DRAFT INTERNATIONAL STANDARD
ISO/IEC DIS 23001-12

ISO/IEC JTC **1**/SC **29** Secretariat: **JISC**

Voting begins on: **2015-01-26**
Voting terminates on: **2015-04-26**

Information technology — MPEG systems technologies —

Part 12: **Sample Variants in the ISO Base Media File Format**

Titre manque

ICS: 35.040

THIS DOCUMENT IS A DRAFT CIRCULATED FOR COMMENT AND APPROVAL. IT IS THEREFORE SUBJECT TO CHANGE AND MAY NOT BE REFERRED TO AS AN INTERNATIONAL STANDARD UNTIL PUBLISHED AS SUCH.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

RECIPIENTS OF THIS DRAFT ARE INVITED TO SUBMIT, WITH THEIR COMMENTS, NOTIFICATION OF ANY RELEVANT PATENT RIGHTS OF WHICH THEY ARE AWARE AND TO PROVIDE SUPPORTING DOCUMENTATION.

Reference number
ISO/IEC DIS 23001-12:2014(E)

Licensed to ccid@sginfo.com
ISO Store order #: OP-17858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2014

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Contents

Page

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC/DIS 23001-12 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

ISO/IEC 23001 consists of the following parts, under the general title *Information technology — MPEG systems technologies*:

— *Part 1: Binary MPEG format for XML*

— *Part 2: Fragment Request Units*

— *Part 3: XML IPMP messages*

— *Part 4: Codec configuration representation*

— *Part 5: Bitstream Syntax Description Language (BSDL)*

— *Part 6: Dynamic adaptive streaming over HTTP (DASH)*

— *Part 7: Common encryption in ISO base media file format files*

— *Part 8: Coding-independent code points*

— *Part 9: Common encryption of MPEG-2 transport streams*

— *Part 10: Carriage of Timed Metadata Metrics of Media in ISO Base Media File Format*

— *Part 11: Energy-Efficient Media Consumption (Green Metadata)*

— *Part 12: Sample variants in the ISO base media file format*

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Introduction

As ISO/IEC 14496-12 ISO Base Media File Format (ISOBMFF) + ISO/IEC 23001-7 (CENC) becomes increasingly used for the distribution of high-value content, there is a growing need to support coding tools to aid with the identification of players from which content has been copied without authorization.

A common capability to aid with identification is known generically as “forensic marking”. A forensic marking system utilizes variants in the sample data that in aggregate are unique to the player or player model. Forensic marking enables efficient technical content protection response to the discovery of unauthorized copies to more promptly prevent future occurrences. The ability to better respond to such events encourages publishers to continue to release high value content using ISOBMFF.

There is today no standardized way to deploy forensic marking with ISOBMFF files. A standardized framework enables:

— industry adoption of forensic marking support with a non-proprietary framework for better interoperability

— support of ISOBMFF standards by services which require forensic marking

— reduced complexity of analysis tools.

This standard defines a general framework for Sample Variants in ISOBMFF. This framework would be used by a forensic marking system to control modifications to rendered samples, but is independent of the forensic marking algorithm; this standard does not standardize any specific forensic marking technique.

Background Design Requirements

The proposed technical approach to support a forensic marking framework in ISOBMFF:

— maintains file format compliance with ISOBMFF and CENC

— supports the provision of one or more Sample Variants for any sample in the ISOBMFF file

— enables cryptographic control over which Sample Variant is made available to a particular player

— is compatible with, and agnostic to, any particular sample and sub-sample based forensic marking system.

— is resilient to scrubbing attacks (the removal of Sample Variants from rendered output)

— minimizes the encoder and player overhead of adding Sample Variant data to the file

— enables independent verification of ISOBMFF files - verification should occur with no dependency on knowledge of the authoring tool or authoring approach

— maintains support for download, progressive download and super-distribution use cases

— maintains support for ISOBMFF file re-fragmentation and editing use cases

— maintains compatibility with CENC, and thus is independent of any specific DRM system

— provides security robustness:

  — Sample keys are of utmost value and thus must not be mixed with other keys since they may be subject to different compliance and robustness rules, for example used within trusted hardware (in contrast to keys encrypting metadata which may be in software)

  — Sample Variant data can be Double Encrypted with both a Media Key and another key

  — Sample Variant Metadata must be encrypted to obscure which Sample Variant data is actually used

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Figure 1 shows a scenario where a sample (Sample 2) has a number of Sample Variants. The figure shows 3 samples in a series left to right, the middle of which has variants. The top row is a conceptual depiction of what is encoded using ISOBMFF and the bottom row shows what is output after Sample Variant processing. Access to samples is under the control of KIDs as depicted in the top row of in Figure 1. For Sample Variants a hierarchy of KIDs is used to provide access to data, with the higher level KIDs providing access to Sample Variant Metadata and the lower level KIDs providing access to media data.

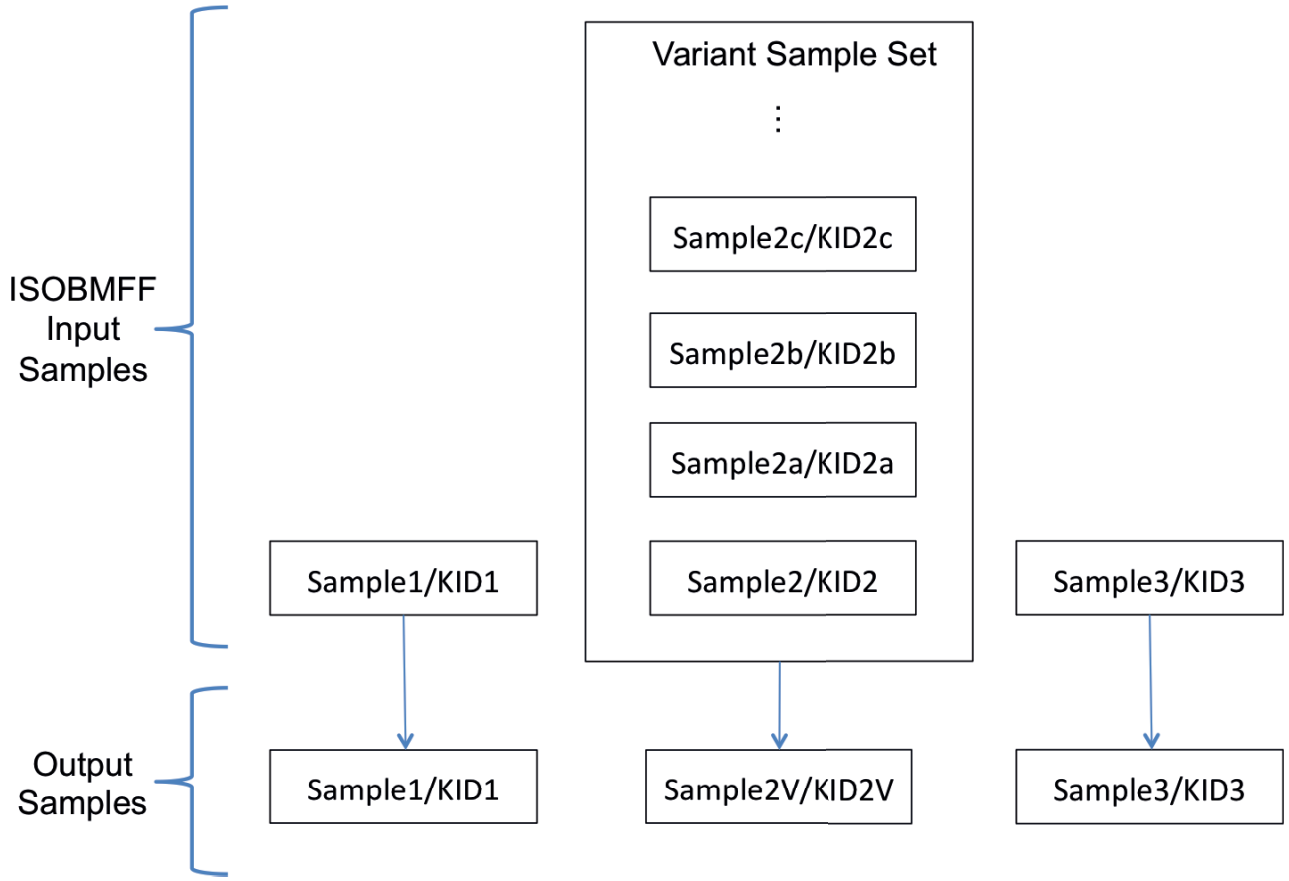

Figure 1 — Sample Variant structure

The control point for the use of the proposed framework is the content publisher:

— the content publisher will encode encrypted, compressed Sample Variant data into the ISOBMFF file and ensure that each set of Sample Variant data for a given sample time is encrypted with a different key and signalled with a different KID.

— the content publisher will work with the DRM to manage the release of KIDs/keys such that the playback path (the actual sample data used during playback) is controlled and the player can only decrypt and render the data that it has been authorized to render.

The decoder model for the processing of the file is shown in Figure 2. Critical to the Sample Variant decoding process is control over if and how the Sample Variants are processed.

Note: the decrypt and decode steps are standard operations as they would be for any CENC-enabled decoder.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

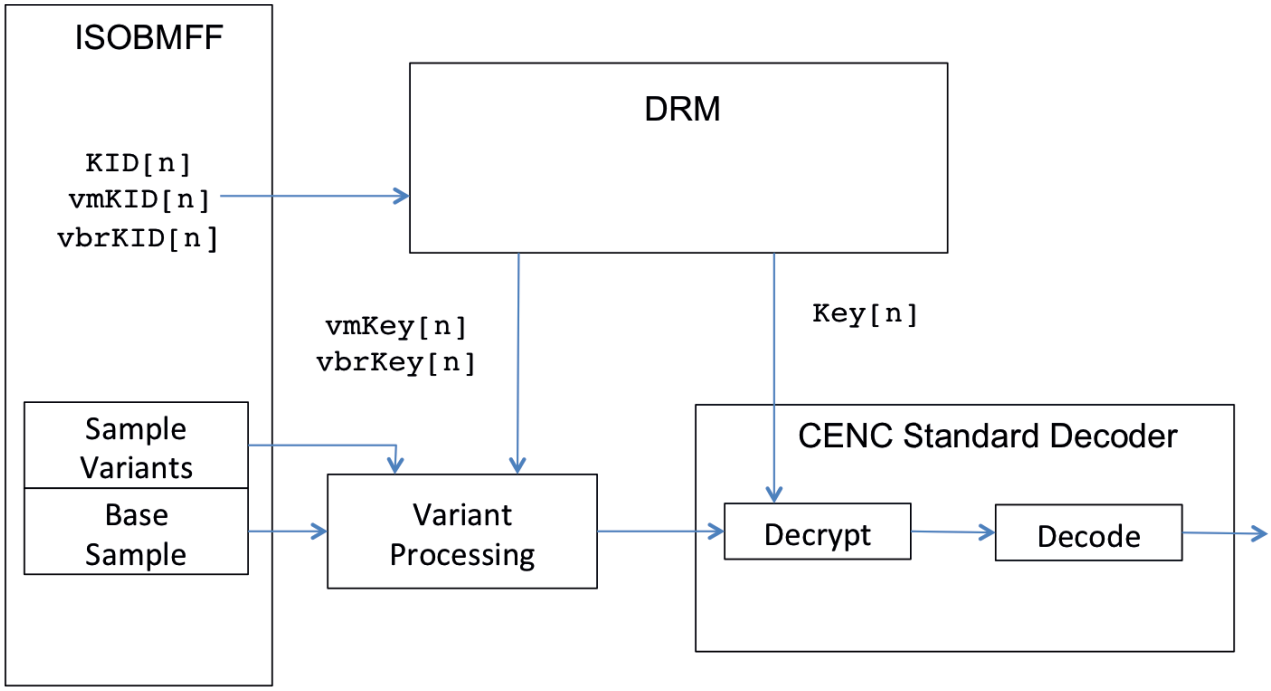

Figure 2 — Variant Decoder Model

By operating in the encrypted/compressed domain, secure baseband link operation (e.g. dedicated, secure video pathways) is preserved and is intended to be fully compatible with CENC.

The proposed approach is intended to support player model identification via DRM client identification, such as a unique player model number.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

Information technology — MPEG systems technologies —

Part 12:
Sample Variants in the ISO Base Media File Format

1 Scope

This part of ISO/IEC 23001 defines the carriage of Sample Variants in the ISO base media file format (ISO/IEC 14496-12).

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14496-12, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format*[1)]

ISO/IEC 23001-7:2014, *Information technology — MPEG systems technologies – Part 7: Common encryption in ISO base media file format files*

3 Terms and definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1
Double Encrypted
encryption of Sample Variant byte range data by first a Media Key (as part of the encryption of the complete Sample Variant) and then second a Variant Byte Range key (see Section 6.1).

3.1.2
Media Key
encryption key associated with one or more media samples.

3.1.3
Media KID
encryption KID associated with one or more media samples.

3.1.4
Sample Variant
media sample which is assembled using Sample Variant processing.

3.1.5
Variant Byte Range
location of a sequence of bytes that might constitute a portion of a Sample Variant.

1) ISO/IEC 14496-12 is technically identical to ISO/IEC 15444-12.

Licensed to c.thomas@mdetc.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

3.1.6
Variant Constructor
Sample Variant metadata that defines how to assemble an individual Sample Variant.

3.1.7
Variant Media Data
media data used to construct a Sample Variant, some of which may come from the original sample media data.

3.1.8
Variant Processor
logical module that performs the processing steps that implement the process of assembling Sample Variants

3.2 Abbreviated terms

For the purposes of this International Standard, the following abbreviated terms apply.

CENC Common ENCryption (as specified by ISO/IEC 23001-7)

DRM Digital Rights Management

ISOBMFF ISO Base Media File Format (as specified by ISO/IEC 14496-12)

IV Initialization Vector

KID Key Identifier

4 Overview (informative)

This Standard defines a framework for the carriage of Sample Variants in the ISOBMFF. Sample Variants are typically used to provide forensic information in the rendered sample data that identifies the DRM client. This variant framework is intended to be fully compatible with ISO/IEC 14496-12 and ISO/IEC 23001-7, and agnostic to the particular forensic marking system used.

The Sample Variant framework uses three core constructs to define and carry Sample Variant data in ISOBMFF: Variant Constructors, Variant Byte Ranges and Variant Samples.

Note The Variant Process Model described in Section 9 can also assist in introducing the concepts.

5 Variant Constructors

5.1 Introduction

A Variant Constructor defines which bytes are used to assemble a Sample Variant. There may be one or more Variant Constructors defined for a given sample presentation time.

The Variant Processor may use a Variant Constructor if the Variant Processor has access to the Variant Constructor. In addition to the presence of the Variant Constructor, “access” includes cryptographic access. A Variant Constructor shall define which data is used to assemble a Sample Variant and the associated Media KID and initialization vector for decrypting the Sample Variant.

5.2 Access to Variant Constructors

If the decoder is given access to the Media Key for the sample defined by the ISOBMFF media track, Sample Variant processing will not occur for this sample. If the decoder does not have access to the Media Key for the sample defined by the ISOBMFF media track, the Variant Processor shall be given access to one Variant Constructor associated with the sample.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

The KID/Key associated with the Variant Constructor controls access to a particular Variant Constructor and is therefore a function of the set of KID/Key value pairs made available to the Variant Processor by the DRM. Only one Variant Constructor per sample should be made available to the Variant Processor. If the Variant Processor is given access to a Variant Constructor, the decoder shall also be given access to the Media Key associated with the Media KID defined in the Variant Constructor.

If the Variant Processor has access to more than one KID/Key associated with Variant Constructor for a given sample, the Variant Processor utilizes the first Variant Constructor that it has access to in data encoding order. The Variant Processor uses at most one Variant Constructor to assemble a Sample Variant.

5.3 Encryption of Variant Constructors

Each Variant Constructor shall be encrypted with a "Variant Constructor key". The Variant Byte Range key is used exclusively for Double Encrypted content, not in isolation i.e. the key is only used to further encrypt data that is already encrypted with a Media Key, not on unencrypted content.

As a Variant Processor is provided only with the Variant Constructor keys for the Variant Constructor that is to be used by that particular Variant Processor, any Variant Constructors not used by that Variant Processor are not exposed by a security compromise of that Variant Processor.

6 Variant Byte Ranges

6.1 Introduction

Each Variant Constructor shall define a sequence of one or more Variant Byte Ranges. Each Variant Byte Range defines the location of a sequence of bytes that might constitute bytes in a Sample Variant.

The sequence of Variant Byte Ranges defined in a Variant Constructor shall be grouped into one or more Variant Byte Range groups. Each Variant Byte Range group shall define one or more Variant Byte Ranges. An individual Variant Byte Range within a Variant Byte Range group:

— May reference bytes of data that constitute bytes in a Sample Variant that is made available to certain Variant Processors ("real Variant Byte Range").

— May reference bytes of data that are not made available to any Variant Processor ("fake Variant Byte Range").

A "fake Variant Byte Range" can be used to hide the amount of actual "real Variant Byte Ranges" defined within a Variant Constructor. The Variant Processor uses all Variant Byte Ranges that it has access to. In addition to the presence of the Variant Byte Range, "access" includes cryptographic access.

Data for different Sample Variants can be stored non-contiguously as referenced by different Variant Constructors. Data for a particular Sample Variant can also be stored non-contiguously using a sequence of two or more Variant Byte Ranges.

6.2 Access to Variant Byte Ranges

If a Variant Byte Range within a Variant Byte Range group signals that the data referenced by the Variant Byte Range is unencrypted (and the Variant Processor has access to the Variant Constructor), then the Variant Processor has access to the Variant Byte Range and the associated unencrypted bytes.

If the Variant Byte Range defined within a Variant Byte Range group signals that the data referenced by the Variant Byte Range is encrypted, then access to the Variant Byte Range and the associated bytes is controlled by the KID/Key associated with each Variant Byte Range - either the Media Key defined by the Variant Constructor if no Variant Byte Range key is defined for the particular Variant Byte Range group or by the Variant Byte Range key if one is defined. Access to the Variant Byte Range and the associated data referenced by a Variant Byte Range is therefore a function of the set of KID/Key value pairs made available to the Variant Processor by the DRM. Only one Variant Byte Range within a Variant Byte Range group should be made available to the Variant Processor.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

If the Variant Processor has access to more than one KID/Key associated with Variant Byte Ranges within the same Variant Byte Range group for a given sample, the Variant Processor uses the first Variant Byte Range that it has access to in data encoding order. The Variant Processor uses at most one Variant Byte Range within a Variant Byte Range group to assemble a Sample Variant.

Variant Byte Ranges can be used to efficiently encode only the (typically small) differences in Sample Variants for a given presentation time without repeating non-difference data or exposing Sample Variant differences to the Variant Processor. This is achieved through Double Encryption, where the difference data is first encrypted by the Media Key and then encrypted by the Variant Byte Range key. As the Variant Processor requires access to the Variant Byte Range key to decrypt such difference data - therefore access to the difference data can be controlled while also enabling reuse of common data and preserving Sample Variant compatibility with common encryption, which requires that only one Media Key be applied to a given sample. If Variant Byte Ranges did not provide this capability then it would be necessary to repeat all data for each Sample Variant, including difference and non-difference data, so as to protect difference data with a different key – this is inefficient.

6.3 Encryption of Variant Byte range information

Variant Byte Range definitions are not individually encrypted (they are encrypted as part of the Variant Constructor).

7 Sample Variants

7.1 Introduction

The data used for rendering a sample is defined by either a Variant Constructor (if the Variant Processor has access to the Variant Constructor for the sample per Section 8.3.3), or by the media data defined by ISOBMFF. When Variant Constructors are used, the actual data used for reconstructing the sample is obtained by assembling, in the order of appearance in the Variant Constructor, the byte data referenced by the Variant Byte Ranges made available to the Variant Processor per Section 6 and this construction shall result in a valid encrypted sample for the signalled underlying encryption system – this sample is a Sample Variant.

7.2 Access to Sample Variants

Once the Sample Variant is assembled from the Variant Byte Ranges, access to the sample data is controlled by the Media Key defined in the Variant Constructor and is therefore a function of the set of KID/Key value pairs made available to the Variant Processor by the DRM.

7.3 Encryption of Sample Variants

Sample Variants shall always be encrypted according to the scheme signalling of the associated media track. Variant Byte Ranges of a Sample Variant may be unencrypted, or may be encrypted with a Media Key. The Media Key is associated with one or more samples.

When bytes in a Sample Variant are encrypted with a Media Key, one or more byte ranges of the encrypted Variant Media Data may be further encrypted (Double Encrypted) according to the common encryption signalling with a “Variant Byte Range key” per Section 6.

As a Variant Processor is provided only with the Variant Byte Range keys for Double Encrypted Variant Media Data that are to be used by that particular Variant Processor, Double Encrypted Variant Media Data not used by that Variant Processor are not exposed by a compromise of that Variant Processor.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

8 ISO Storage

8.1 Introduction

Variant data is stored in an ISOBMFF metadata tracks (variant track). An ISOBMFF media track (media track) or variant track may be associated with one or more variant tracks as defined in section 8.2.2.

— When an association is established between a media track and a variant track, Sample Variant processing will occur whenever a decoder does not have access to the KID/key defined for a sample in the media track as defined in section 5.2.

— When an association is established from a variant track (original variant track) to another variant track (other variant track), variant data contained in the other variant track can be utilized by the original variant track.

— Samples within associated tracks are associated if they are time-parallel as defined in section 8.3.5.

8.2 Variant tracks

8.2.1 Definition

Variant data shall be stored in an ISOBMFF metadata track that complies with the following constraints:

a) The track shall use the `'meta'` `handler_type` in the Handler Reference Box `('hdlr')` per ISO/IEC 14496-12, section 12.

b) The track shall use the `VariantMetaDataSampleEntry()` sample entry as defined in section 8.2.3.

c) Variant data is stored in the track as samples in accordance with section 8.3.

8.2.2 Association

ISOBMFF tracks may be associated with variant tracks via one of the following means:

— An externally defined context.

— In the source track (e.g. in the original media track) provide a Track Reference Type Box in the Track Reference Box `('tref')` of the Track Box `('trak')` which has a `reference_type` of `'cvar'` and one or more `track_IDs` that each correspond to a `track_ID` of a variant track that is to be referenced in the same file.

The following additional requirements apply to track_IDs in a Track Reference Type Box of `reference_type 'cvar'`:

a) `track_ID` may have a value that does not correspond to a `track_ID` of a track in the same file.

b) If the `track_ID` does correspond to a `track_ID` of a track in the same file, the corresponding track shall be a variant track which complies with section 8.2.1.

Variant track references defined for a media track shall be defined in Variant Constructor search order. The Variant Processor will process variant tracks according to this order when searching for an accessible Variant Constructor.

8.2.3 Variant Metadata Sample Entry

8.2.3.1 Syntax

```
class VariantMetaDataSampleEntry() extends MetaDataSampleEntry ('cvar') {
      unsigned int(32)    variant_constructor_scheme_type;
      unsigned int(32)    variant_constructor_scheme_version;
```

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

```
        unsigned int(32)    media_track_scheme_type;
        unsigned int(32)    media_track_scheme_version;
}
```

8.2.3.2 Semantics

`variant_constructor_scheme_type` – shall be set to the four character code defining the protection scheme applied to Variant Constructors in the track, per section 8.3.4.

`variant_constructor_scheme_version` – shall be set to the version of the protection scheme applied to the Variant Constructors in the track, per section 8.3.4.

`media_track_scheme_type` – shall be set to the four character code defining the protection scheme applied to associated media track, as defined by ISO/IEC 14496-12, section 8.12.5.3.

`media_track_scheme_version` – shall be set to the version of the protection scheme applied to associated media track, as defined by ISO/IEC 14496-12, section 8.12.5.3.

8.3 Sample data

8.3.1 Definition

A sample in a variant track is either empty (zero size) or a `VariantData()` structure.

8.3.1.1 Syntax

```
aligned(8) class VariantData
{
        VariantConstructorList()    variant_list;
        VariantConstructor()[]      variant_constructors;
        unsigned int(8)[]           variant_pool;
}
```

8.3.1.2 Semantics

`variant_list` – the Variant Constructor list as defined in Section 8.3.2.

`variant_constructors` – the array of Variant Constructors referenced by the Variant Constructor list.

`variant_pool` – a pool of variant bytes that may be referenced by a Variant Constructor.

8.3.2 Variant Constructor List

The `VariantConstructorList()` defines sample specific information on the location of potential Variant Constructors for Sample Variants.

Each sample definition in a variant track may have one or more Variant Constructor location entries in the `VariantConstructorList()`. As required in Section 5.2, at most one individual Variant Constructor location entry is used during playback of a given sample and the Variant Processor uses the first Variant Constructor that it has access to in order of definition in the `VariantConstructorList()` structure.

Individual entries in the `VariantConstructorList()`:

— may reference a `VariantConstructor()` for the sample definition that is used during particular playback scenarios ("real Variant Constructors"); or

— may reference bytes that are not made available to any Variant Processor ("fake Variant Constructor").

Note Without access to the decryption Key referenced by `vcKID`, fake Variant Constructors and real Variant Constructors are indistinguishable. Fake Variant Constructors can be used to hide the number of real Variant Constructors defined in the `variant_constructors` array.

8.3.2.1 Syntax

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

```
aligned(8) class VariantConstructorList
{
	unsigned int(32)                                    size;
	unsigned int(8)                                     variant_constructors_count;
	for( i=1 ; i<= sample_variant_constructors_count; i++) {
		unsigned int(8)[16]                         vcKID;
		unsigned int(8*IV_Size)                     vcIV;
		if( version == 0 ) {
			unsigned int(32)            variant_constructor_offset;
			unsigned int(32)            variant_constructor_size;
		} else {
			unsigned int(64)            variant_constructor_offset;
			unsigned int(64)            variant_constructor_size;
		}
	}                       }
	unsigned int(8)[]                                   padding
}
```

8.3.2.2 Semantics

`size` – shall be set to the size, in bytes, of the `VariantConstructorList()`.

`variant_constructors_count` – shall be set to the number of Variant Constructor entries in the `constructors` array in the `VariantData()`.

`vcKID` – the "Variant Constructor KID". This KID shall reference the Variant Constructor metadata key used for decrypting the encrypted Variant Constructor.

`vcIV` – the "Variant Constructor Initialization Vector". This field shall reference the initialization vector used for decrypting the encrypted Variant Constructor.

`variant_constructor_offset` – the byte offset of the corresponding `VariantConstructor()`. This offset is relative to the start of the `VariantData()`.

`variant_constructor_size` – the length, in bytes, of the `VariantConstructor()`. The combination of `variant_constructor_offset` and `variant_constructor_size` indicates the location and size of the `VariantConstructor()`. The byte range defined by `variant_constructor_offset` and `variant_constructor_size` shall only reference bytes within the `variant_constructors` array in the `VariantData()` and no other bytes.

`padding` – the byte array may contain any data and be used to increase the size of the `VariantConstructorList()`.

Note this padding can be used to obfuscate the actual size of the `VariantConstructorList()` if it is encrypted.

8.3.3 Variant Constructor

8.3.3.1 Syntax

```
aligned(8) class VariantConstructor
{
	unsigned int(8)[16]                     KID;
	unsigned int(8*IV_Size)                 IV;
	unsigned int(32)                        variant_byte_ranges_count;
	for( i=1; i<= variant_byte_ranges_count; i++ )
	{
		unsigned int(8)                 variant_byte_range_flags;
		if( variant_byte_range_flags & 0x02 )
		{
			unsigned int(8)[16]             vbrKID;
			unsigned int(8*IV_Size)         vbrIV;
		}
		if( variant_byte_range_flags & 0x08 ) {
			unsigned int(8)         variant_track_reference index;
		}
```

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

```
                signed int(8)                       relative_sample_number;
                if( version == 0 ) {
                        unsigned int(32)            variant_byte_range_offset;
                } else {
                        unsigned int(64)            variant_byte_range_offset;
                }
                if( variant_byte_range_flags & 0x06 != 0x02 ) {
                        unsigned int(32)            variant_byte_range_size;
                }
        }
        unsigned int(8) []                          padding;
}
```

8.3.3.2 Semantics

`KID` – the Media KID. This KID shall reference the Media Key that is used for decrypting the encrypted Sample Variant data after re-assembly of the applicable Variant Byte Ranges. Decryption occurs in accordance with the protection scheme signalled in the associated media track.

`IV` – the Initialization Vector that shall be used for decrypting the encrypted Variant Media Data after re-assembly of the applicable Variant Byte Ranges in accordance with the protection scheme signalled in the associated media track.

`variant_byte_ranges_count` – shall be set to the number of Variant Byte Ranges defined for this Variant Constructor. See Section 6 for more information.

`variant_byte_range_flags` – shall be set as follows:

0x01 encrypted When set, the Sample Variant data referenced by the Variant Byte Range shall be encrypted.

0x02 double-enc When set, the Sample Variant data referenced by the Variant Byte Range shall be Double Encrypted with a Variant Byte Range key. The meaning is undefined when `variant_byte_range_flags` signals that the Sample Variant data referenced by the Variant Byte Range is unencrypted.

0x04 group-start When set, the Variant Byte Range shall be the start of a Variant Byte Range group and thus provides a marker for Variant Byte Range groups within the `VariantConstructor()`. As per Section 6, the Variant Byte Ranges defined in the `VariantConstructor()` are grouped into one or more Variant Byte Range groups, and one Variant Byte Range from each Variant Byte Range group is used by the Variant Processor. This therefore requires that even if there is only one Variant Byte Range defined in the `VariantConstructor()`, or there is only one Variant Byte Range within a Variant Byte Range group (i.e. there are no alternative Variant Byte Ranges for a particular byte range of the Variant Media Data), that the start of Variant Byte Range group be signalled with this singular Variant Byte Range. As per Section 6.2, if more than one Variant Byte Range appears in a single Variant Byte Range group, each is Double Encrypted in order to limit the Variant Processor access to one byte range within the byte range group.

Note This flag can be used by a Variant Processor to determine that a data error has occurred - if no Variant Byte Range is in a Variant Byte Range group is recognized, an error has occurred.

0x08 data-source When set to 0, the data source for this range shall be the original media track. When set to 1, the `variant_track_reference_index` indicates which variant track shall be the data source.

`vbrKID` – the "Variant Byte Range KID". This KID shall reference the Variant Byte Range key used for decrypting the Double Encrypted Variant Media Data.

`vbrIV` – the "Variant Byte Range Initialization Vector". This field shall reference the initialization vector used for decrypting the Double Encrypted Variant Media Data.

`variant_track_reference_index` - shall either be the 1-based index (according to order of reference definition - see section 8.2.2) of the track references from this variant sample track to another variant track containing the variant data to be used; or if this value is 0, the data is drawn from this variant track.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

`relative_sample_number` – having found the track data source (see the `data-source` flag and `variant_track_reference_index` field above), this field defines which sample data source shall be used for the Variant Byte Range as follows: when set to 0, the sample data-source is the time-parallel associated sample per section 8.3.5; when set to a negative value, the Nth prior sample is used; when set to positive value, the Nth succeeding sample is used.

`variant_byte_range_offset` – is the byte offset from the start of the referenced sample (original sample in the media track, the `VariantData()` that contains this Variant Constructor, or the `VariantData()` in a referenced variant track, depending on the data-source flag and `variant_track_reference_index)` to the beginning of the data for this Variant Byte Range.

`variant_byte_range_size` – the size of the Variant Byte Range in bytes. The combination of `variant_byte_range_offset` and `variant_byte_range_size` indicates a byte range for the Variant Byte Range in the referenced sample. The Variant Byte Range defined by `variant_byte_range_offset` and `variant_byte_range_size` shall only reference bytes within the referenced sample and no other bytes. If there is more than one Variant Byte Range in a Variant Byte Range group, this field only exists for the first Variant Byte Range as the size of Variant Byte Ranges in a Variant Byte Range group is the same.

`padding` – the byte array may contain any data and be used to increase the size of the Variant Constructor.

Note this padding can be used to obfuscate the actual size of the Variant Constructor as it is encrypted.

8.3.4 Encryption

As defined in section 5.3 Variant Constructors are always encrypted. One of the following ISO/IEC 23001-7 encryption modes shall be used to encrypt Variant Constructors:

a) AES-CTR Full Sample Encryption: signalled with a four character code value of `'cvar'` and a `scheme_version` value of 0x00010000 (Major version 1, Minor version 0) in the `VariantMetaDataSampleEntry()` per section 8.2.3.

b) AES-CBC-128 Full Sample Encryption: signalled using a four character code value of `scheme_type` field value of `'cva1'` and a `scheme_version` field value of 0x00010000 (Major version 1, Minor version 0) in the `VariantMetaDataSampleEntry()` per section 8.2.3.

As defined in section 7.3 Sample Variants assembled from Variant Byte Ranges defined in a Variant Constructor are encrypted according to the scheme signalling of the associated media track and per section 8.2.3 this scheme is also signalled in the `VariantMetaDataSampleEntry()`.

8.3.5 Association

Samples are associated as follows:

a) A sample in a media track shall be associated with a sample in a variant track referenced by the media track if the samples are time-parallel.

b) A sample in a variant track shall be associated with a sample in another variant track referenced by the variant track if the samples are time-parallel.

c) Samples are considered to be time-parallel as follows: If To is the decode time of the sample in the original track, then the time-parallel sample in a referenced track is the sample in that referenced track that has a decode time Tv and duration D, such that Tv<=To<(Tv+D).

Note 1 sample association occurs at media decode time before any consideration of edit lists or composition offset.

Note 2 a sample in a variant track may have zero data size if no variant data is to be provided at that particular sample time.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

8.3.5.1 Example

An example of media track and variant track referencing is shown in Figure 3.

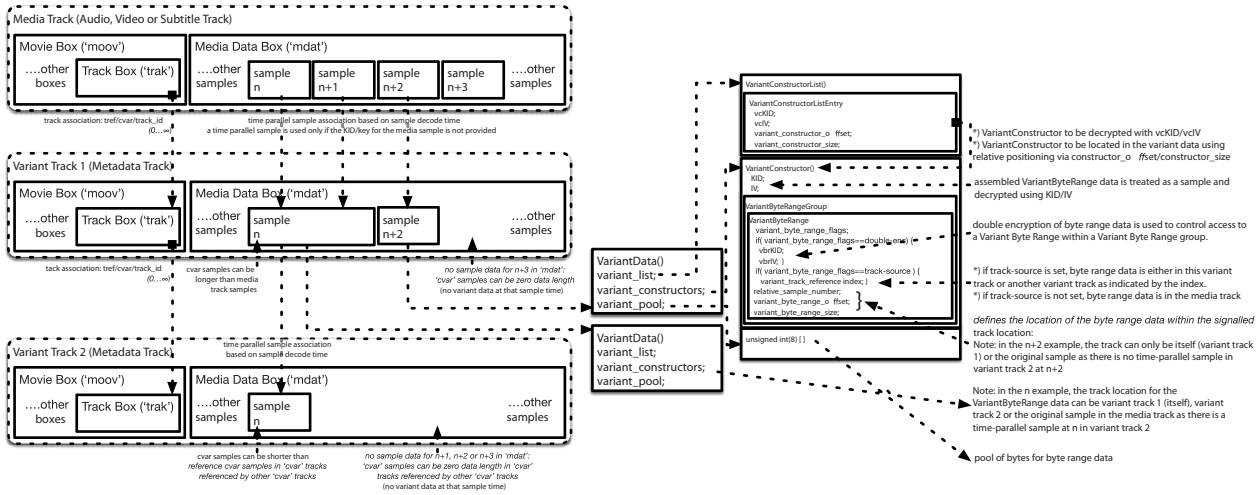

Figure 3 — Media track and variant track referencing

9 Variant Processor Model & Example (Informative)

9.1 Variant Processor Model

The rendering of a sample is expected to satisfy the observable behaviour defined by the following model:

a) The data source for each sample is evaluated as follows:

   1) If the decoder has access to the sample in the media track, the decoder proceeds to render the sample as per Section 5.2.

   2) If the decoder does not have access to the sample in the media track, the Variant Processor will determine which Variant Constructor is the data source for the sample as defined Section 5.2. The Variant Processor searches for an accessible Variant Constructor as follows:

      i) The Variant Processor will search each variant track referenced by the media track in order of reference definition e.g. the order of track references in the Track Reference Box (`'tref'`). See section 8.2.2 for more information.

      ii) In each variant track searched, the Variant Processor will determine if variant data exists for the time-parallel sample in the variant track. If variant data exists, the Variant Processor will search the `VariantConstructorList()` in the time-parallel sample in the variant track.

      iii) The Variant Processor will continue to search until it finds a Variant Constructor KID (`'vcKID'`) in a `VariantConstructorList()` that matches a KID/Key the Variant Processor has access to.

b) Using the Variant Constructor key and initialization vector defined in the `VariantConstructorList()` for the Variant Constructor selected by the Variant Processor, the Variant Processor decrypts the `VariantConstructor()` structure defined in Section 8.3.3.

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

c) The Variant Processor sequentially processes each Variant Byte Range in the sequence of Variant Byte Ranges defined in the decrypted Variant Constructor and assembles the Variant Media Data for the sample as follows:

   1) If the Variant Byte Range is signalled to be unencrypted per the definition of `variant_byte_range_flags` in Section 8.3.3.2, the byte range is put directly in the sample assembly and identified as unencrypted.

   2) If the Variant Byte Range is signalled to be encrypted per the definition of `variant_byte_range_flags` in Section 8.3.3.2:

      i) If the Variant Byte Range data is signalled as single encrypted with the Media Key per the definition of `variant_byte_range_flags` in Section 8.3.3.2, it is put directly in the sample assembly and identified as encrypted.

      ii) If the Variant Byte Range media is signalled as Double Encrypted per the definition of `variant_byte_range_flags` in Section 8.3.3.2:

         I) if the Variant Byte Range KID `('vbrKID')` defined by the Variant Byte Range is available to the Variant Processor, the Variant Byte Range data referenced by the Variant Byte Range is decrypted using the Variant Byte Range key referenced by the Variant Byte Range KID `('vbrKID')` and the Variant Byte Range initialization vector referenced by the Variant Byte Range IV `('vbrIV')`. This operation results in single encrypted data which is put in the sample assembly and identified as encrypted.

         II) if the Variant Byte Range KID `('vbrKID')` defined by the Variant Byte Range is not available to the Variant Processor, the Variant Byte Range is skipped.

d) The assembled Variant Media Data is decrypted using the Media Key defined by the Variant Metadata (as referenced by the KID field in the Variant Metadata defined in Section 8.3.2.2), in accordance with CENC.

9.2 Example

Consider a Variant Constructor consisting of three byte range groups:

— The first byte range group has one Variant Byte Range S1, which is unencrypted.

— The second byte range group has one Variant Byte Range S2, which is encrypted.

— The third byte range group has two Variant Byte Ranges, S3 and S4, each of which are encrypted.

At encryption time:

— The Sample Variant data associated with Variant Byte Range S1 is not encrypted, resulting in unencrypted Sample Variant data D1.

— The Sample Variant data associated with Variant Byte Ranges {S2, S3,S4} are each encrypted with Media Key K1 (KID KID1), resulting in encrypted Sample Variant data {D2*, D3*, D4*}.

— The encrypted Sample Variant data D3 is further encrypted with the Variant Byte Range key K3 (KID KID3) and encrypted Sample Variant data D4 is encrypted with Variant Byte Range key K4 (KID KID4), resulting in doubly encrypted media data D3** and D4**.

The resulting Variant Constructor will have four byte ranges and is structured as [| S1 | S2 | S3 S4], where the symbol "|" indicates the start of a byte range group. The underlying media data is stored as {D1, D2*, D3**, D4**}.

If the Variant Processor has access to KID1 and KID3 only, per the Variant Processor will do the following:

a) Process S1, establish it as unencrypted and consequently add D1 to the sample assembly and identify it as unencrypted (per step 3.a in Section 9).

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

b) Process S2, match KID1 and consequently add D2* to the sample assembly and identify it as encrypted (per step 3.b.i above).

c) Process S3, match KID3 and consequently decrypt D3** using K3, then add the resulting D3* to the sample assembly and identify it as encrypted (per 3.b.ii.a in Section 9).

d) Process S4, not recognize KID4 and consequently skip D4** (per step 3.b.ii.b in Section 9).

e) Decrypt the sample assembly [D1 D2* D3*] by skipping D1 and using the Media Key K1 to decrypt D2* and D3*, resulting in unencrypted Sample Variant [M1 M2 M3] (per step 4 in Section 9).

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

© ISO/IEC 2014 – All rights reserved

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.

ICS 35.040

Price based on 12 pages

© ISO/IEC 2014 – All rights reserved

Licensed to cciordas@irdeto.com
ISO Store order #: OP-67858/Downloaded: 2015-05-13
Single user licence only, copying and networking prohibited.