Distributed DRM System

Technical Report – CS04-27-00

Alapan Arnab
Department of Computer Science
University of Cape Town
Cape Town, South Africa
aarnab@cs.uct.ac.za

Andrew C.M. Hutchison
Department of Computer Science
University of Cape Town
Cape Town, South Africa
hutch@cs.uct.ac.za

ABSTRACT

There is no standardised framework for digital rights management (DRM). With the proliferation of DRM systems, there is an increasing need for portability across multiple platforms and DRM systems. Current DRM systems can also be considered incomplete. Some DRM systems are not scalable enough; some are too focused on a particular application/file format and most do not have adequate mechanisms to address all the needs of the end users of the DRM protected works. In this paper we outline a proposal for an open, componentized rights management framework. This framework includes the architecture and a set of features that we believe solves the requirements for a DRM system.

1. INTRODUCTION

Security of data is no longer only about access to data, but what can be done with the data once access is granted. Encryption is no longer enough, as there is no control on what an authorised user does with the unencrypted data. Furthermore, with the introduction of new laws there are now legislative requirements for corporations and government institutions to control access to their customers' private details. Thus there is a need to create mechanisms to control access to data and control what authorised users can do with the data.

Rosenblatt et al. have defined Digital Rights Management (DRM) as "*persistent protection of digital data*" [24]. Currently, the main focus of DRM systems has been fighting piracy of digital media such as music. However the success of DRM in these endeavours has been mixed. Critics of DRM systems have protested that these systems give too much power to the rights holders [20, 5]. Others, like Felten [17], have commented that legal definitions of copyright and fair use are too complex to ever be implemented correctly in DRM systems. However, the success of Apple's iTunes Music Store [15] does show that the consumer is willing to buy DRM enabled music, as long as the restrictions are reasonable.

This work is supported through grants from the UCT Council, the KW Johnstone and Daimler-Chrysler scholarships. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of UCT or the trustees of the UCT Council, KW Johnstone and Daimler-Chrysler scholarships.

Unlike the use of DRM as an anti-piracy mechanism, usage of DRM and equivalent technologies like Content Management Systems (CMS) [11] or Enterprise Rights Management Systems (ERMS) [8] in enterprises do not have many of the legal overheads. It is acceptable and in many cases common practice for a corporation to define regulations on how their property can be accessed and utilised by their employees and sub contractors [23]. For example, most companies ask their employees to sign confidentiality agreements and have security protocols defining which employees have access to confidential data. However, the concepts in rights management do not differ too much from the DRM concepts for digital media.

Another problem with current DRM systems is that they are not scalable enough to be involed in the full production cycle. For example, in an investigation of pirated movies on the Internet, Byers et al. discovered that the majority of the pirated copies were due to leaks during the production of the movie and not due to members of the public [14]. DRM is required in the full production cycle, but the authors felt that current DRM technologies were not scalable, did not integrate seamlessly with the existing infrastructure and did not support some of the complex policies required in the production cycle.

Most of the current DRM solutions cater for specific file types and applications. Except for Microsoft's Right Management Services (RMS), there are no generic platforms to create and distribute DRM enabled data. Creation of DRM enabled data has been largely left to the music and other media companies, and while RMS allows Microsoft Office 2003 users to enable DRM protection on their works, the current RMS framework restricts the usage of such works to a limited set of users[1].

In this paper, we outline a proposal for an open, componentized rights management framework. We outline an architecture and a set of features that we believe would make an ideal DRM framework. We believe that this framework could then be used as a standard to enable cross platform DRM delivery. In the next section, we outline the goals that we believe an ideal DRM system must fulfil.

2. GOALS

The use of a DRM framework is not necessarily restricted to distributing digital media (like music) or protecting sensitive documents like company financial reports. It should be possible to ex-

[1]RMS is discussed in more detail in section 3.4

tend the use of the DRM framework to cater for software licensing (by using a DRM use license instead of license keys) or as a security mechanism in shared computing environments like grid computing.

Our aim is to create a framework that

1. is scalable,
2. meets the requirements of a general rights management system,
3. allows seamless migration between implementations of the framework, and
4. that allows implementations to be fully portable across multiple operating systems and platforms.

Furthermore, a DRM system built using the framework must

1. be able to handle different file formats,
2. provide flexibility for different levels of security (for example allow different encryption algorithms),
3. provide flexibility in enforcement – not all DRM works require strong enforcement[2],
4. be able to handle most fair use scenarios,
5. promote user privacy and not monitor usage of DRM data,
6. allow for the transfer of rights,
7. allow for flexibility in rights implementations,
8. have a mechanism to collect revenue for rights holders if required,
9. create a secure distribution channel and
10. prevent the illegal use of DRM protected works

3. BACKGROUND

There are currently many frameworks for rights management. However, we believe that these frameworks are incomplete, and in many cases flawed in their approaches. In this section we review a few key frameworks that we will build upon. In section 3.1, we discuss the roles and players in DRM systems as proposed by Bartolini et al. in 1999. In section 3.2 we look at the secure distribution architectures as proposed by Park et al. in 2000, following that we have an overview of rights expression languages in section 3.3. Finally we discuss four frameworks for rights management: Microsoft's RMS platform, Open SDRM, Real Network's Helix DRM and Sidespace's Media-S.

In section 4 we outline some of the main flaws in the DRM systems discussed in this section. In section 5 we use the frameworks discussed in this section to build our proposed framework and outline how our framework overcomes the flaws of the current systems.

[2]For instance, it can be argued that iTunes provides a weaker level of enforcement of DRM (since a song can be written to any number of CDs or iPods where DRM is not enforced). This is opposed to Microsoft's RMS which can require the user to authenticate himself/herself every time they wish to access protected data.

3.1 Roles and Players in a DRM System

Bartolini et al. in [11] discussed a set of roles in a content management system. These roles were:

Author or Creator: Entity that creates the work that is right protected. The creator does not have to be human.

Rights Holder: Manages the use of the work, issues licenses, collects royalties and essentially manages the work. In effect the rights holder is the *owner* of the work. The rights holder is not necessarily the creator of the work.

Service Producer: Packages the works in a security envelop to enable rights management. The service provider can also embed other security features such as watermarks, fingerprints etc. into the digital work.

IPR Register/License server: Maps the rights associated with a work to the user requesting the work.

Unique Number Issuer (UNI): Creates and manages unique identifiers for each of the DRM enabled works. The UNI could also issue *Uniform Resource Identifiers (URIs)*, and thus create and manage unique identifiers for all parties (creator, user, rights holders etc) and resources (works, licenses etc) in the DRM system.

Media Distributor/Service Provider: Responsible for distributing the work, and if required collecting fees from users. In some of the current systems, like the Apple iTunes Music Store, the service provider and the service producer are the same.

Controller: A Trusted Third Party (TTP) responsible for monitoring of the legality of the transactions.

Certificate Authority: Another TTP responsible for authenticating the parties in a DRM transaction.

Bartolini et al. did not consider the user as a player or as having a role in their discussion. We believe that the end user has an important role in a DRM system and thus in an earlier work [9], we proposed the addition of the user as a player in a DRM transaction. We also extended the roles played by the controller and the certificate authority to cover the transactions between the user and the media distributors. Figure 1, shows all the players in current DRM systems. However, most current DRM systems do not make use of any TTPs.

3.2 Secure Distribution Architectures

In [21], Park et al. discusses the various security architectures that could be used for secure content distribution. They gave three factors that distinguished the different architectures outlined below, and shown in figure 2

1. **Virtual Machine (VM):** Park et al. defined a virtual machine as "software that runs on top of [a] vulnerable computing environment and employs control functions to provide the means to protect and manage access and usage of digital information" [21]. Because the VM is

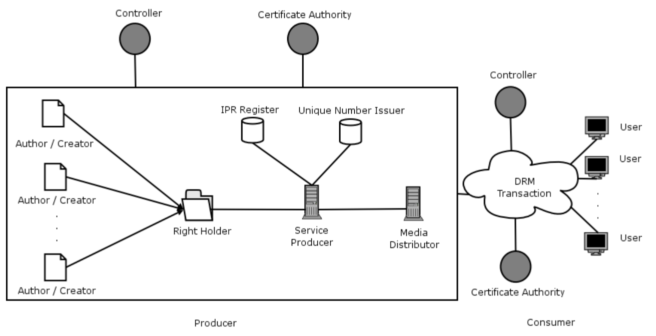

Figure 1: Players in a DRM System
[9]

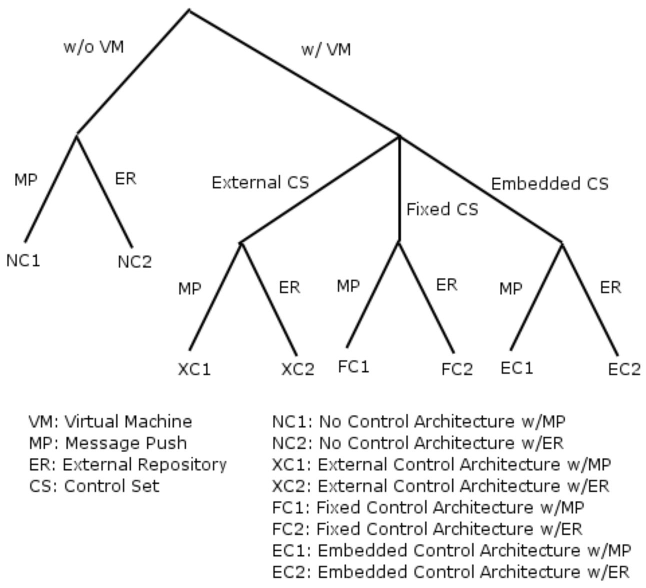
w/o VM
w/ VM
MP
ER
NC1
NC2
External CS
Fixed CS
Embedded CS
MP
ER
MP
ER
MP
ER
XC1
XC2
FC1
FC2
EC1
EC2
VM: Virtual Machine
MP: Message Push
ER: External Repository
CS: Control Set
NC1: No Control Architecture w/MP
NC2: No Control Architecture w/ER
XC1: External Control Architecture w/MP
XC2: External Control Architecture w/ER
FC1: Fixed Control Architecture w/MP
FC2: Fixed Control Architecture w/ER
EC1: Embedded Control Architecture w/MP
EC2: Embedded Control Architecture w/ER

Figure 2: DRM Distribution Architectures
[21]

necessary to enforce the restrictions, architectures without VMs are not suitable for DRM systems. Rosenblatt has further extended the VM definition to include DRM controllers that can be implemented at a hardware level, or as an operating system module [23].

2. **Control Sets:** Control sets are the rules that govern the usage of a DRM protected work. These rules can be expressed using rights expression languages (discussed further in section 3.3). Park et al. categorised control sets into three types, *fixed*, *embedded* and *external*.

   Fixed control sets are fixed to the VM, and are the easiest to implement. However they offer the least flexibility, and can become useless if they are compromised and the VM can not be updated with a newer control set. *Embedded control sets* are packaged together with the DRM protected work. They offer more flexibility, but changing the rules in the control set becomes more difficult. *External control sets* are separate to the DRM protected work. The biggest advantage is the ability to address the rules of multiple works in one control set. These control sets offer the most flexibility and also allow for easier change of the rules.

3. **Distribution Style:** Park et al. distinguished distribution processes as either *message push* or *external repository*. In a message push system, the data is transferred between the creator and the user by a direct communication channel (like email). In an external repository, the user fetches the data from a central repository. Using external repositories also allow streaming data to the user, and thus the user would not need to store the data locally. Neither distribution has any security advantages over the other, but external repositories that allow only streaming allow more control to the rights holders.

All DRM systems use virtual machines, and most use a combination of fixed, embedded and external control sets. Music stores currently use a central server approach to selling music downloads, but Microsoft has offered the alternative of downloading the music file from any location and then downloading a license to play the file (i.e. an external control set architecture) with Windows Media 9 [18].

3.3 Rights Expression Languages

Rights expression languages (RELs) allow for the definition of the constraints, permissions etc. that the rights holder gives to the user and effectively forms the control set as defined in 3.2. In effect, RELs allow for the expression of a usage contract between two or more parties.

The foundations of modern RELs were laid by Marc Stefik at Xerox Parc in the 1990s with Digital Property Rights Language (DPRL) [16]. DPRL was subsequently developed into eXtended rights Markup Language (XrML) by ContentGuard, a company founded by Microsoft and Xerox. Other languages like Open Digital Rights Language (ODRL) and Creative Commons (CC) have arisen to create open rights language specifications. Most modern RELs, like XrML and ODRL, are expressed in XML, using XML Schemas to define the syntax and grammar of the language; enabling portability across operating systems and platforms.

In the XrML 2.0 specifications [4], the requirements of a REL are given as:

- **Comprehensive:** A language that shall be capable of expressing simple and complex rights in any stage in a workflow, lifecycle or business model.

- **Generic:** A language shall be capable of describing rights for any type of digital content or service (an ebook, a file system, a video or a piece of software)

- **Precise:** a language shall communicate precise meaning to all players in the system.

Many of the criticisms of DRM systems stem from the current inability of RELs to express all the legal rights expected by the consumer [17, 5]. Felten has argued that some legal rights like fair use of copyrighted material, can never be expressed using RELs [17]. In [12] Bechtold countered that, while RELs cannot express fair use cases in general; they can do so in individual cases. However, as Mulligan et al. in [19] pointed out, current RELs do not have any syntax or grammatical capabilities to express communication between the end user and the rights holder. This limits the usage of RELs for contract negotiations, and thus current specifications of XrML, ODRL and other RELs are not comprehensive.

3.4 Microsoft RMS

Microsoft's Rights Management Services (RMS) platform aims to provide an end-to-end protection of RMS-protected data from unauthorised use. Unlike other DRM systems currently available, RMS can be implemented with any application (although this is restricted to only certain Microsoft Operating Systems) and on any data type. The RMS package consists of a client module, a server module, and a software development kit (SDK) for developers to create RMS enabled applications.

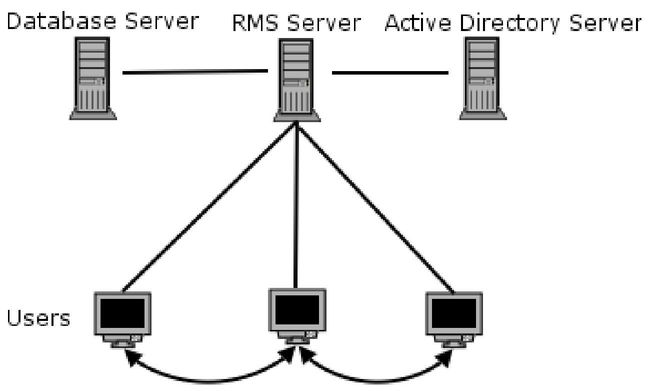

Figure 3: Microsoft RMS Architecture
[7]

Figure 3 shows the components of a RMS deployment. RMS does not make any distinction between a *creator* and an *end user*. Instead, there are only users of a RMS system. The user machines requires the installation of an RMS client software, an operating system module for Windows 98 SE and later, which effectively is the VM as described in section 3.2. The advantage of this approach is two fold; firstly, there is one VM for rights management instead of one for each application; and secondly RMS can override "control+c" or "print screen" commands thus enhancing security.

The *RMS Server* performs the roles of the service producer[3], the unique number issuer and if required that of a certificate authority. RMS utilises an active directory server as a mechanism for authenticating users while the database server (currently RMS specifies the use of Microsoft SQL 2000 Server) to fulfil the role of an IPR Register as well as to perform logging of user activity. RMS does not specify means of distribution. Furthermore, RMS does not make any use of an independent controller, although it can be argued that by keeping logs of every transaction, the database server performs or provides for much of the functionalities required in a controller.

RMS requires that all entities in a RMS transaction be a *trusted entity* [6]. One RMS server needs to be enrolled with Microsoft's RMS Server Enrolment Server. This server can then act as the certificate authority for the enterprise. The enrolment process merely involves signing the public key of the entities. Users are also issued certificates (expressed in XrML) that associate users with certain computers. RMS uses 1024-bit RSA key pairs and 128 bit AES encryption. Licenses, rights and certificates are expressed in XrML while communication between servers and clients is done using the SOAP protocol.

3.5 OpenSDRM

Open and Secure Digital Rights Management Solution (Open SDRM) is an open source project developed by the Moses[4] consortium [25]. The Open SDRM project aims to create a secure framework for delivering multimedia on the Internet, and caters for the creation of the secure content, payment collection, distribution and rendering of multimedia. Figure 4 shows an overview of the Open SDRM architecture.

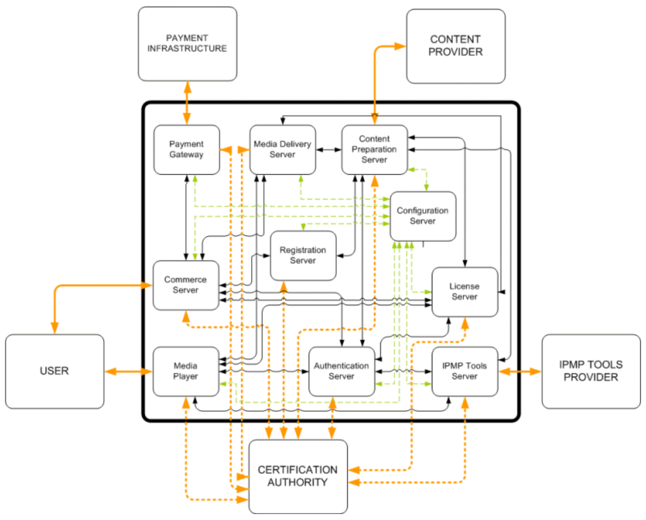

Figure 4: Open SDRM Architecture
[3]

Open SDRM is highly componentized, and allows for external components to plug in to the Open SDRM architecture. The *IPMP Tools* component for example, allows for encryption, watermarking and other related tools to be applied to the content being protected. The Open SDRM architecture uses all the roles suggested in section 3.1, and adds some roles that are not necessarily part of the DRM architecture, like a payment gateway.

Unfortunately, specifications and documentation of the framework is incomplete. For example, there is little documentation on the

[3]Please see section 3.1 for a description of the roles in a DRM system

[4]Moses: MPEG Open Security for Embedded Systems. http://www.crl.co.uk/projects/moses/index.htm

communications protocols used between the internal modules or how external modules interact with the internal modules. The sourceforge project site[5] does not mention any implementations of the Open SDRM framework. This makes it very difficult to evaluate the framework.

3.6 Heilx DRM

Real Networks promoted Helix DRM as *"The first multi-format digital rights management platform for secure delivery of media to any device"*. Helix DRM is based on Real's latest media codecs (Helix) and is aimed at creating a secure DRM system for media files, including streaming media. At its core, Helix DRM consists of three components – a DRM packager to package the media file in question; a license server for granting end users use licenses and client software to render the media. There is also an open source project led by Real to create "Device DRM" that can be used to implement the DRM controller natively on hardware devices.

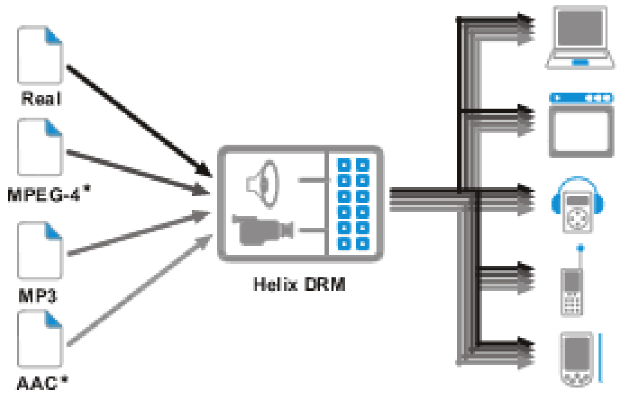

Figure 5: Helix DRM Architecture
[2]

There are is no documentation on Real's website[6] on the protocols and processes used inside the system – just a description of the components and their features. There is nothing that really distinguishes the DRM system from other media DRM systsems like iTunes or Microsofts Janus Windows Media system, except for the support for multiple file formats.

3.7 Media-S

Like Helix-DRM, Media-S is geared to be a multi-format DRM system for delivering multimedia. Like OpenSDRM, Media-S is an open source project. Media-S, Helix DRM and OpenSDRM all share similar characteristics in their componentised structure; and similarly Media-S is divided into three components – a packager, a client and a license server. However, the security mechanisms trumpeted in Media-S have been criticised, with some comments labelling Media-S security as virtually non-existent. While the Media-S project does seem to be more active than OpenSDRM; the last release of their software (Beta-2) was in April 2003. Like OpenSDRM, this project also seems to be dead.

[5]http://sourceforge.net/projects/opensdrm/
[6]http://www.realnetworks.com/products/drm/

3.8 Summary

Our proposed DRM framework builds upon the work mentioned in this section. The roles identified by Bartolini et al. form a crucial role in determining the architecture of our framework. In section 4, we discuss the problems with Microsoft RMS and other current DRM systems and then in section 5, we detail our proposed framework and how it overcomes these problems.

4. PROBLEM DEFINITION

Our proposal is to create an open framework for rights management. While RMS is the most complete rights management package available, we believe that RMS is still incomplete, and changes are required for RMS to be a complete solution for general rights management. Likewise, Open SDRM is an incomplete framework, although it does feature some components that are not present in RMS. However, Open SDRM is focused on multimedia delivery, and is not presented as a full rights management system. The same criticism can be directed at Helix DRM and Media-S; although in the case of Helix, it should be possible to generalise the system to cater for any data file. Below we detail what we believe are the main problems in the RMS framework. Later in this section, we also discuss some of the problems with Real Helix and Media-S.

1. **User Authentication:** In our opinion, the use of active directory for user authentication severely limits the functionality of RMS. The major problem with the use of active directory is that it limits RMS to be used intra-enterprise only. Exchanging secure documents between enterprises, or between the enterprise and the customer requires the recipient to have an active directory account which can be a security risk; and a high overhead. RMS does support the use of Microsoft Passport for authentication also, but Passport itself has a reputation of security failures [13] which has led to many enterprises chosing not to use Passport for authentication. We do believe that the use of a federated identity management (like the federated identity schemes in the Liberty Alliance Project) as a basis for authentication should be supported in our DRM system.

2. **Duplicate Control Sets** In RMS, the DRM protected work has an embedded control set. However when the user wants to access the protected work, the client is required to fetch a *use license* from the RMS server and then only can the user access the work. This process in effect makes the embedded control set useless, since the restriction and rights can be put in the use license in the first place.

   In our opinion, a bigger problem arises should the end user require a change in their use license. This problem can be best represented using an example from the second scenario in Microsoft's overview of RMS [6]. Tom creates a document for Jill, and protects it using RMS. He specifies that the document can only be viewed and edited by Jill for one week. If Jill requires additional time, Tom is required to edit the rights to the document, extend the deadline and then redistribute the document to Jill. With this system; there is a high overhead in network bandwidth and requires active intervention by the rights holder/creator. This problem is increased when dealing with large data files like video.

3. **Lack of user query/feedback mechanism** As mentioned above, should a user request a change in rights; the user needs to communicate with the rights holder/author personally and then ask him/her to reissue the document with new rights. With small documents, the overhead other than irritation for the author/rights holder is not too high; but with larger data files like video, there is also a high bandwidth overhead. However it must be noted that no current DRM system has a feedback mechanism.

   As discussed in section 3.3, the RELs used in current DRM systems do not have the syntax or grammar for bi-directional communication. In [10], we detail our extensions to ODRL and XrML that enable bi-directional communication between two parties. Our extensions add syntax to ODRL and XrML that allow a user to request the rights holder to make changes to his/her current use license. The rights holders can also respond back to the end user granting or denying the request. With ODRL, it should be easy to use this mechanism to build up an *offer* from the rights holder where the user has their rights to a work tailored for them personally.

4. **Proprietary Solution** While RMS does make use of industry standards such as AES for encryption and SOAP for messaging, the protocols used in RMS are proprietary. It is highly likely that computers in an enterprise comprise of different operating systems and different hardware architectures, and thus RMS is not a viable solution for complete data protection.

5. **Payment Gateway** In fairness to Microsoft, RMS is not promoted as a general DRM solution. Thus, there are no mechanisms for payment and customer management systems. However, in a a generic DRM system; it is necessary to have a mechanisms to plug into the company's EFT and customer management systems.

There are also a problems with the standardisation of clients of all the systems described earlier. Every DRM system that is currently available require their own specific client, for example, music bought from iTunes Music Store cannot be played on any other applications except iTunes and the iPod. However, it must be noted that different platforms and operating systems do require different types of clients; since a client designed for a cell phone will have different requirements to one designed for PC. However, we believe that it should still be possible to create a specification that will detail the requirements for how a client should operate and the parameters it must accept. Thus any clients that support these requirements will then be able to inter-operate.

From the documentation provided for Media-S [**?**], there is no globaly unique mechanism in identifying a DRM protected object. This criticism can also be raised against Helix-DRM, but there is not enough documentation to form a definite opinion. The lack of a globally unique identity mechanism means that it could be easy to fake the license for one DRM protected data for another; and also has the potential to confuse the DRM client; if two different DRM enabled data had the same identifier. This makes Media-S unsuitable for mass deployment. Another criticism of Media-S and Helix-DRM is the use of non-standard license RELs. Both make use of their own REL specifications, even with the availability of two well documented REL standards (ODRL and XrML). The use of non-standard components make it even harder for other DRM systems to interoperate.

5. PROPOSED FRAMEWORK

Figure 6 gives an overview of the structure of our proposed DRM framework. In section 5.1, we give details on the structure of our architecture, while in section 5.2 we highlight the underlying features of our framework.

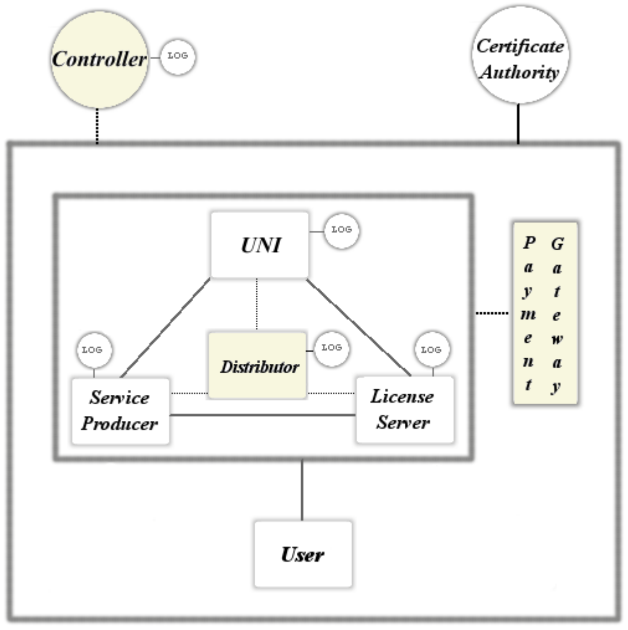

Figure 6: Overview of the proposed DRM Architecture

5.1 Proposed Architecture

As shown in figure 6, our architecture consists of eight different roles. Like RMS, we do not make a distinction between an end user and an author; and we add the role of a *Payment Gateway* which was not present in the roles described in section 3.1. We would also like to separate our proposed components into required and optional; with the Payment Gateway, the Controller and the Distributor being optional roles.

5.1.1 Certificate Authority [Required]

The certificate authority (CA) is a TTP that will be used to authenticate the keys of the parties involved in the DRM transaction. We propose the use of ITU-T X.509 standard for certificates; and thus enable any of the established certificate authorities (like Thawte and Verisign) to be CA's in the framework. Alternatively, PGP can also be used as the certificate standard, and the CA can be regarded as a trusted party with very high integrity.

5.1.2 UNI [Required]

The Unique Number Issuer (UNI) will be responsible for managing unique identifiers for every DRM enabled work. The identifiers should be globally unique, and as such one UNI should be able to serve a number of service producers. The Digital Object Identifier (DOI) has been promoted as a mechanism to uniquely identify intelectual property [1] and DRM protected data [22]. However, the DOI is not a secure mechanism because there is no mechanism to verify whether an object and the given identifier belong to each other. But the underlying handle mechanism does have a lot of advantages. Our current proposal is to make use of an adapted handle service that will be secure enough for use in a DRM system.

5.1.3 Service Producer [Required]

The service producer packages the work in a security envelope, together with the unique identifier from the UNI and creates a template for the rights that can be granted to an end user. The service producer and the author can be the same; but there could be situations where a separate service producer is desired (e.g. the rights holder of the work is not the same as the author). Should the service producer and the author be different machines; communication should be over a secure connection like SSL.

5.1.4 License Server [Required]

The license server serves two functions in our framework. Firstly, the license server hands out use licenses to the end user. The use licenses specify the rights that the user has on a DRM protected work. Should it be required, the license server can make use of the Payment Gateway for the end user to pay for the rights.

The second function of the license server is to handle requests from users for additional rights (see section 5.2.5 for more details on the request system). Some of these requests could be granted automatically (for free or for a price) which can be predetermined by the rights holder. The rights holder could also setup a set of requests to deny automatically. Otherwise the license server should communicate with the rights holder the requests from the end user; and the rights holder can then communicate back with the license server granting or denying the request. The use of a request system should allow for a work around for fair use; which is difficult to express in a REL. The control set is described in more detail in section 5.2.4

5.1.5 User [Required]

The user performs two of the roles mentioned in section 3.1 - the author and the end user. As an author, the user uses the service producer to create a DRM protected work. The user can then make use of a *distributor* (see section 5.1.6) or can distribute the work on his/her own. The author will also require a mechanism to receive requests from end users (forwarded by the license server) and respond to these requests.

As an end user, the user retrieves use licenses from License Server and then can use the DRM protected work as defined by the license. As mentioned before, the user would require a mechanism to request changes with the license server. A key component of the end user is the VM, and this is discussed in more detail in section 5.2.1.

5.1.6 Distributor [Optional]

A DRM protected work is required to have persistent protection regardless of where the work resides. For this reason, the distributor is not a requirement for the system, as DRM protection must work if the work is transferred over peer-to-peer networks, made available for download on the Internet etc. However the distributor does provide some interesting possibilities for DRM enabled work distribution.

5.1.7 Payment Gateway [Optional]

Like the Distributor, the Payment Gateway is an optional component. The Payment Gateway is only required where the framework is used for delivering commercial products, such as an online store. We are not too sure of the specifics of the EFT protocols for the payment gateway; but we believe they should be flexible enough to easily integrate with existing systems of the enterprise. Thus we are more interested in defining mechanisms that would allow other components to communicate with the payment gateway; which can then translate the information to the relevant protocol.

5.1.8 Controller [Optional]

In [11] Bartolini et al. put the Controller as a very important component of a DRM system. While the controller plays a very important role in commercial DRM systems (co-incidentally none of the current DRM system deployments use a controller), its use in an intra-enterprise scenario is not that important. For this reason, we have decided to cater for the Controller as an optional component in our framework. The use of a controller does increase the overall overhead and this could be a major factor against the use of a controller.

5.1.9 Logging [Configurable]

While the use of a controller is optional, the use of logs in a DRM system is of great importance. However the use of logs has its drawback. If used excessively, logs can be used to monitor the usage of DRM enabled work by end users. For this reason our framework must strike a balance between what actions should be logged (issue of a license, revocation of a license) and what should not be logged.

5.2 Underlying Features

In this section, we detail the underlying features of our framework. This includes more details on the VM for implementing the DRM controls and the control set.

5.2.1 VM Location

In the RMS platform, the VM [7] is an operating system module. In the OpenSDRM, Helix-DRM and Media-S the VM is at the application level and is present in the media player [25, 2, **?**]. Having a VM at an application level is less effective than at the operating system or hardware level since developers are required to include a VM engine for each application. However, both the use of a VM as an operating system module or a hardware level controller has their disadvantages. In [23] Rosenblatt contends that VM as an operating system level causes a performance hit as every I/O operation needs to be checked if it is allowed. While such a check is faster in hardware, hardware VMs have had an unflattering history with DVD-CSS and are also harder to deploy en mass.

[7]We use the term VM as described by Park et al. [21]

In our framework, we make a distinction between the implementation of a virtual machine and the framework for managing rights. The VM should be involved with purely the interpretation and enforcement of the control set. The framework should thus specify an interface which presents the VM with the DRM protected work, the control set and related keys and certificates. Rosenblatt in [23] states that hardware implementations of the VM are inevitable and thus we would like our framework to work with any form of VM. For low powered devices such as smart phones, hardware VM could lead to faster encryption and decryption, and hence better performance.

5.2.2 User Authentication

We realise that there are various degrees of flexibility required for end user authentication. In some instances, an online authentication is a neccessity while in other instances, end-users expect no further communication with the license server once they have an use licence. For offline authentication, we forsee the use of hardware based authentication tokens such as tokens from the Trusted Computing Platform as the main authentication mechanism. However current tokens are fixed to the PCs and thus provide no mobility, and a more mobile solution is required. User authentication is strongly linked to the use license, and the user also requires a globally unique identifier.

5.2.3 REL

As pointed out in [16], XrML and ODRL (and their respective derivatives) are currently the only general purpose RELs available. Thus either language could be used for expressing the control set in our framework. We are in favour of using ODRL because it already has a mechanism to describe rights that a rights holder would be willing to offer to the end user through its *offer* syntax. Since ODRL is also an open specification, it fits our purpose for an open framework.

5.2.4 Control Set

As described in section 4, the control set structure used in RMS has a high overhead. To overcome the overhead, we propose the use of external control sets in most cases. The framework will allow embedded control sets when the rights are the same for all cases (e.g. a brochure that is readable by any user) and embedded control sets must be overridden by an external control set. The use of an external control set also means that the distribution of the rights protected work does not necessarily need to be prescribed. However the mechanism for encryption of the work becomes trickier.

5.2.5 Bi-directional Communication

In [10], we present extensions to XrML and ODRL that allow for bi-directional communication. With bi-directional communication, the end user can ask the rights holders for changes in the use license; which is presently not possible in DRM systems. Another feature of bi-directional communication is the ability to negotiate contracts; and thus this would allow our framework to be used for implementing digital contracts.

Digital contracts could be used to replace the current anti-piracy techniques such as key-codes for software distribution. Using digital certificates and licenses are not new. 4Front Technologies for instance use PGP certificates for licensing Open Sound System (OSS) drivers for various Unix flavours to end users. Combined with a DRM VM, software licenses expressed as DRM use licenses could be more effective than current techniques. Such implementations would not require any additional extensions to the framework other than the requirement that the VM be implemented at operating system or hardware level.

6. FUTURE WORK

The first step of the project is to define the initial specifications of the framework. This includes specifying the exact details of each components (e.g. the format of the UNI) and the communication protocols used between the various components. The protocols defining the interaction with the VM will also be defined in this stage.

Following a set of specifications, our intention is to model our framework using a formal specification technique. This would allow us to test our framework for deadlocks in the communication protocols etc. We are going to investigate modelling tools one whether there is a tool that can check the entire model for deadlocks, scalability etc. We will use the results of the modelling to improve the specifications.

Once a full framework is defined and modelled, our intention is to create an open source project to develop the major components such as the DRM VM. We would also like to develop applications that can demonstrate each of the components of the framework. It should then be possible to verify whether the framework meets the goals set out in section 2. In the discussion below, we detail why our framework should be able to meet some of these goals.

1. **Scalability:** The use of components should lead to scalability of the system
2. **Portability:** The open specifications would allow for the implementation on multiple operating systems.
3. **Rights transfer:** With the use of bi-directional REL, a user can request for a change in end user.
4. **Flexibility in rights implementations:** Bi-directional REL allows for the user to request changes. The use of an external control set allows for individual licensing options.
5. **Collection of revenue:** The payment gateway should facilitate for the collection of revenue.

7. CONCLUSION

In this paper we have introduced a framework that could be seen as an ideal rights management system. We address many of the shortcomings of current DRM system, and with the use of a bi-directional REL allow for the possibility of granting fair use, although it will not necessarily be automatic.

The framework can be easily implemented for securing documents for any user, and not necessarily as a mechanism for selling multimedia. Thus the framework can be used as a single DRM mechanism for any data type; distributed by anyone implementing the full framework. This makes the framework more powerful than existing DRM systems.

8. ACKNOWLEDGEMENTS

This report is based on a paper that was originally submitted to the ACM Workshop on Digital Rights Management. While the paper was not accepted, the reviewers gave a lot of constructive criticisms, and many of the recomendations and criticisms have been incorporated and addressed in this report. We would like the thank the reviewers on their detailed reviews.

9. REFERENCES

[1] The digital object identifier system – frequently asked questions.
URL: http://www.doi.org/faq.html.

[2] Helix DRM 10 from real.
URL: http://www.realnetworks.com/products/drm/.

[3] Open SDRM architecture.
URL: http://193.136.190.37/opensdrm.no-ip.org/documentation/OPENSDRM_ARCHITECTURE.pdf.

[4] *eXtensible rights Markup Language (XrML) 2.0 Specification*, 2001.

[5] DRM From the Viewpoint of the Electronic Industry. *Slashdot* (2003).
URL: http://slashdot.org/article.pl?sid=03/11/25/1821218.

[6] Technical overview of windows rights management services for windows server 2003. White paper, Microsoft, 2003.

[7] Windows right management services - data sheet, 2003.
URL: http://www.microsoft.com/windowsserver2003/techinfo/overview/rmsdatasheet.mspx.

[8] Windows rights management services: Protecting electronic content in financial, healthcare, government and legal organizations, 2003.
URL: http://www.microsoft.com/windowsserver2003/techinfo/overview/rmsverticals.mspx.

[9] ARNAB, A., AND HUTCHISON, A. Digital Rights Management - An overview of Current Challenges and Solutions. In *Presented at Information Security South Africa (ISSA) Conference 2004* (2004).

[10] ARNAB, A., AND HUTCHISON, A. Extending ODRL and XrML to enable Bi-directional communication. Departmental technical report, no. cs04-28-00, University of Cape Town, 2004.

[11] BARTOLINI, F., CAPPELLINI, PIVA, A., FRINGUELLI, A., AND M, B. Electronic Copyright Management Systems: Requirements, Players and Technologies. In *Proceedings of the Tenth International Workshop on Database and Expert Systems Applications* (1999), IEEE, pp. 896–899.

[12] BECHTOLD, S. Digital Rights Management in the United States and Europe. IVir, Buma/Stemra - Copyright and the Music Industry: Digital Dilemmas.

[13] BECKER, D. Passport to nowhere? *C-Net News.com*.
URL: http://news.com.com/2100-7345_3-5177192.html.

[14] BYERS, S., CRANOR, L., KORMAN, D., MCDANIEL, P., AND CRONIN, E. Analysis of Security Vulnerabilities in the Movie Production and Distribution Process. In *Proceedings of the 2003 ACM Workshop on Digital Rights Management* (2003), ACM, pp. 1–12.
URL: http://doi.acm.org/10.1145/947380.947383.

[15] COHEN, P. iTunes hits the 50 million song mark. *The Industry Standard - Internet Business News* (2004).
URL: http://www.thestandard.com/article.php?story=20040315173205175.

[16] COYLE, K. Right Expression Languages, A report for the Library of Congress. Tech. rep., Library of Congress, USA, 2004.

[17] FELTEN, E. Skeptical view of DRM and Fair Use. *Communications of the ACM 46*, 4 (2003), 57–59.

[18] MICROSOFT. Microsoft windows media data session toolkit, 2003.

[19] MULLIGAN, D., AND BURSTEIN, A. Implementing Copyright Limitations in Right Expression Languages. In *Proceedings of the 2002 ACM workshop on Digital Rights Management* (2002), ACM.

[20] MULLIGAN, D., HAN, J., AND BURSTEIN, A. How DRM Based Content Delivery Systems Disrupt Expectations of ”Personal Use”. In *Proceedings of the 2003 ACM workshop on Digital Rights Management* (2003), ACM, pp. 77–89.
URL: http://doi.acm.org/10.1145/947380.947391.

[21] PARK, J., SANDHU, R., AND SCHIFALACQUA, J. Security architectures for controlled digital information dissemination. In *Proceedings of the 16th Annual Computer Security Applications Conference* (2000).

[22] ROSENBLATT, B. The digital object identifier. *The Journal of Electronic Publishing 3*, 2 (1997).
URL: http://www.press.umich.edu/jep/03-02/doi.html.

[23] ROSENBLATT, B. DRM for the Enterprise, 2004.

[24] ROSENBLATT, B., AND DYKSTRA, G. Integrating content management with digital rights management - imperatives and opportunities for digital content lifecycles. White paper, Giantsteps Media Technology Strategies, 2003.
URL: http://www.giantstepsmts.com/drm-cm_white_paper.htm.

[25] SERRAO, C., NEVES, D., BARKER, T., BALESTRI, M., AND KUDUMAKIS, P. Open SDRM – an open and secure digital rights management solution. Tech. rep., 2003.
URL: http://193.136.190.37/opensdrm.no-ip.org/documentation/opensdrm.pdf.