Enterprise Digital Rights Management

Technology Comparison:
Authentica Active Rights Management and
Microsoft Windows Rights Management Services

By **Bill Rosenblatt**

July 14, 2005

Table of Contents

Introduction

Enterprise Digital Rights Management (Enterprise DRM) is an increasingly prominent technology to address security problems in corporations, governments, and other institutions. It complements document management systems, perimeter security, and other methods of restricting access to sensitive information. As more and more Enterprise DRM solutions come on the market, adopters need detailed technical information to help them choose the right one.

This white paper examines two technologies that are prominent in Enterprise DRM: Microsoft Windows Rights Management Services (referred to hereafter as Microsoft RMS) and the family of Active Rights Management technologies from Authentica, Inc. (referred to hereafter as Authentica ARM).

After brief descriptions of the two product offerings and their heritages, we provide an overview of Enterprise DRM technology in general, including a Reference Architecture. We then use that Reference Architecture as the basis for a detailed comparison of the two technologies. We conclude with a summary of the comparison, including relative strengths of the two technologies and the types of applications for which each one is more appropriate.

Microsoft DRM Initiatives

Microsoft Windows Rights Management Services (RMS) was released in November 2003. It is the third DRM technology to come out of Microsoft, the previous two being Windows Media Rights Management for audio and video and Digital Asset Server for e-books.

Microsoft RMS arose out of work that Microsoft had been doing since 2001 on unifying its DRM technologies into a single platform that would be extensible to all types of client applications and file formats. Microsoft RMS does have APIs that enable its functionality to be extended to new applications and file formats, but Microsoft chose to focus the product on corporate document protection applications that use Microsoft Office and HTML formats. The two DRM technologies mentioned above for audio/video and e-books are still available.

The latest release of RMS, and the one described in this white paper, is RMS Version 1 Service Pack 1, which shipped in April 2005.

Authentica History

Authentica's founders came from the Internet security field to start the company in 1998. Its original product was PageRecall, a DRM solution specifically for Adobe PDF documents, released in 1999. The technology's rationale was that a user could encrypt and apply protections to documents as part of the process of converting them to PDF.

Unlike most other DRM technology vendors, Authentica never explicitly targeted the consumer media and publishing markets, choosing instead to focus on document and email security applications for corporations, government agencies, and other institutions. Authentica also released MailRecall, a DRM solution for email messaging, in 2000.

Authentica released its first version of Secure Office, which provides DRM functionality for Microsoft Office and HTML format documents, in February 2004.   Its current product line consists of the following:

- Authentica Active Rights Management (ARM) Server Platform: license server platform common across all Authentica applications.

- Authentica Secure Content Server: server-side automated encryption and distribution capabilities.

- Authentica Secure Gateway: web-based secure email server for external clients.

- Authentica Secure Documents: client applications for PDF, Microsoft Office, and HTML formats.

- Authentica Secure Mail: email security for popular email programs.

- Authentica Secure Utilities: server utilities, including Authentica's APIs.

██
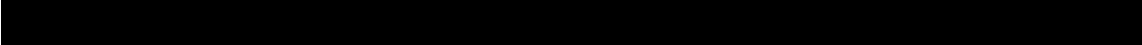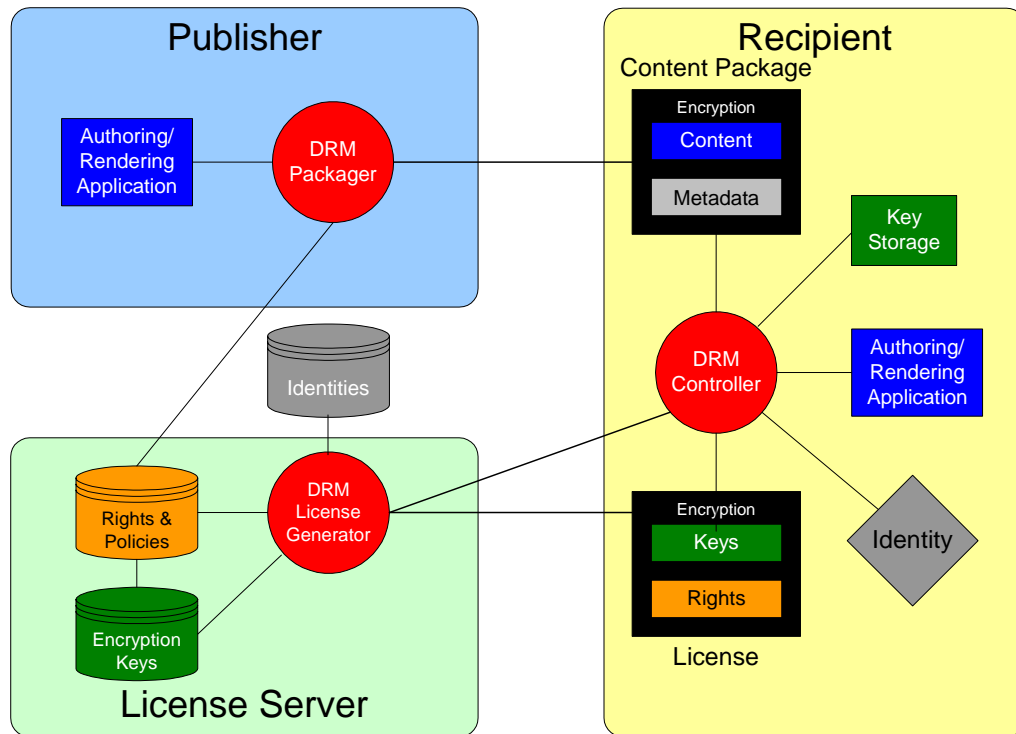
Enterprise DRM Technology Architecture

The following brief introduction to Enterprise DRM technology architecture provides some basic concepts that are common to most Enterprise DRM implementations and should serve as a guide for the ensuing comparison of Microsoft RMS and Authentica ARM.

DRM first appeared in the mid-1990s as a technology for controlling distribution of consumer media. Since then, DRM technology has been making its way into corporate document security applications. Enterprise DRM, as a market segment, came into being around 2003. Enterprise DRM borrows much in the way of technology for consumer media DRM, the main differences between them being:

- File formats and applications to which the DRM applies, such as Microsoft Office and PDF instead of audio, video, or e-book formats.

- Means of authenticating users who need to access DRM-protected content: Enterprise DRM makes use of existing corporate identity management technologies, such as the Lightweight Directory Access Protocol (LDAP) standard, and can often be used with stronger authentication techniques, such as 2-factor authentication (e.g., SmartCards or token cards).

- Consumer DRM generally requires more flexible sets of access rights or policies, in order to implement wide ranges of consumer media business models.

- On the other hand, Enterprise DRM requires more sophisticated ways to change or revoke rights after they are granted, in order to adapt to changing organizational circumstances and security concerns.

Reference Architecture

Enterprise DRM technologies generally conform to a Reference Architecture that is encapsulated in Figure 1.

- Figure 1: Enterprise DRM Reference Architecture. Adapted from B. Rosenblatt et al, *Digital Rights Management: Business and Technology.* New York: John Wiley & Sons, 2001.

The Reference Architecture consists of these major components:

- **Publisher**: can be a user's PC or a server. The Publisher consists of:

  - An *Authoring/Rendering Application* that creates content (whether by user action or automatically on a server) to be securely distributed.

  - A *DRM Packager* function, which encrypts content along with *metadata* that identifies it and, in some cases, adds various descriptive attributes.

- **License Server**: a server component that contains the following:

  - A repository of *Rights and Policies* that can be applied to content files.

  - A secure repository of *Encryption Keys* that are used to unlock content (see below).

  - A repository of *Identities* of users and/or devices. This is usually an external corporate identity repository, such as an LDAP or Active Directory server.

  - A *DRM License Generator* function that packages the client's identity (see below), rights descriptions, and encryption keys into licenses that the client uses to unlock the content (see below).

- **Recipient**: DRM software on a client device, such as a PC, centers around a *DRM Controller.* The DRM Controller does the following:

  o Reads the Content Package to find the identity of the file to be decrypted.

  o Collects *Identity* information about the device and/or user; passes it and the content identity (above) to the License Generator, which generates a License to the content[1].

  o Unlocks the License using a key stored in the *Key Storage.*

  o Retrieves the content key(s) from the License and uses it to decrypt the file in the Content Package.

  o Passes the decrypted content on to the *Authoring/Rendering Application* for viewing, printing, editing, or any other operations allowed by the rights descriptions included in the License.

  o The DRM Controller often provides other functions such as checking applications and the client device for integrity.

Client Integration

DRM Controllers must be integrated with the client's environment so that they can check credentials every time the user tries to execute an operation on the content that requires rights clearance, and so that the handoffs between DRM Controllers and client software applications are efficient and do not introduce security holes.

Here are the most common techniques for integrating DRM technologies with client environments, particularly PC-based environments:

- **Application Plug-In**: if an application is designed so that software can "plug in" to it to take over functions such as view, print, save, copy to clipboard, etc., then DRM functions can be implemented as plug-ins to that application. Adobe Acrobat Reader is an example of an application that was architected specifically for DRM plug-in software. Microsoft Office applications and the Microsoft Internet Explorer web browser can also accommodate DRM plug-in code.

- **Operating System Plug-In**: some approaches to DRM involve patching the kernel of the operating system so that it examines all input/output calls to see if they are on encrypted files. If they are, then the DRM Controller checks credentials of the user or device requesting the I/O before it grants approval.

- **Secure Viewer**: a third approach involves creating a special viewer application that can render certain file formats and acts as its own DRM Controller.

Other Core Concepts

In addition to the Reference Architecture, Enterprise DRM comprises a few other key technical concepts.

---

[1] In some cases, the License is generated previously and/or embedded within the encrypted content file itself.

Policies

DRM technologies enable *policies* on files to be set and enforced. Policies consist of these elements:

- **Principals**: the identities of devices and/or users to which the policies apply[2].

- **Rights**: the operations on content that a policy allows a principal to perform, such as view, print, copy to clipboard, save as, etc.

- **Extents**: the limits of the rights, such as until Date D, N Times, or only on subnet S.

Identities and Authentication

DRM technologies need to establish identities for devices and users, and ways of authenticating (verifying) these identities. Common methods of authentication include:

- Username/password for user identity.

- Entries in external identity management systems, such as LDAP or Active Directory.

- Device serial numbers.

- Ad-hoc ways of establishing unique device identity if (as is the case with PCs) the device does not store a serial number.

- Digital certificates, which are encrypted identities that a third party entity (called a certificate authority) creates and vouches for.

Encryption

Encryption is a key concept in DRM implementations. Here are some of the most common ways in which encryption is used:

- Encryption of content, usually using a symmetric-key algorithm such as AES (Advanced Encryption Standard), the U.S. government standard, or RC4 from RSA Security.

- Encryption of licenses and identities, often using the RSA public-key encryption algorithm.

- Computation of an encrypted hash or digest value on the content to ensure its integrity, using an algorithm such as SHA-1 (Secure Hash Algorithm-1) or TigerHash.

Just as important as the algorithm is the ways in which encryption keys are stored, particularly by the client device. The most secure way to store encryption keys is in hardware, although there are various techniques for hiding encryption keys in software as well.

In this section, we compare the technologies of Microsoft RMS and Authentica ARM based on the principles discussed above.

Server Architecture and Client Installation

Like most other Enterprise DRM implementations, both Microsoft RMS and Authentica ARM are primarily designed so that individual users can encrypt files on their own desktops. Both also have application programming interfaces (APIs) that enable developers to build server and client applications that automatically encrypt and package files; see p. 12 below.

Microsoft Overview

The License Server is the most important server-side element of both Microsoft RMS and Authentica ARM systems. In Microsoft RMS, this is simply called the *RMS server*. The RMS server primarily provides licensing and certifications services, which are explained below. It runs as an ASP .NET service on top of Microsoft Internet Information Server (IIS), a component of Windows Server 2003. It also uses a Microsoft SQL Server[3] relational database that stores configuration information, including RMS user IDs and their account certificates (see below), and activity log entries. RMS servers do not store identifiers or records of files that have been encrypted.

An organization that uses Microsoft RMS starts by designating a *root server*, which communicates with Microsoft itself to get a signed certificate that authorizes it to issue licenses. The root server sends Microsoft its public key, which Microsoft uses to generate an *XrML licensor certificate*. XrML (eXtensible Rights Markup Language) is a language used to represent rights specifications; more on that below. This process is called *enrolling* the root server[4].

Once an organization has a root server, it can enroll other servers within its firewall, and it must enroll all client devices. Enrolling other servers is roughly the same process as the one described above, except there is no need to obtain a certificate from Microsoft.

Establishing client devices in an RMS implementation involves three steps. First is to install the DRM Controller, known as the *RMS client*. The RMS client can be downloaded from the Microsoft website or distributed internally within an organization.

After that, the device must be *activated*. This is done automatically the first time a user attempts to use an RMS-enabled application on his PC. RMS invokes a self-contained activation process on the PC, which creates a *machine certificate* that is based on a unique device identifier and contains the device's public key. This certificate serves as a hardware credential for the PC.

---

[2] Microsoft RMS uses the term *principal* as a superset of this definition, to refer also to licenses and keys.
[3] Or Microsoft SQL Server 2000 Desktop Engine (MSDE), with some limitations, such as lack of ability to use MSDE in a server cluster configuration.
[4] This process takes place either through a direct Internet connection or, for "air gap" networks that are not connected to the public Internet, through a process that involves creating a floppy disc on an Internet-connected service and reading it onto the root server.

The final step in making it possible for a client PC to use a Microsoft RMS-enabled application is creating and *account certificate* for a user identity. This operation takes place the first time that a user on an activated PC attempts to create or access protected files. The organization's RMS root server generates an account certificate, based on the user's Windows login information, which contains a private key for the user. Account certificates expire after a period of time that can be specified.

This arrangement enables multiple users to create or access protected files on a given PC. It also enables a given user to use Microsoft RMS-enabled applications on more than one machine: the system simply generates an account certificate for a user on each machine that he uses.

Authentica Overview

The Authentica Policy Server is the core of the Authentica ARM architecture, which is more server-centric than Microsoft RMS. The Authentica Policy Server contains a relational database (Oracle or Microsoft SQL Server) that stores identifiers of protected documents along with their policies. It also stores encryption keys and activity log entries.

To set up credentials for internal users, an administrator simply needs to "point" the Authentica Policy Server at existing LDAP directories or domain controllers; it will obtain identity information from those servers on the fly.

Setting client PCs up to use Authentica ARM simply involves obtaining Authentica's DRM Controller (client software), via internal distribution or from Authentica's website, and installing it on the PC. No further steps are necessary.

Authentica's server software runs on Windows NT Server 4.0, Windows 2000 Server, Windows Advanced Server, Windows Server 2003, and Sun Solaris 2.8 and 2.9.

Authentication Schemes

The Authentica Policy Server can use several different types of authentication schemes:

- User authentication via Security Support Provider Interface (SSPI, native Windows login).

- User authentication via Microsoft Active Directory.

- User authentication via LDAP.

- User authentication via the Shared Secret username/password database; see p. 15.

- User authentication via X.509 PKI certificates.

- 2-factor user identification using RSA SecurID cards or SmartCards.

In contrast, Microsoft RMS uses the account certificate (described above) as the basis of its authentication scheme, along with Microsoft Active Directory or .NET Passport for user identity management. The resulting scheme is a flexible combination of user and device authentication. Microsoft RMS reads entries for users and groups in an organization's Active Directory and caches them in its database for immediate access. It authenticates a

user's identity via this information, along with the login and hardware information stored in the account certificate.

Both Microsoft RMS and Authentica ARM support 2-factor identification with X.509 certificates on SmartCards. To use this feature with RMS, the RMS administrator would add an X.509 certificate to a user's entry in the Active Directory. It can be configured so that a user must present his SmartCard and enter a password in order to get RMS credentials or every time he wants to publish or access RMS-enabled files.

Both systems also support offline access to content. A Microsoft RMS use license resides on the user's PC, so it can be used to access a file that is also on the user's PC without having to communicate with the RMS server – except if a revocation list is defined on it .

Authentica ARM can package policy information and content keys into a voucher that is made tamper-resistant via a proprietary algorithm and sent to the client. In order to use this offline option, an administrator has to enable it on a system-wide, per-document, or per-user basis *and* the owner of the file has to explicitly allow it.

Both Microsoft RMS and Authentica ARM can be used in multiple-server configurations with load balancing.

Policy Setting

Both Microsoft RMS and Authentica ARM provide ranges of policies that can be set on files. These are summarized in Table 1.

	Microsoft RMS	Authentica ARM
Rights		
View Rights	✓	(Always on)
View	✓	✓
Print	✓	✓
Save	✓	
Export (Save As)	✓	(Owner only)
Copy/Paste	✓	✓
Edit (Modify)	✓	✓
Allow Macros	✓	
Forward (Email)	✓	
Reply (Email)	✓	(Always on)
Reply All (Email)	✓	(Always on)
Principals/Extents		
Individual Users	✓	✓

User Groups	✓	✓
Subnet or IP Address		✓
Page of Document		PDF only
View/print with Watermark		PDF: view and print MS Office formats: print only
Embargo Date		✓
Expiration Date	✓	✓
Expire in N Days	✓	✓
Offline Access	✓	✓

- Table 1: Comparison of Rights and Restrictions

The Extents on rights listed in Table 1 are attributes or modifiers that specify or limit the applicability of rights to which they apply. For example:

- Subnet or IP address: rights apply to devices only if they are on a given subnet or at a given IP address.

- Page of Document: rights can be set differently on individual pages.

- Embargo Date: date *on or after* which the rights can be exercised (like a press release), as opposed to Expiration Date, which is the last date *on or before* which the rights can be exercised.

- Offline Access: rights can be exercised without online authentication.

Microsoft RMS stores policy information in XrML[5] certificates. XrML is a *rights expression language,* an XML-based language for specifying policies. It was designed by ContentGuard, Inc., and based on the Digital Property Rights Language invented by Dr. Mark Stefik at Xerox PARC in the mid-1990s[6].

Apart from specific differences in the types of rights and restrictions that can be set on files, the two systems also differ in the ways in which policies can be administered centrally. Both systems enable administrators to create rights templates, which can be used as shorthand for different sets of rights. These enable organizations to create classifications of documents by policy, such as public, classified, company confidential, etc.

However, Authentica ARM also offers several ways of making policy application mandatory. One of these is to monitor folders on network file servers and automatically set policies on files that users put there; Authentica calls this capability Secure Gateway. Other ways take advantage of Authentica's ability to integrate with external applications, such as email servers and content management systems. We examine these below.

---

[5] Specifically, XrML version 1.2.1.
[6] See Stefik's paper "Letting Loose the Light: Igniting Commerce in Electronic Publication," in his book, *Internet Dreams: Archetypes, Myths, and Metaphors* (MIT Press, 1996).

Client Application Integration

Authentica ARM and Microsoft RMS share the same basic technique of integrating DRM functionality with client applications: the *application plug-in* architecture (see p. 6). The advantage of the plug-in approach to DRM application integration is that it enables the DRM technology to control rights to application-specific features instead of just generic input and output operations. For example, Authentica ARM lets users specify that documents will be visibly watermarked when printed or viewed; Microsoft RMS lets users specify whether other users can run macros on Microsoft Office format documents.

Table 2 summarizes the applications with which Authentica ARM and Microsoft RMS (out of the box) are integrated.

In addition to those listed in Table 2, both Microsoft RMS and Authentica ARM offer client APIs that developers can use to create DRM-enabled applications that can publish as well as consume protected content. Microsoft's partners have already used the RMS Client API to create RMS-enabled versions of older Microsoft Office applications, going back to Office 2000. The exclusion feature (see p. 16) of Microsoft RMS enables an administrator to block any RMS-enabled applications that are malevolent or have been compromised.

	Microsoft RMS	Authentica ARM
Content Creation Apps		
Microsoft Word	2003	2000, XP, 2003
Microsoft Excel	2003	2000, XP, 2003
Microsoft PowerPoint	2003	2000, XP, 2003
Adobe Acrobat	-	Up to 7.0 for Windows, 5.0 for Solaris
Email Clients		
Microsoft Outlook	2003	2000, XP, 2003
Lotus Notes	-	4.6, 5, 6, 6.5
Web Browsers		
Microsoft Internet Explorer	5.5 SP2 on Windows ME, 6.0 SP1 on Windows ME/2000/XP	5.5 SP2 on Windows ME, 6.0 SP1 on Windows ME/2000/XP
Netscape	-	4.79, 7.2, Mozilla

● Table 2: Comparison of integrated client applications.

Server Application Integration

Integration of Enterprise DRM technologies with server-side applications is also important. Here are some common scenarios for server application integration:

- **Content/document/records management**: A content, document, or records management system can be integrated with DRM technology so that when a user checks a file or record out of the system, the system calls the DRM function to protect it. Without this functionality, the checked-out document is in the clear and can be copied and sent to people who are not authorized to see it.

- **Custom server applications**: server-side applications that create documents can be integrated with DRM technology to control access to those documents. For example, a custom human resources application might create documents containing employees' salary and benefits information that only the relevant employees should see. With a server API, it is possible to write a routine in that application (or an extension to it) that encrypts the documents and creates policies that only allow the relevant users to access them.

- **Collaborative workflow**: a system that allows multiple users to work on documents simultaneously can integrate with DRM technology so that only the users authorized to work on a document are able to do so.

- **Outbound email filtering**: a technology that analyzes the semantics of outbound email messages and classifies them according to confidentiality (or other criteria) can integrate with DRM technology, so that each message is protected according to its classification – e.g., sensitive messages are protected so that only the specific recipients can access them.

Microsoft makes its RMS Client API available for server-side applications as well; server applications can use the API to publish protected content only. This is the way in which a developer can integrate Microsoft RMS with server applications such as the above. To date, Microsoft RMS has been integrated with the Titus Message Classification email filtering application for Microsoft outlook.

Although Authentica ARM does not have a client application API (see p. 12) per se, it has an API specifically for integrating with server applications, which can also be used to integrate with authoring/rendering applications on the client side. The Authentica server API can be used in the above scenarios to integrate document protection capabilities with document management, workflow, email, and other applications.

Authentica ARM has been integrated with the following server applications:

- **Document and content management**: Documentum, IBM Content Manager, Hummingbird Enterprise

- **Collaboration**: Documentum eRoom

- **Email filtering**: ClearSwift MIMESweeper, Trend Micro ScanMail, Symantec BrightMail, IronPort.

Authentica also includes other ways of integrating with email, through its Content Security Server (CSS). CSS applies protections to messages and file attachments, then sends notifications to recipients that they have messages along with URLs that they can click to access them. When a user clicks on the URL, he sees a user interface that resembles a web mail application (the message sender uses this same interface to set policies on messages).

Once a user has authenticated himself, CSS decrypts the message and streams it to the user over an SSL connection. This method has the advantage of not requiring any client software, though the message is in the clear once it reaches the user. For tighter security, CSS also offers an Advanced Protection mode, which requires the Authentica client software and which sends the message to the user in its DRM-packaged form, to be decrypted and controlled on the client. If an email message contains any file attachments, CSS converts them to PDF on the server side and then encrypts them using Authentica's standard PDF packaging capabilities. Authentica's server-side PDF utility can convert over 300 different file formats to PDF.

In addition to the web user interface, CSS can also integrate with external applications through web services.

Authentica's server API also contains hooks for externalizing policy decisions and expanding them beyond the policies that Authentica ARM supports by default. With Authentica's server API, it is possible to pass a user ID, file identifier, and description of the intended use to an external application that can return a decision on whether or not to grant permission. The external application can use whatever means it needs to make this determination.

One possible use of this feature is to make the integration of DRM with document management more efficient. A document management system can, as mentioned above, use a DRM server API to protect a document when a user checks it out of the database. But after the document is checked out, it is possible that its permissions within the document management system could change, putting the protected checked-out document "out of sync" with the document management system. With Authentica's server API, however, it is possible to pass decisions of whether to grant access to a given user on directly to the document management system for evaluation on the fly. Microsoft RMS does not support an equivalent feature.

Another possible use of the external policy decision feature might be server-side code that interacts with an external policy engine that processes SAML[7] assertions – by effectively translating between ARM policy settings and SAML assertions. For example, the policy server could send the external policy engine an authentication and attribute assertion for a user who is requesting access to a document. The external engine would respond with an authorization assertion allowing or denying access to the document, which the code using Authentica's API would process accordingly.

External Usage

Although many applications for Enterprise DRM are for protecting information within a company, Enterprise DRM is at least as valuable in restricting access to information that is sent outside the firewall to customers or business partners.

The trickiest aspect of extending Enterprise DRM functionality beyond the firewall is determining how to authenticate identities of external users. Most Enterprise DRM technologies use a company's ID management system, such as an LDAP or Active Directory server, to authenticate user identities; an organization might not wish to put external user ID information into those internal servers.

---

[7] Security Assertion Markup Language, an OASIS standard; see http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.

Authentica ARM solves this problem via a so-called *shared secret database*, which is sort of a lightweight identity repository that contains just enough information to enable external users to be authenticated without replicating entire LDAP or Active Directory entries.

The shared secret database is most typically used to facilitate secure email communication with outsiders. If a protected message is sent to an external user who is unknown to the shared secret database, the Authentica Policy Server will send that user a separate email containing an *initialization token*, which contains a URL. When the user clicks on the URL, he is prompted for a password. The password is then stored in the shared secret[8] database, and the external user can then use it to access files.

The above protocol assumes that the insider who sent the original email is trusted enough to identify users who should get access to the protected content. In that case, the Authentica Policy Server is set up to automatically initialize anyone in the external user's Internet domain; i.e., the external user's domain is a *trusted domain*. The Policy Server can also be set up in other ways, such as to refuse initialization outright to a user who is not in a trusted domain, to refer the decision to a system administrator to make on a case by case basis, or to require that the user authenticate via a digital certificate containing his email address as a condition of authentication.

Microsoft RMS, in contrast, relies on a more restricted concept of trusted domain, while also taking advantage of the .NET Passport identity management scheme that Microsoft uses for its own services, such as Hotmail and MSN.

Microsoft RMS can authenticate external users under three sets of circumstances:

- The organization has an Active Directory server on its extranet for its external users.

- The external user's domain contains its own RMS server. In that case, the RMS server can import the licensor certificate of the external RMS server, thereby making it a trusted domain.

- The external user's domain does not contain an RMS server. In this case, the external user must have a .NET Passport account and must obtain an account certificate (see p. 9) through that account when installing the RMS client software. In addition, the internal RMS server must have the Microsoft Certification Service as a trusted domain.

In other words, Microsoft RMS requires a commonly recognized identity between internal and external users. Microsoft itself serves as a "bridge" between the two domains in the .NET Passport identity scheme. Beyond that, Microsoft has partnerships with service providers like GigaTrust and Certipost to provide inter-enterprise identity management.

Dynamic Rights and Revocation

A vital aspect of DRM technology architecture is how it can respond to security threats or changes in security policies. In the consumer media distribution applications for which DRM was originally designed, rights are generally set once with the intension of never changing them unless the DRM technology is compromised. In Enterprise DRM

---

[8] In cryptographic terminology, a password is a specific example of a "shared secret."

applications, however, there are various scenarios that involve both changing rights on a document and targeted responses to security breaches.

Here are some examples of scenarios that warrant changing rights on a document:

- A press release can only be viewed by certain trusted journalists until its embargo date, after which time it can be viewed (but not changed) by anyone.

- A product pricing sheet is only accessible by certain privileged users until the product's release date; but then management decides to postpone that release date because the product is not ready. Then, if there is a previous version of the pricing sheet, it can be locked so that no one can access it any longer.

- A company may want to lock certain documents permanently, regardless of who may have copies of them, once they have existed for a certain period of time. This can be useful to enforce record retention policy for documents that should be destroyed after the lifespan that the policy specifies.

- An employee is suspected of leaking sensitive information to a competitor or the press, prompting an administrator to revoke his rights to access those particular documents while the matter is being investigated.

- An employee leaves the company, and the company wants to make all sensitive content in his possession – whether on his laptop, home computer, CD-ROM, etc. – inaccessible.

- A company is soliciting proposals from contractors on a project RFP that contains sensitive material. The company wants to ensure that only the intended recipients within the contractors are able to view the material, and after the proposals are completed, it wants to revoke all of the contractors' rights to access the material.

In addition, there are various security breach scenarios that call for appropriate responses from the DRM system, such as:

- A hacker exploits a DRM-enabled application by turning it into a "booby-trap" for stealing content.

- A particular employee or device is suspected of deploying a hack, the exact nature of which is unknown.

- A private key has been compromised.

Microsoft RMS has three general ways of providing dynamic rights functionality. The first is the simplest. If a user wishes to request *more* rights on a document (such as the right to print, or an extension on the expiration date), then Microsoft RMS provides a way for the creator of the document to include a web URL that the user can visit in order to obtain more rights. In this case, another version of the document with the requested rights can simply be created and sent to the user.

The second way that Microsoft RMS handles dynamic rights is its *exclusion policy* feature. Exclusion policies can be set up by administrators on RMS servers (or server clusters) to

bar certain types of system entities (or *principals*, in RMS and XrML parlance) from obtaining licenses to files from RMS servers:

- Versions of the RMS *lockbox* – a code component that is unique to each client installation and generated at client software install time – that are suspected to be compromised. If a user's lockbox is excluded, then he has to reinstall the RMS client software on his PC to obtain a new lockbox.

- Versions of Microsoft Windows that are suspected to be compromised. Users would have to upgrade their versions of Windows to one that is considered to be secure.

- Public keys in account certificates (see p. 9). This is done when a private key in an account certificate is thought to be compromised.

- User IDs. This does not, however, cover the scenario of the employee leaving the company or being suspected of leaking information, because the employee can still access files to which he *already* has a license. A few different techniques can be used to cover this eventuality. One is revocation (see below); another is setting up a license to the file that expires in N days, requiring licenses to be reissued that often. If a user were excluded, then he would not be able to get a new license after the next time it expires.

Finally, Microsoft RMS has a feature called *revocation lists* that implements additional ways to curtail rights. Revocation lists are best viewed as overlapping yet complementary to exclusion policies, because they effectively disqualify other types of system entities from obtaining licenses.

Revocation lists enable these types of entities to be barred from inclusion in content use licenses:

- Content files. Revoking content files causes them to be inaccessible to anyone, even those who have rights on the files and would normally be able to obtain use licenses to them.

- RMS-enabled applications, when they are compromised or "booby-traps" as described above.

- PCs, when they are suspected to be compromised. This is more general than the exclusions of lockbox or operating system versions described above.

- Users, via their IDs. This has a similar effect to the exclusion of user IDs described above.

- Licenses or certificates, including publishing and use licenses. This is useful in the scenario where an employee leaves the company or is otherwise no longer trusted.

The great advantage of the revocation feature is its high degree of flexibility, but that is offset by the fact that only a system administrator can use it, and he must be able to write correct XrML. There is no user-friendly administrative interface or "wizard" to assist in creating revocation list entries. Furthermore, revocation lists for content licenses can only attach to licenses that were generated through policy templates; licenses that were

generated with ad-hoc policies cannot have revocation lists attached to them, because there is no server-side entity to which they can be attached.

Some rights revocation scenarios involve republishing the file, which may lead to inconvenience. For example, assume that someone wants to revoke rights that a particular user has to some sensitive files, under the above scenario that the user is suspected of leaking information. To address this situation, an RMS administrator must take these steps:

1. Revoke the user's use licenses for the files in question.

2. Repackage the files with new publish licenses that do not include the user as one who has access.

3. Revoke the publish licenses on the old files.

4. Send the repackaged files to the other users who are still authorized to access them, along with a notification that the old files are no longer valid.

Steps 2-4 of the above are necessary in order to avoid the possibility of the user attempting to get new use licenses for the files.

Authentica ARM has a different mechanism for dynamic rights. It enables a user to reassign rights to a document (i.e., change the terms in a license) on the fly, without requiring an administrator, and through the same user interface that the user used to assign the rights in the first place.

To revisit the above scenario: the creator of the sensitive files can simply change their rights so that the errant user no longer has access to them; this happens instantaneously. Conversely, a document that someone sent out with an expiration date can have that expiration date extended without the recipient needing to request a new document or even know that the extension is being applied.

The dynamic rights mechanism in Authentica ARM is simpler than Microsoft RMS's exclusion and revocation features, and therefore easier for end-users who do not have administrative privileges or XrML knowledge. It also facilitates building dynamic rights features into server-side applications through Authentica's API. For example, a server-side application can be written that uses the dynamic rights capabilities to enforce retention policies, as in the example on p. 16.

Authentica ARM has an equivalent to the Microsoft RMS exclusion policy for old versions of the RMS lockbox: an administrator can exclude older versions of Authentica software plug-ins if they are thought to be compromised; this forces users to obtain the latest versions of the plug-ins in order to access protected files.

Authentica ARM does not have the ability to bar rights-enabled applications directly; however, it is possible in many cases to write server code that uses Authentica's dynamic rights features to bar access to files created in certain applications if they are no longer trusted.

Encryption

Microsoft RMS makes extensive use of public-key encryption. It maintains public-private key pairs, using the RSA algorithm at 1024 bits, for servers, client licensors (used to issue content use licenses from PCs without communication with the RMS server), client PCs, and users. It uses the AES symmetric-key algorithm to encrypt content. Content keys are encrypted with RSA algorithm public keys for users and client licensors. Other keys form a trust hierarchy whose top level (root) is the Microsoft service that enrolls the organization's root server, as described above on p. 8.

Microsoft does not have access to any content, user, server, or other private keys used in an organization's RMS implementation. The only key Microsoft ever stores is the public key of the root server.

Authentica ARM makes more use of symmetric keys. It uses 128- or 256-bit AES to encrypt content. It then encrypts content keys using session keys that it generates at user authentication time according to the method used to authenticate the user. For example, if the user authenticates himself via a digital certificate, the certificate is used to generate the session key; if the user authenticates via native Windows authentication, then it generates the session key using a proprietary algorithm based on a shared random number and the SSPI credential. All communication between server and client is done via SSL. Session keys are used once and then discarded, making their compromise by a hacker a non-issue.

Summary and Conclusions

Microsoft RMS and Authentica ARM are both effective technologies for protecting sensitive corporate information. Here are some of the most important relative strengths and differentiating features of the two technologies.

- **Client integration**: Starting with Office 2003, Microsoft Office and Internet Explorer applications are developed with RMS integration specifically in mind, thus minimizing the possibility of security holes. Therefore one has to assume that Microsoft RMS is more seamlessly integrated with Microsoft applications, including Office (Word, Excel, PowerPoint, Outlook) and Internet Explorer. Microsoft RMS also has a client API that enables third-party developers to integrate it with other applications, whereas Authentica ARM does not. Microsoft's partners have already used the RMS client API to integrate RMS with older versions of Office, dating back to Office 2000. Authentica ARM does not have a client API per se, but it does already integrate with Adobe Acrobat, Lotus Notes, and various web browsers and email applications as well as Office 2000 through 2003 (including Outlook). Both technologies' APIs for DRM-enabling are dependent on third-party applications being designed to accept DRM plug-ins.

- **Server integration**: Both technologies are capable of integrating with server applications that generate content so that the content can be protected. Microsoft RMS essentially uses its client API to integrate server applications so that they can protect content that they handle. Authentica has a more full-featured server API that includes hooks for external systems to make policy decisions beyond those that the Authentica Policy Server can make on its own. At this point in time, Authentica has a longer list of partners with server applications that have been integrated with DRM.

- **Policies**: Authentica ARM and Microsoft RMS have roughly comparable sets of basic policies that can be applied to documents. Authentica ARM provides more options for limiting access, such as by IP address or subnet, and it provides the capability of attaching visible watermarks to documents. Microsoft RMS has a richer set of policies for email actions (forward and reply), whereas Authentica ARM relies on protecting email messages themselves, regardless of where they are sent (including replies and forwards). Authentica ARM has more ability to enforce the usage of administrator-defined policy templates throughout the enterprise.

- **Dynamic rights and revocation**: Authentica ARM enables users to change rights to existing protected documents through a graphical user interface, without administrator intervention and without requiring the documents to be repackaged. Microsoft RMS requires administrators to write XrML code to revoke rights to existing packaged files and often requires that files be repackaged and redistributed with new sets of rights. Microsoft RMS has a very flexible mechanism for revoking entities from the system, which gives it more options than Authentica ARM for responding to potentially compromised hardware, keys, or application software. But Authentica ARM's architecture makes compromised application software and keys less relevant to begin with.

- **Authentication**: Microsoft RMS uses Microsoft Active Directory and .NET Passport as its means of authenticating users. Authentica ARM has a wider

variety of authentication mechanisms to choose from (including .NET and Active Directory). It also offers a "lightweight" way of authenticating external users, via a password database that is separate from the organization's internal identity database, thus making it easier to send protected content to users outside the firewall; in contrast, Microsoft RMS requires either that the external organization also use RMS or that external users authenticate themselves through Microsoft via .NET Passport.

- **Platforms**: Authentica ARM runs on a wider variety of server and client platforms than Microsoft RMS. Whereas the latter runs only on Windows Server 2003, Authentica runs on Windows server software dating back to Windows NT as well as recent versions of Sun Solaris. And while Microsoft RMS generally requires Microsoft SQL Server database, Authentica works with Oracle as well as SQL Server. Authentica's Secure PDF client runs on Solaris as well as Windows clients.

In general, Authentica ARM is the more desirable technology for organizations with heterogeneous server operating systems, identity management, or database platforms. It is also superior for organizations that need to integrate Enterprise DRM functionality with other elements of enterprise IT security (identity management systems, single sign-on capabilities, document management and collaboration tools, etc.), and organizations that need to convey sensitive information to parties outside the firewall that may have their own arbitrary identity management schemes.

In contrast, Microsoft RMS is a better choice for organizations with Microsoft-centric infrastructure that need to integrate DRM functionality with other client software applications, whether appropriately designed third party packages (e.g., CAD/CAM) or in-house applications.

About the Author

Bill Rosenblatt, president of GiantSteps Media Technology Strategies, has 20 years of experience in technology architecture, business development, and marketing; publishing; new media; and online education. He has been a business development executive at a leading technology vendor, an IT executive at major publishing companies, and chief technology officer of an e-learning startup. He has expertise in digital media technologies such as digital rights management, content management, streaming media, and publishing systems. Bill is the author of several books, including *Digital Rights Management: Business and Technology* (John Wiley & Sons, 2001), and he is Managing Editor of the Jupitermedia newsletter DRM Watch (www.drmwatch.com).

About GiantSteps Media Technology Strategies

GiantSteps Media Technology Strategies is a management consultancy focused on the content industries that help its clients achieve growth through market intelligence and expertise in business strategy and technology architecture. GiantSteps' clients have included branded content providers and digital media technology vendors, ranging from early-stage startups to Global 500 firms.

Contact:
GiantSteps Media Technology Strategies
1841 Broadway, Suite 200
New York, NY 10023
phone: +1 212 956 1045
email: info@giantstepsmts.com
Web: www.giantstepsmts.com

White paper commissioned by Authentica, Inc.

About Authentica

Authentica is the leading provider of Enterprise Rights Management (ERM) solutions for actively protecting and controlling valuable business information, shared internally or externally. ERM solutions let users share valuable documents and e-mail without giving up the rights to determine what happens to this information, regardless of who has it or where it is stored.

Authentica's market-leading enterprise rights management (ERM) solutions help global companies and government agencies actively control, secure, and track sensitive documents and email within a workgroup, across departments or agencies, or with partners and suppliers . Authentica's comprehensive and persistent security is helping corporate entities comply with a wide range of federal and state government regulations, including Sarbanes-Oxley, ITAR, SB 1386, HIPAA, and the CMS-developed Acceptable Risk Safeguards.

Authentica's Active Rights Management platform delivers the unique ability to dynamically control and manage information, even allowing organizations to revoke access after external distribution. And whether content resides in Microsoft Office documents, Adobe PDF format, email, or Web content, Authentica's flexible architecture helps organizations protect data without altering their existing workflows. It's tightly integrated, easy to deploy, and leverages existing enterprise investments, such as content management systems, email gateways, and portal technology. To ensure that its customers gain maximum benefit, Authentica's Professional Services group works with them to implement an ERM solution customized to their unique data classification policies.

Contact:
Authentica, Inc.
91 Hartwell Avenue
Lexington, MA 02421
phone: 781.487.2600
email: info@authentica.com
Web: www.authentica.com