Advances in Electrical and Computer Engineering **_Volume 11, Number 1, 2011_**

A Generalized DRM Architectural Framework

Victor-Valeriu PATRICIU, Ion BICA, Mihai TOGAN, Stefan-Vladimir GHITA
Military Technical Academy, 050141, Romania
vip@mta.ro

***Abstract*—Online digital goods distribution environment lead to the need for a system to protect digital intellectual property. Digital Rights Management (DRM) is the system born to protect and control distribution and use of those digital assets. The present paper is a review of the current state of DRM, focusing on architectural design, security technologies, and important DRM deployments. The paper primarily synthesizes DRM architectures within a general framework. We also present DRM ecosystem as providing a better understanding of what is currently happening to content rights management from a technological point of view. This paper includes conclusions of several DRM initiative studies, related to rights management systems with the purpose of identifying and describing the most significant DRM architectural models. The basic functions and processes of the DRM solutions are identified.**

Index Terms—**Digital Rights Management, DRM architectural framework, DRM functional description, DRM key concepts.**

I. INTRODUCTION

With widespread use of the Internet and recent improvements in multimedia distribution technology, digital music, images, video, books, games, etc can be distributed almost instantaneously to end-users.

Many digital service providers already assume this new way of selling their digital content over computer networks. However, without serious protection and management of digital rights, digital content can be easily illegally copied, altered, and distributed to a large number of Internet users.

In response to these threats, to protect digital content intellectual property, rights management systems are needed to prevent unauthorized access to digital content and manage content usage rights.

According to numerous Digital Rights Management (DRM) specifications [1][2][3], a DRM system must offer a persistent content protection against unauthorized access to content, granting access only to proper authorized principals. Another important feature of DRM is format flexibility [4]: it should be able to manage usage rights for different kinds of digital content (multimedia files/streams, digital books, images, etc) and for different platforms (PCs, laptops, PDAs, mobile phones, etc).

Existing DRM industrial solution have different implementation with different names to DRM components, workflows, usage rules, etc. But, with all these differences most of them fall into the very basic DRM components and processes. However, DRM interoperability is one of the hottest topics in both industrial and academic research activities, and although requirements are well understood [5], security models of different solutions are difficult to analyze together.

The paper overviews the current state in DRM systems and represents an effort to synthesize a generalized DRM architectural framework. It contains 6 sections starting with the introduction (section 1) and reaching conclusions in section 6. Section 2 describes DRM key concepts in general and explains the typical DRM ecosystem. Section 3 presents a conceptual hierarchical model of DRM systems. Section 4 examines some existing open standard DRM systems. Section 5 presents the proposed framework.

II. DRM KEY CONCEPTS

Based on definition: "DRM covers the description, identification, trading, protection, monitoring and tracking of all forms of rights usages over both tangible and intangible assets including management of rights holders relationships" [6], DRM systems must manage two elementary content constraints:

- **Rights Management** - legal rights holders need to identify their content, collect the metadata, assert what are the rights on the content and provide business models for distribution.
- **Rights Enforcement** - rights holders also need to enforce their rights and rules for their content usage. Ensuring this feature for DRM solutions is one the most challenging task being also primordial to the DRM goal.

With minor differences, digital media distribution systems (multimedia, documents, music, etc) involve four major actors [7]:

- **Creator** - creator and legal owner of the content (music, video, documents, etc);
- **Producer** - makes the digital content product, wraps and protects this product;
- **Distributor** - promotes and sells the digital content product to customers;
- **Consumer** - the client for digital content that pays user fees and consumes the product.

The first two entities in some implementations embody one single entity often referred as **Content Provider**.

Because of its nature, digital goods delivery introduces a new party: the licensing subsystem. This subsystem is responsible with the goods rights management, in the terms of intellectual property.

The licensing subsystem, as a trusted party of the entire goods delivery system, has two elementary components: the Licensing Service (LS), which deals with client rights management (according to the goods use rights expressed by their owner) and client components which have the role to enforce those use rights.

During the electronic flow of DRM enabled content distribution systems, depicted in Fig. 1, four major phases can be identified to invoke security measures against content piracy [8]: *content creation*, *content distribution*, *rights*

Digital Object Identifier 10.4316/AECE.2011.01007

distribution and *content consumption*. Each of these stages needs to be compliant to specific DRM operations as described in the following sections.

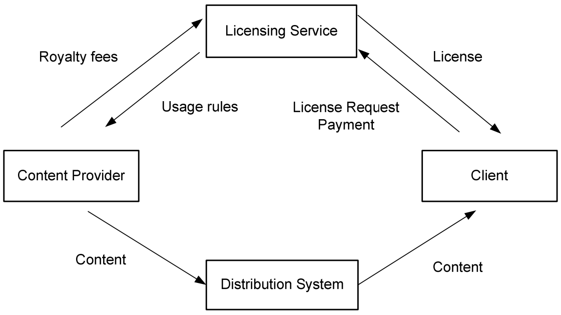

Figure 1. Data flow in a typical DRM system.

In the DRM world, rights management is accomplished both though encryption and also by controlling both service and client side behavior [9]. The licensing service is responsible for delivering user rights and constraints to client, encapsulated with other system specific information (encryption keys, metadata, etc) in the so called license. Concerning client, the DRM client component (also referred as DRM agent) must ensure client obedience to the rules expressed in the license as well as rights.

In a DRM environment, client rights are restricted via a rights expression language [10][11] and encapsulated with other system specific information (encryption keys, metadata, etc) in the license object (information) interpreted by the DRM client (also referred as DRM agent).

DRM Functional Description

From a functional perspective, the essential DRM architectural functions can resume to: Content Creation and Capture, Content Management and Content Usage [1][6][12] (as depicted in Fig. 2).

Content creation management has the role to facilitate trading, including rights assertion when content is first created (or used/modified and extended with appropriate rights). At this phase three measures must be taken: rights validation (which ensure that content being created includes the consistent rights to do so), rights creation (rights assignment for new content) and rights workflow (process content for review and/or approval of rights).

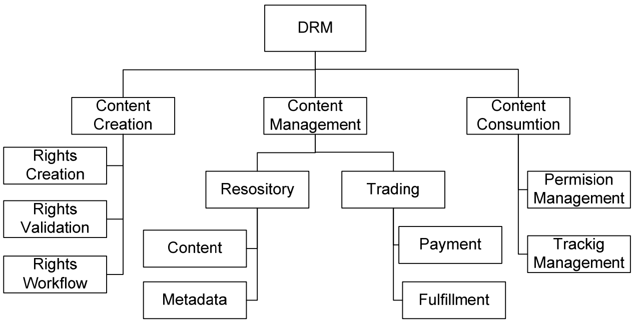

Figure 2. DRM Functions (summarized [1]).

Content management implies content trading and content asset management i.e. repository functions (access content, content metadata and the rights specifications) and trading functions (license assignment to parties who have rights over content, including, for example, royalty payments). According to [4] and [6], content use management, once content has been traded, must support: permissions management - to enforce the rights associated with the content, and tracking management - to monitor the use of content where such tracking is a required.

III. DRM LAYERED VIEW

In paper [13], the authors summarize DRM architectures to a four-layer generic model, describing the functionality and components interactions from a security oriented perspective. Their contribution is abstracting DRM functionality in an OSI like traditional manner (Fig. 3) regardless to content type or business logic.

Even if the model proposed by [13] does not focus on key characteristic of DRM systems, it succeeds in creating an abstract DRM system model, based on key participants and operations. In our opinion, this structured manner of understanding DRM not only offers a clear system view but can contribute greatly to DRM system interoperability definition and standardization.

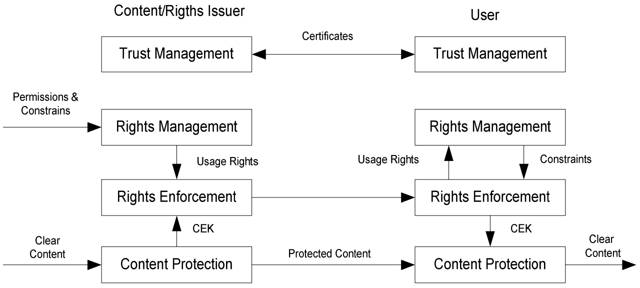

Figure 3. DRM layered view.

The first layer on this hierarchical view is trust management. This layer ensures that only trustful parities interact. The system should deny or provide limited service interaction to any non-trusted participant.

In actual DRM industry leading open source implementations [1][14][15], the typical manner to provide the "trust fabric" is implemented through authentication and digital certificates. A Certification Authority (CA) issues signed certificates for all compliant DRM system components. In addition, the certificate may also contain declaration of capabilities, or requirements of its owner.

In other words, via digital certificates, the issuing CA acknowledges that certified owner is authorized to perform some set of features, and is authenticated for some given set of capabilities.

In this layered DRM model view, as depicted in Fig. 3, the trust management is responsible for checking certificates validity and ensures that only authenticated and compliant issuers are able to create licenses for trusted clients and only authenticated and compliant clients are able to open licenses from trusted services.

In the Rights management layer, content providers or content distributors define commercial rights transposed into user rights and constraints. On the server side, the rights management layer transcribes commercial rights allocated for content into usage rights using a standard expression syntax, like [10][11], and forwards this information to the rights enforcement layer.

Rights Enforcement layer's role is to ensure that content will only be used under specific conditions defined by the usage rights. According to [5], this layer has two main roles: to protect the usage rights associated with content (a possible attacker should not be able to modify the usage rights), and to guarantee that usage rights are obeyed and not bypassed (content will only be used according to usage rights).

Content protection is an essential feature of any DRM model. Content should be securely sealed so that access should not be possible without having associated rights.

From the client's perspective, the content protection layer can only access protected content if rights enforcement layer forwards the right Content Encryption Key (CEK).

Once we introduced the abstract context and the formal functionality of DRM systems, we focus attention to the most significant existing DRM implementations.

IV. CURRENT OPEN STANDARD DRM SYSTEMS

In this section we shortly present four open source DRM systems, specially chosen because of their important contribution to the presented topic. Also, in our opinion, these solutions have a major contribution to the industry, driving most of commercial products implementation in this area.

OMA DRM

In OMA DRM 2.0 system [1] content issuer packages and protects media objects, scrambling them with a 128-bit AES symmetric Content Encryption Key (CEK).

The rights issuer describes associated content usage rights via Open Digital Rights Language (ODRL) [10] in chime with content owner rights definition. These usage rights are then packaged together with CEK into a rights object. The rights object is cryptographically bound with the content it specifies and is associated to one DRM agent it's addressing.

The DRM agent is a trusted entity that is executed on the mobile appliance receiving DRM content and rights object (license). Every OMA DRM agent has a public/private key pair with a certificate delivered by a CA. The certificate, in addition to typical PKI (Public-Key Infrastructure), carries also information on the characteristics of the DRM agent. Based on this, the rights issuer may decide if to accept rights object issuing and delivery to a given DRM agent.

Furthermore, rights issuer encrypts the rights object only for the expected DRM agent, signing every rights object it issues.

When trying to access the protected content, the DRM agent opens the associated rights object (only possible if the rights object was generated for it).

The rights management layer parses the ODRL expression inside rights object extracting permissions and constraints. At this step, the right enforcement layer ensures obedience to these constraints and passes CEK to the content protection layer that descrambles the content.

OpenIPMP

OpenIPMP is an open source project, developed by Objectlab [14], based upon MPEG standards family, PKIs (Public-Key Infrastructures), Digital Object Identifier (DOI) for content identification scheme, and the Open Digital Rights Language (ODRL) as the rights expression language. OpenIPMP implements a fully functional DRM solution supporting content encryption/decryption, license management, content identification and user identification.

As defined by Objectlab, "OpenIPMP is a collection of tools/services capable of delivering a robust, scalable, and adaptive infrastructure to support management and secure delivery of media assets through each step in the asset life cycle" [14].

The OpenIPMP system comprises user management and identification, content encryption algorithms as well as distribution channel protection. OpenIPMP is designed based on a set of open standards, including OMA DRM v2.0 [1], Internet Streaming Media Alliance (ISMA) encryption and DRM signaling (ISMACrypt) [16] and MPEG-IPMP [2] specifications.

For multimedia information there are mainly two types of encryption algorithms available in OpenIPMP: for streamed content, encryption uses stream ISMAcryp or DVB-CSA cipher, whereas for file based content, it uses AES or Blowfish block cipher.

DReaM

DReaM project [15] is a Sun Microsystem initiative to develop a DRM solution based on open standards. DReaM is build upon Opera, a former DRM interoperability specification and implementation in the Opera Eurescom project [17].

DReaM architectural structure supports the separation between the rights management, user authentication and identification, licensing, rights enforcement and protection systems [15]. This disintermediation enables the choice and selection of these technologies independent of each other without any compromise for the overall solution. There are two key elements for disintermediation in DReaM: separation of rights management from the content protection systems and separation of identity and authentication services from individual hardware devices.

DReaM has a central objective towards the creation of an interoperable DRM, offering the capability to interoperate directly with other content protection technologies and supporting services that enable both Conditional Access System (CAS) and DRM.

Because of its key architectural concept, DReaM platform enables multiple instances of these components to exist in a DRM/CAS system. Also, because of its disintermediation, DReaM system allows coexistence and integration of multiple instances of content protection specific components (player, licensor and packager) and components that are not content protection specific (licensing conductor, contracts manager, authentication service, shop and transaction system, custom content delivery system, etc).

Marlin

Marlin [18] is open-standard DRM initiative, developed by Marlin Developer Community (MDC) with the aim of creating an inter-vendor interoperable platform. Based on the previous Nemo and Octoplus projects [19], Marlin system provides a set of capabilities for managing relationships among services, network, and digital content.

Being based on a general-purpose, expressive DRM architecture (Octopus) Marlin rights management allows for substantial flexibility and control.

As in Octopus, Marlin node objects represents system entities (users and devices), and links between nodes represent relationships. This graph system is used to manage where, how, and when content can be used.

To determine if a client has rights over DRM protected content, Marlin client must determine a series of links that connects the user to the subscription. When purchasing content, Marlin client is instructed to request a ‘User Node’, corresponding to the user, and a ‘Link Node’. In Marlin system, the responsibility for links creation is assigned to the e-commerce systems that implements Marlin compliant service.

Marlin architecture also includes an OMA DRM Gateway, which enables Marlin clients to behave as OMA DRM agents. This fact satisfies all requirements for an OMA DRM agent, and therefore, can be considered to be an OMA DRM agent. Having this situation, OMA content can be received, processed, and consumed as on any other OMA DRM compliant device without any modification required on the OMA Rights Issuers component.

An important feature of Marlin is that it avoids the usage of Rights Expressions Languages and so it avoids patent issue regarding Rights Expression Languages.

V. PROPOSED ARCHITECTURAL FRAMEWORK

After evaluating important existing DRM solution [1-2], [14-15], several differences can be denoted among these systems architecture. Most of them involve the same basic actor with similar common operation and it is obvious till now that they follow the same pattern.

As we already mentioned, our goal is to determine the common base architectural framework for DRM systems. In this section, we describe our proposed DRM architectural framework. The synthesized approach, described bellow as the overall DRM architecture, is a generic approach without any content specific technological involvement.

From the architectural point of view, we have chosen the PKI infrastructure to provide capability of effective protection and authentication functionality. The reason for this consideration is justified by the intensive public key operation inside DRM model. Even if not all DRM solution include CA integration [1-2], [14-15] in their functionality, it is easily notable that important PKI functionalities are included in their inside operations.

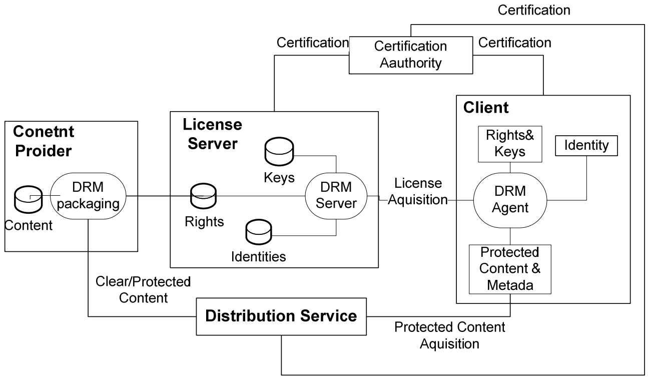

Figure 4. DRM Overall Architecture

In the overall architecture, shown in Fig. 4, content providers (CP), distribution services (DS), license servers (LS), certification authorities (CA) and client (C) interact together driven by specific DRM protocols following the design patterns described in previous sections. The content provider creates the content and distributes it to distribution service using some secure channels. Because this is usually achieved in a secure environment (private or not exposed network communications or secured delivery as SSL, HTTPS, SRTP, FTPS, etc), neither content provider to licensing service nor content provider to distribution service will be treated form a security perspective. But if this is not the case, the security measures described bellow can be easily extended to this situation as well.

In our proposed model, client distribution channels can be anything from web servers to peer-to-peer networks and the license server is used only for acquiring licenses needed to consume the content. Client content distribution can be operated by DS, or CP may have to be contracted to grant licenses usage. There is quite a large liberty to model DRM operations and is up to the implementation business logic to determine which should be the best operational design. In this paper we only describe the most typical case scenario raised in DRM systems.

The certification authority (CA) is part of the public-key infrastructure (PKI). Its role is to provide legal association between the identities of system principals (DS, LS, Client) and their encryption key pairs by using digital certificates.

As already explained in the previous DRM functionality description, the client incorporates controlled components in its system. As a trusted part of the DRM model, the DRM Agent resides within the user terminals as an elementary functional subsystem.

Starting from the functional architecture description, we synthesize DRM transactions in several phases (Fig.5):

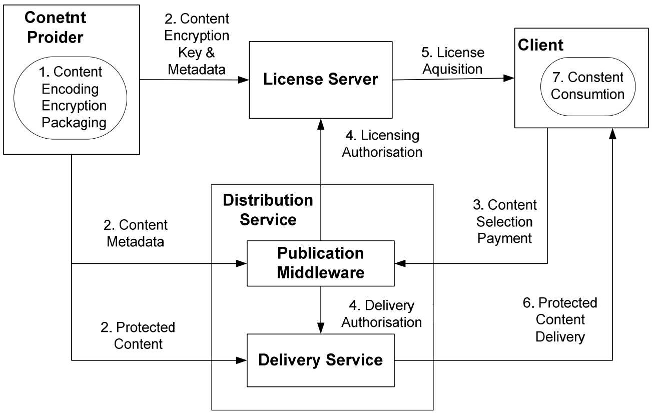
Conetnt Proider
1. Content Encoding Encryption Packaging
2. Content Encryption Key & Metadata
License Server
5. License Aquisition
Client
7. Constent Consumtion
4. Licensing Authorisation
Distribution Service
2. Content Metadata
3. Content Selection Payment
Publication Middleware
4. Delivery Authorisation
6. Protected Content Delivery
2. Protected Content
Delivery Service

Figure 5. DRM - functional workflow

1**. Creating content, content rights, metadata and containers**: trusted users, entities in DRM system, creates and manage protected content using authoring applications and tools that incorporate DRM technology such as watermarking [20], encryption, etc.

Basically are two methods to protect content: first, the content is watermarked and secondly, it is encrypted. Encryption is performed by scrambling the media on content level, not by encrypting the whole media container (file or stream).

This method is scientifically called “selective encryption” [21] and enables encryption only of some special selected parts of the media based on the fact that some formats can not be “consumed” (used) if some of its parts of it are corrupted. This technique significantly reduces computing power needed both at server and at the client side terminal.

At this phase content provider usually generates content identifier (ID_C), content metadata (M) and content encryption secret key ($K_S^{ID_C}$).

2. **Generating keys** used for content encryption or other afferent metadata (specific delivery information, content description, seeds, etc) will be made available to the LS. Also, after finishing content preparation and packaging, necessary information will be sent to DS (M, content description, licensing details, etc).

Used notation:

$K_A^{\Pr}$ - private key of A

K_A^{Pu} - public key of A

$CERT\_A$ - digital certificate of A

$K(M)$ - encryption of M using K (symmetric or asymmetric encryption depending of the key type)

$M1 \| M2$ - message $M1$ concatenated with message $M2$

3. After selecting his desired content, client C sends his **content request** together with his own certificate ($CERT\_C$) to DS. To simplify these cryptographic mechanisms, the client request is materialized by signing ID_C.

$$C \rightarrow DS : CERT\_C \,\|\, K_C^{\Pr}(ID_C)$$

4. After client request authentication and payment acknowledgement, DS signals LS and service delivery, **authorizing the client** identified by $CERT\_C$ for content request ID_C.

$$DS \rightarrow LS : CERT\_DS \,\|\, K_{DS}^{\Pr}(ID_C) \,\|\, CERT\_C$$

In some situations where communication between these entities occurs in unsecured environment it is necessary to protect exchanged messages either by using specialized protocols such as SSL, or by using already acknowledged public keys asymmetric encryption.

$$DS \rightarrow LS : K_{LS}^{Pu}\left(CERT\_DS \,\|\, K_{DS}^{\Pr}(ID_C) \,\|\, CERT\_C\right)$$

5. **Licensing information and rights-object distribution**: at this phase, LS verifies the identity of the distribution service DS, client C and license request signature then extracts the content identifier.

$$LS : VER(CERT\_DS), VER(CERT\_C)$$

$$ID_C = K_{DS}^{Pu}\left(K_{DS}^{\Pr}(ID_C)\right),$$

where $VER(CERT)$ is the certificate validation operation.

The licensing service will issue a license object within the user rights expression and necessary cryptographic information for content consumption and sent it to client DRM agent.

$$LS : L = URE \,\|\, K_S^{ID_C},$$

where URE is *User Rights Expression.*

$$LS \rightarrow C : K_C^{Pu}\left(K_{LS}^{\Pr}(L)\right)$$

The license object needs not only to be confidential but also provide integrity and non-repudiation. For such considerations it will be first encrypted with licensing authority private key and then with client public key.

6. **Licenses acquisition, license interpretation, license utilization**: content consumption is possible only after DRM client will authenticate the license message and decrypt content encryption key and usage policies from the license object.

$$C : L' = K_C^{\Pr}(L^*) = K_C^{\Pr}\left(K_C^{Pu}\left(K_{LS}^{\Pr}(L)\right)\right) = K_{LS}^{\Pr}(L)$$

$$L = K_{LS}^{Pu}(L') = K_{LS}^{Pu}\left(K_{LS}^{\Pr}(L)\right),$$

where L^* is the received license message and L' is the decrypted license object.

Because all messages between LS, DS and C are encrypted using private key and digital certificates, these signatures can also be used for non-repudiation purposes. Message achievement is an often used technique to provide accountability.

7. **Content consumption**: after all necessary information (keys, usage rights, and content) is acquired final content usage takes place.

At this phase important security measures must be implemented inside client subsystem while content is being decrypted and rendered. In this ecosystem the client component security leaks can compromise the entire content protection measures. Having this risks in mind, the solution for any DRM enabled content delivery system should include security mechanism to deal with client component corruption.

As a framework for developing DRM architecture to business specific environment we have proposed PKI usage to provide cryptographic support for entities authentication and usage rights protection through asymmetric encryption of license object. Protecting license objects, which are issued dedicated to every specific user request with license issuer private key, can also be used to guarantee service non-repudiation.

Users are discouraged from sharing their private keys and certificates, as keys can be used by other parities to purchase content on charge of original user.

In our opinion, an important contribution to the overall DRM system is materialized by the DRM component interactions. We consider as essential the existence of interaction policies guided by so called “middleware”. From a security perspective, this entity has the responsibility to signal delivery and licensing services ensuring that only legitimate users can access the system.

Dealing with every user and content request autonomously, the protocol described previously in this section and depicted in Fig. 5, provides a great security advantage in what concerns user and content management.

The coordinated phases within DRM components interaction have the goal to ensure secure transactions between these components and also to minimize system exposure to user domain. As it can be easily determined, a user can not achieve content delivery (even if it is already encryption protected) if he didn't pass appropriate middleware constraints (authentication, payment fulfillment, etc).

If request archival feature is added to the system, user accountability can be achieved, too. This is an essential feature in system like Pay-per-View or Video-on-Demand.

From our opinion, but also from industry perspective [22], an imperative constrain of the entire DRM scheme is the security of the DRM agent. In the overall architecture, DRM agent is presented as being a trusted part of the DRM model which calls for a compromise between DRM agent scalability (in the terms of different system ports) on one hand and security vulnerabilities on the other hand.

We have tried to minimize presentation to only DRM ecosystems fundamentals, without leaving out essential uncovered technical issues.

VI. CONCLUSION

The goal of our research was to synthesize a DRM architectural framework into a generalized view that can be applied to various types of online goods delivery systems. The proposed system comprises strong cryptographic mechanisms for content intellectual property protection while providing flexibility for technological implementation options.

Industry and academic researches are now concerned in open standards DRM development to allow interoperability and not force content providers to encode their works in proprietary formats or systems.

In the future research we will focus on DRM tools to enforce the content protection for different distribution systems and to identify different cryptographic schemes to enable DRM usage for P2P networks as well as for mobile terminals with restricted processing power.

During our research on the DRM systems, we also intend to propose a method for detection of DRM agent corruption detection based on mobile code and mobile agents self protection mechanisms.

REFERENCES

[1] Open Mobile Alliance Digital Rights Management V2.0 [Online]. Available: http://www.openmobilealliance.org/
[2] MPEG-4, IPMP-X (Intellectual Property Management and Protection Extension) [Online]. Available: http://www.mpeg.org/
[3] IETF Internet Digital Rights Management (DRM) Working Group, http://www.ietf.org/
[4] A. Arnab, A. Hutchison, "A Requirement Analysis of Enterprise DRM Systems", Proceedings of Information Security South Africa (ISSA) Conference, Johannesburg, South Africa, 2005.
[5] R.H. Koenen, J. Lacy, M. Mackay, S. Mitchell, "The long march to interoperable digital rights management", Proceedings of the IEEE, vol. 92, pp. 883-897, 2004.
[6] R. Iannella, "Digital Rights Management (DRM) Architectures", D-Lib Magazine, 7(6), 2001.
[7] J.M. Boucqueau, "Digital Rights Management", IEEE Emerging Technology Portal [Online]. Available: http://www.ieee.org/portal/site/emergingtech/techindex.jsp?techId=67
[8] Q. Liu, R. Safavi-Naini, N. P. Sheppard, "Digital rights management for content distribution", Proceedings of the First Australasian Information Security Workshop (ACSW2003), 2003.
[9] M. Stamp, "Digital rights management: the technology behind the hype", Journal of Electronic Commerce Research, Vol. 4, No. 3, 2003.
[10] Open Digital Rights Language [Online]. Available: http://odrl.net/
[11] eXtensible Rights Markup Language [Online]. Available: http://www.xrml.org/
[12] N. Rump, "Definition, aspects, and overview" Digital Rights Management, pp. 315, Springer Berlin / Heidelberg, 2003.
[13] E. D. Thomson, C. Sevig, "A four-layer model for security of digital rights management", Proceedings of the 8th ACM workshop on Digital Rights Management (DRM'08), 2008.
[14] Objectlab, OpenIPMP [Online]. Available: http://objectlab.com/openIPMP.html.
[15] G. Fernando, T. Jacobs, V. Swaminathan, "Project DReaM - An Architectural Overview", White Paper, Sun Labs, 2005.
[16] Internet Streaming Media Alliance, Encryption & Authentication Specification 2.0 (ISMA Cryp 2.0) [Online]. Available: http://www.mpegif.org/m4if/bod/ISMA/ISMA_E&Aspec2.0.pdf
[17] S. Wegner, "OPERA - Interoperability of Digital Rights Management (DRM) technologies, An Open DRM Architecture", Project Report, EURESCOM, 2003.
[18] Marlin Developer Community (MDC), Marlin Architecture Overview [Online]. Available: http://www.marlin-community.com/
[19] Marlin Developer Community (MDC). The Role of Octopus in Marlin [Online]. Available: http://www.marlin-community.com/
[20] F. Hartung, F. Ramme, "Digital rights management and watermarking of multimedia content for m-commerce applications", IEEE Communications Magazine, Vol. 38, No. 11, pp 78-84, 2000.
[21] X. Liu, A. Eskicioglu, "Selective encryption of multimedia content in distribution networks: challenges and new directions", Proceedings of the 2nd IASTED International Conference on Communications, Internet and Information Technology (CIIT2003), pp 527-533, 2003.
[22] B. A. LaMacchia, "Key challenges in DRM: An industry perspective", Proceedings of Digital Rights Management Workshop, pages 51–60, 2002.