between different CPS Units can be executed by some commands for Title change (e.g. Jump Title, Call Title, etc.) defined in *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0*.

Figure 3-4 and Figure 3-5 show the directory structure of the BD-ROM application format. Detailed information is described in the chapter “Directories and Files” in *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0*.

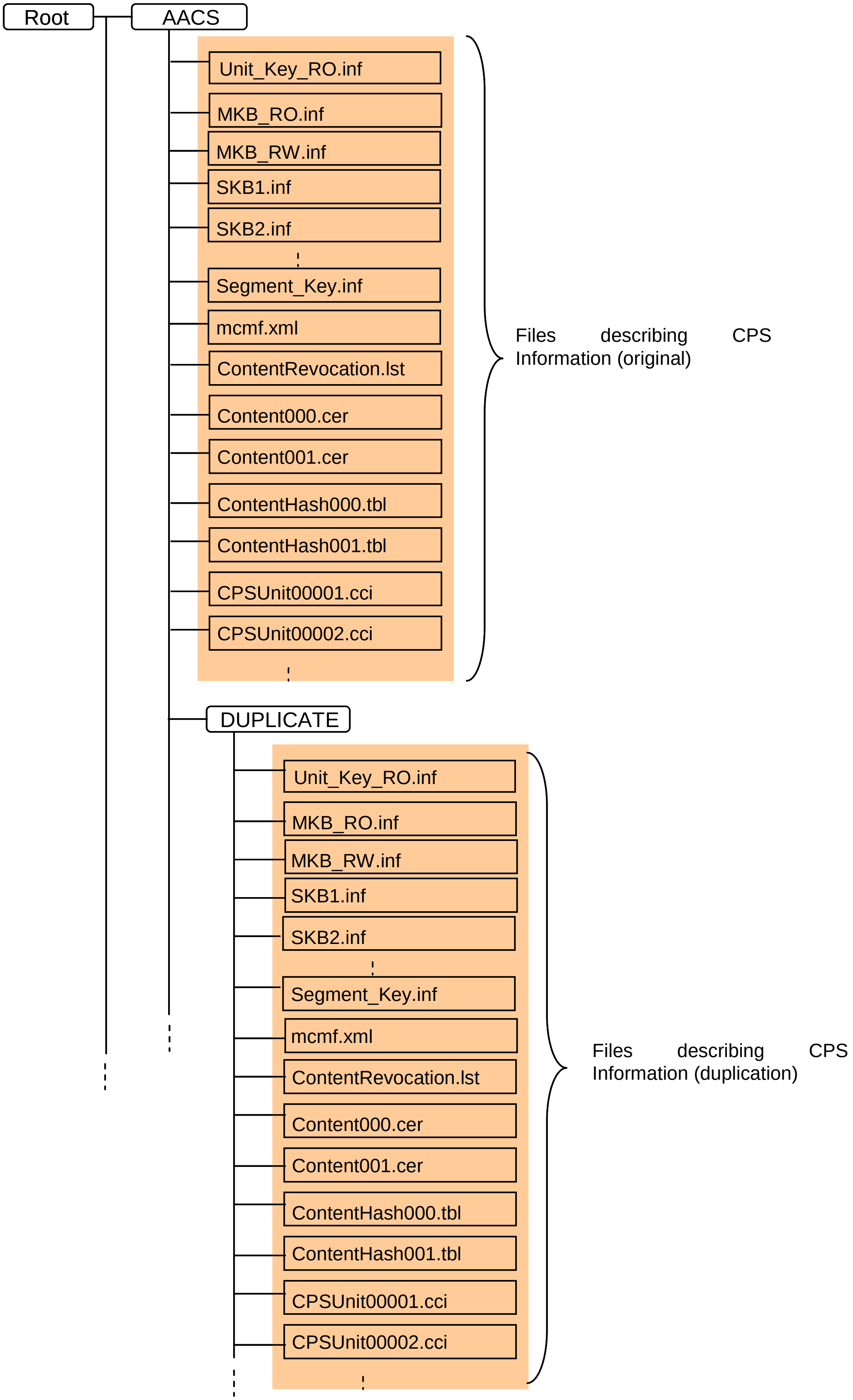

Root
AACS
Unit_Key_RO.inf
MKB_RO.inf
MKB_RW.inf
SKB1.inf
SKB2.inf
Segment_Key.inf
mcmf.xml
ContentRevocation.lst
Content000.cer
Content001.cer
ContentHash000.tbl
ContentHash001.tbl
CPSUnit00001.cci
CPSUnit00002.cci
Files describing CPS Information (original)
DUPLICATE
Unit_Key_RO.inf
MKB_RO.inf
MKB_RW.inf
SKB1.inf
SKB2.inf
Segment_Key.inf
mcmf.xml
ContentRevocation.lst
Content000.cer
Content001.cer
ContentHash000.tbl
ContentHash001.tbl
CPSUnit00001.cci
CPSUnit00002.cci
Files describing CPS Information (duplication)

Figure 3-4 Directory structure for AACS directory

DUPLICATE directory contains the duplication of CPS information files and is used when these files in \AACS directory cannot be read. File name and the file data of the duplicated CPS files shall be the same as original CPS files. The location of the file data of duplicated CPS files should be physically far from the location of the file data of original CPS files.

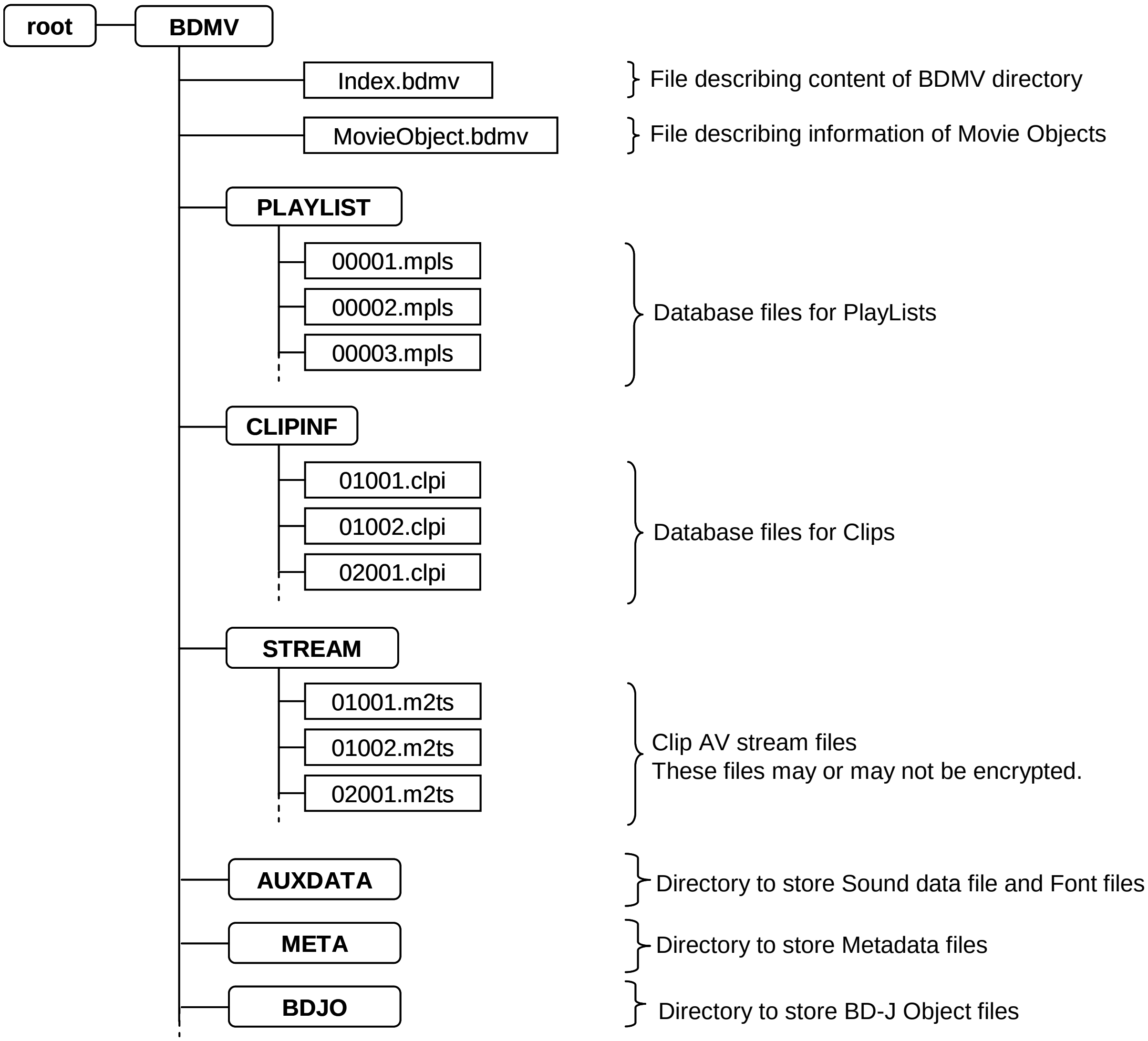

root
BDMV
Index.bdmv
File describing content of BDMV directory
MovieObject.bdmv
File describing information of Movie Objects
PLAYLIST
00001.mpls
00002.mpls
00003.mpls
Database files for PlayLists
CLIPINF
01001.clpi
01002.clpi
02001.clpi
Database files for Clips
STREAM
01001.m2ts
01002.m2ts
02001.m2ts
Clip AV stream files
These files may or may not be encrypted.
AUXDATA
Directory to store Sound data file and Font files
META
Directory to store Metadata files
BDJO
Directory to store BD-J Object files

Figure 3-5 Directory structure for BDMV directory

Clip AV stream files under "\BDMV\STREAM" directory may be encrypted as described in Section 3.10.1. No other files under AACS directory or BDMV directory shall be encrypted using the scheme described in Section 3.10.1.

3.9.3 CPS Unit Key File (Unit_Key_RO.inf)

Each CPS Unit on the BD-ROM disc that is encrypted by AACS has a unique CPS Unit Key. All CPS Unit Keys on one disc shall be stored in the CPS Unit Key File "Unit_Key_RO.inf" in the "\AACS" directory and in the "\AACS\DUPLICATE" directory.

The following requirement is applied to the CPS Unit Key File to reserve enough size of continuous area for the CPS Unit Key File.

- The size of CPS Unit Key File shall be multiple of 65536 bytes.

Table 3-11 shows the data structure for CPS Unit Key File.

Table 3-11 Data Format of CPS Unit Key File for BDMV Application

Syntax	No. of bits	Mnemonic
CPS Unit Key File {		
Unit_Key_Block_start_address	32	uimsbf
Reserved for future use	96	bslbf
Unit_Key_File_Header()		
For (I=0 ; I<X ; I++){	(*1)	
padding word	16	bslbf
}		
Unit_Key_Block()		
For (J=0 ; J<Y ; J++){	(*2)	
padding word	16	bslbf
}		
}		

(*1) X (size of padding word) shall be such a value less than 16 that Unit_Key_Block() begins at 16-byte boundary.

(*2) Y (size of padding word) shall be such a vaule less than 65536 that the size of CPS Unit Key File becomes multiple of 65536 bytes.

Unit_Key_Block_start_address field (32 bits) indicates the start address of Unit_Key_Block() in the relative byte number from the first byte of CPS Unit Key File. The value of Unit_Key_Block_start_address field shall be a multiple of 16.

Table 3-12 shows the data structure for Unit_Key_File_Header() of CPS Unit Key File.

Table 3-12 Data Format of Unit_Key_File_Header() for BDMV Application

Syntax	No. of bits	Mnemonic
Unit_Key_File_Header(){		
Application_Type (= 01_{16})	8	uimsbf
Num_of_BD_Directory (= 01_{16})	8	uimsbf
Use_SKB_Flag	1	bslbf
(reserved)	15	bslbf
For(I=0; I < Num_of_BD_Directory; I++){		
CPS_Unit_number for First Playback#I	16	uimsbf
CPS_Unit_number for Top Menu#I	16	uimsbf
Num_of_Title#I	16	uimsbf
For(J=1; J < Num_of_Title+1; J++){		
(reserved)	16	bslbf
CPS_Unit_number for Title#J in Directory #I	16	uimsbf
}		
}		
}		

Application Type field (8 bits) indicates the type of AV Application which is used with the CPS Unit Key File. For BDMV Application, the value of Application Type shall be 1 to indicate that the CPS Unit Key File is associated to the BDMV Application and the syntax complies with what is described in Table 3-12.

Num_of_BD_Directory field (8 bits) indicates the number of BD application directories recorded on the media. For BDMV Application, the value of Num_of_BD_Directory shall be 1, because BDMV Application uses only one directory (“\BDMV” directory).

Use_SKB_Flag indicates whether Sequence Key Block is used on the disc or not. Table 3-13 shows the meaning of Use_SKB_Flag.

Table 3-13 Use_SKB_Flag

Use_SKB_Flag	Meaning
0_2	Sequence Key Block is not used on the disc
1_2	Sequence Key Block is used on the disc

CPS_Unit_number for First Playback#I field (16 bits) indicates the CPS Unit number that First Playback belongs to. If First Playback is not on the BD Prerecorded Disc, this field shall be set to 0000_{16}.

CPS_Unit_number for Top Menu#I field (16 bits) indicates the CPS Unit number that Top Menu belongs to. If Top Menu is not on the BD Prerecorded Disc, this field shall be set to 0000_{16}.

Num_of_Title#I field (16 bits) indicates the number of ttitles on the disc.

CPS_Unit_number for Title#J in Directory #I field (16 bits) indicates the CPS Unit number that each Title belongs to.

Table 3-14 shows the data structure for Unit_Key_Block() of CPS Unit Key File for BDMV Application.

Table 3-14 Data Format of Unit_Key_Block() for BDMV Application

Syntax	No. of bits	Mnemonic
Unit_Key_Block(){		
Num_of_CPS_Unit	16	uimsbf
(reserved)	112	bslbf
For(I=1; I < Num_of_CPS_Unit+1; I++){		
MAC of PMSN#I	128	bslbf
MAC of Device Binding Nonce#I	128	bslbf
Encrypted CPS Unit Key for CPS Unit#I	128	bslbf
}		
}		

Num_of_CPS_Unit field (16 bits) indicates the number of CPS Units on the disc.

MAC of PMSN field contains the 16-byte MAC of Pre-Recorded Media Serial Number calculated by using CPS Unit Key for each CPS Unit. The MAC of PMSN is generated as follows:

CMAC(K_{cu}, Pre-recorded Media Serial Number)

If Binding Type defined in Table 3-31 of this specification indicates “Media Binding” or “Device/Media Binding” and a Licensed Player is about to activate such Title, the Licensed Player shall calculate the MAC of PMSN using the PMSN recorded on the media inserted and shall verify the matching between MAC of PMSN in CPS Unit Key File and the MAC value calculated by the Licensed Player. If the verification fails, the Licensed Player shall not start the playback of Titles in the corresponding CPS Unit. Activation of Title is defined in Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0.

(Note) In case that the CPS Unit is not bound to the Pre-Recorded Media Serial Number, the MAC of PMSN field shall be set to all-zero. In other words, this field on the BD-ROM disc is always set to all-zero. Practically, this field is used only in the case that the Virtual File System is used for downloaded content in Binding Unit Data Area of Local Storage. For the Virtual File System, refer to Section 4.1 of this specification.

MAC of Device Binding Nonce field contains the 16-byte MAC of Device Binding Nonce by using CPS Unit Key for each CPS Unit. The Device Binding Nonce shall be 128-bit statistically unique number, and shall be stored in Licensed Player that has AACS On-line capability. Player shall be implemented in a way that the Device Binding Nonce can not be modified by the user.

. The MAC of Device Binding Nonce is generated as follows:

CMAC(K_{cu}, Device Binding Nonce)

If Binding Type defined in Table 3-31 of this specification indicates “Device/Content Binding” or “Device/Media Binding” and a Licensed Player is about to activate such Title, the Licensed Player shall calculate the MAC of Device Binding Nonce stored in the Licensed Player and shall verify the matching between MAC of Device Binding Nonce in CPS Unit Key File and the MAC value calculated by the Licensed Player. If the verification fails, the Licensed Player shall not start the playback of Titles in the corresponding CPS Unit.

(Note) In case that the CPS Unit is not bound to the Device, the MAC of Device Binding Nonce field shall be set to all-zero. In other words, this field on the BD-ROM disc is always set to all-zero. Practically, this field is used only in the case that the Virtual File System is used for downloaded content in Binding Unit Data Area of Local Storage. For the Virtual File System, refer to Section 4.1 of this specification.

Encrypted CPS Unit Key field contains the 16 bytes of the encrypted CPS Unit Key (K_{cu}) for each CPS Unit. The CPS Unit Key is encrypted as follows:

AES-128E(K_{vu}, K_{cu})

where K_{vu} denotes a Volume Unique Key defined in Section 3.3 of the AACS *Pre-recorded Video Book* of this specification.

3.9.4 CPS Unit Usage File (CPSUnitXXXXX.cci)

Each CPS_Unit on the BD-ROM disc that is encrypted by AACS has an associated CPS Unit Usage file. CPS Unit Usage file is the Usage Rules for BD-ROM disc and describes the CCI and related information of each CPS_Unit. Each CPS Unit Usage file associated to a CPS_Unit shall be stored in the “CPSUnitXXXXX.cci” file in the “\AACS” directory and in the “\AACS\DUPLICATE” directory. Here, XXXXX shall be the 5-digit number. XXXXX shall be equal to the CPS Unit number to which the CCI file is associated. The extension shall be “cci”.

Table 3-15 shows the data structure for the CPS Unit Usage File.

Table 3-15 Data Structure for CPS Unit Usage File

Bit Byte	7	6	5	4	3	2	1	0		
0 : 15	Primary Header								16 bytes	2048 bytes
16 : 2047	Primary CCI Area								2032 bytes	
2048 : 2063	Secondary Header								16 bytes	(2048*N) bytes : Option
2064 : 2048*N-1	Secondary CCI Area								(2048*N-16) bytes	

Primary Header (16 bytes) includes the number of CCI loops in the Primary CCI Area.

Primary CCI Area (2032 bytes) includes one or more CCI_and_other_info() blocks.

Secondary Header (16 bytes) includes the number of CCI loops in the Secondary CCI Area.

Secondary CCI Area (2048*N -16 bytes) includes one or more CCI_and_other_info() blocks.

(Note) The data structure after Byte2048 is Option. However, if Secondary CCI Area is used, the structure in Table 3-15 shall be used. The player shall refer to the Primary CCI Area. If the Secondary CCI Area is on the disc, the player may refer to the both CCI Areas.

Table 3-16 shows the syntax for the CPS Unit Usage File.

Table 3-16 Syntax for CPS Unit Usage File

Syntax	No. of bits	Mnemonics	Data Block
CPS Unit Usage File {			-
Number_of_Primary_CCI_loops	16	uimsbf	Primary Header
(reserved)	112	bslbf	
for(I=0; I< Number_of_Primary_CCI_loops; I++){			Primary CCI Area
CCI_and_other_info()			
}			
(reserved)	X (*1)	bslbf	
			-
Number_of_Secondary_CCI_loops	16	uimsbf	Secondary Header
(reserved)	112	bslbf	
for(I=0; I< Number_of_Secondary_CCI_loops; I++){			Secondary CCI Area
CCI_and_other_info()			
}			
(reserved)	Y (*2)	bslbf	
}			-

(*1) X is decided to fill the Primary CCI Area (2032 bytes)

(*2) Y is decided to fill the Secondary CCI Area (2048*N–16 bytes)

Number_of_Primary_CCI_loops indicates the number of CCI_and_other_info() blocks in the Primary CCI Area.

Number_of_Secondary_CCI_loops indicates the number of CCI_and_other_info() blocks in the Secondary CCI Area.

3.9.4.1 CCI_and_other_info( )

CCI_and_other_info() contains CCI and title usage information for each CPS Unit.

Table 3-17 shows the data structure for CCI_and_other_info( ).

Table 3-17 Syntax for CCI_and_other_info( )

Syntax	No. of bits	Mnemonic
CCI_and_other_info() {		
CCI_and_other_info_type	16	uimsbf
CCI_and_other_info_version	16	uimsbf
CCI_and_other_info_data_length	16	uimsbf
CCI_and_other_info_data()	L*8	
}		

CCI_and_other_info_type indicates what type of CCI and related information of a CPS Units is described in CCI_and_other_info_data( ). Table 3-18 shows the bit assignment of CCI_and_other _info_type.

Table 3-18 Bit assignment for CCI_and_other_info_type

CCI_and_other_info_type	Meaning
0000_{16}	Reserved
0001_{16}	Reserved for Basic CCI for BD-CPS
0002_{16}-0100_{16}	Reserved
0101_{16}	Basic CCI for AACS
0102_{16}	Reserved for CCI Sequence Information
0103_{16}-0110_{16}	Reserved
0111_{16}	Enhanced Title Usage for AACS
0112_{16}	Key Management Information for On-line Function
0113_{16}	Content Owner Authorized Outputs Information
0114_{16}-FFFF_{16}	Reserved

Basic CCI for AACS (CCI_and_other_info_type=0101_{16}) is used to describe the basic CCI information for AACS. There shall be exactly one Basic_CCI for AACS on one CPS Unit, and it shall be contained in the Primary CCI Area.

Enhanced Title Usage for AACS (CCI_and_other_info_type=0111_{16}) is used to describe the Enhanced Title Usage information for AACS. Enhanced Title Usage for AACS is not required for Basic Title but is mandatory for the Enhanced Title.

Key Management Information for On-line Function (CCI_and_other_info_type=0112_{16}) is used to describe the Binding type for this CPS Unit. Four binding types are defined in Section 5.5 of the *Introduction and Common Cryptographic Elements* of this specification. If Key Management Information for On-line Function is recorded, it shall be contained in the Primary CCI Area.

CCI_and_other_info_version indicates the version number of CCI_and_other_info_data() for each CCI_and_other_info_type. This value is defined for each CCI_and_other_info_type.

CCI_and_other_info_data_length indicates the byte length of CCI_and_other_info_data() for each CCI_and_other_info_type. This value is defined for each CCI_and_other_info_type.

CCI_and_other_info_data() is the description area for CCI and related information of a CSP Unit. The structure of this field is separately defined for each CCI_and_other_info_type.

The length of the CCI_and_other_info() field in the Primary CCI Area shall be less than or equal to 2012 bytes. The Primary CCI Area may contain multiple different types of CCI_and_other_info().

The Secondary CCI Area may also contain multiple different types of CCI_and_other_info(). The Secondary CCI Area can contain the CCI_and_other_info() that cannot be stored in the Primary CCI Area. When the size of CCI_and_other_info() that is greater than 2012 bytes, the CCI_and_other_info() shall be stored in the Secondary CCI Area.

If there is an unknown (Reserved) CCI_and_other_info_type, player shall ignore this CCI_and_other_info( ).

If there is a higher version of CCI_and_other_info_version than the version supported by player, player shall ignore this CCI_and_other_info( ).

If reserved bits in each CC_and_other_info_data( ) are not set to zero, player shall ignore these bits and only use non-reserved bits.

Note: If the player cannot find the supporting version of Basic CCI for AACS, the player shall not start playback of the contents.

3.9.4.2 Basic CCI for AACS

Table 3-19 shows the data structure of CCI_and_other_info( ) for Basic CCI for AACS.

Table 3-19 Syntax of Basic CCI for AACS

Syntax	No. of bits	Mnemonics
Basic CCI for AACS {		
CCI_and_other_info_type (=0101_{16})	16	uimsbf
CCI_and_other_info_version (=0100_{16})	16	uimsbf
CCI_and_other_info_data_length (=0084_{16})	16	uimsbf
(reserved)	5	bslbf
EPN	1	bslbf
CCI	2	bslbf
(reserved)	3	bslbf
Image_Constraint_Token	1	bslbf
Digital_Only_Token	1	bslbf
APSTB	3	bslbf
Num_of_Title	16	
for (I = 0; I < Num_of_Title; I++){		
Type_of_Title#I	1	Uimsbf
}		
(reserved)	1024 – Num_of_Title	bslbf
}		

CCI_and_other_info_type shall be 0101_{16} for Basic CCI for AACS.

CCI_and_other_info_version shall be 0100_{16} for this version.

CCI_and_other_info_data_length shall be 0084_{16} for Basic CCI for AACS.

The EPN field indicates the value of the Encryption Plus Non-assertion (EPN). Table 3-20 shows the meaning of EPN.

Table 3-20 EPN

EPN	Meaning
0_2	EPN-asserted
1_2	EPN-unasserted

The CCI field indicates the value of the copy control information. Table 3-21 shows the meaning of CCI.

Table 3-21 CCI

CCI	**Meaning**
00_2	Copy Control Not Asserted
01_2	Reserved for No More Copy
10_2	Copy One Generation
11_2	Never Copy

The Image_Constraint_Token field indicates the value of Image Constraint Token. Table 3-22 shows the meaning of Image_Constraint_Token.

Table 3-22   Image_Constraint_Token

Image_Constraint_Token	**Meaning**
0_2	High Definition Analog Output in the form of Constrained Image
1_2	High Definition Analog Output in High Definition Analog Form

The Digital_Only_Token field indicates the value of Digital Only Token. Table 3-23 shows the meaning of Digital_Only_Token.

Table 3-23   Digital_Only_Token

Digital_Only_Token	**Meaning**
0_2	Output of decrypted content is allowed for Analog/Digital Outputs
1_2	Output of decrypted content is allowed only for Digital Outputs

The APSTB field indicates the value of analog copy protection information. Table 3-24 shows the meaning of APSTB.

Table 3-24   APSTB

APSTB	Meaning
000_2	APS off
001_2	APS1 on: type 1 (AGC)
010_2	APS1 on: type 2 (AGC + 2L colourstripe)
011_2	APS1 on: type 3 (AGC + 4L colourstripe)
100_2-101_2	Reserved
110_2-111_2	APS2 on

Num_of_Title indicates the number of Title contained in this CPS Unit. Note that the First Playback and the Top Menu are not included in the “Title” and shall be basic.

Type_of_Title#I indicates whether the Title#I in this CPS Unit is basic or enhanced. Table 3-25 shows the meaning of Type_of_Title#I. Note that Title number in a specific CPS Unit is assigned in the ascending order of the title_id of each Title, which belongs to this CPS Unit.

Enhanced Title means a Title that requires Permission from Remote Server before playback. On the other hand, Basic Title does not require Permission. Details of Permission are defined in Section 3.9.4.3 and 4.4.1.5 of this specification.

Note that the Licensed Player that does not have AACS On-line capability shall not playback Enhanced Title.

Table 3-25 Type_of_Title#I

Type	Meaning
0_2	Basic Title
1_2	Enhanced Title

3.9.4.3 Enhanced Title Usage for AACS

Table 3-26 shows the data structure of CCI_and_other_info( ) for Enhanced Title Usage for AACS. Enhanced Title Usage for AACS shall be used only for Enhanced Title and shall be defined for each Title not CPS Unit .

Note that if the Licensed Player is capable of storing Cacheable Permission (i.e. a Secure Clock is implemented) and storage for Cacheable Permission is available at that time, the Licensed Player shall refer to this Enhanced Title Usage for AACS. For the details of Cacheable Permission, refer to Section 4.4.1.5 of this specification.

Table 3-26 Syntax of Enhanced Title Usage for AACS

Syntax	No. of bits	Mnemonics
Enhanced Title Usage for AACS {		
CCI_and_other_info_type (=0111$_{16}$)	16	uimsbf
CCI_and_other_info_version (=0100$_{16}$)	16	uimsbf
CCI_and_other_info_data_length (=0020$_{16}$)	16	uimsbf
Title_id	16	uimsbf
(reserved)	7	bslbf
Cacheable	1	uimsbf
Period	16	uimsbf
After( )	56	
Before( )	56	
(reserved)	104	bslbf
}		

CCI_and_other_info_type shall be 0111$_{16}$ for Enhanced Title Usage for AACS.

CCI_and_other_info_version shall be 0100$_{16}$ for this version.

CCI_and_other_info_data_length shall be 0020$_{16}$ for Enhanced Title Usage for AACS.

Title_id indicates the title_id of Title which this Title Usage is covered, where title_id is defined in *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0*.

Cacheable indicates whether this Permission can be cached or not. Table 3-27 shows the meaning of Type.

Table 3-27 Cacheable

Cacheable	Meaning
0_2	Cacheable Permission
1_2	Instant Permission

Period indicates the number of integer hours that the Cacheable Permission may stay in the cache until it must be deleted. A player may always delete it earlier.

After( ) indicates that a player may not begin playing the title until the date specified. The date is specified by the format shown in Table 3-28.

Before( ) indicates that a player may not begin playing the title after the date specified. The date is specified by the format shown in Table 3-28.

Note that fields “Period”, “After” or “Before” are valid only in case of Cacheable Permission. In case of Instant Permission, these fields shall be set to all-zero, and shall be ignored by a player.

Table 3-28 Syntax for After( ) and Before( )

Syntax	**No. of bits**	**Mnemonics**
After( ) or Before( ){		
First_digit_of_year	4	uimsbf
Second_digit_of_year	4	uimsbf
Third_digit_of_year	4	uimsbf
Fourth_digit_of_year	4	uimsbf
First_digit_of_month	4	uimsbf
Second_digit_of_month	4	uimsbf
First_digit_of_date	4	uimsbf
Second_digit_of_date	4	uimsbf
First_digit_of_hour	4	uimsbf
Second_digit_of_hour	4	uimsbf
First_digit_of_minute	4	uimsbf
Second_digit_of_minute	4	uimsbf
Timezone	8	imsbf
}		

Timezone is reserved for future use, to indicate the time difference in quarter hours between Coordinated Universal Time (UTC) and the local standard time. In this revision, this field shall be set to zero. The player shall ignore this field, and shall interpret the date/time value for After() or Before() as UTC.

Note that the URL of the Remote Server is not included in Enhanced Title Usage for AACS, but is handled by BD-J Application.

3.9.4.4 Key Management Information for On-line Function

Table 3-29 shows the data structure of CCI_and_other_info( ) for Key Management Information for On-line Function.

Table 3-29 Syntax of Key Management Information for On-line Function

Syntax	**No. of bits**	**Mnemonics**
Key Management Information for On-line Function {		
CCI_and_other_info_type (=0112$_{16}$)	16	uimsbf
CCI_and_other_info_version (=0100$_{16}$)	16	uimsbf
CCI_and_other_info_data_length (=0010$_{16}$)	16	uimsbf
Unit Key Status	8	uimsbf
Binding Type	8	uimsbf
(reserved)	112	bslbf
}		

CCI_and_other_info_type shall be 0112$_{16}$ for Key Management Information for On-line Function.

CCI_and_other_info_version shall be 0100$_{16}$ for this version.

CCI_and_other_info_data_length shall be 0010$_{16}$ for Key Management Information for On-line Function.

The Unit Key Status field indicates the status of Unit Key associated to the CPS_Unit. Table 3-30 shows the meaning of Unit Key Status. For example, if the Unit Key Status is 02$_{16}$, the Unit Key for the CPS_Unit does not exist on the BD-ROM Media, and some additional process (e.g. network transaction) to get Unit Key is necessary before the playback of the contents.

Table 3-30 Unit Key Status

Unit Key Status	**Meaning**
00$_{16}$	Reserved
01$_{16}$	Unit Key is recorded on the BD-ROM Media
02$_{16}$	Unit Key is not recorded on the BD-ROM Media
Others	Reserved

The Binding Type field indicates the Binding Type applied to the downloaded contents that belong to the CPS_Unit. Table 3-31 shows the meaning of Binding Type. If Key Management Information for On-line Function is not recorded in the CPS Unit Usage File, Licensed Player shall regard this CPS Unit as Content Binding.

Further information and definition of each binding type are described in Section 5.4 of the *Introduction and Common Cryptographic Elements* of this specification.

Table 3-31 Binding Type

Binding Type	Meaning
00_{16}	Reserved
01_{16}	Media Binding
02_{16}	Content Binding
03_{16}	Device/Content Binding
04_{16}	Device/Media Binding
Others	Reserved

3.9.4.5 Content Owner Authorized Outputs Information

Table 3-32 shows the data structure of CCI_and_other_info( ) for Content Owner Authorized Outputs Information.

Table 3-32 Syntax of Content Owner Authorized Outputs Information

Syntax	No. of bits	Mnemonics
Content Owner Authorized Outputs Information {		
CCI_and_other_info_type (=0113_{16})	16	uimsbf
CCI_and_other_info_version (=0100_{16})	16	uimsbf
CCI_and_other_info_data_length (=0010_{16})	16	uimsbf
Output Control Bits	128	uimsbf
}		

CCI_and_other_info_type shall be 0113_{16} for Content Owner Authorized Information.

CCI_and_other_info_version shall be 0100_{16} for this version.

CCI_and_other_info_data_length shall be 0010_{16} for Content Owner Authorized Information.

The Output Control Bits field contains Content Owner Authorized Output Control Bits. This field shall be filled with 00_{16} unless otherwise defined in the Compliance Rules.

3.10 Encrypted Packs

3.10.1 Encryption Scheme

When AACS encryption is applied to Clip AV stream files under the “\BDMV” directory, encryption is applied to every Aligned Units in the file. An Aligned Unit consists of 32 MPEG source packets: Each MPEG source packet consists of the TP_extra_header (4 bytes) and an MPEG Transport packet (188 bytes). The total size of an Aligned Unit is 6144 bytes, which is equal to the size of 3 logical sectors.

The final 6128 bytes of each Aligned Unit is encrypted using the Block Key and AES-128CBCE. A new CBC cipher chain is started for each Aligned Unit (see Figure 3-6).

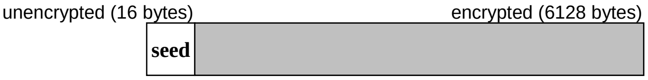


Figure 3-6 CBC chaining on “Aligned Unit” basis

The Initialization Vector of CBC Mode used in this scheme is described in Section 2.1.2 of *Introduction and Common Cryptographic Elements* of this specification.

The first 16 bytes of each Aligned Unit is used as the seed for calculating the Block Key. Calculation method for the Block key is described in Figure 3-7.

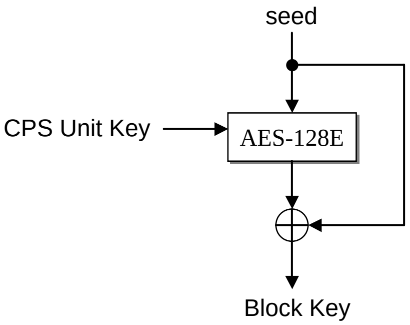


Figure 3-7 Calculation method for the Block Key from the CPS Unit Key

3.10.2 Copy Permission Indicator

MPEG source packet in Clip AV stream file consists of the TP_extra_header (4 bytes) and an MPEG Transport packet (188 bytes). Table 3-33 shows the data structure for TP_extra_header.

Table 3-33 TP_extra_header

Syntax	No. of bits	Mnemonic
TP_extra_header {		
Copy_permission_indicator	2	unimsbf
Arrival_time_stamp	30	unimsbf
}		

Copy_permission_indicator shall be set to 11_2 if the data is encrypted, or shall be set to 00_2 if the data is not encrypted. If the player encounters the packet with Copy_permission_indicator set to 10_2 or 01_2, the data shall be considered encrypted.

3.11 Embedded CCI in AV Content

As specified in *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0,* HDMV_copy_control_descriptor shall be embedded in AV Content.

The HDMV_copy_control_descriptor is used for the DTCP and contains the same fields and the same meaning defined in accordance with the DTCP_descriptor specified in *Digital Transmission Content Protection Specification Volume 1 Revision 1.4*. Table 3-34 presents the syntax. The information recorded in the CPS Unit Usage File defined in 3.9.4 and this HDMV_copy_control_descriptor shall be consistent unless otherwise defined in this chapter. For AACS compliant player implementation, the information recorded in the CPS Unit Usage File has priority rather than the information recorded in Embedded CCI.

Table 3-34 HDMV_copy_control_descriptor

Syntax	**No. of bits**	**Mnemonics**
HDMV_copy_control_descriptor {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
CA_System_ID	16	uimsbf
for (I = 0 ; I < descriptor_length – 2 ; I++){		
private_data_byte	8	bslbf
}		
}		

Descriptor_tag field (1 byte) shall be set to 88_{16}. Descriptor_length (1 byte) indicates the number of bytes immediately following this field and up to the end of this descriptor. CA_System_ID (2 bytes) shall be set to $0FFF_{16}$.

3.11.1 private_data_byte

Table 3-35 shows the data format for private_data_byte.

Table 3-35 private_data_byte

Syntax	No. of bits	Mnemonics
Private_data_byte {		
(reserved)	1	bslbf
Retention_Move_Mode	1	bslbf
Retention_State	3	bslbf
EPN	1	bslbf
CCI	2	bslbf
(reserved)	5	bslbf
Image_Constraint_Token	1	bslbf
APS	2	bslbf
}		

Retention_Move_mode and Retention_State are defined in the DTCP_descriptor, but these fields are not used in this specification.

EPN field indicates the value of the Encryption Plus Non-assertion (EPN) as shown in Table 3-36.

Table 3-36 EPN

EPN	Meaning
0_2	EPN-asserted
1_2	EPN-unasserted

CCI field indicates the value of the copy control information as shown as Table 3-37.

Table 3-37 CCI

CCI	Meaning
00_2	Copy Control Not Asserted
01_2	Reserved for No More Copy
10_2	Copy One Generation
11_2	Never Copy

Image_Constraint_Token field indicates the value of the Image_Constraint_Token as shown in Table 3-38.

Table 3-38 Image_Constraint_Token

Image_Constraint_Token	Meaning
0_2	High Definition Analog Output in the form of Constrained Image
1_2	High Definition Analog Output in High Definition Analog Form

APS field indicates the value of the analog copy protection information as shown in Table 3-39. The value of APS field shall be set in accordance with the *AACS Compliance Rules*.

Table 3-39 APS

APS	Meaning
00_2	copy control not asserted
01_2	APS on: type 1 (AGC)
10_2	APS on: type 2 (AGC + 2L colourstripe)
11_2	APS on: type 3 (AGC + 4L colourstripe)

Reserved bits are reserved for future definition and currently defined to have a value of one.

Chapter 4
Details for Uses of On-line Connections

4. Introduction

The information related to the contents use with network transaction is specified in Chapter 5 of *Introduction and Common Cryptographic Elements* of this specification. This chapter describes additional details of on-line functions that are specific to the use of AACS encryption with BD-ROM Media and Application Format.

4.1 Virtual File System

BD-ROM application format introduces a concept of Virtual Package. By use of this concept, downloaded content in the Binding Unit Data Area of Local Storage (e.g. HDD) and pre-recorded content on the BD-ROM are combined as one virtual "packaged media".

According to the application image described in Chapter 5 of AACS *Introduction and Common Cryptographic Elements* book of this specification, downloaded files include not only content files, but also files for copy protection (e.g. CPS Unit Key File, etc. recorded in AACS directory).

To realize the concept of Virtual Package, the BD-ROM application format specifies Binding Unit Data Area in local storage and the mechanism of Virtual File System (VFS) to play the contents on BD-ROM Disc together with the files recorded in Binding Unit Data Area. Files for copy protection are also covered by this mechanism. For more detail, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0.*

If the Licensed Player implements VFS for the files defined in the BD-ROM application format, the Licensed Player shall apply the VFS for copy protection files as defined in this specification.

If the Licensed Player implements AACS On-line APIs in addition to BD-J network capability, the Licensed Player shall have capability to process the contents recorded in Binding Unit Data Area with any of four binding methods defined in Section 5.5 of AACS *Introduction and Common Cryptographic Elements* book of this specification.

This section describes the application of the Virtual Package concept to files in the AACS directories.

Figure 4-1 shows the example of the VFS concept applied to files in the AACS directories.

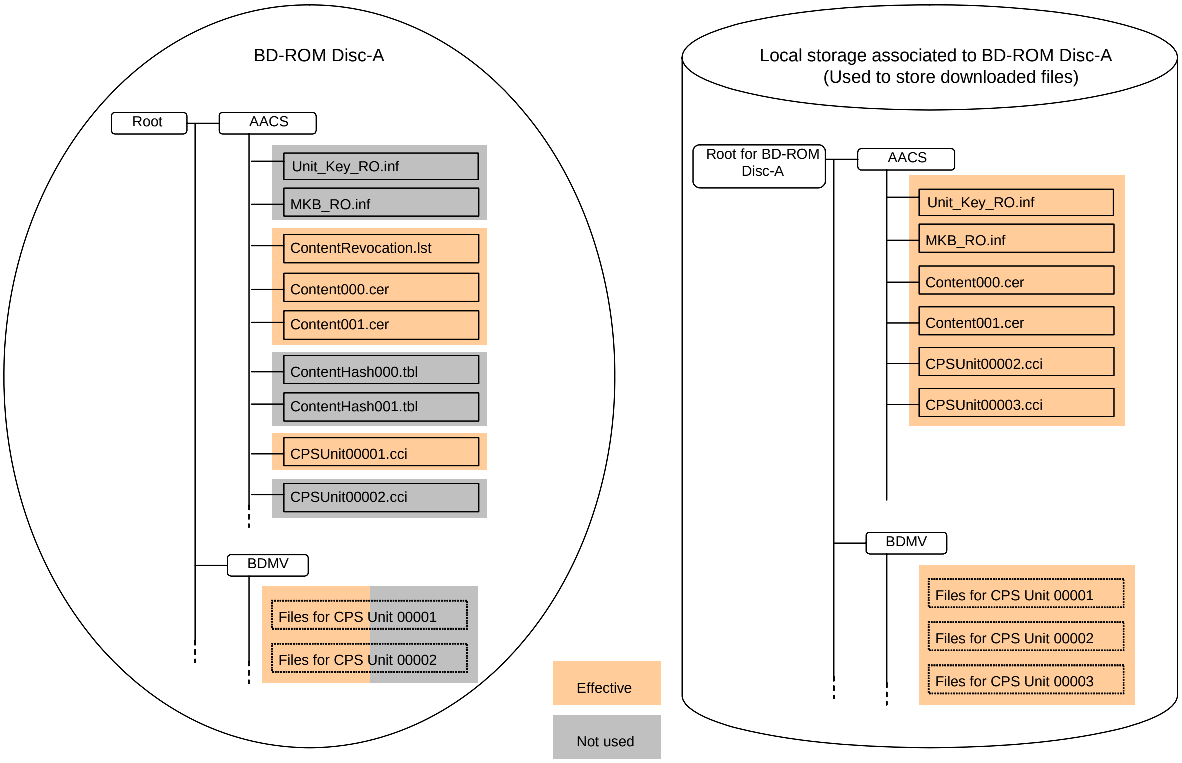


Figure 4-1 Virtual File System Concept to files in the AACS and BDMV directory

In this example, CPS Unit#1 and CPS Unit#2 are originally recorded on the BD-ROM Disc-A as described in the left hand side of Figure 4-1.

The downloaded files are recorded in the specific area of the Binding Unit Data Area which is associated with the specific disc (ex. BD-ROM Disc-A). In this example, the downloaded contents are some updated files for CPS Unit#1 and CPS Unit #2, and new contents for CPS Unit#3. Figure 4-1 shows only the partial update of CPS Unit#1 and CPS Unit#2, and new addition of CPS Unit#3. And the details of AV Application files are omitted.

Note that the actual directory name and file name registered in the File System of Local Storage media may be different from the directory name and file name defined for BD-ROM. However, VFS provides name mapping mechanism from the actual file name used on Local Storage to the BD-ROM defined file name, and VFS recognizes the directory and file structure as described in Figure 4-1, Figure 4-2, Figure 4-5, Figure 4-8, and Figure 4-11 are also described based on the applicacation of name mapping mechanism, but the details of name mapping process is omitted in the figures.

For the detail of AV application files in the BDMV directory and VFS for AV application files, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specification, version 2.0*.

Even when some downloaded files are recorded in the Binding Unit Data Area, verifying, Contents Certificate process shall be completed according to the procedure defined in Section 2.3.3 of this specification. Two procedures for Content Certificate verification are defined in Section 2.6 of *AACS Pre-recorded Video Book* of this specification. For procedure a), verification shall be completed before the construction of the VFS. On the other hand, for procedure b), verification shall be done during playback. Note that for both procedures, only the files recorded in BD-ROM Disc are used for verification. Refer to Section 4.1.1 of this specification for the details of these specific files.

For each file in the AACS directories, the actual meaning of updates is explained using the example of Figure 4-1.

CPS Unit Key File: CPS Unit Key File is updated in this example. The CPS Unit Key File in BD-ROM Disc has the encrypted keys for CPS Unit#1 and CPS Unit#2. The CPS Unit Key File in the Binding Unit Data Area has the encrypted keys for all of CPS Unit#1, CPS Unit#2, and CPS Unit#3. Therefore, the CPS Unit Key File in the Binding Unit Data Areahas all encrypted keys that are necessary to play all the content in the VFS.

Content Certificate: Content Certificates are updated in this example in parallel with updating the CPS Unit Usage File. Content Certificate includes the hash values of each CPS Unit Usage File. It is necessary to update the Content Certificate when the CPS Unit Usage File is updated or added.

CPS Unit Usage File: CPS Unit Usage File for newly downloaded CPS Unit is added during the downloading transaction. And the CPS Unit Usage File in the BD-ROM Disc can be used unless the CPS Unit Usage was changed during the downloading transaction according to the intention of contents participants. In the case that CPS Unit Usage File in the BD-ROM Disc is updated by the downloading transaction, the new CPS Unit Usage File which has the same file name is downloaded to the Binding Unit Data Area, and set to effective. On the other hand, in the case that CPS Unit Usage File is added by the downloading transaction, the new CPS Unit Usage File which has the different file name is downloaded to the Binding Unit Data Area, and set to effective. The Licensed Player shall verify the Hash_Value_of_CPS_Unit_Usage_File as defined in Section 2.3.3.2 of this specification. In addition to verification of the Hash_Value_of_CPS_Unit_Usage_File, the Licensed Player shall also verify the Signature of Content Certificate associated with that CPS Unit Usage File before playback of the associated title.

If at least one AACS file listed above is replaced (by e.g. construction or update of Virtual Package), the Licensed Player shall process the file again before playback.

Figure 4-2 shows the disc image of the contents on the Binding Unit Data Area.

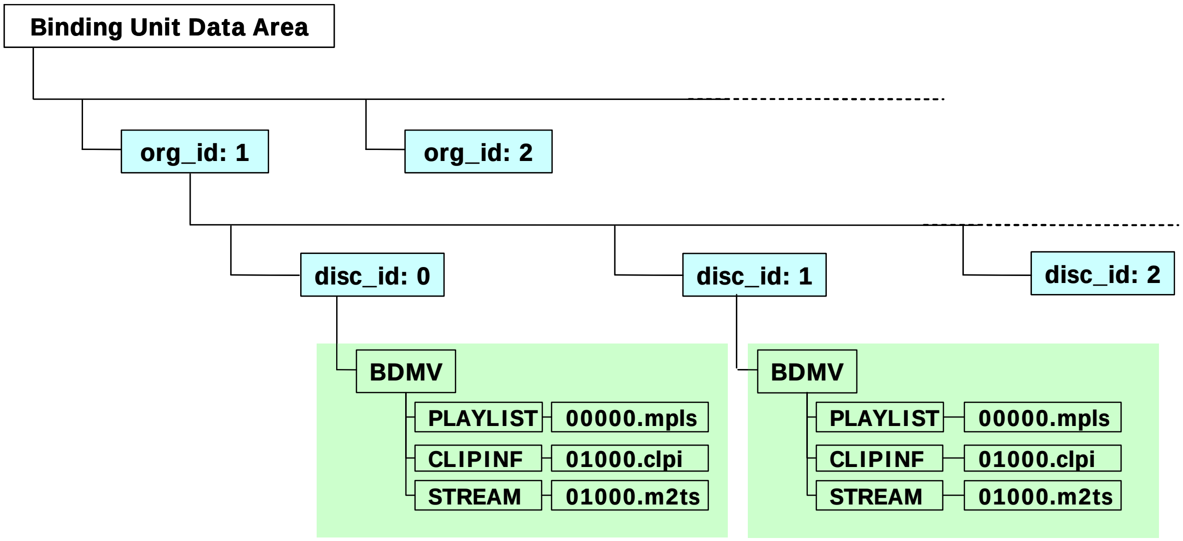


Figure 4-2 Disc Image of Content on the Binding Unit Data Area

The contents on the Binding Unit Data Area are stored under the disc-dependent directory in organization-dependent directory by using the disc_id and organization_id recorded on associated BD-ROM Disc. Organization_id may be assigned by each Content Provider and disc_id is assigned to each content. For the details of the organization_id and the disc_id, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0.*

4.1.1 AACS Files for VFS

When AACS files are recorded on Binding Unit Data Area, the same files on BD-ROM Disc shall be replaced by the files on Binding Unit Data Area when VFS is constructed.

However, Media Key Block and Sequence Key Block shall be processed without VFS, so the replacement of these files has no effect on MKB and SKB process. Content revocation process shall be also processed without VFS, so the replacement of Content Hash Table and Content Revocation List has no effect on content revocation.

In other words, the Licensed Player shall not use MKB, SKB, Content Hash Table and Content Revocation List on the Local Storage, but use these AACS files on the BD-ROM for Process MKB, SKB or Content Revocation.

4.2 System Model

As an overview, the On-line System based on AACS and BD-ROM application format consists of three modules: Remote Server, BD-J Application and AACS Layer. Figure 4-3 shows the relation between these three modules.

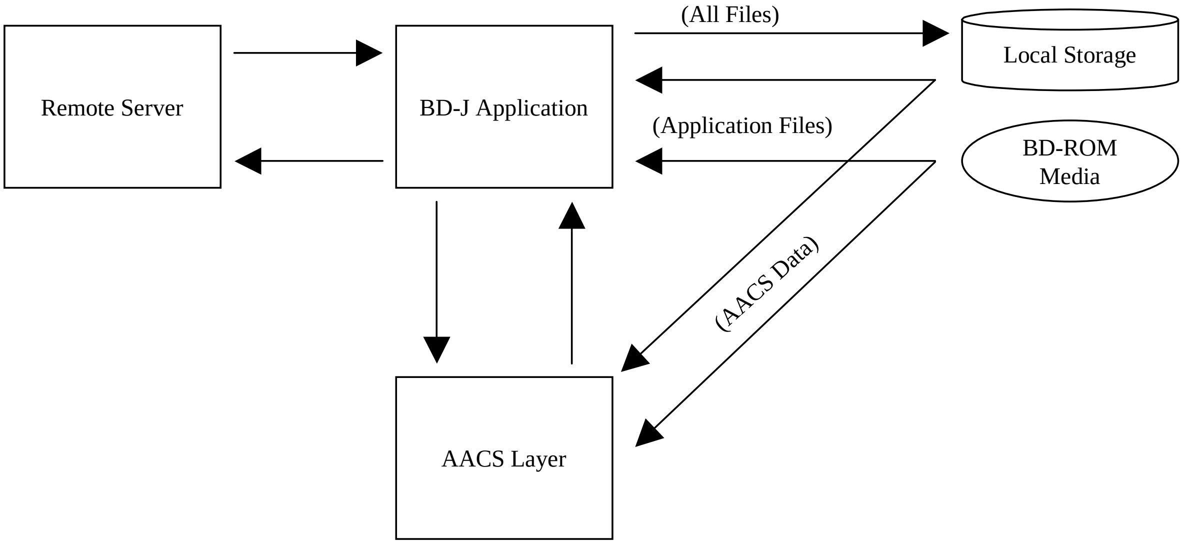


Figure 4-3 System Model: Relation between three modules

BD-J Application sends a request message and receives a response message. These messages are defined in Section 4.3 of this specification as "connection protocol". Some types of response message may be recorded as a file to the Binding Unit Data Area; e.g. Clip AV stream file, CPS Unit Key File and CPS Unit Usage File.

BD-J Application reads a file from the Binding Unit Data Area and BD-ROM. This file includes Clip AV stream file and Database files defined in the BD-ROM application format, and AACS files. However, BD-J Application does not have direct read access to defined data (e.g. Volume ID, PMSN etc.).

When BD-J Application needs the information related to the AACS Layer, the BD-J Application calls the AACS Layer and receives a return message from it. These messages are defined in Section 4.4 of this specification as APIs between AACS Layer and BD-J Application. Using these APIs, BD-J Application requests AACS-Layer to access AACS defined data.

4.3 Connection Protocol between Remote Server and BD-J Application

BD-ROM application format defines a programmable environment to enhance its interactive feature. By use of this programmable environment, the connection protocol between Remote Server and BD-J Application can be implemented by a service provider's own choice. As an example, the Example Protocol for On-line Enabled Content defined in the *Introduction and Common Cryptographic Elements* of this specification can be utilized for this connection protocol between the Remote Server and the BD-J Application. Note that BD-J supports TLS with cipher suite TLS_RSA_WITH_AES_128_CBC_SHA, which is used in the Example Protocol for On-line Enabled Content.

4.4 APIs between AACS Layer and BD-J Application

The connection protocol for the on-line transactions is defined in Section 5.3 of AACS *Introduction and Common Cryptographic Element* book of this specification. This section provides the list of APIs that can be used by BD-J Applications to execute the network transactions. The BD-J Application shall check player's VFS implementation and AACS On-line capability by referring to Player Status Register 31 (PSR31) value and System Property respectively. The APIs defined in this section shall be called only when the player has capability to process such APIs appropriately The Licensed Player with AACS On-line capability is regarded as Enhanced Device, as defined in Chapter 5 of AACS *Introduction and Common Cryptographic Elements* book of this specification.

Note that the BD-ROM application format defines that a player with BD-J network connectivity shall support the VFS. This means that the player with AACS On-line capability has VFS capability.

For Player Status Register, refer to *Blu-ray Disc Association, System Description Blu-ray Disc Read-Only Format, part 3: Audio Visual Basic Format Specifications, version 2.0*.

Further requirement and recommendation for AACS On-line API implementation and BD-J application are defined in Annex D of this specification.

4.4.1 Package com.aacsla.bluray.online

4.4.1.1 Class Summary

Following four classes are defined as AACS On-line API, and shall be supported by the Licensed Player with AACS On-line capability. If the AACS On-line API is supported, such players shall support BD-J network connectivity also.

MediaAttribute

The MediaAttribute handles media attributes provided by AACS Layer.

DeviceAttribute

The DeviceAttribute handles device attributes provided by AACS Layer.

ContentAttribute

The ContentAttribute handles content attributes provided by AACS Layer.

EnablePermission

The EnablePermisson handles Permission for AACS On-line Enabled Content as defined in Chapter 5 of AACS *Introduction and Common Cryptographic Elements* book of this specification.

4.4.1.2 Class MediaAttribute

```
java.lang.Object
      |
      +--com.aacsla.bluray.online.MediaAttribute
```

public class **MediaAttribute**

extends java.lang.object

The MediaAttribute handles media attributes provided by AACS Layer.

4.4.1.2.1 Constructors

4.4.1.2.1.1 MediaAttribute

public **MediaAttribute** ( )

Create MediaAttribute object.

4.4.1.2.2 Methods

4.4.1.2.2.1 getVolumeID

public byte[ ] **getVolumeID**( )

Provide the Volume ID of the currently inserted media. Note that Volume ID is 16 bytes.

Returns:

the Volume ID. If there is no currently inserted media or any other error, returns null.

4.4.1.2.2.2 getPMSN

public byte[ ] **getPMSN**( )

Provide the Pre-recorded Media Serial Number of the currently inserted media. Note that Pre-recorded Media Serial Number is 16 bytes. Integrity of Pre-recorded Media Serial Number needs not be guaranteed outside of AACS Layer.

Returns:

the Pre-recorded Media Serial Number. If Pre-recorded Media Serial Number is not recorded in the currently inserted media, returns null. If there is no currently inserted media or any other error, returns null also.

4.4.1.3 Class DeviceAttribute

```
java.lang.Object
     |
     +--com.aacsla.bluray.online.DeviceAttribute
```

public class **DeviceAttribute**

extends java.lang.object

The DeviceAttribute handles device attributes provided by AACS Layer.

4.4.1.3.1 Constructors

4.4.1.3.1.1 DeviceAttribute

public **DeviceAttribute** ( )

Create DeviceAttribute object.

4.4.1.3.2 Methods

4.4.1.3.2.1 getDeviceBindingID

public byte[ ] **getDeviceBindingID**( )

Provide the Device Binding Nonce of the device. Note that Device Binding Nonce is 16 bytes.

Returns:

the Device Binding Nonce. If a Licensed Player does not have Device Binding Nonce or there is any other error, returns null.

4.4.1.4 Class ContentAttribute

```
java.lang.Object
    |
    +--com.aacsla.bluray.online.ContentAttribute
```

public class **ContentAttribute**

extends java.lang.object

The ContentAttribute handles content attributes provided by AACS Layer.

4.4.1.4.1 Constructors

4.4.1.4.1.1 ContentAttribute

public **ContentAttribute** ( )

Create ContentAttribute object.

4.4.1.4.2 Methods

4.4.1.4.2.1 getContentCertID

public byte[ ] **getContentCertID**( )

Provide the Content Certificate ID of the currently inserted media. Note that Content Certificate ID is 6 bytes, and defined in Section 2.1 of this specification.

Returns:

the Content Certificate ID. If there is no currently inserted media or any other error, returns null. When the Content Certificate was replaced by VFS, Content Certificate ID retuned by this method shall be retrieved from the Content Certificate on the Binding Unit Data Area of Local Storage. In case of dual layer disc, Content Certificate ID in Content000.cer shall be returned.

4.4.1.5 Class EnablePermission

```
java.lang.Object
     |
     +--com.aacsla.bluray.online.EnablePermission
```

public class **EnablePermission**

extends java.lang.object

The EnablePermisson handles on-line Permission as defined in Chapter 5 of AACS *Introduction and Common Cryptographic Elements* book of this specification.

Note that the Licensed Player with AACS On-line capability shall be capable of handlng both Instant Permission and Cacheable Permission. However, the Licensed Player which is not capabile of storing Cacheable Permission, shall treat every Cacheable Permission as an Instant Permission. The treatment of Cacheable Permission for a Licensed Player which is capable of storing Cacheable Permission is defined in Section 4.4.1.5.2.2 of this specification.

The BD-J Application shall not try to play the Titles which do not have valid Permission. The Licensed Player treats such a playback request as an illegal request and shall block the playback of such Titles. The player behavior after such an illegal request is implementation dependent.

4.4.1.5.1 Constructors

4.4.1.5.1.1 EnablePermission

public **EnablePermission** (int title_id)

Create EnablePermission object.

Parameters:

title_id – title_id of the Title which this Permission corresponds to.

(Note) First Playback and Top Menu may not be appointed by this call, because those are always Basic Title and title_id is not assigned.

4.4.1.5.2 Methods

4.4.1.5.2.1 getNonce

public byte[ ] **getNonce**( )

Provide the Nonce generated by AACS Layer. The number of Nonces, which a player can hold, shall be one. If there is another existing Nonce that has been already generated, this call shall clear the existing Nonce in AACS Layer. If the existing Nonce is cleared, setPermission() corresponding to the existing Nonce returns false because Nonce doesn't match to message. In the case of disc ejection or power off, the existing Nonce shall be cleared. Note that Nonce is 16 bytes.

Note that getNonce() can be called even for the Basic Title, and a player behavior shall be the same as Enhanced Title. If the media is inserted, the player shall always return a non-null value even if the title specified by the parameter of the constructor does not exist on the media.

Returns:

the Nonce generated by AACS Layer. If there is no currently inserted media or any other error, returns null.

4.4.1.5.2.2 setPermission

public boolean **setPermission** (byte[ ] message)

throws

java.lang.NullPointerException

Set the message which is received from Remote Server to verify and activate the Permission. Note that the length of message shall be 16 bytes. This call shall clear existing Nonce. This also applies in case the argument is null.

If the Licensed Player is capable of storing Cacheable Permission, Cacheable Permission shall be kept for each combination of VolumeID and title_id.

Returns:

true: the message is verified correctly and the Permission is activated. If the Permission was already activated for the same title_id prior to this call, the Permission is overwritten by new Permission.

false: the message is not verified correctly, the length of message is not 16 bytes, or there is any other error. If the Permission was already activated for the same title_id prior to this call, the Permission shall be deactivated.

Throws:

java.lang.NullPointerException – if any of the arguments are null. If the Permission for the title_id has been already activated, the Permission for that title_id shall be deactivated.

The concept of Permission is defined in Section 5.3 of the AACS *Introduction and Common Cryptographic Elements* book of this specification.

In the case of BD-ROM, the Encrypted Title Key (CPS Unit Key) formula is defined as follows:

$$\text{AES-128E}(K_{vu}, K_t \oplus \text{Nonce} \oplus \text{AES\_H}(\text{Volume ID} \,||\, \text{title\_id}))$$

The procedure to check the message is also defined in Section 5.3 of the AACS *Introduction and Common Cryptographic Elements* book of this specification. In this formula, title_id is 4 bytes. title_id is the one used to construct the instance of EnablePermission, which calls setPermisson. Kvu and VolumeID are derived from the currently inserted media. Kt is the CPS Unit key which is associated with the title specified by title_id in the package (virtual or disc) that was being used when setPermission() was called. Nonce is the one existing in AACS Layer. Concretely, a Licensed Player collects the values (Kvu, Kt, Nonce, VolumeID, and title_id) from a Licensed Player itself, BD-ROM or Local Storage, and calculates above formula. If the results of the calculation matched to the message set by setPermission(), the verification is successful. . Otherwise, the verification is failed.

Only in the case when setPermission() or checkPermission() is called, or a Licensed Player is about to activate the Enhanced Title, a Licensed Player shall check if the Permission for that title is activated. The Licensed Player shall not start the playback of the Enhanced Title, unless the Permission for this title is activated.

A Licensed Player which is capable of storing Cacheable Permission shall treat the Cacheable Permission as following;

A Licensed Player shall treat the Cacheable Permission under the rule defined in Enhanced Title Usage for AACS in the associated CPS Unit Usage File. If the current time is out of playable time specified by “Period”, “After” or “Before” fields of Enhanced Title Usage for AACS, a Licensed Player shall not start the playback of the Enhanced Title even if the Title has activated Permission. If all of “Period”, “After” and “Before” are undefined (the values are all zero), a Licensed Player can cache the Permission forever. Note that the Licensed Player shall evaluate “Period”, “After” or “Before” field, when the Licensed Player tries to start the playback of the Enhanced Title. Once the playback of the Enhanced Title started, that Enhanced Title may be continuously played until the player stops the playback of the corresponding Title even if the current time became out of playable time.

Note that the maximum number of Cacheable Permissions to be stored is implementation specific.

A Licensed Player may implement two types of Secure Clock as defined in Section 5.2.1 of the *Introduction and Common Cryptographic Elements* book of this specification. Secure Clock for elapsed time purpose can handle only “Period” and shall ignore “After” and “Before”. On the other hand, Secure Clock for calendar time purpose can handle “Period”, “After” and “Before”. Each type of Secure Clock can handle the Usage Rules as defined in Table 4-1.

Table 4-1 Capability of handling time-based Usage Rules

	Usage Rule: **- Period is defined, and** **- After and/or Before is defined**	**Usage Rule:** **- Period is defined** **- (After and Before are undefined)**	**Usage Rule:** **- (Period is undefined)** **- After and/or Before are defined**	**Usage Rule:** **- (Period is undefined)** **- (After and Before are undefined)**
Secure Clock for elapsed time purpose	**Can handle (and ignore After and Before field)**	**Can handle**	**Cannot handle (and shall handle as Instant Permission)**	**Can handle**
Secure Clock for calendar time purpose	**Can handle**	**Can handle**	**Can handle**	**Can handle**

(Note 1) For a Licensed Player that has a Secure Clock for calendar time purpose, if either Period or After/Before is in playable time, the title can be played back.
(Note 2) In case storage for Cacheable Permission is not available (i.e. storage is full), a Licensed Player shall

handle Cacheable Permission as Instant Permission.

A Licensed Player shall treat the Instant Permission as following;

In the case of disc ejection or power off, the activated Instant Permission shall be deactivated. Even if the VFS is re-constructed or the instance of the EnablePermisson is deleted (e.g. due to garbage collection), the activated Instant Permission shall not be deactivated.

Note that the maximum number of Instant Permissions is the maximum number of Titles (i.e. 999) on the BD-ROM and a Licensed Player shall be capable of storing all the Instant Permissions.

(Note 3) setPermission() can be called even for the Basic Title, and behavior of a Licensed Player (including treatment of Permission) shall be the same as Enhanced Title. If setPermission() is called for the Title which does not exist on the media or Local Storage, a Licensed Player shall return false.

(Note 4) When a PlayList is played back, BD-J Application shall not use setPermission() for the Title that belongs to a different CPS Unit. In other words, both the Title which includes a BD-J Application that uses setPermission() and the Title corresponding to an enablePermission instance to which setPermission was used shall belong to the same CPS Unit, during PlayList playback.

4.4.1.5.2.3 checkPermission

public boolean **checkPermission** ( )

Check activation state of the Permission in AACS Layer. In addition, if the associated Permission is the Cacheable Permission and the Licensed Player is capable of storing Cacheable Permission, check whether the current time is out of playable time specified by “Period”, “After” or “Before” fields of associated Enhanced Title Usage for AACS.

Returns:

true: the Permission for the Title is active. In addition, if the Permission is Cacheable Permission, the current time is in playable time specified by “Period”, “After” or “Before” fields of associated Enhanced Title Usage for AACS. Note that the Licensed Player that is not capable of storing Cacheable Permission cannot check current time, therefore if the Permission is active, the Licensed Player shall return “true” regardless of current time.

false: the Permission for the Title is not active, the current time is out of playable time specified by “Period”, “After” or “Before” fields of associated Enhanced Title Usage for AACS, or any other error case.

To avoid unexpected results for the consumer, it is strongly recommended that the BD-J Application uses this API before Title transition to ensure activation state of the Permission for the next Title.

4.4.1.5.2.4 isCacheable

public boolean **isCacheable** ( )

Check the capability of storing the Cacheable Permission.

Returns:

true: the Licensed Player is capable of storing Cacheable Permission (i.e. a Secure Clock is implemented) and storage for Cacheable Permission is available at that time.

false: the Licensed Player is not capable of storing Cacheable Permission (i.e. a Secure Clock is not implemented) or storage for Cacheable Permission is not available (i.e. storage is full).

A Licensed Player which returns false to isCacheable() method shall treat the Cacheable Permission as Instant Permission when such Licensed Player got Cacheable Permission in the transaction for Enhanced Title.

A Licensed Player which returns true to isCacheable() method shall be capable of treating Cacheable Permission with or without time period in proper manner.

4.5 AACS Media Binding

Types of AACS Media Binding scheme is defined in Section 5.5 of AACS *Introduction and Common Cryptographic Element* book of this specification. Binding Type of the contents is stored in CPS Unit Usage File as defined in Section 3.9.4.4 of this specification.

Since the playback of Media and/or Device Binding Content is supported by only the Licensed Player with AACS On-line capability, it is strongly recommended for BD-J Application to check the player's AACS On-line capability before the title transition to the title which uses Media and/or Device Binding mechanism indicated by Binding Type in Key Management Information for On-line Function.

Note that a player behavior is also implementation specific when a player tries to play Title but MAC of PMSN (or Device Binding Nonce) recorded in CPS Unit Key File on the Binding Unit Data Area is different from the one calculated from CPS Unit Key and current PMSN (or Device Binding Nonce). To avoid unexpected results for the consumer, it is strongly recommended that the BD-J Application checks that PMSN (or Device Binding Nonce) is the expected value by use of getPMSN() defined in Section 4.4.1.2.2.2 (or getDeviceBindingID() defined in Section 4.4.1.3.2.1) of this specification before the title transition to each title which is using Binding scheme with MAC check mechanism. To realize this, for example, Figure 4-4 shows how to check PMSN (or Device Binding Nonce). Here is a sequence of procedure.

1. BD-J Application prepares the expected value of PMSN (or Device Binding Nonce) in Local Storage, which is associated with CPS Unit Key File on the Binding Unit Data Area.
2. BD-J Application requests PMSN (or Device Binding Nonce) to AACS Layer.
3. AACS Layer returns PMSN (or Device Binding Nonce) to BD-J Application.
4. BD-J Application compares PMSN (or Device Binding Nonce) from AACS Layer with the expected value.

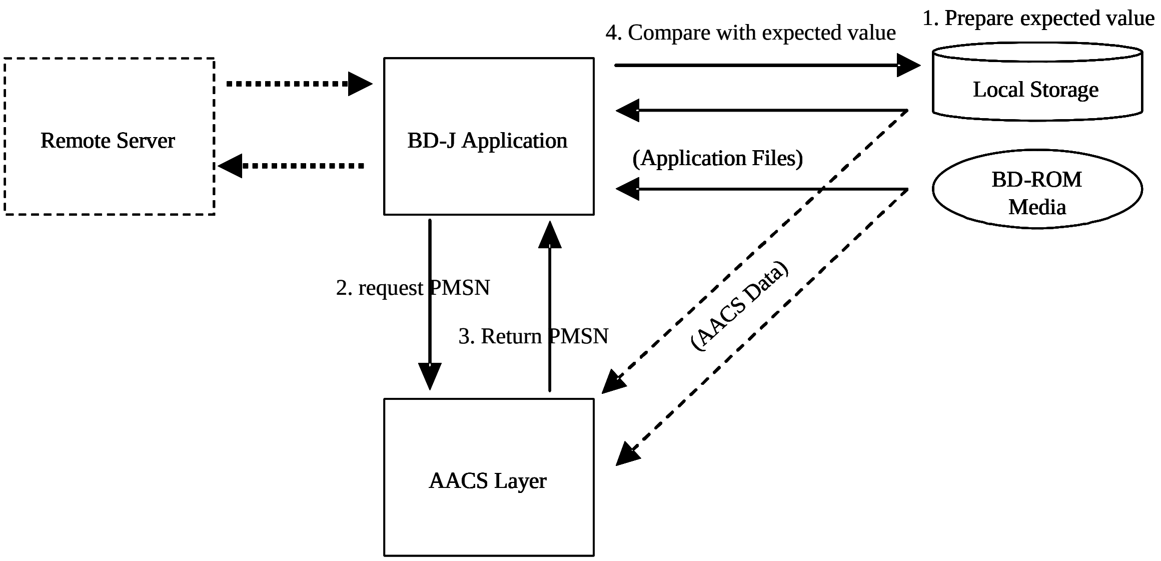


Figure 4-4 How to Check PMSN (or Device Binding Nonce)

The content shall not try to play the titles which do not have playable status in Media and/or Device Binding. The player treats such playback request as an illegal request and shall block the playback of such titles. The player behavior after such an illegal request is implementation dependent.

4.6 Example for the contents use with network transaction

4.6.1 Download additional Content

In this example, additional content is downloaded and stored into the Binding Unit Data Area of Local Storage. There are two cases for this example. One case is that the content is added as a new Title, and the other case is that the content is added to the existing Title.

Figure 4-5 shows the directory structure of this example.

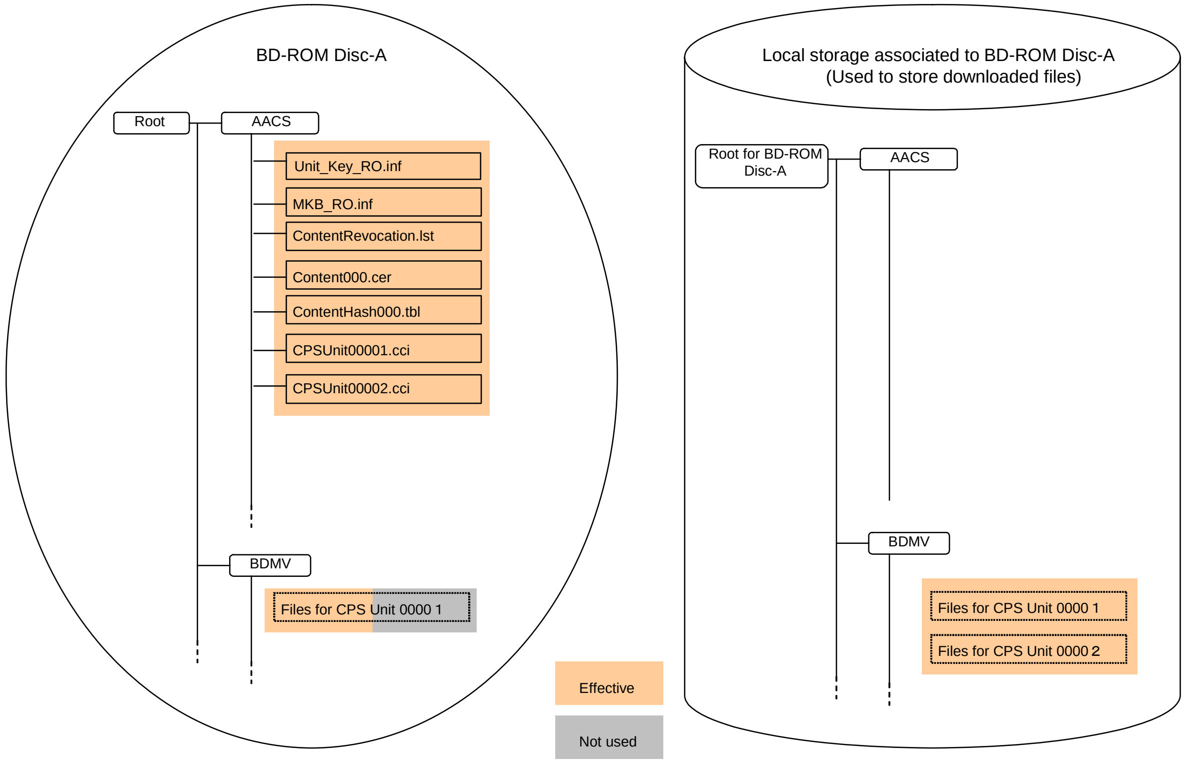


Figure 4-5 Example: Download additional Content

All files under AACS directory are pre-recorded on the media, and there is no download for these files. Files for CPS Unit 00001 are added in this example. Then some files in the media might be overridden by the corresponding files in the Binding Unit Data Area of Local Storage. This case might be useful to update a set of trailers in timely manner.

There are no files for CPS Unit 00002 in the media, and all files are downloaded and stored into the Binding Unit Data Area of Local Storage. This case might be useful to add bonus material after the packaged media are sold.

In both cases, CPS Unit Usage Files and CPS Unit Key File are pre-recorded on the media. The users, who have the media, might be able to receive additional content without charge.

Figure 4-6 shows how to realize this example.

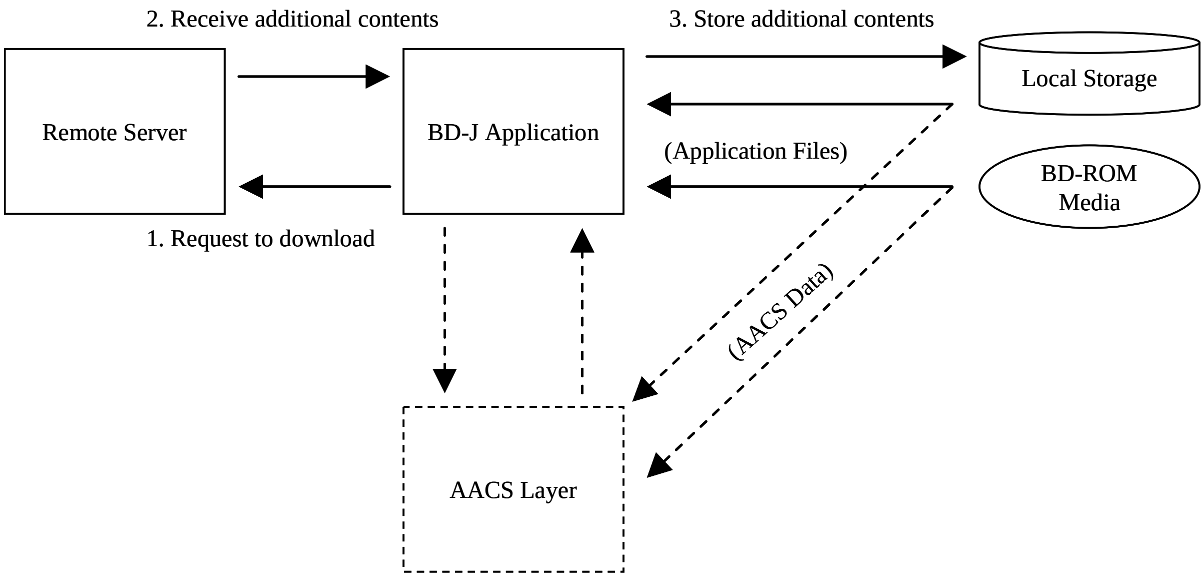
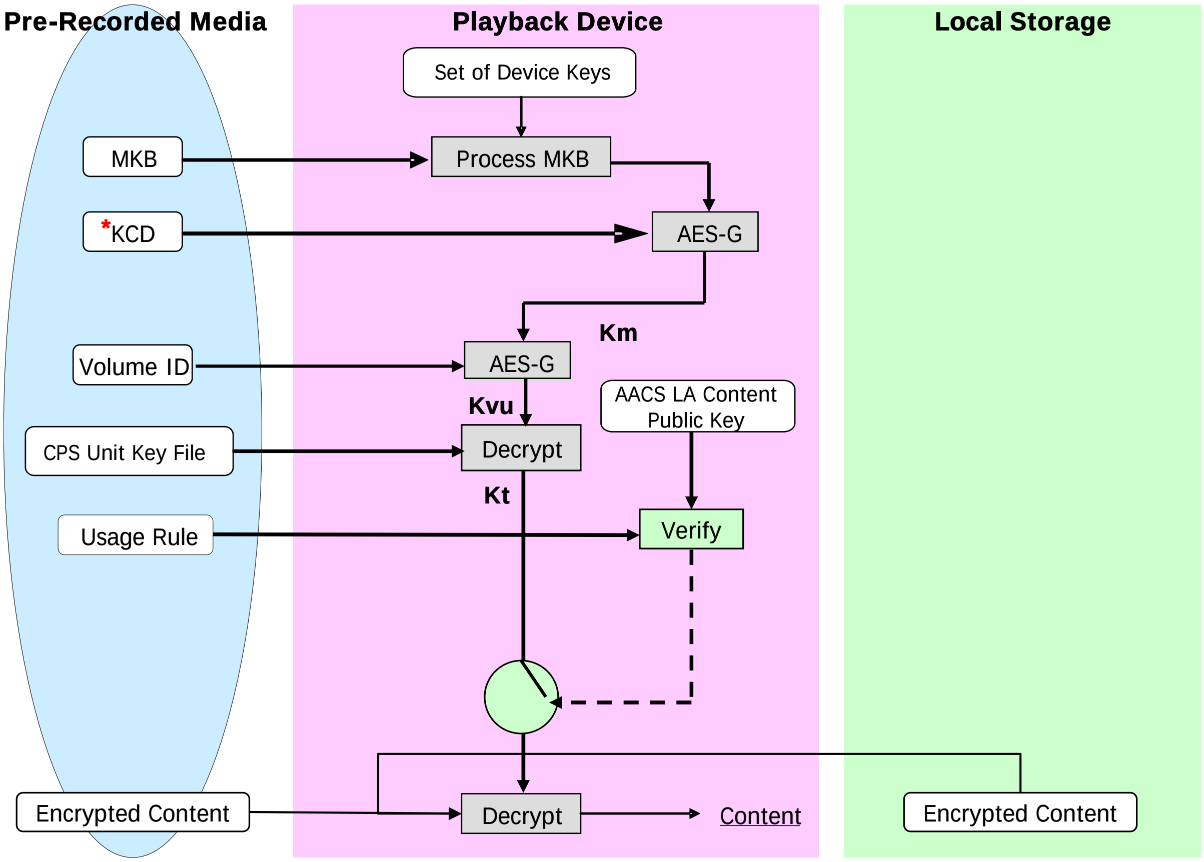


Figure 4-6 How to realize Download additional content

To realize this example, it is not necessary to utilize on-line functionality of the AACS Layer. This example can be realized without the AACS Layer. The BD-J Application requests to download additional content to a Remote Server and stores it into the Binding Unit Data Area of Local Storage.

Of course, after the download process is completed, the AACS Layer is necessary to play the content in both media and the Binding Unit Data Area of Local Storage.

Figure 4-7 describes a decryption overview for the BD-ROM and the Binding Unit Data Area of Local Storage in case of download additional content.



*KCD is used by only certain classes of devices.

Figure 4-7 Decryption Overview for BD-ROM and Binding Unit Data Area (1)

4.6.2 Download updated Usage Rule

In this example, an updated Usage Rule is downloaded and stored into the Binding Unit Data Area of Local Storage. There are two cases for this: one case is that the binding of the Title Key is still Content Binding, and the other case is that the binding of the Title Key is changed to another type of Binding.

Figure 4-8 shows the directory structure of this example. For both cases, directory structure is identical.

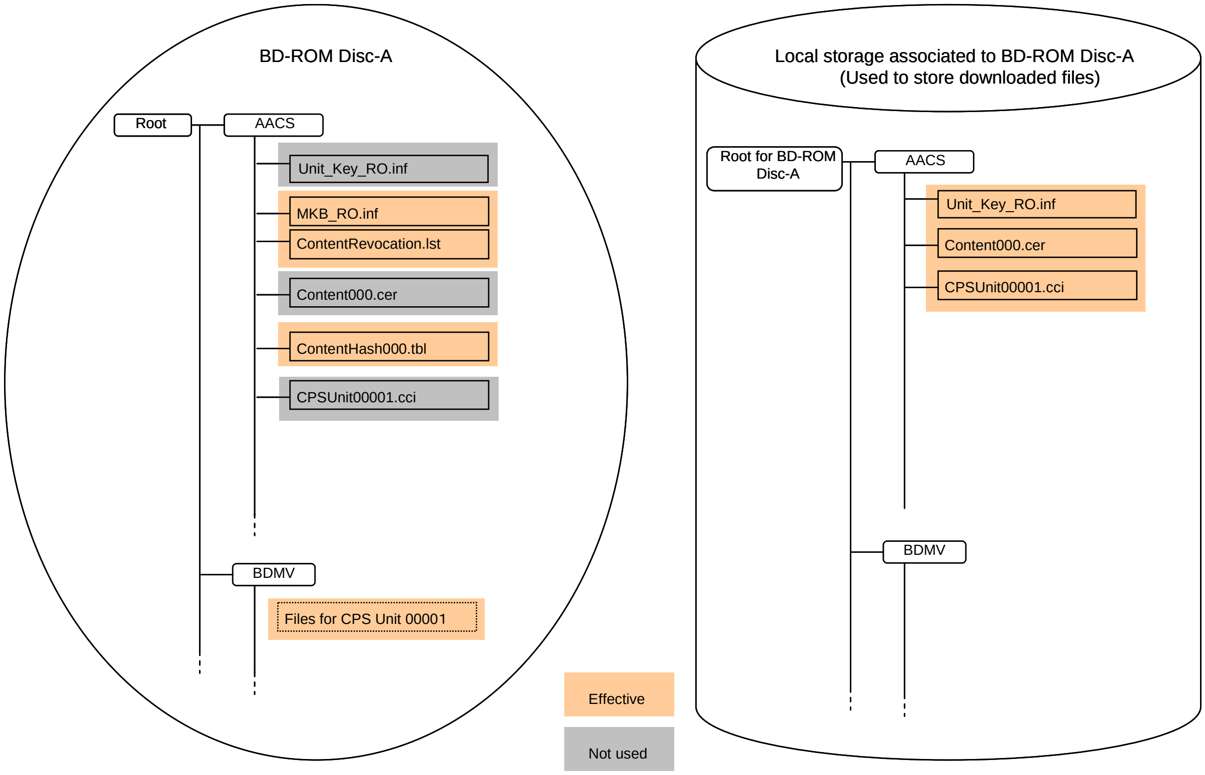


Figure 4-8 Example: Download updated Usage Rule

All files under BDMV directory are pre-recorded on the media, and there is no download for these files. CPS Unit Usage File for CPS Unit 00001 is pre-recorded on the media, and it would be updated (overridden) by CPS Unit Usage File for CPS Unit 00001 stored in the Binding Unit Data Area of Local Storage. Related to this, Content Certificate is also updated, because there is a hash of CPS Unit Usage File in this file. When the binding of the CPS Unit Key is changed to another type of Binding, CPS Unit Key File is also updated, because there is a binding information (MAC value) in this file.

For the case that the binding of the CPS Unit Key is still Content Binding, all files (i.e. CPS Unit Usage File, CPS Unit Key File and Content Certificate) are identical for all users. This case might be useful to update usage rules corresponding to a time after the packaged media is released. The users, who have the media, might be able to receive additional content without charge.

For the case that the binding of the CPS Unit Key is changed to another type of binding, CPS Unit Key File is different for each user. This means that the Remote Server shall return a different CPS Unit Key File for each user. This case might be useful to update usage rules based on a charge to each user.

Figure 4-9 shows how to realize this example.

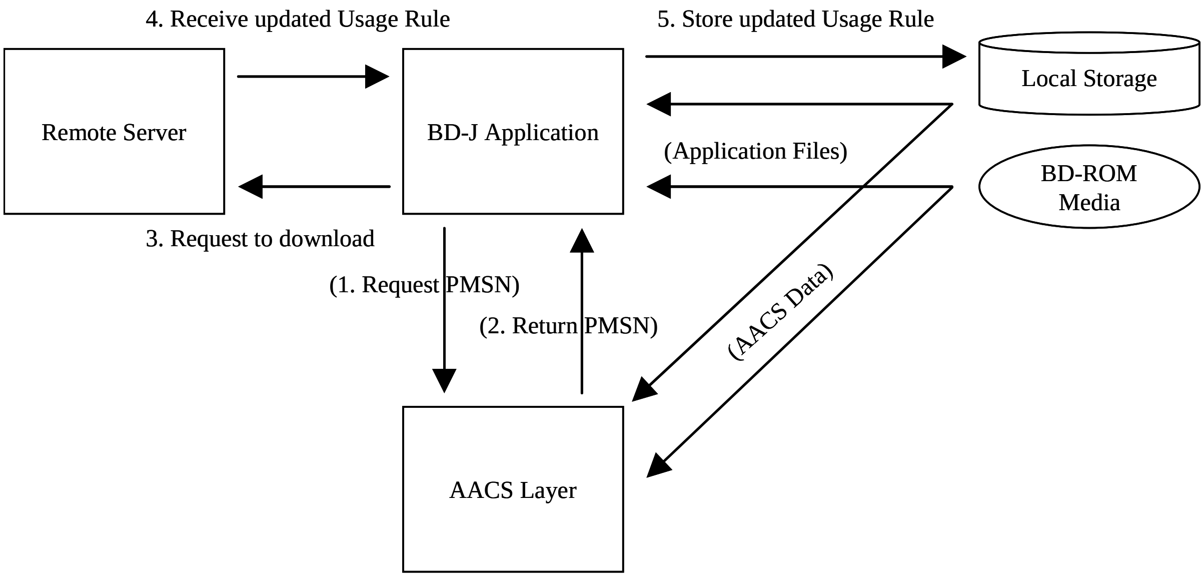
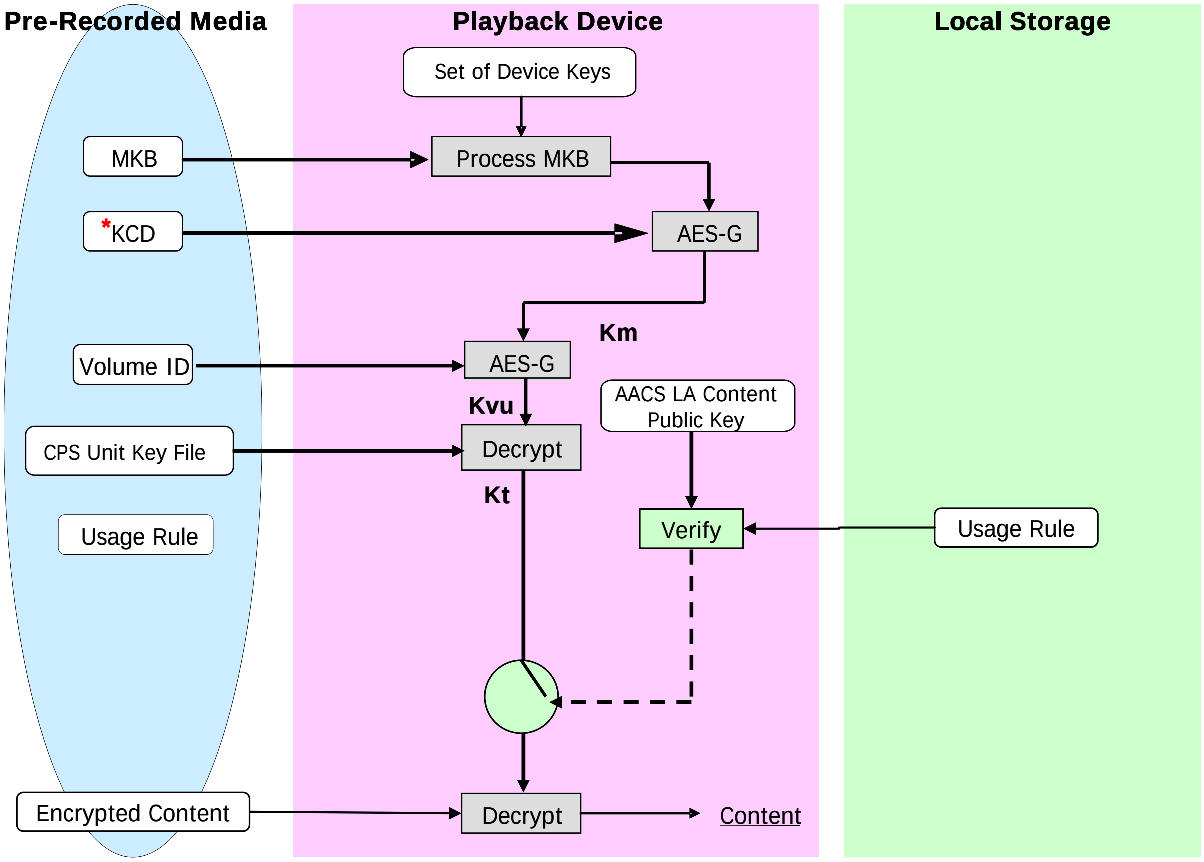


Figure 4-9 How to realize Download updated Usage Rule

To realize the first case of the examples, it is not necessary to utilize on-line functionality of the AACS Layer. The BD-J Application requests to download an updated Usage Rule to a Remote Server and stores it into the Binding Unit Data Area of Local Storage.

To realize the second case, it is necessary to utilize on-line functionality of the AACS Layer. Pre-recorded Media Serial Number is required to bind the Title Key to a specific media. Method defined in Section 4.4.1.2.2.2 of this specification is utilized by BD-J for this purpose.

Figure 4-10 describes a decryption overview for the BD-ROM and the Binding Unit Data Area in case of Download updated Usage Rule.



*KCD is used by only certain classes of devices.

Figure 4-10 Decryption Overview for BD-ROM and Binding Unit Data Area (2)

4.6.3 Download CPS Unit Key

In this example, CPS Unit Key is downloaded and stored into the Binding Unit Data Area of Local Storage. There are two cases for this example. One case is that the binding of the CPS Unit Key is Content Binding, and the second case is that the binding of the CPS Unit Key is not Content Binding.

Figure 4-11 shows the directory structure of this example. For both cases, directory structure is identical.

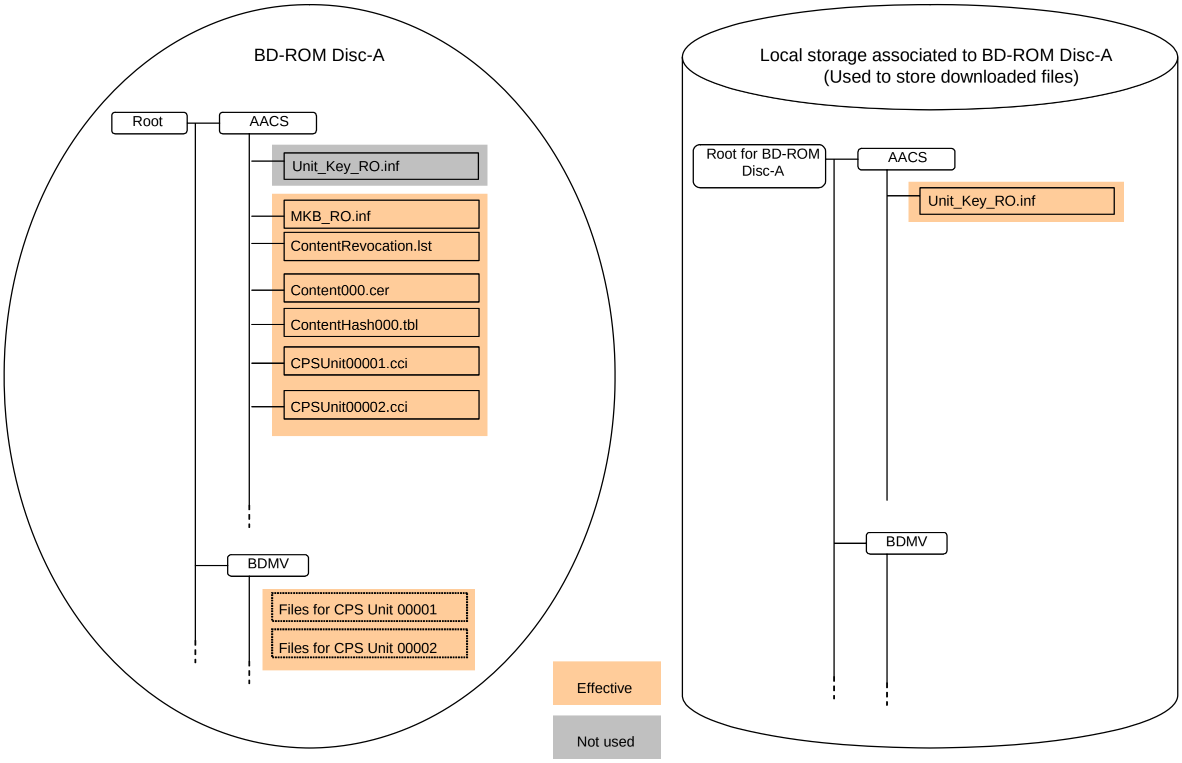


Figure 4-11 Example: Download CPS Unit Key

All files under BDMV directory and AACS directory are pre-recorded on the media, and only one file to be downloaded is CPS Unit Key File. The Original CPS Unit Key File on the BD-ROM might have the CPS Unit Key only for CPS Unit 00001. This means that a Title in the CPS Unit 00002 cannot be played back without downloading an updated CPS Unit Key. Downloading a CPS Unit Key File might have a CPS Unit Key for all CPS Units. Then, all Titles in the media can be played back with this downloaded CPS Unit Key File.

For the case that the binding of the CPS Unit Key is Content Binding, this downloaded CPS Unit Key File is identical for all users. This case might be useful to unlock the content in timely manner without charge.

For the case that the binding of the CPS Unit Key is not Content Binding, CPS Unit Key File is different for each user. This means that the Remote Server shall return different CPS Unit Key File for each user. This case might be useful to unlock the content based on the charge to each user.

Figure 4-12 shows how to realize this example.

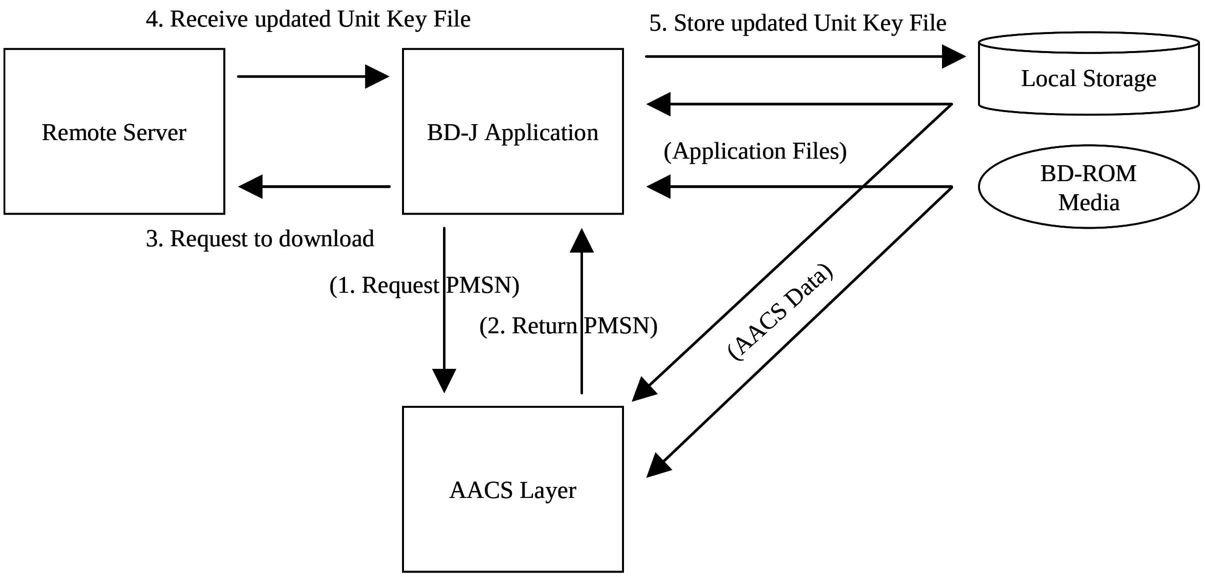
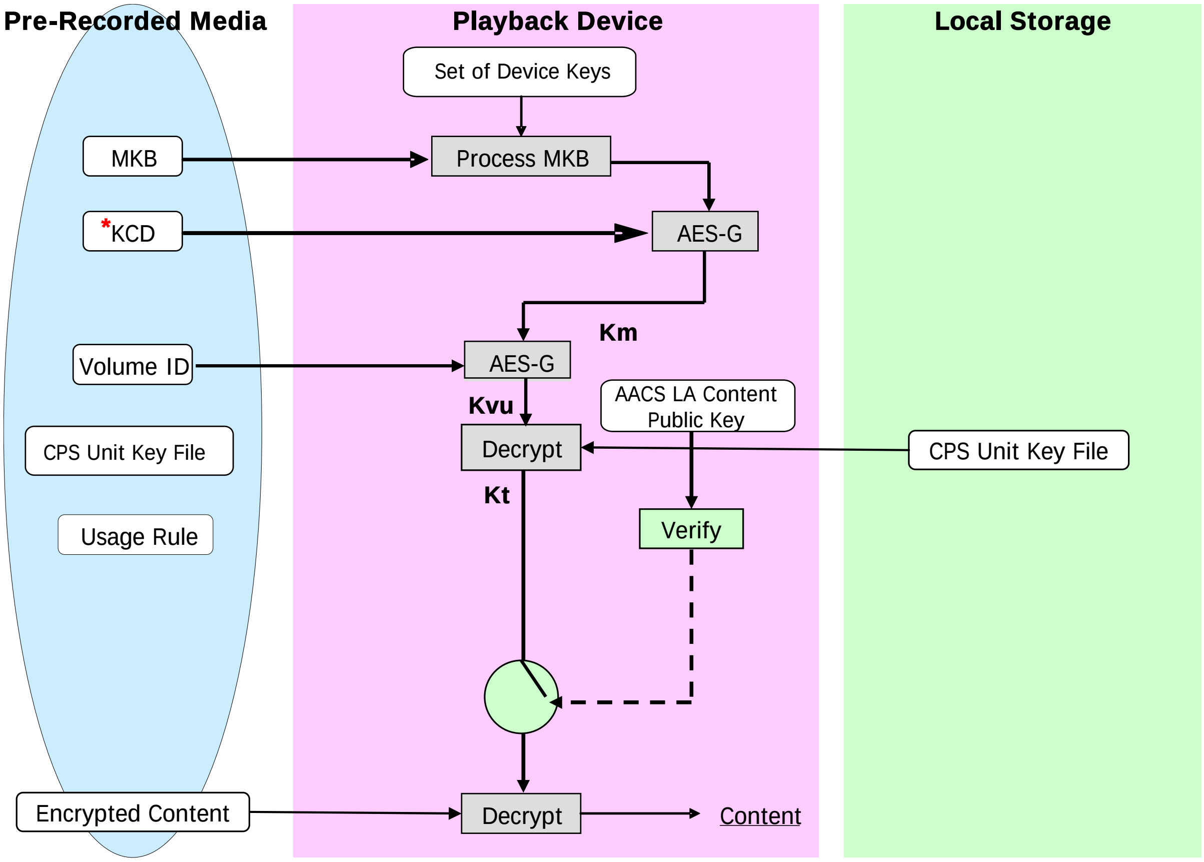


Figure 4-12   How to realize Download Title Key

To realize former case of this example, it is not necessary to utilize on-line functionality of the AACS Layer. This example can be realized only by BD-J. BD-J Application requests to download an updated CPS Unit Key File to Remote Server and stores it into the Binding Unit Data Area of Local Storage.

To realize the later case of this example, it is necessary to utilize on-line functionality of the AACS Layer. A Pre-recorded Media Serial Number is required to bind the CPS Unit Key to a specific media. Method defined in Section 4.4.1.2.2.2 of this specification is utilized by BD-J for this purpose.

Figure 4-13 describes a decryption overview for the BD-ROM and the Binding Unit Data Area of Local Storage in case of download CPS Unit Key.



*KCD is used by only certain classes of devices.

Figure 4-13 Decryption Overview for BD-ROM and Binding Unit Data Area (3)

4.6.4 Download Permission

In this example, Permission is downloaded and is stored if Permission is set as Cacheable. Permission may be stored into the Binding Unit Data Area of Local Storage as one example of implementation. Different from other examples, this example does not utilize the concept of VFS

Figure 4-14 shows how to realize this example.

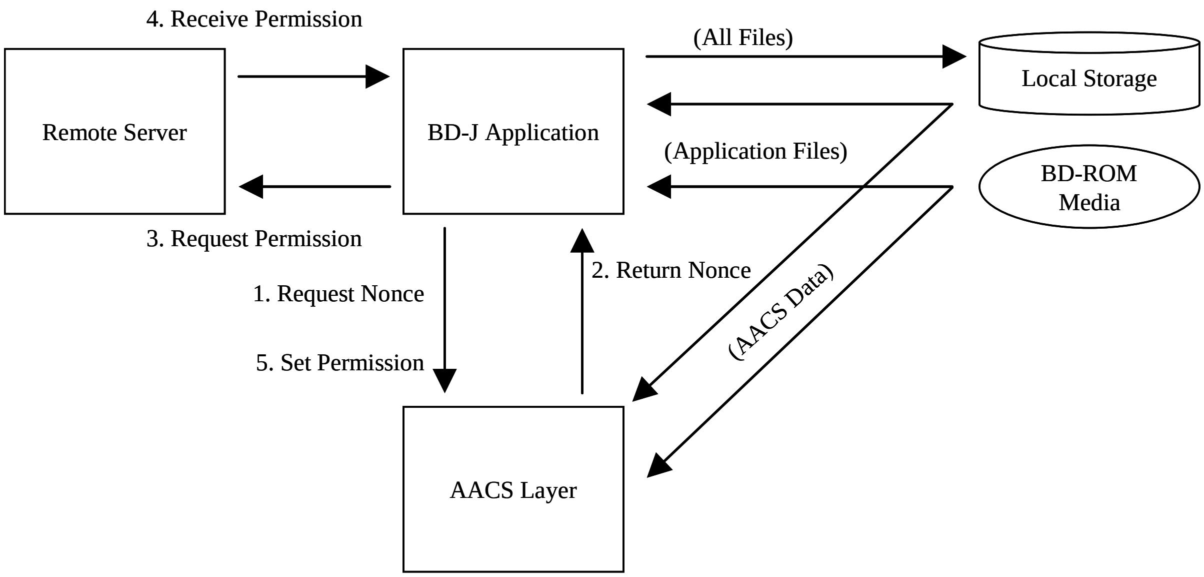


Figure 4-14 How to realize Download Permission

To realize this example, method defined in Section 4.4.1.2.1 and Class defined in Section 4.4.1.3 of this specification are utilized by BD-J. Here is a sequence of procedure.

1. Request Volume ID and Nonce
    - BD-J Application creates the instance of the class defined in Section 4.4.1.3 of this specification by use of the constructor defined in Section 4.4.1.5.1.1 of this specification with a specific title_id
    - BD-J Application request to notify Nonce by use of the method defined in Section 4.4.1.5.2.1 of this specification
    - BD-J Application optionally requests to notify Volume ID by use of the method defined in Section 4.4.1.2.2.1 of this specification
    - BD-J Application optionally requests to notify Pre-recorded Media Serial Number by use of the method defined in Section 4.4.1.2.2.2 of this specification
2. Return Volume ID and Nonce
    - AACS Layer generates random value as Nonce, and store it temporally
    - AACS Layer retrieves Volume ID and Pre-recorded Serial Number from the media
    - BD-J Application receives the responses (Nonce, Volume ID and Pre-recorded Media Serial Number) from AACS Layer
3. Request Permission
    - BD-J Application sends a request of Permission to Remote Server.
    - At least, Nonce received from AACS Layer needs to be sent to Remote Server.
    - Optionally, BD-J Application may send the Volume ID and Pre-recorded Media Serial Number, which are received from AACS Layer.
    - Optionally, BD-J Application may send the title_id, which is descrbed in the BD-J Application itself.

- Optionally, BD-J Application may send the User ID and password, which is inputted by user via the user interface displayer by BD-J itself.
- TLS or other proprietary secure authenticated channel may be used for this transaction.

4. Receive Permission
    - BD-J Application receives a Permission from Remote Server
5. Set Permission
    - BD-J Application sets the received Permission to AACS Layer, then AACS Layer verify the Permission with temporally stored Nonce
    - AACS Layer may cache the Permission

Once Permission is set into the AACS Layer, BD-J Application may start the playback of the Title corresponding to the Permission. Before BD-J Application sends the request of Permission to the Remote Server, BD-J Application may query the existence of cached Permission to AACS Layer by use of the method defined in Section 4.4.1.5.2.3.

This page is intentionally left blank.

This page is intentionally left blank.

Chapter 5
Managed Copy of Pre-recorded Content

5. Introduction

The information related to the Managed Copy functionality specified in Chapter 5 of *AACS Pre-recorded Video Book* of this specification. This chapter describes additional definition of interface and structure related to Managed Copy for the use with BD-ROM Media and Application Format.

5.1 System Model

As an overview, the Managed Copy System based on AACS and BD-ROM application format consists of four modules; BD-J Application, Managed Copy object, Managed Copy Machine and Managed Copy Server. Figure 5-1 shows the relation between these three modules.

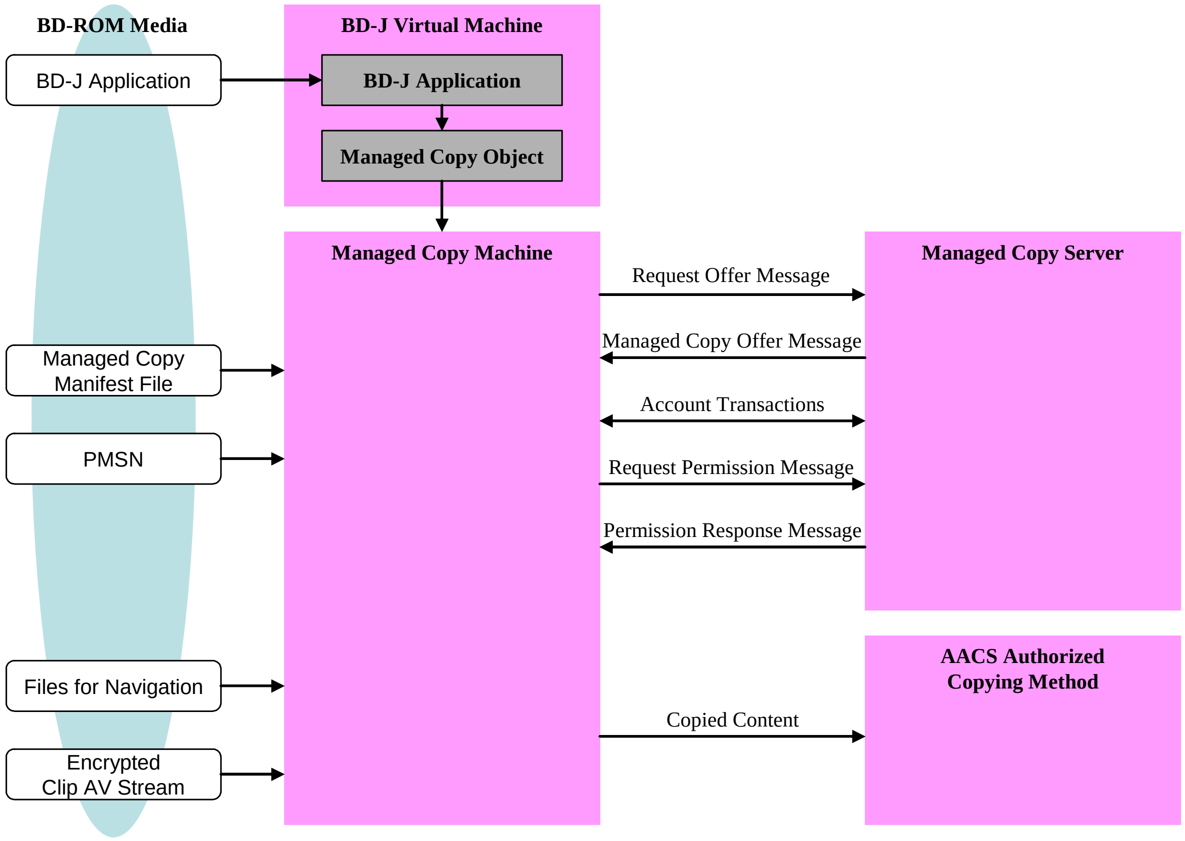


Figure 5-1 Managed Copy System Model: Relation between three modules

BD-J Application uses a Managed Copy Object to control the behavior of Managed Copy Machine. An API for Managed Copy Object is defined in Section 5.2. Using this API, BD-J Application can know the availability of Managed Copy Machine and request to invoke a Managed Copy Machine. BD-J Application optionally requests the account transaction or its result to Managed Copy Machine. Note that How Managed Copy Object controls Managed Copy Machine is player implementation specific.

The role of Managed Copy Machine and Managed Copy Server and the message between them are defined in Chapter 5 of *AACS Pre-recorded Video Book* of this specification. BD specific version of Managed Copy Manifest File, web service description, and Permission Response Message are defined in 5.3 and 5.4.

5.2 APIs between Managed Copy Machine and BD-J Application

The normative API to facilitate the initiation of a Managed Copy is defined in Section 5.2 of *AACS Pre-recorded Video Book* of this specification. This section provides the list of APIs that BD-J Application can initiate the Managed Copy Machine. In addition, API to control an accounting transaction is defined in Annex C as an interface. When a Managed Copy Machine provides the control interface for an accounting transaction, Managed Copy Object shall support this interface. Further requirement and recommendation for Managed Copy API implementation and BD-J application are defined in Annex D.

5.2.1 Package com.aacsla.bluray.mc

5.2.1.1 Class Summary

ManagedCopy

The ManagedCopy handles ManagedCopy functions required by AACS.

5.2.1.2 Class ManagedCopy

```
java.lang.Object
   |
   +--com.aacsla.bluray.mc.ManagedCopy
```

public class **ManagedCopy**

extends java.lang.object

The ManagedCopy handles ManagedCopy functions required by AACS.

5.2.1.2.1 Constructors

5.2.1.2.1.1 Managed Copy

public **ManagedCopy** ( )

Create ManagedCopy object.

5.2.1.2.2 Methods

5.2.1.2.2.1 IsMCMSupported

public boolean **IsMCMSupported**( )

Return the capability to support Managed Copy Machine function.

Returns:

the capability to support Managed Copy Machine function.

true: Managed Copy is supported in the system.

false: Managed Copy is not supported in the system.

5.2.1.2.2.2 InvokeMCM

public void **InvokeMCM**( )

Invoke Managed Copy Machine function.

5.3 Managed Copy Manifest File

The Managed Copy Manifest File “mcmf.xml” shall be stored in the “\AACS” directory and in the “\AACS\DUPLICATE” directory if a BD-ROM disc is made ready for Managed Copy or for the on-line transaction. The Managed Copy Manifest File defines the list of files which enables the Managed Copy Machine to identify the necessary files to process Managed Copy of each Managed Copy Unit (MCU).

The following XML description is the example of Managed Copy Manifest File.

```
<?xml version="1.0" encoding="UTF-8"?>
<mcmfManifest xmlns="http://www.aacsla.com/2006/02/bdmcManifest"
contentID="0x00000000000000000000000000000001">
        <URIList>
                <URI>http://example.com/ManagedCopy/00000001/</URI>
                <URI>http://example.net/ManagedCopy/00000001/</URI>
        </URIList>
        <MCUALL>
                <DirectoryName>"BDMV"</DirectoryName>
        </MCUALL>
        <MCUPARTIAL ID="0x0001">
                <FileName>"BDMV/PLAYLIST/00000.mpls"</FileName>
                <FileName>"BDMV/CLIPINF/00000.clpi"</FileName>
                <FileName>"BDMV/STREAM/00000.m2ts"</FileName>
                <FileName>"BDMV/BDJO/00000.bdjo"</FileName>
                <FileName>"BDMV/JAR/00000.jar"</FileName>
        </MCUPARTIAL>
</mcmfManifest>
```

5.3.1 Rules to use Managed Copy Manifest File

To use Managed Copy Manifest File information, the following behaviors are required in Managed

Copy Machine.

- Managed Copy Machine uses the URI information from the first URI to the last URI. The latter URI can be used only the case the prior URI has the problem to be used for Managed Copy.
- When “DirectoryName” is listed in a MCU, all files in the indicated directory can be used for Managed Copy. (In the example in Section 5.2.1, all files in BDMV directory can be used for the Managed Copy of “MCUALL”.)
- The BD-J Root Certificate file is recorded in CERTIFICATE directory under root directory. BD-J Root Certificate file can be used in the managed copy process if necessary.

5.3.2 XML schema of Managed Copy Manifest File

The Managed Copy Manifest File is an XML File.
The Managed Copy Manifest File XML Schema is defined as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.aacsla.com/2006/02/bdmcManifest"
    xmlns:bdmcmf="http://www.aacsla.com/2006/02/bdmcManifest"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified" attributeFormDefault="unqualified">

    <xs:element name="mcmfManifest">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="bdmcmf:URIList" minOccurs="0" maxOccurs="1" />
                <xs:element ref="bdmcmf:MCUALL" minOccurs="0" maxOccurs="1" />
                <xs:element ref="bdmcmf:MCUPARTIAL" minOccurs="0" 
maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="contentID" type="bdmcmf:contentIDType" use="required" />
        </xs:complexType>
    </xs:element>

    <xs:element name="URIList">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="URI" minOccurs="0" maxOccurs="16">
                    <xs:simpleType>
                        <xs:restriction base="xs:anyURI">
                            <xs:maxLength value="1024"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="MCUALL">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="DirectoryName" minOccurs="0"
maxOccurs="unbounded">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:maxLength value="1024"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="MCUPARTIAL">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="FileName" minOccurs="0" maxOccurs="unbounded">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:maxLength value="1024"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="ID" type="bdmcmf:IDType" use="required"/>
        </xs:complexType>
    </xs:element>

    <xs:simpleType name="IDType" final="restriction">
```

```
        <xs:restriction base="xs:string">
            <xs:pattern value="(0x([0-9]|[a-f]|[A-F])+)"/>
            <xs:length value="6" fixed="true"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="contentIDType" final="restriction">
        <xs:restriction base="xs:string">
            <xs:pattern value="(0x([0-9]|[a-f]|[A-F])+)"/>
            <xs:maxLength value="34" fixed="true"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

(Note) FileName and DirectoryName shall indicate only the files and Directories that are actually recorded in the BD-ROM Medium.

5.4 Managed Copy Web Service

Managed Copy web service and the message used in this service that are specific to Blu-ray Disc Pre-Recorded Media are defined in this section.

5.4.1 Web Service Description

Managed Copy web service description is used for communication between the MCM and the MCS. Managed Copy web service description for Blu-ray Disc Pre-recorded Media is defined in this section based on Managed Copy web service description defined in Appendix C of *AACS Pre-recorded Video Book* of this specification.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions
    targetNamespace="http://www.aacsla.com/2006/02/managedCopyService"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
    xmlns:tns="http://www.aacsla.com/2006/02/managedCopyService"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:bdmcp="http://www.aacsla.com/2006/02/bdmcPermission"
    xmlns:aacsoffer="http://www.aacsla.com/2006/02/managedOffer"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

    <wsdl:documentation>Managed Copy Web Service</wsdl:documentation>

    <wsdl:types>

        <xs:schema elementFormDefault="qualified"
            targetNamespace="http://www.aacsla.com/2006/02/managedCopyService">
            <xs:import
                namespace="http://www.aacsla.com/2006/02/bdmcPermission"
                schemaLocation="aacs_bdmanaged_permission.xsd" />
            <xs:import
                namespace="http://www.aacsla.com/2006/02/managedOffer"
                schemaLocation="aacs_copy_offer.xsd" />

            <xs:element name="RequestOffers">
                <xs:complexType>
                    <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1"
                name="cid" type="xs:string" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="mcotList" type="tns:ArrayOfMCOTsArrayOfString" />
            <xs:element minOccurs="0" maxOccurs="1"
                name="SerialNumber" type="xs:string" />
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:complexType name="ArrayOfString">
    <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded"
            name="string" nillable="true" type="xs:string" />
    </xs:sequence>
</xs:complexType>
<xs:element name="RequestOffersResponse">
    <xs:complexType>
        <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1"
                ref="aacsoffer:offers" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="RequestPermission">
    <xs:complexType>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1"
                name="cid" type="xs:string" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="sessionID" type="xs:string" />
            <xs:element minOccurs="0" maxOccurs="1"
                name="mcotInfo" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="mcmNonce" type="xs:string" />
        </xs:sequence>
    </xs:complexType>
```

```
        </xs:element>
        <xs:element name="RequestPermissionResponse">
            <xs:complexType>
                <xs:sequence>
                    <xs:element minOccurs="0" maxOccurs="1"
                        ref="bdmcp:permission" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:schema>
</wsdl:types>
<wsdl:message name="RequestOffersSoapIn">
    <wsdl:part name="parameters" element="tns:RequestOffers" />
</wsdl:message>
<wsdl:message name="RequestOffersSoapOut">
    <wsdl:part name="parameters"
        element="tns:RequestOffersResponse" />
</wsdl:message>
<wsdl:message name="RequestPermissionSoapIn">
    <wsdl:part name="parameters" element="tns:RequestPermission" />
</wsdl:message>
<wsdl:message name="RequestPermissionSoapOut">
    <wsdl:part name="parameters"
        element="tns:RequestPermissionResponse" />
</wsdl:message>
<wsdl:portType name="ServiceSoap">
    <wsdl:operation name="RequestOffers">
        <wsdl:input message="tns:RequestOffersSoapIn" />
        <wsdl:output message="tns:RequestOffersSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="RequestPermission">
        <wsdl:input message="tns:RequestPermissionSoapIn" />
        <wsdl:output message="tns:RequestPermissionSoapOut" />
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ServiceSoap" type="tns:ServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
```

```
        <wsdl:operation name="RequestOffers">
            <soap:operation
                soapAction="http://www.aacsla.com/2006/02/managedCopyService/RequestOffers"
style="document" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="RequestPermission">
            <soap:operation
                soapAction="http://www.aacsla.com/2006/02/managedCopyService/RequestPermission"
style="document" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="ServiceSoap12" type="tns:ServiceSoap">
        <soap12:binding
            transport="http://schemas.xmlsoap.org/soap/http" />
        <wsdl:operation name="RequestOffers">
            <soap12:operation
                soapAction="http://www.aacsla.com/2006/02/managedCopyService/RequestOffers"
style="document" />
            <wsdl:input>
                <soap12:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap12:body use="literal" />
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="RequestPermission">
```

```
            <soap12:operation
                soapAction="http://www.aacsla.com/2006/02/managedCopyService/RequestPermission"
style="document" />
            <wsdl:input>
                <soap12:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap12:body use="literal" />
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
</wsdl:definitions>
```

5.4.2 Permission Response Message

Permission Response Message is a web service message as defined in the Appendix B of *AACS Pre-recorded Video Book* of this specification, using the Managed Copy Permission Schema. This chapter defines a Managed Copy Permission Schema specifically for Blu-ray Disc Pre-Recorded Media.

```
<?xml version="1.0"?>
<xs:schema
    targetNamespace="http://www.aacsla.com/2006/02/bdmcPermission"

    xmlns:bdmcp="http://www.aacsla.com/2006/02/bdmcPermission"
    xmlns:bdmcmf="http://www.aacsla.com/2006/02/bdmcManifest"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">
    <xs:import namespace="http://www.aacsla.com/2006/02/bdmcManifest"
        schemaLocation="bdmcmanifest.xsd" />

    <xs:element name="permission">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="permissionSignedContent">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="bdmcp:status" minOccurs="1"
                                maxOccurs="1" />
```

```
              <xs:element ref="bdmcp:mcmNonce" minOccurs="1"
                maxOccurs="1" />
              <xs:element ref="bdmcmf:mcmfManifest"
                minOccurs="0" maxOccurs="1" />
              <xs:element ref="bdmcp:mcotInfo" minOccurs="0"
                maxOccurs="1" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element ref="bdmcp:MCScert" minOccurs="1"
          maxOccurs="1" />
        <xs:element ref="bdmcp:signature" minOccurs="1"
          maxOccurs="1" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="status" type="xs:boolean" />
  <xs:element name="mcmNonce" type="xs:string" />
  <xs:element name="mcotInfo" type="xs:base64Binary" />
  <xs:element name="MCScert" type="xs:base64Binary" />
  <xs:element name="signature" type="xs:base64Binary" />
</xs:schema>
```

Note: In order to minimize the player burden for canonicalization, Permission Response Message shall be canonicalized in the Managed Copy Server as UTF-8 bytes according to the Exclusive XML Canonicalization specification (http://www.w3.org/TR/xml-exc-c14n/#sec-Specification).

Chapter 6
Details for Sequence Keys

6. Introduction

Sequence Keys and Sequence Key Block are specified in Chapter 4 of the *Pre-recorded Video Book* of this specification. This chapter describes additional details of Sequence Keys for BD-ROM disc and Application Format.

BD-ROM applies the multiple PlayList approach and 256 PlayLists are used per a Sequence Key Block for Sequence Key purpose.

The Segment Keys are used for encrypting the Sequence Key segment portion in Clip AV stream file and are stored in the Segment Key File.

6.1 PlayList approach for Sequence Keys

A BD-ROM disc has at most six Sequence Key Blocks and 1024 Variant Data per SKB. The Variant Number is calculated from each Sequence Key Block. The Variant Number is used to determine the PlayList_id of the PlayList to be played back. Each PlayList contains a set of PlayItems for SK segment and non-SK portion and each PlayItem for SK segment portion points out to one of the SK variations for that SK segment.

Figure 6-1 describes an overview of PlayList approach for Sequence Keys.

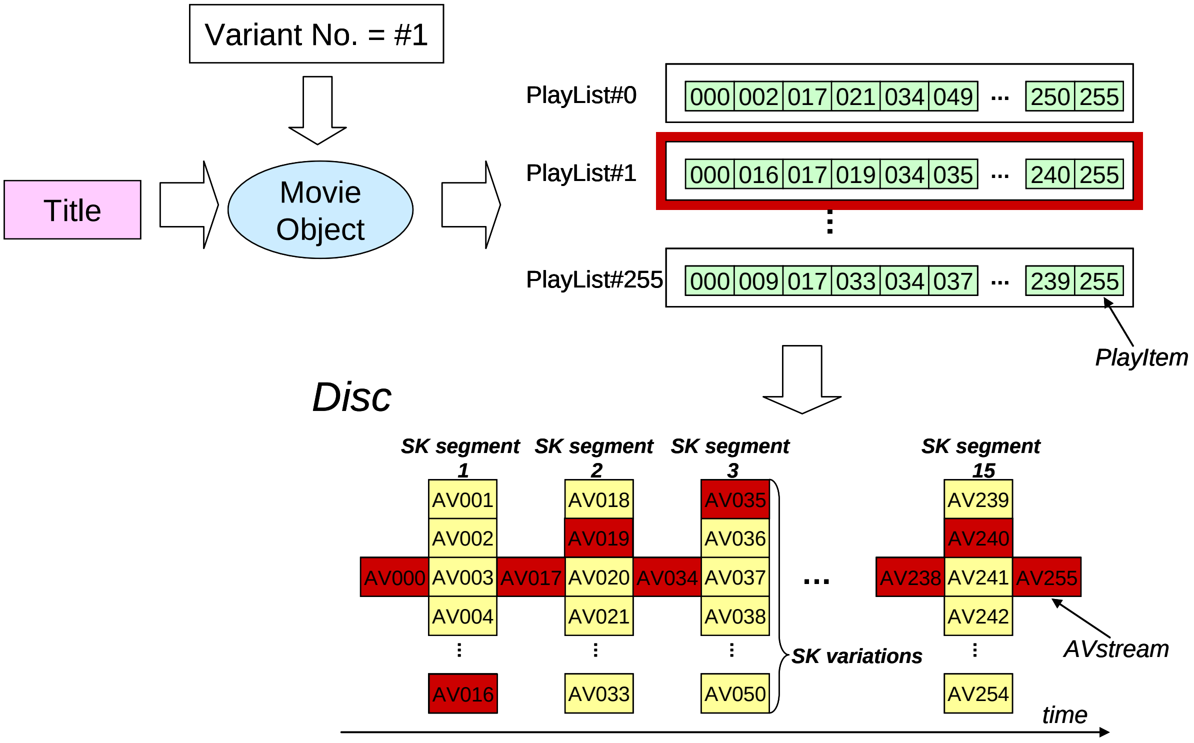


Figure 6-1 Overview of PlayList approach for Sequence Keys

Each Clip AV stream referred from PlayItem (AV000, AV001, AV002, …, AV255) is recorded as an individual Clip AV stream file and each SK segment portion (AV001, AV002, AV003, …, AV254) is encrypted by a different Segment Key.

(Note 1) At least one Clip AV stream of non-SK portion shall be allocated between SK segment i and SK segment (i + 1).

(Note 2) Sequence Keys are applicable for only main TS and are not applicable for sub TS.

6.2 Playback process for BD-ROM Player

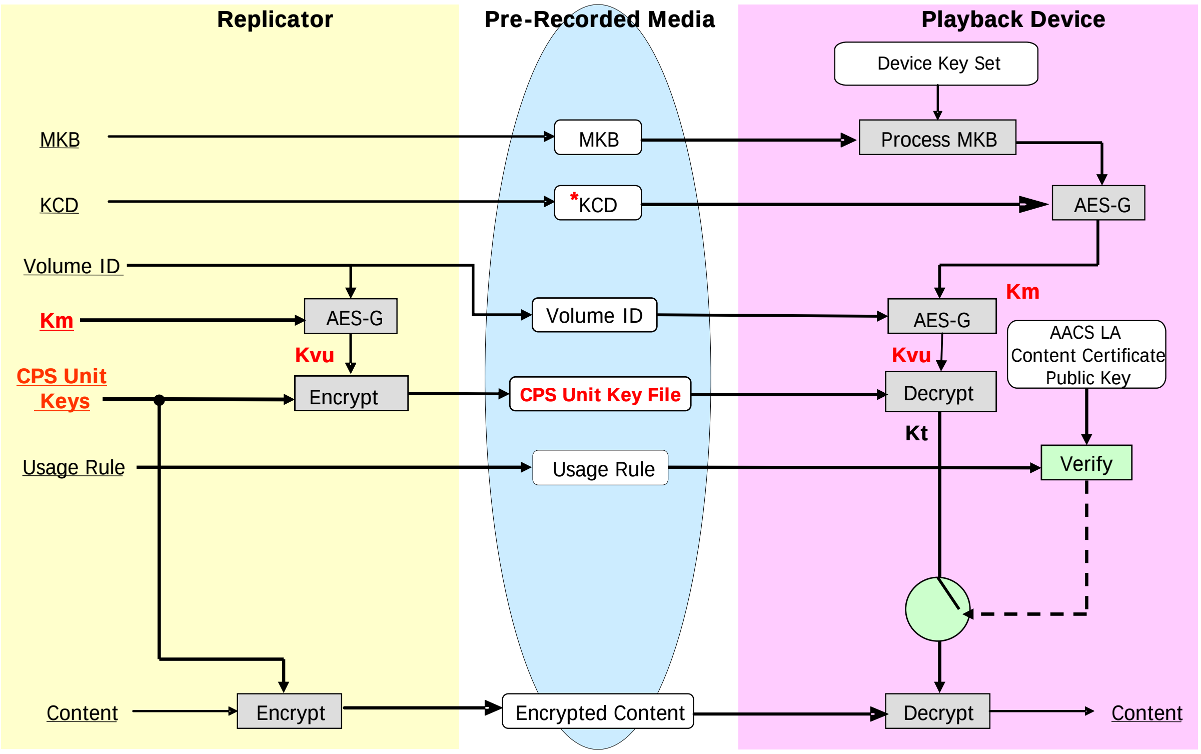
6.2.1 Encryption and Decryption Overview

This section describes the encryption and decryption process for (a) SK segment and (b) non-SK portion on the BD-ROM Disc on which the SKB is assigned. The Sequence Key Block Files “SKB1.inf”, “SKB2.inf”, “SKB3.inf”, “SKB4.inf”, “SKB5.inf” and “SKB6.inf” shall be recorded in the “\AACS” directory and in the “\AACS\DUPLICATE” directory. In case of the BD-ROM disc on which the SKB is assigned, the number of the SKB shall be between one and six and the index of SKB file name shall be defined in continuous order, starting from one. For example, in case of three SKBs are assigned on the BD-ROM disc, the SKB1.inf, SKB2.inf and SKB3.inf shall be recorded on the disc.

SKB data shall be recorded from the first byte of the file, and the null (00_{16}) padding may be attached after the SKB data in the file for the authoring and the mastering purpose.

On the other hand, for the BD-ROM disc on which the SKB is not assigned, Process SKB is omitted and the Volume Unique Key is used instead of the Volume Variant Unique Key. In this case, the Sequence Key Block and the Segment Key file are not recorded on the disc.

Figure 6-2 describes an encryption and decryption overview for the BD-ROM disc on which the SKB is not assigned.



*KCD is used by only certain classes of devices.

Figure 6-2 Encryption and Decryption Overview for BD-ROM on which SKB is not assigned

6.2.1.1 Key Hierarchy for SK segment portion

For the SK segment, the Segment Key is used for encrypting instead of the CPS Unit Key. 240 (16 variations * 15 segments) Segment Keys are used for one SKB and these keys are recorded in the Segment Key File.

Figure 6-3 describes an encryption and decryption overview for the SK segment portion on the BD-ROM disc on which the SKB is assigned.

Figure 6-3

Figure 6-3 Encryption and Decryption Overview for SK segment portion

6.2.1.2 Key Hierarchy for non-SK portion

For the non-SK portion which means that it is not the SK segment portion, the CPS Unit Keys are used for encrypting instead of the Segment Key.

Figure 6-4 describes an encryption and decryption overview for the non-SK portion on the BD-ROM disc on which the SKB is assigned.

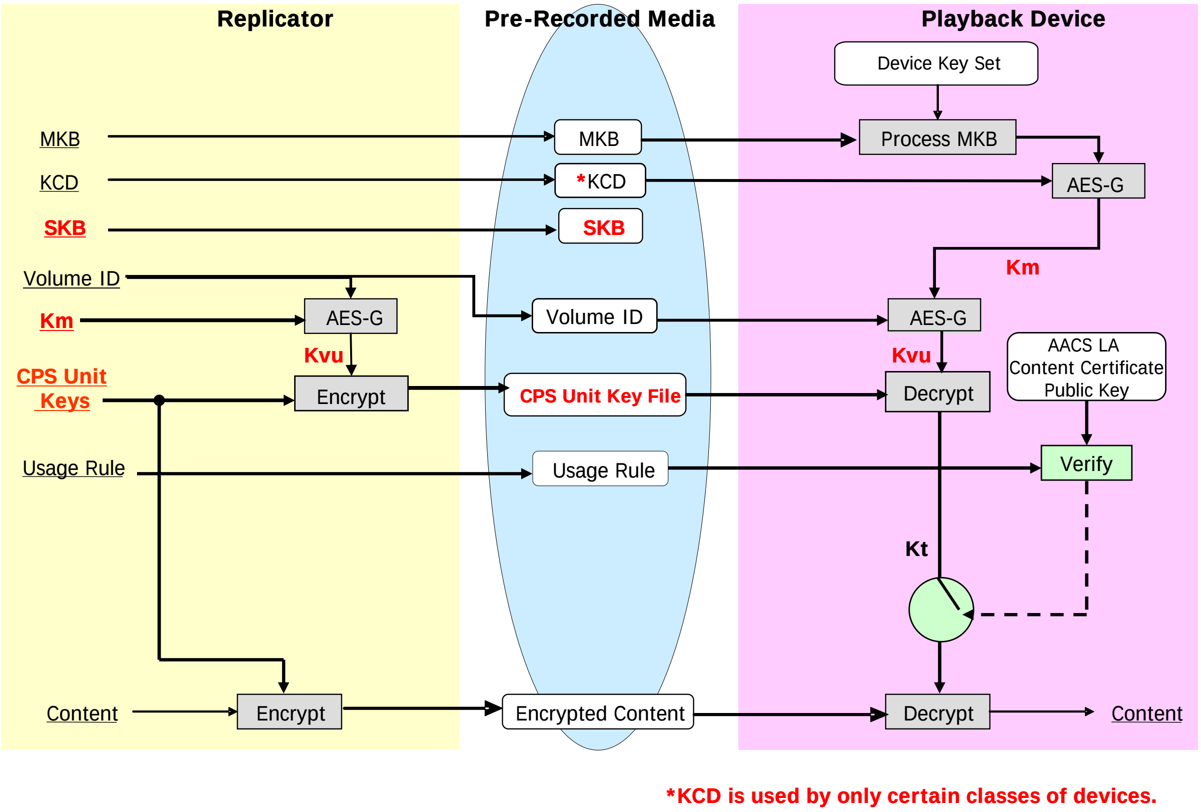


Figure 6-4 Encryption and Decryption Overview for non-SK portion

6.2.2 Selection process of a PlayList

The BD player selects a proper PlayList to be player back by using a Movie Object for Title defined in Section 3.9.1.8 of this specification.

(Note) The assignment of the Player Status Registers for the Playlist_Indicator is PSR96 and PSR97.

This is the example of the Movie Object programmed a PlayList selection for one SKB. This example assumes that PlayList_id #0 to #255 are assigned for SKB1.

```
MovieObject(){
        Number_of_navigation_commands (=4);
        Move[GPR#Y][PSR96];
```

```
            And[GPR#Y][0xFF000000];
            Shift Right[GPR#Y][0x18];
            PlayPL[GPR#Y];
    }
```

For example, the Movie Object for PlayList selection includes “Number_of_navigation_commands” and “PlayPL”.

“Number_of_navigation_commands” indicates the number of navigation_command structures that are contained with the Movie Object( ).

“PlayPL (PlayList_id = PSR)” commands the playback of PlayList#(PlayList_id). Note that each PlayPL for each SKB shall not command the playback of the same PlayList#(PlayList_id). In other words, for six SKBs, at least 1536 PlayLists are necessary.

This is the example of the Movie Object programmed a PlayList selection for two SKBs. This example assumes that PlayList_id #0 to #255 and PlayList_id #256 to #511 are assigned for SKB1 and SKB2 respectively.

```
    MovieObject(){
             Number_of_navigation_commands (=9);
             Move[GPR#Y][PSR96];
            And[GPR#Y][0xFF000000];
            Shift Right[GPR#Y][0x18];
            PlayPL[GPR#Y];

             Move[GPR#Y][PSR96];
            And[GPR#Y][0x00FF0000];
            Shift Right[GPR#Y][0x10];
            Add[GPR#Y][0x100];
            PlayPL[GPR#Y];
    }
```

“PSR” is the Player Status Register, which can be stored a fixed length variable. The PlayList Indicator for each SKB derived from the PlayList_id is set to the PSR.

Figure 6-5 describes an example of the data format of PSR for Sequence Key purpose. Playlist_Indicator #1, Playlist_Indicator #2, Playlist_Indicator #3, … and Playlist_Indicator #6 corresponds to “SKB1.inf”, “SKB2.inf”, “SKB3.inf”, …, and “SKB6.inf” respectively. These Playlist_Indicators are computed as follows:

Playlist_Indicator #i = PlayList_id #i mod 256 (i = 1, 2, 3, …, 6)

where PlayList_id #i denotes the PlayList_id corresponding the SKBi.

PSR96	PSR97
Playlist_Indicator #1(b31-b24)	Playlist_Indicator #5(b31-b24)
Playlist_Indicator #2(b23-b16)	Playlist_Indicator #6(b23-b16)
Playlist_Indicator #3(b15-b8)	Reserved (b15-b8)
Playlist_Indicator #4(b7-b0)	Reserved (b7-b0)

Figure 6-5 Data format of PSR

6.3 Segment Key File

Each SK segment portion is encrypted by the Segment Key and each Segment Key is encrypted by the Volume Variant Unique Key. The Volume Variant Unique Key is defined for each PlayList, in other words, 1024 Volume Variant Unique Keys are used for encrypting the Segment Keys per one Sequence Key Block. The Segment Key File "Segment_Key.inf" shall be recorded in the "\AACS" directory and in the "\AACS\DUPLICATE" directory.

Table 6-1 shows the data format of the Segment Key File.

Table 6-1 Data Format of Segment Key File

Syntax	No. of bits	Mnemonic
Segment_Key_File(){		
Num_of_SKB	16	
For(I=0; I < Num_of_SKB; I++){		
For(J=0; J < 1024; J++){		
PlayList_id (= X)	16	
For(K=0; K < 15; K++){		
PlayItem_id(X, K)	16	uimsbf
Encrypted Segment Key for PlayList/PlayItem(X, K)	128	uimsbf
}		
}		
}		
}		

Num_of_SKB indicates the number of Sequence Key Blocks on the BD-ROM disc.

PlayList_id indicates the PlayList for a particular Variant Number for a particular SKB.

PlayItem_id indicate the PlayItem assigned corresponding encrypted Segment Key.

Encrypted Segment Key for PlayList/PlayItem(X, K) contains the 16 bytes of the encrypted Segment Key for used for encrypting the PlayItem(X, K). The Segment Key is encrypted as follows:

AES_128E($K_{vvu\text{-}J}$(I, J), Segment Key(X, K))

where $K_{vvu\text{-}J}$ denotes a Volume Variant Unique Key defined in Section 3.3 of the *Pre-recorded Video Book* of this specification and corresponds to the Variant Number "J".

(Note) Different Segment Keys shall be assigned to different SK segment portions.

Calculation method for the Block key for SK segment portion is described in Figure 6-6.

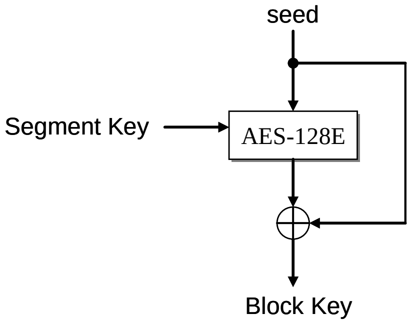


Figure 6-6 Calculation method for the Block Key from the Segment Key

Chapter 7
Clarifications for AACS unencrypted contents

7. Introduction

For the BD Prerecorded Disc (BD-ROM), both encrypted contents and unencrypted content can be recorded as AACS contents. This chapter describes details of disc structure for the BD-ROM containing AACS unencrypted contents.

7.1 Disc structure

CPS Unit structure defined in Section 3.9 shall be also applied for AACS unencrypted contents, i.e. Usage Rule shall be also defined for AACS unencrypted contents. Encrypted portion and unencrypted portion cannot be mixed in one CPS Unit, because a Usage Rule is constant in one CPS Unit.

7.1.1 CPS information files for AACS unencrypted contents

This section describes clarifications for the necessary CPS information files for BD-ROM composed of the followings:

- only unencrypted content(s)
- both encrypted content(s) and unencrypted content(s).

7.1.1.1 BD-ROM composed of only unencrypted contents

BD-ROM that contains unencrypted contents and does not contain any encrypted contents shall store all the CPS information files defined in Figure 3-4. In other words, such BD-ROM shall store the followings:

- CPS Unit Key File
- MKB and Read/Write MKB
- Sequence Key Block (Optional)
- Segment Key File (Optional)
- Managed Copy Manifest File (Optional)
- Content Revocation List
- Content Certificate
- Content Hash Table
- CPS Unit Usage File.

In addition to above CPS information files, Volume ID and Partial MKB shall be stored on BD-ROM.

For such BD-ROM, CPS Unit Key shall be assigned for all CPS Units (and the key is not used for decryption).

Note that for the BD-ROM that does not contain any encrypted contents, the Type 3 MKB defined in Section 3.2.5.1 of *Introduction and Common Cryptographic Elements* book of this specification shall be used as the MKB “MKB_RO.inf”.

The Content Certificate and the Content Hash Table are applied for all unencrypted Clip AV streams on such BD-ROM. A Licensed Player shall verify the Content Certificate and the Content Hash Table for unencrypted Clip AV stream.

7.1.1.2 BD-ROM composed of both encrypted contents and unencrypted contents

BD-ROM that contains both encrypted contents and unencrypted contents shall also store all the CPS information files clarified in Section 7.1.1.1. For such BD-ROM, the CPS Unit number for unencrypted contents shall be different from the number for encrypted contents.

For such BD-ROM, CPS Unit Key shall be assigned for all CPS Units (and some keys are used for decryption and others are not).

For the BD-ROM that contains both encrypted contents and unencrypted contents, the Type 3 or Type 4 MKB shall be used as the MKB “MKB_RO.inf”.

The Content Certificate and the Content Hash Table cover both encrypted and unencrypted contents. A Licensed Player shall verify the Clip AV stream regardless of encrypted or unencrypted.

7.2 Usage Rules for AACS unencrypted contents

Basic CCI for AACS defined in Section 3.9.4.2 shall contain the following Usage Rules for a CPS Unit composed of such contents:

- EPN: EPN-unasserted (=1_2)
- CCI: Copy Control Not Asserted (=00_2)
- Image_Constraint_Token: High Definition Analog Output in High Definition Analog Form (=1_2)
- Digital_Only_Token: Output of decrypted content is allowed for Analog/Digital Outputs (=0_2)
- APSTB: APS off (=000_2)
- Type_of_Title: Basic Title (=0_2)

Note that an Enhanced Title Usage for AACS defined in Section 3.9.4.3 of this specification shall not be used for such a CPS Unit.

7.3 Copy Permission Indicator for AACS unencrypted contents

Copy_permission_indicator defined in Section 3.10.2 of this specification indicates whether a corresponding Aligned Unit is encrypted or not and shall be set as shown in Table 7-1.

Table 7-1   Copy_permission_indicator

Copy_permission_indicator	**Meaning**
00_2	unencrypted
01_2	Reserved
10_2	Reserved
11_2	encrypted

For unencrypted contents, i.e. Copy_permission_indicator = 00_2, Licensed Player shall treat such contents according to the Usage Rules defined in Section 7.2 of this specification. If the Licensed Player encounters the packet with Copy_permission_indicator set to 10_2 or 01_2, the data shall be considered encrypted.

Annex A. Restriction on Data Allocation (Informative)

This annex includes the information for Authoring Facility.

AACS introduces the following restrictions on data allocation for ease of mastering and content hash verification. When the Authoring Facility makes the disc image, the Authoring Facility shall comply with these restrictions.

- All the extents of each Clip AV stream file shall be allocated with ascending order in physical layer.
- Each physical sector in an Aligned Unit shall be allocated contiguously on the BD-ROM disc.
- If a Clip AV stream file is recorded over both physical layers in dual-layer disc, the total size of extents for the Clip AV stream file recorded in layer 0 shall be multiple of a hash unit.

This page is intentionally left blank.

Annex B. Carriage of System Renewability Message

B.1 Introduction

This chapter describes the method to store the System Renewability Message (SRM) on the BD-ROM in the case where an SRM is to be stored on the BD-ROM.

B.2 SRM for DTCP

SRM for DTCP shall be stored as a file “DTCP.srm” in the root directory.

B.3 SRM for HDCP

SRM for HDCP shall be stored as a file “HDCP.srm” in the root directory.

This page is intentionally left blank.

Annex C. MCM Transaction for Managed Copy

If a Managed Copy Machine can be controlled by BD-J Application, it shall support the API described in this Annex.

C.1 Package com.aacsla.bluray.mt

C.1.1 Interface Summary

MCMTransaction

The MCMTransaction Interface allows notifying the status of financial and/or account transactions to Managed Copy Machine.

C.1.2 Interface MCMTransaction

public interface **MCMTransaction**

The MCMTransaction Interface allows notifying the status of financial and/or account transactions to Managed Copy Machine.

C.1.2.1 Fields

C.1.2.1.1 offers

public String **offers**

Provides the XML object "Offers" returned from the managed copy server using the Request Offer message. See 5.3.3 in the *AACS Pre-recorded Video Book* of this specification.

C.1.2.2 Methods

C.1.2.2.1 completeTransaction

public void **completeTransaction**(String coupon, String MCOT, String MCUi,
String status, String MCOTParams)

Notify the completion of Financial Transaction to Managed Copy Machine function.

Parameters:

coupon – A string uniquely identifying the financial or account transaction. If no financial or account transaction has been completed, Coupon must be a null string.

MCOT – A string identifier of the managed copy output technology selected for the managed copy, as defined in the AACS Compliance rules.

MCUi – An ID which identifies a particular offer that was selected as a part of transaction.

status – an optional string containing further information on the transaction. Informative: For example, if the transaction failed, Status may contain information about why that transaction failed.

MCOTParams – A string value with additional information specific to the managed copy output technology to be used in customization of MCOTInfo to be sent in the RequestPermission message.

Annex D. Requirements for On-line and Managed Copy API

This annex defines requirements and recommendations for On-line and Managed Copy APIs at BD-J specific aspect, in addition to Chapter 4 and Chapter 5 of this specification.

D.1. PSR31 value and VFS capability / BD-J network connectivity

Player Status Register 31 (PSR31) includes 4-bit Player Profile at the position of bit16~19. Contents shall check player's VFS capability and BD-J network connectivity by referring to Player Profile.

Following table explains the relationship between Player Profile value and VFS capability / BD-J network connectivity.

0000b and 1000b :	Player Profile for the player that does not have VFS capability nor BD-J network connectivity
0001b :	Player Profile for the player that has VFS capability, but does not have BD-J network connectivity
0011b :	Player Profile for the player that has both VFS capability and BD-J network connectivity

(Other values are reserved)

Note: Implementation of AACS On-line function is mandatory for all Licenced Players that has BD-J network connectivity. However, the capability of storing Cacheable Permission (i.e. Secure Clock implementation) is optional for such kind of Licensed Player. The existence of compliant AACS On-line API implementation shall be checked by referring to system property value defined in D.2 because the Licensed Player with Player Profile "0011b" developed before this revision might not implement AACS On-line function.

D.2. System property and API implementation

D.2.1. System Property

The System Propery for AACS On-line and Managed Copy API is defined as follows.

Other propery names with prefix "aacs.bluray." are reserved for future use.

- aacs.bluray.online.capability = YES | NO

    This means that System.getProperty("aacs.bluray.online.capability") in BD-J Application returns the value according to the requirement defined in D.2.2.

- aacs.bluray.mc.capability = YES | NO

    This means that System.getProperty("aacs.bluray.mc.capability") in BD-J Application returns the value according to the requirement defined in D.2.3.

D.2.2. Implemenation requirement for On-line

The following table describes the rules for AACS On-line API related player implementation.

Table D-1 System Property and API implementation for AACS On-line

	System Property (aacs.bluray.online.capability = YES | NO)	API implementation
Player Profile = 0000b, 0001b, or 1000b	NO (*1)	Recommended to implement AACS On-line API stubs to return at least dummy values
Player Profile = 0011b	YES	Required to implement AACS On-line API correctly.

(*1) Note: The Licensed Player developped before Revision 0.912 might not implement this property. They may throw SecurityException as a response for unknown system property request.

D.2.3. Implementation requirement for Managed Copy

The following table describes the rules for AACS Managed Copy API related player implementation.

Table D-2 System Property and API implementation for AACS Managed Copy

	System Property (aacs.bluray.mc.capability = YES | NO)	API implementation
Player which does not support Managed Copy	NO (*1)	IsMCMSupported() shall be implemented to return false. (Recommended to implement other AACS Managed Copy API stubs to return at least dummy values.)
Player which supports Managed Copy	YES	Required to implement AACS Managed Copy API correctly.

(*1) Note: The Licensed Player developped before Revision 0.912 might not implement this property. They may throw SecurityException as a response for unknown system property request.

D.2.4. Player Implementation options for VFS, On-line and Managed Copy

The following table describes the options for Player capability and the related indication method to check the Player implementation.

Table D-3 Player Implementation options for On-line and Managed Copy

Option	VFS (for BDMV and AACS files)	BD-J Network Connectivity	AACS Online APIs
1	NO	NO	NO
2	YES	NO	NO
3	YES	YES	YES
Indication Method	YES : Player Profile = 0001b or 0011b NO: Player Profile (others)	YES : Player Profile = 0011b NO: Player Profile (others)	SystemProperty (aacs.bluray.online.capability)

Note: Capability for Managed Copy is independent from other implementation options described in this table, and shall be checked by referring System Property (aacs.bluray.mc.capability).

D.3. Consideration for the use of com.aacsla.bluray package

The following requirements are applied for the contents and player implementation to consider the use of com.aacsla.bluray package.

Requirement for contents:

> BD-J applications shall not define classes in “com.aacsla.bluray” package. The fully qualified class name of any class defined by an application shall not start with “com.aacsla.bluray”.
>
> BD-J applications shall not use or reference API elements of “com.aacsla.bluray” that are not defined in AACS *Blu-ray Disc Pre-recorded Book*.

Requirement for Players:

> BD-ROM Terminals shall protect the overriding of APIs in “com.aacsla.bluray” package by using the SecurityManager.checkPackageDefinition mechanism.

D.4. Method to check the player’s capability by content

D.4.1. Method to check the player’s AACS On-line capability

To avoid the compatibility problem, it is strongly recommended to implement BD-J Application to check the player’s capability for AACS On-line APIs before it calls AACS On-line APIs. Following steps are example of capability check.

1. Read PSR31 value and check that Player Profile is 0011b.
2. Check that the return value to System.getProperty(“aacs.bluray.online.capability”) is “YES”.
   (Note 1): System.getProperty() may throw SecurityException. It is strongly recommended that the BD-J Application catch this exception and treat it as no AACS On-line capability.
   (Note 2): This check shall be done in addition to PSR31 check because the Licensed Player with Player Profile “0011b” developed before this revision might not implement AACS On-line function.
3. If the check in step1 and step2 succeeds, call AACS On-line APIs.

D.4.2. Method to check the player’s AACS Managed Copy capability

As defined in Chapter 5 of *AACS Pre-recorded Video Book* of this specification, IsMCMSupported() API is used to check the capability of AACS Managed Copy before it calls other AACS Managed Copy APIs. Following steps are example of capability check.

1. Check that the return value of IsMCMSupported() is true.
2. If the check in step1 succeeds, call other AACS Managed Copy APIs.