DRM as a Layered System

Pramod A. Jamkhedkar
Department of Computer Science
University of New Mexico
Albuquerque, NM 87131
pramod54@cs.unm.edu

Gregory L. Heileman
Department of Electrical & Computer Engineering
University of New Mexico
Albuquerque, NM 87131
heileman@ece.unm.edu

ABSTRACT

The current landscape for digital rights management (DRM) consists of various ad hoc technologies and platforms that largely focus on copy protection. The fragmented nature of the DRM industry in 2004 is somewhat reminiscent of the telecommunications industry in the late 1980's. At that time various networking technologies were available, and what was needed was a technology that could integrate existing networks and provide various services to users. The OSI layered framework and the TCP/IP communications protocol suite provided a solution to this situation. The OSI model divides the process of digital data communications into layers. Likewise, in this paper we divide the process of DRM into layers in which various services are offered to the users of digital content at each layer. Three blocks of layers have been identified. The upper layers deal with the end-to-end functions of the application, the middle layers deal with rights expression and interpretation, and the lower layers ensure rights enforcement. This paper describes how responsibilities might be distributed among the various layers, and considers where in these layers it would be appropriate to define protocols and standards.

Categories and Subject Descriptors

C.2.6 [**Computer-Communication Networks**]: Internetworking—*Standards*; H.5.1 [**Information Systems**]: Interfaces and Presentation—*Multimedia Information Systems*; K.5.1 [**Legal Aspects of Computing**]: Hardware/Software Protection—*Copyrights*

General Terms

Design, Legal Aspects, Security, Standardization

Keywords

DRM, Content Protection, OSI Layers

1. INTRODUCTION

The concept of digital rights management (DRM), beyond that of copy protection, was introduced in the late 1980's. By the late 1990's, IBM had implemented some of these notions in their Cryptolope technology [16, 17], and subsequently InterTrust, in what is perhaps the most ambitious DRM system implemented to date, released their Digibox technology [21]. Since that time, numerous products and technologies have been created that address various aspects of DRM. The commercial success of these products has been limited [7, 10], to say the least, which is surprising given the universally recognized importance of the problem. Pundits have provided a myriad of reasons for these failures, including ease-of-use arguments and the lack of either formal or de facto standards for DRM. In spite of the limited successes of the DRM systems that have been built to date, there are important lessons associated with these early attempts that may prove useful in the development of subsequent systems. For example, the large software infrastructure associated with the Digibox [28], which therefore necessitated a onerous client-side download, was resisted by end users. In addition, the lack of support for non-PDF formats was a limiting factor in Digibox adoption. The Cryptolope project attempted to implement not only content protection, but also to support superdistribution [16, 17]. Its lack of success was a result of factors similar to those just described, and both of these systems certainly suffered from their early introduction into a markets not yet prepared to accept DRM.

DRM systems have undergone an evolution from the earlier forms we have just mentioned. The aim of early DRM systems focused on content protection [18, 19], and therefore there was a heavy emphasis on client-side security technologies that attempted to implement what we have termed digital rights enforcement (DRE). Typically the important right that one was interested in enforcing was the "read only" right—if it were possible to enforce this right, then perhaps some of the problems associated with file sharing on peer-to-peer networks could be addressed. Thus, the first generation of DRM represented a substantial narrowing of DRM's scope and broader capabilities. We have lately seen a broadening of this viewpoint as discussions about the expression and implementation of digital rights, rather than rights enforcement, are common. Indeed there are now a number of competing proposed rights expression languages developed to express digital rights in machine-readable form [8, 15, 31]. Second generation generation DRM systems are sure to to include this technology, and will therefore offer the ability to implement more sophisticated DRM scenarios in more complicated environments, e.g., in end-to-end supply chain settings. Certainly the need for DRM standards will become even more important in second generation systems.

A lack of standards in the DRM space is also a reason often offered by content distributors for not deploying DRM solutions [1]. There have been discussions regarding what aspects of DRM technology should be standardized. Some have argued that DRM standards should only cover those aspects of content providers' business on which they will want to standardize [24]. That is, standards should not encompass those aspects that give a content provider their competitive advantage in the marketplace. However, this alone does not seem like the appropriate principle that should be guiding the development of standards in DRM. This led us to consider the importance of standards in DRM, what aspects of DRM should be standardized, and more importantly at what level they should be defined. When addressing these important issues, it makes sense to consider how standards have influenced the development of other similar technologies. In particular, it seems that much can be learned by reviewing the evolution of the Internet, and the role that standards have played in its evolution [2, 5]. One of the most important concepts that aided in the development of the Internet, along with its associated standards, was the treatment of it as a layered system. Specifically, the Open Source Initiative (OSI) layers provide a way for developers to create proprietary products that address specific telecommunications needs, while still allowing them to function within a larger setting that may include products from a large variety of vendors. It is our belief that similar benefits are possible in the DRM industry if the functionality of DRM as a whole where to be considered as a layered system.

In reconsidering first generation DRM systems in light of what we have just discussed, there was success in the development of numerous technologies that have served to advance the state-of-the-art in rights enforcement—secure containers, watermarking, encryption, digital certificates, and trusted computing platforms to name a few [9, 14, 26, 29]. Many of these technologies may play a specific role in a layered DRM structure. Furthermore, we can learn from the successes and failures of these earlier systems in constructing a layered framework for DRM that may provide a scaffolding upon with DRM technologies can be designed and built, just as the OSI layers helped to clarify and guide development in the telecommunications sector.

It seems that most first generation DRM systems failed because they either did not cover certain aspects of DRM that a customer required, or they crumbled under their own weight while trying to achieve everything. This points to the need for establishing an agreed upon DRM framework, where different kinds of services can be provided to the user leveraging the existing technologies for rights enforcement. A framework with a layered architecture would achieve this, allowing only those technologies required by a particular application to be included in a DRM system. Within this framework, each layer would use the services offered by lower layers, and each lower layer would assure the level of trust that it can provide to an upper layer. The layers would most likely form an hourglass-shaped structure, where upper layers would provide services, middle layers would be concerned with the rights expression and interpretation, and the lower layers would ensure the enforcement of rights. The layers dealing with rights expression and interpretation would form the notch of the hourglass-shaped structure. To assure trust and to ensure correct functioning of the layers, the security requirements of each layer would need to be defined. It is our belief that much can be learned in building such a structure by studying the highly successful framework provided by the OSI layers in telecommunications. Thus, in Section 2 we study the design of the OSI layers and the requirements of computer networks that led to the development of TCP/IP. In Section 3 we discuss the current requirements of DRM. Section 4 provides a detailed explanation of the layers of a "strawman" DRM framework. Next, Section 5 describes how the layered DRM structure would help achieve certain DRM requirements and considers the important question of where standards should be defined within these layers. Finally, in Section 6 we provide some useful concluding remarks concerning a layered DRM architecture, and how any standards defined within these layers must allow for interoperability at some level, while at the same time providing enough "room" for vendors to leverage their proprietary technologies.

2. LAYERED SYSTEMS

Large systems are often divided into parts in order to reduce complexity, and layering is a common technique for accomplishing this division. In layering, the entire system is partitioned into layers, and the issue typically becomes one of determining the services that each layer should provide to the layer above it. The functioning of all the layers together to achieve the purpose of the system then becomes independent of the way each layer is actually implemented. The actual implementation and algorithms used in a particular layer are hidden from the other layers. Furthermore, each layer has a well-defined interface to communicate with the layers above and below it. This type of well-defined interface is particularly important as it allows different vendors to develop technologies that address different parts of the system. Also, with such layering it is possible for products from different vendors to interoperate. New services can be introduced at any time when such well-defined interfaces are available, and each layer typically has a protocol or handshaking method that allows it to interact with peer layers on other machines.

2.1 OSI Layers

The OSI layers can be broadly divided into three categories, the upper, middle and lower layers. The upper layers consist of the Application, Presentation and Session layers. These layers involve interaction of peer processes across a network. Different kinds of needs with varying specifications can be built using these layers. The services which constitute the upper layers have well-defined protocols, and have been placed at this level due to the end-to-end arguments that have served as a central set of guiding principles in the design of the Internet. Specifically, end-to-end arguments consider how application requirements should be met within a system. When applied to computer networks, these arguments suggest that application-level functions, e.g., encryption, checksums, etc., should not be built into the core of the network; rather, they should be placed at the endpoints. By doing so, the complexity of the core is reduced, which reduces the cost of the core and facilitates future network upgrades. Furthermore, the generality of the network increases since new applications can be easily added without having to change the core. Similarly, the reliability of the network increases by simplifying the core. Thus, due to end-to-end arguments, functions at the core of the Internet (routers that forward packets) are simple, and the bulk of the functionality appears in the computers that sit on the edge of the network. The upper layers of the OSI model are concerned with implementing this functionality [25]. Specifically, the Application and Presentation Layers provide services that are frequently used by applications that involve communication. These layers are generally built without any concern for what underlying networking technology is being used. The middle layers act as a buffer isolating these services from the intricacies of the underlying networks.

The middle layers of the OSI model are the most crucial in terms of communications as they constitute the Transport and Network Layers. The Transport Layer is also considered an end-to-end layer, whereas the Network Layer is not—it is referred to as a chain layer [30]. The Transport Layer ensures end-to-end delivery of messages, and therefore the protocols in this layer are concerned with end-to-end machines, not with neighboring machines. The Transport Layer uses the services offered by the underlying networks or internetworks to provide the upper layers with the transfer of messages that meet a certain quality of service (QoS). The Network Layer on the other hand plays a true role in interconnection among heterogeneous machines. It provides for the transfer of data in the form of packets across a communication network. The Network Layer acts as a true buffer for the end-to-end layers. Thus, it is these middle layers that act as the "glue" allowing all of the layers to work together. For this reason, there are many standards defined in these layers, and thus these layers will be discussed in more detail shortly.

The lower layers constitute the underlying networking technologies; that is, the hardware technologies. Each of these technologies has its own set of protocols, and each may have a different transmission media such as copper wire, optical fiber, or unguided media such as lasers and radio waves. Many technologies with varying needs are present today, and no one particular technology has become a standard. With advances in science this block of layers is bound to undergo significant changes as new networking technologies with innovative transmission media and protocols will replace existing ones. In the following section we look at the requirements and problems faced by computer networks and how TCP/IP evolved as a solution in order to address these problems.

2.2 TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is the basic communication language of the Internet, and basically constitutes the middle layer of the OSI model. TCP/IP is composed of two parts. The higher-level part, TCP, manages the breaking of a message or file into smaller packets that are transmitted over the Internet and received by a TCP process on another machine that can reassemble the packets back into the original message. The lower-level part, IP, handles the addressing which allows the packets to be routed to the proper destination. This allows packets from the same message to be routed through different machines and still end up at the same destination. The original objective of TCP/IP was to transfer packets across three different networks, ARPANET, packet switching networks and packet radio networks. Since this work was related to military applications, another important requirement was that the communication be robust. Thus, the basic aim was to develop a set of protocols that are highly effective in enabling communications among many different types of computers systems and networks. Figure 1 shows how the TCP/IP protocol forms an hourglass-shaped structure. Note that IP, which provides the minimal service, is situated at the notch of the hourglass. The lower dome comprises different networking technologies. The IP layer is actually an abstract layer, which shields the intricacies of the underlying networks from the upper layers. It also provides a lowest common denominator upon which other services can be built.

One of the major requirements of computer communications is to provide different types of services at the transport service level. For example, in some applications reliable data transfer is most important, while in others, real-time data transfer is more important. This was one of the main reasons for the separation of TCP and IP [6]. Different types of services have different requirements dealing with speed, latency and reliability. More importantly it was required that existing networks be used. Hence an abstract layer using a datagram as the building block was introduced in order to shield the intricacies of the underlying networks from applications built on top of IP. This layer assured what can be called a minimalist service, and therefore it was reasonable to introduce standards at this level. Since the service was minimal, very basic standards were needed for its functioning, one of them being IP addresses. Thus, IP provides a lowest common denominator (best effort data transfer) upon which other services can be built. Furthermore, the role of IP is to ensure service irrespective of the underlying network.

The aforementioned properties, as we have already mentioned, give the TCP/IP protocol suite what can be called as an hourglass shape, with IP at the notch. The key protocol in this suite is the IP protocol. The mantra behind the TCP/IP protocol suite is "IP over everything and everything over IP". Though IP defines the basic requirement in computer networks, IP by itself is incomplete. For IP to function it must have supporting protocols—protocols such as TCP, UDP, ARP, RARP, HTTP, FTP support IP to create the various services on top of this common denominator. These protocols are used to either create a service with a new functionality or to enhance the performance of a given service. Furthermore, IP is able to withstand the constant changes in telecommunications technology due to its minimal definition. Such a minimal definition gives a great degree of flexibility to create new services by defining new supporting protocols.

3. DRM REQUIREMENTS

As telecommunications networks are about data transfer, DRM is about the management of digital rights. Most digital content available today is managed by at best rudimentary DRM systems, if DRM is deployed at all. The issue of what services should be provided by a DRM system was considered by the MPEG-21 Committee [20]. Below we provide a list of DRM requirements that was influenced by this work:

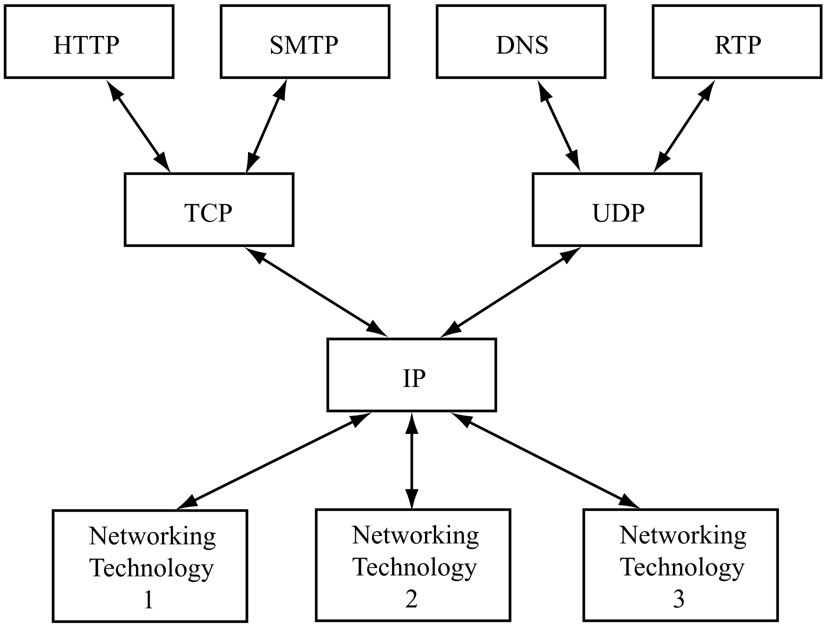

Figure 1: The hourglass shape of communications via TCP/IP. Note that IP, which provides the minimal service, is located at the notch of the hourglass.

1. No DRM system today has emerged as the de-facto standard, and what is needed are systems that work universally.

2. There is a need for a DRM system that can associate the rights attached to content with some entity and manage those rights across different machines and external devices.

3. No framework exists for allowing vendors to develop systems focusing on different aspect of DRM that could be made to interoperate. Such a framework is necessary.

4. There is a lack of standardized methods for monitoring and detecting infringement of rights. These must exist if any DRM system is to be practical.

5. DRM systems must support all widely-used digital content formats. It is important, on the other hand, that the system be independent of specific formats, thereby allowing the system to withstand changes in content formats as well as content access/rendering software.

6. There is need for a technology that supports the implementation of end-to-end supply chains. This technology would facilitate fast distribution of digital content items while protecting the rights associated with them.

7. A choice of service is necessary. There are a range of services required in DRM applications today—some of these are very complex involving superdistribution and are license based. Enforcing these rights requires strong rights enforcement techniques. On the other hand there is a need for very simple DRM systems whose rights can be enforced using simple rights enforcement techniques. Thus, DRM systems should be capable of providing different types of services according to these differing requirements.

8. There are various platforms available and various external devices are used to render content. If DRM systems are to be useful, they need to support all of these devices and platforms.

It will be difficult to build a DRM system that can simultaneously satisfy all of these requirements. Even if such a system were built, it would soon have to be changed to meet changing needs and technology advances. For example, the rendering platforms over which DRM systems will be built will change, and new external devices will be developed that support evolving media formats. Furthermore, the DRM requirements themselves are not static, they are likely to evolve with the passage of time in conjunction with changes in technology. For example, the increase in broadband penetration will likely lead to the development of new services. These are precisely the types of problems that layered systems have proved useful in mitigating. Specifically, properly specified layered systems act as a buffer against rapid changes in technology. Thus, it appears that a layered DRM framework, which identifies and divides the different aspects of DRM, makes sense.

To develop such a layered framework, we must identify the parts that would likely undergo change along with the parts that probably would not change. The parts that are likely to change include the services, external devices, platforms, and networking technologies, to name a few, while the parts that are not likely to change include those that address the basic requirements of a DRM system. That is, the latter parts would provide the minimalist DRM services, and would sit at the core of the framework. The goal is to encapsulate these parts within layers, and to provide well-defined interfaces between the layers. This would allow the internal workings of the different parts of a DRM system to change without destroying the overall functionality of a DRM system. Furthermore, this partitioning would allow vendors to separately and independently develop technologies and services that address different parts of a DRM system. New sublayers might evolve as the requirements change within a layer, but the overall framework itself should be flexible enough to withstand changing requirements and technologies. In the next section we present a proposal for such a DRM architecture, acknowledging the strong influence provided by the previously developed and highly successful OSI model.

4. DRM IN LAYERS

Consider the client-server model associated with a typical DRM system, where the user of a client machine wishes to purchase a piece of content, and the server-side is responsible for delivering the content along with its associated rights. There are three basic DRM processes that must be supported within this framework. At the highest level content is used by an application according to the rights associated with it. An intermediate level is concerned with how the rights are specified, as well as how they should be interpreted in particular environments. At the lowest level the concern is how the rights will be enforced on the client side. Thus, once an application on the client side is able to understand the rights associated with a piece of content, it has the responsibility of enforcing those rights. From the server's perspective, it should not matter how this is accomplished. That is, the server-side only needs assurances that the client is able to enforce the rights, not how this will be accomplished. The key to allowing this is the rights expression and interpretation process. If this process could be standardized, it would form the glue that holds the other two processes together, similar to IP in telecommunications networks. Thus, let us now draw some parallels between telecommunications networks and DRM systems. Figure 2 shows how a DRM system could be structured in order to form an hourglass shape. Rights expression and interpretation (REI), which provides the minimal services in DRM is situated at the notch of the hourglass. The lower dome consists of the various rights enforcement technologies available, and the upper dome consists of applications that need to make use of content according to the rights associated it. That is, they deal with the end-to-end arguments of DRM. The REI layer shields the intricacies of digital rights enforcement from the service layers above. It provides a lowest common denominator over which new services would be created according to needs of the DRM environment, leading to specific applications in the upper layer of the DRM architecture. Such a framework would be flexible to the changing demands of DRM going forward.

It is interesting to consider the InterTrust Digibox from this layered perspective. The Digibox allowed a content publisher to encrypt content, define rules for content usage, and publish this encrypted content on a web site—the encrypted content and rights management rules where combined into a single object, the Digibox. Intellectual property transactions (e.g., payments to certain parties) were handled by MetaTrust "clearing houses". Thus, the Digibox, along with MetaTrust facility, was a complete DRM system on its own. The MetaTrust facility played the role of the upper layers, while the Digibox played the role of the middle and lower layers.

Let us now consider a layered DRM architecture in more detail. Consider first the minimal requirements of any DRM system. Obviously, DRM deals with rights management, where rights management involves the association of rights to digital content intended for particular uses and users. Once digital content with its associated rights is provided to a user, the user should be able to interpret those rights and use the content while at the same time respecting the rights. Thus, in an idealized sense, DRM is first and foremost about rights association and interpretation, and therefore the ability to express and interpret rights should constitute the minimal required services in any DRM environment. All other aspects of DRM such as rights enforcement, trading, superdistribution, etc. can make use of these minimal services in order to implement a specific DRM task. As an analogy, the role of trading and superdistribution in a DRM system is the same as that of the IP supporting protocols (e.g., TCP, UDP, HTTP, SMTP, etc.) in a communications system—they are creating new services out of the basic available services.

It is also important to note that the requirements of communications systems have changed over time, and new networking technologies have been introduced in order to deal with these changing requirements. Thus, to continue the analogy, the role of rights enforcement technologies such as secure containers, encryption, watermarking, and trusted platforms is the same as that of networking technologies such as the ethernet, packet radio networks, etc. These rights enforcement techniques are bound to change with advances in mathematics and science, though their role remains the same. Thus if the core of the framework is limited to rights expression and interpretation, then the framework will be flexible enough to change with the changing requirements of DRM environments, without invalidating useful applications that have been previously built on top of the minimal DRM services.

Given what we have just discussed, it makes sense to have a standard for associating rights with digital content, and to encapsulate this within a layer in our layered DRM architecture. Such a standard should include all forms of digital content irrespective of format. This particular layer should have the ability to associate any reasonable right(s) to any type of digital content. A logical means of accomplishing this would be to agree upon a standard DRM language for the expression of rights, i.e., a standard rights expression language (REL). Then, once rights are associated with a particular content item, a standard REL would ensure that peer entities on other machines could interpret these rights. It is important to note that rights are unique to not only content but also to users, and this must be accounted for in any standard REL. The functionality we have just described is provided in the central REI block shown in Figure 2.

Above the REI layer is where the supporting protocols and applications would be built. These services would include those that are determined by the end-to-end arguments necessary in DRM system design. Below the REI layer are the technologies that enforce rights. Similar to the underlying networks that serve the purpose of actually transferring data in computer communications, the rights enforcement technologies attempt to enforce the specific rights associated with particular content items and users. A more detailed breakdown of the OSI data communications layers is provided in Figure 3. In this figure we also provide a more

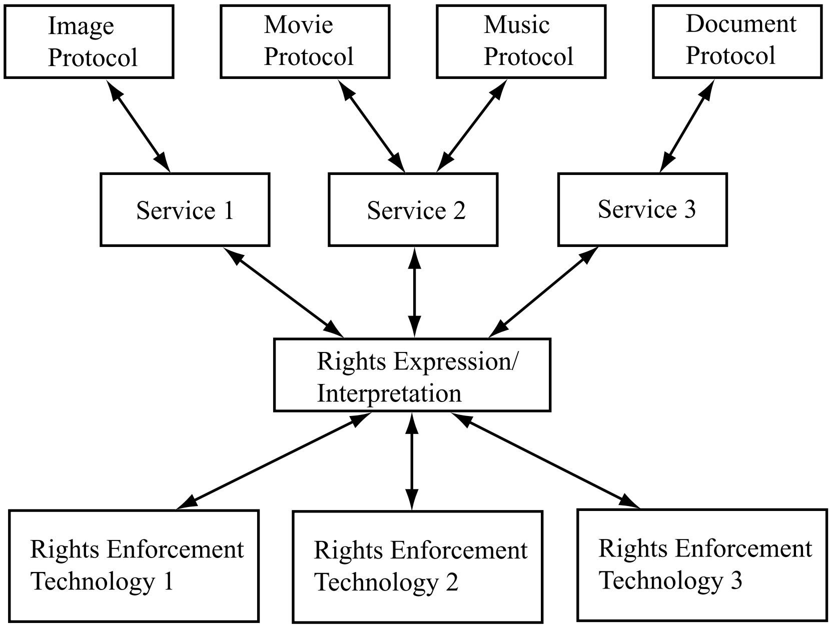
Image Protocol
Movie Protocol
Music Protocol
Document Protocol
Service 1
Service 2
Service 3
Rights Expression/ Interpretation
Rights Enforcement Technology 1
Rights Enforcement Technology 2
Rights Enforcement Technology 3

Figure 2: DRM as a layered system with an hourglass structure. In this case Rights Expression and Interpretation provides the minimal service, and this service is therefore located at the notch of the hourglass.

detailed breakdown of possible DRM layers. In the following sections we address some of the specific functionalities associated with these layers.

4.1 Upper Layers

The upper layers of the DRM architecture shown in Figure 3 essentially create services that are frequently used by applications that involve DRM. Thus, these layers are end-to-end layers that would deal with the end-to-end arguments of a DRM system. The arguments might include deciding upon the type (i.e., resolution, format, etc.) of data that will be delivered, payment transactions, identification and authentication, security, etc. The upper layers would be divided into at least two parts. An Application Layer would create services for the applications that will use the content according to the rights, and a Negotiations Layer would create services out of the middle layer in order to provide a choice of services to the Application Layer.

In data communications there are different ways to transfer data (e.g., web pages, DNS queries, file transfer, real time data transfer, etc.), and hence there are different protocols at the Application layer (e.g., HTTP, DNS, FTP, RTP, etc.) to satisfy the varying needs of different applications that use the TCP/IP protocol structure. Similarly in DRM we have different types of digital content such as music, videos, movies, research documents, corporate documents, etc. The manner in which this content is rendered, distributed, traded, and the way rights associated with them are managed is different for each of type of content, and may differ from application to application as well. In the DRM framework, the Application Layer has separate protocols for each content type, and there may be yet additional protocols in order to support how the content will be used. For example, some of the functions that these protocols would deal with include trading, superdistribution, user identification, and tracking of data.

The Application Layer uses the services provided by the Negotiations layer. In data communications, TCP and UDP are created out of the basic IP protocol to provide a choice of services at the Transport Layer to the Application Layer protocols. The Negotiations Layer plays the same role in DRM systems, it provides a choice of services to the protocols of the Application Layer. For instance, at a minimum we envision protocols dealing with data assurance, data security, and data views. Let us consider each of these.

Once the rights are decided upon and the identity of the user is verified, the Negotiations Layer would need assurances from its peer layer on the client side. The assurance might be that the client software is capable of enforcing the rights associated with the digital content. If the client software is unable to enforce the rights associated with the content, the Application Layer would be informed about such a situation and negotiations may proceed towards settling on a lower-valued rendering of the content. The assurances themselves might be managed by using digital certificates. In this case rights enforcement technology would be built to work on certain hardware and operating systems, and would be registered with some certifying authority. The Negotiations Layer would demand such a certificate to verify the authenticity of the rights enforcement software. Once verified, the server-side entity has some assurances about how the rights will be enforced on the client machine. It is important to note here that the trust in the client-side entity is established through a certifying authority and that the complete process be transparent to the user.

Security protocols would decide on the type of encryption, watermarking, and other security technologies that should be used. There is a need for standards in these protocols. For example, encryption methods and algorithms must be

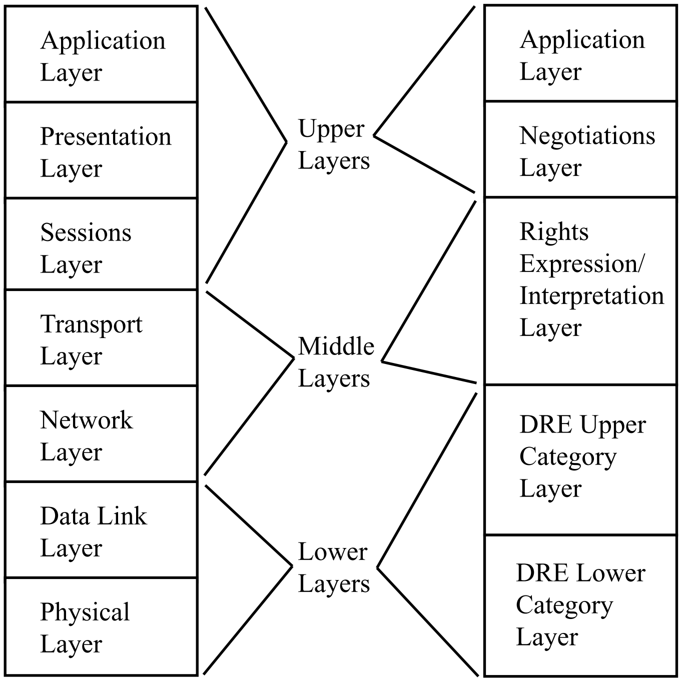

Figure 3: A breakdown of the OSI data communications layers and the proposed DRM layers into upper, middle, and lower layers.

agreed upon since content is encrypted on the server side and used on the client side. Some applications may require that a watermark be inserted in the content before it is delivered to the client. It is the Negotiations Layer that should decide on the type of watermark that would be used.

The data view protocol is concerned with the way the content should be rendered. As there are different content types, and different types of software for viewing content, this protocol ensures that only authenticated software would be allowed to read and present the data to the user. The goal would be to prevent "illegal" software from gaining access to cleartext (i.e., decrypted) content on the client side.

It is important to point out that all of these services may not be required by all applications. For instance, some applications may not require watermarking at all, others may require a very strong security policy, while yet others may settle for a weaker encryption algorithm. Thus, the Negotiations Layer provides a choice of services to satisfy the varying needs of different applications.

It is also important to recognize that rights are associated not only with content, but also with users. Therefore, rights should be preserved for a user across different machines. These upper layer protocols would be involved in handling this issue. Once the negotiations in this layer are completed, it is the job of the middle layers to create licenses, associate rights with the content and encrypt the combination.

4.2 Middle Layers

These layers encapsulate the minimal requirements associated with the management of digital rights. Once the content is identified and it can be determined what rights are required, the rights can be associated with content for a particular user. As discussed earlier, this layer would form the notch of the hourglass-shaped structure of the system; that is, it is the intermediary between the upper and lower layers. Thus, it is necessary that certain aspects of the middle layers be standardized. In particular, there must be a common language for expressing digital rights, and the way in which rights are associated with content may also require standardization. Without these standards the universality of the DRM process would be compromised.

The universal functioning of DRM is essential to its success, and this is only possible when there is a standard way of expressing rights. Thus, there is a need for either standardizing on an existing REL, or developing a new one. Of course, given the central role that this piece plays, the consistency and completeness of any such language should be considered in detail, preferably by a standards committee. Given that the middle layers serve the purpose of buffering the process of rights enforcement from DRM applications, given any type of digital content, a standard REL must be capable of associating rights with that content. Furthermore, peer layers on different machines must be able to interpret these rights accordingly. In addition, it is envisioned that these layers would be the ones responsible for encrypting content using the algorithms determined by the Negotiations Layer. Once delivered, the peer middle layers on other machines are responsible for decrypting the content package. In particular, if a content package is delivered encrypted, then the client machine must contract with decryption services in order to decrypt the content, and the rights enforcement technologies at the lower layers must coordinate this so that the decryption stream can be protected.

Most of the existing RELs are markup languages. They do not enforce or mandate any policies for DRM, but provide the mechanisms to express them. Examples include XrML, ODRML, and XMCL [8, 15, 31]. Each of these machine-readable languages has their own architecture and syntax, the semantics of which should be sufficient to express all realistic and conceivable rights expressions. Thus, this is an area where additional formal analyses on the capabilities of RELs is required [13, 23].

4.3 Lower Layers

The lower layers of the DRM framework shown in Figure 3 are concerned with rights enforcement, and can be divided into the Upper Category Layer and the Lower Category Layer (each of which may contain sublayers). The Upper Category Layer is responsible for the handling of content according to its type, while the Lower Category Layer handles content irrespective of its type. More specifically, the Upper Category Layer is responsible for acknowledging the format of the content, granting access to editors and players, and in general for presenting the rendered content to the user. The Lower Category Layer is responsible for preventing unauthorized low-level access to DRM programs and rendered content. Let us consider the responsibilities of each of these layers in turn.

In addition to using players/editors to render content, the Upper Category Layer must also interact with the middle layers for encryption/decryption purposes. Furthermore, this layer is responsible for reporting every time content is accessed, and for keeping track of how the data is used. This layer might also be responsible for destroying the data (or rendering it useless) under the condition that the user no longer has rights to access the content. This tracking of content usage must be done so that it preserves the rights of users across different machines. Well-defined algorithms and protocols can be developed to achieve this purpose. For example, security architectures have been proposed that allow one to preserve the rights of users for a content item across different devices. This is accomplished by allowing devices to establish dynamic groups, called authorized domains, where legally acquired copyrighted content can be used across the devices registered within the authorized domain [22]

As we have mentioned, the Lower Category Layer protocols would not acknowledge the format of digital data. The responsibility of this layer is to prevent any low-level "illegal" access to DRM programs, and as such this layer must work closely with the underlying operating system and hardware. For example, this layer would be responsible for authenticating device drivers associated with rendering on the client machine. The goal is to monitor malicious programs and prevent any illegal memory accesses. A well-known application that fits in the Lower Category Layer is the notion of secure containers. Within a secure container, content will only be decrypted on a client machine if certain security conditions can be met. Furthermore, during the time the content is rendered for viewing, the client machine is prevented from saving or otherwise copying the content. Microsoft's Palladium architecture is an example of an emerging technology that would fit into the Lower Category Layer. Palladium (now referred to as the Next-Generation Secure Computing Base for Windows) provides a trusted computing environment that is intended to be resistant to malicious codes and software attacks that include illegal memory accesses. This technology is designed to work side-by-side with the existing functionality of Windows [4].

Another important responsibility of the Lower Category Layer involves interaction with devices, e.g., when digital content items such as MP3s are transferred to external devices. It is important for users to be able to use digital content across multiple devices. To manage the rights across these devices, the Lower Category Layer will require a well-defined protocol with the devices regarding the use of data. Thus, the devices themselves must be capable of supporting the protocols.

Both the Upper and Lower Category Layers must work closely with operating systems, hardware platforms, and external devices. If the interfaces of these layers with the middle layers and external devices are well defined, then it will be possible for operating system vendors, hardware manufacturers and device manufacturers to work independently to achieve the goal of rights enforcement. That is, with the proper protocols, the way that rights enforcement takes place should be independent of the systems built above them. The particular rights enforcement technology deployed may depend upon the type and value of the content, the trust associated with a particular user or group of users, and the particular rights that need to be enforced. The available devices, hardware support, and operating systems support all influence the efficacy of these rights enforcement technologies. Thus, just as QoS is an important measure associated with a particular piece of networking hardware, the quality of rights enforcement should be a measure associated with every rights enforcement technology. This measure is related to but somewhat different from a QoS measure, as it will change according to the attacks that have been applied to a rights enforcement technology. It is also worth mentioning that this is an area where DRM has seen the most "snake oil", divergent opinions, and confusion in general. Thus, let us consider these technologies in more detail.

The extremes regarding rights enforcement technologies range from some reckless vendor claims of 100% copy protection security [3], to the argument that these technologies are useless since all of them can be circumvented [27]. We take an intermediate view. Certainly all realistic rights protection technologies can be circumvented, but the same claim can be made for all realistic encryption technologies as well, yet we all rely on encryption when banking on line, when using a VPN, etc. Thus, like encryption, the real questions regarding rights enforcement should be what levels of resources are needed to defeat various types of protection, and what types of business gain can be built upon these levels. For example, if the cost of protected content is appropriately priced relative to the cost of hardware copying devices, then the potential pirate will be inclined to purchase the content rather than steal it. This is particularly true if protection technology can be created such that its continual development allows particular attacks to be made obsolete. This is precisely the situation that has developed within the cable television industry—each new round of cable equipment upgrade is meant, among other things, to thwart the current set of attacks, forcing hackers to develop new sets of attacks. Increased sales and revenue have consistently followed these upgrades, indicating that this strategy is effective at reducing illegal copying [11]. Furthermore, consider that rights enforcement technology may be more effective if it is delivered according to a previously negotiated trust arrangement between the client and server entities. For example, it is possible to place invisible watermarks for tracking purposes in content if the identity of the user is known prior to delivering the content. In addition, the idea of a "trust history" similar that of a credit history can be developed, and content may be delivered according to a user's established trust record.

5. ADVANTAGES OF LAYERS

We have seen how the processes associated with DRM can be structured in the form of layers, and how responsibilities can be distributed among the layers to achieve specific DRM goals. Let us now consider in more detail the advantages of such a layered DRM system, referring back to the DRM requirements discussed in Section 3.

A layered DRM framework separates rights enforcement technologies from DRM services, and therefore allows for separate and independent development to take place in these two areas. The ability to do this would aid in the disciplined growth of the DRM field. Specifically, agreement upon well-defined interfaces and roles for each of the DRM layers will make it possible for DRM vendors to develop products that interoperate. Another critical requirement of DRM is for users to have easy interaction with digital content. In fact, this is a well-understood lesson from first generation DRM systems—if DRM gets in the way of using digital content, users will be reluctant to use it, and in fact they will resist it. The layers we have described do not specifically address interaction with digital content, this job is left to the developers of editors and players. However, a layered architecture has proved useful in addressing this very concern in communication systems. Specifically, with a layered architecture, the higher-level applications that users use to directly interact with content can change drastically, as long as they interface properly to the layers below. In this way, the proper functioning of the application within a DRM architecture is maintained. These matters address the first and third DRM requirements.

The second DRM requirement concerns the importance of perserving the rights associated with a piece content and a user across different machines and external devices. Protocols can be developed in the lower layers to allow the rendering of content on different devices and computers. A layered system would help achieve this without disturbing the upper layers. Also the REI would buffer the DRM services from the lower layers. Thus it would be possible to develop services over various platforms and external devices. Therefore, this also addresses the eighth requirement which is concerned with providing support for different platforms and external devices.

The forth requirement stresses that any successful DRM system will require a method for detecting rights infringement. Within a layered DRM framework, this particular functionality can be implemented in the Negotiations Layer, and the Application Layer could be informed of any such violations, along with the entity involved in this act. This would allow an application to keep track of "blacklisted" entities on a server.

The REI in the middle layer associates rights with data and is independent of the format of the content. It is important that the REI be independent as it forms the core of the system. A DRM system can be made to support different data formats by making use of the upper layers. The protocols by themselves would be independent of any content formats, thereby fulfilling the fifth DRM requirement.

Layers can provide a choice of services. It is not necessary to achieve every aspect of DRM in a particular system. In fact, we have previously made the case that it is useful to distinguish between heavyweight and lightweight DRM applications [12]. Heavyweight DRM tends to be license based, involving the use of complex rights such as sharing content with other users, the loaning of content, etc., whereas lightweight DRM includes simple rights such as the right to view, the right to print, etc. Furthermore, some DRM applications will require a high level of security, while others can settle for a weaker security. These all relate to the seventh requirement of a DRM system. In a layered DRM system, the Negotiations Layer can achieve the goal of providing a choice of services to DRM applications. Once provided with this service, protocols in the Application Layer can then be developed to create end-to-end supply chains supporting business-to-consumer and business-to-business channels.

One reason for the failure of some previous DRM systems was that they tried to address every aspect of the DRM requirements. Such systems could not stand on their own as DRM systems within an Internet-based environment. What is needed is an environment wherein different vendors can work on different aspects of DRM systems, which can then be made to work together within a given framework. The REI layer acts as a glue to make the different parts work together and also to achieve universality in DRM applications. A systematic evolution of standards and protocols can help to achieve this important DRM requirement, so let us now consider where they should be defined.

Where standards play. The development of a successful layered DRM system would depend upon how standards are defined in such a system. The definition of the standards in turn would depend upon the goals that such a DRM system should meet. As we see it, the primary goal of the next generation of DRM systems should be to make use of the existing infrastructure (and in particular the Internet) to create a DRM framework which would ensure global functioning. This leads to the fundamental structure of the layered DRM framework: A standard global facility for rights expression and interpretation. This can achieved by adopting a standard REL. This is vital, as rights expression and interpretation constitutes the minimal requirement of any DRM system. Thus the REI in the middle layer would need to be completely standardized, as it forms the core of the entire system. The processes that would act upon the digital content in this layer would also have to be standardized. These processes include the way in which rights are associated with content and the way rights are interpreted.

The next area in which standards should be considered includes that protocols that allow the various layers to communicate with one another. That is, a consistent set of relatively simple rules that define the way these layers communicate must be provided. This would allow supporting protocols to be defined, along with the creation of complex services out of the basic available services.

Because the Negotiations Layer provides a choice of services to the Application Layer protocols, standards should be defined in this layer. These standards should encompass such things as encryption algorithms and processes, negotiation protocols, and watermarking methods. The standards could thus be used to guarantee a standard set of services. In this way the Negotiations Layer would have a set of services and protocols to satisfy the different needs of various DRM applications. The standards associated with these services would change with the development of new methods and advances in mathematics. This is similar to what we have seen in the telecommunications industry.

In the Application Layer we could consider the development of standard protocols that would support superdistribution and trading structures for various content types. It is likely that these superdistribution and trading patterns would undergo frequent changes. Thus, the standards in this layer would most likely be the shortest-lived, and frequent introduction of new standards and changes in existing standards would be a feature of this layer.

Most parts of the lower layers should be relatively free of standards. How individual machines manage to enforce the rights is something that should be encapsulated within those layers. What is needed though is an assurance from the lower layers that they are indeed capable of protecting the rights, as such, some type of assurance protocol would need to be developed. In addition, the interfaces of the lower layers with external devices would need to be standardized. That is, there should be a well-defined protocol regarding the way content is used across devices.

The interests of the parties involved in using a particular system often determine how standards evolve [2]. Thus, the specific and detailed goals of a DRM system must be indentified by taking into consideration the interests of the various parties that play a crucial role in the DRM process. The main parties involved in the DRM process are the content producers, the content distributors and the content consumers. These parties are sure to have conflicting goals that must be reconciled. Once these goals are identified, it is necessary to prioritize them. It is important to note here that the order of importance of goals can greatly influence the way in which DRM standards are shaped [6].

6. CONCLUSION

In this paper we have proposed the idea that the adoption of a layered framework may facilitate the development of DRM systems. In order to support this position, we began by noting that the current landscape of DRM technologies is somewhat reminiscent of the telecommunications industry prior to the widespread adoption of networking standards. The subsequent growth of the Internet, with concomitant growth in equipment sales and services offered, was expedited by the broad acceptance of the basic OSI layered architecture. At the core of this architecture is IP, the basic protocol which serves as the glue for holding together the disparate networks of the Internet. Similarly, we have proposed a basic layered architecture for DRM, where rights expression and interpretation (REI) serves as the analog of IP in the DRM layers. That is, REI provides the minimal service that is required by any DRM system to support global functionality. In this sense, REI may serve as the glue allowing different DRM technologies to interoperate. We have discussed how different services can be built on top of the REI layer, and how the REI layer can serve as a common denominator over disparate rights enforcement technologies. Furthermore, such a layered DRM architecture is quite synergistic with the predominant layered networking architecture. Thus, interleaving of these architectures seems natural. This is, of course, essential, as the Internet has and will continue to be one of the most important technologies underlying DRM.

The success of a layered DRM architecture requires the systematic development of standards and protocols that support the needs of applications that reside at the various layers within the architecture. A well-defined layered DRM architecture will facilitate the separate and independent development of DRM technologies by various vendors that could be pieced together in order to address specific needs. Therefore it is vital that the architecture provide enough space and flexibility for different developers to create their own technologies and services, thereby minimizing the chances that one vendor can monopolize much of the DRM market.

7. REFERENCES

[1] J. Alvear. Realnetworks, Microsoft face off on DRM. *streamingmedia.com*, June 22, 2001.

[2] H. Alverstrand. The role of the standards process in shaping the internet. *Proceeding of the IEEE*, 92(9):1371–1374, 2004.

[3] ArtistScope. `http://www.artistscope.com/copysafe/about.html`.

[4] A. Carroll, M. Juarez, J. Polk, and T. Leininger. Microsoft palladium: A business overview. Technical report, Microsoft Content Security Business Unit, Aug. 2002.

[5] V. G. Cerf. On the evolution of internet technologies. *Proceeding of the IEEE*, 92(9):1360–1370, 2004.

[6] D. D. Clark. The design philosophy of the DARPA internet protocols. In *ACM SIGCOMM*, pages 106–114, Stanford, CA, Aug. 1988.

[7] T. Clark. IBM closes cryptolopes unit, 2002. `http://news.com.com/2100-1001-206465.html?legacy=cnet`.

[8] Contentguard. *XrML 2.0 Technical Overview*, March 2002.

[9] I. J. Cox, M. L. Miller, and J. A. Bloom. *Digital Watermarking*. Morgan Kaufmann, San Francisco, CA, 2002.

[10] The DRM vendor graveyard. `http://www.info-mech.com/drm_vendor_graveyard.html`.

[11] J. Feigenbaum, D. Boneh, and R. Venkatesan. Report on the DIMACS workshop on management of digital intellectual property. Technical report, DIMACS, Apr. 2000. `http://dimacs.rutgers.edu/Workshops/Management2/`.

[12] H. Goldstein, G. L. Heileman, M. D. Heileman, T. Nicolakis, C. E. Pizano, B. Prumo, and M. Webb. Protecting digital archives at the greek orthodox archdiocese of america. In *Proceedings of the Third ACM Workshop on Digital Rights Management*, pages 13–26, Washington, DC, Oct. 2003.

[13] J. Y. Halpern and V. Weissman. A formal foundation for XrML licenses. In *Proceedings of the 17th IEEE Computer Security Foundations Workshop*, pages 251–265, Asilomar, CA, June 2004.

[14] G. L. Heileman and C. E. Pizano. An overview of digital rights enforcement and the MediaRights technology. Technical report, Elisar Software Corporation, Apr. 2001.

[15] R. Iannella. Open digital rights language (ODRL), Version 0.7. http://odrl.net/ODRL-07.pdf, Oct. 2000.

[16] M. A. Kaplan. IBM cryptolopes, superdistribution and digital rights management, 1996. `http://researchweb.watson.ibm.com/people/k/kaplan/cryptolope-docs/crypap.html`.

[17] U. Kohl, J. Lotspiech, and M. A. Kaplan. Safeguarding digital library contents and users: Protecting documents rather than channels. *D-Lib Magazine*, Sept. 1997.

[18] M. Krochmal. Alchemedia protects pictures with plug-ins. TechWeb News, Oct. 28, 1999. `http://www.techweb.com/wire/story/TWB19991028S0012`.

[19] M. McKenzie. Copyright protection: Understanding your options. Technical Report 4, The Seybold Report on Internet Publishing, Dec. 1996.

[20] MPEG-21 Committee. Information technology — multimedia framework (MPEG-21) — part 1: Vision, technologies and strategy. Working document: ISO/IEC JTC1/SC29/WG11 N 4333, July 2001.

[21] M. L. Nelson, B. Argue, M. Efron, S. Denn, and M. Pattuelli. A survey of complex object technologies for digital libraries. Technical report, NASA TM-2001-211426, 2001.

[22] B. C. Popescu, F. L. A. J. Kamperman, B. Crispo, and A. S. Tanenbaum. A DRM security architecture for home networks. To appear in DRM'04, Washington, D.C., Oct. 2004.

[23] R. Pucella and V. Weissman. A logic for reasoning about digital rights. In *Proceedings of the 15th IEEE Computer Security Foundations Workshop*, pages 282–294, Nova Scotia, Canada, June 2002.

[24] B. Rosenblatt, B. Trippe, and S. Mooney. *Digital Rights Mangement: Business and Technology*. M&T Books, New York, NY, 2002.

[25] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Transactions in Computer Systems*, 2(4):277–288, Nov. 1984.

[26] F. B. Schneider, editor. *Trust in Cyberspace*. National Academy Press, Washington, D.C., 1999.

[27] B. Schneier. The fallacy of trusted client software (cryptorhythms column). *Information Security Magazine*, Aug. 2000.

[28] N. Shivakumar. *Detecting Digital Copyright Violations on the Internet*. PhD thesis, Stanford University, 1999.

[29] O. Sibert, D. Bernstein, and D. Van Wie. Securing the content, not the wire, for information commerce. Technical report, InterTrust Technologies Corp., 1996.

[30] A. S. Tanenbaum. *Computer Networks*. Prentice Hall, 4th edition, 2002.

[31] The eXtensible Media Commerce Language (XMCL). `http://www.xmcl.org`.