Secure distribution of digital goods
Extended Abstract

Vítor Manuel Pires Gaspar

Thesis to obtain master degree in
Information Systems and Computer Engineering

Jury
President: Prof. Luís Eduardo Teixeira Rodrigues
Tutor: Prof. José Manuel da Costa Alves Marques
Jurors: Prof. Carlos Nuno da Cruz Ribeiro
Eng.º José Afonso Vasconcelos Pires

September 2007
1 Introduction

The login mechanisms which allow users to authenticate themselves on websites or on systems based on security perimeter allow the system to protect itself against external attacks. However, authorized users can also be a risk. If these users redistribute the information which they have access to unauthorized users there is no way to control them and the information is in risk of being misused.

Companies and organizations need nowadays to store and distribute large amounts of information with different degrees of sensitivity. These organizations are faced with the challenge of protecting these goods of careless users or with bad intentions.

To protect information from these attacks, it’s proposed in this document the Digital Rights Management Systems (DRM) which protects sensitive information by managing and enforcing access and usage rights to the information throughout its lifecycle, no matter where the information is distributed. These mechanisms distinguish themselves from traditional access control because it’s easier to define and control the rights of each user, denying, for example the copy of a protected file while allowing read access.

1.1 Goals

The main goal of this work is to study digital goods protection technologies after they have been distributed, and the implementation of a system using the acquired knowledge. It’s desired that the actions that are performed on the distributed documents are restricted by the rights defined for each user. In particular, it’s studied how these mechanisms can be applied to web applications, because they are the most common applications to distribute digital goods in many formats. After that we’ll study where these systems can be useful, giving some use cases.

It will be studied, among other use cases, a Link’s content management system (e-Portal) which allows publishing content on-line. Link wants to protect the content that this CMS publish, selectively, against unauthorized copy and redistribution.

It’s desired that the proposed solution can be used in any web application in an easy way, as long as it has compatible technology (.NET), so that it’s use can be considered in Link’s future projects.
2 Digital Rights Management Systems

Digital Rights Management (DRM) designates a group of measures used by publishers and other copyright holders to limit usage of digital media or devices, controlling use restrictions of specific instances of a digital work. These systems allow the creation protected digital content, specifying a group of rights which can later be managed by the publishers. When a user attempts to access the protected content, he first has to obtain a licence created by the rights holder (or represented entity).

2.1 Architecture

Most DRM systems can be considered as a variation of the default architecture described next.

2.1.1 Rights model

The rights model is a rights specification of the content which the system manages and specifies the actions that an entity can do with or without them. It describes right’s types (what the users can do with the protected content) and their attributes (for how long they are valid, which users, etc.). A rights model correctly defined is the first step to the construction of a DRM system for an organization. It must be capable of determining which rights a particular user can have for a particular sensitive document, and for how long the rights are valid before they expire.

2.1.2 Conceptual architecture

Conceptually a digital rights management system is composed of the following components:

1. **Content server**: Stores the sensitive content and prepares it to be distributed, packaging it along with metadata and encrypting everything;

2. **License server**: Generates licences necessary for specifying rights, identifying the content, and the user. These licences will be necessary to decrypt the content;

3. **DRM client**: A component located in the user side which enforces the defined rights.

A user, to use the protected content, first obtains the packaged content from the content server using the traditional methods. After that, the user makes a request to exercise rights over the content using the interface of the protected application. The DRM client must be designed to be activated in response to such requests. After its activation, the DRM client identifies the user and asks the licence server to generate a licence for that user. If the request is legal, i.e. the user identity is valid and the user has rights to access the protected content, the licence server creates a licence containing the rights specification for the content, information about the identity of the user and the key to decrypt the content. The DRM client receives this data and reads it to determine if the operation requested by the user is valid.
2.2 DRM systems limitations

2.2.1 Security
The security strength of a DRM system is strongly tied with the extent to which the trusted DRM client can protect itself from being tampered within a potentially malicious environment, because in most cases the protected content is physically located at the client machine. For a motivated attacker who has been authorized to access the rights-protected information on a computer that is completely under his control, it is only a matter of time to break the protection mechanism (if there is any), bypass the rights enforcement checking and access the data. If a privileged user has been granted rights like exporting unencrypted content or modifying rights policies to a protected file, once she is authorized to access the file content, there is simply no way to control the content's distribution.

2.2.2 Analog copy
When the content is on the user's monitor, he can copy the information without authorization using analog copying mechanisms like screen captures or photos. This problem is inevitable, because when the content is converted to a human perceptible form it is a relatively simple matter to digitally recapture that analog reproduction in an unrestricted form, thereby circumventing the restrictions placed on copyrighted digitally-distributed work.

2.2.3 Portability
DRM systems in general work only with application and formats that the organization that developed the system chose to make compatible. For specific application it is necessary to create a plug-in to do it, if an API interface was conceived, or the user can be forced to change to a compatible application. In extreme cases, the applications can be so specific that there is no alternative. Besides, once the application is chosen it becomes hard to switch to another one, if it's compatible with the DRM. As a result, the installation of DRM systems can incur in severe changes in business processes inside the organization which can lead to performance degradation.
3 Concepts, Techniques and Methodologies

In this section is described the conceptual phases in the development of the web enabled DRM system.

3.1 Specification and architecture

A request to access a web page begins in the user’s browser which is treated by a web application on the server side. To protect the target document with a group of rights minimizing the changes in the involved applications it’s necessary to treat the document during transmission between the web server application and the browser. From this comes the idea of applying a filter in the communication between the web application and the user’s browser, protecting the information just before beginning the transmission, applying the defined rights. In the client side it must be present an interpreter of the rights, which enforces them.

The document to which the rights are applied must be protected through encryption and distributed across the web to the target browser. This browser must be capable of identifying the content as a DRM protected document and then request a rights controller to render the content in accordance to the defined rights. In the project it was used Microsoft’s platform – Windows Rights Management Services – which provides a server licence to regulate the access to protected documents. The rights controller to access the protected content must ask a licence to this server which validates if the user is authorized and in that case the server reveals the decryption key to the controller which will allow it to access the content. The licence is valid for a period of time defined during licence generation and thus allows off-line access, as long as the period of time doesn’t expire.

3.2 Windows Rights Management Services (RMS)

Windows Rights Management Services (RMS) is a Microsoft Windows technology that is a digital rights management system used for protecting documents such as corporate e-mail, Word documents, and web pages.

Compared with the DRM system architecture described previously, the main differences of RMS are in the following two aspects:

1. Windows RMS does not have to keep sensitive document files in a content repository. All the sensitive information can be distributed freely once created by trusted authors. There is no separate content server in Windows RMS systems. The content packaging process is done locally.

2. Rights policies are stored in a publishing license attached with the sensitive document and travel with the document together. The license server does not keep any rights information. This feature allows mobile users to access the rights-protected information off-line.
3.3 Comparison with default architecture

The content server on this work will be Internet Information Services (IIS) which will run the ASP.NET protected application, the source of the content, and the web content filter which will protect the web pages served by that application. In this process it’ll be generated a RMH (Rights Managed HTML), which is essentially an encrypted HTML file along with the rights description for each user.

The rights definitions will be stored in a database, as in the default architecture described in chapter 2.1, which will be queried every time a page is asked to the protected application.

RMS will be the licence server, and will be accessed every time a browser tries to open a protected file and doesn’t possess a licence. It’ll play a part in the packaging of the content to validate and generate licences distributed with the protected content.

In the client side, Internet Explorer will work with the RMA (Rights Management Add-on) plug-in which will allow it to interpret protected files and will supply the content to IE while enforcing the defined rights.

3.4 Filter configuration

The rights database links each protected document with a set of rights. Because we want to maintain independent the filter and the protected application, the filter will identify the documents to protect using the URL. The filter can access the HTTP headers and identify which content a user asked for without the need to interact with the protected application. So the only aspect that it’s dependent of the context of the application is the filter configuration. This configuration will define how the filter will react to each request.

The filter was designed so that it can be easy to configure a set of URL’s to the same set of rights. It’s possible to identify documents through regular expressions or sets of arguments to an aspx form.

In case there is no rights defined for a particular URL, it must also be configured a default policy which will be followed if no rights are found. Default Deny will deny any document without rights defined. This policy favours security. Default Permit will act as if there was no filter when an unprotected document is asked. This policy favours functionality.
4 System description

The adopted system is the one described in Figure 4.1:

![System Diagram]

Figure 4.1 – Rights controlled web system architecture

The main task on this project was to construct the Web Content Filter. This filter is integrated in the HTTP pipeline of the ASP.NET application. To install it’s a matter of configuring the application configuration file (web.config) and add this filter to the HTTP modules section.

4.1 Content protection

If the filter determines that the requested page is protected in creates a web archive (MHTML file) of all the content in the page (text, images, sound, animations, video, and other media) and encrypts the content using AES. Then it creates a publishing license where it describes the rights it found in the database. After it’s creation the filter submits the license to RMS so that it can be signed. In the end the RMH file is created and delivered to the user in the HTTP response.

4.2 Content consumption

When IE receives the content it determines if it’s a RMS protected file and delivers it to RMA. RMA will check if the environment is safe, by checking the application signature, and will ask RMS for a use license sending the publishing license found along the encrypted content, and the user’s credentials. If the request is valid, RMS will generate a license and deliver it to RMA so that it can decrypt the content and deliver it to IE to render it.
4.3 Security of the RMA plug-in

RMA relies in a component called lockbox to which is a personalized piece of software responsible for authenticating the valid use of the protected content and to protect the modification of trusted software. It’s guaranteed that the lockbox isn’t misused by ensuring that every application that uses as its signature registered in a manifest which is checked every time an application tries to use the lockbox. If the application is not listed in the manifest the lockbox refuses to work.

The lockbox to ensure security has to guard its keys in a safe place. In fact there is only one key that needs to be hidden from the users – the machine private key. This key will allow the lockbox to read all the other protected content that is used in the process. This key is guarded in DPAPI which is a cryptographic API to protect content so that only the user that encrypts data with it can decrypt it.

Since the lockbox encrypts content with an account on the user’s computer the user may have access to it. However, Microsoft does not disclosure the full process of protecting this sensitive key, and so it’s assumed that some of the protection relies on the secrecy of this process.

4.3.1 Trusted computing base

We saw that RMS, and all DRM system in general, have security problems coming from the fact that they need to hide something from the user in the user’s computer. The fundamental security question on the client side is that it’s not possible to create a completely trusted environment because the client has full control over his system. A secure computational environment is created, in first place, through the trusted computing base (TCB). TCB is a group of components of hardware and software which are critical to the security of a system and is distinguished from the rest in the sense that problems occurring inside the TCB might jeopardize the security properties of the entire system. By contrast, parts of a computer system outside the TCB supposedly cannot misbehave in a way that would leak any more privileges than was granted to them in the first place in accordance to the security policy.

In RMS case, the lockbox is the TCB which depends on an unsafe operating system (because it’s controlled by a possible attacker) and therefore the TCB is not safe.

However, Microsoft is studying improvements to the system which will include a hardware chip which assures trust in the systems that implement it. This component, which is an implementation of the Trusted Platform Module (TPM) is a microcontroller that stores information in a proved safe way in the client side. This component will assure that the information that it protects is safe from external attacks.

This technology used in place of the lockbox would resolve all the security problems describe previously. Since the technology offers proven security it’s possible to seal all the authentication and validation functions of the current lockbox and seal the machine private key assuring that the system is not compromised by the user.

However these components bring ethical problems since the user doesn’t completely control his machine, and so he cannot trust it. This can bring vendor lock-in platforms and systems that can be used to repudiate compromising declarations which could be used in legal actions.
5 Case studies

5.1 e-Portal

e-Portal is a Content Management System by Link Consulting used to publish, manage and administrate the content of a website. This application can be easily configured to be used with the developed filter. It's only a matter of identifying the content to be protected and configure the filter to protect the corresponding URL's.

This application identifies the content using the arguments on a single aspx form, so the filter is configured to associate a set of rights to any request that includes the identifying argument. Once the filter detects the argument in the URL it'll protect the document with the defined rights.

5.2 SQL Server Reporting Services

Reporting services is a server based report generation environment developed by Microsoft. It can be used to deliver a variety of interactive and printed reports. Because it's a highly distributed and interactive application this system recurs intensively to client-side scripting which changes the documents sent to the browser. Since the architecture of this project relies on a static document format – the RMH format – to protect sensitive data, the reporting services don’t work well with the web content filter. When the client-side scripts run on a protected document, they fail to join new content to the protected document.

5.3 Microsoft Office Sharepoint Server

Microsoft Office Sharepoint Server (MOSS) offers web portal, content management and business intelligence functionalities. From this platform it’s possible to perform searches on people, documents, and data, participate in business processes through forms, and access and analyze business data. MOSS brings compatibility out-of-the-box with RMS, which allow defining rights policies to apply to Office documents so that they are protected once they're downloaded.

On this case study we explore the possibility of protecting common web pages of the MOSS web portal. So as in the e-Portal case study we identify the protected content which in this case are only simple aspx forms distributed through folders. In this case we can use simple rules to protect one document and regular expressions to protect all the web pages inside a folder.
6 Conclusion

In this work we realize that internal attacks are dangerous and can’t be stopped with regular protection mechanisms used against external attacks.

DRM systems try to solve this problem protecting sensitive content through encryption preventing free access and restricting the allowed actions to the rights specified by the content publishers.

It was created in this work a DRM enabled filter which can be adapted to any ASP.NET application to protect web pages served by the application. The web pages served by the web application can be easily protected, being only a matter of configuring the rights database with rules that associate each protected URL with a set of rights.

When a client issues a request for a web page, the filter examines the URL, queries the database and determines if it’s a protected page. In that case, it creates a protected version of the content and restricts what a remote user can do with the served pages. The protected pages will maintain the rights protection even if a user with full rights copies the content to a user with no rights.

It was also referred that these systems have security problems because the final user handles both the protected content and the decryption key. These systems should not be considered alone but as part of a more complex security infrastructure. Corporate users should also be monitored and educated so that they understand the problems that can arise to their companies when they unintentional distribute information without authorization.
7 References


[2] Barak, Boaz; Goldreich, Oded; Impagliazzo, Russell; Rudich, Steven; Sahai, Amit; Vadhan, Salil; Yang, Ke: On the (Im)possibility of Obfuscating Programs, CRYPTO 2001, August 2001


[17] Yu, Yang; Chiueh, Tzi-cker: Enterprise Digital Rights Management: Solutions against Information Theft by Insiders, Professional experience report, Experimental Computer Systems Lab, Department of Computer Science of the State University of New York (SUNY), September 2004