# DASH-IF Implementation Guidelines: Content Protection Information Exchange Format (CPIX)

**July 17, 2015**

**DASH Industry Forum**

**Version 1.0**

# Scope

The scope of this document is to define a Content Protection Information Exchange Format (CPIX) for MPEG-DASH and DASH-IF based video streaming. The CPIX file contains key and DRM information used for encrypting and protecting DASH content, and can be used for exchanging this information among entities needing it in many possibly different workflows for preparing DASH content. The CPIX file itself can be encrypted and authenticated so that its receivers can be sure that its confidentiality and integrity are also protected.

The current version of the CPIX document remains generic and only defines a container for carrying key and DRM information used for encrypting and protecting DASH content. Future versions of this document may also include, for example, definition of APIs for exchanging CPIX files and additional specifications addressing some specific workflows.

The current version of the CPIX document provides elements for supporting key rotation, nevertheless it does not fully define how to use this for use cases needing key rotation. Note that it is expected that an update to this specification will clarify how such use cases can be supported.

# Disclaimer

This is a document made available by DASH-IF. The technology embodied in this document may involve the use of intellectual property rights, including patents and patent applications owned or controlled by any of the authors or developers of this document. No patent license, either implied or express, is granted to you by this document. DASH-IF has made no search or investigation for such rights and DASH-IF disclaims any duty to do so. The rights and obligations which apply to DASH-IF documents, as such rights and obligations are set forth and defined in the DASH-IF Bylaws and IPR Policy including, but not limited to, patent and other intellectual property license rights and obligations. A copy of the DASH-IF Bylaws and IPR Policy can be obtained at http://dashif.org/.

The material contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, this material is provided AS IS, and the authors and developers of this material and DASH-IF hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of workmanlike effort, and of lack of negligence.

In addition, this document may include references to documents and/or technologies controlled by third parties. Those third party documents and technologies may be subject to third party rules and licensing terms. No intellectual property license, either implied or express, to any third party material is granted to you by this document or DASH-IF. DASH-IF makes no any warranty whatsoever for such third party material.

# Contents

# 1 Introduction

## 1.1 General

This document defines a container allowing exchanging between entities content protection information. Typically, content protection information is made of keys used for encrypting content and any associated DRM specific information. There may be one or several keys and these keys may be protected by one or several DRMs, hence there may be one or several DRM specific information. There is no assumption on the entities exchanging this information but it is not expected that a client device will use this exchange format. The goal is to allow entities involved in the content preparation workflow to get the content protection information so that the MPD can be generated with all content protection information.

## 1.2 References

[DASH] ISO/IEC 23009-2:2014 Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats.

[DASH-IF-IOP] Guidelines for Implementation: DASH-IF Interoperability Points, March 2015.

[DASH-attributes] http://www.dashif.org/identifiers/content-protection/

[RFC6030] IETF RFC 6030, “Portable Symmetric Key Container (PSKC)”, October 2010.

[CPIX-XML] http://dashif.org/wp-content/uploads/2014/12/DASH-IF-CPIX-v1.0.zip

## 1.3 Normative Language

See [DASH-IF-IOP] section 2.3.

## 1.4 Terms & Definitions

**Content**: One or more audio-visual elementary streams and the associated MPD if in DASH format.

**Content Key**: A cryptographic key used for encrypting part of the Content.

**Content Protection**: The mechanism ensuring that only authorized devices get access to Content.

**DRM Signalization**: The DRM specific information to be added in Content for proper operation of the DRM system when authorizing a device for this Content. It is made of proprietary information for licensing and key retrieval.

**PSSH**: “Protection System Specific Header” box that is part of an ISOBMFF file. This box contains DRM Signalization.

# 2 Use Cases and Requirements

## 2.1 Introduction

Content Keys and DRM Signalization need to be created and exchanged between some system entities when preparing content. The flows of information are of very different nature depending on where Content Keys are created and also depending on the type of content that can be either On-Demand or Live.

This section presents different use cases where such exchanges are required.

## 2.2 Overview of the End to End Architecture

This informative section gives a general overview of the context in which content protection information made of keys and DRMs signalization need to be exchanged between entities in the backend. It completes section 7.5 of [DASH-IF-IOP] by putting more emphasis on the backend aspects.

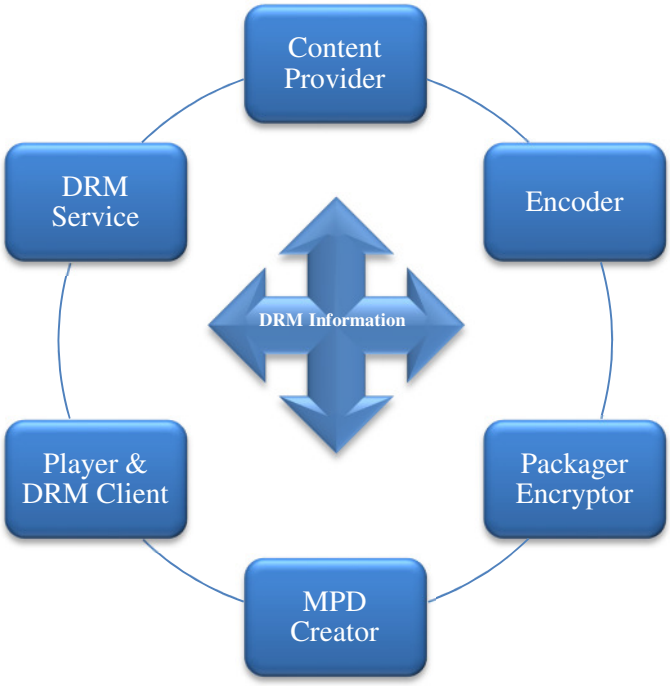



**Figure 1: Logical roles that exchange DRM information and media.**

Figure 1 shows logical entities that may send or receive DRM information such as media keys, asset identifiers, licenses, and license acquisition information. A physical entity may combine multiple logical roles, and the point of origin for information, such as media keys and asset identifiers, can differ; so various information flows are possible. This is an informative example of how the roles are distributed to facilitate the description of workflow and use cases. Alternative roles and functions can be applied to create conformant content. The different roles are:

**Content Provider** – A publisher who provides the rights and rules for delivering protected media, also possibly source media (mezzanine format, for transcoding), asset identifiers, key identifiers (KID), key values, encoding instructions, and content description metadata.

**Encoder** – A service provider who encodes Adaptation Sets in a specified media format, number of streams, range of bitrates and resolutions, seamless switching constraints, etc., possibly determined by the publisher. An asset identifier needs to be assigned to each

encoded track in order to associate a key identifier, a Representation element in an MPD, a possible 'pssh' box in the file header, and a DRM license separately downloaded.

**Packager / Encryptor** – A service provider who encrypts and packages media files, inserting default_KID in the file header '`tenc`' box, initialization vectors and subsample byte ranges in track fragments indexed by '`saio`' and '`saiz`' boxes, and possibly packages '`pssh`' boxes containing license acquisition information (from the DRM Provider) in the file header. Tracks that are partially encrypted or encrypted with multiple keys require sample to group boxes and sample group description boxes in each track fragment to associate different KIDs to groups of samples. The Packager could originate values for KIDs, media keys, encryption layout, etc., then send that information to other entities that need it, including the DRM Provider and Streamer, and probably the Content Provider. However, the Packager could receive that information from a different point of origin, such as the Content Provider or DRM Provider.

**MPD Creator** – The MPD Creator is assumed to create one or more types of DASH MPD, and provide indexing of Segments and/or '`sidx`' indexes for download so that players can byte range index Subsegments. The MPD must include descriptors for Common Encryption and DRM key management systems, and SHOULD include identification of the default_KID for each AdaptationSet element, and sufficient information in UUID ContentProtection Descriptor elements to acquire a DRM license. The default_KID is available from the Packager and any other role that created it, and the DRM specific information is available from the DRM Provider.

**DRM Client** – Gets information from different sources: MPD, Media files and DRM License.

**DRM Service** – The DRM Provider creates licenses containing a protected media key that can only be decrypted by a trusted client.

The DRM Provider needs to know the default_KID and DRM SystemID and possibly other information like asset ID and player domain ID in order to create and download one or more licenses required for a Presentation on a particular device. Each DRM system has different license acquisition information, a slightly different license acquisition protocol, and a different license format with different playback rules, output rules, revocation and renewal system, etc. The DRM Provider typically must supply the Streamer and the Packager license acquisition information for each UUID ContentProtection Descriptor element or 'pssh' box, respectively.

The DRM Service may also provide logic to manage key rotation, DRM domain management, revocation and renewal and other content protection related features.

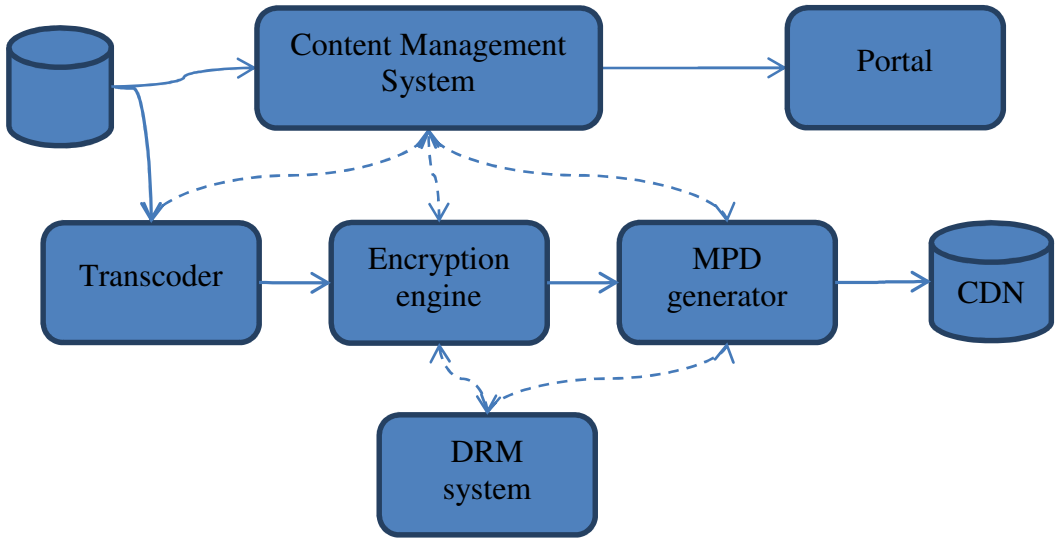
## 2.3 Content On-Demand

The flow for preparing Content On-Demand requires that a file is available non-encrypted, ideally in the maximum resolution so that DASH content can be prepared.

One possible flow is that a Content Management System (CMS) creates a workflow ensuring that DASH Content is prepared. The CMS makes the file available to a transcoder. The transcoder outputs the segmented files that can be encrypted. The encryption engine either generates the Content Keys or requests them from a DRM system. The DRM system also provides any information to be added in the PSSH boxes. When the encrypted DASH Content is ready, the MPD is generated by a "MPD Generator". It asks the DRM system the required DRM signalization to be added in the MPD. DASH content is then uploaded by the CMS on a CDN making it available to users.
In parallel, editorial metadata is exported to the Portal, enabling access to users. DRM

systems receive relevant metadata information that needs to be included in the license (output controls) when creating a license.

This flow is summarized in the following figure where arrows show the flow of information.




## 2.4 Live

Metadata is regularly imported with new or updated information. Metadata can include different type of information on the EPG events such as the duration of the event, the list of actors, the output controls usage rules, a purchase window…

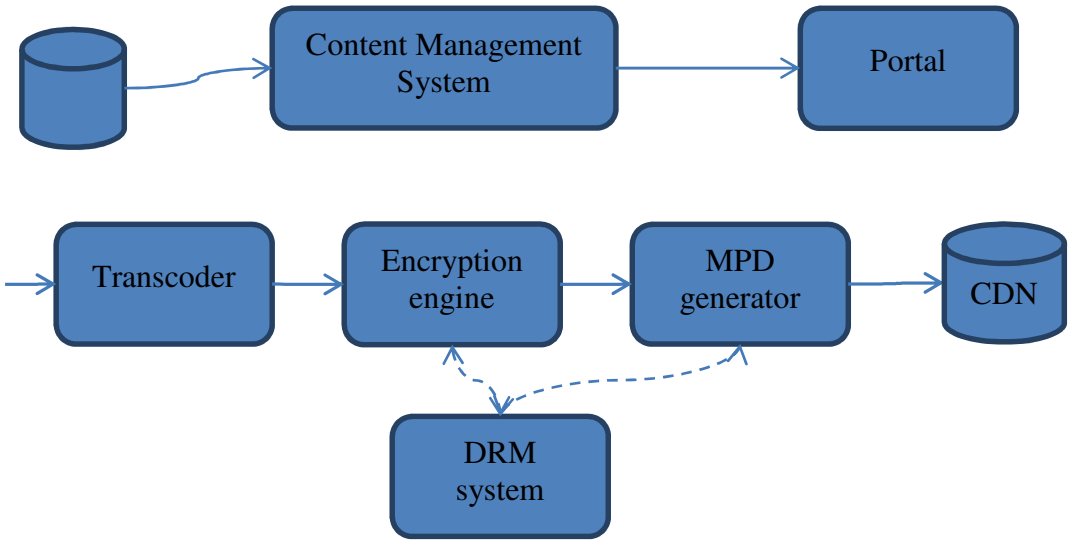
Content is continuously received, transcoded in the desired format and encrypted if any type of entitlement is required.

One or many Content Keys can be used if key rotation is used or not. Such setting is static and configuration is hard-coded in the relevant equipment, hence a Content Management System is not required for this workflow to operate. As for Content on-Demand, keys are generated by the encryption engine or the DRM system and are available to all DRM systems and the encryption engine at the right moment depending on how these keys are used. The encoder requests to the DRM systems their specific signaling, if any, to be added in the MPD.

Encrypted segments and the MPD are uploaded on a CDN making it available to users.

Metadata is exported to the Portal, enabling access to users. DRM systems receive relevant metadata information that needs to be included in the license (output controls).

This flow is summarized in the following figure where arrows show the flow of information.




## 2.5 Catch-up

Live Content has already been encoded and encrypted (if required) for Live unicast. All DRM systems have access to the keys.

Additional metadata may be required for ensuring that events are effectively available in catch-up. These are made available to the Portal and some Live events are identified as being able to be replayed as On-demand. Optionally, the operator may choose to replace the advertising content with targeted ads.

## 2.6 Electronic Sell Through

In order to make available its Content in a defined and controlled quality, a content owner is preparing it. Preparation includes transcoding to the desired format and encryption of the resulting segments. The content owner is generating also the Content Key(s). At the end of the process, Content is ready and stored along with the Content Key(s).

Later the content owner distributes the prepared Content to multiple locations, in addition metadata describing it is also made available to retail platforms so that Content becomes salable on multiples Portals. In parallel, the content owner distributes the Content Key(s) to any authorized DRM system. A DRM system is authorized if it is one used by one of the Portal that has this Content for sale.

## 2.7 Requirements

It shall be possible to exchange Content Key(s) and DRM Signalization between entities involved in DASH Content preparation workflows, an example of such interface where the exchange shall be possible is between a DRM system and the encryption engine.

It shall be possible that the MPD generator receives DRM signalization for several DRMs.

Update of Content Key(s) shall be possible at periodic time or based on events, following Section 5 of [DASH-IF-IOP]. Some period of time could be in the clear (no encryption).

It shall allow generating MPD conformant to [DASH-IF-IOP].

Content Key(s) shall be secured over the interface.

# 3 XSD Schema Definition

## 3.1 Introduction

This section describes the Content Protection Information Exchange Format to provide a framework to securely exchange Content Key(s) and DRM Signalization between different system entities (see Section 2.2). This is an XML file that is described by the XSD provided in Section 5.1. This section describes in details elements part of the file.

## 3.2 Structure Overview

The structure is similar to the MPD structure defined in [DASH]. A Presentation is the root element of this schema and contains all information required for getting the common encryption keys which is used to encrypt all representations within an adaptation sets. It follows these principles:

- Following the constraints defined in Section 5.4.4 of [DASH-IF-IOP], it is assumed that the same key is used for encrypting all Representations of a given Adaptation Set. For supporting key rotation, several Content Keys can be used for encrypting all Representations, each key with a validity period (`@start`, `@end` see Section 3.3.5)
- The same XML file can be shared between several receiving entities, hence, each one must be able to decrypt keys and must be properly identified.

Taking this into account, the Presentation contains:

- **`DeliveryData`**, each instance of the **`DeliveryData`** describes an entity which permitted to decrypt common content encryption key which XML file contains.
- **`AdaptationSet`**: Each **`AdaptationSet`** contains the **`DefaultKey`** information (the common content encryption key itself and all associated DRM Signalizations which is protection system specific information for every DRM.)

The Default Key and content keys can be encrypted inside the XML file using information provided in the **`DeliveryData`** element. The XML file also allows storing the common content encryption keys in the clear and then the protection of the delivery mechanism is used for securely deliver the file.

Figure 2 shows the first elements of the structure. Detailed description of the structure is given in the following sections.

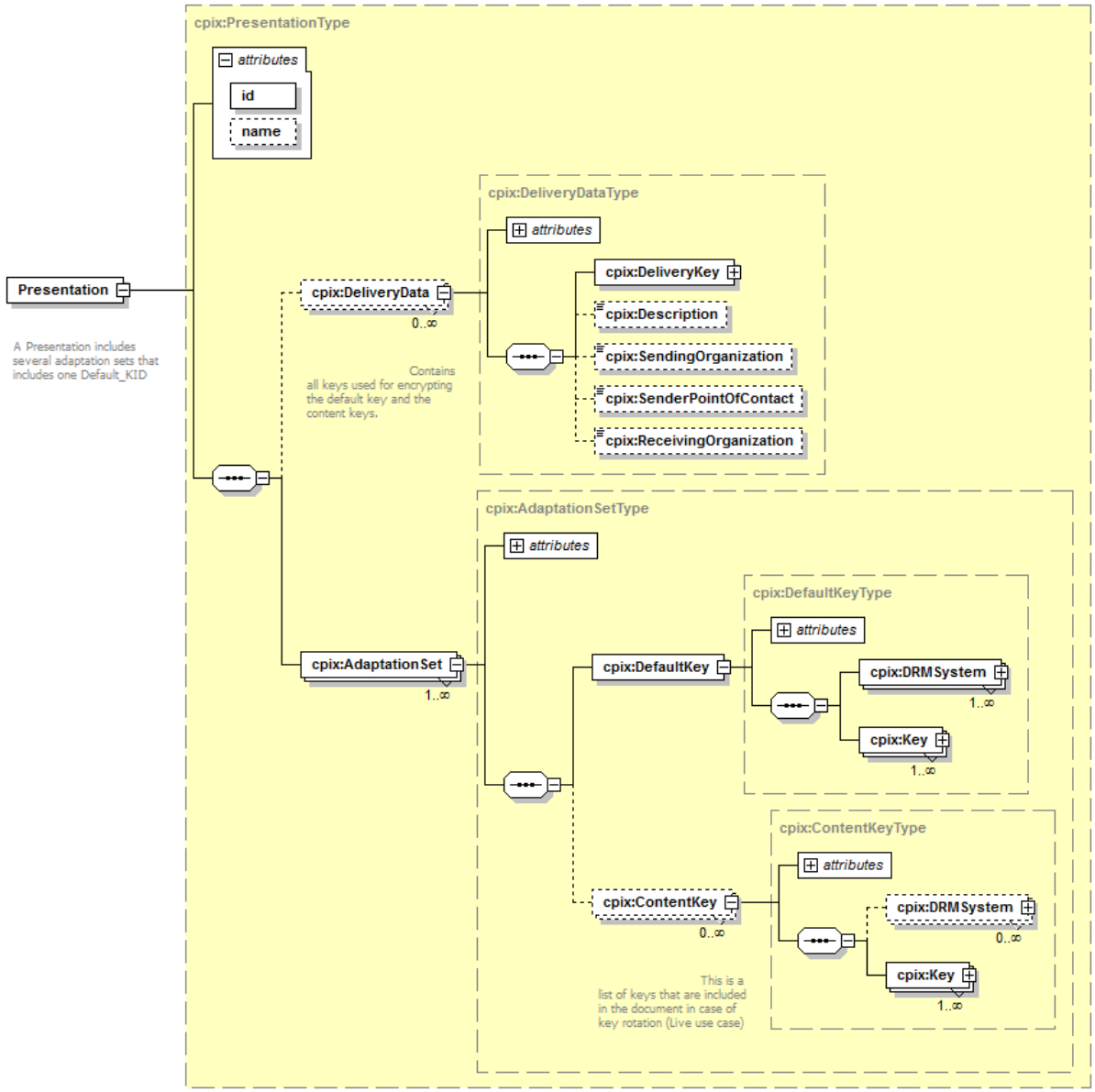



**Figure 2: Content Protection Information Exchange Format high level view.**

## 3.3 Hierarchical data model

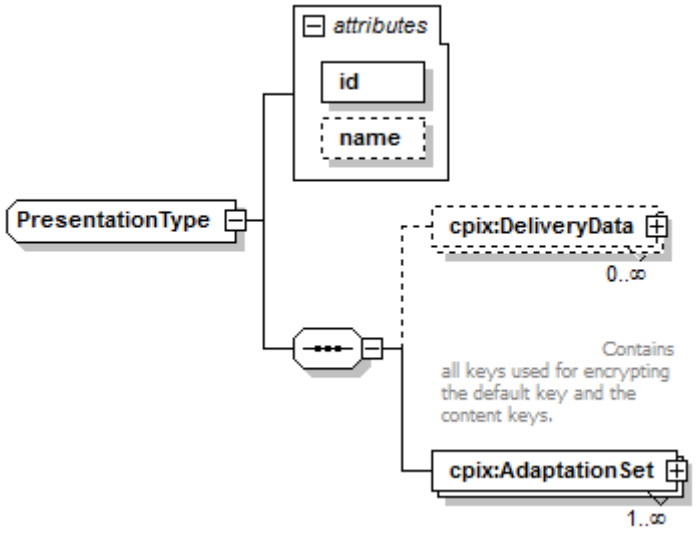
In all tables of this section, the following convention is used

- **`Elements`** are bold and the "Use" values are `<minOccurs>…<maxOccurs>` (N=unbounded).
- **`Attributes`** are non-bold preceded with an @ and the "Use" values are M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory.

The XSD schema for this model is provided in Section 5.1.

### 3.3.1 Presentation

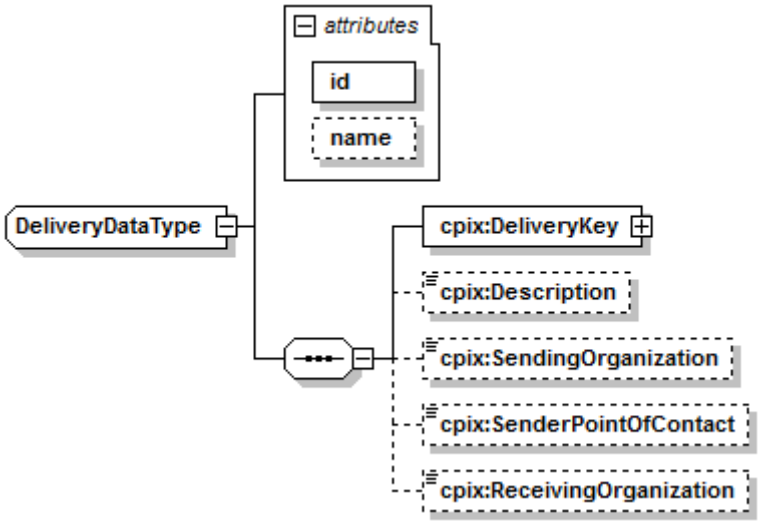
| Element or Attribute | Use | Description |
|---|---|---|
| **Presentation** | | The root element that carries the Content Protection Information for a Media Presentation. |
| @id | M | It specifies an identifier for the Media Presentation. It is also referred as the Asset ID. It is recommended to use an identifier that is unique within the scope in which this key file is published. |
| @name | O | It is the name of the Presentation. |
| **DeliveryData** | 0…N | It contains the required information allowing defining which entities can get access to the keys encrypted in this key file.<br>There is one DeliveryData element per entity capable of decrypting keys in the key file. If this element is not present, then the keys are in the clear in the file. |
| **AdaptationSet** | 1...N | It contains all information on keys used for encrypting the Adaption Sets and all the DRM information required for properly generate content and create the MPD.<br>There is one AdaptationSet element per Adaptation Set the Representation has. |




### 3.3.2 Delivery Data

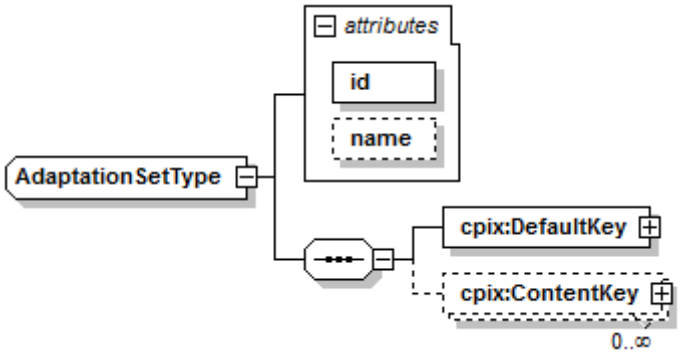
| Element or Attribute | Use | Description |
|---|---|---|
| **DeliveryData** | | |
| @id | M | It specifies an identifier for the Delivery Data. It is recommended to use an identifier that is unique within the scope in which this key file is published. |

| | | |
|---|---|---|
| @name | O | It is the name of the Delivery Data. |
| **DeliveryKey** | 1 | It contains the crypto material required to uniquely identify the entity capable of decrypting the DefaultKey and Content Key elements in this key file. |
| **Description** | 0…1 | A description of the element. |
| **SendingOrganization** | 0…1 | The description, such as the name, of the entity generating this key file. |
| **SenderPointOfContact** | 0…1 | The information, such as an email address, of the Sending Organization. |
| **ReceivingOrganization** | 0…1 | The description, such as the name, of the entity capable of decrypting keys in this key file. |



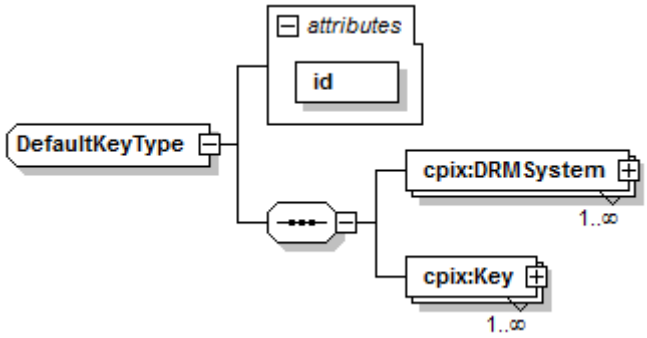

### 3.3.3 Adaptation Set

| Element or Attribute | Use | Description |
|---|---|---|
| **AdaptationSet** | | |
| @id | M | It specifies an identifier for the Adaptation Set. It is recommended to use an identifier that is unique within the scope in which this key file is published. |
| @name | O | It is the name of the Adaptation Set. |
| **DefaultKey** | 1 | This is the default key as defined in [DASH] and all related DRM information. |
| **ContentKey** | 0…N | These are the keys and all related DRM information used when the Adaptation Set is encrypted with key rotation. |




### 3.3.4 Default Key

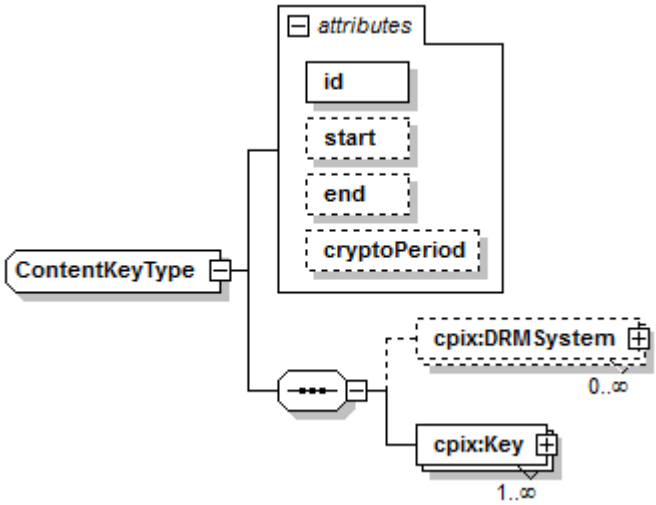
| Element or Attribute | Use | Description |
|---|---|---|
| **DefaultKey** | | |
| @id | M | It specifies an identifier for the Default Key. It is recommended to use an identifier that is unique within the scope in which this key file is published. |
| **DRMSystem** | 1…N | This is the DRM information for every DRM that is used for protecting all Representations part of this Adaptation Set.<br>There are as many **DRMSystem** elements as there are DRM allowing accessing this adaptation Set. |
| **Key** | 1…N | This is the key itself. It is represented as KeyType as defined in [RFC6030]. It can be encrypted or in the clear.<br>If it is encrypted, then the Extension element shall contain a **DeliveryKeyReference** element allowing retrieving the information allowing decrypting this key.<br>If it is encrypted, there are as many **Key** elements as there are **DeliveryData** capable of accessing (decrypting) it. |




### 3.3.5 Content Key

| Element or Attribute | Use | Description |
|---|---|---|
| **ContentKey** | | |

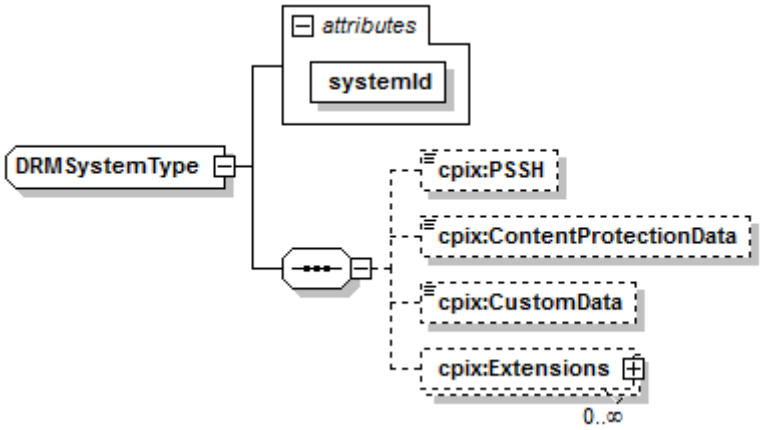
| | | |
|---|---|---|
| `@id` | M | It specifies an identifier for the Content Key. It is recommended to use an identifier that is unique within the scope in which this key file is published. |
| `@start` | O | This is the absolute time when this key starts to be used for encrypting the Representations part of this Adaptation Set. The attribute must be present is the `@cryptoPeriod` attribute is not present. |
| `@end` | O | This is the absolute time when this key stops to be used for encrypting the Representations part of this Adaptation Set.<br>If not present, then the end time is defined by the earliest @start time of another **`ContentKey`** element part of the same **`AdaptationSet`** element. |
| `@cryptoPeriod` | O | This is the unique identifier of the period of time when this key starts to be used for encrypting the Representations part of this Adaptation Set. The attribute must be present is the `@start` attribute is not present. |
| **`DRMSystem`** | 0…N | This is the DRM information for every DRM that is used for protecting all Representations part of this Adaptation Set.<br>There are as many **`DRMSystem`** elements as there are DRM allowing accessing this adaptation Set. |
| **`Key`** | 1…N | This is the key itself. It is represented as KeyType as defined in [RFC6030]. It can be encrypted or in the clear.<br>If it is encrypted, then the Extension element shall contain a **`DeliveryKeyReference`** element allowing to retrieve the information allowing to decrypt this key.<br>If it is encrypted, there are as many Key elements as there are **`DeliveryData`** capable of accessing (decrypting) it. |



### 3.3.6 DRM System

The DRMSystem element contains all information on a DRM that can be used for retrieving licenses for getting access to an Adaptation Set.

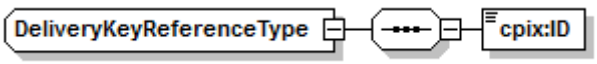
| Element or Attribute | Use | Description |
|---|---|---|
| **DRMSystem** | | |
| @systemId | M | This is the unique identifier of the DRM system. Values are defined on [DASH-attributes]. |
| **PSSH** | 0…1 | This is the full PSSH box that is added either under the moov box or the moof box in the ISOBMFF depending on the parent element of the DRMSystem element (**DefaultKey** and **ContentKey** respectively). |
| **ContentProtectionData** | 0…1 | This is the full XML element to be added in the MPD under the **ContentProtection** element for this DRM. |
| **CustomData** | 0…1 | This is any data the DRM system needs to exchange with other entities for proper generation of the MPD. |




### 3.3.7 Delivery Key Reference

The DeliveryKeyReference element is used for linking a Key element to a DeliveryData element. It is added under the Extension element of the Key element as allowed by [RFC6030].

| Element or Attribute | Use | Description |
|---|---|---|
| **DeliveryKeyReference** | | |
| @id | M | This is the identifier of the **DeliveryData** element that contains the required key material for decrypting the Key element it is inserted in. |



# 4 Workflow Overview

## 4.1 Introduction

There are two models that are possible, as shown in Figure 3 and Figure 4 with the example of an Encryptor and DRM Systems as two entities exchanging information.

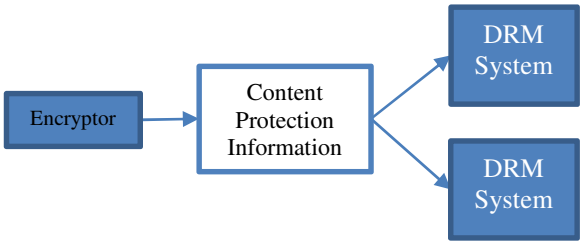



**Figure 3: Push model.**

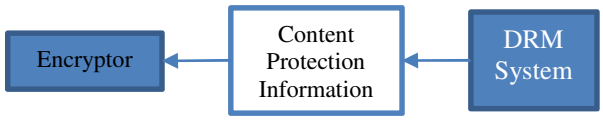



**Figure 4: Pull model.**

Both models require content protection information and content keys been exchanged between two or more entities. In the examples in Figure 3and Figure 4 the entities are the Encryptor and DRM System:

- The Push model allows, in this case, the Encryptor to generate keys and to push them to one or many DRM systems. The Encryptor could expect to receive from the DRM systems some DRM signalization (content protection information).
- The Pull model allows the Encryptor to pull keys and DRM signalization (content protection information) from a DRM system. In this case keys are generated by the DRM System.

The Content Protection Information Exchange Format proposed in this document allows supporting both models. The Exchange Format allows sending and receiving the key and DRM signalization information. Each model has its own advantages and limitations. It is a matter of requirements for selecting one of the models or both.

It is recommended to encrypt the Content Protection Information which is exchanged between entities as it contains very sensitive data. The Content Protection Information consists of two main elements:

- The **`DeliveryData`** that contains required information allowing defining which entities can get access to the encrypted keys in the key file as well as information about the sending entity itself.
- The **`AdaptationSet`** that contains the encrypted keys.

Before exchanging key information in a secure manner both entities must know about each other and share public keys so that one entity could encrypt data and the other entity could decrypt it. This important step of Trust establishment is out of the scope of this document.

The encryption of the keys is the second important step. It could be done within the Exchange Format or on the distribution protocol. This important step of securing the Key exchange is out of the scope of this document. The DRM signalization contains proprietary information and if protection is required, it is under the DRM responsibility.

Some information on the possibilities proposed by the industry for securing the exchange is given in Section 4.4.

## 4.2 Secure Push Model

This informative section shows a possible workflow for securing the exchange of the key information between entities when the Push model is used. In this model, shown in Figure 3, the Encryptor is the entity which is taking responsibility for generating the keys, protecting them and pushing them into the DRM Systems.

- The first step is the Trust establishment. Public keys must be exchanged between two or more entities (the Encryptors and the DRM Systems) prior exchanges.
- Once the trust is established and the necessary associated key material is shared between entities, keys for encrypted content can be exchanged. The Encryptor is encrypting these keys using DRM Systems public keys. The DRM Systems can decrypt using their own private key.
- The Encryptor provides **`DeliveryData`** elements with crypto material required to uniquely identify the entity capable of decrypting the **`AdaptationSet`** element. The Content Protection Information might contain one or multiple or no **`DeliveryData`** elements and at least one **`AdaptationSet`** element must be present.

All these steps are summarized in Figure 5.

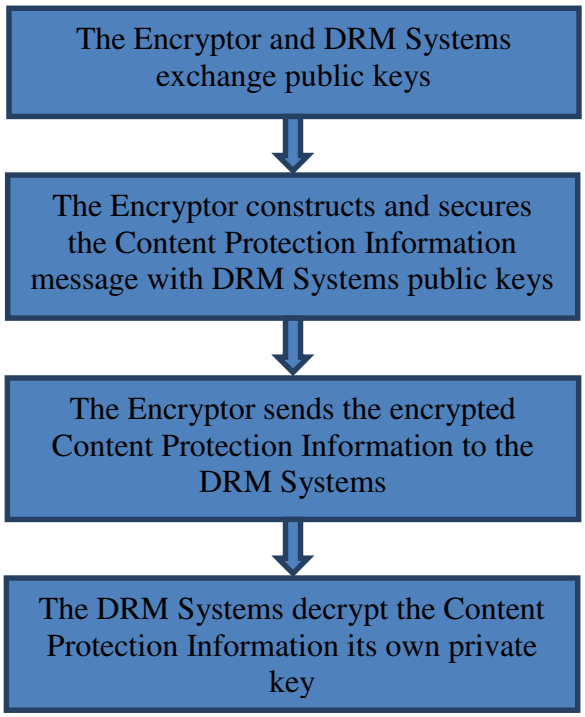



**Figure 5: Secure Push model steps.**

## 4.3 Secure Pull Model

This informative section shows a possible workflow for securing the exchange of the key information between entities when the Pull model is used. In this model, shown in Figure 4, the Encryptor can pull Content Protection Information directly from a DRM System. In this case, the DRM System is generating keys and is encrypting them for a secure delivery to the Encryptor.

- As in the case of the Push model, the first step is the Trust establishment. Public keys must be exchanged between two or more entities (the Encryptors and the DRM Systems) prior exchanges.

- The DRM System will use the public key of the Encryptor to encrypt keys to be inserted in the DeliveryData element and will send it to Encryptor.
- The Encryptor can decrypt the keys using its private key.

All these steps are summarized in Figure 6.

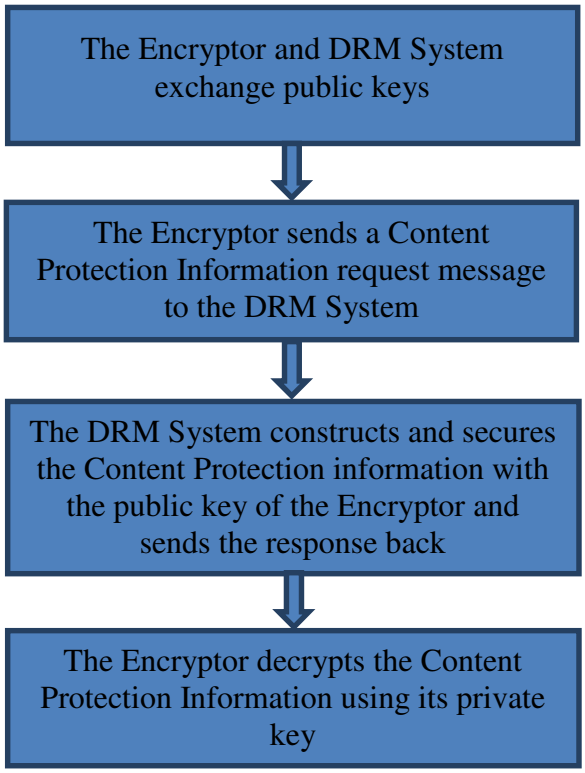



**Figure 6: Secure Pull model steps.**

## 4.4 Real world scenarios

This section shows some examples of existing implementations of the models presented in Sections 4.2 and 4.3.

**Content encryption by the Encoder.**

One of the common scenarios is when Encoder performs content encryption. Encryption is a feature of encoder. Encoder fetches the content protection information and content key from DRM System and then uses this information to apply encryption after or during the encoding.

**Content encryption by Media Server**

There are companies who build Media Servers and also integrate DRM encryption within their products same as encoders Media Servers also integrated with DRM System and fetch encryption keys from DRM System. Media Servers can encrypt the content in real time as they serve it.

# 5 XML Schema and Examples

## 5.1 XSD file

The XSD is available on DASH-IF web site as part of zip files [CPIX-XML] containing examples. Validation of this XSD requires the XSD associated to [RFC6030] in a file called pskc.xsd.

## 5.2 Examples

The zip file available on DASH-IF web site contains two examples of XML files [CPIX-XML].

**Encrypted keys**

This example shows a XML key file where keys can be decrypted by two entities ("Authorization Service 1234" and "Authorization Service 5678"). Both are identified by their X509 certificates.

It contains two Adaptation Sets, each with only Default Key. The Default Key is therefore encrypted two times (once for Authorization Service 1234 and once for Authorization Service 5678).

In term of DRMs, there are two different DRMs that can be used for accessing the Adaptations Sets. There are therefore two **`DRMSystem`** elements per Adaptation Set.

**Clear Keys**

This example shows a XML key file where keys are not encrypted. It can be read by any entity and therefore does not contain **`DeliveryData`** elements.

It contains two Adaptation Sets, each with only Default Key. The Default Key is available as plain data for each Adaptation Set.

In term of DRMs, there are two different DRMs that can be used for accessing the Adaptations Sets. There are therefore two **`DRMSystem`** elements per Adaptation Set.