OMA DRM Rights Issuer Common Domain Profile

Candidate Version 1.0 – 22 Nov 2011

Open Mobile Alliance

OMA-TS-RICD_Profile-V1_0-20111122-C

Contents

Figures

Tables

1. Scope

The profile describes additions and clarifications to OMA DRM V2.1 Enabler Technical specifications concerning OMA DRM Domain functionality and associated Domain RO delivery. It also clarifies how multiple CEKs may be delivered via a single RO.

2. References

2.1 Normative References

[SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[DRM-v2.1] "Digital Rights Management V2.1", Open Mobile Alliance™, OMA-TS-DRM-DRM-V2_1, URL:http://www.openmobilealliance.org

[DRMCF-v2.1] DRM Content Format V2.1", Open Mobile Alliance™, OMA-TS-DRM-DCF-V2_1, http://www.openmobilealliance.org/

[DRMREL-v2.1] DRM Rights Expression Language V2.1", OMA, Open Mobile Alliance™, OMA-TS-DRM-REL-V2_1, http://www.openmobilealliance.org/

[RFC2387] "The MIME Multipart/Related Content-type", E. Levinson, 1998, URL:http://www.ietf.org/

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version x.y, Open Mobile Alliance™, OMA-ORG-Dictionary-Vx_y, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Content	One or more Media Objects
Device	A Device is the entity (hardware/software or combination thereof) within a user-equipment that implements a DRM Agent. The Device is also conformant to the OMA DRM specifications. In the case where functionality is specific to either Connected Devices or Unconnected Devices the explicit terminology (i.e. Unconnected Device or Connected Device) will be used, in all other cases the term Device generically applies to both Connected Devices and Unconnected Devices.
Domain	A set of Devices, which are able to share Domain Rights Objects. Devices in a Domain share a Domain Key. A Domain is defined and managed by an RI or a set of RIs.
Domain baseID	The first (leading) characters that precede the Domain Generation Counter in the Domain Identifier.
Domain Identifier	A unique string identifier of the Domain Key
Domain Key	A 128 bit symmetric cipher key
Domain Generation	A Counter reflecting the number of times the Domain has been upgraded. The Domain Generation is a part of the Domain Identifier (the last three digits).
Domain Context	The Domain Context consists of information necessary for the Device to install Domain Rights Objects, such as Domain Key, Domain Identifier and Expiry Time.
Domain Context Expiry Time	An absolute time after which the Device is not allowed to install ROs for this Domain. Usage of ROs installed before the expiry time are not affected by the expiry.
Domain Revocation	The process of an RI indicating that a Domain Key is not trusted for protection of Domain ROs.
Domain Rights Object	An RO that is dedicated to Devices in a particular Domain by means of a Domain Key.
DRM Agent	The entity in the Device that manages Permissions for Media Objects on the Device.
DRM Content	Media Objects that are consumed according to a set of Permissions in a Rights Object.
Join Domain	The process of an RI including a Device in a Domain.
Leave (De-Join) Domain	The process of an RI excluding a non-revoked Device from a Domain.
Media Object	A digital work e.g. a ringing tone, a screen saver, a Java game or a Composite Object.
Rights Issuer	An entity that issues Rights Objects to OMA DRM Conformant Devices.
RI Context	RI Context (Rights Issuer Context) consists of information that was negotiated with a given Rights Issuer, during the 4-pass Registration Protocol such as RI ID, RI certificate chain, version, algorithms and other information. This RI Context is necessary for a Device to successfully participate in all the protocols of the ROAP suite, except the Registration Protocol.
RI Common Domain	A set of RIs which share domain context, where some or all RIs of RICD may manage the Domain and all or some the RIs of the RICD generate and issue Domain ROs.
Rights Object	A collection of Permissions and other attributes which are linked to DRM Content.
Rights Object Acquisition Protocol (ROAP)	A protocol defined within this specification. This protocol enables Devices to request and acquire Rights Objects from a Rights Issuer.

ROAP Trigger An XML document including a URL that, when received by the Device, initiates the ROAP.

3.3 Abbreviations

CEK	Content Encryption Key
DCF	DRM Content Format
DRM	Digital Rights Management
HTTP	HyperText Transfer Protocol
OMA	Open Mobile Alliance
RI	Rights Issuer
RICD	Rights Issuer Common Domain
RO	Rights Object
ROAP	Rights Object Acquisition Protocol

4. Introduction

This specification details the additions, changes, and clarifications to [DRM-v2.1], implement the RICD profile which includes the following:

1. Describe how Trusts Authorities RIs that share and support a Common Domain context shall be identified by Devices.
2. Describe how Trust Authorities RICD RIs that support only management functions shall be identified by devices.
3. Describe how Trust Authorities RICD RIs that only provide ROs shall be identified by devices.
4. Clarify how multiple CEKs may be transfer via a single RO to support new content formats that require multiple CEKs (e.g. one for all audio tracks and one for all video tracks).
5. Define the Rights Context information to be transported along with RO in a DCF or other content formats to enable consumption of content when a connection cannot be made to the issuing RI.

4.1 Version 1.0

This version 1.0 introduces RIs that share a Common Domain context where some or all RIs manage the domain and where some or all RIs generate Domain ROs. These RIs are referred to as RICD and throughout this profile the term RI may be replaced with RICD.

The Trust Authority exists to support business requirements associated with the use of OMA DRM and so the decision as to which RI preforms domain management or generates Domain ROs is left to Trust Authority so it can respond to business needs.

This profile enables the consumption of Domain ROs by including RI context information in the DCF so when a Device does not have RI context and is unable to connect to the RI is may render the content as prescribed by the Trust Authority.

The DRM RICD PROFILE does not require the following OMA DRM 2.1 features:

- Metering including all associated protocols and commands.
- Rights Object installation confirmation is not required.
- Devices are not required to support WBXML decoding of ROAP Triggers.

5. Modifications to DRM TS

5.1 Domain Functionality Modifications

Replaces Domains chapter of [DRM-V2.1]

5.1.1 Overview

A Domain is a set of Devices that possess a common Domain Key provisioned by a Rights Issuer. Devices in a Domain may share Domain Rights Objects and are able to consume and share any DCFs controlled by Domain Rights Objects.

The OMA DRM Domain concept is network centric. An RI defines the Domains, manages the Domain Keys, and controls which and how many Devices are included and excluded from the Domain. A user may request to add Devices to a Domain before acquiring Domain-bound content, or make these requests incrementally after receiving Domain-bound content.

A Domain is associated with a unique Domain Identifier, which includes a Domain Generation counter, and one or more Domain Keys. Multiple Domain Keys are a result of Domain upgrades performed by the Rights Issuer that manages the Domain. Each Domain Key corresponds to a specific Domain Generation. The value of the Domain Generation counter indicates the number of upgrades performed on the Domain.

Devices may join multiple Domains managed by one or more RIs.

5.1.2 Device Joins Domain

To join a Domain, a Device must have an RI Context established with the RI administering the Domain. A Device joining a Domain is the process of an RI authorizing a particular Device to be able to use all ROs for this Domain. When a Device joins a Domain it receives the necessary Domain information to be able to install Domain ROs.

A Device executes the Join Domain protocol (see sections 5.1.4 and 5.4.4 of [DRM-v2.1]) to join a given Domain. The result of a successful execution is the establishment in the Device of a Domain Context for the given Domain. The Domain Context includes Domain Key(s), Domain Identifier(s) and a Domain Expiry Time.

A Device MAY join multiple Domains managed by one or more RIs.

The Join Domain protocol is triggered by the **<joinDomain>** ROAP trigger.

If a Device joins a Domain with multiple Domain Generations (i.e. a Domain where more than one Domain Keys have been issued), the RI SHOULD issue to the Device the Domain Keys of all previous generations of the Domain, to allow use of all ROs bound to this Domain. But, if both the Device and RI are using the hash chain mechanism, the RI only needs to supply the most recent generation Domain key.

5.1.3 Domain RO Acquisition & Consumption

Domain ROs can be acquired by the same mechanism as Device ROs, using the 2-pass RO Request/Response protocol or the 1-pass RO Response protocol. The Device specifies the Domain Identifier in the RO Request. Domain ROs can also be acquired without being wrapped in a ROAP PDU, e.g. delivered to Devices as a result of a browsing session.

In order to consume a Domain RO, a Device MUST have a Domain Context for the Domain that the RO refers to. A Device MAY continue to consume Domain ROs that belong to a Domain where the Domain Context has expired. See section 5.1.7.2.1 for the procedures for installing Domain ROs.

5.1.4 Device Leaves a Domain

In order for a Device to leave a Domain, it must assure the RI that it has deleted all information about the Domain that enables it to use any ROs for the Domain. When leaving a Domain a Device MUST delete the associated Domain Context, without a Domain Context ROs issued for that Domain will no longer be consumable. When leaving a Domain a Device MAY, but is not required to, remove the corresponding Domain ROs and associated Content. The Device SHOULD obtain user confirmation before deleting Domain ROs and associated Content.

A Device MUST execute the Leave Domain protocol (see sections 5.1.5 and 5.4.4 of [DRM-v2.1]) to leave a Domain. A Device may do this by sending a LeaveDomainRequest message to the riURL as stored in the RI Context associated with the Domain Context or as a result of receiving a **<leaveDomain>** ROAP Trigger. The riURL from a <**leaveDomain**> ROAP trigger MUST be used if the LeaveDomain is triggered by a ROAP trigger (See section 5.2.1 of [DRM-v2.1]).

Prior to sending a Leave Domain Request, the Device MUST ensure that the corresponding Domain Context is deleted.

5.1.5 Domain Context Expiry

If a Device determines that a Domain Context has expired, for example as part of the process of consuming a Domain RO, then the Device SHOULD attempt to rejoin the Domain in order to establish a valid Domain Context. If the Device receives a response where the status is not equal to "Success", with the exception of "NotRegistered" and "DeviceTimeError", then it MUST NOT retry this attempt. For behaviour in the case where the status is equal to "NotRegistered" or equal to "DeviceTimeError" see section 5.3.6 of [DRM-v2.1].

The **<notAfter>** element in the **roap:JoinDomainResponse** specifies the domain context expiry. After the Domain Context has expired the DRM Agent MAY continue to consume existing Domain ROs (as per section 5.1.3). Installation of new Domain ROs is not permitted for an expired domain without domain renewal (as per section 5.1.7.2.1).

5.1.6 Domain Support Scenarios

5.1.6.1 Domain Support for Multiple Domains per Rights Issuer

To provide flexibility in Domain management, the system supports multiple Domains per Rights Issuer. The Device SHALL support the ability to join multiple Domains for each RI Context it establishes.

To ensure that each DRM Agent is able to provide a minimum level of functionality, a Device SHALL support at least 6 Domains, distributed among the established RI Contexts in any proportion.

The Device MAY optionally support more than 6 Domains. These additional Domains may also be distributed among the established RI Contexts in any proportion.

5.1.6.2 Domain Support for Multiple RIs per a Domain

To provide flexibility in Domain management, as an option, multiple RIs can share a single Domain context. RIs that share a domain context are identified by additional key purpose values in the Extension field of the Rights Issuer Certificate. Specifically, a RICD OID value that identifies the RI a member of a group of RIs that share a common domain context and is set and managed by the associated trust model. Domain Context is assumed across all RIs with the same RICD OID value and not per RI.

Furthermore, domain management and rights issuer functionality maybe be separated, where at the discretion of the trust model RIs that share domain context are divided, so that some RIs exclusively perform domain management function only and other RIs only generate Domain Right Objects. Trust model can indicate this in the RI certificate with additional OID values as needed. Note OID assignment is left to trust model in order to provide flexibility and accommodate market needs.

5.1.7 Domain RO Processing Rules

5.1.7.1 Overview

As a general principle, the processing rules for inbound Domain ROs are agnostic to the origin of the Domain RO i.e. it does not matter whether the Domain RO was delivered OTA from an RI or copied from another Device. There is no binding to a specific transport mechanism or protocol.

Domain ROs MAY be delivered to the Device either in the course of the RO acquisition protocol, inside a DCF file, as a separate standalone MIME object, or as part of a MIME multipart/related message [RFC2387]. As part of the installation of an RO, the Device must perform integrity and authenticity checks and replay attack related checks as described below.

5.1.7.2 Inbound Domain RO

The Device MUST support receiving a Domain RO in a ROAP-ROResponse message.

The Device MUST support receiving a Domain RO as a separate object.

The Device MUST support receiving a Domain RO inside a DCF or other file format as required by Trust Authority

Before installing and using a Domain RO to render the media objects inside the associated DCF the Device MUST process the Domain RO as defined in chapter 5.1.7.2.1.

5.1.7.2.1 Installing a Domain RO

When a Device receives a Domain RO, it MUST determine if it has a valid RI Context with the RI that issued the RO, by comparing the value of the **roap:ROPayload**'s **<riID>** element with the RI Identifiers in all valid RI Contexts stored in the Device. If the value of the **<riID>** element does not match that of an RI Identifier in a valid RI Context, the device SHALL NOT install the Domain RO. In this case the Device MAY keep the Domain RO and MAY send an HTTP GET to the URL specified in the ***riURL*** attribute of the **roap:ROPayload**. An HTTP GET on this URL SHOULD return either a *JoinDomain* ROAP Trigger or a (X)HTML page that starts an interaction with the User which may eventually lead to a *JoinDomain* ROAP Trigger. It should be noted that in the event that a *JoinDomain* ROAP Trigger is returned and the Device does not have a valid RI context then the Device MUST automatically register with the RI (as specified in section 5.2.1 of [DRM-v2.1]) prior to sending a *JoinDomainRequest* message. The Device may also use RI context information contained in the content format if it is unable to obtain RI context and it is left to Trust Authority to determine how the RI context information is able to be used before Device must contact RI and obtain RI context.

The Device MUST verify the signature of the Domain RO using the RI's Public Key. If the verification fails the Device SHALL NOT install the Domain RO. In this case the Device MAY request a new Rights Object by sending a HTTP GET to the RightsIssuerURL in the relevant DCF.

After the Device verifies the signature of the Domain RO, it MUST compare the **<domainID>** field within the Domain RO with the Domain identifiers for any valid Domain Contexts already established with the RI that issued the Domain RO, as identified by the **<riID>** field. There are three possible outcomes of this comparison:

1. The **<domainID>** field matches a Domain identifier in a valid Domain Context already established with the RI. The Device MAY install the Domain RO.

2. The Domain baseID of the **<domainID>** field matches the Domain baseID of a stored Domain identifier in a valid Domain Context already established with the RI, but the Domain Generation of the RO is greater than the Generation of the stored domain ID. The device MAY attempt to upgrade the Domain by sending a ROAP-JoinDomainRequest to the riURL in the RI Context associated with the Domain Context. The Device may have to obtain user consent to contact the RI, section 5.1.8 defines when explicit user consent is required.

   If the Domain upgrade is successful, the Device MAY install the Domain RO. Otherwise the Device SHALL NOT install the Domain RO.

3. The Domain baseID of the **<domainID>** field does not match a Domain baseID in any valid Domain Context already established with the RI. The Device MAY attempt to join the Domain by sending an HTTP GET request to the URL specified in the ***riURL*** attribute of the **roap:ROPayload**. The Device may have to acquire the user's

consent prior to sending the HTTP GET request (section 5.1.8 of [DRM-v2.1] defines when explicit user consent is required).

In RICD case the Device MUST compare <domainID> field within the Domain RO with the Domain identifiers for any valid Domain Contexts already established with the RICD to which the RI that issued the Domain RO belongs to. The Device can recognize RIs that belong to same RICD by using key purpose values that are defined by Trust Authority.

Comparison in RICD case:

1. The **<domainID>** field matches a Domain identifier in a valid Domain Context already established with the RICD. The Device MAY install the Domain RO.

2. The Domain baseID of the **<domainID>** field matches the Domain baseID of a stored Domain identifier in a valid Domain Context already established with the RICD, but the Domain Generation of the RO is greater than the Generation of the stored domain ID. The device MAY attempt to upgrade the Domain by sending a ROAP-JoinDomainRequest to the riURL in the RI Context associated with the Domain Context. The Device may have to obtain user consent to contact the RI, section 5.1.8 defines when explicit user consent is required.

   If the Domain upgrade is successful, the Device MAY install the Domain RO. Otherwise the Device SHALL NOT install the Domain RO.

The Domain baseID of the **<domainID>** field does not match a Domain baseID in any valid Domain Context already established with the RICD. The Device MAY attempt to join the Domain by sending an HTTP GET request to the URL specified in the ***riURL*** attribute of the **roap:ROPayload**. The Device may have to acquire the user's consent prior to sending the HTTP GET request (section 5.1.8 of [DRM-v2.1] defines when explicit user consent is required).

At the point where the Device sends an HTTP GET request to the URL specified in the ***riURL*** attribute of the **roap:ROPayload** the RO installation process as specified within this section is effectively aborted, however, the installation process may be restarted as a result of subsequent user interaction, by some other Device specific means that is outside the scope of this specification or as a direct result of responding to a subsequent ROAP Trigger. As a result of an HTTP GET to this URL the RI can choose (using its own criteria) whether to allow the Device to join the Domain or not and SHOULD return either a *JoinDomain* ROAP Trigger or a (X)HTML page that starts an interaction with the User which may eventually lead to a *JoinDomain* ROAP Trigger. In the event that the RI chooses not to allow the Device to join the Domain the RI MAY offer the user the opportunity to acquire a Device RO.

Before installing a Domain RO, the Device MUST successfully verify the MAC (using the **<mac>** element of the **roap:ProtectedRO**). If this verification fails, the Device SHALL NOT install the Domain RO. In this case the Device MAY initiate the process of acquiring a new Rights Object by sending a HTTP GET request to the RightsIssuerURL in the relevant DCF

If the Domain RO is stateful**,** then the Device MUST perform the replay protection related checks defined in section 9.4 of [DRM-v2.1].

If the Domain Context has expired (indicated by the Domain Context Expiry Time) the Device MUST NOT install ROs for this Domain.

In the case where the Domain RO is received within a DCF, if the Device cannot verify the signature of the Domain RO, the Device MAY leave the Domain RO as is within the DCF. The Device MAY request a valid RO for the DCF by sending an HTTP GET request to the RightsIssuerURL in the DCF.

5.1.7.2.2 Postprocessing after installing the Domain RO

There are cases where a Device installs a Domain RO that it received separately from the DCF to which it refers. In these cases, the Device SHOULD insert a copy of the Domain RO into the corresponding DCF (see [DRMCF-v2.1]) as soon as possible after installation.

The Device MAY insert the Domain RO into the DCF at a later stage, for example when the user requests to render the DCF or send it out of the Device. The Device MAY insert more than one Domain RO into a single DCF, as long as all of the inserted RO's are valid and correspond to a Domain that it is a member of.

When the Device inserts a Domain RO into a DCF, it SHOULD remove from the DCF all Domain RO's corresponding to Domains that the Device is not a member of.

The Device SHOULD NOT insert a copy of the Domain RO into the corresponding DCF if it concludes, using an algorithm not defined in this specification, that sending the installed Domain RO to other Devices does not add value for the end user, for example if the Domain RO has expired.

If the Device finds multiple DCF instances bound to the installed Domain RO, it SHOULD insert a copy of the Domain RO into each one of them.

5.1.8 Domain Upgrade

A Rights Issuer may *upgrade* a Domain if, for example, a Domain Key has been compromised or if a Device in the Domain has been revoked. This will probably be a rare event, but may be necessary as a last resort to stop DRM Content from leaking out of the system in the clear.

In order to upgrade a Domain, a RI MUST change the Domain Key and MUST increment the Domain Generation by one. If the Domain Generation value reaches 999 the Domain becomes obsolete. An RI MUST NOT issue ROs for an obsolete Domain and MUST NOT allow new Devices to join an obsolete Domain.

A Domain upgrade does not result in any Domain Context being deleted in any Device. After an upgrade, Domain ROs issued before the upgrade may still be used and shared. This applies to all Devices (revoked and unrevoked) previously in the Domain, and to any new Devices added to the Domain after the upgrade.

A Rights Issuer performs a Domain upgrade using the Join Domain protocol (see section 5.1.2). An RI MAY initiate this protocol for the purposes of Domain upgrade by sending a ROAP trigger to a Device whose Domain membership it wishes to upgrade. If a Device receives a Join Domain ROAP trigger, it SHOULD compare the **<domainID>** field with the domain ID for any domains already established with the RI that sent the ROAP trigger, with the sending RI as identified by the **<riID>** field. There are two possible outcomes of this comparison:

1. The Domain baseID of the **<domainID>** field matches Domain baseID of a stored domain ID, but the value of the Domain Generation in the trigger is greater than the value stored by the Device. The incoming trigger represents a Domain upgrade, as described in this section. The Device SHOULD in this case silently upgrade the Domain using the Join Domain protocol.

2. If the Domain baseID of the **<domainID>** field does not match Domain baseID of a stored domain ID, then the Device is not a member of the Domain. The Device MUST execute the Join Domain protocol (see 5.4.4 of [DRM-v2.0]) just as if it was joining the domain for the first time (see section 5.1.2).

RICD RIs that provide and manage the Domain Context may *upgrade* a Domain. One must ensure that all such RICD RIs receive the Domain upgrade information. The mechanism for this is not specified in this specification.

5.1.8.1 Use of hash chains for Domain key management

To avoid storage of multiple keys per Domain in the Device and in the RI (for the purpose of using old and new Domain ROs after Domain upgrade) it is possible to have a relation between the Domain Keys using Hash Chains (see section 7.3 of [DRM-v2.1]), as illustrated in the example below. The Device MAY support Hash Chains and the RI MAY support Hash Chains.

Example1. Without hash chains

When generating a new Domain, the RI generates:

- a unique Domain Identifier DI, the Domain Generation is set to 000.
- a random secret Domain Key DK_0

At Domain upgrade the Domain Generation *g* is increased by 1, which is reflected in the Domain Identifier, and a new Domain Key DK_g is generated. The old Domain Key(s) must be stored in RI and Device to allow use of ROs issued before the upgrade. When Devices join a Domain, all Domain Keys of this Domain are sent in the Protected Domain Info of ROAP-JoinDomainResponse (see section 5.4.4.2 of [DRM-v2.1]).

Example 2. With Hash Chains (optional)

When generating a new Domain, the RI

- generates a unique Domain Identifier DI, the Domain Generation is set to 000
- generates an initial master key K_M for the Domain
- selects the maximum number of generations *n* for this Domain (not larger than 999)
- defines a sequence of Domain Keys using the method described in section 7.3 of [DRM-v2.1].

Since old Domain Keys (with low generation value) are possible to efficiently derive from new Domain Keys (with higher generation value), it is only necessary to store the newest Domain Key in the Device (and corresponding Domain Identifier so the Domain Generation is known). For the RI it is sufficient to store DK_n ($=K_M$) and the current Domain Identifier.

5.2 RI Context Information

RI Context Information is included inside a DCF when a Domain RO is also inside a DCF. This permits Devices to consume the Content when they do not have RI context and are unable to connect to the relevant RI. The use of this functionality must be prescribed by the Trust Authority.

At a minimum the RI Context Information consists of roap:CertificateChain which corresponds to the information needed to process the associated RO.

6. Modification to OMA DRM REL TS

Modifications to OMA-TS-DRM_REL-V2_1-2008014-A

6.1.1 Element <asset>

Element	<!ELEMENT o-ex:asset (o-ex:context?, o-ex:inherit?, o-ex:digest?, ds:KeyInfo?)>
Semantics	The <asset> element specifies the identity of the DRM Content governed by the containing <agreement> element via the <context> child element. The optional <inherit> element instructs the DRM Agent to apply the rights from the inherited Rights Object, specified in the <inherit> element context, to this asset. Note that the <KeyInfo> element SHOULD be omitted if the Rights Object functions as a parent Rights Object in the inheritance case. The optional <digest> element provides integrity protection for the reference to the DRM Content. The optional <KeyInfo> element provides the functionality to access the DRM content if granted the rights to do so. The <asset> element enables expression linking via its “id” and “idref” attributes. This enables reuse of Permissions defined for one asset, for other assets inside the same Rights Object. When the <asset> element is contained in a <permission> element, it MUST contain an “idref” attribute, and MUST be empty, i.e., all its optional child elements MUST be omitted. Note that one or more assets are permitted and can be used to transport additional Content Encryption Keys as needed. For example, for AV content there may be one CEK for all Audio tracks and one CEK for all Video tracks.

7. Format Clarifications

For avoidance of doubt, nothing precludes OMA DRM being used with other content formats. If no other format is specified DCF should be used. Note, the use of other formats is not specified by OMA specification.

8. Modifications to OMA DRM DCF TS

8.1 RI Context Information Box

The RI context information box MAY be used to insert a RI context information, into a DCF or PDCF. A `MutableDRMInformation` box MAY include zero or more RI Context Information boxes. The RI Context Information is treated as binary data and a Device MAY add or delete RI Context Information boxes in the `MutableDRMInformation` box. Note there must be an Associated Right Object Box.

```
aligned(8) class OMADRMRiContextInfo extends FullBox('rici', 0, 0) {
      byte    Data[];              // binary Rights Object
}
```

Table 1: OMA DRM RI Context Information box fields

Field name	Type	Purpose
Data	byte[]	A RI Context Information as binary data

9. XML Examples

9.1 Domain ROs with Multiple CEKs

```
<roap:protectedRO
 xmlns:roap="urn:oma:bac:dldrm:roap-1.0"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:o-ex="http://odrl.net/1.1/ODRL-EX"
 xmlns:o-dd="http://odrl.net/1.1/ODRL-DD"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <roap:ro        id="n8yu98hy0e2109eu09ewf09u"
domainRO="true"
version="1.1"
 riURL="http://www.RI1.com">
  <riID>
    <keyIdentifier xsi:type="roap:X509SPKIHash">
     <hash>aXENc+Um/9/NvmYKiHDLaErK0fk=</hash>
    </keyIdentifier>
  </riID>
  <rights o-ex:id="REL1">
    <o-ex:context>
     <o-dd:version>2.1</o-dd:version>
     <o-dd:uid>n8yu98hy0e2109eu09ewf09u</o-dd:uid><!-- same as roID -->
    </o-ex:context>
    <o-ex:agreement>
     <o-ex:asset o-ex:id =”Video Key”>
       <o-ex:context>
        <o-dd:uid>ContentID</o-dd:uid>
       </o-ex:context>
       <ds:KeyInfo>
        <xenc:EncryptedKey>
         <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes128"/>
          <ds:KeyInfo>
            <ds:RetrievalMethod URI="#K_MAC_and_K_REK"/>
          </ds:KeyInfo>
          <xenc:CipherData>
           <xenc:CipherValue>EncryptedCEK</xenc:CipherValue>
          </xenc:CipherData>
        </xenc:EncryptedKey>
       </ds:KeyInfo>
     </o-ex:asset>
    <o-ex:asset o-ex:id =”Audio Key”>
       <o-ex:context>
        <o-dd:uid>ContentID</o-dd:uid>
       </o-ex:context>
       <o-ex:digest>
        <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <ds:DigestValue>bLLLc+Um/5/NvmYKiHDLaErK0fk=</ds:DigestValue>
       </o-ex:digest>
       <ds:KeyInfo>
        <xenc:EncryptedKey>
         <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes128"/>
```

```
          <ds:KeyInfo>
            <ds:RetrievalMethod URI="#K_MAC_and_K_REK"/>
          </ds:KeyInfo>
          <xenc:CipherData>
           <xenc:CipherValue>EncryptedCEK</xenc:CipherValue>
          </xenc:CipherData>
         </xenc:EncryptedKey>
        </ds:KeyInfo>
      </o-ex:asset>
      <o-ex:permission>
       <o-dd:play/>
<!-- outputs -->
       <o-dd:export oma-dd:mode="copy" oma-dd:transcribe=”false”>
                  <o-ex:constraint>
                   <oma-dd:system>
                    <o-ex:context>
                     <o-dd:uid>urn:oma:drms:org-XXXX:analog-outputs</o-dd:uid>
                    </o-ex:context>
                    <o-ex:context>
                     <o-dd:uid>urn:oma:drms:org-XXXX:dtcp-ip</o-dd:uid>
                    </o-ex:context>
                    <o-ex:context>
                     <o-dd:uid>urn:oma:drms:org-XXXX:hdcp</o-dd:uid>
                    </o-ex:context>
                   </oma-dd:system>
                   </o-ex:constraint>
       </oma-dd:export>
      </o-ex:permission>
    </o-ex:agreement>
   </rights>
   <signature>
    <ds:SignedInfo>
     <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
           <ds:SignatureMethod
      Algorithm="http://www. rsasecurity.com/rsalabs/pkcs/schemas/pkcs-1#rsa-pss-
default"/>
      <ds:Reference URI="#REL1">
       <ds:Transforms>
        <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
       </ds:Transforms>
       <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
       <ds:DigestValue>sIo5hb+id8JtuOMNKs12=drf5+3df=</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:SignatureValue>
    <ds:KeyInfo>
     <roap:X509SPKIHash>
      <hash>aXENc+Um/9/NvmYKiHDLaErK0fk=</hash>
     </roap:X509SPKIHash>
    </ds:KeyInfo>
   </signature>
   <encKey Id="K_MAC_and_K_REK">
    <xenc:EncryptionMethod
     Algorithm="http://www.w3.org/2001/04/xmlenc#kw-aes128"/>
    <ds:KeyInfo>
     <roap:domainID>Domain-XYZ-001</roap:domainID>
    </ds:KeyInfo>
    <xenc:CipherData>
     <xenc:CipherValue>32fdsorew9ufdsoi09ufdskrew9urew0uderty5346wq</xenc:CipherValue>
    </xenc:CipherData>
```

```
   </encKey>
  </roap:ro>

<mac>
   <ds:SignedInfo>
    <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    <ds:SignatureMethod
     Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"/>
    <ds:Reference URI="#n8yu98hy0e2109eu09ewf09u">
     <ds:Transforms>
      <ds:Transform Algorithm=http://www.w3.org/2001/10/xml-exc-c14n#/>
     </ds:Transforms>
     <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
     <ds:DigestValue>sIo5hb+id8JtuOMNKs12=drf5+3df=</ds:DigestValue>
    </ds:Reference>
   </ds:SignedInfo>
   <ds:SignatureValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:SignatureValue>
   <ds:KeyInfo>
    <ds:RetrievalMethod URI="#K_MAC_and_K_REK"/>
   </ds:KeyInfo>
  </mac>
</roap:protectedRO>
```

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version –or- No previous version within OMA

A.2 Draft/Candidate Version <current version> History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-RICD_Profile-V1_0	07 Jul 2011	All	Baseline
	19 Jul 2011		OMA-DRM-2011-0073R01-INP_DRM_RICD_TS
	11 Aug 2011	All	Updated according to agreed CR: OMA-DRM-2011-0086R03-CR_RICD_TS_additions_1
	31 Aug 2011	All	Updated according to agreed CR: OMA-DRM-2011-0081R01-CR_RICD_DM_and_D_RO
	02 Sep 2011	A.2	History box update
	11 Oct 2011	All	Updated according to: OMA-CONRR-RICD-V1_0-20111006-D
	18 Oct 2011	5.1.7.2.1, 5.1.8	Updated according to: OMA-CONRR-RICD-V1_0-20111013-D
	11 Nov 2011	All Cover Page A.2.	Updated according to agreed CR: OMA-DRM-2011-0105R01-CR_RICD_TS_commentfixes Corrected the title and updated document version date History box update
Candidate Version: OMA-TS-RICD_Profile-V1_0-20111122-C	22 Nov 2011	All	Status changed to Candidate by TP: OMA-TP-2011-0405-INP_RICD_V1_0_RRP_for_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

B.1 SCR for XYZ Client

Item	Function	Reference	Requirement

B.2 SCR for XYZ Server

Item	Function	Reference	Requirement