ISO/IEC JTC 1/SC 29 N

Date: 2011-08-20

ISO/IEC 23001-7:201X/PDAM 1

ISO/IEC JTC 1/SC 29/WG 11

Secretariat:

Information technology — MPEG systems technologies — Part 7: Common encryption format for ISO base media file format, AMENDMENT 1: AES-CBC-128 and key rotation

Élément introductif — Élément central — Partie 7: Titre de la partie

Warning

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

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.

Document type: International Standard
Document subtype: Amendment
Document stage: (30) Committee
Document language: E

STD Version 2.1c2

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.

Amendment 1 to ISO/IEC 23001-7:201X was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Information technology — MPEG systems technologies — Part 7: Common encryption format for ISO base media file format, AMENDMENT 1: AES-CBC-128 and key rotation

<<Editor Note: Organization defects may be present that don't fit into the re-organized base text of the 2nd Edition DIS. Some editorial re-organization might be needed to correspond to the latest FDIS >>

In Clause 10.1, replace the first sentence with the following:

Media data using ‘cbc1’ or ‘cbc2’ Protection Scheme uses the Advanced Encryption Standard specified by AES [FIPS197] using 128-bit keys in Cipher-block chaining mode (AES-CBC-128), as specified in Block Cipher Modes [NIST 800-38A], with IVs stored as described in 6 and 9.2.

In Clause 10.1, replace the second sentence with the following:

Encrypted NAL Structured Video Tracks shall follow the scheme outlined in 10.2.4 or 10.3, which defines a NAL unit based encryption scheme to allow access to NAL units and unencrypted NAL unit headers in an encrypted stream of NAL Structured Video.

In Clause 10.1, replace the Note with the following:

NOTE Support for ‘cbc1’ or ‘cbc2’ scheme is not mandatory in the common encryption mechanism, however implementations that process the ‘cbc1’ or ‘cbc2’ scheme are also required to process the ‘cenc’ scheme so that files using the ‘cenc’ scheme may be processed on all implementations of this standard.

Add the following new sub-Clause at the end of Clause 10.2:

10.3 AES-CBC-128 bit Pattern Based Partial Encryption Scheme

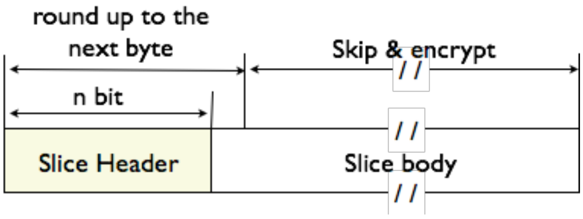
10.3.1 Introduction

The scheme_type field of the Scheme Type Box (‘schm’) shall be set to ‘cbc2’ to signal the usage of AES-CBC-128 bit pattern based partial encryption mode. Media data protected using this mode utilizes a partial skip/encrypt pattern that spreads the encryption over the entire slice body, while the NAL unit headers and video slice headers are in the clear.

10.3.2 Partial Encryption Scheme for NAL Structured Video Tracks

In this mode of encryption, VCL NAL unit types are protected while non-VCL NAL unit types may be protected.

For VCL NAL units, the encryption process shall protect the slice body starting at an offset rounded to the next byte after the slice header. The slice headers are in the clear to enable decoders to read all relevant information without having to start the decryption process.



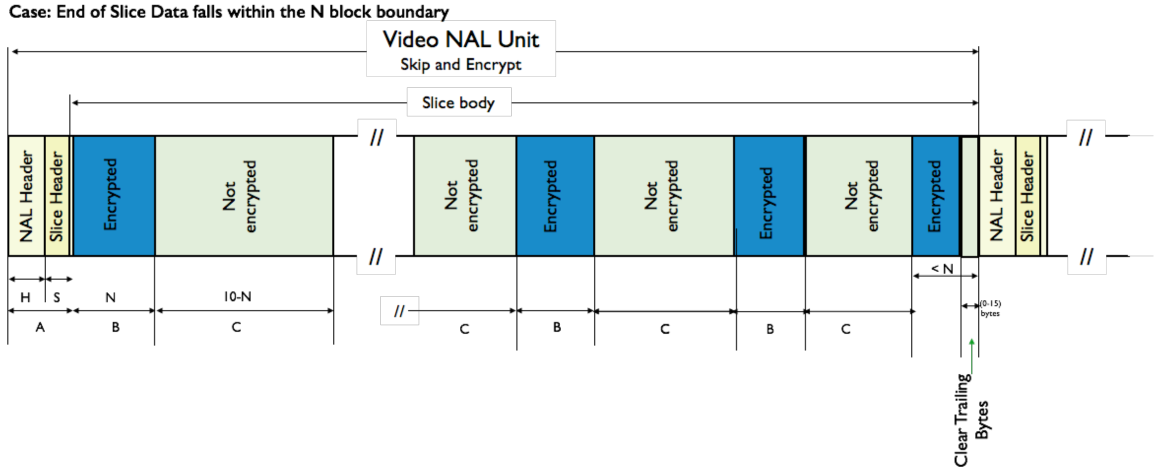
Note: For AVC VCL NAL units, the encryption process starts at an offset rounded to the next byte after the slice header. For HEVC, the encryption pattern starts after the byte_alignment() that terminates the slice_segment_header(), i.e. on the first byte of slice data.

The range of the slice body following the slice header is protected by applying AES-CBC 128-bit encryption in a repeating pattern of encrypted and unencrypted 16-byte blocks, with no cipher padding. CBC is restarted at every NAL unit.

This scheme recommends a pattern of length 160 bytes (ten 16-byte blocks). To achieve 10% encryption, the first 16-byte block of the pattern is encrypted and the following nine 16-byte blocks are left unencrypted. The pattern is repeated every 160 bytes of the protected range, until the end of the range. If the protected range of the slice body ends in a partial 16-byte block (*i.e. the byte length of the range is not evenly divisible by 16*), the remaining partial 16-byte block is left unencrypted.

The following scenarios illustrates a use-case where the end of slice data falls within an encrypted block boundary, where all blocks that are a multiple of 16-bytes should be encrypted and any remaining bytes be left in the clear.

Case: End of Slice Data falls within the N block boundary



H: 4 bytes
S: Slice Header and additional first slice body bits to round up to the next byte
A: First bytes aligned on a byte to skip
B: Encrypted unit(s)
C: Skipped unit(s)
N: Encrypted block count (block is 16bytes)
10-N: Skipped block(s) (block is 16 bytes)
If the end of the slice data falls on within the N block boundary, encrypt as much as possible in 16 byte increments and leave rest in the clear.

Since no per-sample initialization information is required for this scheme, there is no Sample Auxiliary Information defined for this mode (Both KeyID and IV could be carried out of band).

10.3.3 Box Definitions

The sample entry is transformed and a Protection Scheme Information Box (‘sinf’) is added to the standard sample entry in the Sample Description Box to denote that the stream is encrypted. The Protection Scheme Information Box shall contain a Scheme Type Box (‘schm’) so that the scheme is identifiable. The Scheme Type Box has the following constraints:

- The scheme_type field is set to a value of ‘cbc2’.

- The scheme_version field is set to 0x00010000 (Major version 1, Minor version 0).

The Protection Scheme Information Box shall also contain a Scheme Information Box (‘schi’). The partial encryption skip/encrypt parameters are specified within the schi box, whose format is given below.

10.3.3.1 Scheme Information Box

10.3.3.1.1 Definition

Box Types: ‘schi’
Container: Protection Scheme Information Box (‘sinf’)
Mandatory: Yes
Quantity: One

The Scheme Information Box is a container Box that is interpreted by the scheme ‘cbc2’ used to signal the partial encryption mode of operation. Information about the encryption pattern is stored here.

10.3.3.1.2 Syntax

```
aligned(8) class SchemeInformationBox extends Box(‘schi’, version=0, flags=0)
{
   CryptSpanBox               cspn_atom;
   CryptKeyIDBox              ckid_atom;    // optional
   CryptIVBox                 criv_atom;    // optional
   CryptNonVideoNALTypesBox   cnvd_atom;    // optional
}
```

10.3.3.2 Crypt Span Box

10.3.3.2.1 Definition

Box Types: ‘cspn’
Container: Scheme Information Box (‘schi’)
Mandatory: Yes
Quantity: One

The `CryptSpanBox` documents the partial encryption pattern. The scheme recommends a pattern of length 160 bytes (i.e., 10 blocks of 16-bytes each).

10.3.3.2.2 Syntax

```
aligned(8) class CryptSpanBox extends FullBox(‘cspn’, version=0, flags=0)
{
   unsigned int(32) crypt_byte_block;   //Encrypted Byte block count (N)
   unsigned int(32) skip_byte_block;    //Block count to skip (M);
                                        //Recommend that N+M=10 (160 bytes)

}
```

10.3.3.2.3 Semantics

`crypt_byte_block` specifies the count of the encrypted blocks (N), where each block is of size 16-bytes.
`skip_byte_block` specifies the count of the unencrypted blocks (M).

The scheme recommends an encrypt:skip (N:M) pattern of 1:9 (i.e., 10% partial encryption), while the syntax allows tunable encryption patterns in steps of 10% from 0% to 100%. The number of encrypted blocks N could span multiple contiguous 16-byte blocks. This means that the skipped blocks will be (10-N).

10.3.3.3 Crypt KeyID Box

10.3.3.3.1 Definition

Box Types: ‘ckid’
Container: Scheme Information Box (‘schi’)
Mandatory: No
Quantity: Zero or One

The KID is optional, and may be over-ridden by the signaling used, even if present. This box may be used, for example, to provide a ‘safety back-pointer’ and make files self-documenting.

10.3.3.3.2 Syntax

```
aligned(8) class CryptKeyIDBox extends FullBox(‘ckid’, version=0, flags=0)
{
   unsigned int(8)[16]  KID;
}
```

10.3.3.4 Crypt IV Box

10.3.3.4.1 Definition

Box Types: ‘criv’
Container: Scheme Information Box (‘schi’)
Mandatory: No
Quantity: Zero or One

The IV is optional, and may be over-ridden by the signaling used, even if present. This box may be used, for example, to provide a ‘safety back-pointer’ and make files self-documenting.

10.3.3.4.2 Syntax

```
aligned(8) class CryptIVBox extends FullBox(‘criv’, version=0, flags=0)
{
      unsigned int(128)  InitializationVector;
}
```

10.3.3.5 CryptNonVideoNALTypes Box

10.3.3.5.1 Definition

Box Types: ‘cnvd’
Container: Scheme Information Box (‘schi’)
Mandatory: No
Quantity: Zero or One

Definition of this box is only needed if certain non-VCL NAL units are desired for encryption.

10.3.3.5.2 Syntax

```
aligned(8) class CryptNonVideoNALTypesBox extends FullBox(‘cnvd’, version=0,
flags=0)
{
  unsigned int(8)  encrypted_non_video_nal_list[];   // to end of box
}
```

10.3.3.5.3 Semantics

encrypted_non_video_nal_list is a list, to the end of the box of non-VCL NAL unit types that are encrypted using the same pattern as specified in the Crypt Span Box starting from the end of the NAL unit header. Each entry in this list signals a non-VCL NAL unit type that is protected, where the entry is a 5-bit (AVC) or 6-bit (HEVC) value, padded on the leading end with three(AVC) or two(HEVC) zero bits.

10.3.4 Audio Streams

For audio tracks every audio sample is encrypted starting at offset 0 (no preamble) up to the last 16-byte boundary, leaving any trailing 0-15 bytes in the clear. The IV is reset at every sample.

The Scheme Information Box (‘schi’) for audio tracks is shown below. The contained boxes are defined above.

```
aligned(8) class SchemeInformationBox
extends Box(‘schi’, version=0, flags=0)
{
   CryptKeyIDBox    ckid_atom;    // optional
   CryptIVBox       criv_atom;    // optional
}
```