# SCE User Domains

Candidate Version 1.0 – 26 May 2009

**Open Mobile Alliance**

OMA-TS-SCE_DOM-V1_0-20090526-C

# Contents

# Figures

# Tables

# 1. Scope

Open Mobile Alliance (OMA) specifications are the result of continuous work to define industry-wide interoperable mechanisms for developing applications and services that are deployed over wireless communication networks.

The scope of OMA "Digital Rights Management" (DRM) is to enable the distribution and consumption of digital content in a controlled manner. The content is distributed and consumed on authenticated Devices per the usage rights expressed by the content owners. OMA DRM work addresses the various technical aspects of this system by providing appropriate specifications for content formats, protocols, and a rights expression language.

A number of DRM specifications have already been defined within the OMA. The latest accepted release of the OMA DRM enabler ([DRM-v2.1], including [DRM-DRM-v2.1], [DRM-DCF-v2.1], [DRM-REL-v2.1]), is referred to within this document as "OMA DRM v2.1".

This specification defines the mechanisms and protocols necessary to implement a central domain management function, as required per [SCE-RD]. More specifically, this specification will specify the interfaces SCE-2-DMP and SCE-3-RDP as defined in [SCE-AD].

# 2. References

## 2.1 Normative References


**[DRM-v2.1]** The OMA DRM 2.0 enabler as described in "Enabler Release Definition for DRM V2.0, Approved Version 2.0", OMA-TS-DRM-DRM-V2_0-20060303-A, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[DRM-DRM-v2.1]** "DRM Specification, Approved Version 2.0", OMA-TS-DRM-DRM-V2_0-20060303-A, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[DRM-REL-v2.1]** "DRM Rights Expression Language, Approved Version 2.0", OMA-TS-DRM-REL-V2_0-20060303-A, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[DRM-DCF-v2.1]** "DRM Content Format, Approved Version 2.0", OMA-TS-DRM-DCF-V2_0-20060303-A, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-RD]** "Secure Content Exchange Requirements, Draft Version 1.0", OMA-RD-SCE-V1_0-20060908-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-AD]** "Secure Content Exchange Architecture, Draft Version", OMA-AD-SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-DRM]** "DRM Specification – SCE Extensions, Draft Version", OMA-TS-DRM-DRM-SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-REL]** "DRM Rights Expression Language – SCE Extensions, Draft Version", OMA-TS-DRM-REL-SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-LRM]** "DRM Local Rights Management, Draft Version", OMA-TS-DRM-LRM- SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-DOM]** "DRM User Domains, Draft Version", OMA-TS-DRM-DOM-SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-A2A]** "DRM Agent-to-Agent transfer, Draft Version", OMA-TS-DRM-REL- SCE-Vx_y-D, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

**[SCE-GEN]** "SCE Generic Mechanisms", OMA-TS-SCE_GEN-Vx-y-D, Open Mobile Alliance™ URL:http://www.openmobilealliance.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

| | |
|---|---|
| **Constraint** | A restriction on the Permission over DRM Content (DRM V2.0). |
| **Consume** | To Play, Display, Print or Execute DRM Content on a Device or to render DRM Content on a Render Client. |
| **Content** | One or more Media Objects (DRM V2.0). |
| **Copy** | To make Rights existing on a source Device available for use by a recipient Device, without affecting availability on the source Device. Rights may be restricted on the recipient Device. Note: this is different from the V2.0 definition. |
| **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. The Device may include a smart card module (e.g. a SIM) (DRM V2.0). |
| **Device Rights Object** | A Rights Object that is initially targeted to a specific entity. Subsequently, the Rights Object may be allowed to be targeted to other entities to be consumed, serially or in parallel, independently of membership in a Domain or User Domain. |
| **Domain** | A set of v2.x and/or SCE DRM Agents that can consume Domain Rights Objects. |
| **Domain Authority** | The entity to specify the Domain Policy for a User Domain or an Ad Hoc Domain. |
| **Domain Enforcement Agent** | The entity to enforce the Domain Policy on behalf of the Domain Authority. It may reside in the network as a service or in a User’s device. |
| **Domain Policy** | A collection of attributes which defines the policy determining characteristics of the membership of a User Domain or Ad Hoc Domain, as set by the Domain Authority that the Domain Enforcement Agent will enforce. |
| **Domain Rights Object** | A Rights Object that is targeted to a specific v2.x Domain. The Rights Object can be consumed independently by each v2.x or SCE DRM Agent that is a member of the Domain. |
| **DRM Agent** | The entity in the Device that manages Permissions for Media Objects on the Device (DRM V2.1). In this document, the DRM Agent implements some or all the functionality defined in this specification. |
| **DRM Content** | Media Objects that are consumed according to a set of Permissions in a Rights Object. |
| **DRM Time** | A secure, non-user changeable time source. The DRM Time is measured in the UTC time scale. |
| **Import** | To convert Import-Ready Data into OMA (P)DCFs and ROs. |
| **Import-Ready Data** | Content and associated Rights derived from Non-OMA DRM-sourced data that can be converted into OMA (P)DCFs and ROs. |
| **Imported-Rights-Object** | An OMA RO resulting from converting Import-Ready Data. |
| **Local Rights Manager (LRM)** | An entity that is responsible for aspect(s) of Import and it may also manage an Imported-Content for a limited group of OMA DRM Agents. |
| **Media Object** | A digital work e.g. a ringing tone, a screen saver, a Java game or a Composite Object. |

| | |
|---|---|
| **Non-OMA DRM** | A protection system other than OMA DRM, which may include copy protection mechanisms for storage medium and/or transport mechanisms. |
| **Move** | To make Rights existing initially on a source Device fully or partially available for use by a recipient Device, such that the Rights or parts thereof that become usable on the recipient Device can no longer be used on the source Device. |
| **Partial Rights** | A subset of a set of Rights, such that the Partial Rights are equally or more restrictive than those in the set. |
| **Permission** | Actual usage or activities allowed (by a Rights Issuer or Local Rights Manager) over DRM Content. |
| **Rights** | The collection of permissions and constraints defining under which circumstances access is granted to DRM Content. |
| **Rights Issuer** | An entity that issues Rights Objects to OMA DRM conformant Devices (DRM V2.1). |
| **Rights Object** | A collection of Permissions and other attributes which are linked to DRM Content. |
| **State Information** | A set of values representing current state associated with Rights. It is managed by the DRM Agent only when the Rights contain any of the stateful constraints (e.g. interval, count, timed-count, accumulated, etc.). |
| **User** | The human user of a Device. The User does not necessarily own the Device (DRM V2.0). |
| **User Domain** | A set of v2.x and/or SCE DRM Agents that can consume User Domain Rights Objects. |
| **User Domain Authorization** | A digitally signed data object that provides proof of authorization related to a User Domain. |
| **User Domain Generation** | A Counter reflecting the number of times the User Domain has been upgraded. The User Domain Generation is a part of the User Domain Identifier (the last three digits). |
| **User Domain Identifier** | A unique string identifier of the User Domain Key. |
| **User Domain Key (UDK)** | A 128 bit symmetric encryption key that is used by a member of the User Domain. A User Domain Rights Object is encrypted with the User Domain Key. |
| **User Domain Rights Object** | A Rights Object that is targeted to a specific User Domain. Besides requiring membership in the User Domain, consumption may require being targeted to an SCE DRM Agent. |
| **v2.x DRM Agent** | A DRM Agent that is conformant to either [DRM-DRM-v2.0] or [DRM-DRM-v2.1]. |

## 3.3 Abbreviations

| | |
|---|---|
| **A2A** | Agent to Agent |
| **AES** | Advanced Encryption Standard |
| **CEK** | Content Encryption Key |
| **DA** | Domain Authority |
| **DCF** | DRM Content Format |
| **DEA** | Domain Enforcement Agent |
| **DRM** | Digital Rights Management |
| **HTTP** | HyperText Transfer Protocol |
| **LRM** | Local Rights Manager |
| **MAC** | Message Authentication Code |
| **MIME** | Multipurpose Internet Mail Extensions |
| **MK** | Message Integrity Key |
| **N/A** | Not Applicable |

| | |
|---|---|
| **OCSP** | Online Certificate Status Protocol |
| **OMA** | Open Mobile Alliance |
| **REL** | Rights Expression Language |
| **REK** | Rights Object Encryption Key |
| **RI** | Rights Issuer |
| **RI/LRM** | RI or LRM |
| **RO** | Rights Object |
| **ROAP** | Rights Object Acquisition Protocol |
| **RSA** | Rivest-Shamir-Adelman public key algorithm |
| **SA** | Security Association |
| **SCE** | Secure Content Exchange |
| **SCR** | Static Conformance Requirement |
| **SHA-1** | Secure Hash Algorithm |
| **SK** | Session Key |
| **UDI** | User Domain Identifier |
| **UDK** | User Domain Key |
| **URI** | Uniform Resource Identifier |
| **URL** | Uniform Resource Locator |
| **XML** | Extensible Markup Language |

# 4. Introduction

A User Domain is a set of v2.x and/or SCE DRM Agents that can share ROs created for the User Domain. DRM Agents can join multiple User Domains managed by one or more DEAs.

The DEA defines the User Domain, manages the key material, and controls which and how many DRM Agents are included and excluded from the User Domain. The DEA also controls which entities (RIs and/or LRMs) are authorized to create User Domain ROs.

## 4.1 User Domain Authorization

Before a DEA, RI or LRM can participate in any User Domain functionality, it must be authorized to do so. Before a DRM Agent can receive a User Domain RO with the <userDomain> constraint ([SCE-REL]) from an RI, it must prove that it is a member of the User Domain. The mechanism to do so is via a "User Domain Authorization" data structure. The User Domain Authorization is used to provide proof of authorization related to a User Domain.

The following table summarizes the types of User Domain Authorizations.

| Entity | Signer | |
|---|---|---|
| | **DA** | **DEA** |
| **DEA** | DEA is authorized to manage a User Domain | N/A |
| **RI/LRM** | RI/LRM is associated with a DEA | RI/LRM can create (or import) ROs for a User Domain |
| **Device** | Device is associated with a DEA | Device is a member of a User Domain |

**Table 1 - User Domain Authorization Types**

The following XML schema fragment defines a User Domain Authorization:

```
<complexType name="UserDomainAuthorizationType">
    <sequence>
        <element name="body" type="dom:UserDomainAuthorizationBody"/>
        <element name="signature" type="base64Binary"/>
        <element name="certChain" type="gen:CertificateChain" minOccurs="0"/>
    </sequence>
</complexType>

<complexType name="UserDomainAuthorizationBody">
    <sequence>
        <element name="dadeaID" type="gen:Identifier"/>
        <element name="dadeaURL" type="gen:Identifier" minOccurs="0"/>
        <element name="deaID" type="gen:Identifier" minOccurs="0"/>
        <element name="userDomainID" type="dom:UserDomainIdentifier" minOccurs="0"/>
        <element name="entityID" type="gen:Identifier"/>
        <element name="notBefore" type="gen:dateTimeOrInfinite" minOccurs="0"/>
        <element name="notAfter" type="gen:dateTimeOrInfinite" minOccurs="0"/>
        <element name="policyInfo" type="dom:PolicyInfo"/>
        <any minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
    </sequence>
    <attribute name="isDea" type="boolean" use="optional" default="false"/>
    <attribute name="isRIorLRM" type="boolean" use="optional" default="false"/>
    <attribute name="isDRMAgent" type="boolean" use="optional" default="false"/>
</complexType>

<complexType name="PolicyInfo">
    <sequence>
        <element name="policyID" type="gen:String64"/>
```

```
        <any minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
    </sequence>
    <attribute name="daRiAuthorizationRequired" type="boolean" use="optional" default="false"/>
    <attribute name="daDeviceAuthorizationRequired" type="boolean" use="optional" default="false"/>
    <attribute name="allowProxyJoin" type="boolean" use="optional" default="false"/>
</complexType>

<simpleType name="UserDomainIdentifier">
    <restriction base="string">
        <pattern value=".{1,24}\d{3}"/>
    </restriction>
</simpleType>
```

*body*: This element contains information about a User Domain Authorization. Its child elements are described below.

*signature*: This element contains the digital signature over the <body> element.

*certChain*: This element, if present, contains the certificate chain of entity that signed the User Domain Authorization, i.e. either a DA or a DEA. It has a schema of CertificateChain as described in [SCE-GEN].

*dadeaID*: This element contains the ID of the entity by which this User Domain Authorization is provided. It has a schema of Identifier as described in [SCE-GEN]. The valid entity types are DA or DEA. The User Domain Authorization is said to be signed by this entity.

*dadeaURL*: This element contains a URL from which a DRMTrigger or XHTML document can be acquired that enables a DRM Agent to receive a new User Domain Authorization. Typically this may be used in case a User Domain Authorization has become invalid.

*userDomainID*: This element identifies for which User Domain the authorization is valid. Some authorizations apply to any User Domain, in which case this element is not present. In the case where the Signer is a DA and the Entity is a DEA (in **Error! Reference source not found.**), this element is present and the User Domain Generation counter within this element is set to 000 by the DA.*deaID*: This element identifies for which DEA the authorization is valid. Some authorizations do not apply to a specific DEA, in which case this element is not present.

*Isdea, isRIorLRM, isDRMAgent*: These elements signal the role that the entity identified by entityID is authorized to fulfil. More than one can be present.

*entityID*: This element contains the ID of the entity for which this User Domain Authorization is being provided. It has a schema of Identifier as described in [SCE-GEN]. The valid entity types are DEA, RI, LRM or DRM Agent. The User Domain Authorization is said to be signed for the entity or said to associate the entity with the DEA.

*notBefore*: This element specifies the date/time before which the User Domain Authorization is not valid. It has a schema of dateTime as described in [SCE-GEN].

*notAfter*: This element specifies the date/time after which the User Domain Authorization is not valid. It has a schema of dateTime as described in [SCE-GEN].

*policyInfo*: This element contains information about domain policy of the User Domain. Its sub-elements are described below.*policyId*: This element is used to identify the Domain Policy associated with the particular User Domain. This field MUST be present when a DA signs the authorization for an RI/LRM.

*daRiAuthorizationRequired*: This element, if present, indicates that an RI/LRM MUST get from the DEA (if the RI/LRM does not already have one) a User Domain Authorization signed by the DA associating the RI or LRM with the DEA. This element MAY be present in a User Domain Authorization signed by a DA for a DEA and MUST NOT be present in other User Domain Authorization types.

*daDeviceAuthorizationRequired*: This element, if present, indicates that a Device MUST get from the DEA (if the Device does not already have one) a User Domain Authorization signed by the DA associating the Device with the DEA. This element MAY be present in a User Domain Authorization signed by a DA for a DEA and MUST NOT be present in other User Domain Authorization types.

*allowProxyJoin*: This element, if present, indicates that this User Domain allows DRM v2.x Devices to become members via the Proxy Join User Domain Protocol (see section 7.4). If this element is not present, then the DEA MUST NOT allow Proxy Join User Domain protocol for the particular User Domain. This element MAY be present in a User Domain Authorization signed by a DA for a DEA or in a User Domain Authorization signed by a DEA for an RI/LRM and MUST NOT be present in other User Domain Authorization types.

Elements of type *UserDomainIdentifier* contain the ID of the User Domain. The last three characters (digits) represent the User Domain Generation. The other, preceding characters represent the User Domain baseID. The value of the User Domain Generation counter indicates the number of upgrades performed on the User Domain.

## 4.2 Acquiring User Domain ROs

A DRM Agent can acquire User Domain ROs from an RI, from an LRM or from another DRM Agent. The SCE enabler defines the following mechanisms to transport User Domain ROs:

- The SCE-1-ROAP protocol ([SCE-DRM]). This mechanism may be used to deliver a User Domain RO from an RI to a DRM Agent. In the case of an LRM, the SCE-6-LRMP protocol will re-use the RO Acquistion protocol from SCE-1-ROAP.
- The SCE-7-A2AP protocol ([SCE-A2A]). This mechanism may be used to transport a User Domain RO from one DRM Agent to another.
- The SCE-8 interface (“out-of-band”, section 9.1). This data specification allows the distribution of User Domain ROs via other protocols or mechanisms not defined in the SCE Enabler. For example, a User Domain RO can be delivered inside a (P)DCF file, as a separate standalone MIME object, or as part of a MIME multipart/related message [RFC2387]. This mechanism may be used to transport User Domain ROs to a DRM Agent from any source. However this mechanism MUST NOT be used to transport User Domain ROs that have the <userDomain> constraint ([SCE-REL]).

The DRM Agent MUST support receiving a User Domain RO in a ROAP-ROResponse message.

The DRM Agent MUST support receiving a User Domain RO as a separate object.

As a general principle, the processing rules for inbound User Domain ROs are agnostic to the origin of the User Domain RO or the mechanism by which it is transported, i.e. it does not matter whether the User Domain RO was delivered OTA from an RI using ROAP or copied from another DRM Agent using SCE-8. There is no binding to a specific transport mechanism or transport protocol.

The process of checking the validity of inbound User Domain ROs and storing them is called installation of the User Domain RO. After the RO has been installed, a User may request the DRM Agent to grant any of the permissions related to a specific Content. This process is called consumption of the User Domain RO. To render the media objects inside the associated DCF the DRM Agent MUST process the User Domain RO as defined in section 8.

# 5. User Domain Management

This section describes the processes used by a DEA to manage a User Domain.

## 5.1 Devices

### 5.1.1 Registration with a DEA

Prior to being able to install and consume a User Domain RO, a DRM Agent needs to register with the DEA that manages the User Domain that the RO refers to and subsequently become a member of that User Domain. As a result of the successful execution of the Registration protocol (section 6.1) with a DEA, a DRM Agent will establish a logical DEA Context for the given DEA. At a minimum, the DEA Context consists of the following:

- The identity of the DEA
- An indication of verification of the DEAs certificate chain
- The negotiated protocols
- A Session Key - used to encrypt data between the Device and the DEA
- A MAC Key - used to provide integrity of certain data exchanged between the Device and the DEA

### 5.1.2 Joining a User Domain

After registering with a DEA, a DRM Agent MUST become a member of the User Domain by joining the User Domain. As a result of the the successful execution of the Join User Domain protocol (section 6.2) with a DEA, the DRM Agent will establish a logical User Domain Context for the given User Domain. At a minimum, the User Domain Context MUST contain at least:

- Identity of the User Domain (which includes the User Domain Generation)
- Expiry time of the User Domain Context.
- An indication of the verification of the User Domain Authorization for the DEA (signed by a DA). The User Domain Generation part of the User Domain Identifier is ignored by the DRM Agent.
- Expiry time of the User Domain Context (which MUST be the same or earlier than the <notAfter> element in the DEAs User Domain Authorization)
- The User Domain Key (UDK)
- The DRM Agents User Domain Authorization (signed by the DEA)

Additionally, the User Domain Context MAY contain:

- The alias of the User Domain – used in communication with the User to refer to the User Domain. The User Domain Alias is only present in the User Domain Context if it was included in the dmpJoinDomainTrigger.

A DRM Agent MAY join multiple User Domains managed by one or more DEAs.

Even though a DRM Agent is a member of a User Domain, it MAY have to renew its membership for one of the following reasons:

- The DEA has upgraded the UDK of the User Domain.
- The DRM Agents User Domain Authorization has expired. If the User Domain Authorization for a DRM Agent has expired, the DEA MUST treat the DRM Agent as if the DRM Agent has left the User Domain.

### 5.1.3 Leaving a User Domain

At the behest of the User, a DRM Agent MAY leave a User Domain at any time. As a result of the succesful execution of the Leave User Domain protocol (section 6.3) with a DEA, the DRM Agent will no longer have a User Domain Context for the given User Domain. Without the User Domain Context, ROs issued for that User Domain MUST NOT be consumed. After leaving a User Domain a DRM Agent MAY, but is not required to, remove the corresponding User Domain ROs and associated DRM Content. The DRM Agent SHOULD obtain User confirmation before deleting User Domain ROs and associated DRM Content.

Prior to sending a Leave User Domain Request, the DRM Agent MUST disable the corresponding User Domain Context. After receiving the Leave User Domain Response with 'Success' as the status, the DRM Agent MUST delete the corresponding User Domain Context.

## 5.2 RIs and LRMs

### 5.2.1 Registration with a DEA

Prior to being able to create User Domain ROs, an RI/LRM needs to register with the DEA that manages the User Domain that the RO refers to and subsequently get a User Domain Authorization for that User Domain. As a result of the successful execution of the RI-DEA Registration Protocol (section 7.1) of an RI/LRM with a DEA, the RI/LRM will establish a logical DEA Context for the given DEA. At a minimum, the DEA Context consists of the following:

- The identity of the DEA
- An indication of verification of the DEAs certificate chain
- The negotiated protocols
- A Session Key – used to encrypt data between the RI/LRM and the DEA
- A MAC Key – used to provide integrity of certain data exchanged between the RI/LRM and the DEA

### 5.2.2 Getting Authorization for a User Domain

After registering with a DEA, an RI/LRM MUST get the authorization to create User Domain ROs for a particular User Domain from the DEA. As a result of the successful execution of the Get User Domain Authorization Protocol (section 7.2) with a DEA, the RI/LRM will establish a logical User Domain Context for the given User Domain. At a minimum, the User Domain Context consists of:

- Identity of the User Domain (which includes the User Domain Generation)
- An indication of the verification of the DEAs User Domain Authorization (signed by a DA). The User Domain Generation part of the User Domain Identifier is ignored by the RI/LRM.
- Expiry time of the User Domain Context (which MUST be the same or earlier than the <notAfter> element in the DEAs User Domain Authorization)
- The RI/LRM's User Domain Authorization (signed by the DEA)
- A User Domain Key Set compromised of $K_{MAC}$, $K_{REK}$, AES-WRAP( UDK, $K_{MAC}$ | $K_{REK}$ ), $K_{MAC\text{-}Leave}$ and AES-WRAP( UDK, $K_{MAC\text{-}Leave}$ ).

An RI/LRM MAY get authorized for multiple User Domains managed by one or more DEAs.

### 5.2.3 Dropping Authorization for a User Domain

If the DEA wants to indicate to an RI/LRM that it is to cease creating User Domain ROs for a particular User Domain managed by that DEA, the DEA MUST send a Drop User Domain Authorization Trigger. An RI/LRM that receives such a

Drop User Domain Authorization Trigger MUST execute the Drop User Domain Authorization protocol. Prior to sending a Drop User Domain Authorization Request message, the RI MUST disable the functionality to create any ROs for the User Domain.

As a result of the successful execution of the Drop User Domain Authorization Protocol (section 7.3) with a DEA, the RI/LRM MUST delete the User Domain Context for the given User Domain. Without the User Domain Context, the RI/LRM MUST NOT create ROs for the User Domain. If the DEA has sent a Drop User Domain Authorization Trigger, the DEA MAY consider the RI/LRM to no longer be eligible to receive User Domain Authorizations for that User Domain even if the DEA does not receive any Drop User Domain Authorization Request. The DEA MAY later reinstate the eligibility to receive User Domain Authorizations of an RI/LRM to which it has sent a Drop User Domain Authorization Trigger and/or a Drop User Domain Authorization Response.

## 5.3 Memberships and Authorizations

A DEA will enforce certain limits when allowing DRM Agents to join a User Domain. It can for example enforce a maximum on the number of concurrent members. This means it should be possible to remove a DRM Agent as member of a User Domain. Or a DEA MAY request that a DRM Agent leave a User Domain, e.g. no longer be a member of the User Domain.

For reasons related to its business or to the trust management, a DEA MAY NOT renew the User Domain Authorization for a RI/LRM.

A DEA can upgrade the User Domain (see section 8.3), which forces all DRM Agents to re-join and RI/LRMs to get re-authorized for the User Domain.

## 5.4 Compatibility with DRM v2.x Domains

The functionality provided by the Domain mechanism defined in [DRM-DRM-v2.1] is subset of the SCE User Domain functionality. User Domain ROs that do not use any new features can be used by DRM v2.x DRM Agents. Specifically, User Domain ROs that meet the criteria below are compatible with v2.x Domains:

- The RO does not contain a <userDomain> constraint ([SCE-REL])
- The RO is signed by an RI or by an LRM that has the **`oma-kp-rightsIssuer`** key purpose

User Domain ROs meeting the criteria above can be delivered to v2.x DRM Agents via any mechanism allowed in [DRM–DRM-v2.1]. The installation and consumption of this User Domain RO on a v2.x DRM Agent will cause the DRM Agent to attempt to join the domain using the ROAP protocol as specified in [DRM-DRM-v2.1]. This will cause the RI/LRM that signed the RO to execute the Proxy Join User Domain protocol (see section 7.4). If this protocol is successful, the DRM v2.x Agent will receive the information required to access the User Domain RO and will be able to acquire User Domain ROs from RI using RO Acquisition protocol as specified in [DRM-DRM-v2.1].

An RI in the User Domain can also actively trigger a v2.x DRM Agent to initiate Join Domain protocol as specified in [DRM-DRM-v2.1]. And, once RI has dropped the User Domain, the RI MUST NOT issue any more Domain ROs for this User Domain. The DEA MAY use leave Domain triggers or MAY instruct the RI (in an out-of-scope way) to trigger to leave all v2.x DRM Agents which are legal Domain members and have joined this User Domain via this RI.

# 6. The Domain Management Protocol Suite (SCE-2-DMP)

The Domain Management Protocol Suite (SCE-2-DMP) is the interface provided by a DEA to Devices. It provides the following protocols:

- DRM Agent-DEA Registration Protocol – used by Devices to register with a DEA
- Join User Domain Protocol – used by Devices to join a User Domain
- Leave User Domain Protocol – used by Devices to lease a User Domain

In the message parameter tables below, "M" stands for "parameter is mandatory" and "O" stands for "parameter is optional". If a parameter is not listed in the table, then that parameter MUST NOT be present in the mssage.

## 6.1 DRM Agent-DEA Registration Protocol

The DRM Agent-DEA Registration Protocol is a complete security information exchange and handshake between the DRM Agent and the DEA and is generally only executed at first contact, but may also be executed when there is a need to update the exchanged security information. This protocol includes negotiation of protocol parameters and protocol version, cryptographic algorithms, exchange of certificate preferences, optional exchange of certificates, mutual authentication of DRM Agent and DEA, and integrity protection of protocol messages.

Successful completion of the Registration protocol results in the establishment of a DEA Context in the DRM Agent containing DEA-specific security related information such as agreed protocol parameters, protocol version, and certificate preferences. A DEA Context is necessary for execution of the other protocols in the SCE-2-DMP suite. Figure 1 depicts the 4-pass DRM Agent-DEA Registration Protocol.

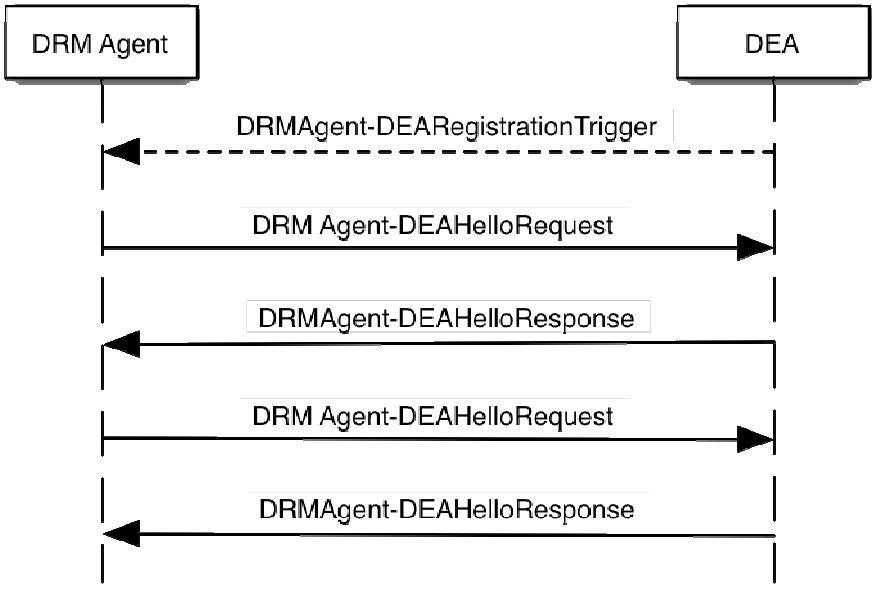


**Figure 1 - 4-pass DRM Agent-DEA Registration Protocol**

### 6.1.1 DRM Agent-DEA Registration Trigger

The DRM Agent-DEA Registration Trigger message is sent to a DRM Agent to initiate the 4-pass DRM Agent-DEA Registration protocol. The message MUST be a <drmTrigger> element as defined in the following XML schema fragment:

```
<element name="drmTrigger" type="gen:DrmTrigger"/>
```

A DRM Agent-DEA Registration trigger MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| id | O | Default, as specified in [SCE-GEN] |
| type | M | “dmpDRMAgent-DEARegistration” |
| version | M | “1.0” |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | O | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |

The DRM Agent-DEA Registration Trigger contains a <body> element that MUST NOT have a <trgInfo> child element.

### 6.1.2 DRM Agent-DEA Hello Request

The DRM Agent-DEA Hello Request message is sent from the DRM Agent to the DEA to initiate the 4-pass Registration protocol. This message expresses DRM Agent information and preferences. The message is defined by the following XML schema fragment:

```
<element name="helloRequest" type="gen:Request"/>
```

The message MUST be formatted as specified in [SCE-GEN].

Upon receipt of a DRM Agent-DEA Hello Request message, the DEA MUST perform the default processing specified in [SCE-GEN] and then it MUST return a DRM Agent-DEA Hello Response message.

### 6.1.3 DRM Agent-DEA Hello Response

The DRM Agent-DEA Hello Response message is sent from the DEA to the DRM Agent in response to a DRM Agent-DEA Hello Request message. The message expresses DEA preferences and decisions based on the values supplied by the DRM Agent. The message is defined by the following XML schema fragment:

```
<element name="helloResponse" type="gen:Response"/>
```

The messageMUST be formatted as specified in [SCE-GEN].

### 6.1.4 DRM Agent-DEA Registration Request

A DRM Agent sends the DRM Agent-DA Registration Request message to a DEA to request registration with the DEA after receiving a successful DRM Agent-DEA Hello Response. The message is defined by the following XML schema fragment:

```
<element name="registrationRequest" type="gen:Request"/>
```

The message MUST be formatted as specified in [SCE-GEN].

### 6.1.5 DRM Agent-DEA Registration Response

The DRM Agent-DEA Registration Response message is sent from the DEA to the DRM Agent in response to a DRM Agent-DEA Registration Request message. This message completes the Registration protocol and, if successful, enables the DRM Agent to establish a DEA Context for this DEA. The message is defined by the following XML schema fragment:

```
<element name="registrationResponse" type="gen:Response"/>
```

The message MUST be formatted as specified in [SCE-GEN].

## 6.2 Join User Domain Protocol

The following figure illustrates the Join User Domain Protocol.

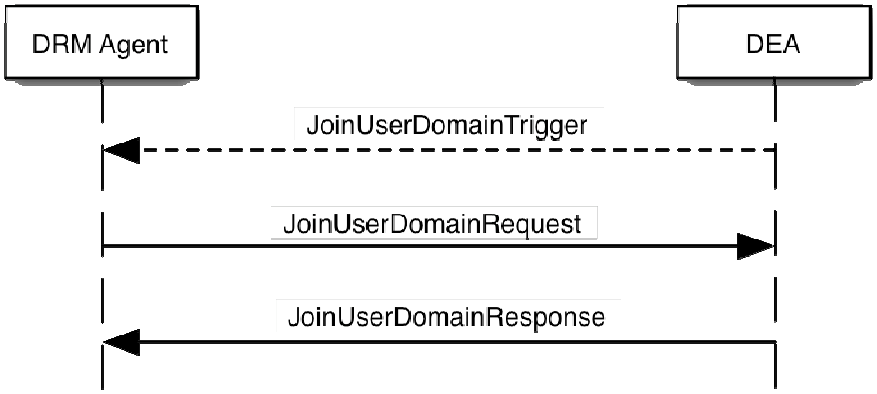


**Figure 2 - Join User Domain Protocol**

### 6.2.1 Join User Domain Trigger

A Join User Domain Trigger message is sent to a DRM Agent to initiate the 2-pass Join User Domain protocol. The message MUST be a <gen:trigger> element as defined in the following XML schema fragment:

<element name="drmTrigger" type="gen:DrmTrigger"/>A Join User Domain Trigger message MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| id | O | Default, as specified in [SCE-GEN] |
| type | M | “dmpJoinUserDomain” |
| version | M | “1.0” |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | O | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |
| signature | M | Specified below |

**Table 2 - Join User Domain Trigger Message Parameters**

The Join User Domain Trigger contains a <body> element that MUST have a <trgInfo> child element which MUST have a <dmpJoinUserDomainTrgInformation> element as defined by the following XML schema fragment:

```
<element name=" dmpJoinUserDomainTrgInformation">
    <complexType>
        <sequence>
            <element name="userDomainAlias" type="gen:String80" minOccurs="0"/>
            <element name="userDomainID" type="dom:UserDomainIdentifier"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element contains the identification of User Domain, see section 4.1.

*userDomainAlias*: This element, if present, contains a string value that SHALL be used by the DRM Agent whenever it refers to the domain specified by <userDomainID> in a message to the User. The content of the <userDomainAlias> element SHALL be saved in the User Domain Context. The maximum length of this element SHALL be 80 bytes.

*signature*: This element contains a MAC value over the trigger besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the DRM Agent.

Upon receipt of a Join User Domain Trigger, the DRM Agent MUST perform the default processing as specified in [SCE-GEN] and if successful post a Join User Domain Request.

### 6.2.2 Join User Domain Request

A Join User Domain Request message is sent from a DRM Agent to a DEA as the first message of the 2-pass Join User Domain protocol. The message MUST be a <dmpJoinUserDomainRequest> element as defined in the following XML schema fragment:

```
<element name="dmpJoinUserDomainRequest" type="gen:Request"/>
```

A Join User Domain Request message MUST be formatted as specified in the table below:

| element / attribute | Usage | value |
|---|---|---|
| triggerNonce | O | Default, as specified in [SCE-GEN] |
| reqID | M | DRM Agent ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below |
| signature | M | Specified below |

**Table 3 - Join User Domain Request Message Parameters**

The Join User Domain Request message MUST contains a <reqInfo> element which MUST contain a <dmpJoinUserDomainRequestInformation> element as defined by the following XML schema fragment:

```
<element name="dmpJoinUserDomainRequestInformation">
    <complexType>
        <sequence>
            <element name="peerKeyIdentifier" type="gen:KeyIdentifier" minOccurs="0"/>
            <element name="userDomainID" type="dom:UserDomainIdentifier" />
        </sequence>
        <attribute name="hashChainsSupported" type="boolean" default="false"/>
    </complexType>
</element>
```

*userDomainID*: If the DRM Agent received a Join User Domain Trigger, then the value of the <userDomainID> element MUST be equal to the <userDomainID> received in the trigger. Otherwise, the value MUST be equal to the <userDomainID> from a ProtectedRO that is received out-of-band.

*peerKeyIdentifier*: An identifier for a DEA public key stored in the DRM Agent. If the identifier matches the stored DEA ID as specified in section 7.1, it means the DRM Agent has already stored the DEA ID and the corresponding DEA certificate chain, and the DEA need not send its certificate chain in its response message.

*hashChainsSupported*: This element, if present, indicates that the the DRM Agent supports the hash chain mechanism (see section 8.4).

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

Upon receipt of a Join User Domain Request, the DEA MUST perform the default processing, as specified in [SCE-GEN] and MUST return a Join User Domain Response.

### 6.2.3 Join User Domain Response

A Join User Domain Response message is sent from a DEA to a DRM Agent as the last message of the 2-pass Join User Domain protocol. A Join User Domain Response message is also sent from a DEA to a DRM Agent as the first message of

the 1-pass Join User Domain protocol. The message MUST be a <dmpJoinUserDomainResponse> element as defined in the following XML schema fragment:

```
<element name="dmpJoinDomainResponse" type="gen:Response"/>
```

If the processing of the Join User Domain Request message was not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | Usage | Value |
|---|---|---|
| Status | M | "Success" |
| reqID | M | DRM Agent ID |
| resID | M | DEA ID |
| Nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below |
| Signature | M | Specified below |

**Table 4 - Join User Domain Response Message Parameters**

The Join User Domain Response message MUST have a <resInfo> element which MUST have a <dmpJoinUserDomainResponseInformation> as defined by the following XML schema fragment:

```
<element name="dmpJoinUserDomainResponseInformation">
    <complexType>
        <sequence>
            <element name="daCertificateChain" type="gen:CertificateChain" minOccurs="0"/>
            <element name="userDomainAuthorization" type="dom:UserDomainAuthorizationType"
                maxOccurs="unbounded"/>
            <sequence maxOccurs="unbounded">
                <element name="userDomainID" type="dom:UserDomainIdentifier"/>
                <element name="udk" type="base64Binary" />
            </sequence>
        </sequence>
        <attribute name="willUsehashChains" type="boolean" default="false"/>
    </complexType>
</element>
```

*joinUserDomainInfo*: This element contains information on the User Domain. Its child elements are described below.

*daCertificateChain*: This element contains the certificate chain of the DA that authorizes the DEA to managed the User Domain. The DRM Agent MUST verify this certificate chain.

*userDomainAuthorization*: This element contains User Domain Authorizations.

The DEA MUST include a "DEA" User Domain Authorization for which:

1. The <dadeaID> identifies the DA
2. The <entityID> identifies the DEA
3. The User Domain baseID of the <userDomainID> equals User Domain baseID of the <userDomainID> from the JoinUserDomainRequest
4. An <isDea> element is present.
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element

6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element.

7. The <signature> verification using the DAs public key is successful.

The DRMAgent MUST check that such User Domain Authorization is present. If no such User Domain Authorization is present, then the DRM Agent MUST abort the join domain process.

The DRMAgent MUST check that such User Domain Authorization is present. If no such User Domain Authorization is present, then the DRM Agent MUST abort the join domain process.

If the User Domain Authorization above includes a *daDeviceAuthorizationRequired* element then the DEA MAY include a second User Domain Authorization, for which:

1. The <dadeaID> identifies the DA
2. The <entityID> identifies the DRM Agent
3. The <deaID> identifies the DEA
4. An <isDRMAgent> element is present.
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element
6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element.
7. The <signature> verification using the DA's public key is successful.

The DRM Agent MUST check that such second User Domain Authorization is present if

a) the DRM Agent has not previously received a User Domain Authorization meeting these criteria that is currently valid, and

b) the first User Domain Authorization includes a *daDeviceAuthorizationRequired* element.

If such second User Domain Authorization is not present and the DRM Agent has not previously received a User Domain Authorization meeting these criteria that is currently valid, then the DRM Agent MUST abort the join domain process if the first User Domain Authorization includes a *daDeviceAuthorizationRequired* element.

The DEA MUST include a "DEA" User Domain Authorization for which:

1. The <dadeaID> identifies the DEA
2. The <entityID> identifies the DRM Agent
3. The User Domain baseID of the <userDomainId> equals User Domain baseID of the <userDomainId> from the JoinUserDomainRequest
4. An <isDRM Agent> element is present.
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element
6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element.
7. The <signature> verification using the DEA's public key is successful.

The DRMAgent MUST check that such User Domain Authorization is present. If no such User Domain Authorization is present, then the DRM Agent MUST abort the join domain process.

The DEA SHOULD include the User Domain Authorizations of all RI/LRMs that are currently authorized to create ROs for the User Domain. These are required for the installation of User Domain ROs, as specified in section 8.2.

*udk*: This element contains the UDK for the User Domain encrypted by the SK. If Hash Chains are supported by both the Device and the DEA, only the UDK corresponding to the most recent User Domain generation SHOULD be included, otherwise all UDKs for all User Domain generations MUST be included (including their User Domain identifiers as Id attributes).

*willUsehashChains*: This element, if present, indicates that the DEA will use the hash chain mechanism of section 8.4. This element MUST be omitted if the DRM Agent did not indicate that it supports the hash chain mechanism in the Join User Domain Request.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the DRM Agent.

# 6.3 Leave User Domain Protocol

The following figure illustrates the Join User Domain Protocol.

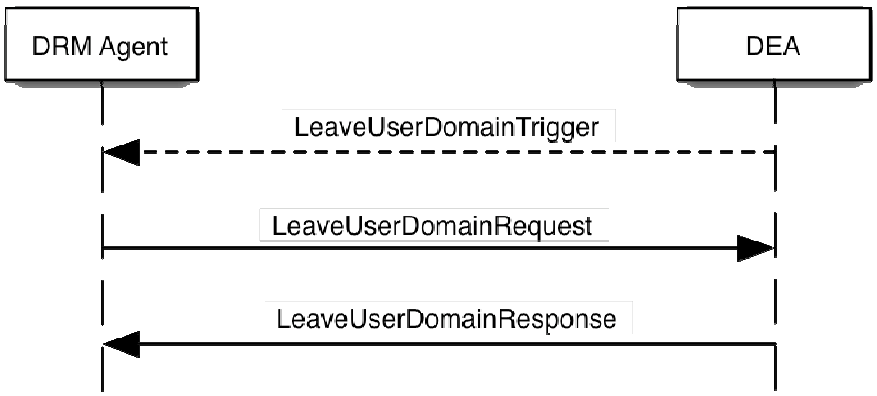


**Figure 3 - Leave User Domain Protocol**

## 6.3.1 Leave User Domain Trigger

A Leave User Domain Trigger is delivered to a DRM Agent to initiate the 2-pass Leave User Domain protocol. The message MUST be a <gen:trigger> element as defined in the following XML schema fragment:

```
<element name="drmTrigger" type="gen:DrmTrigger"/>
```

A Leave User Domain Trigger message MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| id | O | Default, as specified in [SCE-GEN] |
| type | M | "dmpLeaveUserDomain" |
| version | M | "1.0" |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | M | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |
| signature | M | Specified below |

**Table 5 – Leave User Domain Trigger Message Parameters**

The Leave User Domain Trigger contains a <body> element that MUST have a <trgInfo> child element which MUST have a <dmpLeaveUserDomainTrgInformation> element as defined by the following XML schema fragment:

```
<element name="dmpLeaveUserDomainTrgInformation">
    <complexType>
        <sequence>
            <element name="userDomainAlias" type="gen:String80" minOccurs="0"/>
            <element name="userDomainID" type="dom:UserDomainIdentifier"/>
            <element name="deviceID" type="gen:Identifier" minOccurs="0"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element identifies the User Domain, see section 4.1.

*userDomainAlias*: This element contains a string value that SHALL be used by the DRM Agent whenever it refers to the domain specified by <userDomainID> in a message to the User. The maximum length of this element SHALL be 80 bytes.

*deviceID*: This element, if present, MUST be verified by the DRM Agent as to whether the value of the <deviceID> matches its Device ID. If the *deviceID* does not match, the DRM Agent MUST discard the trigger.

*signature*: This element contains a MAC value over the trigger besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the DRM Agent.

Upon receipt of a Leave User Domain Trigger, the DRM Agent MUST perform the default processing, as specified in [SCE-GEN] and if successful send a Leave User Domain Request.

### 6.3.2 Leave User Domain Request

A Leave User Domain Request message is sent from a DRM Agent to a DEA as the first message of the 2-pass Leave User Domain protocol. The message MUST be a <dmpLeaveUserDomainRequest> element as defined in the following XML schema fragment:

```
<element name="dmpLeaveUserDomainRequest" type="gen:Request"/>
```

A Leave User Domain Request message MUST be formatted as specified in the table below:

| element / attribute | Usage | value |
|---|---|---|
| triggerNonce | O | Default, as specified in [SCE-GEN] |
| reqID | M | DRM Agent ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below |
| signature | M | Specified below |

**Table 6 – Leave User Domain Request Message Parameters**

The Leave User Domain Request message MUSR contains a <reqInfo> element which MUST contain a <dmpLeaveUserDomainRequestInformation> element as defined by the following XML schema fragment:

```
<element name="dmpLeaveUserDomainRequestInformation">
    <complexType>
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier" />
        </sequence>
        <attribute name="notAMember" type="boolean" default="false"/>
    </complexType>
</element>
```

*userDomainID*: This element contains the identity of the User Domain from which the Device is leaving. If the DRM Agent received a Leave User Domain Trigger, then the value of this element MUST be equal to the <userDomainID> element received in the Leave User Domain trigger (see section 6.3.1).

*notAMember*: This element, if present, indicates that the DRM Agent does not consider itself a member of this User Domain. This could happen, for example, if the DRM Agent already has left the User Domain, but receives a new trigger to leave it (perhaps because the DEA never received the previous Leave User Domain Request).

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

Upon receipt of a Leave User Domain Request, the DEA MUST perform the default processing, as specified in [SCE-GEN] and MUST return a Leave User Domain Response.

### 6.3.3 Leave User Domain Response

A Leave User Domain Response message is sent from a DEA to a DRM Agent as the last message of the 2-pass Leave User Domain protocol. The message MUST be a <dmpLeaveUserDomainResponse> element as defined in the following XML schema fragment:

```
<element name="dmpLeaveUserDomainResponse" type="gen:Response"/>
```

If the processing of the Leave User Domain Request is not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | Usage | value |
|---|---|---|
| status | M | "Success" |
| reqID | M | DRM Agent ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below. |
| signature | M | Specified below |

**Table 7 – Leave User Domain Response Message Parameters**

The Leave User Domain Response message MUST have a <resInfo> which MUST have a <dmpLeaveUserDomainResponseInformation> element as defined by the following XML schema fragment:

```
<element name="dmpLeaveUserDomainResponseInformation">
    <complexType>
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier" />
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element identifies the User Domain from which the DEA removed the DRM Agent. The User Domain Generation part of the User Domain Identifier SHALL be ignored.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the DRM Agent.

The DEA sends the Leave User Domain Response after having deleted the association of this DRM Agent to the User Domain (i.e. updated the User Domain membership status).

# 7. The Rights Issuer – Domain Enforcement Agent Protocol Suite (SCE-3-RDP)

The Rights Issuer – Domain Enforcement Agent Protocol Suite (SCE-3-RDP) is the interface provided by a DEA to RIs. This interface provides the following functionality:

- RI-DEA Registration - used by RIs to register with a DEA
- Getting User Domain Authorization - used by RIs to get authorization for a User Domain
- Dropping User Domain Authorization - used by RIs to drop authorization for a User Domain

Since DRM v2.x Devices support only DRM v2.x Domains and can not interact with a DEA by any defined protocol, the following protocols are also available to allow DRM v2.x Devices to take advantage of User Domains.

- Proxy Join User Domain - used by RIs to allow a v2.x Device to join a User Domain
- Proxy Leave User Domain - used by RIs allow a v2.x Device to leave a User Domain

All the protocols of the SCE-3-RDP interface are also available to LRMs via the SCE-5-LRMP interface by replacing “RI” with “LRM”.

## 7.1 RI-DEA Registration Protocol

The RI-DEA Registration Protocol is a complete security information exchange and handshake between the RI and the DEA and is generally only executed at first contact, but may also be executed when there is a need to update the exchanged security information. This protocol includes negotiation of protocol parameters and protocol version, cryptographic algorithms, exchange of certificate preferences, optional exchange of certificates, mutual authentication of RI and DEA, and integrity protection of protocol messages.

Successful completion of the Registration protocol results in the establishment of a DEA Context in the RI containing DEA-specific security related information such as agreed protocol parameters, protocol version, and certificate preferences. A DEA Context is necessary for execution of the other protocols in the SCE-3-RDP suite. Figure 4 depicts the 4-pass RI-DEA Registration Protocol.

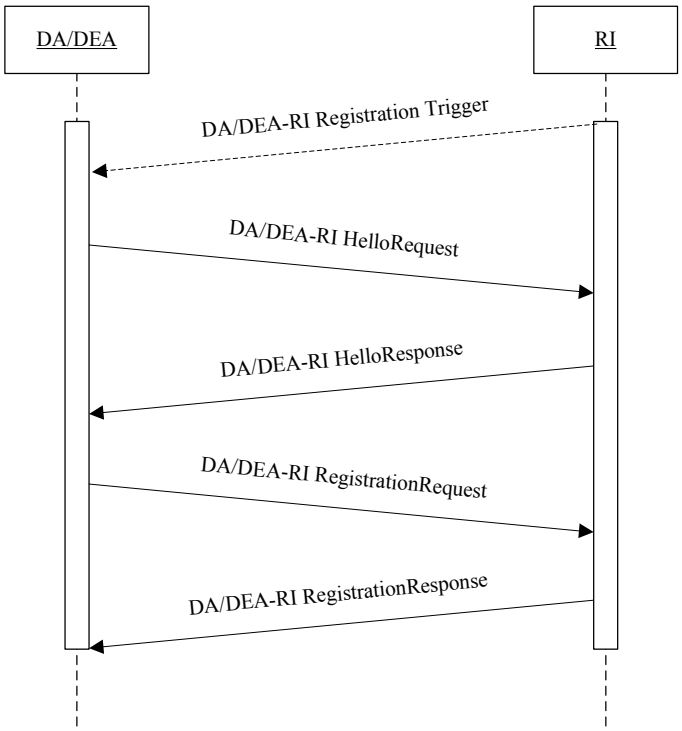


**Figure 4 - The 4-pass RI-DEA Registration Protocol**

### 7.1.1 RI-DEA Hello Request

The RI-DEA Hello Request message is sent from the RI to the DEA to initiate the 4-pass Registration protocol. This message expresses RI information and preferences and MUST be formatted as specified in [SCE-GEN].

Upon receipt of a RI-DEA Hello Request, the DEA MUST perform the default processing, as specified in [SCE-GEN] and MUST return a RI-DEA Hello Response.

### 7.1.2 RI-DEA Hello Response

The RI-DEA Hello Response message is sent from the DEA to the RI in response to a RI-DEA Hello Request message. The message expresses DEA preferences and decisions based on the values supplied by the RI and MUST be formatted as specified in in [SCE-GEN].

### 7.1.3 RI-DEA Registration Request

An RI sends the RI-DEA Registration Request message to a DEA to request registration with the DEA. The message is sent as the third message in the 4-pass Registration protocol and MUST be formatted as specified in [SCE-GEN]

### 7.1.4 RI-DEA Registration Response

The RI-DEA Registration Response message is sent from the DEA to the RI in response to a RI-DEA Registration Request message. This message completes the Registration protocol, and if successful, enables the RI to establish a DEA Context and MUST be formatted as specified in [SCE-GEN]. Additionally, the <resInfo> element MUST include a <rdpRegistrationResponseInfo> element as specified by the following XML schema fragment:

```
<element name="rdpRegistrationResponseInfo" >
    <complexType>
        <sequence>
            <element name="resURL" type="anyURI"/>
            <element name="encSa" type="base64Binary"/>
        </sequence>
    </complexType>
</element>
```

*resURL*: This element contains the URL that the RI MUST send future requests to the DEA. The value of this element MUST be a URL according to [RFC2396], and MUST be an absolute identifier.

*encSa*: This element contains an encrypted security association (SA). The SA contains a symmetric key (referred to as the SK) for encrypting data between the RI and DEA, concatenated a Message Integrity Key (MK) for providing integrity protection. For the default algorithms, the SA contains a 128-bit AES key followed by a 160-bit HMAC-SHA1 key. The SA is encrypted using the RI public key.

## 7.2 Get User Domain Authorization Protocol

The following figure illustrates the Get User Domain Authorization Protocol:

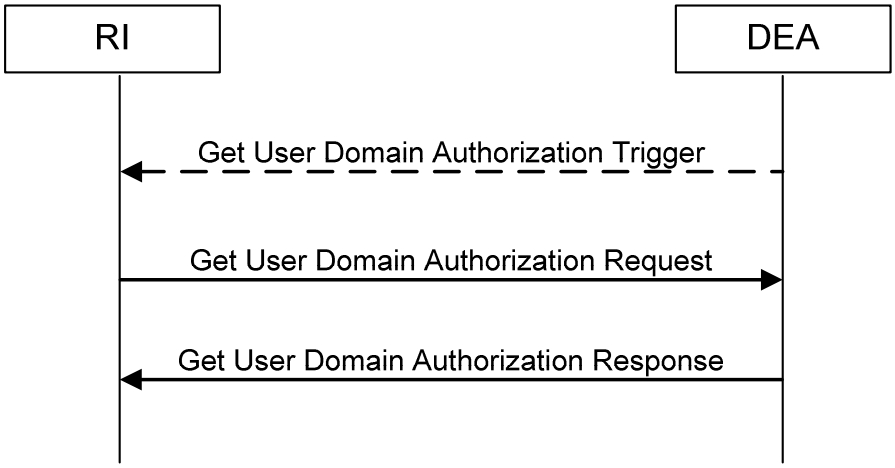


**Figure 5 - Get User Domain Authorization Protocol**

### 7.2.1 Get User Domain Authorization Trigger

A Get User Domain Authorization Trigger message is sent to an RI from a DEA to initiate the 2-pass Get User Domain Authorization protocol. The message MUST be a <gen:trigger> element as defined by the following XML schema fragment:

```
<element ref="gen:trigger"/>
```

A Get User Domain Authorization Trigger MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| id | O | Default, as specified in [SCE-GEN] |
| type | M | “rdpGetUserDomainAuthorization” |
| version | M | “1.0” |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | O | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |
| signature | M | Specified below |

**Table 8 - Get User Domain Authorization Trigger Message Parameters**

The Get User Domain Authorization Trigger contains a <body> element that MUST have a <trgInfo> child element which MUST contain a <rdpGetUserDomainAuthorizationTrgInformation> as defined by the following XML schema fragment:

```
<element name="rdpGetUserDomainAuthorizationTrgInformation">
    <complexType>
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier"/>
            <element name="userDomainAlias" type="gen:String80" minOccurs="0"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element identifies the User Domain, see section 4.1.

*userDomainAlias*: This element contains a string value that SHALL be used by the RI whenever it refers to the domain specified by <userDomainID> in a message to the User. The content of the <userDomainAlias> element SHALL be saved in the User Domain Context. The maximum length of this element SHALL be 80 bytes.

*signature*: This element contains a MAC value over the trigger besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

Upon receipt of a Get User Domain Authorization Trigger, the RI MUST perform the default processing as specified in [SCE-GEN] and if successful post a Get User Domain Authorization Request.

## 7.2.2 Get User Domain Authorization Request

The Get User Domain Authorization Request message is sent from the RI to the DEA to initiate the 2-pass Get User Domain Authorization protocol. The message MUST be a <rdpGetUserDomainAuthorization> element as defined by the following XML schema fragment:

```
<element name=”rdpGetUserDomainAuthorization” type=”gen:Request”/>
```

A Get User Domain Authorization Request MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| triggerNone | O | Default, as specified in [SCE-GEN] |
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below |
| signature | M | Specified below |

**Table 9 - Get User Domain Authorization Request Message Parameters**

The Get User Domain Authorization Request message contains a <reqInfo> element which MUST contain a <rdpGetUserDomainAuthorizationRequestInformation> element as defined by the following XML schema fragment:

```
<element name="rdpGetUserDomainAuthorizationRequestInformation">
    <complexType >
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier" />
            <element name="domainID" type="roap:DomainIdentifier" minOccurs="0"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element contains the identification of the User Domain the RI wants the User Domain Authorization for. This element MAY have a *domainID* attribute. This attribute, if present, indicates that the RI wants to map the Domain identified by the attribute's value to the User Domain. If this attribute is not present, then the DEA MUST not allow the Proxy Join User Domain protocol for this User Domain.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

Upon receipt of a Get User Domain Authorization Request and processing the message, the DEA MUST return a Get User Domain Authorization Response.

### 7.2.3 Get User Domain Authorization Response

The Get User Domain Authorization Response message is sent from the DEA to the RI as the last message of the 2-pass Get User Domain Authorization Request protocol. A Get User Domain Authorization Response message is also sent from a DEA to an RI as the first message of the 1-pass Get User Domain Authorization protocol. The message MUST be a <dmpGetUserDomainAuthorizationResponse> element as defined in the following XML schema fragment:

```
<element name="dmpGetUserDomainAuthorizationResponse" type="gen:Response"/>
```

If the processing of the Get User Domain Authorization Request message was not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | usage | value |
|---|---|---|
| status | M | ”Success” |
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below |
| signature | M | Specified below |

**Table 10 - Get User Domain Authorization Response Message Parameters**

The Get User Domain Authorization Response message MUST have a <resInfo> element which MUST have a <rdpGetUserDomainAuthorizationResponseInformation> element as defined by the following XML schema fragment:

```
<element name="rdpGetUserDomainAuthorizationResponseInformation">
    <complexType >
        <sequence>
            <element name="daCertificateChain" type="gen:CertificateChain" minOccurs="0"/>
            <element name="userDomainAuthorization" type="dom:UserDomainAuthorizationType"
                maxOccurs="unbounded"/>
            <element name="encUserDomainKeySet" type="base64Binary"/>
        </sequence>
    </complexType>
</element>
```

*userUserDomainAuthorization*: This element contains a User Domain Authorizations Several of these elements may be included in the response.

The DEA MUST include a User Domain Authorization for which:

1. The <dadeaID> identifies the DA
2. The <entityID> identifies the DEA
3. The User Domain baseID of the <userDomainID> equals User Domain baseID of the <userDomainID> from the getUserDomainAuthorizationRequest.
4. An <isDea> element is present
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element
6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element.
7. The <signature> verification using the DA’s public key is successful.

The RI/LRM MUST check that such User Domain Authorization is present. If no such User Domain Authorization is present, then the RI/LRM MUST abort the get user domain authorization process.

If the User Domain Authorization above includes a *daRiAuthorizationRequired* element then the DEA MAY include a second User Domain Authorization, for which:

1. The <dadeaID> identifies the DA
2. The <entityID> identifies the RI/LRM
3. The <deaID> identifies the DEA

4. An <isRIorLRM> element is present.
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element
6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element.
7. The <signature> verification using the DA's public key is successful.

The RI/LRM MUST check that such second User Domain Authorization is present if

a) the RI/LRM has not previously received a User Domain Authorization meeting these criteria that is currently valid, and

b) the first User Domain Authorization includes a *daRiAuthorizationRequired* element.

If such second User Domain Authorization is not present and the RI/LRM has not previously received a User Domain Authorization meeting these criteria that is currently valid, then the RI/LRM MUST abort the get user domain authorization process if the first User Domain Authorization includes a *daRiAuthorizationRequired* element.

The DEA MUST include a "RI/LRM" User Domain Authorization for which:

1. The <dadeaID> identifies the DEA
2. The <entityID> identified the RI/LRM
3. A <isRIorLRM> elemtent is included.
4. The <signature> verification using the DEA's public key is successful.

This User Domain Authorization is required for the installation of User Domain ROs, as specified in section 4.2. The RI/LRM will pass this on to DRM Agents in the ROs. See section 8.1.

*encUserDomainKeySet*: This element contains an encrypted set of keys for the User Domain. The set of keys are encrypted using the negotiated algorithm and using the SK of the ReqContext for the RI. The plaintext keys are described in the following table.

| **Key** | **Description** |
|---|---|
| $K_{MAC}$ | A MAC key used by the RI when delivering a User Domain RO. |
| $K_{REK}$ | A REK used by the RI when delivering a User Domain RO. |
| AES-WRAP(UDK, $K_{MAC}$\|$K_{REK}$) | The AES-WRAP, using the UDK, of the concatenation of KMAC and KREK. Used by the the RI when delivering a User Domain RO. |
| $K_{MAC\text{-}Leave}$ | A MAC key used by the RI when a Device is leaving the User Domain. |
| AES-WRAP(UDK, $K_{MAC\text{-}Leave}$) | The AES-WRAP, using the UDK, of KMAC-Leave. Used by the RI when a Device is leaving the User Domain. |

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

## 7.3 Drop User Domain Authorization Protocol

The following figure illustrates the Drop User Domain Authorization Protocol:

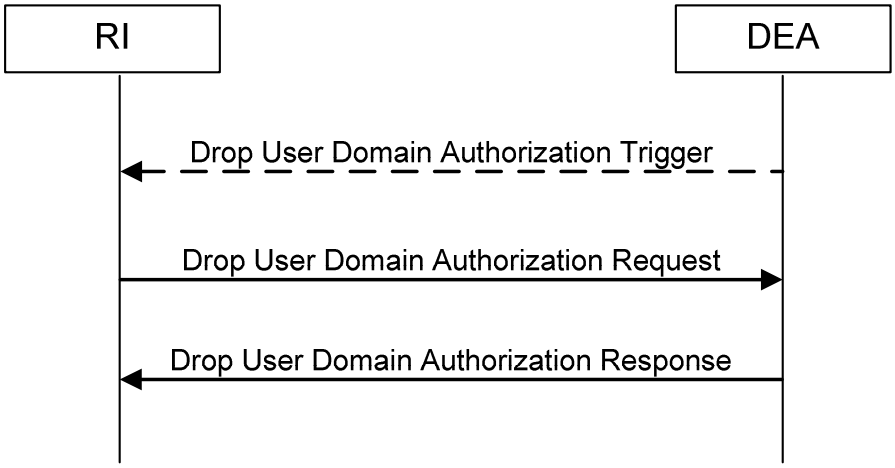


**Figure 6 - Drop User Domain Authorization Protocol**

### 7.3.1 Drop User Domain Authorization Trigger

A Drop User Domain Authorization Trigger message is sent to a RI to initiate the 2-pass Drop User Domain Authorization protocol. The root element of the message MUST be a <gen:trigger> element as defined in the following XML schema fragment:

```
<element name="drmTrigger" type="gen:DrmTrigger"/>
```

A Drop User Domain Authorization Trigger message MUST be formatted as specified in the following table.

| element / attribute | usage | value |
|---|---|---|
| id | O | Default, as specified in [SCE-GEN] |
| type | M | “rdpDropUserDomainAuthorization” |
| version | M | “1.0” |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | O | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |
| signature | O | Specified below |

**Table 11 – Drop User Domain Authorization Trigger Message Parameters**

The Drop User Domain Authorization Trigger contains a <body> element that MUST have a <trgInfo> child element which MUST have a <rdpDropUserDomainAuthorizationTrgInformation> element as defined by the following XML schema fragment:

```
<element name="rdpDropUserDomainAuthorizationTrgInformation">
    <complexType>
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier"/>
            <element name="userDomainAlias" type="gen:String80" minOccurs="0"/>
```

```
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element identifies the User Domain to drop, see section 4.1.

*userDomainAlias*: This element, if present, contains a string value that SHALL be used by the RI whenever it refers to the domain specified by <userDomainID> in a message to the User. The content of the <userDomainAlias> element SHALL be saved in the User Domain Context. The maximum length of this element SHALL be 80 bytes.

*signature*: This element contains a MAC value over the trigger besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

In the case that an RI receives an rdpDropDomainTrigger, the RI SHALL check the authorization status of the RI and then SHALL send the rdpDropDomainRequest to the DEA.

### 7.3.2 Drop User Domain Authorization Request

The Drop User Domain Authorization Request message is sent from a RI to a DEA as the first message of the 2-pass Drop User Domain Authorization protocol. The message MUST be a <rdpDropUserDomainAuthorizationRequest> element as defined in the following XML schema fragment:

```
<element name="rdpDropUserDomainAuthorizationRequest" type="gen:Request"/>
```

A Drop User Domain Authorization Request message MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| triggerNonce | O | Default, as specified in [SCE-GEN] |
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below. |
| signature | M | Specified below |

**Table 12 - Drop User Domain Authorization Request Message Parameters**

The Drop User Domain Authorization Request message contains a <reqInfo> element which MUST have a <rdpDropUserDomainAuthorizationRequesInformation> element as defined by the following XML schema fragment:

```
<element name="rdpDropUserDomainAuthorizationRequesInformation">
    <complexType >
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="notAuthorized" type="boolean" default="false"/>
    </complexType>
</element>
```

*userDomainID*: If the RI received a Drop User Domain Authorization Trigger, then the value of the <userDomainID> element MUST be equal to the <userDomainID> received in the trigger. Otherwise, the value contains the ID of the User Domain the RI wishes to drop its authorization for.

*notAuthorized*: This element, if present, indicates that the RI does not have a User Domain Authorization for the User Domain. This could happen, for example, if the RI already has dropped the authorization for a User Domain but receives a new trigger to drop the authorization (perhaps because the DEA never received the previous Drop User Domain Authorization Request or the authorization of the RI has already expired).

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

Prior to sending a Drop User Domain Authorization Request message, the RI MUST disable the ability to create any ROs for the User Domain.

### 7.3.3 Drop User Domain Authorization Response

If the DEA receives a valid Drop User Domain Authorization Request message, then the DEA MUST update the authorization status of the RI/LRM and send the Drop User Domain Authorization Response message to the RI/LRM.

The Drop User Domain Authorization Response message is sent from the DEA to the RI/LRM as the last message of the 2-pass Drop User Domain Authorization protocol. A Drop User Domain Authorization Response message is also sent from a

DEA to an RI/LRM as the first message of the 1-pass Drop User Domain Authorization protocol. The message MUST be a <rdpDropUserDomainAuthorizationResponse> element as defined in the following XML schema fragment:

```
<element name="rdpGetUserDomainAuthorizationResponse" type="gen:Response"/>
```

If the processing of the Drop User Domain Authorization Request message was not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | usage | value |
|---|---|---|
| status | M | “Success” |
| reqID | M | RI/LRM's ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below. |
| signature | M | Specified below |

**Table 13 - Drop User Domain Response Message Parameters**

The Drop User Domain Authorization Response message MUST have a <resInfo> element which MUST have a <rdpDropUserDomainAuthorizationResponseInformation> element as defined by the following XML schema fragment:

```
<element name="rdpDropUserDomainAuthorizationResponseInformation">
    <complexType >
        <sequence>
            <element name="userDomainID" type="dom:UserDomainIdentifier"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element identifies the User Domain for which the RI/LRM is dropping its User Domain Authorization.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI/LRM.

After receiving a valid Drop User Domain Authorization Response message with "Success" as the status, the RI/LRM MUST delete the corresponding User Domain Context.

If the value of the <status> element within a valid Drop User Domain Authorization Response message is not “Success”, the RI/LRM MUST re-enable the functionality to create ROs for the User Domain.

# 7.4 Proxy Join User Domain Protocol

By an RI serving as a proxy, a v2.x DRM Agent can indirectly join a User Domain as illustrated in Figure 1.

The v2.x DRM Agent sends the RI a **JoinDomainRequest** message ([DRM-DRM-v2.1]) to convey the information about the target Domain that it will join. When the RI determines that the target Domain is a User Domain, it sends the DEA a **rdpProxyJoinUserDomainRequest** message to forward the v2.x DRM Agent's request as indicated by the preceding **JoinDomainRequest** message. After processing the **rdpProxyJoinUserDomainRequest** message, the DEA returns a **rdpProxyJoinUserDomainResponse** to the RI to convey its reaction to the request, and the RI subsequently returns a **JoinDomainResponse** to the v2.x DRM Agent to forward the reaction indicated by preceding **rdpProxyJoinUserDomainResponse**.

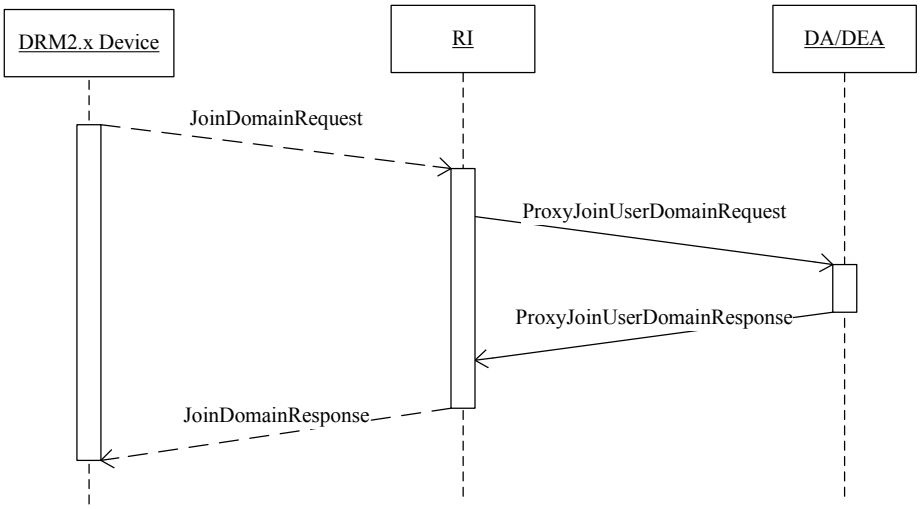


**Figure 7 - v2.x DRM Agent indirectly joins a User Domain**

### 7.4.1 Proxy Join User Domain Request

The Proxy Join User Domain Request message is sent from a RI to a DEA as the first message of the 2-pass Proxy Join User Domain protocol. The message MUST be a <rdpProxyJoinUserDomainRequest> element as defined in the following XML schema fragment:

```
<element name=" rdpProxyJoinUserDomainRequest " type="gen:Request"/>
```

A Proxy Join User Domain Request message MUST be formatted as specified in the table below:

| element / attribute | usage | value |
|---|---|---|
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below |
| signature | M | Specified below |

**Table 14 - Proxy Join User Domain Request Message Parameters**

The Proxy Join User Domain Request message contains a <reqInfo> element which MUST have a <rdpProxyJoinUserDomainRequestInformation> element as defined by the following XML schema fragment:

```
<element name="rdpProxyJoinUserDomainRequestInformation">
    <complexType >
        <sequence>
            <element name="joinDomainRequest" type="base64Binary"/>
            <element name="deviceCertificateChain" type="gen:CertificateChain" minOccurs="0"/>
        </sequence>
    </complexType>
</element>
```

*joinDomainRequest*: this element contains the joinDomainRequest message as received by the RI from the DRM Agent.

*deviceCertificateChain*: this element, if present, contains the certificate chain for the Device. If the joinDomainRequest message does not include the <certificateChain> field, this element MUST be present. The value of the <deviceCertificateChain> element SHALL be a certificate chain including the Device's certificate. The chain SHALL not include the root certificate. The Device certificate MUST come first in the list. Each following certificate MUST directly certify the one preceding it.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

After reception of the Proxy Join User Domain Request message, the DEA MUST perform the default processing, as specified in [SCE-GEN], and MUST execute the following procedure:

- Verify the validity of the Device's certificate chain. This certificate chain is contained in either the <certificateChain> element in the <joinDomainRequest> element, or in the <deviceCertificateChain> element in the <ProxyJoinUserDomainRequest> message. The certificate chain validation includes verification of the revocation status. If the certificate chain validation fails, the DEA MUST send to the RI a Proxy Join User Domain Response message with the *status* attribute set to "InvalidCertificateChain".

- The DEA MUST check that the Device certificate includes the `oma-kp-drmAgent` -keypurpose, but does NOT include the `oma-kp-sceDrmAgent` -keypurpose. If this check fails, the DEA MUST send to the RI a Proxy Join User Domain Response message with the *status* attribute set to "InvalidCertificateChain".

- Verify the signature in the <signature> element in the <joinDomainRequest> element. If the signature verification fails, the DEA MUST send the RI a Proxy Join User Domain Response message with the *status* attribute set to "SignatureError".
- If the <roap:domainID> in the joinDomainRequest is unknown to the DEA, the DEA MUST send to the RI a Proxy Join User Domain Response message with the *status* attribute set to "InvalidDomain".
- If the DRM Agent cannot be joined to the User Domain because the User Domain has already reached the maximum number of Devices, the DEA MUST send to the RI a Proxy Join User Domain Response message with the *status* attribute set to "DomainFull". Note: a DRM Agent MUST only be counted once against the maximum number of Devices in a User Domain, no matter how many times it appears in a Proxy Join User Domain Request message.
- If the DEA wants to reject the DRM Agent joining the User Domain for any other reason than the ones stated above, the DEA MUST send to the RI a ProxyJoinUserDomainResponse message with the *status* attribute set to "DomainAccessDenied".

When the DEA allows the DRM Agent to join the User Domain, it MUST return a Proxy Join User Domain Response message to the RI to convey the User Domain Information including the Domain Keys and the lifetime of the Domain. The DEA SHOULD record the association of the DRM Agent and the User Domain and the RI.

## 7.4.2 Proxy Join User Domain Response

The Proxy Join User Domain Response message is sent from the DEA to the RI as the last message of the 2-pass Proxy Join User Domain Request protocol. The message MUST be a <rdpProxyJoinUserDomainResponse> element as defined in the following XML schema fragment:

```
<element name="rdpProxyJoinUserDomainResponse" type="gen:Response"/>
```

If the processing of the Proxy Join User Domain Request message was not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | usage | value |
|---|---|---|
| status | M | “Success” |
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below |
| signature | M | Specified below |

**Table 15 - Proxy Join User Domain Response Message Parameters**

The Proxy Join User Domain Response message MUST have a <resInfo> element which MUST have a <rdpProxyJoinUserDomainResponseInformation> element as defined by the following XML schema fragment:

```
<element name="rdpProxyJoinUserDomainResponseInformation">
    <complexType >
        <sequence>
            <element name="domainInfo" type= "roap:DomainInfo" minOccurs="0"/>
            <element name="deviceID" type="gen:Identifier" minOccurs="0"/>
            </sequence>
            <attribute name="hashChainSupport" type="boolean" default="false"/>
    </complexType>
</element>
```

*domainInfo*: this element contains is described below in section 7.4.1.1.

*deviceID*: this element contains the ID of the v2.x Device, copied from the Proxy Join User Domain Request.

*hashChainSupport*: this element, if present, indicates that the DEA will use the hash chain mechanism (see [DRM-DRM-v2.1]) for managing the UDK of the User Domain. This element MUST only be present if the <joinDomainRequest> element> of the Proxy Join User Domain Request message contains the "Hash Chain Support" extension (see [DRM-DRM-v2.1]).

*signature*: this element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

#### 7.4.1.1 <domainInfo> Element

The <domainInfo> element is defined in [DRM-DRM-v2.1] and contains the following child elements: <notAfter> and <domainKey>. These are described below.

*notAfter*: this element expresses, in UTC, the expiry time of the User Domain Context. The value "Infinite" indicates infinite lifetime of the User Domain Context.

*domainKey*: this element contains the following child elements: <encKey>, <riID> and <mac>. These are described below.

*encKey*: this element contains a MAC key, $K_{MAC}$, and the UDK associated with the current generation of the User Domain, and the RI over which the Proxy Join User Domain protocol is performed. The keys are wrapped as specified in the Key management of [DRM-DRM-v2.1], where $K_D$ is replaced by the UDK. The value of the <encKey> element's "Id" attribute MUST be equal to the value of the <domainID> element in the <joinUserDomainRequest> element in the ProxyJoinUserDomainRequest message, save for the Domain Generation part. If the DRM Agent does not support hash chains or the DEA does not use hash chains, then all UDKs for all User Domain Generations MUST be included (including their domain identifiers as Id attributes). If the DEA uses Hash Chains and the DRM Agent supports Hash Chains, only the UDK associated with the latest generation MUST be included. The child of the <ds:KeyInfo> element inside the <encKey> element SHALL be the <roap:X509SPKIHash> element, identifying the public key of the DRM Agent through the hash of the `subjectPublicKeyInfo` in its certificate.

*riID*: this element MUST contain the same value as the <reqID> element in the Proxy Join User Domain Request message.

*mac***:** this element provides key confirmation via a MAC on the canonical version according to [DRM-DRM-v2.1] of the <domainKey> element (excluding the <mac> element itself) using the MAC key $K_{MAC}$ wrapped in the <encKey> element. The MAC algorithm to use is defined by the DEA.

#### 7.4.1.2 Sending JoinDomainResponse

When receiving a Proxy Join User Domain Response message, the RI MUST verify the included signature. If the signature verification fails, the RI MUST send a ROAP-JoinDomainResponse message with only the <status> field, which is set to "DomainAccessDenied". If the *status* attribute in the Proxy Join User Domain Response message contains "Success", and the signature verification did not fail, the RI MUST return a ROAP-JoinDomainResponse to the v2.x DRM Agent to convey the User Domain information in the ProxyJoinUserDomainResponse message. The ROAP-JoinDomainResponse message, defined in [DRM-DRM-v2.1] contains the following fields;

*status*: this value of this attribute MUST be "Success".

*deviceID*: this element MUST contain the same value as the <deviceID> field in the Proxy Join User Domain Response message.

*riID*: this element MUST contain the RI ID from the sending RI.

*nonce*: this element MUST contain the same value as the <nonce> field in the ROAP-JoinDomainRequest message.

*domainInfo*: this element MUST contain the same value as the <domainInfo> element in the associated Proxy Join User Domain Response message.

*certificateChain*: this element MUST be present unless a preceding ROAP-JoinDomainRequest message contained the Peer Key Identifier extension, the extension was not ignored by the RI, and its value identified the RI's current key. When present,

the value of a *Certificate Chain* element SHALL be as described for the *Certificate Chain* element of the ROAP-RegistrationResponse message (see [DRM-DRM-v2.1]).

*ocspResponse*: this element MAY be present. When present, it SHALL contain a complete set of valid OCSP responses for the RI's certificate chain. This element will not be sent if the DRM Agent sent the extension "No OCSP Response" in the preceding JoinDomainRequest (and the RI did not ignore that extension).

*extensions*: if the Proxy Join Request Response message contained a <hashChainSupport> element (as a child element of the <resInfo> element), then the RI must add the “Hash Chain Support” extension (see [DRM-DRM-v2.1]).

*signature*: this element MUST contain the RI signature on this message. The signature is calculated as defined in [DRM-DRM-v2.1].

If the <status> field in the ProxyJoinUserDomainResponse does not contain "Success", but the signature verification over the ProxyJoinUserDomainResponse did not fail, the JoinUserDomainResponse message SHALL only contain the <status> field, which MUST contain the same value as the <status> field in the associated ProxyJoinUserDomainResponse message.

# 7.5 Proxy Leave User Domain Protocol

A v2.x DRM Agent can indirectly leave via the RI a User Domain as indicated by Figure 8. When the v2.x DRM Agent performs this procedure, the DEA deletes the association of this DRM Agent to the User Domain. However, a v2.x DRM Agent may have been joined to this User Domain via multiple RIs. Therefore, after successfully processing a **rdpProxyLeaveUserDomainRequest**, the DEA MUST send an **rdpProxyLeaveUserDomain Trigger** to all RIs that have a valid context for this User Domain related to this v2.x DRM Agent. See section 7.6 for more details. The DEA SHALL NOT allow the v2.x DRM Agent to join the User Domain via any RI as long as it has not performed the Proxy Leave User Domain protocol with the v2.x DRM Agent over all RIs for which the v2.x DRM Agent still has a Domain context associated with this User Domain.

To request leaving a Domain, the v2.x DRM Agent sends the RI a **LeaveDomainRequest** message to convey the information about the target Domain that it will leave. When the RI determines that the target Domain is a User Domain, it sends the DEA a **rdpProxyLeaveUserDomainRequest** message to forward the v2.x DRM Agent request as indicated by the preceding **LeaveDomainRequest** message. After processing the **rdpProxyLeaveUserDomainRequest** message, the DEA returns a **rdpProxyLeaveUserDomainResponse** to the RI to convey its reaction to the request. The RI subsequently returns a **LeaveDomainResponse** to the v2.x DRM Agent to forward the reaction indicated by preceding **rdpProxyLeaveUserDomainResponse**.

For more detail of this procedure, please refer to the following two sections. Please note, the **LeaveDomainRequest** and **LeaveDomainResponse** messages have been described in [DRM-DRM-v2.1].

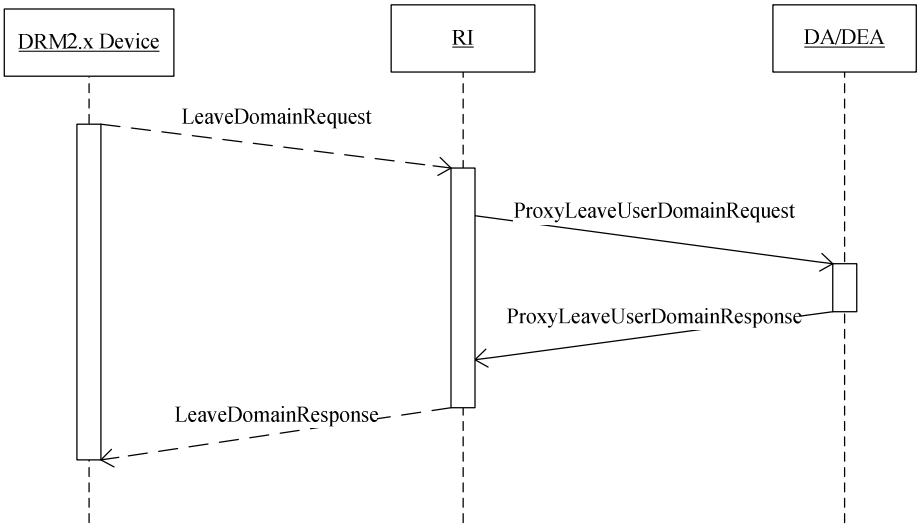


**Figure 8 - v2.x DRM Agent indirectly and partially leaves a User Domain**

## 7.5.1 Proxy Leave User Domain Request

The Proxy Leave User Domain Request message is sent from a RI to a DEA as the first message of the 2-pass Proxy Leave User Domain protocol. The message MUST be a <rdpProxyLeaveUserDomainRequest> element as defined in the following XML schema fragment:

```
<element name="rdpProxyLeaveUserDomainRequest " type="gen:Request"/>
```

A Proxy Leave User Domain Request message MUST be formatted as specified in the table below:

| element / attribute | Usage | value |
|---|---|---|
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| reqInfo | M | Specified below |
| signature | M | Specified below |

**Table 16 - Proxy Leave User Domain Request Message Parameters**

The Proxy Leave User Domain Request message contains a <reqInfo> element which MUST have a <rdpProxyLeaveUserDomainRequestInformation> element as defined by the following XML schema fragment:

```
<element name="rdpProxyLeaveUserDomainRequestInformation">
    <complexType>
        <sequence>
            <element name="leaveDomainRequest" type="base64Binary"/>
            <element name="deviceCertificateChain" type=" roap:CertificateChain " minOccurs="0"/>
        </sequence>
    </complexType>
</element>
```

*leaveDomainRequest*: this field MUST contain the leaveDomainRequest message as received by the RI from the DRM Agent.

*deviceCertificateChain*: if the leaveDomainRequest message does not include the <certificateChain> field, this parameter MUST be present. The value of the <deviceCertificateChain> parameter SHALL be a certificate chain including the Device's certificate. The chain SHALL not include the root certificate. The Device certificate MUST come first in the list. Each following certificate MUST directly certify the one preceding it.

*signature*: This element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ResContext for the DEA.

After reception of the Proxy Leave User Domain Request message, the DEA MUST perform the default processing, as specified in [SCE-GEN], and MUST perform the following checks:

- Verify the validity of the Device's certificate chain. This certificate chain is contained in either the <certificateChain> element in the <leaveDomainRequest> element, or in the <deviceCertificateChain> element in the Proxy Leave User Domain Request message. The certificate chain validation includes verification of the revocation status. If the certificate chain validation fails, the DEA MUST send to the RI a Proxy Leave User Domain Response message with the *status* attribute set to "InvalidCertificateChain".
- The DEA MUST check that the Device certificate includes the `oma-kp-drmAgent` -keypurpose, but does NOT include the `oma-kp-sceDrmAgent` -keypurpose. If this check fails, the DEA MUST send to the RI a Proxy Leave User Domain Response message with the *status* attribute set to "InvalidCertificateChain".
- Verify the signature in the <signature> element in the <leaveDomainRequest> element. If the signature verification fails, the DEA MUST send the RI a ProxyLeaveDomainResponse message with the *status* attribute set to "SignatureError".
- If the UserDomainIdentifier is unknown to the DEA, the DEA MUST send to the RI a Proxy Leave User Domain Response message with *status* attribute set to "InvalidDomain".

When the DEA allows the DRM Agent to leave the User Domain, it MUST return a Proxy Leave User Domain Response message to the RI to convey its reaction to the request. Before sending the Proxy Leave User Domain Response message the

DEA MUST delete the association of this DRM Agent to the User Domain if the DRM Agent is NOT associated with another RI for the same User Domain.

## 7.5.2 Proxy Leave User Domain Response

The Proxy Leave User Domain Response message is sent from the DEA to the RI as the last message of the 2-pass Proxy Leave User Domain Request protocol. The message MUST be a <rdpProxyLeaveUserDomainResponse> element as defined in the following XML schema fragment:

```
<element name="rdpProxyLeaveUserDomainResponse" type="gen:Response"/>
```

If the processing of the Proxy Leave User Domain Request message was not successful, then the response MUST be formatted as specified in [SCE-GEN]. Otherwise the response MUST be formatted per the table below:

| element / attribute | Usage | value |
|---|---|---|
| status | M | "Success" |
| reqID | M | RI ID |
| resID | M | DEA ID |
| nonce | M | Default, as specified in [SCE-GEN] |
| certificateChain | O | Default, as specified in [SCE-GEN] |
| resInfo | M | Specified below |
| signature | M | Specified below |

**Table 17 - Proxy Leave User Domain Response Message Parameters**

The Proxy Join User Domain Response message MUST have a <resInfo> element which MUST have a <rdpProxyLeaveUserDomainResponseInformation> element as defined by the following XML schema fragment:

```
<element name="rdpProxyLeaveUserDomainResponseInformation">
    <complexType>
        <sequence>
            <element name="userDomainID" type="roap:DomainIdentifier"/>
            <element name="deviceID" type="roap:Identifier"/>
        </sequence>
    </complexType>
</element>
```

*userDomainID*: This element contains the identifier of the User Domain being dropped.

*deviceID*: This element contains the ID of the v2.x Device dropping from the User Domain.

*signature*: this element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

### 7.5.2.1 Sending LeaveDomainResponse

When receiving a Proxy Leave User Domain Response message, the RI MUST verify the included signature. If the signature verification fails, the RI MUST discard the message withoud further processing.

After receiving a valid Proxy Leave User Domain Response, the RI MUST return a ROAP-LeaveDomainResponse to the v2.x DRM Agent to forward the reaction from the DEA.

If the signature verification was successful, the parameters in the Proxy Leave User Domain Response message SHALL be set to the following values:

*status*: this field MUST contain the same value as the *status* attribute in the associated Proxy Leave User Domain Response message.

*nonce*: this field MUST contain the same value as the <nonce> field in the LeaveDomainRequest message.

*domainIdentifier*: this field MUST contain the same value as the *Domain Identifier* in the Leave Domain Request message. This MUST match the value of the *domainID* attribute of the *userDomainID* element in the Get User Domain Authorization Request messages corresponding to this User Domain as sent by the RI.

*extensions*: there are currently no extensions defined.

#### 7.5.2.2 Sending LeaveDomainResponse

When receiving a Proxy Leave User Domain Response message, the RI MUST return a ROAP-LeaveDomainResponse to the v2.x DRM Agent to forward the reaction from the DEA.

If the <status> field in the Proxy Leave User Domain Response does not contain "Success", the LeaveDomainResponse message SHALL only contain the <status> field, which MUST be set to the same value as the <status> field in the associated Proxy Leave User Domain Response message.

When receiving a Proxy Leave User Domain Response including the <status> field set to "Success", the RI MUST verify the included DEA signature. If the signature verification fails, the RI MUST send a ROAP LeaveDomainResponse message including only the <status> field, which MUST be set to "DomainAccessDenied". If the signature verification was successful, the parameters in the Proxy Leave User Domain Response message SHALL be set to the following value:

**status**: this field MUST contain the value "Success".

**nonce**: this field MUST contain the same value as the <nonce> field in the LeaveDomainRequest message.

**domainIdentifier**: this field MUST contain the same value as the <UserDomainIdentifier> field in the preceding Proxy Leave User Domain Response message.

**extensions**: there are currently no extensions defined.

## 7.6 DEA Indirectly Triggers a v2.x DRM Agent to Leave a User Domain

A DEA can indirectly trigger a v2.x DRM Agent to leave a User Domain as indicated by Figure 9.

The DEA sends a Proxy Leave User Domain Trigger message to the RI, so that the RI knows to trigger which DRM Agent to leave which User Domain. After some necessary process on the trigger, the RI subsequently sends a ROAP Leave Domain Trigger to the v2.x DRM Agent. Then the v2.x DRM Agent conducts a procedure as indicated by [DRM-DRM-v2.1] to leave indirectly the target User Domain.

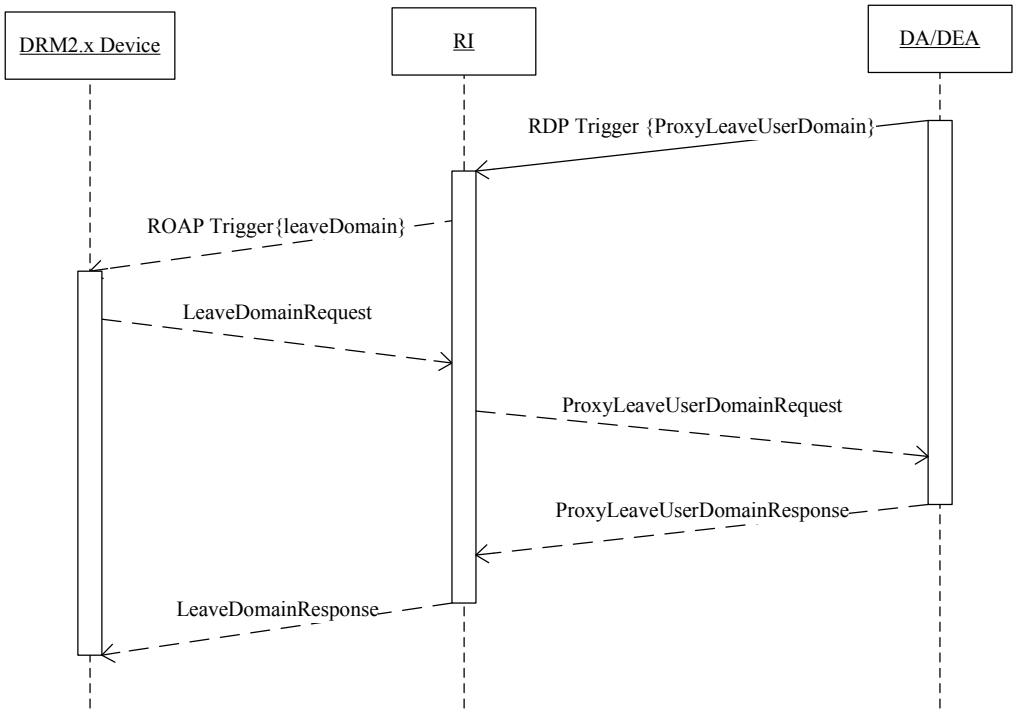


**Figure 9 - DEA indirectly triggers v2.x DRM Agent leave a User Domain**

## 7.6.1 Proxy Leave User Domain Trigger

The trigger message MUST be a <gen:trigger> element as defined by the following XML schema fragment:

```
<element name="drmTrigger" type="gen:DrmTrigger"/>
```

A Proxy Leave User Domain Trigger MUST be formatted as specified in the table below:

| Proxy Leave User Domain Trigger | | |
|---|---|---|
| element / attribute | usage | value |
| id | O | Default, as specified in [SCE-GEN] |
| type | M | “rdpProxyLeaveUserDomain” |
| version | M | “1.0” |
| resID | M | DEA ID |
| resAlias | O | Default, as specified in [SCE-GEN] |
| nonce | M | Default, as specified in [SCE-GEN] |
| reqURL | M | Default, as specified in [SCE-GEN] |
| body | M | Specified below |
| signature | M | Specified below |

**Table 18: Proxy Leave User Domain Trigger**

The Proxy Leave User Domain Trigger contains a <body> element that MUST have a <trgInfo> child element which MUST have a <rdpProxyLeaveUserDomainTrgInformation> element as defined by the following XML schema fragment:

```
<element name="rdpProxyLeaveUserDomainTrgInformation">
    <complexType>
        <sequence>
            <element name="domainID" type= "dom:UserDomainIdentifier"/>
            <element name="deviceID" type="roap:Identifier" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
</element>
```

*domainID*: This element identifies the User Domain, see section 4.1.

*deviceID*: this element contains the Device ID of the v2.x Device that is to leave the User Domain.

*signature*: this element contains a MAC value over the message besides the <signature> element itself. It is made using the negotiated algorithm and using the MK of the ReqContext for the RI.

## 7.6.2 Sending ROAP Leave Domain Trigger

Upon reception of a Proxy Leave User Domain Trigger from the DEA, the RI subsequently sends a ROAP Leave Domain Trigger or ROAP Extended Leave Domain Trigger (see [DRM-DRM-v2.1]) to the v2.x DRM Agent indicated by the <deviceID> element in the Proxy Leave User Domain Trigger to trigger the v2.x DRM Agent to leave the Domain indicated by the <domainID> element in the Proxy Leave User Domain Trigger. When the v2.x DRM Agent receives the ROAP Leave Domain Trigger or ROAP Extended Leave Domain Trigger, it indirectly leaves the target User Domain as specified in [DRM-DRM-v2.1]

# 8. User Domain RO Processing

## 8.1 User Domain RO format

A User Domain RO is formatted as a <protectedRO> element as specified in [DRM-DRM-v2.1] and more specifically as a domainRO. This means that the <ro> element in the <protectedRO> element SHALL contain the <encKey> element, which SHALL have a child <ds:KeyInfo> element, which SHALL have a child <roap:domainID> element. Also the "domainRO"-attribute SHALL be present and set to "true".

When the RI/LRM executed the Get User Domain Authorization protocol, it RI/LRM has chosen a value for the <roap:domainID> element in the <protectedRO> as specified in [DRM-DRM-v2.1]. Note that this value is not equal to the value of the <dom:userDomainId> element that is received from the DEA as part of the Get User Domain Authorization protocol (see section 7.2). A v 2.x DRM Agent, if it initiates the [DRM-DRM-v2.1] joinDomainProtocol upon receipt of the User Domain RO , will use the <roap:domainID> element in the <protectedRO> in the joinDomainRequest call. In this case, the RI will need to use this identifier to determine which UserDomain the [DRM-DRM-v2.1] Agents requests to join and subsequently execute the Proxy Join User Domain Protocol (section 7.4) with the DEA that managed this User Domain.

To show that the RI or LRM is authorized to create (or import) ROs for the User Domain, the RI/LRM MUST include a <party> element in the <agreement> element. The <party> element MUST contain a <context> element with a <userDomainAuthorization> element (see [SCE-REL]). The <userDomainAuthorization> MUST equal the "RI/LRM" <userDomainAuthorization> element that was signed by the DEA and received via the Get User Domain Authorization protocol (see section 7.2).

## 8.2 Installing a User Domain RO that is received from an RI

To render the protected media objects inside a DCF the DRM Agent MUST install the associated User Domain RO. To install a User Domain RO that is received from an RI (out-of-band or using ROAP), the DRM Agents MUST execute the following procedure:

- The DRM Agent MUST ensure that it is a valid member of the User Domain to which the User Domain RO is bound, as specified in 8.2.1
- The DRM Agent MUST ensure that the User Domain RO is valid, as specified in 8.2.2
- The DRM Agent SHOULD perform the post-processing as specified in 8.2.3.

### 8.2.1 Ensuring User Domain membership

To ensure that the DRM Agent is a member of the User Domain to which the User Domain RO is bound, it needs to determine if it has a valid DEA Context with the DEA that manages the User Domain. The DRM Agent MUST compare the value of the User Domains ROs **<dadeaID>** element (child of <userDomainAuthorization> element in the <party> element) with the DEA Identifiers in all valid DEA Contexts stored in the DRM Agent. If the value of the <dadeaID> element does not match that of a DEA Identifier in a valid DEA Context, the DRM Agent SHALL NOT install the User Domain RO. In this case the DRM Agent MAY keep the User Domain RO and MAY send an HTTP GET to the URL specified in the <dadeaURL> element in the <userDomainAuthorization>. An HTTP GET on this URL SHOULD return either a JoinUserDomain Trigger or a (X)HTML page that starts an interaction with the User which may eventually lead to a JoinUserDomain Trigger. It should be noted that in the event that a JoinUserDomain Trigger is returned and the DRM Agent does not have a valid DEA context then the DRM Agent MUST automatically register with the DEA prior to sending a JoinUserDomainRequest message.

Next, the DRM Agent MUST compare the <userDomainId> element within the User Domain RO with the User Domain identifiers for any valid User Domain Contexts already established with the DEA, as identified by the <dadeaId> element. There are four possible outcomes of this comparison:

1. The <userDomainId> element matches a User Domain identifier in a valid User Domain Context already established with the DEA. The DRM Agent MAY install the User Domain RO, provided that verification of User Domain RO validity as specified in section 8.2.2 concludes successfully.

2. The User Domain baseID of the <userDomainId> element matches the User Domain baseID of a stored User Domain identifier in a valid User Domain Context already established with the DEA, but the User Domain Generation of the RO is greater than the User Domain Generation of the stored User Domain ID. The DRM Agent MAY attempt to upgrade the User Domain by sending a JoinUserDomainRequest message to the dadeaURL in the DEA Context associated with the User Domain Context. If the User Domain upgrade is successful, the DRM Agent MAY install the User Domain RO, provided that verification of User Domain RO validity as specified in section 8.2.2 concludes successfully. Otherwise the DRM Agent SHALL NOT install the User Domain RO.

3. The User Domain baseID of the <userDomainId> element matches the User Domain baseID of a stored User Domain identifier in a valid User Domain Context already established with the DEA, but the User Domain Generation of the RO is less than the User Domain Generation of the stored User Domain ID. As discussed below in section 8.2.2, the DRM Agent SHALL NOT install the User Domain RO if it cannot acquire a User Domain Authorization for the RI/LRM such that its User Domain Generation matches the User Domain Generation of the stored User Domain ID.

4. The User Domain baseID of the <userDomainId> field does not match a User Domain baseID in any valid User Domain Context already established with the DEA. The DRM Agent MAY attempt to join the User Domain by sending an HTTP GET request to the URL specified in the <dadeaURL>.

At the point where the DRM Agent sends an HTTP GET request to the URL specified in the <deaURL> element 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 DRM Agent specific means that is outside the scope of this specification or as a direct result of responding to a subsequent DRM Trigger. As a result of an HTTP GET to this URL the DEA can choose (using its own criteria) whether to allow the DRM Agent to join the User Domain or not and SHOULD return either a JoinUser Domain ROAP Trigger or a (X)HTML page that starts an interaction with the User which may eventually lead to a JoinUser Domain ROAP Trigger.

## 8.2.2 Ensuring User Domain RO validity

If the User Domain RO is received out-of-band, then the DRM Agent MUST NOT install the User Domain RO if it includes a <userDomain> constraint.

To further ensure the validity of the User Domain RO the DRM Agent MUST have a valid User Domain Authorization that proves that the RI/LRM is authorized by the DEA to create ROs for the User Domain. A valid User Domain Authorization will be present in the User Domain RO, but it may also have expired or be otherwise invalid. In this case a valid User Domain Authorization may be acquired from the DEA through execution of the dmpJoinDomain protocol.

The DRM Agent MUST check that it has a RI/LRM User Domain Authorization for which:

1. The <userDomainId> element equals a User Domain Identifier in a valid DEA Context as described in section 8.2.1. (e.g. equal values for <dadeaID> and <userDomainId>, both User Domain baseID and User Domain Generation parts)

   (In other words: RI/LRM's User Domain Authorization is of the same User Domain Generation as the most current one in the DRM Agent)

2. This User Domain Authorization element contains

   a. A <deaID> element that equals the <deaID> in the <user Domain Authorization> element in the <party> element in the User Domain RO

   b. A <userDomainId> element of which the User Domain baseID equals the User Domain baseID of the <userDomainId> element in the <User Domain Authorization> element in the User Domain RO. If the UserDomainRO is received using ROAP, then also the User Domain Generation Parts MUST be equal.

(In other words: If received out-of-band, then the User Domain RO may be created for a different generation of the User Domain than the most recent)

3. The <entityId> element equals the <riID> element in the User Domain RO
4. An <isRIorLRM> element is present.
5. If the <notBefore> element is present, the Current DRM Time is later than the value of the <notBefore > element
6. If the <notAfter> element is present, the Current DRM Time is earlier than the value of the <notAfter> element
7. The signature verification using the DEA's Public Key is successful.

If such <user Domain Authorization> exists, than the DRM Agent MUST verify the signature of the User Domain RO using the RI/LRM public key. Also, the DRM Agent MUST successfully verify the MAC (using the <mac> element of the roap:ProtectedRO).

If any of these verifications fails the DRM Agent SHALL NOT install the User Domain RO. In this case the DRM Agent MAY request a new Rights Object by sending a HTTP GET to the RightsIssuerURL in the relevant DCF.

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

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

### 8.2.3 User Domain RO post-processing

There are cases where a DRM Agent installs a User Domain RO that it received separately from the DCF to which it refers. In these cases, the DRM Agent SHOULD insert a copy of the User Domain RO into the corresponding DCF as soon as possible after installation.

In the case where the User Domain RO is received within a DCF, if the DRM Agent cannot verify the signature of the User Domain RO, the DRM Agent MAY leave the User Domain RO as is within the DCF. The DRM Agent MAY request a valid RO for the DCF as described in [DRM-DRM-v2.1].

The DRM Agent MAY insert the User 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 DRM Agent. The DRM Agent MAY insert more than one User Domain RO into a single DCF, as long as all of the inserted ROs are valid and correspond to a User Domain that it is a member of.

When the DRM Agent inserts a User Domain RO into a DCF, it SHOULD remove from the DCF all User Domain ROs corresponding to User Domains that the DRM Agent is not a member of.

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

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

## 8.3 User Domain Upgrade

As is possible in [DRM-DRM-v2.1], a DEA may upgrade a User Domain if, for example, a UDK has been compromised or if a DRM Agent in the User Domain has been compromised or an RI or an LRM in the User Domain has been compromised. This may be necessary to stop DRM Content from leaking out of the system in the clear. In order to upgrade a User Domain, a DEA MUST change the UDK and MUST increment the Domain Generation by one. If the Domain Generation value reaches 999 the Domain becomes obsolete. An RI/LRM MUST NOT issue ROs for an obsolete User Domain. A User Domain upgrade does not result in any Domain Context being deleted in any DRM Agent. After an upgrade, User Domain ROs issued before the upgrade may still be used and shared. This applies to all Devices (revoked and unrevoked) previously in the User Domain, and to any new Devices added to the User Domain after the upgrade.A DEA performs a User Domain

upgrade using the 2-pass JoinUserDomain protocol with JoinUserDomain Trigger. A DEA MAY initiate this protocol for the purposes of User Domain upgrade by sending a JoinUserDomain trigger to a DRM Agent whose Domain membership it wishes to upgrade. If a DRM Agent receives a JoinUserDomain trigger, it compares the <userDomainID> field in the trigger with the domain ID in the DRM Agent for any User Domains already established with the DEA that sent the JoinUserDomain trigger, with the sending DEA as identified by the <resID> field. There are two possible outcomes of this comparison:

1. If the Domain baseID of the <userDomainID> field matches the Domain baseID of a stored domain ID, then the DRM Agent compares the value of the Domain Generation in the trigger with the value of the Domain Generation in the DRM Agent. If the value of the Domain Generation in the trigger is greater than the value stored in the DRM Agent, then the DRM Agent stores all UDKs (of this User Domain) which are included in JoinUserDomainResponse. If the value of the Domain Generation in the trigger is smaller than or equal to the value stored in the DRM Agent, then the DRM Agent ignores the trigger. If Hash Chains are supported by both the DRM Agent and the DEA, the DEA SHALL insert only the latest UDK into the JoinUserDomainResponse. The incoming trigger represents a User Domain upgrade, as described in this section. The DRM Agent silently upgrades the User Domain using the 2-pass JoinUserDomain protocol with JoinUserDomain Trigger.
2. If the Domain baseID of the <userDomainID> field does not match the Domain baseID of a stored domain ID, then the DRM Agent is not a member of the User Domain. The DRM Agent MUST execute the 2-pass JoinUserDomain protocol (See section 5.1.2).

As an alternative method for User Domain upgrade, the DEA MAY execute the 1-pass JoinUserDomain protocol to all SCE members of the domain that are still trusted. If a DRM Agent receives a JoinUserDomainResponse, it compares the <userDomainID> field in the JoinUserDomainResponse with the domain ID in the DRM Agent for any domains already established with the DEA. A comparison procedure of the <userDomainID> field is as follows:

1. If the Domain baseID of the <userDomainID> field matches the Domain baseID of a stored domain ID, then the DRM Agent compares the value of the Domain Generation in the JoinUserDomainResponse with the value of the Domain Generation in the DRM Agent. If the value of the Domain Generation in the JoinUserDomainResponse is greater than the value stored in the DRM Agent, then the DRM Agent stores all UDKs (of this User Domain) which are included in JoinUserDomainResponse. If the value of the Domain Generation in the JoinUserDomainResponse is smaller than or equal to the value stored in the DRM Agent, then the DRM Agent ignores the JoinUserDomainResponse. If Hash Chains are supported by both the DRM Agent and the DEA, the DEA SHALL insert only the latest UDK into the JoinUserDomainResponse.
2. If the Domain baseID of the <userDomainID> field does not match the Domain baseID of a stored domain ID, then the DRM Agent is not a member of the User Domain.

Because the domain generation number is part of the User Domain ID, RI/LRMs may need to re-execute the GetDomainAuthorization protocol to receive User Domain Authotizations that include the most recent generation. DEA MAY initiate the GetDomainAuthorization protocol by sending a Get User Domain Authorization Trigger to RI/LRM whose Domain membership it wishes to upgrade.

When a User Domain is upgraded, for the purpose of maintaining consistent Domain information, an RI that supports Proxy Join User Domain SHOULD send a roap:joinDomain trigger to each 2.x DRM Agent who has joined this User Domain via this RI to trigger that 2.x DRM Agent to re-execute the Join Domain protocol.

## 8.4 Use of hash chains for User Domain key management

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

Example1. Without hash chains

When generating a new User Domain, the DEA generates:

- A unique User Domain Identifier UDI, the User Domain Generation is set to 000.

- A random secret User Domain Key $UDK_0$

At User Domain upgrade the Domain Generation g is increased by 1, which is reflected in the User Domain Identifier, and a new User Domain Key UDKg is generated. The old UDK (s) must be stored in DEA and DRM Agent to allow use of ROs issued before the upgrade. When Devices join a User Domain, all UDKs of this User Domain are sent in the <rspInfo> element of JoinUserDomainResponse (see 6.2.3).

Example 2. With Hash Chains (optional)

When generating a new User Domain, the DEA

- Generates a unique User Domain Identifier UDI, the User Domain Generation is set to 000
- Generates an initial User Domain Key UDK for the User Domain
- Selects the maximum number of generations n for this User Domain (not larger than 999)
- Defines a sequence of UDKs using the method described in [DRM-DRM-v2.1].

Since old UDKs (with low generation value) are possible to efficiently derive from new UDKs (with higher generation value), it is only necessary to store the newest User Domain Key in the DRM Agent (and corresponding User Domain Identifier so the User Domain Generation is known). For the DEA it is sufficient to store UDK, n and the current User Domain Identifier.

# 9. User Domain related transfer operations (informative)

[DRM-DRM-v2.1] allows Domain ROs to be embedded into a (P)DCF and exchanged freely between DRM Agents, using any mechanism or protocol. This enabler builds on this functionality and adds mechanisms that provide more fine-grained control over the exchange of content in relation to a User Domain. Rights Issuers are enabled to limit the number of copies that are usable in the User Domain and the system will enforce that the limited number of usable copies are securely Moved between Devices in the User Domain.

The free exchange of Protected Rights Objects as enabled by [DRM-DRM-v2.1] is also possible in SCE. However, not all User Domain ROs can be used simultaneously on all Devices in the User Domain. Therefore Rights Issuers will explicitly express content exchange related rights in the Rights Object. This section clarifies the semantics of content exchange related elements of the [SCE-REL] for User Domain ROs. The mechanisms that must be employed to enforce these rights are specified
(in [SCE-A2A] for direct (Device-to-Device) Move and Copy of <userDomain>-constrained ROs utilizing the Move RO transaction and Copy RO operation, respectively, and in [SCE-DRM] for RI-assisted Move utilizing the Move <userDomain>-constrained RO Protocol).

This section is NOT concerned with limited sharing of Rights with recipient Devices that are not required to join the specific User Domain in order to consume. Discussion of such operations (e.g., Lend RO and Share RO) is found in [SCE-A2A].

## 9.1 Out-of-band delivery to Devices in a User Domain

If a User Domain RO does not have a top-level <userDomain> constraint, it can be delivered out-of-band and used by any SCE Device that is member of the User Domain to which the RO was issued (where SCE Devices join the User Domain via the DEA that manages that User Domain). Such a User Domain RO can also be used by any v2.x DRM Agent that is a member of the User Domain by way of joining the User Domain via the specific Rights Issuer (or LRM with **`oma-kp-rightsIssuer`** key purpose) that generated the <signature> element over the <rights> element of the RO (where a v2.x DRM Agent may be a member of the User Domain through multiple RIs and/or LRMs).

An SCE DRM Agent will NOT accept out-of-band delivery of ROs that have a top-level <userDomain> constraint (see section 8), and a v2.x DRM Agent will reject top-level <userDomain>-constrained ROs regardless of the delivery mechanism. A User Domain RO that allows the same usage as an OMA DRM v2.x DomainRO, does not have any <copy> or <move> permission and does not have any <userDomain> constraint, so that out-of-band delivery is allowed.

## 9.2 Move between Devices in a User Domain

The <move> element grants permission to transfer the User Domain RO to another DRM Agent. If the “allowPartial” attribute is false, then this transfer is performed in such a way that after the transfer process the User Domain RO and all its related state information is usable by the recipient DRM Agent and is no longer usable by the source DRM Agent. If the “allowPartial” attribute is true, then after the transfer process the User Domain RO may be usable on the source a as well as the sink DRM Agent, but the total amount of state information that is available on source and sink remains constant. In other words, “part” of the state information may be transferred and the remaining portion of the state information may be retained.

A <count> element contained in a <constraint> child element to <move> is used, if present, to specify the number of times the <move> permission may be granted.

If a <move> permission is included in a User Domain RO, also a <userDomain> element contained in a top-level <constraint> is included. This is used to signal that the DRM Agent is only permitted to make a User Domain RO available to other DRM Agents that are (or become) members of the same User Domain. Note that the top-level <userDomain> constraint prevents existing OMA DRM v2.x implementations from using this User Domain RO.

Note that a <move> element in a User Domain RO may have an associated <system> constraint that limits the allowable Move protocols to those which are explicitly identified.

## 9.3 Copy to Devices in a User Domain

The <copy> element, if included in a <userDomain>-constrained User Domain RO, grants permission to transfer a User Domain RO to another DRM Agent, in such a way that after the transfer process the User Domain RO is usable on the recipient DRM Agent and also on the source DRM Agent.

A <count> element contained in a <constraint> child element to <copy> is used, if present, to specify the number of times the <copy> permission may be granted. The count is decremented in the source Device upon successful completion of a Copy operation. The Copy operation results in the source Device will keeping any remaining copies, while the recipient DRM Agent which, receives a copy of the User Domain RO will not be able to copy the User Domain RO any further.

If a <copy> permission is included in a User Domain RO, also a <userDomain> element contained in a top-level <constraint> is included. This is used to specify that the DRM Agent is only permitted to make the User Domain ROs available to other DRM Agents that are (or become) members of the same User Domain. Note that the top-level <userDomain> constraint prevents existing OMA DRM v2.x implementations from using this User Domain RO.

## 9.4 Creation of User Domain ROs (normative)

User Domain ROs MAY be created by the following entities:

1. Rights Issuers: entities that have the **`oma-kp-rightsIssuer`** key purpose, but do not have the **`oma-kp-localRightsManagerDevice`** key purpose and do not have the **`oma-kp-localRightsManagerDomain`** key purpose.
2. LRMs that have at least the **`oma-kp-localRightsManagerDomain`** key purpose.

An LRM MAY be deployed in a device that implements an Import function that creates User Domain ROs for content derived from Non-OMA DRM-protected content [SCE-LRM]..

Notice that LRMs that do not have the **`oma-kp-localRightsManagerDomain`** key purpose, but that do have the **`oma-kp-localRightsManagerDevice`** key purpose, MUST NOT create User Domain ROs, even if they also have the **`oma-kp-rightsIssuer key purpose`**.In all cases, the RI/LRM function in the importing device MUST execute the SCE-3-RDP protocol with the DEA managing the User Domain and establish a User Domain Context.

If the RI or LRM creates a User Domain RO that includes a <userDomain> constraint, then it MUST execute the RO Acquisition protocol [SCE-DRM] (over the SCE-1-ROAP /SCE-6-LRMP interface, respectively) to securely deliver this User Domain RO to an SCE DRM Agent. The SCE DRM Agent MUST be a current member of the User Domain as a condition of the RI or LRM creating a <userDomain>-constrained RO for that Device. Move of a <userDomain>-constrained RO by an RI (using the Move <userDomain>-constrained RO Protocol [SCE-DRM]) does not require the recipient Device to be a member of the User Domain in order to receive the RO.

Registration of SCE DRM Agents and of v2.x DRM Agents with an LRM over the SCE-6-LRMP interface is discussed in [SCE-LRM].

User Domain ROs without a <UserDomain> constraint MAY be delivered to DRM Agents via an out-of-band protocol.

# 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 1.0 History

| Document Identifier | Date | Sections | Description |
|---|---|---|---|
| Draft Versions<br>OMA-TS-SCE_DOM-V0_1-20070620-D | 20 Jun 2007 | | Initial version per 2007-0284 |
| OMA-TS-SCE_DOM-V0_2-20070924-D | 24 Sep 2007 | 6.2<br>8<br>All | Implemented CR 2007-398R01<br>Implemented CR 2007-401<br>Implemented CR 2007-300R03<br>Update of all internal cross-references |
| OMA-TS-SCE_DOM-V0_3-20071025-D | 25 Oct 2007 | 7.<br>1.<br>6.2.4. | Implemented CR 2007-0448R03<br>Implemented CR 2007-0449R01<br>Implemented CR 2007-0488R01 |
| OMA-TS-SCE_DOM-V0_4-20071109-D | 9 Nov 2007 | 8.2 | Implemented CR 2007-0485 |
| OMA-TS-SCE_DOM-V0_5-20080214-D | 14 Feb 2008 | | Implemented:<br>2007-0533R03<br>2007-0537<br>2007-0538<br>2008-0021R02<br>2008-0022R03 |
| OMA-TS-SCE_DOM-V0_6-20080314-D | 14 Mar 2008 | | Implemented:<br>2008-0016<br>2008-0029R01<br>2008-0030R05<br>2008-0038R01<br>2008-0050R02<br>2008-0051R02<br>2008-0052R01<br>2008-0073R01<br>2008-0075R02<br>2008-0089R01 |
| OMA-TS-SCE_DOM-V0_7-20080320-D | 20 Mar 2008 | | Implemented<br>2007-0326R02 |
| OMA-TS-SCE_DOM-V0_8-20080325-D | 25 Mar 2008 | | Implemented:<br>2008-108 – CR was not agreed – but it was agreed to put it into the draft |
| OMA-TS-SCE_DOM-V0_8_1-20080611-D | 25 Mar 2008 | | Implemented:<br>2008-0155R04 – Addr DOM 069, 070,124, 125<br>2008-0156R04 – Addr. DOM 131<br>2008-0157R01 – Addr. DOM 119<br>2008-0159 – Addr. DOM 133<br>2008-0191R01– Addr. DOM 136<br>2008-0228R01 – Addr. DOM 128 |
| OMA-TS-SCE_DOM-V1_0-20080922-D | 22 Sep 2008 | | 2008-0334R04 |
| OMA-TS-SCE_DOM-V1_0-20080923-D | 23 Sep 2008 | | 2008-0387<br>2008-0386R01 (partially) |
| OMA-TS-SCE_DOM-V1_0-20080926-D | 25 Sep 2008 | | Implemented:<br>2008-404R02, 2008-405R02, 2008-0407, 2008-409R01, 2008-410R01, 2008-0420 – XML Schema |

| Document Identifier | Date | Sections | Description |
|---|---|---|---|
| OMA-TS-SCE_DOM-V1_0-20081031-D | 31 Oct 2008 | | 2008-0147R02, 2008-0158R03, 2008-0189R02, 2008-0296R02, 2008-0297R02, 2008-0357, 2008-0369R01, 2008-0412R02, 2008-0426R02, 2008-0442R01, 2008-0450R01, 2008-0451R01 |
| OMA-TS-SCE_DOM-V1_0-20081103-D | 3 Nov 2008 | | 2008-0300R01, 2008-0386R01 (remainder) |
| OMA-TS-SCE_DOM-V1_0-20081105-D | 5 Nov 2008 | | Editorials, update registered key purpose object identifiers |
| OMA-TS-SCE_DOM-V1_0-20081114-D | 14 Nov 2008 | | Editorials |
| Candidate Version<br>OMA-TS-SCE_DOM-V1_0-20081209-C | 09 Dec 2008 | n/a | Status changed to Candidate by TP<br>TP ref# OMA-TP-2008-0475-INP_SCE_V1_0_ERP_for_Candidate_Approval |
| Draft Version<br>OMA-TS-SCE_DOM-V1_0-20090126-D | 26 Jan 2009 | n/a | Status changed to Draft with the following change requests:<br>OMA-DRM-2008-0528-CR_SCE_DOM_Add_Definitions<br>OMA-DRM-2008-0532-CR_SCE_DOM_Key_purpose_and_user_domain_creation<br>OMA-DRM-2008-0538R01-CR_DA_Signed_DEA_User_Domain_Authorization |
| Draft Version<br>OMA-TS-SCE_DOM-V1_0-20090511-D | 11 May 2009 | n/a | Incorporated the following agreed CR:<br>OMA-DRM-2009-0071R02-CR_Resolving_Inconsistencies_within_SCE_DOM.doc |
| Candidate Version<br>OMA-TS-SCE_DOM-V1_0-20090526-C | 26 May 2009 | n/a | Status changed to Candidate by TP<br>TP ref# OMA-TP-2009-0228-INP_SCE_V1_0_ERP_for_Notification |

# Appendix B. Static Conformance Requirements (Normative)

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

## B.1 SCR for DRM Agent

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-DRMAGENT-C-001-O | Acquiring User Domain ROs | 4.2 | SCE-DOM-DRMAGENT-C-002-O AND SCE-DOM-DRMAGENT-C-003-O AND SCE-DOM-DRMAGENT-C-004-O AND SCE-DOM-DRMAGENT-C-007-O AND SCE-DOM-DRMAGENT-C-008-O AND SCE-DOM-DRMAGENT-C-009-O AND SCE-DOM-DRMAGENT-C-010-O AND SCE-DOM-DRMAGENT-C-011-O |
| SCE-DOM-DRMAGENT-C-002-O | Registration between DRM Agent and DEA | 5.1, 6.1 | |
| SCE-DOM-DRMAGENT-C-003-O | Join User Domain protocol | 6.2 | |
| SCE-DOM-DRMAGENT-C-004-O | Leave User Domain protocol | 6.3 | |
| SCE-DOM-DRMAGENT-C-007-O | User Domain RO format | 8.1 | |
| SCE-DOM-DRMAGENT-C-008-O | Installing a User Domain RO that is received from an RI | 8.2 | |
| SCE-DOM-DRMAGENT-C-009-O | Ensuring User Domain Membership | 8.2.1 | |
| SCE-DOM-DRMAGENT-C-010-O | Ensuring User Domain validity | 8.2.2 | |
| SCE-DOM-DRMAGENT-C-011-O | User Domain RO post-processing | 8.2.3 | |
| SCE-DOM-DRMAGENT-C-012-O | Hash chains for UDK | 8.4 | |

## B.2 SCR for DEA

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-DEA-S-001-M | User Domain Authorization | 4.1 | |
| SCE-DOM-DEA-S-002-M | Registration between DRM Agent and-DEA | 5.1, 5.2 | |
| SCE-DOM-DEA-S-003-M | Join User Domain protocol | 6.2 | |

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-DEA-S-004-M | Leave User Domain protocol | 6.3 | |
| SCE-DOM-DEA-S-005-M | Registration between RI and DEA | 7.1 | |
| SCE-DOM-DEA-S-006-M | Get User Domain Authorization Protocol | 7.2 | |
| SCE-DOM-DEA-S-007-M | Drop User Domain Authorization Protocol | 7.3 | |
| SCE-DOM-DEA-S-008-O | Proxy Join User Domain Protocol | 7.4 | SCE-DOM-DEA-S-009-O |
| SCE-DOM-DEA-S-009-O | Proxy Leave User Domain Protocol | 7.5 | SCE-DOM-DEA-S-010-O |
| SCE-DOM-DEA-S-010-O | Proxy Leave User Domain Trigger | 7.6.1 | |
| SCE-DOM-DEA-S-011-M | User Domain Upgrade | 8.3 | |
| SCE-DOM-DEA-S-012-O | Hash chains for UDK | 8.4 | |

## B.3 SCR for RI

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-RI-S-001-O | User Domain Authorization | 4.1 | SCE-DOM-RI-S-002-O<br>AND<br>SCE-DOM-RI-S-003-O<br>AND<br>SCE-DOM-RI-S-004-O<br>AND<br>SCE-DOM-RI-S-008-O<br>AND<br>SCE-DOM-RI-S-009-O |
| SCE-DOM-RI-S-002-O | Registration between RI and DEA | 7.1 | |
| SCE-DOM-RI-S-003-O | Get User Domain Authorization Protocol | 7.2 | |
| SCE-DOM-RI-S-004-O | Drop User Domain Authorization Protocol | 7.3 | |
| SCE-DRM-RI-S-005-O | Proxy Join User Domain Protocol | 7.4 | SCE-DOM-RI-S-006-O |
| SCE-DRM-RI-S-006-O | Proxy Leave User Domain Protocol | 7.5 | SCE-DOM-RI-S-007-O |
| SCE-DRM-RI-S-007-O | Proxy Leave User Domain Trigger | 7.6.1 | |
| SCE-DRM-RI-S-008-O | User Domain RO Format | 8.1 | |
| SCE-DRM-RI-S-009-O | User Domain Upgrade | 8.3 | |

## B.4 SCR for LRM with LRM-Domain key purpose only

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-LRMDOM-S-001-O | User Domain Authorization | 4.1 | SCE-DOM-LRMDOM-S-002-O |

| Item | Function | Reference | Requirement |
|---|---|---|---|
| | | | AND<br>SCE-DOM-LRMDOM-S-003-O<br>AND<br>SCE-DOM-LRMDOM-S-004-O<br>AND<br>SCE-DOM-LRMDOM-S-008-O<br>AND<br>SCE-DOM-LRMDOM-S-009-O |
| SCE-DOM-LRMDOM-S-002-O | Registration between RI and DEA | 7.1 | |
| SCE-DOM-LRMDOM-S-003-O | Get User Domain Authorization Protocol | 7.2 | |
| SCE-DOM-LRMDOM-S-004-O | Drop User Domain Authorization Protocol | 7.3 | |
| SCE-DRM-LRMDOM-S-007-O | Proxy Leave User Domain Trigger | 7.6.1 | |
| SCE-DRM-LRMDOM-S-008-O | User Domain RO Format | 8.1 | |
| SCE-DRM-LRMDOM-S-009-O | User Domain Upgrade | 8.3 | |

# B.5 SCR for LRM with LRM-Domain key purpose and LRM-Device key purpose, but without RI key purpose

The SCR table for an LRM with an LRM-Domain key purpose and LRM-Device key purpose, but without RI key purpose, is equal to the SCR table for an LRM with LRM-Domain key purpose only in section B.4.

# B.6 SCR for LRM with LRM-Domain key purpose and RI key purpose

| Item | Function | Reference | Requirement |
|---|---|---|---|
| SCE-DOM-LRMDOM/RI-S-001-O | User Domain Authorization | 4.1 | SCE-DOM-LRMDOM/RI-S-002-O<br>AND<br>SCE-DOM-LRMDOM/RI-S-003-O<br>AND<br>SCE-DOM-LRMDOM/RI-S-004-O<br>AND<br>SCE-DOM-LRMDOM/RI-S-008-O<br>AND<br>SCE-DOM-LRMDOM/RI-S-009-O |
| SCE-DOM-LRMDOM/RI-S-002-O | Registration between RI and DEA | 7.1 | |
| SCE-DOM-LRMDOM/RI-S-003-O | Get User Domain Authorization Protocol | 7.2 | |
| SCE-DOM-LRMDOM/RI-S-004-O | Drop User Domain Authorization Protocol | 7.3 | |
| SCE-DRM-LRMDOM/RI-S-005-O | Proxy Join User Domain Protocol | 7.4 | SCE-DOM-LRMDOM/RI-S-006-O |
| SCE-DRM- | Proxy Leave User | 7.5 | SCE-DOM-LRMDOM/RI-S-007-O |

| Item | Function | Reference | Requirement |
|---|---|---|---|
| LRMDOM/RI-S-006-O | Domain Protocol | | |
| SCE-DRM-LRMDOM/RI-S-007-O | Proxy Leave User Domain Trigger | 7.6.1 | |
| SCE-DRM-LRMDOM/RI-S-008-O | User Domain RO Format | 8.1 | |
| SCE-DRM-LRMDOM/RI-S-009-O | User Domain Upgrade | 8.3 | |

## B.7 SCR for LRM with LRM-Domain key purpose, LRM-Device key purpose and RI key purpose

The SCR table for an LRM with an LRM-Domain key purpose and an LRM-Device key purpose is equal to the SCR table for an LRM with LRM-Domain key purpose and RI key purpose in section B.6.

# Appendix C. Certificate Profiles (Normative)

## C.1 Domain Authority Certificates

The profile for DA certificates follows the profile for "X.509-compliant server certificate" in [CertProf] with the following modifications:

| | |
|---|---|
| Signature | MUST be RSA with SHA-1 |
| Serial Number | MUST be less than, or equal to, 20 bytes in length |
| Issuer Name | MUST be present and MUST use a subset of the following naming attributes from [CertProf] – countryName, organizationName, organizationalUnitName, commonName, and stateOrProvinceName. |
| Subject Name | MUST be present and MUST use a subset of the following naming attributes from [CertProf] – countryName, organizationName, organizationalUnitName, commonName, and serialNumber.<br><br>The structure and contents of a DA subject name shall be as follows:<br><br>[countryName=<Country of manufacturer>]<br><br>[organizationName=<Manufacturer company name>]<br><br>[organizationalUnitName=<Manufacturing location>]<br><br>[commonName=<Model name>]<br><br>serialNumber=<Unique identifier for DA, as assigned by the Certificate Issuer.<br><br>The serialNumber attribute MUST be present. The countryName, organizationName, organizationalUnitName, and commonName may be present. Other attributes are not allowed and must not be included. For all naming attributes of type DirectoryString, the PrintableString or the UTF8String choice must be used.<br><br>Note that the maximum length (in octets) for values of these attributes is as follows: countryName – 2 (country code in accordance with ISO/IEC 3166), organizationName, organizationalUnitName, commonName, and serialNumber – 64.<br><br>Example:<br><br>C="US";O="DRM Devices 'R Us"; CN="DRM Device Mark VI"; SN="1234567890" |
| Extensions | The extKeyUsage extension SHALL be present, and contain (at least) the **`oma-kp-domainAuthority`** key purpose object identifier:<br><br>`oma-kp-domainAuthority OBJECT IDENTIFIER ::= {oma-kp 9}`<br><br>CAs MUST set this extension to critical.<br><br>If the keyUsage extension is present (recommended), then the digitalSignature bit shall be set. When present, this extension shall be set to critical.<br><br>CAs MAY include the certificatePolicy extension, indicating the policy the certificate has been issued under, and possibly containing a URI identifying a source of more information about the policy.<br><br>CAs are recommended to not include any other extensions, but may, for compliance |

| | |
|---|---|
| | with [RFC3280], include the authorityKeyIdentifier extension.<br><br>CAs MUST NOT include any other critical extensions. |

SCE DRM Agents processing DA certificates MUST meet the requirements on clients processing "X.509-compliant server certificates" defined in [CertProf]. In addition, SCE DRM Agents:

- MUST be able to process DA certificates up to 1500 bytes long;
- MUST be able to process DA certificates with serial numbers 20 bytes long; and
- MUST recognize the presence of the **`oma-kp-domainAuthority`** object identifier defined above in the extKeyUsage extension in DA certificates. If the extensioin is present, then the SCE DRM Agent MUST consider the subject certified by the certificate to be a DA while processing information received from it.

# C.2 Domain Enforcement Agent Certificates

The profile for DEA certificates follows the profile for "X.509-compliant server certificate" in [CertProf] with the following modifications:

| | |
|---|---|
| Signature | MUST be RSA with SHA-1 |
| Serial Number | MUST be less than, or equal to, 20 bytes in length |
| Issuer Name | MUST be present and MUST use a subset of the following naming attributes from [CertProf] – countryName, organizationName, organizationalUnitName, commonName, and stateOrProvinceName. |
| Subject Name | MUST be present and MUST use a subset of the following naming attributes from [CertProf] – countryName, organizationName, organizationalUnitName, commonName, and serialNumber.<br><br>The structure and contents of a DEA subject name shall be as follows:<br><br>[countryName=<Country of manufacturer>]<br><br>[organizationName=<Manufacturer company name>]<br><br>[organizationalUnitName=<Manufacturing location>]<br><br>[commonName=<Model name>]<br><br>serialNumber=<Unique identifier for Domain Enforcement Agent, as assigned by the Certificate Issuer.<br><br>The serialNumber attribute MUST be present. The countryName, organizationName, organizationalUnitName, and commonName may be present. Other attributes are not allowed and must not be included. For all naming attributes of type DirectoryString, the PrintableString or the UTF8String choice must be used.<br><br>Note that the maximum length (in octets) for values of these attributes is as follows: countryName – 2 (country code in accordance with ISO/IEC 3166), organizationName, organizationalUnitName, commonName, and serialNumber – 64.<br><br>Example:<br><br>C="US";O="DRM Devices 'R Us"; CN="DRM Device Mark VI"; SN="1234567890" |
| Extensions | The extKeyUsage extension SHALL be present, and contain (at least) the **`oma-kp-domainEnforcementAgentLocal`** or the |

| | |
|---|---|
| | **oma-kp-domainEnforcementAgentNetwork** key purpose object identifier:<br><br>`oma-kp-domainEnforcementAgentLocal OBJECT IDENTIFIER ::= {oma-kp 10}`<br><br>`oma-kp-domainEnforcementAgentNetwork OBJECT IDENTIFIER ::= {oma-kp 11}`<br><br>CAs MUST set this extension to critical.<br><br>If the keyUsage extension is present (recommended), then the digitalSignature bit shall be set. When present, this extension shall be set to critical.<br><br>CAs MAY include the certificatePolicy extension, indicating the policy the certificate has been issued under, and possibly containing a URI identifying a source of more information about the policy.<br><br>CAs are recommended to not include any other extensions, but may, for compliance with [RFC3280], include the authorityKeyIdentifier extension.<br><br>CAs MUST NOT include any other critical extensions. |

The **oma-kp-domainEnforcementAgentLocal** indicates that the DEA is an entity in a local location such as a home or office. It is assumed that these types of DEA are owned and managed by a User. The **oma-kp-domainEnforcementAgentNetwork** indicates that the DEA is an entity in a remote location that is accessible via a network such as the Internet. It is assumed that these types of DEA are not owned or managed by a User. A DEA certificate MUST have only one of these key purposes.

SCE DRM Agents processing DEA certificates MUST meet the requirements on clients processing "X.509-compliant server certificates" defined in [CertProf]. In addition, SCE DRM Agents:

- MUST be able to process DEA certificates up to 1500 bytes long;
- MUST be able to process DEA certificates with serial numbers 20 bytes long; and
- MUST recognize the presence of the **oma-kp-domainEnforcementAgentLocal** and **oma-kp-domainEnforcementAgentNetwork** object identifiers defined above in the extKeyUsage extension in DEA certificates. If one of these is present, then the SCE DRM Agent MUST consider the subject certified by the certificate to be a DEA while processing information received from it.