

Information and Communication Technologies (ICT) Programme

Project N°: FP7-ICT- 214063

SEA



D5.1 - SVC/MVC Crypting Framework & Content Management

Author(s): Federico Álvarez, Laura Arnaiz, Lara García
(Universidad P. de Madrid), K. Grüneberg, Th. Schierl
(HHI), Enrico Magli (POLITO), Costis Contopoulos
(VODAFONE)

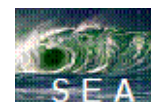
Status -Version: Final

Date: 30 September 2008

Distribution - Confidentiality: Public

Code: SEA_D5.1_UPM_FF_20080930.doc

Abstract: In the present deliverable, the SEA content protection and management system and its architecture are presented. The SEA project aims to provide an end-to-end solution for content protection management for IP and P2P networks, exploiting the full potential of the content protection and creator's rights maintenance. SEA has developed a beyond state-of-the art content protection technology, extending the ISMACryp content protection mechanism in a way that the ciphering technology can be applied to H.264 MVC/SVC encoder and decoder, besides designing and developing new media protection paradigms and solutions for P2P networking using a lightweight asset management approach. The solution proposed is based on the creation of a secure and adaptable content delivery architecture and the underlying mechanisms to ensure the correct content management which along with the content protection mechanisms can be useful for, on the one hand ensure user privacy and, on the other hand, enable the possibility of offering commercial IPTV services over a P2P environment.



Disclaimer

This document contains material, which is the copyright of certain SEA contractors, and may not be reproduced or copied without permission. All SEA consortium partners have agreed to the full publication of this document. The commercial use of any information contained in this document may require a license from the proprietor of that information.

The SEA Consortium consists of the following companies:

No	Participant name	Participant short name	Country	Country
1	ST Microelectronics	STM	Co-ordinator	Italy
2	Synelixis Solutions	Synelixis	Contractor	Greece
3	Thomson Grass Valley	Thomson	Contractor	France
4	Philips Consumer Electronics	Philips	Contractor	Netherlands
5	Vodafone Panafon AEET	Vodafone	Contractor	Greece
6	Nomor Research	Nomor	Contractor	Germany
7	Fraunhofer HHI	HHI	Contractor	Germany
8	Politecnico di Torino	Polito	Contractor	Italy
9	Universidad Politécnica de Madrid	UPM	Contractor	Spain
10	University of California, Los Angeles	UCLA	Contractor	USA

The information in this document is provided “as is” and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.



This page has been intentionally left blank

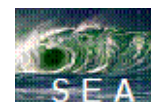


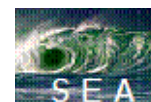
Table of contents

1. Introduction	9
2. Background – Technology Overview	10
2.1. <i>Digital Rights Management</i>	10
2.1.1. Introduction	10
2.1.2. Fundamental DRM Concepts	10
2.1.3. DRM System Architecture and Requirements.....	14
2.1.4. Rights Description Languages	18
2.2. <i>Peer to Peer Networking</i>	23
2.2.1. Basic concepts and applications - Architecture	23
2.2.2. Streaming Content in P2P Networks	24
2.2.3. Security and P2P networks	25
2.2.4. Basic DRM functions for P2P content delivery.....	26
2.2.5. Protecting digital rights of streamed content in P2P Networks	27
3. The SEA Concept	29
3.1. <i>Content protection & authentication of SVC/MVC media</i>	29
3.1.1. Implementation of the SEA crypto framework.....	30
3.1.2. Conditional access system	35
3.2. <i>P2P Content management</i>	38
3.2.1. The proposed architecture for secure and adaptable content delivery	38
3.2.2. Solution implementation.....	41
3.2.3. Interoperability of the system	62
4. Conclusions	68
5. References	69



Table of figures

Figure 1. DRM System Architecture _____	15
Figure 2. DRM System basic services _____	17
Figure 3. ODRL Foundation Model _____	20
Figure 4. Structure of a simple License _____	23
Figure 5. SEA encrypting server _____	29
Figure 6. SEA decrypting client _____	30
Figure 7. RTP payload type 'mpeg4-generic' (RFC 3640) _____	30
Figure 8. 'mpeg4-generic' AU Header and ISMACryp extension _____	31
Figure 9. Server architecture with encrypting proxy _____	32
Figure 10. Connecting different CA systems through SimulCrypt _____	33
Figure 11. Decrypting proxy at receiver side _____	35
Figure 12. Content protection and key management system overview _____	40
Figure 13. DRM System _____	41
Figure 14. Solution implementation for the content protection and key management system _____	42
Figure 15. Messages exchanged between the SCS and the ECMG _____	43
Figure 16. Entitlement Management Message Generator _____	47
Figure 17. Messages exchanged between the Client and the EMMG _____	48
Figure 18. Control Word encryption key selection _____	53
Figure 19. Sub-parameter of the CW_encryption parameter _____	54
Figure 20. Key distribution and decryption process _____	55
Figure 21. Key management system for license encryption _____	55
Figure 22. Principal window for the license creator _____	57
Figure 23. Window to add new free charge users _____	59
Figure 24. License server _____	60
Figure 25. Decrypt license: decrypts and read the license _____	61
Figure 26. Window to select the license (browser form the previous figure) _____	61
Figure 27. XSTL Transformation _____	64

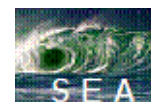


Abbreviations

AAA	<i>Authentication, Authorization and Accounting</i>
AEM	<i>Adaptation Execution Modules</i>
AES	<i>Advanced Encryption Standard</i>
API	<i>Application Programming Interface</i>
AU	<i>Access Unit</i>
B2B	<i>Business-to-Business</i>
CA	<i>Conditional Access</i>
CAS	<i>Conditional Access System</i>
CEK	<i>Content Encryption Key</i>
CP	<i>Crypto Period</i>
CPU	<i>Central Processing Unit</i>
CSM	<i>Content Storage Module</i>
CW	<i>Control Word</i>
CWG	<i>Control Word Generator</i>
DCF	<i>DRM Content Format</i>
DCT	<i>Discrete Cosine Transform</i>
DI	<i>Digital Item</i>
DNS	<i>Domain Name System</i>
DPRL	<i>Digital Property Rights Language</i>
DRM	<i>Digital Rights Management</i>
ECM	<i>Entitlement Control Message</i>
ECMG	<i>Entitlement Control Message Generator</i>
EMM	<i>Entitlement Management Message</i>
EMMG	<i>Entitlement Management Message Generator</i>
GUI	<i>Graphical User Interface</i>
HTTP	<i>Hypertext Transfer Protocol</i>
ICT	<i>Information and Communications Technologies</i>
ID	<i>Identifier</i>
IETF	<i>Internet Engineering Task Force</i>
IP	<i>Internet Protocol</i>
IPMP	<i>Intellectual Property Management & Protection</i>
IPTV	<i>IP Television</i>
ISMA	<i>Internet Streaming Media Alliance</i>
KI	<i>Key Indicator</i>
MDC	<i>Multiple Description Coding</i>
MIME	<i>Multipurpose Internet Mail Extensions</i>
MMS	<i>Multimedia Messaging Service</i>



MPEG	<i>Moving Picture Experts Group</i>
MVC	<i>Multi-view Video Coding</i>
ODRL	<i>Open Digital Rights Language</i>
OMA	<i>Open Mobile Alliance</i>
P2P	<i>Peer-to-Peer</i>
PDA	<i>Personal Digital Assistant</i>
QCIF	<i>Quarter Common Intermediate Format</i>
QoS	<i>Quality of Service</i>
REL	<i>Rights Expression Language</i>
RM	<i>Rights Management</i>
RFC	<i>Request for Comments</i>
RO	<i>Rights Object</i>
RTP	<i>Real Time Protocol</i>
RTSP	<i>Real Time Streaming Protocol</i>
SAP	<i>Session Announcement Protocol</i>
SCS	<i>SimulCrypt Synchronizer</i>
SDK	<i>Software Development Kit</i>
SDP	<i>Session Description Protocol</i>
SEA	<i>SEAmless Content Delivery</i>
sHMG	<i>SEA Home Media Gateway</i>
SIP	<i>Session Initiation Protocol</i>
SK	<i>Service Key</i>
sNMG	<i>SEA Network Media Gateway</i>
SRTP	<i>Secure Real-time Transport Protocol</i>
SVC	<i>Scalable Video Coding</i>
TC	<i>Trust Computing</i>
TCP	<i>Transmission Control Protocol</i>
UID	<i>User Identifier</i>
URL	<i>Uniform Ressource Locator</i>
VGA	<i>Video Graphics Array</i>
VLC	<i>VideoLAN Client</i>
WAP	<i>Wireless Application Protocol</i>
WSP	<i>Wireless Session Protocol</i>
XML	<i>Extensible Markup Language</i>
XrML	<i>Extensible rights markup language</i>
XSL	<i>Extensible Stylesheet Language</i>
XSLT	<i>Extensible Stylesheet Language Transformations</i>



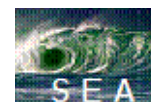
Executive Summary

This document describes the Crypting Framework and Content Management structure, tailored to Scalable Video and Multi-View Coding of the SEA project.

In the present deliverable, the SEA content protection and management system and its architecture are presented. Firstly, an introduction to the state-of-the-art of the base technologies used for the framework and the content management design are presented: The digital rights management parts used (the fundamental DRM concepts, a common system architecture, the rights description languages), the Peer-to-Peer Networking (basic concepts and architecture, content streaming in P2P, the security issues and the basic DRM functions for P2P content delivery).

Secondly we present the SEA end-to-end solution for content protection management for heterogeneous IP and P2P networks, exploiting the full potential of the content protection and creator's rights maintenance. SEA has developed a beyond state-of-the-art content protection technology, extending the ISMACryp content protection mechanism in a way that the ciphering technology can be applied to H.264 MVC/SVC encoder and decoder, besides designing and developing new media protection paradigms and solutions for P2P networking using a lightweight asset management approach. The solution proposed is based on the creation of a secure and adaptable content delivery architecture and the underlying mechanisms to ensure the correct content management which along with the content protection mechanisms can be useful for, on the one hand ensure user privacy and, on the other hand, enable the possibility of offering commercial IPTV services over a P2P environment.

The development of the system has been carried out according to the identified use cases and its compatibility and interoperability ensured.



1. Introduction

SEA aims to offer innovative content protection via a personalized interoperable content protection solution, aiming to target all identified types of networks. SEA will offer a lightweight content management environment to avoid disturbing or overloading the content delivery. The solution proposed is adequate to the P2P heterogeneous network design including mobile and fixed networks along with terminals ranging from mobile or handheld devices to IPTV STBs.

SEA aims to support the separation of rights management system components, which is the decoupling of authentication, licensing, rights management and protection technologies in a distributed environment, focusing on P2P networking technologies. This disintermediation will enable the choice and selection of these technologies independent of each other, without compromising the integrity of the solution, while achieving maximum interoperability with existing solutions. The proposed content protection system will enable multiple instances of these components to exist in a Right Management/Conditional Access System (RM/CAS) system, while facilitating P2P networking. The design philosophy is that clients should be able to negotiate for rights through standardised protocols rather than downloading a license with an embedded expression of rights.

In SEA prototyping, the content encryption is carried out using asymmetric encryption technology. In the event of data loss the impact on the audio-visual quality can be minimized by sending protected stream keys and their associated initialization vectors at frequent intervals. However, the system will be able to accommodate other symmetric encryption technologies in the future. For the protected content and the associated keys/licenses distribution, an open, well specified server will be used. Distribution may be carried out using various different processes, including broadcast (one-to-many) or unicast (one-to-one), and real-time/streaming or download (non-real-time). The licenses may be delivered with the content (in-band) or separately (out-of-band) depending upon the method of delivery.

In the document firstly the background technologies are described in section 2, in order to give the reader a complete perspective of the base technologies used for the content management and protection inside the SEA environment. After this technology presentation, the SEA concept for content management and protection/authentication can be found in section 3. Conclusion can be found in section 4, and references in section 5.



2. Background – Technology Overview

2.1. Digital Rights Management

2.1.1. Introduction

Contemporary Digital Rights Management Systems (DRM) fundamental role is to enable efficiently controllable, robust, scalable and, most importantly, secure distribution of digital content created by a vast variety of sources towards a large audience of interested recipients. To this end, DRM augments procedures related to actual content distribution with procedures for, on the one hand, content encryption, so as to diminish unauthorised access during distribution and advanced control during actual content production or rendering at the client side, i.e. application of conditional access on the content, according to a list of access rights expressed by the owner, on the other. In the following a brief description of the most common architectural and functional components of contemporary DRM systems is provided. Prospective readers may refer to [1]-[5] for a more elaborate analysis.

2.1.2. Fundamental DRM Concepts

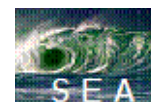
In principle, the functional entities that constitute a DRM system range from content owners, content developers and content distributors, DRM and billing service providers, to network equipment vendors and device manufactures, and finally consumers of content. These entities implement specific roles in a DRM system. It has to be noted that the functional entities are logical and they not necessarily represent physical network nodes (located in the core or at the networks' edge). Depending on configuration, different functional entities may be implemented by the same or different physical nodes, and be operated by the same or different actors.

DRM is a technology that enables owners of digital content¹ to control access on the content and also restrict its usage in various ways specified by the owners or specific delegates. The term digital content in general refers to audio, video, images, text, other binary data elements and any combinations of these in a digitally represented format. These constructs may be in the form of common self standing data files or data streams, distributed in real-time or not by specific sources or devices. Obviously, computer programs are also considered as digital content items. In this context, DRM can be defined as the management and enforcement of usage rules, including copyright rules, on digital content. Optionally, it can also be responsible for all the accounting aspects.

The main objectives of a typical DRM system may be summarized as follows:

- *association with access rights and copyrights*: DRM enables the owner of digital content to associate copyrights with the access rights on a digital item, and furthermore to enable enforcement of these rights during usage of the item at the consumer side. For instance, file copying may only be possible for people who have obtained the necessary rights. This can be based on their role and identity within an organization or it can be based on the fact that they have paid for the proper copyright.
- *dynamic and efficient management of usage rules*: Sophisticated DRM systems enable restrictions on the consumption of digital content to a predefined period of time, conditional access to the content, e.g. the number of times digital content can be reproduced, or detailed

¹ Digital Item (DI): Any material that can be digitised and transferred over a network from a source towards a consumer. An edge network device that can render or reproduce the original content if authorized and most importantly according to the expression of the rights issued by the rights holder/issuer. Digital items may be packaged as any kind of digital file but also they may be represented as any kind of digitized bit-stream, originated by an already stored file or captured in real-time by a specific device.



specification of the allowed actions on the digital item (e.g. view, but not print, view only in low quality as a basic service, require premium service fee for higher quality).

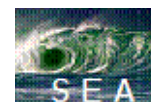
- *accounting aspects*: The fee that consumers usually pay for consuming content has to be managed and distributed among various parties. In principle it is considered that content owners require to be compensated in proportion to the effective usage of the content they own. In addition, the content publisher should get a compensation for making content available to consumers. Furthermore, the network operator may have a financial agreement based on the amount of traffic associated with specific content. The DRM system must support accounting and financial aspects, or at least enable to cooperate with external accounting or payment systems.

The main roles involved in DRM are the *producer*, the *consumer* and the *publisher*. Each role represents one or more persons, services, or entities in general, that play a specific role in the DRM processing and content delivery chain. The producer is the entity that produces content, or at least owns the rights to distribute and sell it, and represents the starting point of the DRM chain. This can be, for instance, an author, a musician or a movie industry. At the other end of the DRM chain is the consumer. Consumers represent the entities that actually want to obtain and reproduce content at their own devices for their own benefit (e.g. entertainment, information). A typical example is a home user that downloads and plays music or video files. The publisher is the entity that owns and in principle manages the DRM system used to distribute the content. It actually mediates among the producer and the consumer and the basic functional objective is to guarantee that digital content is accessed only by authorised by the publisher consumers and that content consumption at the latter is performed as intended by the former.

Most DRM technologies available today focus on the consumer part. The *DRM Agent* is an entity at consumer side that is responsible for performing the DRM-specific operations in a secure way while obeying the right specifications. On the other hand, producer tools enable producers to add content and corresponding contracts to the DRM system. In this way, content becomes publicly available and usage licenses can be issued for it. In this context, a license represents a complete description of the copyright and the access rights on a particular digital item that associates the item with the consumer and the means and media used for content reproduction at the consumer side. License creation and distribution is a task assigned to the publisher. It is obvious that the publisher role cannot be centrally located in a single node; in principle it requires the intervention of both the producer and the consumer. From the implementation viewpoint, publishing encompasses description of the copyright and access rights on the digital content on the producer side, while on the consumer/client side to unprotect the protected content and guarantee that all access rights are enforced and respected as intended. Usually, some kind of encryption is used to protect content.

To formalize the rights a consumer may obtain on some content, usage rules can be defined in a Rights Expression Language (REL). A REL defines a language and vocabulary that enable the specification and interpretation of usage rules in an unambiguous way. For example it can be used to formally describe that a person can listen to a specific DRM protected song only for a specific number of times in a particular time period. The two most well established RELs are ODRL and MPEG REL (see sections 2.1.4.2 and 0). The concept of licenses introduces a separation of DRM protected content and the sets of usage rules to be associated with it. Licenses are typically bound to protected content, but may also be associated with one or more specific devices or specific consumers. In the latter case, only these specific devices or persons are able to consume the corresponding protected content using that license. Each distinct set of usage rules can define another license type. In this way different license types may correspond to the same content.

Typically producers specify different sets of usage rules describing the rights particular consumers can obtain. Moreover, they may also require associating business information with a license type, such as the price and distribution period. All this information is negotiated with the publisher and combined in a contract corresponding to the distributed content. This contract contains all information that is needed by the DRM system to issue different license types. From a technical viewpoint, a REL must specify a number of concepts:

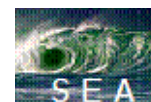


- The digital item for which rights are expressed.
- The parties that obtain the rights in terms of end-users or particular devices or any kind of association among them; for instance in this way the following access control scenario may be implemented: a group of employees of a company may have access to a particular file only within the company's premises (computers located only in the company's offices), making in this way impossible to access the file at home or meaningless to distribute it to unauthorised users.
- The actions that can be performed by a user on a digital item. It is common practice to distinguish between client-side reproduction actions (e.g. play, print, execute etc.) and distribution actions (copy, sell, duplicate and other). It has to be noted that actions are often content-type specific.
- The conditions that must be fulfilled before being able to do one or more associated actions on the content. Conditions can be grouped:
 - Party bounds (what subject). Only devices or users with certain properties may be able to do certain actions. A device may need for example a valid certificate that identifies it as a secure device, or it may have to meet certain hardware specifications.
 - Content bounds (what object). The associated action is only possible on a part of the content. The content can for example only be consumed if it has a certain quality or format or if it has other properties.
 - Repeating bounds (how long, how many). The associated action can only be done a limited number of times (e.g. 10 times) or the accumulated time of the action can be limited (e.g. you can listen to the song max. 60 minutes altogether).
 - Geographical bounds (where). It is possible to specify where content can be consumed.
 - Temporal bounds (when). It is possible to specify when content can be consumed. This can be a period (e.g. 2 months) that starts either at a predefined time (e.g. January 1st) or at the time the associated action is performed for the rest time. Another option is to define intervals within limited time zones such as a day, a week, a month or a year. For example, usage rules can define that a movie can only be viewed between 8.00 p.m. and 11.00 p.m. each Saturday and Sunday.

Contemporary DRM technologies opens a whole set of new possibilities, not only by enabling enforcement of rules on how the content can be consumed, but also in the way a consumer can distribute its rights to consume content on other devices or by other persons. The REL enables the specification and interpretation of usage rules in a precise manner according to the intentions of the content producer.

Associating rights to protected content

As aforementioned, rights are defined by usage rules that have to be associated with the digital items being protected. The rights have to be bound to a specific piece of protected content. In the past, attempts were made to embed usage rules into the content so that these would be impossible or very difficult to be accessed. An alternative method still used today is defining a fixed, technology dependent set of usage rules. In this case, each piece of content that is protected has associated with it the same set of usage rules. Clearly, this approach is not so flexible and has serious drawbacks: content producers should, for instance, accept the consumption limitations which the DRM technology vendor has embedded within the DRM modules employed, while business models such as pay-per-view are not possible. The third and most flexible method introduces the concept of licenses. According to it, a separation between content and (sets of) usage rules is made: on the one hand we have protected content and on the other hand we have licenses containing usage rules corresponding to that protected content. Different license types can correspond to the same content. Sometimes, one license contains usage rules corresponding to multiple pieces of content.



Different sets of rights corresponding to the same piece of content can be specified, resulting in different license types. Different users can thus have different rights on the same content. Content owners or their delegates that are able to submit content to a DRM system (i.e. the producers) often want to specify the license types by themselves, i.e. they want to specify different sets of usage rules specifying the rights that can be obtained. The producer also wants to set some business information, such as the price and distribution period, for the different license types. All this information is combined in a contract associated with the submitted content. This contract contains all information that is needed by the online DRM system to issue different license types.

Before consumers are able to use protected content, they first have to obtain it along with a corresponding license that enables them to use the content according to the usage rules described therein. In a typical simplified scenario, a consumer uses a DRM system agent to contact the available Content Service. This service enables to look up and download protected content. Content can be consumed after a license has been requested (if it is required by the DRM technology), while the license is acquired via the License Service.

Instead of distributing content using online services, another scenario called super-distribution is possible. Super-distribution allows the distribution of the protected content using peer-to-peer networks, portable media, e-mail, FTP, etc. In fact, there is no limitation of the distribution mechanisms employed. In any case however each consumer is obliged to obtain a license by the publisher in order to be capable of consuming the content in one of its devices.

Licenses are not only bound to protected content, but also to one or more specific devices or persons that are legitimate consumers. Only these specific devices or persons are able to consume the corresponding protected content using that license. Distributing a license to other, unauthorized parties is thus useless. The DRM Agent on the device is responsible for the enforcement of the rights: no other rights than these specified by the usage rules in the license may be possible by the consuming entity.

The procedure followed in order for a digital item to be protected, distributed and consumed according to the intentions of the producer and/or the publisher may be summarised in the following steps:

1. *Content production*: This is the phase where the producer creates a file or network stream for which it is willing to distribute to a set of consumers (already known or to the general public).
2. *Content protection/access rights expression/license creation*: Digital item protection objective is twofold; on the one hand it has to be assured that not everybody has access to the particular item but only users authorised by the producer and/or the publisher. This task is typically implemented by employing encryption techniques. On the other hand it has to be assured that authorised consumers reproduce the content as intended by the producer and/or the publisher. Access rights expression is performed by using a particular for this purpose Rights Expression Language (REL) that leads to the creation of a particular document (in contemporary DRM systems this is typically an XML document) that corresponds to the license. Licenses associate all functional entities of the DRM system, i.e. the producer, the publisher and the consumer (user and/or user device), with the particular digital item being protected.
3. *License and content distribution*: this is the step where a consumer is enabled to obtain a particular digital item by employing any kind of networking means available. Content distribution may be considered as a loosely controllable procedure (the consumer may use any network or distribution mechanism available) as typically the digital item is encrypted and accessed only by consumers that have successfully acquired the appropriate licence. License distribution requires employment of advanced security technologies and mechanisms since the license creation and acquisition procedure requires authentication of the producer and consumer entities respectively to the DRM system (to or via the publisher).



4. *Content consumption*: this is the last step where the consumers are enabled to reproduce the obtained digital item according to the access rights described in the obtained license. Typically this step may be broken down into the following sub-steps:
 - a. *License interpretation*: during this step the obtained license is interpreted that results into the creation of context object containing a list of the allowed or not allowed actions on the digital item.
 - b. *Digital item “un-protection”*: in this step the digital item is obtained in its original form, i.e. the one created by the producer. Typically this step involves decryption of the obtained digital item according to a pre-specified decryption procedure. It has to be noted that this procedure is not performed in the user space of the device used for content re-production so as not to enable the consumer to have access to the unprotected digital item freely (in most cases this task is performed by a delegated DRM Agent that is integrated with specific hardware or a particular application).
 - c. *Access right enforcement and content consumption*: this is the last step where a particular application in combination with an access right enforcement module are responsible for both reproducing the content on behalf of the end-user while enforcing that the expressed within the license access rights are respected verbatim.

2.1.3. DRM System Architecture and Requirements

In order to protect digital content from unauthorised access, contemporary DRM systems use device content packaging techniques that are applied to the digital items being protected. Basic packaging techniques comprise of at least content encryption; advanced contemporary DRM systems also employ association of the encrypted content with a Digital Rights Object. As aforementioned, the latter contains all necessary information to enable the digital item’s reproduction at the consumer. In principle, the basic steps of content distribution deal content encryption and also Rights Object (i.e. the license) generation procedures as a combined task, performed by both the producer and the publisher. Depending on the deployment scenario implemented, the content and rights issuer roles may be fulfilled by the same or different actors, and thus implemented by the same or different network nodes. For example, content owners may pre-package the content, which is then distributed by a content distributor acting as both content issuer and rights issuer.

A Rights Object governs how DRM protected content may be used. In most cases, this is an XML document specifying permissions and constraints associated with a piece of DRM content. DRM protected content cannot be used without its associated Rights Object; in other words it may only be used according to the permissions and constraints specified within the Rights Object. Most DRM systems make a logical separation of DRM content from the Rights Objects. DRM content and Rights Objects may be requested separately or not, and also may be delivered separately or at the same time. For example, a user can select a piece of content, pay for it, and receive DRM content and a Rights Object in the same transaction. Later, if the Rights Object expires, the user may acquire a new Rights Object, without having to download the DRM Content again by directly contacting the publisher of the license itself. It has to be noted that the licence accompanying protected digital items usually contain all necessary information not only for the digital item and the producer but also the publisher of the license and available ways to purchase or update them.

Rights Objects associated with DRM content have to be enforced at the point of consumption. This in most cases is implemented by a particular module operating at the consumer side. In the following we will refer to this module with the term DRM Agent. The DRM Agent comprises of various components responsible for the decryption of the protected digital item. Additionally it includes a trusted component used for enforcing the permissions and constraints expressed within the licenses for the particular digital item possibly even on the particular device that the consumer uses (optionally). A Rights Object is cryptographically bound to a specific DRM Agent, so that only a specific DRM Agent is enabled to access it. Additionally DRM content can only be accessed with a valid Rights Object, and so can be freely distributed. This enables, loose distribution of the DRM



content, as users can freely exchange DRM packaged digital items. In DRM terminology, the term super-distribution refers to the procedure where a DRM Agent is enabled to distribute already protected content to other DRM Agents freely, as this procedure does not influence the fundamental access and control of the protected content. To access DRM content on a client, a new Rights Object has to be requested and delivered to a DRM Agent on that device from the core DRM system where the licenses are generated and distributed. In contrast to the super-distribution procedure followed for content distribution, license distribution requires interaction of a consumer with an AAA (Authentication, Authorisation and Accounting) system that is integrated with the DRM system providing the license. Similar interaction is required from the side of the content producer for authenticating, authorising the producer and also for interfacing it with the accounting and payment system.

In order to describe the complete picture of a contemporary DRM system, all participating components in the content delivery chain (i.e. content protection and delivery procedures) should be identified. In the sequel all these actors, their roles and assigned functionality are described from various perspectives such as the consumer, the producer and the publisher, while also identifying the requirements posed from the application but also the core system viewpoints. Figure 1 following depicts the basic architecture of a DRM system and the procedures executed during DRM-enabled content distribution.

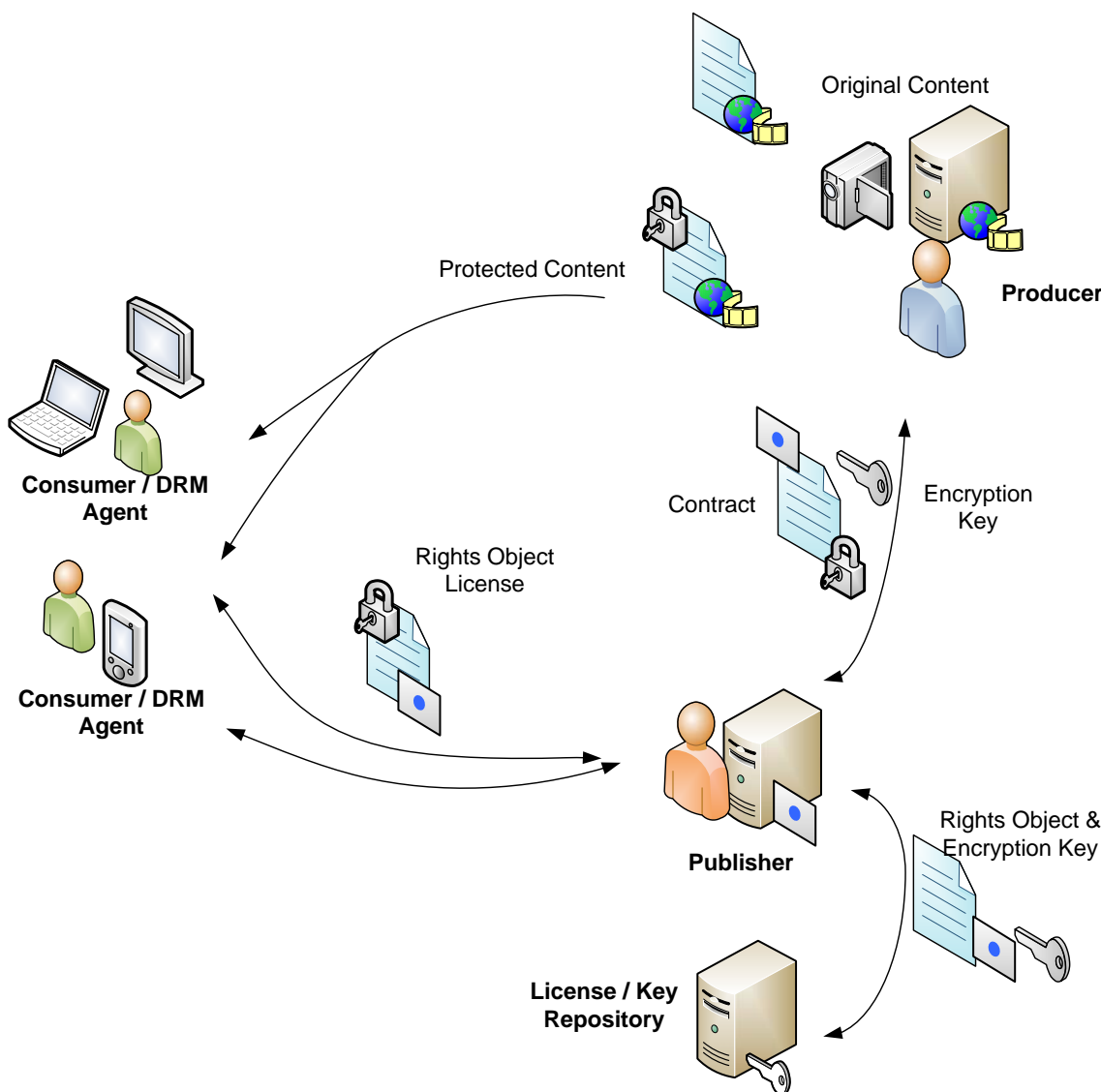
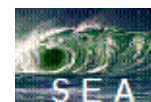


Figure 1. DRM System Architecture



Consumers typically require having an easy to use way so as to obtain securely the content according to their demands. This comprises of the ability to browse a content catalogue of an online system. Since consumers also need a license, they must be able to select a license type and view the usage rules, in a human readable format, associated with it. Generally, consumers first have to pay for the content or service, and thus a system should facilitate payments via various business models (e.g. subscription, pay-per-license, or pay-per-use). Moreover, when time-based licenses expire, an update procedure should be provided accompanied by a financial transaction for the acquisition of the updated or new license. Consumers also want to browse their obtained licenses locally and view the usage rules in a human readable format. Finally, consumers want to consume the protected content, according to the usage rules associated with the corresponding license. In order to fetch licenses (and sometimes also protected content), consumers need to authenticate to the DRM system.

Producers on the other hand basically require using a straightforward way to easily compose a contract that will lead to the creation of the respective licenses to be used by the consumers. The content must be submitted to the DRM system and stored in a specific place so that licenses to be available to consumers regardless of the online availability of the producer². Furthermore, given that contract definition is a dynamic procedure, producers also require to be enabled to perform alterations to already created contracts that eventually may lead to license update or even revocation. Last but not least producers also require to be informed about the consumption of the content they own. For this reason they also require from publishers to be informed about the number of licenses acquired by consumers via the publishing services offered and be compensated respectively. Obviously since all these procedures are specific for each producer additional requirements are posed in respect to the authentication and authorisation actions performed by producers and publishers.

Publishers pose requirements for managing content licenses, authentication and authorisation of the end-users that interact with them, i.e. consumers or producers, and also interaction with the DRM system's core elements so as to securely store the licenses but also to obtain usage statistics related to the license distribution.

Contemporary DRM systems foresee the operation of additional entities that handle functionality related to accounting and billing but also interaction with external security infrastructure (for instance PKI) such as established Certification Authorities used for authenticating of the various end-users of the system and also for checking the validity of the used certificates. Generally all contemporary DRM systems are considered to have integrated AAA (Authentication, Authorisation and Accounting) functionality. It has to be noted that these systems may be internal to the core DRM system (i.e. be developed as part of the entire system) or external to it (e.g. an external accounting facility may be used to handle all financial issues raised).

In what follows a brief description of the interaction of each of the consumer, producer and publisher entities with the services provided by a typical DRM system is provided. Figure 2 following provides a block diagram of the interaction among the basic entities and the system services.

2.1.3.1 Content Consumers

We identify four key DRM services with respect to the content consumer: the Content Service the License Service, the Access Service, and the Tracking Service. In addition, we describe two additional services for payment and certification. In most cases the latter services are provided by an external system.

² In principle the license distribution system is separated with the content distribution one. Either system has different performance and functional requirements; the former focuses on secure distribution of the licenses and possibly the interaction with an accounting and billing system whereas for the latter poses requirements relate to the offered quality of offered services (e.g. in case of real-time service offerings) and the networking technologies employed for the content distribution (e.g. FTP file transfer, P2P networks etc.).



Content Service: In order for consumers to obtain protected content, they use their DRM Agent to contact the Content Service where they can search for content. Before a consumer obtains any content, the Content Service protects it. Since this protected content may be personalized, the Content Service needs certainty about the identity of the consumer. This is obtained by interaction (i.e. running a protocol) between the content distribution service, the DRM Agent and the access service. This identification phase, however, is not always needed. The Content Service may need to send protection data, specific to that piece of content, to the License Service, to allow this service to issue associated licenses. The Content Service receives its content and potentially some additional data from the Import Service, which is discussed in the producer part. The Content Service does not necessarily force content storage at a central point (this is not forbidden however) but it is rather a registration service where an online catalogue is created with all necessary information regarding the meta-information related to a particular piece of distributed and protected content.

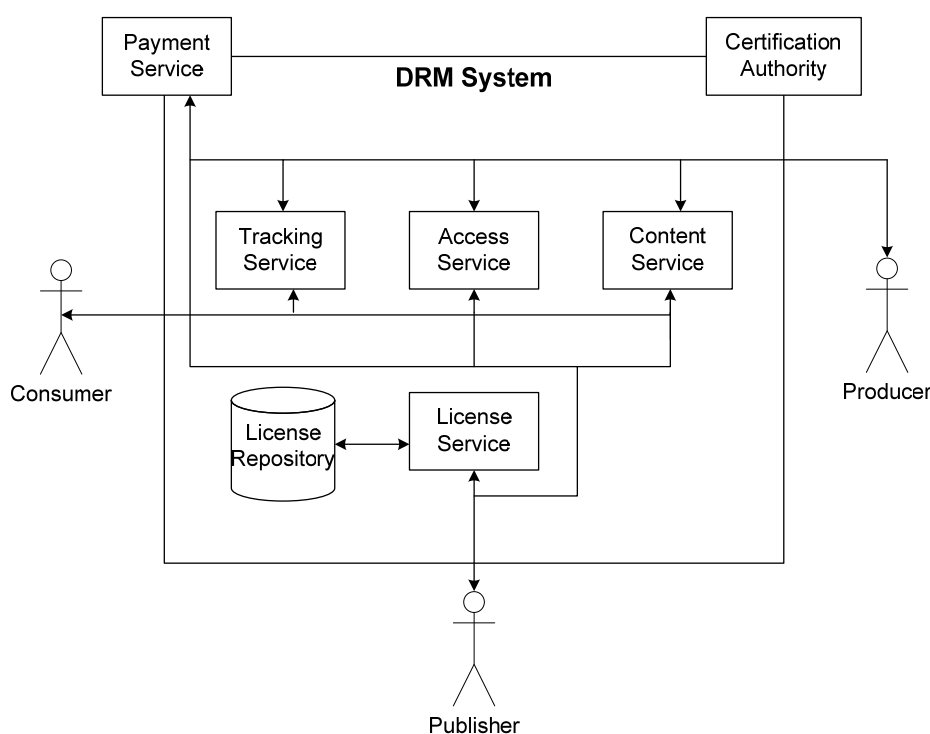
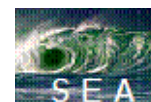


Figure 2. DRM System basic services

License Service: Once the License Service has received the protection data from the Content Service, it is able to issue corresponding licenses. According to the contract signed among the producer and the publisher, licenses may be pre-defined or generated upon consumer request. The License Service may offer different license types corresponding to the same content. The consumer's DRM Agent provides some system or user specific parameters to generate a license that is only usable by itself. Before the license is issued (distributed), the consumer usually has to pay for it. Therefore, the License Service asks the Access Service if it may issue the license. The Access Service in turn contacts the Payment Service and will only answer positively if the consumer has effectively paid. Expired licenses can be updated by sending them to the License Service, which will issue new ones to the consumer, often after a new payment.

Access Service: The Access Service is responsible for the authentication of the content producers, consumers (or its DRM Agents) and the publisher(s). It is also responsible for checking payments of both consumers and producers before allowing particular actions on the DRM system. The Access Service may, for example, deny access if some bills are not paid or if consumers fail to identify themselves. Before content consumers are able to obtain licenses or even content, some registration



procedure may be necessary. This procedure usually involves the Access Service and, if some financial transaction is needed, the Payment Service.

Payment Service: The financial aspects are taken care of by a Payment Service, typically a bank or another financial institution. Obviously, some interaction is needed between the Payment Service, the Access Service and the DRM client, producer tool or consumer tool.

Certification Authority (CA) is only necessary if certificates are used. The CA is responsible for issuing, distribution and revocation of certificates used by the consumer while interacting with the licence service.

2.1.3.2 Content producer

Before producers are enabled to “put” (i.e. typical upload in case of central storage or register in case of distributed storage) content into the DRM system, they must identify themselves via the Access Service. This is necessary to prevent misuse of the DRM system: only the owners of the rights on the content may submit³ that content to the DRM system. After identification, the producer can submit the content to the DRM system, including the meta-information and a corresponding contract. In the sequel, the content is sent to the Content Service (i.e. centrally stored or registered), while the contract and additional information (such as the content identifier) are sent to the License Service. In a similar way, content can be updated or removed from the DRM system.

2.1.3.3 Content publisher

Content publishers are in most cases entities integrated into the core of a DRM system. To this end they basically use the Access Service in order to authenticate themselves to the system (in case they are considered separated from it). Additionally apart from the interaction described earlier as part of their communication with the content producers and consumers, they also interact with the available Tracking Service in order to obtain necessary information for statistical information regarding the utilisation of the digital items they manage.

2.1.4. Rights Description Languages

The fundamental issue for a DRM system is how to express the rights specification of protected digital content. For a specific DRM system, the content rights can be expressed in arbitrary syntax, and its interpretation totally depends on the system developers. However, it is true that a piece of protected content can be distributed to other DRM systems, whose users may expect that the rights management for the guest content can work there as well. Therefore, different DRM systems need a standard way to express and interpret the rights specification for realizing interoperability, and a common language that can be shared among all the participants in the digital workflow is required [6].

In order to support a wide variety of business models, the standard rights expression language must be:

- Comprehensive, providing a framework to express rights at different stages of a work flow or life cycle.
- Generic, defining a large body of format and business neutral terms that you use to specify rights for any digital content or service.

³ Content submission to a DRM system does not necessarily correspond to a file upload to a central document or media file server. On the contrary it basically requires only the registration of the digital item being protected to a central repository, so that consumers are enabled to browse a catalogue to obtain the particular item. Content distribution service and procedures may be implemented by making use of every available mean from direct download from a central file server (devising FTP for instance) or by making use of a Peer -to-Peer network where the file is distributed among various peers and downloading is performed simultaneously from a group of them (see section 2.2.2)



- Precise, using a grammar and processing rules to ensure unique interpretation of the language.
- Extensible, allowing any third party to define elements to meet specific business needs.

In the following sections, a brief description of some rights description languages is provided.

2.1.4.1 eXtensible rights Markup Language (XrML)

XrML [7] is an XML-based usage grammar for specifying rights and conditions to control the access to digital content and services. XrML had its roots in Xerox Palo Alto Research Center [8]. Digital Property Rights Language (DPRL) [9] was first introduced in 1996. DPRL became XrML when the metalanguage (used to construct the language) was changed from a lisp-style metalanguage to XML in 1999.

XrML provides the possibility to anyone owning or distributing digital resources (such as content, services, or software applications) to identify the parties allowed to use those resources, the rights available to those parties, and the terms and conditions under which those rights may be exercised. These four elements are the Core of the language and determine the full context of the rights that are specified. In other words, it is not sufficient to just specify that the right to view certain content has been granted, but also who can view it and under what conditions.

Since its inception, the language has evolved through industry feedback, critical review, and product implementation. The language has become comprehensive by providing a framework to express rights at different stages of a workflow or lifecycle, generic by defining a large body of format and business neutral terms (about 100) and using these terms to specify rights to any digital content and service, and precise through the development of a grammar and processing rules that enable unique interpretation of the language. XrML is by far the most advanced and mature rights language in use today. Since 1999, the emphasis has been to get the language implemented in real life systems. This experience has resulted in additional system-related features (trust, for example) that are now part of the language. The current version of the language is 2.0.

The data model in XrML 2.0 includes the concepts of license, grant, principal, right, resource and condition. The key top-level construct is a license, which contains a set of grants that convey to certain principals certain rights to certain resources under certain conditions. A principal encapsulates the identification of principals to whom rights are granted. It can be represented by a “key holder”, someone identified as possessing a secret key such as the private key of a public/ private key pair. A right is the action that a principal can be granted to exercise against resources, such as play, edit, print, copy, delete and etc. A license file for a specific piece of protected content is composed from these concepts to describe the rights specification.

2.1.4.2 Open Digital Rights Language (ODRL)

The Open Digital Rights Language (ODRL) provides the semantics for DRM expressions in open and trusted environments whilst being agnostic to mechanisms to achieve the secure architectures.

It is envisaged that ODRL will “plug into” an open framework that enables peer-to-peer interoperability for DRM services. However, ODRL can also be used as a mechanism to express rights statements on its own and to plug into existing DRM architectures and frameworks.

ODRL complements existing analogue rights management standards by providing digital equivalents, and supports an expansible range of new services that can be afforded by the digital nature of the assets in the Web environment. In the physical environment, ODRL can also be used to enable machine-based processing for rights management. ODRL is a standard language and vocabulary for the expression of terms and conditions over assets.

ODRL covers a core set of semantics for these purposes including the rights holders and the expression of permissible usages for asset manifestations. Rights can be specified for a specific asset manifestation (i.e. format) or could be applied to a range of manifestations of the asset.

ODRL is focused on the semantics of expressing rights languages and definitions of elements in the data dictionary.

ODRL can be used within trusted or untrusted systems for both digital and physical assets. However, ODRL does not determine the capabilities nor requirements of any trusted services (e.g. for content protection, digital/physical delivery, and payment negotiation) that utilises its language. Clearly, however, ODRL will benefit transactions over digital assets as these can be captured and managed as a single rights transaction. In the physical world, ODRL expressions would need an accompanying system with the distribution of the physical asset.

ODRL defines a core set of semantics. Additional semantics can be layered on top of ODRL for third-party value added services with additional data dictionaries.

ODRL does not enforce or mandate any policies for DRM, but provides the mechanisms to express such policies.

ODRL is based on an extensible model for rights expressions which involves a number of core entities and their relationships. This ODRL Foundation Model is shown in Figure 3.

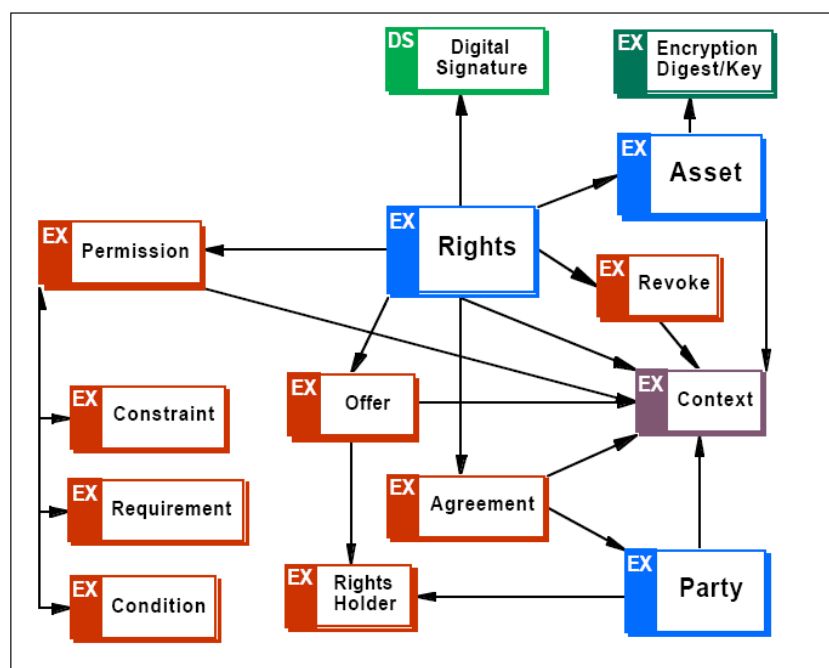


Figure 3. ODRL Foundation Model

The model, as shown in Figure 3, consists of the following three core entities:

- *Assets*: include any physical or digital content. The Assets must be uniquely identified and may consist of many subparts and be in many different formats. Assets can also be non- tangible expressions of works and/or manifested in particular renditions. Assets may also be encrypted to enable secure distribution of content.
- *Rights*: include Permissions which can then contain Constraints, Requirements, and Conditions. Permissions are the actual usages or activities allowed over the Assets (e.g. Play a video Asset). Constraints are limits to these Permissions (e.g. Play the video for a maximum of 5 times). Requirements are the obligations needed to exercise the Permission (e.g. Pay \$5 each time you Play the video). Conditions specify exceptions that, if become true, expire the Permissions and renegotiation may be required (e.g. If Credit Card expires then all Permissions are withdrawn to Play the video).
- *Parties*: include end users and Rights Holders. Parties can be humans, organisations, and defined roles. End users are usually the asset consumers. Rights Holders are usually parties



that have played some role in the creation, production, distribution of the Asset and can assert some form of ownership over the Asset and/or its Permissions. Rights Holders may also receive royalties.

With these three core entities, the foundation model can then express *Offers* and *Agreements*. Offers are proposals from Rights Holders for specific Rights over their Assets. Agreements are when Parties enter into contracts or deals with specific Offers. The model can also then express revoking of any Offers or Agreements.

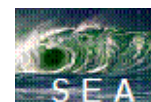
The representation of Offers and Agreements is important core aspect of ODRL. This makes clear what the purpose of the rights expressions is achieving. Many different Offers can be created to meet various business models for assets. Offers can be linked, creating a hierarchy of options for end users. Agreements are the transformation of an Offer into a license for rights over an asset by parties. Also, there is no requirement that Offers be made prior to Agreements. After human interactions, Agreements can be created to express the accepted terms and conditions.

Most entities in the model can support a specific Context. A Context, which is relative to the entity, can describe further information about that entity or the relationship between entities.

For example, the Context of an Agreement may specify the date of the transaction, the Context of a Party may specify their role. Although not mandatory, the use of Context to assign unique identifiers to the entire rights expression is highly recommended.

The Context also plays a very important role in identifying the entity (using a unique number/code from a standard identification scheme). This ability to uniquely reference any entity can be utilised in providing linkages between entities. For example, an Agreement can be linked to the original Offer by the latter's unique id number.

The ODRL language has been adopted by the SEA license system, so there will be more mentions along this document to the ODRL language, concretely in sections 3.2.2.4 and 3.2.3. More details of the ODRL standard can be found in [10].



2.1.4.3 MPEG-21 REL

The MPEG REL, as defined by ISO/IEC 21000-5, provides flexible, interoperable mechanisms to support transparent and augmented use of digital resources throughout the value chain in a way that protects the digital resource and honours the rights, conditions, and fees specified for it. For instance, it provides mechanisms in support of publishing, distributing, and consuming digital content such as electronic books, digital movies, digital music, broadcast content, interactive games, computer software, and other creations in digital form. It also supports specification of access and usage controls for digital content in cases where financial exchange is not a term of use, and supports exchange of sensitive or private digital content and personal information.

The standard REL can support guaranteed end-to-end interoperability, consistency, and reliability among different systems and services. To do so, it offers richness and extensibility in declaring rights, conditions, and obligations; ease and persistence in identifying and associating these with digital content; and flexibility in supporting multiple usage/business models.

The MPEG REL adopts a simple and extensible data model for many of its key concepts and elements.

Its data model for a rights expression includes four basic entities. The basic relationship among these entities is defined by the MPEG REL assertion "grant." Structurally, a grant consists of the following elements, as can be seen in Figure 4 :

1. The principal to whom the grant is issued: a principal encapsulates the identification of a party to whom rights are granted. Each principal identifies exactly one party. A principal denotes the party that it identifies by information unique to that party. Usefully, this is information with an associated authentication mechanism by which the principal can prove its identity.
2. The right that the grant specifies: a right is the "verb" that a principal may exercise against some resource under some condition. Typically, a right specifies an action (or activity) or a class of actions that a principal may perform on or using the associated resource.
3. The resource to which the right in the grant applies: a resource is the "object" to which a principal can be granted a right. A resource can be a digital work (such as an e-book, an audio or video file, or an image), a service (such as an email or B2B transaction service), or a piece of information a principal can own (such as a name or an email address).
4. The condition that must be met before the right can be exercised: a condition specifies the terms, conditions, and obligations under which rights can be exercised. A simple condition is a time interval during which a right can be exercised. A slightly complicated condition is to require a principal to have a valid, prerequisite right. In this way, eligibility to exercise one right can depend on the eligibility to exercise other rights.
5. The issuer identifies the party who issued the license. It can contain a principal or a signature to identify the issuing party. Using a signature to identify the issuer is useful to address trust and authentication issues and to ensure license integrity. The issuer element can also contain details relating to the issuance of the license (for instance, the time it was issued or the mechanism by which it might be revoked).

By itself, a grant is not a complete rights expression that can be transferred unambiguously from one party to another. A full rights expression is called a license. A typical license consists of one or more grants and an issuer, which identifies the party who issued the license. Figure 4 illustrates the structure of a simple license. More information can be found in [11].

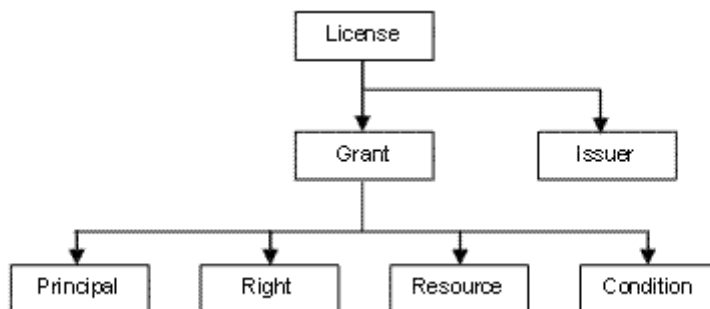


Figure 4. Structure of a simple License

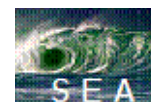
2.2. Peer to Peer Networking

Peer-to-peer (P2P) networking technologies have gained significant attention during the last years. In principle they introduce an alternative to client/server concept according to which an overlay network of equivalent peers is formed, within which two or more individuals are enabled to communicate without central coordination. In contrast to the client/server communication paradigm, P2P networks offer improved scalability, self-organisation and decentralized coordination of previously underused or limited resources. Moreover, they achieve greater fault tolerance by eliminating single points of failure limitations as perceived in centralized distribution systems.

2.2.1. Basic concepts and applications - Architecture

Pure peer-to-peer networking refers to totally distributed systems, in which all nodes are completely equivalent in terms of the functionality and the tasks they perform. It has to be noted though that contemporary P2P systems also encompass “supernodes”, i.e. nodes that function as dynamically assigned localized miniservers, or systems that rely on some centralized server infrastructure for a subset of basic tasks (e.g. bootstrapping, reputation ratings, etc). P2P networking encompasses applications that take advantage of resources such as storage, CPU processing cycles, content distribution, or human presence available at the edges of an internet. It is the lack of central control, for instance such as the existence of a DNS service, that sets the P2P network to be autonomous in the sense that it can provide its services to their peers without it. The two defining characteristics of contemporary peer-to-peer technologies and architectures are:

- *Resource sharing*: P2P networking enables direct communication and resource exchange among communicating peers forming the P2P overlay network, rather than requiring the mediation of centralized infrastructure (hardware/software). Core functional servers may be used for particular tasks like system bootstrapping, P2P node management (participation, leaving) or data encryption key storage and distribution. However, systems that rely on one or more global centralized servers for their basic operation (e.g. for maintaining a global index and searching through it) are in general broadening the definition and scope of P2P networking principles. As the nodes of a P2P network cannot rely on a central server to coordinate the content distribution and the management of the entire network, they are required to actively operate independently and for this purpose to perform tasks such as searching for other nodes, locating or caching content, routing information and messages, connecting to or disconnecting from other neighbouring nodes, encrypting, introducing, retrieving, decrypting and verifying content.
- Their ability to treat instability and variable connectivity as the norm, automatically adapting to failures in both network connections and computers, as well as to a transient population of nodes. This fault-tolerant, self-organizing capacity suggests the need for an adaptive network topology that will change as nodes enter or leave and network connections fail or recover, in order to maintain its connectivity and performance.



2.2.2. Streaming Content in P2P Networks

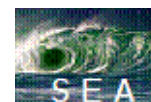
In recent years, peer-to-peer computing has been employed for efficient live streaming distribution. In a peer-to-peer overlay network, every peer works in a relay mode to download streaming data flowing from a number of peers and concurrently upload the same stream to others. Through sufficiently utilizing the bandwidth and computation resources possessed by peers, workload placed on the media source servers is reduced and hence the overall scalability of the offered service is greatly increased.

Media streaming over P2P is implemented by considering a shifting time window at which a real-time distributed media stream is shared among multiple senders. Media sharing/distribution may be performed in (almost) real-time towards multiple recipients as long as it is guaranteed that all recipients will have obtained the stream chunks corresponding to the time window on time. In more detail, at the beginning of the stream session, recipients start buffering every stream chunk arriving from senders that correspond to the current time window. After the expiration of the initial time window two actions take place: (i) recipients start playing the media stream and (ii) senders start forwarding chunks corresponding to the subsequent time window. In parallel, recipients continue to download media stream chunks corresponding to the subsequent time window (the window is time shifting) that becomes the current window.

In contrast to typical client-server streaming model in which roles of the end-points are clearly defined among communicating entities, i.e. clients that receive content, and servers, providing it, in P2P streaming model, the roles are not clearly defined; each peer can operate as a client, server, or both simultaneously. Moreover, all entities are obliged to cooperate with each other. However, P2P streaming facilitates more efficient data distribution as content (i.e. the buffered live stream) is located in more than one point of the network. Hence, it is not only one streaming server that actually shares the media stream (bandwidth, processing time) but all receivers to enable the live broadcast. Thus the problem of P2P streaming actually deals with the formation of a robust overlay network that is actually in place to satisfy the majority of QoS requirements posed by the streaming nature of the distributed media in principle related to delay and bandwidth constraints in the context of a P2P networking. In this case the latter is used for relaxing processing load at the media server and congestion at the content provider network. On the other hand, however, the overlay topology becomes the bottleneck from the architectural perspective as it is the topology that requires to be formed efficiently so as to overcome the media distribution constraints. The following issues are important in designing an efficient P2P technique [19]:

- The end-to-end delay from the source to a receiver may be excessive because the content may have to go through a number of intermediate receivers. To shorten this delay (whereby, increasing the liveness of the media content), the distribution tree height should be kept small and the join procedure should finish fast. The end-to-end delay may also be long due to an occurrence of bottleneck at a tree node. The worst bottleneck happens if the tree is a star rooted at the source. The bottleneck is most reduced if the tree is a chain, however in this case the leaf node experiences a long delay. Therefore, apart from enforcing the tree to be short, it is desirable to have the node degree bounded.
- The behaviour of receivers is unpredictable; they are free to join and leave the service at any time, thus abandoning their descendant peers. To prevent service interruption, a robust technique has to provide a quick and graceful recovery should a failure occur.
- For efficient use of network resources and due to the resource limitation at each receiver, the control overhead at each receiver should be small. This is important to the scalability of a system with a large number of receivers.

P2P live streaming applications address in principle crowded audiences comprised of a set of simultaneously communicating receivers. For this reason they require to employ robust mechanisms facilitating group communication, such as application layer multicasting, for facilitating efficient content distribution. In this context, discrimination may be performed based on the way the overlay



network is formulated. In more detail three approaches may be discriminated: the source-driven, the receiver-driven and the data-driven ones.

In the source-driven case a separation is performed among media and control information. For this reason a data channel is used to handle media stream distribution to all peers whereas separate communication is established over a control channel to manage participation to established distribution groups of peers (arrivals and departures). Usually, data plans result from control plans. Examples of this approach have been presented in [18] and [19].

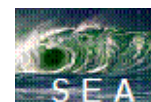
In the receiver-driven approach, control and data plan are clearly separated similarly to source-driven approach; however the control plan can be a tree, a cluster or a mesh whereas the data plan is a tree rooted at the receiver side instead of the source side. The receivers select the peers from which they want to obtain the content. However, receiver-driven approach is usually related to data encoding like layered coding (Scalable Video Coding, SVC) or multiple description coding (MDC). With SVC data are encoded in several layers and it is only mandatory to receive at least the base layer. The rest (higher) layers will only improve the quality of reception and is on the receivers' discretion to download it or not. With MDC, it is not necessary to receive a particular layer. However reception is facilitated through various sources.

Unlike source-driven approach, the data-driven approach does not clearly separate control and data plans. Peers exchange control messages about data availability in the network and each one chooses for itself its neighbourhood according to the content being downloaded. Protocols like Donet ([20]) use specialized algorithms that do not force separation among data and control information exchange for constructing and operating the overlay network. The main idea of these algorithms is as follows: an entity wishing to send a message to all other entities sends it to randomly selected entities. These entities forward the message randomly to other entities. At the end, the message reaches all receivers without a having used a clearly defined topology built in the overlay. A main difficulty however to resolve, for the group communication mechanism implemented at applicative layer, is the dynamicity of the hosts.

The push models discussed above are one-to-many distribution models that achieve minimization of the end-to-end delay. However, they are not resilient to network congestion and underutilize the resources at the network edges (close to clients). To improve services robustness, the pull models enable also delay and storage information exchange in order to achieve better performance. The hybrid push and pull combined models inherit the low-latency and stability advantages of either models. As presented in the literature, Wong introduced the idea of download with "helpers" in [21]. Peers join the swarm of BitTorrent [22] to contribute their bandwidth. When and how many helpers should be introduced into the swarm is decided by the content providers according to the network conditions. The model is not sensitive to the degradation of streaming quality of minority peers owing to central management. Furthermore, the lack of suitable incentive mechanism makes it impractical in real-world. More recently, Pouwelse proposed a social-based P2P file-sharing paradigm to exploit social group phenomena in content discovering and downloading [23]. However, problems in live streaming area have to be further studied. As in [24], we build a social network with distributed relationship memory to tackle various attacks in our assisting scheme. AnySee [33] adopts an inter-overlay optimization scheme to find the nearest neighbors, and improves global resource utilization to achieve better service quality. However, AnySee neglects the resources of idle peers who are not in any channel. The idle peers will be willing to contribute their resources by an effective incentive mechanism.

2.2.3. Security and P2P networks

The decentralized nature of P2P systems offers many positive features for digital content publishing and distributed multimedia search, not least of which is their resilience to legal and physical attacks and censorship. In order to really affect a decentralized P2P system, malicious attacks need to occur at or near every part of the system they wish to affect, while centralization generates a single point of failure. On the other hand, the lack of centralized administration tools makes it much more difficult to



implement security protection and authentication on deployed P2P systems. Moreover, the absence of a defensible border of a system means that it is hard to know friend from foe. Malicious users can actively play the role of insiders, i.e., they can run peers themselves, and often a great number of them. Also experts in the area[24] note that it is incredibly hard for any open system to defend itself against certain attacks (e.g. a Sybil attack).

Several solutions for P2P security have been proposed in the literature[24][25]. Secure distributed transport mechanisms like *Tarzan*[26][27] ensure that the identity of the user and the content of the message remain a secret, while secure overlay mechanisms are designed to ensure that the message gets through and gets there efficiently.

Rather than protecting the anonymity and content of communications, routing overlays such as *Chord*, *CAN* and *Pastry*[28] try to provide mechanisms for nodes to access objects in a fashion resistant to malicious interference. *Freenet* and *Free Haven* ensure adequate file maintenance, so that users can reliably access any file they want, or any file that have been published, respectively. The *Freenets* [29] P2P structure ensures that documents spread in a “sticky” fashion to resist attempts to stamp out important information, while still protecting those who would either access or host that information. *Free Haven* [30] uses cryptographic techniques to protect the identity of the reader, the server, the author, the document and the query, preventing any party other than that which is explicitly authorized to read or link information. *Groove* [31] is designed to ensure a robust collaborative work environment even with some parties behind firewalls, others on low-bandwidth channels, and still more with intermittent connections. All applications running on the Groove network use “delta” messages, or document updates that communicate changes of state to other members of the network, whether in a collaborative file or in a chat session. Each delta is encrypted using the group’s symmetric key for fast encryption and decryption. The group has a separate shared key for authentication, which is used to sign the digest to ensure message integrity; both keys are re-established every time someone leaves the group.

2.2.4. Basic DRM functions for P2P content delivery

The main function of DRM is to restrict the use of content to users who satisfy predefined conditions.

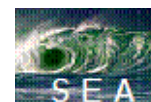
In this project, the following cases have been taken into account as conditions to be handled during DRM system design:

- 1) User name: the content can be used only by predefined/authorized users
- 2) Terminal identifier: the content can be used only in predefined terminals
- 3) Number of times the content can be used by one user
- 4) Amount of time for which the content can be used (e.g.)
- 5) Time period during which the content can be used
- 6) Type of conditional access over the content (e.g. the file may be read on screen but not printed)
- 7) User account: the content can be used only by a user who has access to a particular account

In cases **1)** and **2)**, the use of content is restricted to users who have paid charges.

In cases **3)**, **4)** and **5)**, irrespective of payment, anyone can use the content, provided the usage threshold has not been exceeded. The threshold is also supervised by the content usage control function.

In case **6)**, anyone who has access to the user account can use the content, but the charges are recorded and payment is demanded later. Obviously, any combination of the above cases enables the development of advanced access control cases (e.g. content may be used only one time within a particular period)



2.2.4.1 Methods to restrict the usage per user / terminal

The methods of restricting content usage on a user or terminal basis are based on content encryption technologies, which prohibits the free use it. However, existing content usage control function, tuned for the server-client communication model, does not permit the relay of contents. When a client (originating peer) sends a particular content to another user (destination peer) the received content cannot be used at the destination. Recipients obtain encrypted content and based on the license acquisition procedure they decrypt it and consume it according to the access rights described within the license.

There are two approaches to solving this problem:

- 1) Separate the content and its license
- 2) Rewrite the license at the destination

In case 1), the license, i.e., the right to use a piece of content is separated from the content to be delivered. Licenses are managed by a license server (DRM server). Content cannot be decrypted without the license, which should prevent illegal use of the content. Content is delivered from one peer to another peer in a P2P manner, while the license is transferred to the destination peer only when the charge has been appropriately paid. To prevent the use by another peer that gets the license improperly, the license is given an identifier linking it to each user having permission.

In case 2), the license is not separated from the content, but can be rewritten based on permission given by content owner. This mechanism, installed in content owner's terminal, controls the following procedure: when peer B receives a piece of content from another peer, e.g., peer A, the content contains A's license, so peer B cannot use this content. If peer B pays the owner, it receives from the owner a key to rewrite the license. Then peer B can use the content.

2.2.5. Protecting digital rights of streamed content in P2P Networks

With the improvement of broadband IP network, multimedia services based on streaming live media have gained much attention recently, among which IPTV has become a hot topic.

A challenge of IPTV is how to distribute programs to millions of end-users simultaneously, which requires huge network bandwidth. IP multicast may be a choice. However, due to the practical issues of routers, it has not been widely deployed. Peer-to-Peer (P2P) technology is a highly efficient method for multimedia streaming. P2P based IPTV system does not rely on dedicated application-level multicast servers. Instead, each IPTV client is potentially a server, receiving contents from up-level peers and redistributing them to down-level peers, thus alleviating the pressure of multimedia server.

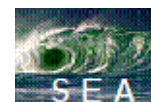
However, nowadays there is still a critical problem within the system: most of the P2P based IPTV systems (such as PPLive [34]) do not take into account content management. Digital content can be copied or redistributed without any restriction, which of course is undesired.

Therefore a content management system is required for the practical deployment of P2P based IPTV. Digital Rights Management (DRM) is such a system that includes encryption and other technologies which can control the usage and redistribution of the digital contents.

2.2.5.1 Related work

In recent years, the topic of DRM has gained more and more popularity. With the P2P technology based applications widely deployed, the topic of DRM in P2P systems is of great importance and many researches have been done in this area.

Androutsellis-Theotokis et al. [35] gives an overview of the P2P content distribution technologies and then deals with the security issues characteristic of P2P content distribution systems. These issues include secure storage, secure routing, access control, authentication, and identity management.



For multimedia content distribution in P2P networks, Chu et al. [36] proposes a mobile DRM system and business model, which utilize client side application and tamper-resistant hardware to enforce digital rights. The approach in this system is based on peer side hardware authentication, which is similar to the Trust Computing (TC) technologies. Similarly, Zhang et al. [37] and Balfe et al. [38] use the TC technologies to enhance or provide security for P2P systems. Iwata et al. [39] discuss the DRM system applicable to P2P content sharing and focus on DRM methods and DRM function assignments suitable for P2P content delivery.

Balfe et al. [40] classify existing researched approaches into three types: conventional Server-Client based DRM architecture, distributed P2P based DRM architecture and semi-distributed P2P based DRM architecture. After comparing the pros and cons of these architectures, they propose a new type of DRM applied P2P system architecture that still keeps existing P2P system's advantage.

2.2.5.2 DRM requirements for P2P IPTV

DRM architecture should provide security for P2P based IPTV system while maintaining advantages of P2P technologies. From this aspect, current three architectures mentioned above have their own advantages and deficiencies. Conventional Server-Client based DRM architecture has good security characteristics; however the DRM server in the system may be the bottleneck of the whole system, which cripples the P2P's advantages.

Distributed P2P based DRM architecture doesn't have a DRM server and have nearly all of the DRM functions assigned on the peer node (Super Node). However this distribution affects the security adversely. Semi-distributed P2P based DRM architecture falls between them.

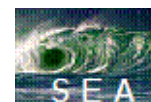
TC technologies are effective methods to provide security in P2P systems. However, at present TC technologies always rely on hardware to provide security, and there is no such hardware on the client side of most IPTV systems. It is still a long way before TC technologies can be deployed in practical DRM systems.

Obviously a P2P system should also provide other security mechanisms such as access control, superdistribution, authentication etc.

To achieve access control and superdistribution, the content delivered between peers should be encrypted. Then the encrypted content can be freely distributed on the Internet and when the end-users get it, only the authorized ones can access that content.

Authentication is also required. From the perspective of users, they often want guarantees of integrity and sometimes non-repudiation on the received data. On the other side, the content provider does not want to be impersonated by another party. The source authentication is also necessary to prevent malicious attacks which frequently occur in many broadcast networks.

To address these problems, in the following a DRM architecture is proposed which considers both data encryption and authentication schemes, and can guarantee both security and P2P advantages.



3. The SEA Concept

SEA provides the necessary technology for content protection and rights protection, not only addressing professional content creators, but also user generated content to the users groups or individuals. New concepts how to deal with SVC and MVC coded contents are described in section 3.1. New media protection paradigms and solutions for P2P networking using a lightweight asset management to ensure, along with the management of the content protection that the content arrives to the right user and the full chain is respecting the required features and quality are covered in section 3.2.

3.1. Content protection & authentication of SVC/MVC media

SEA aims to provide an end-to-end solution for content protection management for IP and P2P networks, exploiting the full potential of the content protection and creator's rights maintenance. SEA has developed a beyond state-of-the art content protection technology, extending the ISMACryp content protection mechanism in a way that the ciphering technology can be applied to H.264 MVC/SVC encoder and decoder. The design supports both point-to-multipoint and point-to-point topologies in the sense of real time encoding and decoding.

Figure 5 shows the basic architecture of the encrypting server. The server can either capture live video or read media files stored in MP4 format [41][42], including its latest amendment "File format support for Scalable Video Coding" [43]. Depending on the use case, either the SVC or the MVC encoder is employed for video encoding.

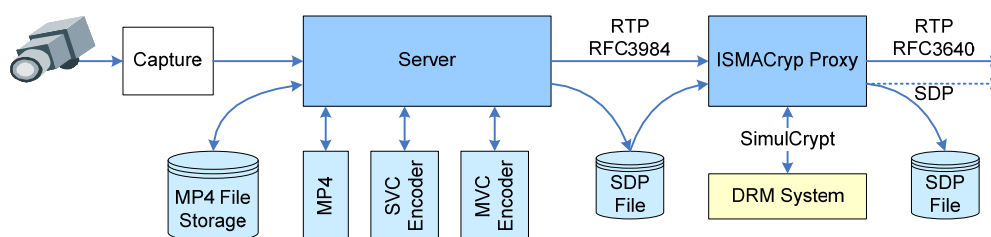


Figure 5. SEA encrypting server

The SEA server makes use of the RTP protocol [47] and the appropriate payload format standard for media streaming. For SVC video streams, RFC3984 [48] is used anticipating its recent extensions for SVC transport by the latest version of Internet Draft "RTP Payload Format for SVC Video" [50]. For MVC video streams, RFC3984 is also used anticipating future extensions for the transport of MVC video [51].

Transport related as well as media specific information is stored in an SDP file according to RFC4566 [53] also following the latest version of Internet Draft "Signaling media decoding dependency in Session Description Protocol" [54].

The media stream is then transmitted to the SEA encrypting proxy, which operates according to the ISMACryp standard [56]. The encrypted media stream is forwarded according to RFC3640 [57] using MIME type "enc-mpeg4-generic" specified by the ISMACryp standard (cf. subsection 3.1.1.1). The proxy provides a control interface according to the SimulCrypt standard [58]. Through this interface, different DRM systems can exchange information necessary to provide access for licensed users at the client side. The internal structure of the encrypting proxy is described in more detail in subsection 3.1.1, the DRM system and its functionality is described in subsection 3.1.2.

On the receiver side, a similar architecture is used for the decryption (see Figure 6). In order to inhibit any license violation, decrypting and decoding has to be applied in a trusted system (shaded area in



Figure 6). Due to the responsibilities of different partners, there are interfaces between different modules. The data is first received by a decrypting proxy, which is controlled by the client part of the DRM system (cf. subsection 3.2.2.4.3). If the user has purchased or by some other means obtained a license which allows decrypting, the decrypted stream is directly passed to the client decoder which decodes and displays the video. In case of layered video, different licensing may apply to different layers of the media stream, and the proxy forwards only those decrypted layers for which the user has obtained licenses. The interface between the client part of the DRM system and the decrypting proxy is similar to the one at the server side. More details on internal functionality and interfaces are described in subsection 3.1.1.2.4. The trusted system as a whole can be integrated in the same monolithic block in a commercial implementation.

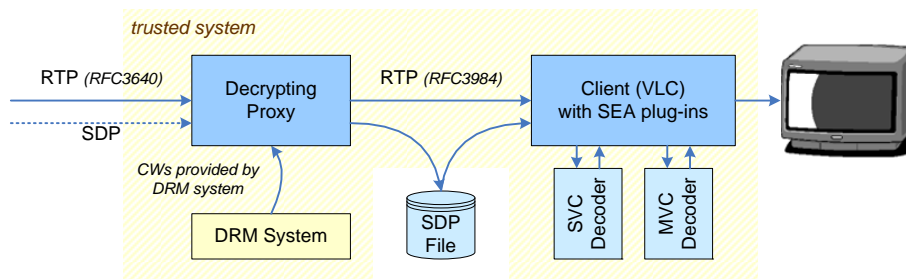


Figure 6. SEA decrypting client

3.1.1. Implementation of the SEA crypto framework

3.1.1.1 Encryption process

The ISMACryp standard [56] specifies media encryption based on RTP packets. Its transport packets comply with RFC 3640, RTP Payload Format for Transport of MPEG-4 Elementary Streams [57], which can be used for different MPEG-4 elementary streams, in particular MPEG-4 systems [44], video [45], and audio [46]. The MIME type defined in RFC 3640 for any of these elementary streams is "mpeg4-generic". The payload is structured in three sections as shown in the upper part of Figure 7:

1. Access Unit Header Section
2. Auxiliary Section
3. Access Unit Data Section

The structure of the Access Unit Header Section is shown in the lower part of Figure 7. It consist of a sequence of access unit headers, preceded by a length field and padded to the next bytes where required.

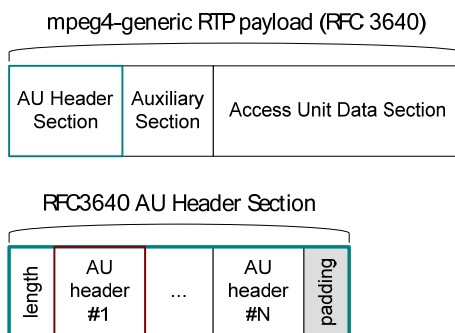


Figure 7. RTP payload type 'mpeg4-generic' (RFC 3640)



The MIME type for ISMACryp encrypted data is "enc-mpeg4-generic". In contrast to "mpeg4-generic" payload, the access units transported in the Access Unit Data Section (cf. upper part of Figure 7) with type "enc-mpeg4-generic" are encrypted with a 128-bit AES ciphering algorithm.

3.1.1.2 Proxy implementation

3.1.1.2.1 Server architecture

The SEA server architecture with a more detailed view inside the encrypting proxy is shown in Figure 9. The media data are received through RTP streaming sessions. The incoming session setup is read from an SDP file written by the server. SVC or MVC video streams, packetized according to [50] in "non-interleaved" mode according to RFC3984, are received by the ISMACryp Scrambler module. The scrambling algorithm is based on a Control Word (CW) which is generated in regular intervals by a Control Word Generator (CWG) module inside the proxy. The proxy is controlled by the SimulCrypt Synchronizer (SCS) module which

- transfers the CW to the Scrambler module,
- provides a control interface to the DRM system, and
- forwards Entitlement Control Messages (ECMs) from the DRM system to the client.

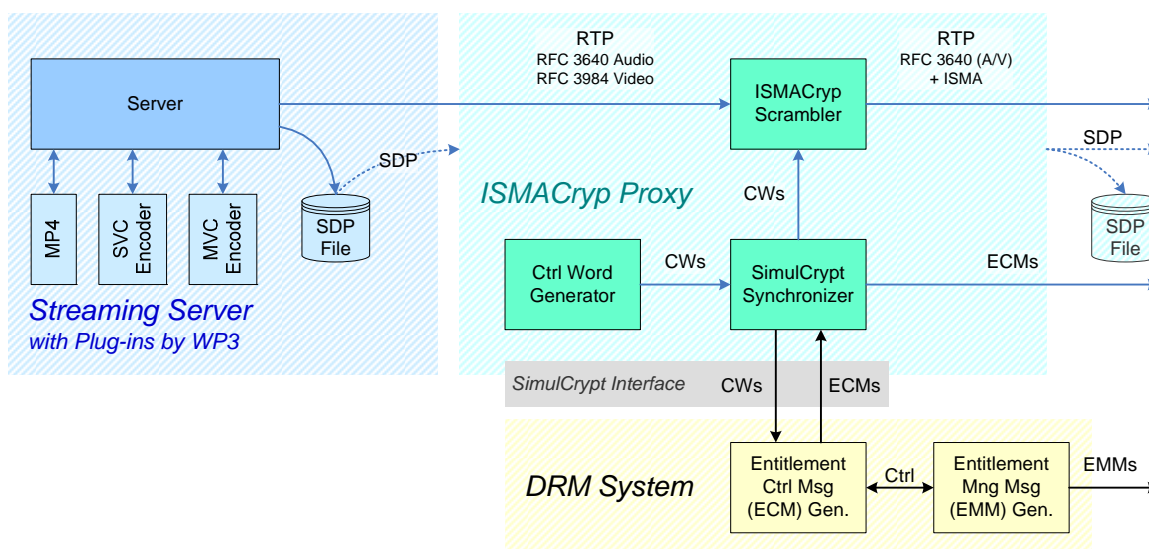


Figure 9. Server architecture with encrypting proxy

The SimulCrypt interface standard allows for connecting different Conditional Access Systems at the same time, each encrypting and managing the same CW with its own proprietary algorithm. If more than one ECMG is active, ECMs from each active ECMG are forwarded by the SCS module.

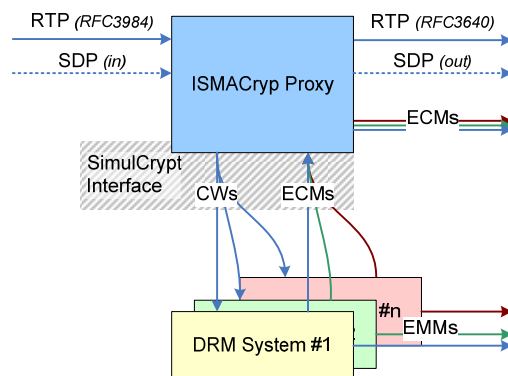


Figure 10. Connecting different CA systems through SimulCrypt

3.1.1.2.2 Encrypting proxy configuration

CWG and SCS are configured through XML files. The main ISMACryp configuration file parameters are listed in Table 2, where the following terms apply:

ISMACryp Stream ID:

The *ISMACryp Stream ID* uniquely identifies one single stream (e.g. a complete video stream or a single layer of an SVC stream, or a view of an MVC stream if conveyed in its own RTP session) that will be scrambled by the ISMACrypProxy.

Service ID:

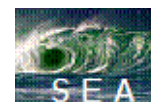
The *Service ID* uniquely identifies a service consisting of one or more ISMACryp Streams. All ISMACryp Streams with the same Service ID will be encrypted with the same CW.

Table 2 – ISMACryp Scrambler configuration parameters

Scope	Parameter	Description
all streams	TCP Control Ports	for stream setup and CW/key change
for each stream	ISMACryp Stream ID	
	SDP payload type	
	Input SDP file name	
	Output SDP file name	
	Destination IP address and destination ports	encrypted RTP, RTCP
	ISMACryp initialisation vector length	
	ISMACryp key indicator length	
	ISMACryp salt	

Table 3 – Simulcrypt Synchronizer configuration parameters

Scope	Parameter	Description
overall	Crypto Period duration	
for each ECMG	Super_CAS ID	
	ECM ID	



	ECMG IP address and port	TCP/IP for CW scrambling
	ECM channel ID	
for each service	Service ID	same as ECM stream ID
	ISMACryp Stream IDs	for each stream
	ISMACryp Control IP/Port	TCP/IP, for CW distribution
	Destination IP address and destination ports	for ECM encrypted CWs

3.1.1.2.3 Interface between SCS and ECMG

A detailed specification of the interface between SCS and ECMG is given in section 5 of the SimulCrypt standard [58]. The communication between these modules is illustrated by the following example. Basically, the SCS initiates the communication, and the ECMG answers to a request.

Initially, the ECMG can generate a new or retrieve the status of an already active ECM channel according to Table 4. New streams are added by the following command to each service according to Table 5. After each CP duration, a control word is provided for each service which is answered by an ECM datagram from the ECMG according to Table 6, where the CP_number corresponds to the ISMACryp key_indicator.

Table 4 – Messages to set up an ECM channel

Direction	Message
<i>SCS → ECMG</i>	channel_setup(ECM_channel_id, Super_CAS_id)
<i>ECMG → SCS</i>	channel_status(ECM_channel_id, section_TSpkt_flag, AC_delay_start, AC_delay_stop, delay_start, delay_stop, transition_delay_start, transition_elay_stop, ECM_rep_period, max_streams, min_CP_duration, lead_CW, CW_per_msg, max_comp_time)

Table 5 – Messages adding a stream to a service

Direction	Message
<i>SCS → ECMG</i>	stream_setup(ECM_channel_id, ECM_stream_id, ECM_id, nominal_CP_duration)
<i>ECMG → SCS</i>	stream_status(ECM_channel_id, ECM_stream_id, ECM_id, access_criteria_transfer_mode)



Table 6 –Control Word exchange

Direction	Message
<i>SCS → ECMG</i>	CW_provision(ECM_channel_id, ECM_stream_id, CP_number, CW_encryption, CP_CW_combination, CP_duration, access_criteria)
<i>ECMG → SCS</i>	ECM_response(ECM_channel_id, ECM_stream_id, CP_number, ECM_datagram)

3.1.1.2.4 Client architecture

The interfaces of the decrypting proxy on the receiver side are shown in Figure 11. If the DRM client receives an ECM and the user has been entitled sufficient rights for decoding and displaying the content, the DRM client decrypts the Control Word and sends an ECM containing the key indicator (KI) and the decrypted Control Word to the decrypting proxy. The message sent from the DRM Client to the decrypting proxy the same way as described in 3.1.1.2.3, Table 6. The decrypting proxy buffers the KI/CW pairs in a map. Based on the key indicator found in the RTP packet with encrypted payload the appropriate CW is fetched from this map and used for decryption.

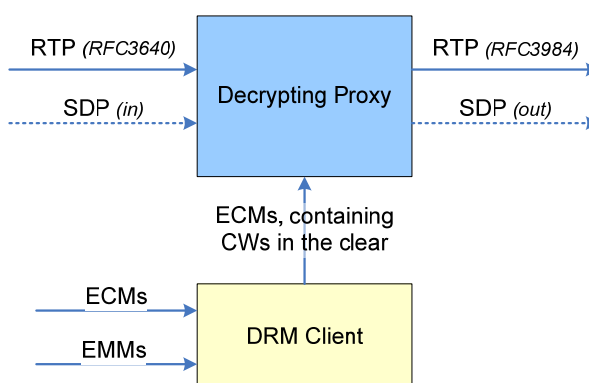


Figure 11. Decrypting proxy at receiver side

3.1.2. Conditional access system

3.1.2.1 State-of-the-art

The application of DRM to multimedia security has become a very important issue in the deployment of content distribution systems. The ability to enforce flexible DRM policies allows content owners/distributors to overcome several limitations of existing systems.

Encryption is a key enabling technology, as it guarantees that only the users that know a secret key are going to be able to decrypt and display the multimedia content. Recent techniques such as advanced encryption standard (AES) allow protecting data communications in a reliable way, providing a high degree of security with a reasonable computational cost.

Nevertheless, encryption of a multimedia file has to be carried out carefully. On one hand, ciphering the complete compressed file may result in excessive computational burden and/or power consumption at the decoder. While this is not critical for computer-based systems, it can lead to



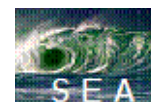
excessive battery drain on a handheld device. Therefore, particular care must be taken in a heterogeneous system, such as the SEA framework, in order to design the security functionalities in a way that is compatible with all types of terminals expected to use the network. Even more importantly, multimedia compressed files typically exhibit well-defined hierarchical structures that can be exploited in several ways, e.g. for scalability, transcoding, rate shaping, and so forth. However, these structures may not be recognizable in the ciphertext, potentially limiting the adaptation and hindering the efficient operation of a complex system.

Prior to the advent of scalable video coding standards such as H.264/SVC, there have been several attempts to design systems that process the video information in the compressed domain, as this domain is more amenable to providing enhanced functionalities. As outlined in [59], multimedia encryption can be carried out at different stages of the compression process, namely the original pixels, the transform coefficients, the quantization indexes, the bit-planes, the entropy coder, or the final codestream. Particular care should be taken to select the most appropriate place to perform encryption, in order to obtain the desired functionality and avoid performance losses. For example, encrypting data *before* the entropy coder may result in a coding efficiency loss due to the modified data statistics.

A possible application of these ideas is in the field of “selective encryption”, in which only some parts of the compressed files are encrypted or otherwise protected. The concept of “partial” or “selective” encryption has been formalized in [60], where it is proposed to cipher only the visually most significant information in order to prevent display of the video by an unauthorized user. Another interesting application, namely “conditional access”, has been proposed in [61]. In conditional access, a low-quality version of the video is left in the clear, and can be used to preview the multimedia content; the user can purchase a key, and then decode the content at full quality. In this document, though, we use the term “conditional access” in a broader sense. In particular, we define *conditional access* as the feature that enables the SEA system to allow only selected parts of the multimedia stream to be displayed by the user terminal. In the SEA system, we identify as the *conditional access system* a decision module located at the server, which makes decisions about which parts of a multimedia stream will be encrypted (or left in the clear).

These and other features can be achieved in several different ways, according to where and how the ciphering process takes place. Although ciphering can be done directly on the image pixels, e.g. on their least significant bit-planes as in [62], or applying a secret transform to each pixel before compression [63], this usually leads to major compression efficiency losses. Several authors have investigated various algorithms that operate in the transform domain or the colorspace domain. E.g., in [64] the signs of some wavelet coefficients are randomly changed. In [65][66][67] the discrete cosine transform (DCT) coefficients of MPEG video, along with the motion vectors and color planes, are changed; in [68][69] the DCT coefficients are shuffled, while in [70] they are modified adding or subtracting a constant. In [71] wavelet code-blocks are shuffled during the JPEG 2000 encoding process. In [72] the motion vectors are also scrambled. In [73] different wavelet packet transforms are used to hide the original information, whereas in [74] different wavelet filters, belonging to the same family, are used in JPEG 2000. A few authors [59][61] propose algorithms based on modifications of the bit-planes of the transformed coefficients; this is particularly useful for JPEG 2000, as it allows a high degree of flexibility in the design of the protection scheme and its functionality, since the entropy coding is done bit-plane by bit-plane.

A large body of work has also been made with the objective of performing encryption in conjunction with the entropy coding stage. In [75] it is suggested to map the syntactical elements to indexes in a table, to encrypt the indexes, and to use the resulting integers as indexes of new codewords; index mapping is also proposed in [59]. Another possible approach, which avoids any compression performance loss, is based on the idea of encrypting fixed-length parts of entropy-coded codewords, such as the eight-bit Huffman fields in JPEG [62], or some bits of Exp-Golomb codes [76][66][77], which are used in many standards, including H.264/SVC, to encode all but the quantized DCT coefficients. A potential problem with these techniques is that the encrypted codewords can be individually extracted from the codestream, facilitating an attacker in the task of deciphering them.



This can be avoided using secure Huffman codes, [78][79][80]; in this case the ciphered codewords have variable lengths, and hence cannot be straightforwardly extracted from the codestream. However, Huffman codes are memoryless, and this facilitates possible attacks. Another option is to adopt secure arithmetic codes [81][82][83][84]; the basic versions of these codes (e.g. [81]) generate a compressed file whose expected size is the same as that of the non-encrypted file. Several researchers have also investigated the option of encrypting the information after entropy coding. For example, in [85][86] it is proposed to iteratively encrypt portions of the compressed file until marker emulation is completely avoided; in [87] it is proposed to packetize the compressed data so that encryption does not hinder further processing, e.g. transcoding.

When employing an international standard, the issue of syntax compliance of the encrypted file should also be taken into account. Syntax compliance is important because the syntax often allows for compressed-domain processing. For example, a compressed JPEG 2000 image or H.264/SVC video sequence can undergo rate adaptation by simply discarding a few quality layers; this only requires parsing the compressed file and looking for the relevant data structures to be kept or discarded. To preserve these functionalities, encryption must be performed in such a way as to not disrupt the compressed file structures that allow this kind of processing. In the ideal case, the encryption should preserve the syntax of the compressed file, i.e., it should only encrypt the compressed data, but not the headers that allow parsing the compressed file without decoding the content. Full syntax compliance has the advantage of not requiring an encryption stage external to the video encoder/decoder. However, this is generally quite difficult to achieve and not always necessary. Moreover, syntax compliance is typically achieved at the expenses of some performance loss (see e.g. [88]), and/or imposing some requirements on the design of the entropy coding stage. To avoid that, it is possible to decouple the scalable coding and encryption stages, i.e. the video encoder generates different codestreams (e.g. quality layers) that are encrypted separately using an appropriate cryptosystem.

3.1.2.2 Specification of conditional access functionalities

In light of the definition of conditional access techniques for H.264/SVC and H.264/MVC in the framework of the SEA system, which is the subject of this section, the following remarks can be made. The main feature of these standards is to deal with “multi-part” video sequences. In particular, in the following we will denote “layers” (H.264/SVC) or “views” (H.264/MVC) of a compressed video sequence as the “parts” that have to be protected using encryption and DRM. The abstraction of considering a “part” of the video allows decoupling the design of a suitable encryption technique, and the development of a proper DRM system, from the advanced functionalities achieved through encryption. For a user, receiving and being able to decrypt and decode one or more parts will yield different functionalities, e.g. improved quality or a larger number of views. From the application standpoint, even though decoding of multiple parts may have to be done jointly, the encoding and transmission stages are completely separated for each file part. Therefore, it follows that DRM can be performed by selectively allowing access to each part of the video sequence according to the rights that are going to be granted to each user.

As for encryption, the ISMACryp cryptographic framework employed in the SEA system has been described in Sect. 3.1, while the DRM system is the subject of Sect. 3.2. Regarding the interface between the cryptographic primitives and the conditional access functionalities, in order to maintain this system as flexible as possible, it has been chosen to perform post-compression encryption. The resulting encrypted file is not syntax-compliant with the H.264/SVC or H.264/MVC standards, but the preliminary decryption stage yields a compliant file whenever a valid decryption key is available. Care must be taken, during the decryption/decoding process, to forward to the VLC decoder only the decrypted layers/views.

It is assumed that the multimedia stream consists of L parts P_1, P_2, \dots, P_L . These can be layers of H.264/SVC (e.g., spatial scalability, temporal and/or quality scalability layers) or views of H.264/MVC. Each part P_i is protected independently, including the option to not protect some or all parts of a multimedia stream. The encryption and DRM systems provide all the functionalities



required to feed the decoder with the decrypted versions of all parts that she is allowed to decode and display. The conditional access system can perform one or more of the following functionalities:

1. Leave all parts of the multimedia stream in the clear. This is the case of content that does not need to be protected.
2. Encrypt all parts of the multimedia stream. This is performed if the maximum degree of security is required, and if all terminals expected to use the multimedia stream have the capabilities of decrypting the stream in real time.
3. Encrypt parts P_1, P_2, \dots, P_S of an H.264/SVC quality layer, leaving parts $P_{S+1}, P_{S+2}, \dots, P_L$ in the clear. This “selective encryption” mode is intended to support low-power decoding terminals, which may not have the computational capabilities to decrypt the multimedia stream in real-time. The selective encryption encodes only the base layer (plus possibly a few enhancement layers). The rationale is that an unauthorized user that cannot decode the base layer, will also be unable to decode the enhancement layers. Thus, encrypting only a fraction of the multimedia stream protects the whole stream.
4. Leave parts P_1, P_2, \dots, P_S of an H.264/SVC in the clear, while encrypting parts $P_{S+1}, P_{S+2}, \dots, P_L$. This “video preview” mode does not perform any encryption on the base layer (plus possibly a few enhancement layers), allowing any user to decode a low-quality version of the video for preview purpose. Upon deciding to be interested in the content, the user can purchase, through the DRM system, the rights to decode the complete stream. This mode allows a significant degree of flexibility, as it embeds the preview feature seamlessly in the SEA system, without the need of an additional external component.
5. Encrypt a given set of parts part P_i , with $i \in A$, and leave in the clear a given set of parts part P_j , with $j \in B$. A and B must form a partition of the set $\{1, \dots, L\}$ that specifies in an arbitrary way what parts of the multimedia stream must be encrypted. This mode encompasses all the previous modes as sub-cases.

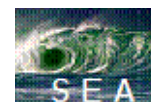
We end this section with an example of application of the conditional access system. In the example, a user is interested in viewing a movie. The movie is encoded using H.264/SVC with four layers, namely 2 quality scalability layers P_1 and P_2 at low/high quality and QCIF spatial resolution, and two spatial scalability layers P_3 and P_4 bringing the resolution of the first two layers to VGA. The quality layers are used for rate adaptation, while the spatial layers for dealing with heterogeneous terminals with different display capabilities. A possible application of the conditional access system is to encrypt layers P_2, P_3 and P_4 , leaving P_1 in the clear. As a result, any user will be able to display the movie in low-quality QCIF format, but only authorized users will be able to display the full-quality VGA video.

3.2. P2P Content management

SEA project aims to establish new media protection paradigms and solutions for P2P networking using a lightweight asset management. The solution proposed is based on the creation of a secure and adaptable content delivery architecture and the underlying mechanisms to ensure the correct content management which along with the content protection mechanisms can be useful for, on the one hand ensure user privacy (as the P2P content goes through many nodes) and on the other hand enable the possibility of offering commercial IPTV services over a P2P environment. In the following subsections different architectures are described, and the solutions used are described in detail, including all the elements forming the system.

3.2.1. The proposed architecture for secure and adaptable content delivery

As is has been mentioned in section 2.2.1, distributed P2P architecture clients do not establish connection with a server and download content.



This network architecture entails serious security problems. Once content is introduced into the network, control over it is lost. Everyone can get and consume it without permission or authorization. This model is inefficient if we want to encrypt and have consumption control over a set of media content. Digital content can be copied and redistributed without any restriction, which is undesired. Therefore, a content management system, in which all the media data that we want to protect should be encrypted, is required.

The encryption of the content entails a complex management system. A lot of modules with different functions are necessary to develop a secure system in which all contents are available through the network.

A system with the following functions should be implemented:

- Register users and hold an updated database with the registered users and their contents.
- Generate licenses
- Identify the licenses that belong to the contents.
- Generate the key used to encrypt and decrypt each piece of content.

For all these reasons, the proposed architecture for the SEA project is a semi-distributed P2P based architecture, in which the above critical functions exist in the separated DRM server while all others lie in peer nodes.

We propose an SVC/MVC content management and secure sharing system. The SEA system is divided into server side and client side. To avoid building up the workload, we propose a light-weight system. The server has the following basic functions:

- 1) Processing peers' registry
- 2) Generating and issuing ECM messages
- 3) Providing the EMM to the peer who has downloaded the whole content
- 4) Managing data base with the information of all contents related with its licenses.

The first and second functions do not add as much payload onto servers as content storage and download services do. Therefore, compared to centralized system, we eliminate the function of content storage and download services, but compared to a distributed system we take rights of issuing licenses and publishing content information back from peers to the server side. We include the key management system in the server, so that it has all the functionalities the client needs to manage its content.

Client, in the proposed system, is a normal peer with a list of functions:

- a) Encrypting content
- b) Generating keys for encrypting the content
- c) Creating and sending licenses to the server
- d) Reading and interpreting licenses
- e) Providing contents to other peers

Figure 12 shows the system that we proposed for content protection and key management system.

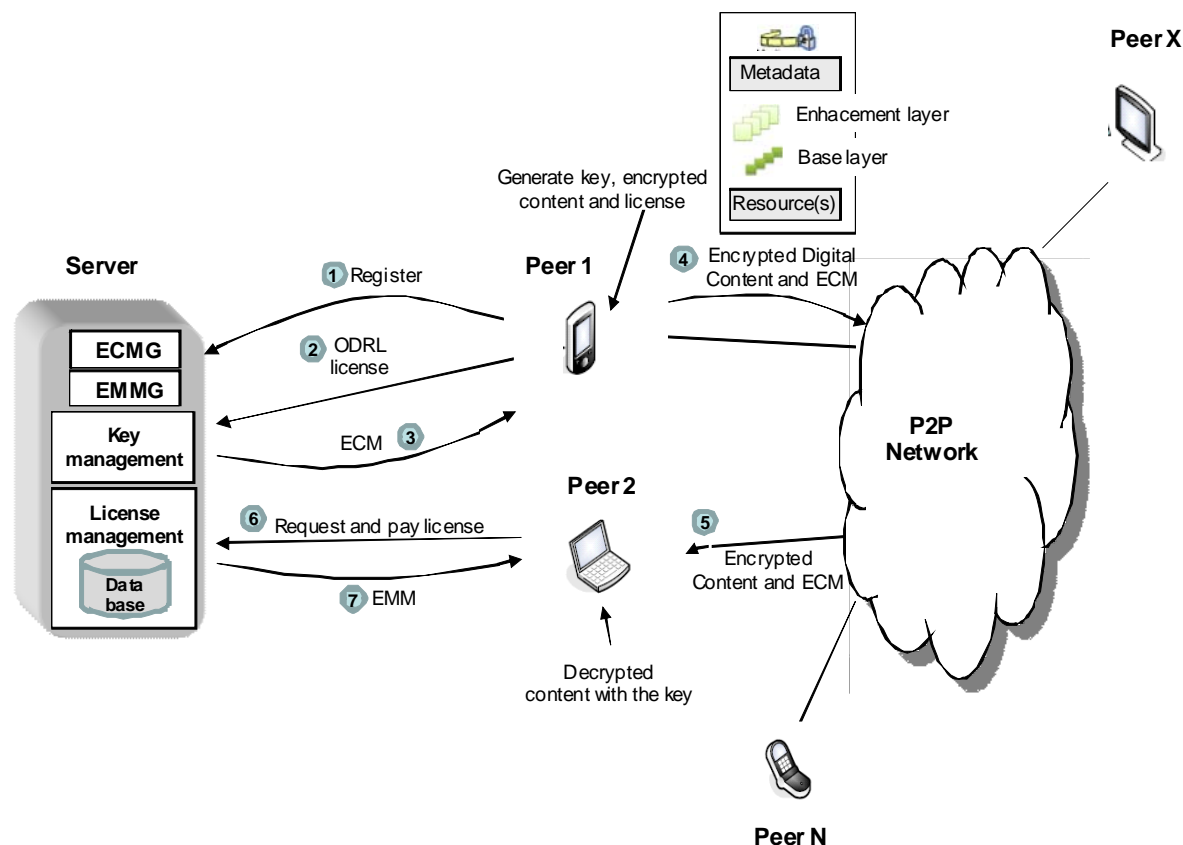
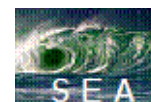


Figure 12. Content protection and key management system overview

The principal system modules are the ECM generator, the EMM generator, the key management system and the license management system. And, of course, it must be taken into account the importance of creating an interoperable system with other similar systems, terminals or networks. This interoperability has been studied and will be explained in section 3.2.3.

The ECM generator creates an ECM message which includes the control word that has been used to encrypt the content with ISMACryp. This module also communicates with the SCS to create a channel to send these messages.

The content of the ECM is used to establish if a user has access to the content or not.

The EMM messages help to decide the rights of a user. These messages are generated by the EMM generator. An EMM contains the license which indicates the actions that a user can take upon a specified content.

The key management system module makes the decisions related to the keys involved in the encryption of the content.

Finally, a module is needed that creates licenses with the appropriate permissions and manages these licenses to provide the correct license of a content when a user sends a request and pays for it (in case he needs to pay). The module responsible for doing that is the license management system.

To help accelerate peer discovering and content downloading we use a data base which is stored and published in the server side.

All these modules will be explained in more detail in the following sections. Before, we present an example of the functionality of the P2P content management system.

A normal process of file sharing is shown in Figure 12:



- 1) Peer 1 has a content to share. First it registers to the server. The server will create at this moment a *user_ID*.
- 2) Peer 1 generates metadata (e.g. file name, file type, file length), and sets access rights to the content (e.g. the enhancement layer that the user can access). With this entire information peer 1 creates the license in XML and sends it to the server.
- 3) Server identifies the content and creates an ECM which is send to the user.
- 4) At this point, Peer 1 is able to encrypt the content. At the same time, the server stores the license in the database and associates it with the content. A license consists of right object, client's information (*user_ID*, *content_ID*) and encryption key. Right object is expressed in ODRL. The Server publishes content information on the Data Base to let other peers know that Peer 1 has issued a piece of content.
- 5) Peer 2 discovers from the data base that a piece of content is being distributed. It searches and obtains encrypted content in regular P2P manner.
- 6) At this moment, Peer 2 asks the server for a license.
- 7) The server sends an EMM message that contains the license to Peer 2. Peer 2 decrypts the content with the key included in the ECM that can be decrypted with the key of the EMM message, and executes the content as the right object authorizes it to.

The DRM system consists of a server agent with several function functions (see Figure 13). First of all, as it has been mentioned in section 3.1.1.2, the system must generate ECM messages by means of the ECM Generator (ECMG) and send them to the SCS through the SCS-ECMG interface. On the other hand, it must generate EMM messages by means of the EMM Generator (EMMG), which will be explained in section 3.2.2.1. Finally, the DRM server must act as a license management agent, keeping a list of all the available licenses in order to let the clients know which licenses they have purchased, modifying the content of the license. This last function is performed by a Data Base, which will be explained in depth in section 3.2.2.4.2.1.

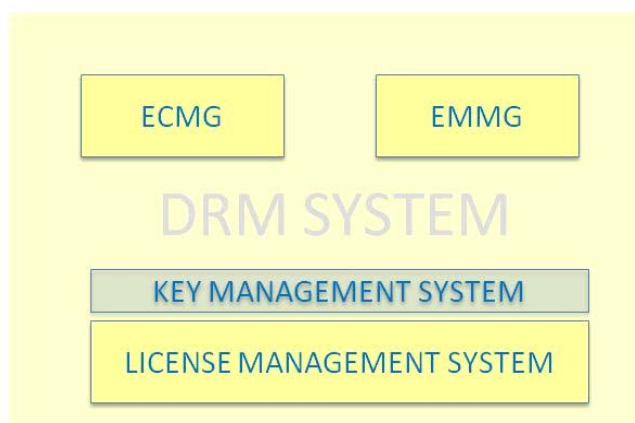


Figure 13. DRM System

3.2.2. Solution implementation

For the implementation, the VLC media player (www.videolan.org) has been chosen, which is a highly portable multimedia player supporting various audio and video formats as well as various streaming protocols. To complete our system, we need to add some plug-ins into the VLC media player. Firstly, we add an SVC codec plug-in to the client's VLC media player. Thus, the content is available in several resolutions to satisfy different clients' needs. We also add a plug-in to the VLC to make it able to create and understand the licenses and execute the content as the right object authorizes to. Other fundamental action is included in the client environment. It consists in a proxy



that adds ISMACryp encrypting and decrypting capabilities to the VLC. Figure 14 shows the implementation of the DRM system in the SEA project.

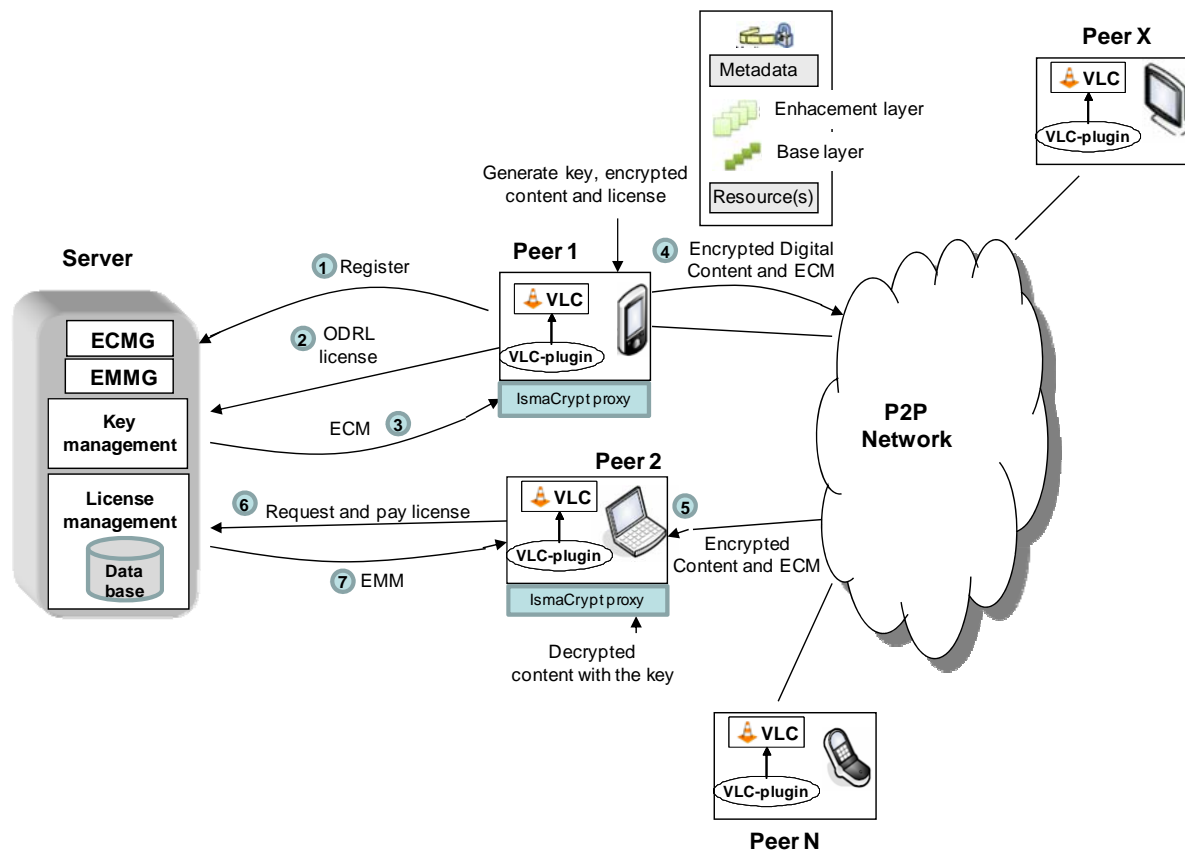


Figure 14. Solution implementation for the content protection and key management system

3.2.2.1 Entitlement Control Message Generator (ECMG)

In this section, the main functions of the ECMG will be explained, extending what has been mentioned in section 3.1.1.2, in order to provide a clearer vision of this particular module.

As the interface between the SCS and the ECMG has already been specified, this section will focus on the different messages that are exchanged, especially on the `ECM_response_message`, which ECM datagram and its encryption.

In Figure 15 the most important messages exchanged through the SCS-ECMG interface are shown.

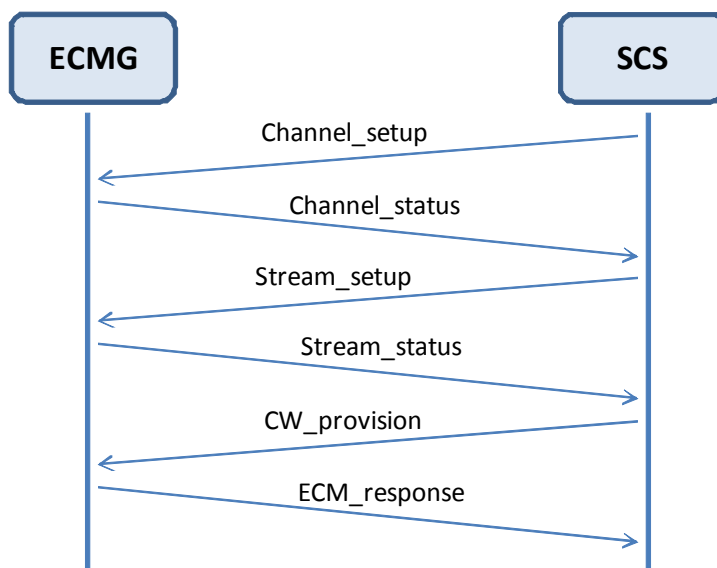


Figure 15. Messages exchanged between the SCS and the ECMG

3.2.2.1.1 Channel and stream specific messages

The messages exchanged between the SCS and ECMG modules are [58]:

1. Channel establishment

There is always one (and only one) channel per TCP connection. Once the TCP connection is established, the SCS sends a channel_setup message to the ECMG. In case of success the ECMG replies by sending back a channel_status message.

In case of a rejection or a failure during channel setup the ECMG replies with a channel_error message. This means that the channel has not been opened by the ECMG and the SCS shall close the TCP connection.

- Channel_setup message: ECMG \leftarrow SCS

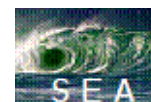
The channel_setup message is sent by the SCS to setup a channel once the TCP connection has been established. It shall contain the Super_CAS_id parameter, to indicate to the ECMG to which CA system and subsystem the channel is intended (indeed, there could be several Super_CAS_ids handled by a single ECMG host).

- Channel_status message: ECMG \leftrightarrow SCS

The channel_status message is a reply to the channel_setup message or the channel_test message.

When the message is a response to a setup, the values of the parameters are those requested by the ECMG. All these parameter values will be valid during the whole lifetime of the channel, for all the streams running on it.

When the message is a response to a test, the values of the parameters shall be those currently valid in the channel.



2. Stream establishment

The SCS sends a stream_setup message to the ECMG. In case of success, the ECMG replies by sending back a stream_status message. In case of a rejection or a failure, the ECMG replies with a stream_error message.

Once the connection, channel and stream have been correctly established the ECM will be transferred. It can be transferred as sections or as TS packets. The ECMG indicates at channel setup which kind of data object will be used.

- Stream_setup message: ECMG \leftarrow SCS

The stream_setup message is sent by the SCS to setup a stream once the channel has been established.

- Stream_status message: ECMG \leftrightarrow SCS

The stream_status message is a reply to the stream_setup message or the stream_test message.

When the message is a response to a setup, the value of the access_criteria_transfer_mode parameter is the one requested by the ECMG.

When the message is a response to a test, the values of the parameters shall be those currently valid in the stream.

3. Data exchanged

Once the connection, channel and stream have been correctly established, ECMs will be transferred to the SCS as a response to the CW_provision message.

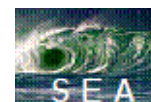
- CW_provision message: ECMG \leftarrow SCS

CW_provision message is sent by the SCS to the ECMG and serves as a request to compute an ECM. The value of the CP_number parameter is the Crypto period number of the requested ECM. The control words are carried by this message with their associated Crypto period numbers in the CP_CW_combination parameter, according to the value of lead_CW and CW_per_msg as defined during the channel setup.

The specific CWs that are passed in the CP_CW_combination to the ECMG via the CW_provision message are derived from the values of lead_CW and CW_per_msg.

- ECM_response message: ECMG \Rightarrow SCS

The ECM_response message is a reply to the CW_provision message. It carries the ECM datagram, computed by the ECMG, from the information provided by the CW_provision message. The ECM datagram carries the CW encrypted with a Service Key (SK) as it is explained in section .



The value of the CP_number parameter shall be the same in the replied ECM_response message as in the previous incoming CW_provision message.

The time-out for the ECM_response message shall be computed by the SCS from the max_comp_time value defined during channel setup, and the typical network delays.

4. Stream closure

Stream closure is always initiated by the SCS. This can occur when an ECM stream is no longer needed or in the case of an error. This is done by means of a stream_close_request message. A stream_close_response message indicates the stream has been closed.

- Stream_close_request message: ECMG \leftarrow SCS

The ECM_stream_id is sent by the SCS in the stream_close_request message to indicate which of the streams in a channel is due for closure.

- Stream_close_response message: ECMG \Rightarrow SCS

The ECM_stream_id is sent by the ECMG in the stream_close_response message to indicate which of the streams in a channel is closing.

Message	Parameters
Stream_close_request	ECM_channel_id ECM_stream_id
Stream_close_response	ECM_channel_id ECM_stream_id

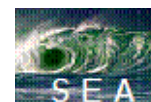
5. Channel closure

Channel closure can occur when the channel is no longer needed or in case of error (detected by SCS or reported by ECMG). This is done by means of a channel_close message sent by the SCS. Subsequently, the connection shall be closed by both sides.

- Channel_close message: ECMG \leftarrow SCS

The channel_close message is sent by the SCS to indicate the channel is to be closed.

Message	Parameters
Channel_close_message	ECM_channel_id



6. Channel/Stream testing and status

At any moment either component can send a channel_test/stream_test message to check the integrity of a channel/stream. In response to this message the receiving component shall reply with either a channel/stream status message or a channel/stream error message.

- Channel_test message: ECMG ⇔ SCS

The channel_test message can be sent at any moment by either side to check:

- the channel is in an error free situation
- the TCP connection is still alive

The peer shall reply with a channel_status message if the channel is free of errors or a channel_error message if errors occurred.

- Stream_test message: ECMG ⇔ SCS

The stream_test message is used to request a stream_status message for the given ECM_channel_id and ECM_stream_id. The stream_test message can be sent at any moment by either entity. The peer shall reply with a stream_status message if the stream is free of errors or a stream_error message if errors occurred.

- Channel_error message: ECMG ⇔ SCS

A channel_error message is sent by the recipient of a channel_test message or by the ECMG at any time to indicate that an unrecoverable channel level error occurred.

- Stream_error message: ECMG ⇔ SCS

A stream_error message is sent by the recipient of a stream_test message or by the ECMG at any time to indicate that an unrecoverable stream level error occurred.

Message	Parameters
Channel_test_message	ECM_channel_id
Stream_test_message	ECM_channel_id ECM_stream_id
Channel_error_message	ECM_channel_id error_status error_information



Stream_error_message	ECM_channel_id ECM_stream_id error_status error_information
----------------------	--

7. Unexpected communication loss

Both SCS and ECMG shall be able to handle an unexpected communication loss (either on the connection, channel or stream level).

Each component, when suspecting a possible communication loss (e.g. a 10 second silent period), should check the communication status by sending a test message and expecting to receive a status message. If the status message is not received in a given time (implementation specific) the communication path should be re-established.

8. Handling data inconsistencies

If the ECMG detects an inconsistency it shall send an error message to the SCS. If the SCS receives such a message or detects an inconsistency it may close the connection. The SCS (as the client) will then (re-)establish the connection, channel and streams.

3.2.2.1.2 Security in ECMG ↔ SCS protocol

The control words conveyed in the CP_CW_combination parameter within the CW_provision message constitute the clear cryptographic keys that are used to directly scramble content. Knowledge of these keys by unauthorized agents can result in the compromise of the security of the system.

Therefore the SEA implementation of the DRM system employs effective and appropriate methods to preserve the confidentiality of the control words traversing this interface. The SEA approach uses a control word encryption scheme based on the Data Encryption Standard [89] to deliver the CW to the ECMG in a secure manner.

3.2.2.2 Entitlement Management Message Generator (EMMG)

The EMMG have the task of generating EMM messages. These messages are exchanged between the client who wants to acquire the license to see a content, and the server where all the licenses of the respective contents are stored. In Figure 16 is shown the described situation.

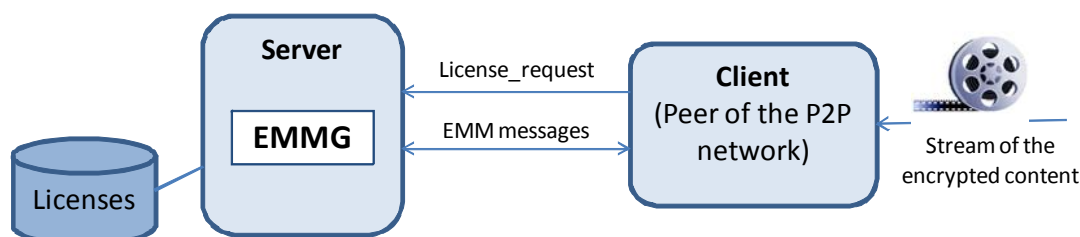
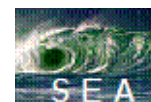


Figure 16. Entitlement Management Message Generator

Before sending an EMM message with the license is necessary to establish a channel connection. For this task we follow the specification [58] which describes the messages that have to be sent.



The principal messages necessary for the exchange of data consists in a channel_setup message send by the EMMG after receiving a license request from a client. This message will be replied by a channel_status message. Then the EMMG will send a stream_setup message and the client will response with a stream_status message. At this moment the connection has been establish and the EMMG can send the data with the license and key needed to decrypt the CW included in the ECM message associated with the content. These exchanges of messages are illustrated in Figure 17.

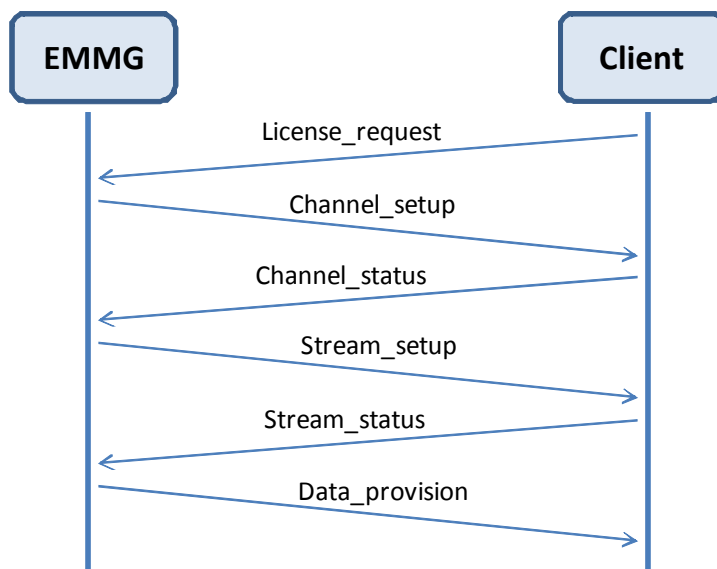


Figure 17. Messages exchanged between the Client and the EMMG

We will see in more detail the functionality of each of the messages implicated in the communication between the EMMG and the client (the license_request message will be described in section 3.2.2.4).

3.2.2.2.1 Channel and stream specific messages

1. Channel establishment

The EMMG sends a channel_setup message to the client. In case of success the client replies by sending back a channel_status message.

In case of a rejection or a failure during channel setup the client replies with a channel_error message. This means that the channel has not been opened by the client and the EMMG shall close the TCP connection.

- Channel_setup message: EMMG ⇒ Client

The channel_setup message is sent by the EMMG to the client to setup a channel once the TCP connection has been established.

- Channel_status message: EMMG ⇐ Client

The channel_status message is a reply to the channel_setup message or the channel_test message. All the parameters listed are mandatory.

When the message is a response to a setup, the values of the parameters are those requested by the EMMG and currently stored in the client. All these parameter values will be valid during the whole life time of the channel, for all the streams running on it.



When the message is a response to a test, the values of the parameters shall be those currently valid in the channel.

Message	Parameters
Channel_setup	client_id data_channel_id section_TSpkt_flag
Channel_status	client_id data_channel_id section_TSpkt_flag

2. Stream establishment

After the channel establishment the EMMG sends a stream_setup message to the client. In case of success the client replies by sending back a stream_status message. In case of a rejection or a failure the client replies with a stream_error message.

When a new EMM/private data stream is created in a transport stream, a new data_id shall be assigned to it by the CAS, according to the operational context (EMM/private data stream creation or EMMG replacement). The value of the data_id parameter remains unmodified as long as the EMM/private data stream exists. The combination {stream type + client_id + data_id} identifies uniquely this new EMM/private data stream in the whole system.

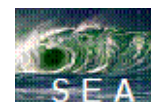
- Stream_setup message: EMMG ⇒ Client

The stream_setup message is sent by the EMMG to the client to setup a channel once the channel has been established.

- Stream_status message: EMMG ⇐ Client

The stream_status message is a reply to the stream_setup message or the stream_test message. The values of the parameters shall be those currently valid in the stream.

Message	Parameters
Stream_setup	client_id data_channel_id data_stream_id data_id data_type
Stream_status	client_id data_channel_id data_stream_id data_id data_type



3. Data provision

Once the connection, channel and stream have been correctly established the EMMs/private data will be transferred. They can be transferred as sections or as TS packets. The EMMG indicates at channel setup which kind of data object will be used. The EMMs/private data shall be inserted in the transport stream in the same order as they are provided by the EMMG.

- Data_provision message: EMMG ⇒ Client

The data_provision message is used by the EMMG to send the datagram, on a given data_stream_id. The datagram is the EMM/private data. In this project, this data is the license of the content the client wants to view and the key that will decrypt the CW which can decrypt the content.

Message	Parameters
Data_provision	client_id data_channel_id data_stream_id data_id datagram

4. Stream closure

Stream closure is always initiated by the EMMG. This can occur when an EMM/private data stream is no longer needed. This is done by means of a stream_close_request message. A stream_close_response message indicates the stream has been closed.

- Stream_close_request message: EMMG ⇒ Client

The stream_close_request message is sent by the EMMG to indicate the stream is to be closed.

- Stream_close_response message: EMMG ⇐ Client

The stream_close_response message is sent by the client indicating the stream that is being closed.

Message	Parameters
Stream_close_request	client_id data_channel_id data_stream_id
Stream_close_response	client_id data_channel_id data_stream_id



5. Channel closure

Channel closure can occur when the channel is no longer needed or in case of error (detected by EMMG or reported by the client). This is done by means of a channel_close message sent by the EMMG. Subsequently, the connection shall be closed by both sides.

- Channel_close message: EMMG ⇒ Client

The channel_close message is sent by the EMMG to indicate the channel is to be closed.

Message	Parameters
Channel_close_message	client_id data_channel_id

6. Channel/Stream testing and status

At any moment either component can send a channel_test/stream_test message to check the integrity of a channel/stream. In response to this message the receiving component shall reply with either a channel/stream status message or a channel/stream error message.

- Channel_test message: EMMG ⇔ Client

The channel_test message can be sent at any moment by either side to check:

- the channel is in an error free situation
- the TCP connection is still alive

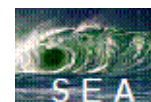
The peer shall reply with a channel_status message if the channel is free of errors or a channel_error message if errors occurred.

- Stream_test message: EMMG ⇔ Client

The stream_test message is used to request a stream_status message for the given client_id, data_channel_id and data_stream_id. The stream_test message can be sent at any moment by either entity. The peer shall reply with a stream_status message if the stream is free of errors, or a stream_error message if errors occurred.

- Channel_error message: EMMG ⇔ Client

A channel_error message is sent by the recipient of a channel_test message or by the client at any time to indicate that an unrecoverable channel level error occurred.



- Stream_error message: EMMG ↔ Client

A stream_error message is sent by the recipient of a stream_test message or by the client at any time to indicate that an unrecoverable stream level error occurred.

Message	Parameters
Channel_test_message	client_id data_channel_id
Stream_test_message	client_id data_channel_id data_stream_id
Channel_error_message	client_id data_channel_id error_status error_information
Stream_error_message	client_id data_channel_id data_stream_id error_status error_information

3.2.2.3 Key management system

In the proposed DRM system, two different set of keys are implicated. One of them is needed to encrypt the CW generated by the IsmaCryp Proxy. This encryption is done in the ECMG module. The other set of keys is needed to encrypt the license. This is done in the license management system.

The key management involves the generation, selection, and distribution ok key data to be used in the algorithm for the encryption.

3.2.2.3.1 Key management for the CW encryption

For providing the system with more security, the CW used to encrypt the content is also encrypted. This way it is very difficult that a user can view a piece of content if he does not have the suitable rights.

In sections 3.2.2.3.1.1, 3.2.2.3.1.2 and 3.2.2.3.1.3 the process followed in the SEA project for the key generation, key selection and key distribution will be explained.

3.2.2.3.1.1 Key generation

The encryption algorithm implementation uses key data generated by a good random source and stored on media for use in the ECMG device.

In the event a key list is suspected or known to be compromised, the ECMG device is responsible for generating and distributing a replacement list as soon as possible. Moreover, the ECMG may generate and distribute new key lists on a periodic basis to insure key security.

The suggested size of the key-space is 2,048 bytes. This provides 292 possible 7-byte keys per key list for use in the algorithm.



Highly-cautious ECMG device may wish to detect and exclude weak and semi-weak keys when generating key-lists, however due to the nature of the application, the occasional use of these keys does not pose a security threat. Moreover, if the random source is robust, neither weak nor semi-weak keys are likely to be generated with any significant probability.

To facilitate seamless transition from one key list to another, two independent 2,048-byte lists are used. Both the active list and the future list shall reside on all Simulcrypting ECMGs. The list in current use shall be identified using the most significant bit in the CW_encryption parameter; this bit is designated as key_list_sel (key list select bit). When reset (0), the A key list is in use; when set (1), the B list is in use. Transition from one key list to the other is accomplished by setting or resetting the key_list_sel bit.

In order to avoid placing headers in the key lists, the ECMG software shall examine the key_list_sel bit at the time lists are loaded to determine whether the list being loaded is to be designated the A or B list. If the active list is A, the list is loaded as B; if the active list is B, it is loaded as A. If no lists are active (during initialization) the software shall allow the device to manually enter the designator.

3.2.2.3.1.2 Key selection

Selection of the 56-bit key data for use in the algorithm is accomplished by using a randomly-generated 11-bit key select vector. This vector is used as an index into the 2,048 key space as shown in Figure 18.

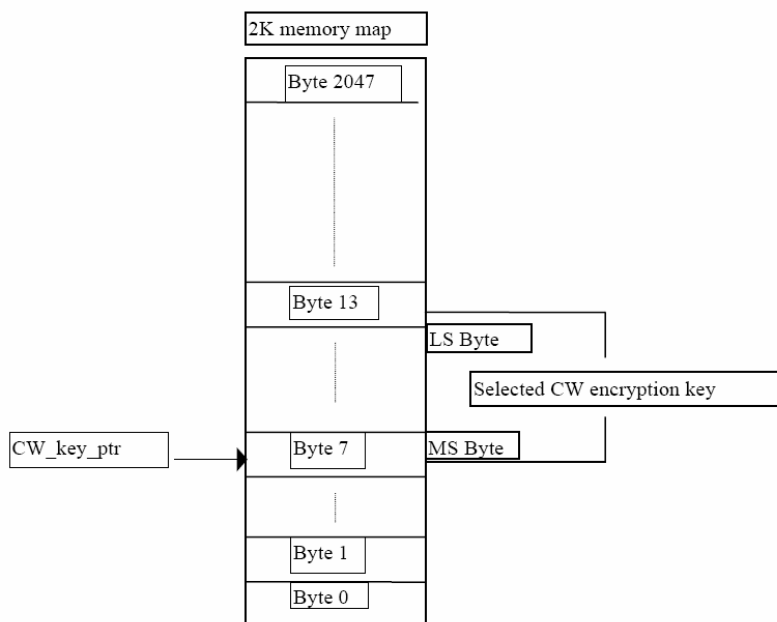


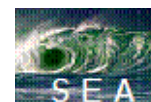
Figure 18. Control Word encryption key selection

Only every seventh address is legal as a key select vector (i.e. 0, 7, 14, 21, etc.) through 2,037. This provides 292 possible completely independent keys.

New keys are selected for each instance of the CW_provision message, and where more than one CW is conveyed in a single CW_provision message, all CWs are encrypted using the same key. One key is required per each CW encryption or decryption operation.

The present document uses a symmetric algorithm, which is the same key is used for both encryption and decryption.

Therefore both the client who requests to view a content and the ECMG shall use the same key as selected by the CW_key_ptr parameter within the CW_encryption parameter. This parameter is generated in the SCS and shall be conveyed in the CW_provision message immediately following the CP_CW_combination parameter.



In Figure 19 is shown the sub-parameters specific to this security method of the CW_encryption parameter of the CW_provision message.

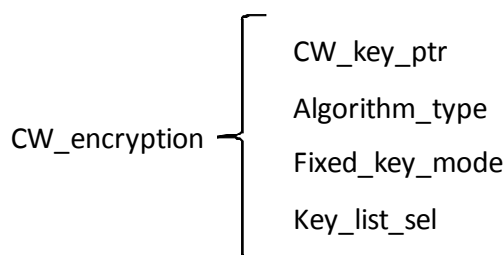


Figure 19. Sub-parameter of the CW_encryption parameter

The meaning of each of the sub-parameters used in the CW_encryption parameter is explained in the following lines. This parameter contains the four sub-parameters listed below that enable encrypting

of control words over the SCS \Leftrightarrow ECMG interface. If the parameter is included in the CW_provision message, control word scrambling is invoked; if omitted, CWs are being issued in the clear.

- **CW_key_ptr:** This 11-bit field contains an index that points to the active CW encryption key contained on a 2,048 byte (or smaller) key list. It is a randomized value (1 of 292) generated within the ECMG which points to the MS byte of a seven-byte key used in block cipher Electronic Code Book mode. Legal values include every 7th address in the 2,048 space (0, 7, 14, 21, etc).
- **Algorithm_type:** This field may be used either to signal the type of encryption algorithm in use for CW encryption or the key length of a given algorithm. It is useful where it may be desirable to change either the fundamental algorithm or its key length, providing both the head-end and all external ECMGs have the appropriate capabilities. In most cases, the Simulcrypting participants will agree on these parameters in advance.
- **Fixed_key_mode:** This bit is used to bypass the key list and use a key contained in Read-Only Memory (ROM) for encryption of the control word. This key will need to be agreed upon by all Simulcrypting participants.
- **Key_list_sel:** In order to facilitate smooth changeover from one key list to another, two independent lists should be maintained on the ECMG. This bit allows selection of one of the two lists as the active list.

3.2.2.3.1.3 Key distribution

The key generated for the CW encryption is distributed by an EMM message. This key can be distributed in clear or encrypted for more security. In the proposed system the key will be distributed in clear in the Data_provision message created by the EMMG.

As mentioned in section 3.2.2.2, the EMM messages are exchanged when a client makes a license request. When that occurs, it is necessary to provide the client with the license and the key for the decryption of the CW. This proceeding is shown in Figure 20.

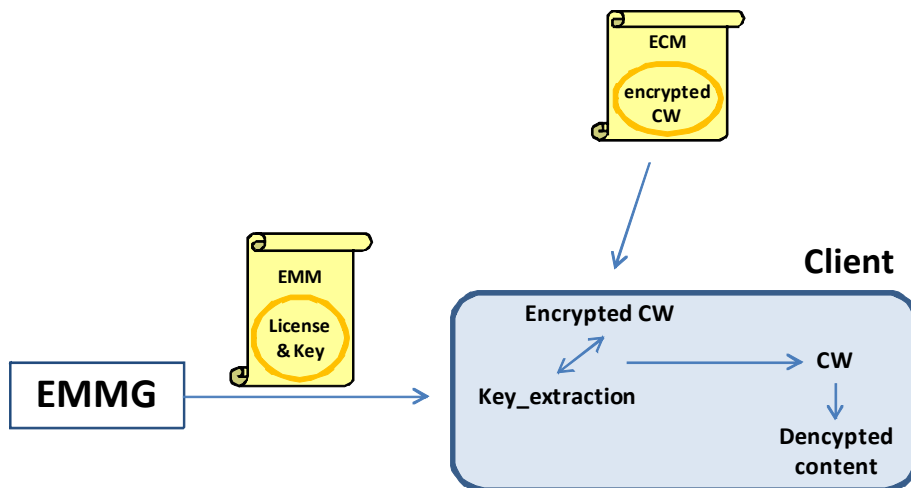


Figure 20. Key distribution and decryption process

3.2.2.3.2 Key management for the license encryption

The encryption of the license is important in order to prevent users to exchange the licenses. To avoid this situation, the SEA project has implemented a key mechanism associated to the *user_id*.

The mechanism consists of the following four steps:

1. The license generated by the license creator (the content owner) is encrypted at the same time that the license is created, transparently to the creator. The key used is a transformation of the *user_id*. This encrypted license is sent to the license server, which is informed of this transformation and, of course, knows the *user_id* of every client because it is in charge of generating them.
2. Once the license reaches the server, it is decrypted by the same mechanism and with the same key that it has been encrypted.
3. Whenever a client asks for a license, the server modifies the license and encrypts it using the *user_id* of the new client. At this point, the license is ready to be sent to the client who requested it.
4. The client receives the license encrypted with the particular information of that particular client, so it cannot be used by any other client. The license is transparently decrypted by the client system, which also knows the mechanism and the key to decrypt it.

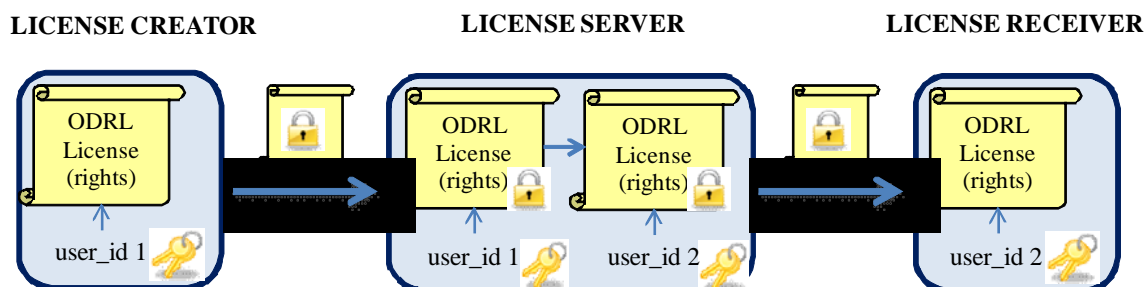
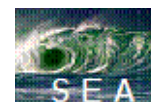


Figure 21. Key management system for license encryption

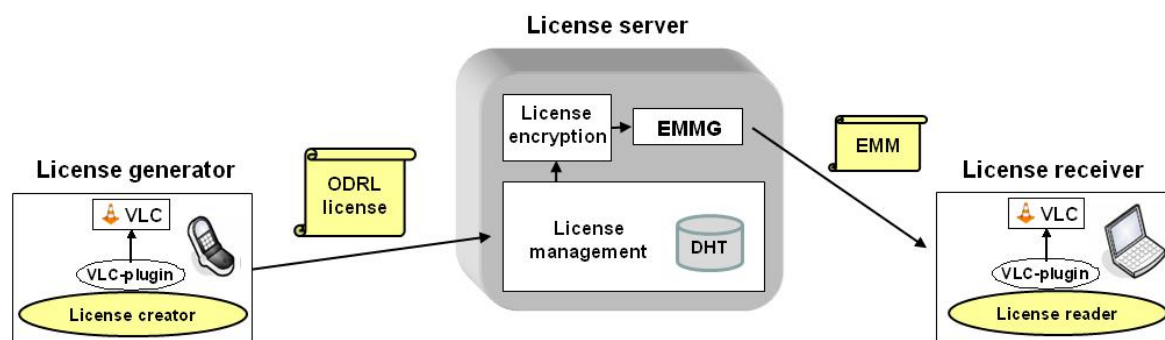


3.2.2.4 License management system

The SEA license system needs to be located at the client side for the generation and performing of the license, and also at the server side for the modification and delivery of the license to the users that request them.

This section explains how the SEA license system is structured, which is reflected in 3.2.2.4. The proceeding is as follows:

1. The license is generated by the license generator peer (the content creator), which creates the license by means of the VLC plug-in. This plug-in provides the user with a tool to select all the rights that he/she wants to give to his/her content. This license is transparently created in ODRL, and sent to the license server. The license details will be extended in section 3.2.2.4.1.
2. The license reaches the license server where it is modified, encrypted and encapsulated in an EMM (Entitlement Management Message). The details of the EMM format and encryption are explained in sections 3.2.2.2 and 3.2.2.4.2, respectively.
3. The EMM is sent to the license receiver which has previously asked for it. Once the license is in the receiver peer, the content is reproduced according to the rights included in the license associated to it. The specific details of the license receiver

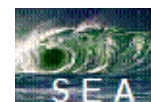


3.2.2.4.1 License generator

The license system has two different functionalities depending if the user acts as a content generator or content consumer. When the user acts as a content generator, it is necessary to encrypt that content and to generate a license to provide the content with rights and constrains.

For the license generation, special requirements and characteristics of the P2P and mobile domain to express consumption rights over DRM content have been taking into account. Some of the specific goals include:

- Light-weight and simple way of expressing rights
- Lowering the entrance barrier for content providers and other players to adopt DRM technologies
- Suitable for specifying rights independently of the content type
- Suitable for specifying rights independently of the transport mechanism
- Enable specification of right to preview DRM content enabling users to experience the content first hand, possibly prior to purchasing it
- Enable specification of constraints to restrict permissions to the number of times content can be accessed, and time limits and intervals during which content can be accessed.



The content generator establishes the rights for the content and reflects them in the license. To do so, the SEA project has implemented and integrated a new functionality for the VLC to generate ODRL licenses [90]. Figure 22 shows the GUI of the license generator.

Figure 22. Principal window for the license creator

In the implemented application of the license creator it can be defined the following characteristics:

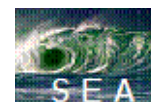
a) Content and author data

- Author name. The name of the content's author.
- Author UID. The *author uid* is the user identifier of the author. Each member of the SEA network has subscribe to the system in order to use its functionalities. When a member subscribes, the system assigned it a user identifier. This user identifier must be unique. In this way for each license it could be known who has been the creator without any doubt.
- Content name. The name of the content. This name will be shown to the P2P network clients together with a brief description of the content.

b) Permissions

- Display. The display element grants the permission to make a transient visible rendering of the content. It contains an optional constraint element. If the constraint element is specified the DRM Agent must grant display rights according to the *constraint* child element. If no *constraint* element is specified, the DRM Agent must grant unlimited display rights.

The *display* element has the semantics of rendering the DRM Content onto a visual device, for example, image/gif or image/jpeg. The DRM Agent must not grant access according to *display* to content that cannot be rendered in this way.



- **Play.** The *play* element grants the permission to create a transient representation of audio or video content. It contains an optional *constraint* element. If the *constraint* element is specified the DRM Agent must grant play rights according to the *constraint* child element. If no *constraint* element is specified, the DRM Agent must grant unlimited play rights.

The *play* element has the semantics of rendering the DRM Content into transient audio/video form, for example, audio/midi, video/quicktime. The DRM Agent must not grant access according to *play* to content that cannot be rendered in this way.

c) Terms of use

- **Number of layers.** The *number of layers* element specifies the number of layers that would be represented when the client play the content. It contains a positive integer value between 1 and 3.

The DRM Agent must not grant the corresponding permission to the DRM Content representing more layers than specified by the contained value. Similarly, the DRM Agent must not grant the corresponding permission to the DRM Content if the contained value is non-positive.

If the contained value is greater than 3 it will be played the maximum layers transmitted of the content. In the present project this maximum is three layers.

- **Number of times.** The *number of times* element specifies the number of times a permission may be granted over an asset. It contains a positive integer value.

The DRM Agent must not grant the corresponding permission to the DRM Content more often than specified by the contained value. Similarly, the DRM Agent must not grant the corresponding permission to the DRM Content if the contained value is non-positive.

When used to constrain the play permission, the count must be decremented immediately upon play.

When used to constrain the display permission, the count must be decremented immediately upon display.

- **Number of views.** The *number of views* element specifies the number of views that would be represented when the client play the content. It contains a positive integer value that could be 1 or 2.

The DRM Agent must not grant the corresponding permission to the DRM Content representing more views than specified by the contained value. Similarly, the DRM Agent must not grant the corresponding permission to the DRM Content if the contained value is non-positive.

If the contained value is greater than 2 it will be played the maximum views transmitted of the content. In the present project this maximum is of two views.

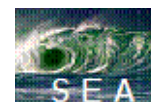
d) Payments requirements

- **Coin.** The kind of currency in which is specified the *amount* field.
- **Amount.** The amount of money the author request to a client for him to see the content with the rights and constrains indicated in the license.
- **Payment modality.** The content author will indicate the payment modalities that he admits. These payments modalities will be shown to the client (buyer) when he proceeds with the payment of the corresponding license.

e) Date and time of the requirements: intervals of time the user can access to the content.

These elements are include to specify the time range for a containing permission. It contains the optional *start date/time* and *end date/time* elements.

If the *start* element is present, its semantics are ‘not before’ the specified time/date.



If the *end* element is present, its semantics are ‘not after’ the specified time/date.

If both are present, the value of the *start* element must be smaller than the value of the *end* element. If the value of the *start* element is greater than the value of the *end* element then the DRM Agent must not grant access to the DRM Content according to the containing permission.

If both are absent, the *start* and *end* element does not have a meaning and must be ignored.

- Start date/time. The *start* element specifies the start date/time. Its semantics are ‘not before’. The general format used for specifying date/time values is defined in [91].

To increase interoperability and facilitate ease of implementation values must conform to a single lexical representation.

The start date is specified indicating the year, the month and the day. The start time is specified indicating the hour, minutes and seconds.

The DRM Agent must not grant the corresponding permission to the DRM Content before the date/time specified by the value of the *start* element.

- End date/time. The *end* element specifies the end date/time. Its semantics are ‘not after’. The general format used for specifying date/time values is defined in [91].

To increase interoperability and facilitate ease of implementation values must conform to a single lexical representation.

The end date is specified indicating the year, the month and the day. The end time is specified indicating the hour, minutes and seconds.

The DRM Agent must not grant the corresponding permission to the DRM Content after the date/time specified by the value of the *end* element.

- Number of days or hours. This element specifies a period of time during which the permissions can be exercised over the DRM Content. This period must begin when the associated permission is first exercised. The permission can then be exercised any number of times within the *interval* period indicated in the license. The DRM Agent must not grant the corresponding permission to the DRM Content after the period specified by the value of the *interval* element has elapsed. Also, the DRM Agent must not allow execution of the permission to continue beyond the specified period.

The specified period should be greater than zero. If the specified period is equal to zero, then the permission must not be granted.

Another functionality added is that the content generator is able to give all the rights (without the need of paying) to several fixed users, just knowing the user name and UID. (See Figure 23)

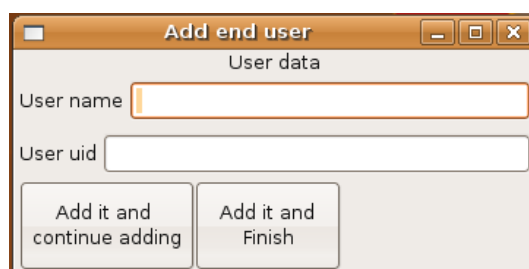


Figure 23. Window to add new free charge users

Once all the gaps are filled in, the license is encrypted with a key that both, user generator and server, know, and sent to the server.

Note that when the license is created, the system will associate the license with a content ID. This information together with the license is sent to the license server. This association must be unique

and it's done in order to provide the correct license when the server receives a client request indicating the content ID.

3.2.2.4.2 License server

As it has been mentioned before, the SEA license server has several functions, as can be seen in Figure 24.

- 1) First of all, the server receives registration requests from the users that have a content to share, and gives the user a *user_id*.
- 2) As it has been explained in section 3.2.2.4.1, the licenses are created at the license creator and the licenser server receives them, with the *content_id*, stores them and keeps an updated list of them in a Data Base. The details of the Data Base are in section 3.2.2.4.2.1.
- 3) Whenever a user asks for a license, the server decrypts the license with the user generator key and encrypts it again with the user consumer key. This measure has been taken in order to avoid the peers from sending the license between each other and use the content freely.
- 4) The server must also include the Service Key (SK) that encrypts the CW that, at the same time, encrypts the stream following the ISMACryp standard, as it has been mentioned before. The service key is provided by the ECMG that is in charge of the encryption of the CW. At this point, the encrypted license and the SK are ready to be included in the EMM message, which is created in the EMMG (see section 3.2.2.2).

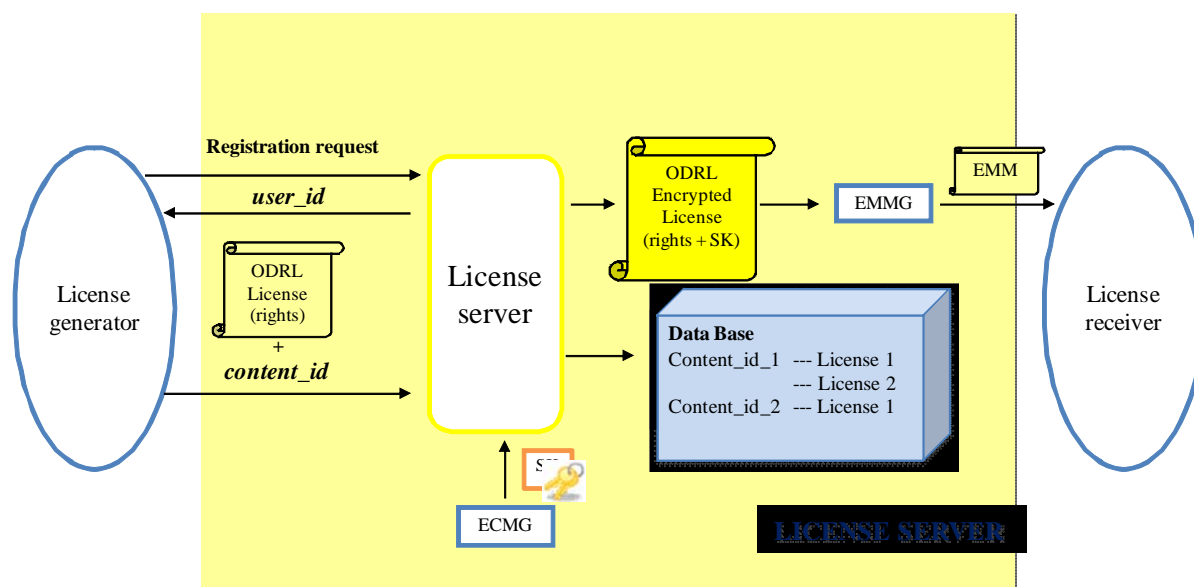


Figure 24. License server

3.2.2.4.2.1 Data base

The function of the data base included in the license server is to keep a list of the contents and the licenses that are linked to each piece of content. This way, there is a univocal association between the content and the licenses associated to that content. This functionality is important to let the users know which contents they have access to and their licenses. This database is updated every time a user generates a license.

3.2.2.4.3 License receiver

At the license consumer side, the SEA project has also implemented a new functionality for the VLC in order to make it able to read ODRL licenses. The user receives the license and decrypts it just introducing its own user id. The license encryption process is transparent to the user (see Figure 25 and Figure 26).

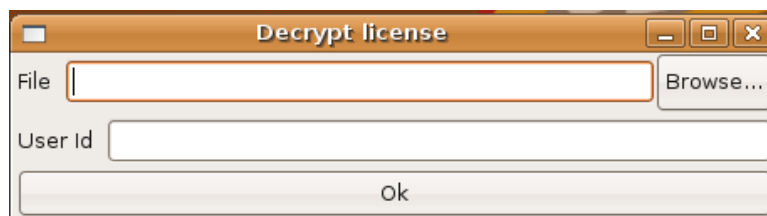


Figure 25. Decrypt license: decrypts and read the license

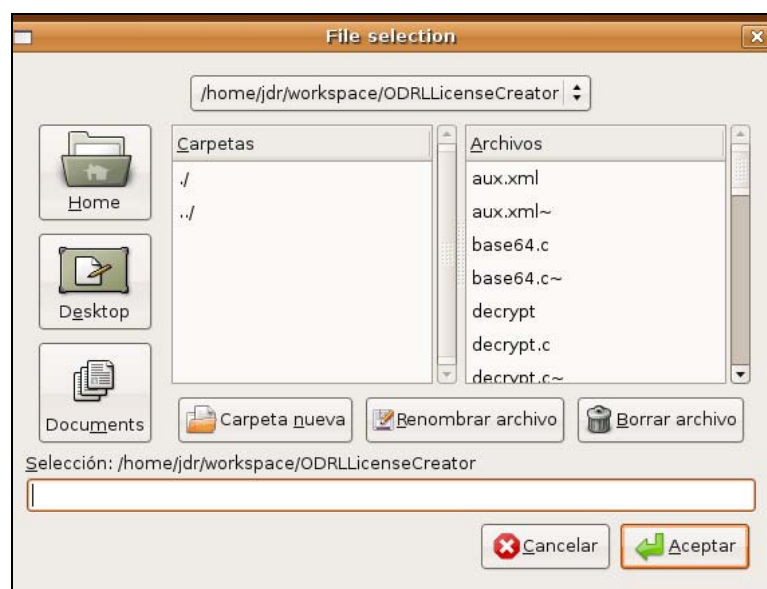


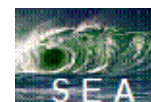
Figure 26. Window to select the license (browser form the previous figure)

The VLC extracts the SK and decrypts the CW. The VLC reads the rights of the license and reproduces de content according to the access rights described within the license.

3.2.2.5 Use cases for P2P content management system

According to the consumption permission given to the media content, we can distinguish several use cases that should be treated in our model. The encryption scheme combined with the proposed network architecture allows implementing a P2P network in which content can be introduced unencrypted if the author wants to distribute it freely, or encrypted if the author wants to take control over de users authorized to view that content. These use cases are described below:

1. **Content that can be consumed by any user.** There are two different situations:
 - 1.1. A piece of content is introduced in the network without any restriction of consumption. Any user can acquire and consume it. There is no key for this kind of content because it is not necessary to decrypt it. The content must carry the corresponding signaling information showing that it is unencrypted.
 - 1.2. Content that can be consumed by any user belonging to the SEA platform, but not by others. Hence, this content should be encrypted. Any user can acquire it, but only



registered users in the SEA platform would be able to view it. All these contents can have the same key, or different ones.

2. Encrypted content restricted to a set of users selected by the author. A user (creator) creates some content and gives permissions to consume it to a reduced group of users, for example, his friends. He encrypts it with a key and he introduces the content in the network. Now we can distinguish two cases:

- 2.1 The creator allows his “friends” to distribute the content to other peers, so he sends them the key (e.g. by email, phone call, etc.) giving them the freedom to consume it whenever they desire.
- 2.2 The creator wants to have control over the content and avoid its consumption by another people. A user creates a piece of content and sets a series of restrictions for its consumption. There are different types of restrictions, as it has been mentioned, such as:
 - Number of times that can be played: It depends on the kind of license a user owns. For example, the more expensive it is, the more times it can be played.
 - Layer to which a user has access to: the SEA content protection mechanism can be applied to H.264 MVC/SVC encoder and decoder. The licenses system implemented provides the consumer with the chance of selecting the number of views and the number of layers the consumer wants to display, depending on the terminal capabilities or the consumer preferences. Section 3.1.2.2 presents many use cases based on the protection of different layers and views. The SEA project provides the option of selecting up to 3 layers and 2 views. The selection of the consumer will be reflected in the license as it has been explained in section 3.2.2.4.1.

The SEA project has implemented only the most critical use case (number 2) in order to provide the system with the highest security. An example could be the following: A user has a content to share and creates two licenses for the rest of the user to consume his/her content. One license gives the consumer rights to see one view of the content and the other license gives the consumer rights to see two views of the same content. The consumer buys the license that contains the rights to see two views and this parameter is included in the license, together with the key to decrypt the content and other rights such as the number of times the consumer can display the content, etc.

3.2.3. Interoperability of the system

In the SEA project it is accorded to follow mainly the recommendations of two standards: MPEG-21 and OMA (Open Mobile Alliance), with the purpose of making the network compatible with other systems, networks or devices.

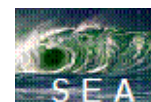
In the proposed architecture, previously described in section 1.1.8, the encrypted content must be a DI (Digital Item), which is the format of a content described in MPEG-21. So, in this way there are no problems of interoperability.

The only module of the system where can be found an incompatibly is in the choice of the format for the license, ODRL.

ODRL has been officially accepted by the OMA as the standard rights expression language for all mobile content. OMA found that ODRL meets its requirements of a light-weight and simple language for expressing rights, easy to implement, optimized expression for delivery over constrained bearers and suitability for specifying rights independently of the content type and transport mechanism. So, ODRL would work well in an OMA network.

However, ODRL it is not compatible with MPEG-21. The Right Expression Language (REL) used in the standard MPEG-21 is MPEG-21 REL, so a conversion between ODRL licenses and MPEG-21 REL licenses must be considered.

ODRL is based on an extensible model for rights expressions, which involves three core entities and their relationships. They are:



- **Party:** includes end users and Rights Holders. Parties can be humans, organizations, and defined roles.
- **Right:** includes permissions, which can then contain constraints, requirements, and conditions. Permissions are the actual usages or activities allowed over the assets (e.g. play, print, etc.) Constraints are limits to these permissions (e.g. print an e-book for a maximum of 3 times) Requirements are the obligations needed to exercise the permission. Conditions specify exceptions that, if they become true, expire the permissions and re-negotiation may be required.
- **Asset:** includes any physical or digital content. They must be uniquely identified and may consist of many subparts and be in many different formats.

The core of MPEG-21 REL is the following four elements: principal, resource, right and condition:

- **Principal:** identifies an entity such as the person, organization, or device to whom rights are granted. Each principal identifies exactly one party. Typically, this information has an associated authentication mechanism by which the principal can prove its identity.
- **Right:** specifies the activity or action that a principal can be granted to exercise against some resource. Example rights include play, print, issue, obtain, etc.
- **Resource:** identifies an object which the principal can be granted a right. It can be a digital work, a service or a piece of information that can be owned by a principal. A Uniform Resource Identifier (URI) can be used to identify a resource.
- **Condition:** specifies one or more conditions that must be met before the right can be exercised. For example, a principal may need to pay a fee to exercise a right, a limit to the number of times, a time interval within which a right can be exercised, etc.

ODRL and MPEG-21 REL have many similarities: syntactically they are both based on XML, while structurally they both conform to the Stefik's axiomatic principles of rights modelling.

A difference between ODRL and MPEG-21 REL is that ODRL seems more adapted to actual transactions in the commerce environment, whereas MPEG-21 REL has designs on broader cross-vertical applicability.

We have seen that they have different entities, but these try to represent the same information. After analyzing both languages, we can conclude that there are four main entities in a license:

- **Subject:** actor who performs some operation. In ODRL, it is the party and in MPEG-21REL it is the principal.
- **Right:** what a subject can do to an object. In ODRL it is the permission (right) and in MPEG-21 REL it is represented by the right.
- **Object:** content acted upon by a subject. In ODRL it is the asset and in MPEG-21 REL it is the resource.
- **Condition:** describes when a right can be performed. In ODRL it is the constraint and is included in the permission (right), and in MPEG-21 REL it is the condition.

Equivalences between ODRL and MPEG-21 REL licenses can be seen in Table 7, while in Table 8 the XML equivalences between both licenses are presented.

Table 7. Equivalences between ODRL and MEPEG-21 REL licenses

ENTITY	ODRL	MPEG-21 REL
Subject	Party	Principal
Object	Asset	Resource
Right (action)	Permission (Right)	Right
Condition (terms)	Constraint (Right)	Condition

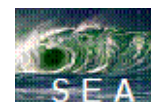


Table 8. XML Equivalences between ODRL and MPEG-21 REL licenses

ENTITY	ODRL	MPEG-21 REL
Subject	o-ex:party	r:keyHolder
Object	o-ex:asset	r:digitalResource
Right (action)	o-ex:permission	r:grant
Condition (terms)	o-ex:constraint	r:allConditions

If we consider the similarities that have been shown, it can be concluded that the interoperability between both languages is possible. To transform an ODRL license into an MPEG-21 REL license, or vice versa, it is equivalent to transform a XML document to another XML document, where the information to represent is the same one, but with a different XML structure.

In order to obtain this transformation, XSL (Extensible Stylesheet Language) can be used. The XSL is one of the most important intricate specifications in the XML family. Using XSLT (XSL Transformation) is not the only way to transform XML documents. A general purpose programming language like C, C++, or Java can also be used. XSLT has the advantage of being more lightweight than those languages and it is oriented to XML interaction. It is adequate for transformation and is well-equipped as a language to perform this main design goal. It allows writing programs that are much smaller than with a general purpose programming language. XSL can be broken in two parts: the said XSLT and XSL-FO (XSL Formatting Objects). XSLT applies transformation rules to the document source and, by changing the tree structure, produces a new document, such as another XML document as shown in Figure 27.

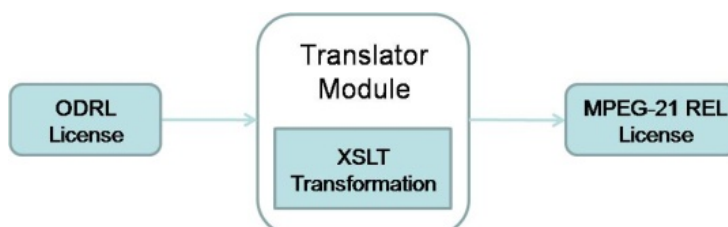


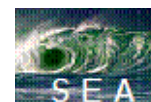
Figure 27. XSTL Transformation

We conclude that is possible to create a MPEG-21 REL license from an ODRL license, which is used in our proposed system. Therefore, we used a license light-weight format that can work correctly in the two main networks considered: OMA and MPEG-21.

3.2.3.1 Compatibility of the system with OMA

OMA DRM is a Digital Rights Management system invented by the Open Mobile Alliance (OMA), whose members represent the entire value chain, including mobile phone manufacturers, mobile system manufacturers, mobile network operators, and IT companies. This scheme is implemented on many recent phones and is intended to be used by mobile content providers to add DRM to their products. To this date two versions of OMA DRM have been released: OMA DRM 1.0 and OMA DRM 2.0.

OMA DRM 1.0 (approved in June 2004) is a basic DRM standard without strong protection. It specifies three methods:



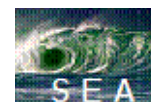
- *Forward Lock*: Being the simplest method, it guarantees that a legitimate user will be the only one to have (unlimited) use of a delivered content; the user is prevented to forward the content to any other device or render it on any but trusted applications. This assumes a model of trusted devices.
- *Combined Delivery*: Usage permissions (e.g. limited number of uses, period of playback, etc.) are defined and enforced by a Rights Object (RO). Rights Object will be delivered to the user accompanying the download of the content.
- *Separate Delivery*: This method separates objects from the rights on using them, therefore it enables re-distribution of content while keeping it protected. It is the most flexible and powerful of the three.

On the other hand, OMA DRM 2.0 (approved in March 2006) was an extension to OMA DRM 1.0 mechanism of separate delivery. We will give an overview of this release, intending to identify its compatibility with the DRM mechanism developed in SEA.

According to the OMA DRM architecture, the following functional (logical) entities are recognised, which embody specific roles and implement different tasks in the DRM system:

- *DRM Agent*: A DRM Agent embodies a trusted entity in a device. This trusted entity is responsible for enforcing permissions and constraints associated with DRM Content, controlling access to DRM Content, etc. It is the equivalent to the VLC plug-in.
- *Content Issuer*: The content issuer is an entity that delivers DRM Content. OMA DRM defines the format of DRM Content delivered to DRM Agents, and the way DRM Content can be transported from a content issuer to a DRM Agent using different transport mechanisms. The content issuer may do the actual packaging of DRM Content itself, or it may receive pre-packaged content from some other source. According to the SEA concept, content can be provided either by a public provider or a private user, both of which we collectively refer to as *peers*.
- *Rights Issuers*: The rights issuer is an entity that assigns permissions and constraints to DRM Content, and generates Rights Objects. A Rights Object (RO) is an XML document expressing permissions and constraints associated with a piece of DRM Content; additionally, it contains the Content Encryption Key by which the content has been encrypted. Rights Objects govern how DRM Content may be used – DRM Content cannot be used without an associated Rights Object, and may only be used as specified by the Rights Object. In SEA architecture the role of the rights issuer is embodied by the license generator and management server, which in combination with the ECM creates and publishes Rights Objects (EMMs) to enforce the proper consumption of the digital item.
- *User*: A user is the human user of DRM Content, referring to the *subject* entity presented earlier. Users can only access DRM Content through a DRM Agent.
- *Off-device Storage*: DRM Content can inherently be stored in off-device storage media (e.g. network storage devices, PC removable hard disks, etc.), for backing up purposes, in order to free up memory in devices, and so on. The use of such media is optional, and in our case corresponds to the Content Storage Module (CSM). Similarly to the content items, ROs that only contain stateless permissions may also be stored off-device.

The basic procedure for distributing a digital item, according to OMA DRM 2.0, follows the steps below:



1. Content packaging: Content is packaged in a secure content container (DRM Content Format, DCF). DRM Content is encrypted with a symmetric content encryption key (CEK). Content can be pre-packaged, i.e. content packaging does not have to happen on the fly.

Although not required by the OMA DRM specifications or the OMA DRM architecture, it is recommended that the same CEK is not used for all instances of a piece of content. Using the same CEK for all content instances would pose a greater risk if a single device was to be hacked and a CEK stored on that device exposed. Using a different CEK for different deliveries or different devices will limit this risk.

2. DRM Agent authentication: All DRM Agents have a unique private/public key pair and a certificate. The certificate includes additional information, such as maker, device type, software version, serial numbers, etc. This allows the content and rights issuers to securely authenticate a DRM Agent.
3. Rights Object generation: A Rights Object is an XML document, expressing the permissions and constraints associated with the content. The Rights Object also contains the CEK – this ensures that DRM Content cannot be used without an associated Rights Object.
4. Rights Object protection: Before delivering the Rights Object, sensitive parts are encrypted (e.g. the CEK), and the Rights Object is then cryptographically bound to the target DRM Agent. This ensures that only the target DRM Agent can access the Rights Object and thus the DRM Content. In addition, the Rights Issuer digitally signs the RO.
5. Delivery: The RO and DCF can now be delivered to the target DRM Agent. Since both are inherently secure, they can be delivered using any transport mechanism (e.g. HTTP/WSP, WAP Push, MMS). They can be delivered together, e.g. in a MIME multipart response, or they can be delivered separately. In the latter case the DCF header contains the Rights Issuer URL where the target DRM Agent can connect to and acquire the needed RO.

Similarly to the release 1, the second release of OMA DRM is based on a trust model where the burden of rights management is lifted from the content or rights publisher and moved to the (mobile) device space. DRM Agent has to be trusted by the rights issuer, both in terms of correct behaviour and in terms of secure implementation. In OMA DRM, each DRM Agent is provisioned with a unique key pair, and an associated certificate, identifying the DRM Agent and certifying the binding between the agent and this key pair. This allows rights issuers to securely authenticate the DRM Agent using standard PKI procedures.

The DRM mechanism implemented in SEA follows the general concept scheme of the OMA DRM architecture. However, due to the nature of distributed content delivery amongst many peers, and also due to the fact that even a private user should be able to publish his or her original material under a protection shell for controlling its consumption, a slight modification has been chosen. An ECM, generated by the server, is distributed alongside the encrypted content. When a user decides to obtain usage licenses for that received content, the ECM pushes the generation of an EMM at the server, which is bound to that particular user. This way the protected content can be decrypted and viewed only but that user.

However, the license negotiation procedure and the decryption key acquisition, in between the requesting user and the server, are generally in accordance with the OMA DRM specifications. Hence, the chosen mechanism can allowed for use cases as specified also by OMA DRM. In particular, these can be:

- Basic downloading/streaming
- Superdistribution, i.e. the redistribution of a downloaded content to other devices, using various networked links or removable storage media, for which the proper licenses have been paid for and acquired



- Domains distribution, i.e. the distribution of content to a group of users that are enrolled in a domain, which is created, managed and administrated by the Rights Issuer



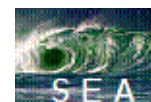
4. Conclusions

This document describes a lightweight solution for content protection and management in P2P environments using personalized and adaptable video environments (SVC/MVC).

The solutions proposed, including content protection and authentication using ISMACryp and interoperable content management with DRM/CA solutions, are tailored for achieving the expected lightweight system load and correct content management and protection performance in the identified network environment, advancing over the state-of-the-art of the content protection mechanisms adapted to the SVC/MVC delivery networks.

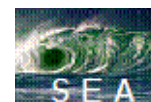
A general architecture has been described, including the licenses managers, encryption mechanisms, key management elements, mediators, etc., which is fully compatible and adapted to a SVC/MVC content delivery network. In addition, the content management system has been defined to be interoperable with as much standards as possible, ensuring a real seamless content delivery across heterogeneous networks and terminals.

In the next steps of the project, this solution will be integrated with the rest of the SEA system components and its performance and abilities will be demonstrated.

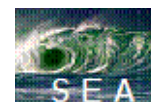


5. References

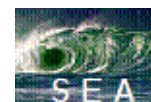
- [1] Sam Michiels, Wouter Joosen, Eddy Truyen, Kristof Verslype, "Digital Rights Management - A Survey of Existing Technologies", Katholieke Universiteit Leuven, Report CW 428, November 2005
- [2] DRM Architecture, Open Mobile Alliance, Draft Version 2.0, March 2004
- [3] Bart J. Van Rijnsoever, Peter Lenoir, Jean-Paul M.G. Linnartz, "Interoperable Protection for Digital Multimedia Content", Journal of VLSI Signal Processing Vol. 34, pp. 167–179, 2003
- [4] Bohyun Wang, Zonghua Liu, Byungwook Lee, "A Study of Superdistribution Model for Family Domain in DRM System", in proc. of Fifth International Conference on Computational Science and Applications, ICCSA 2007, Aug. 2007
- [5] Pramod A. Jamkhedkar, Gregory L. Heileman, "DRM as a Layered System", in Proc. of the 4th ACM workshop on Digital rights management, pp. 11-21, 2004
- [6] Yang Yu Tzi-cker Chiueh, "Enterprise Digital Rights Management: Solutions against Information Theft by Insiders"
- [7] XrML 2.0 Technical Overview Version 1.0 March 8, 2002.
- [8] <http://www.parc.com>.
- [9] The Digital Property Rights Language Manual and Tutorial - XML Edition Version 2.00 — November 13, 1998 (<http://xml.coverpages.org/DPRLmanual-XML2.html>).
- [10] Open Digital Rights Language (ODRL) Version: 1.1 Date: 2002-08-08 Available at: <http://odrl.net/1.1/ODRL-11.pdf>.
- [11] Coding of Moving Pictures and Audio ISO/IEC JTC1/SC29/WG11/N5231 Shanghai, October 2002.
- [12] Stephanos Adroutsellis-Theotokis, Diomidis Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies", ACM Computing Surveys, Vol. 36, No. 4, pp. 335–371, December 2004
- [13] W. Sullivan, D. Werthimer, S. Bowyer, J. Cobb, D. Gedye, D. Anderson, "A new major seti project based on project serendip data and 100,000 personal computers" in procof the 5th International Conference on Bioastronomy, 1997
- [14] P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, I. Zaihrayeu, "Data management for peer-to-peer computing: A vision", in Proc. of the Workshop on the Web and Databases (WebDB'02), 2002
- [15] A. Halevy, Z. Ives, P. Mork, I. Tatarinov, "Piazza: Data management infrastructure for semantic web applications", in Proc. of the 12th International Conference on World Wide Web. Budapest, Hungary, pp. 556–567, 2003
- [16] www.wikipedia.org
- [17] M. Bawa, H. Deshpande, H. Garcia-Molina, "Transience of peers & streaming media" ACM SIGCOMM Computer Communication Review, Vol. 33, No 1, pp. 107–112, 2003
- [18] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast. In Measurement and Modeling of Computer Systems", in proc. of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, pp 1–12, 2000
- [19] Duc A. Tran, Kien A. Hua, Tai Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming", in proc. of The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), vol.2, pp. 1283-1292, 2003



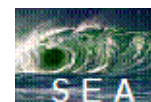
- [20] Xinyan Zhang, Jiangchuan Liuy, Bo Liz, Tak-Shing Peter Yum, "CoolStreaming/DONet: A Data-Driven Overlay Network for Efficient Live Media Streaming", in proc. of The 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005), vol. 3, pp. 2102 - 2111, 2005
- [21] J. Wong, "Enhancing Collaborative Content Distribution with Helpers" Master's thesis, The University of British Columbia, 2004
- [22] B. Cohen, "Incentives Build Robustness in BitTorrent", in proc. of the 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley, 2003
- [23] J. Pouwelse, P. Garbacki, J. Wang, "Tribler: A Social-based Peer-to-Peer System", in proc. of the 5th Int'l Workshop on Peer-to-peer Systems, pp. 391–394, 2006
- [24] J.R. Douceur, "The Sybil Attack," 1st International Workshop on Peer-to-Peer Systems (IPTPS '02). Cambridge, MA., 2002
- [25] D.S. Wallach, "A Survey of Peer-to-Peer Security Issues". International Symposium on Software Security Tokyo, Japan, 2007.
- [26] T. Ngan, D.S. Wallach, P. Druschel "Enforcing Fair Sharing of Peer-to-Peer Resources" 6th International Workshop on Peer-to-Peer Systems (IPTPS '06), 2006
- [27] M.J. Freedman, R. Morris, "Tarzan: A Peer-to-Peer Anonymizing Network Layer," ACM Conference on Computer and Communications Security, Washington, D.C., 2004
- [28] A. Rowstron et. al. "Pastry: Scalable, distributed object location and routing for large-scale P2P systems," IFIP/ACM Conference on Distributed Systems Platforms, Heidelberg, Germany, 2003
- [29] I. Clarke et. al. "Freenet: A Distributed Anonymous Information Storage and Retrieval System" Workshop on Design Issues in Anonymity and Unobservability, Berkely, CA, 2004
- [30] R. Dingledine, et. al. "The Free Haven Project: Distributed Anonymous Storage Service." Workshop on Design Issues in Anonymity and Unobservability, Berkely, CA., 2004
- [31] Groove Networks, "Groove Security Architecture", <http://www.groove.net/pdf/security.pdf>, 2005
- [32] W. Wang, L. Zhao, R. Yuan, "Improving Cooperation in Peer-to-Peer Systems Using Social Networks", in proc. of 3rd Workshop on Hot Topic in Peer-to-Peer Systems, Greece, pp. 50–57, 2006
- [33] Xiaofei Liao, Hai Jin, Yunhao Liu, Lionel M. Ni, Dafu Deng, "AnySee: Peer-to-Peer Live Streaming", in proc. of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006), pp. 23–29, April 2006
- [34] PPLive, www.pplive.com
- [35] Stephanos Androutsellis-Theotokis and Diomidis Spinellis, "A survey of peer-to-peer content distribution technologies", ACM Computing Surveys, 36(4):335-371, December 2004
- [36] Chu C.C., Su X., Prabhu B.S., Gadh R., Kurup S., Sridhar G. Sridhar V., "Mobile DRM for multimedia content commerce in P2P networks", Consumer Communications and Networking Conference, 2006 3rd EEE Volume 2, 8-10 Jan. 2006 Page(s):1119- 1123.
- [37] Xinwen Zhang, Songqing Chen, Ravi Sandhu, "Enhancing Data Authenticity and Integrity in P2P Systems," EEE Internet Computing, vol.09, no.6, pp. 42-49, Nov/Dec, 2005.
- [38] Shane Balfe, Amit D. Lakhani, Kenneth G. Paterson, "Trusted Computing: Providing Security for Peer-to-Peer Networks," p2p, pp. 117-124, Fifth EEE International Conference on Peer-to-Peer Computing (P2P'05), 2005)



- [39] Iwata T., Abe T., Ueda K., et al. "A DRM System Suitable for P2P Content Delivery and the Study on Its Implementation", The 9th Asia-Pacific Conference on Communications, 2003, pp. 806-811.
- [40] Jae -Youn, Sung Jeong -Yeon Jeong, Ki -Song Yoon "DRM Enabled P2P Architecture"
- [41] Information technology — Coding of audio-visual objects — Part 12: ISO base media file format, [ISO/IEC 14496-12:2005\(E\)](#)
- [42] Information technology — Coding of audio-visual objects — Part 15: Advanced Video Coding (AVC) file format, [ISO/IEC 14496-15:2004](#)
- [43] Information technology — Coding of audio-visual objects — Part 15: file format, AMENDMENT 2: File format support for Scalable Video Coding, [ISO/IEC 14496-15:2004/FDAM 2:2008\(E\)](#)
- [44] Information technology — Coding of audio-visual objects — Part 1: Systems, [ISO/IEC 14496-1](#)
- [45] Information technology — Coding of audio-visual objects — Part 2: Video, [ISO/IEC 14496-2](#)
- [46] Information technology — Coding of audio-visual objects — Part 3: Audio, [ISO/IEC 14496-3](#)
- [47] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, IETF STD 0064, RFC 3550, July 2003, <http://tools.ietf.org/html/rfc3550>
- [48] S. Wenger, M.M. Hanuksela, T. Stockhammer, M. Westerlund, D. Singer, RTP Payload Format for H.264 Video, February 2005, <http://tools.ietf.org/html/rfc3984>
- [49] Y.-K. Wang, S. Wenger, M.M. Hanuksela, T. Stockhammer, M. Westerlund, D. Singer, RTP Payload Format for H.264 Video, [draft-wang-avt-rfc3984bis-01.txt](#), July 14, 2008
- [50] S. Wenger, Y.-K. Wang, T. Schierl, A. Eleftheriadi, RTP Payload Format for SVC Video, September 26, 2008, [draft-ietf-avt-rtp-svc-14.txt](#)
- [51] Y.K. Wang, Th. Schierl, RTP Payload Format for MVC Video, August 21, 2008, [draft-wang-avt-rtp-mvc-02.txt](#)
- [52] H. Schulzrinne, A. Rao, R. Lanphier: "Real Time Streaming Protocol (RTSP)", IETF RFC 2326, April 1998, <http://tools.ietf.org/html/rfc2326>
- [53] M. Handly, V. Jacobson and C.Perkins, "SDP: Session Description Protocol", IETF RFC 4566, July 2006, <http://tools.ietf.org/html/rfc4566>
- [54] T. Schierl, S. Wenger, Signaling media decoding dependency in Session Description Protocol, September 25, 2008, [draft-ietf-mmusic-decoding-dependency-03.txt](#)
- [55] M. Handley, C. Perkins, E. Whelan: Session Announcement Protocol", IETF RFC 2974, October 2000, <http://tools.ietf.org/html/rfc2974>
- [56] ISMA Implementation Specification TD00094, ISMA Encryption and Authentication, Version 1.1, September 15, 2006
- [57] J. van der Meer, D. Mackie, V. Swaminathan, D. Singer, P. Gentric, RTP Payload Format for Transport of MPEG-4 Elementary Streams, November 2003, IETF RFC 3640, <http://tools.ietf.org/html/rfc3640>
- [58] Digital Video Broadcasting (DVB); Head-end implementation of DVB SimulCrypt, ETSI TS 103 197 V1.4.1, September 2004
- [59] Y. Mao and M. Wu, "A joint signal processing and cryptographic approach to multimedia encryption," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 2061-2075, July 2006.



- [60] H. Cheng and X. Li, "Partial encryption of compressed images and videos," *IEEE Transactions on Image Processing*, vol. 48, no. 8, pp. 2439-2451, Aug. 2000.
- [61] R. Grosbois, P. Gerbelot, and T. Ebrahimi, "Authentication and access control in the JPEG 2000 compressed domain," in *Proceedings of the SPIE 46th Annual Meeting*, 2001.
- [62] M.V. Droogenbroeck and R. Benedett, "Techniques for a selective encryption of uncompressed and compressed images," *Proceedings of ACIVS*, 2002.
- [63] M. Pazarci and V. Dipcin, "A MPEG2-transparent scrambling algorithm," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 2, pp. 345-355, Feb. 2002.
- [64] S. Lian, J. Sun, and Z. Wang, "Perceptual cryptography on SPIHT compressed images or videos," *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, 2004.
- [65] S. Lian, J. Sun, and Z. Wang, "Perceptual cryptography on MPEG compressed videos," *Proceedings of IEEE International Conference on Signal Processing (ICSP)*, 2004.
- [66] S. Lian, Z. Liu, Z. Ren, and H. Wang, "Secure advanced video coding based on selective encryption algorithms," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 2, pp. 621-629, May 2006.
- [67] B.B. Zhu, C. Yuan, Y. Wang, and S. Li, "Scalable protection for MPEG-4 fine granularity scalability," *IEEE Transactions on Multimedia*, vol. 7, no. 2, pp. 222-233, Apr. 2005.
- [68] M.S. Kankanhalli and T.T. Guan, "Compressed domain scrambler/descrambler for digital video," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 2, pp. 356-365, May 2002.
- [69] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 118-129, Mar. 2003.
- [70] C. Wang, H.-B. Yu, and M. Zheng, "A DCT-based MPEG-2 transparent scrambling algorithm," *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 1208-1213, Apr. 2003.
- [71] S. Lian, J. Sun, and Z. Wang, "Perceptual cryptography on JPEG2000 compressed images or videos," *Proceedings of the Fourth International Conference on Computer and Information Technology*, 2004.
- [72] Y. Bodo, N. Laurent, and J.-L. Dugelay, "A scrambling method based on disturbance of motion vectors," *Proceedings of the 10th ACM International Conference on Multimedia*, 2002.
- [73] D. Engel and A. Uhl, "Secret wavelet packet decompositions for JPEG 2000 lightweight encryption," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2006.
- [74] D. Engel and A. Uhl, "Security enhancement for lightweight JPEG 2000 transparent encryption," *Proceedings of International Conference on Information, Communication, and Signal Processing*, 2005.
- [75] J. Wen, M. Severa, W. Zeng, M.H. Luttrell, and W. Jin, "A format-compliant configurable encryption framework for access control of video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 6, pp. 545-557, June 2002.
- [76] S. Li, G. Chen, A. Cheung, B. Bhargava, and K.-T. Lo, "On the design of perceptual MPEG-video encryption algorithms," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 2, pp. 214-223, Feb. 2007.
- [77] C. Bergeron and C. Lamy-Bergot, "Compliant selective encryption for H.264/AVC video streams," *Proc. of IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2005.



- [78] A. Torrubia and F. Mora, "Perceptual cryptography of JPEG compressed images on the JFIF bitstream domain," *Proceedings of Dig. ICCE*, 2003.
- [79] X. Ruan and R.S. Katti, "A new source coding scheme with small expected length and its application to simple data encryption," *IEEE Transactions on Computers*, vol. 55, no. 10, pp. 1300-1305, Oct. 2006.
- [80] C.-P. Wu and C.-C. Jay Kuo, "Design of integrated multimedia compression and encryption systems," *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 828--839, Oct. 2005.
- [81] M. Grangetto, E. Magli, and G. Olmo, "Multimedia selective encryption by means of randomized arithmetic coding," *IEEE Transactions on Multimedia*, vol. 9, no. 5, pp. 905-917, Oct. 2006.
- [82] M. Grangetto, A. Grosso, and E. Magli, "Selective encryption of JPEG 2000 images by means of randomized arithmetic coding," *Proceedings of IEEE International Workshop on Multimedia Signal Processing*, 2004.
- [83] H. Kim, J. Wen, and J.D. Villasenor, "Secure arithmetic coding," *IEEE Transactions on Signal Processing*, v. 55, n. 5, pp. 2263-2272, May 2007.
- [84] R. Bose and S. Pathak, "A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system," *IEEE Transactions on Circuits and Systems I*, vol. 53, no. 4, pp. 848-857, Apr. 2006.
- [85] B.B. Zhu, Y. Yang, and S. Li, "JPEG 2000 syntax-compliant encryption preserving full scalability," *Proceedings of IEEE International Conference on Image Processing*, 2005.
- [86] Y. Wu and R.H. Deng, "Compliant encryption of JPEG2000 codestreams," *Proceedings of IEEE International Conference on Image Processing*, 2004.
- [87] S. Wee and J. Apostolopoulos, "Secure scalable streaming and secure transcoding with JPEG-2000," *Proceedings of IEEE International Conference on Image Processing*, 2003.
- [88] E. Magli, M. Grangetto, G. Olmo, "Conditional access to H.264/AVC video with drift control," *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, 2006.
- [89] Data Encryption Standard (DES). U.S. Department of Commerce/national Institute of Standards and Technology. Federal information processing standards publication (FIPS PUB 46-3.). Reaffirmed 1999, October 25.
- [90] Open Digital Rights Language (ODRL) Version: 0.9 Date: 2001-06-29 URI: <http://odrl.net/0.9/ODRL-09.pdf>
- [91] Data elements and interchange formats — Information interchange — Representation of dates and times. ISO 8601.
- [92] Open Mobile Alliance, DRM Architecture, Approved Ver. 2.0 – 20 Dec. 2006 (OMA-AD-DRM-V2_0_1-20061220-A)