



Choosing an Enterprise Rights Management System: Architectural Approaches

One of the prime features of electronic documents is their ease of movement. They can be passed from one person to another not just to share and disseminate information but also to collaborate on that information. This key attribute of electronic documents gives them the technological edge over paper supporting the ultimate goal of the 'paperless office'.

In contradiction to this freedom of movement, is a need to control document content and ensure that it does not get into the wrong hands or is used illegitimately. Enterprise rights management systems (ERM) attempt to combat this data leakage by going a step beyond encryption and adding controls to the use of the content of a document (not just protecting the file itself). However, the way that an ERM system approaches this task is vital in retaining the fluid communication characteristic of electronic documents.

If the system in any way reduces or removes the inherent fluidity of a document, then this vital feature of free-flowing movement will be lost.

Architectural Approaches of ERM Systems

The underlying architecture of current enterprise rights management (ERM) software can be divided into two main approaches: Tethered (server |client) and untethered (standalone or peer-to-peer).

Each of these approaches has its merits and deficits. The purpose of this article is to summarise the differences in these two approaches and to show that consideration of the architecture can be an important factor when choosing an ERM system.

Consideration will be given to the impact that the two approaches have on their use as content security systems in the following areas:

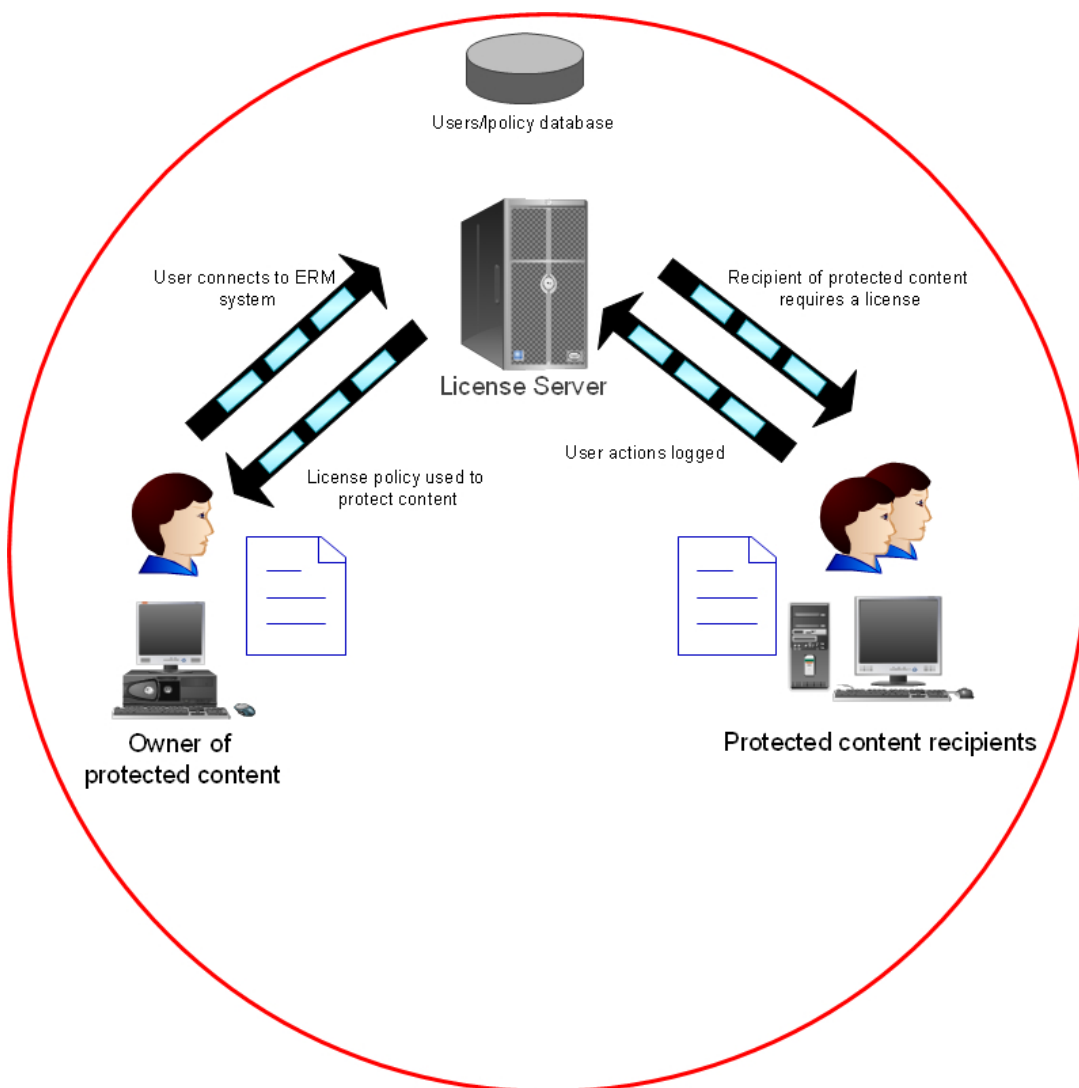
1. Closed or Open Systems: Affect on Sharing of Protected Content
2. Installation and Setup Implications
3. Integration with third party products
4. Usability and Transparency
5. Audit and Dynamic Changes to Rights Restrictions
6. Encryption Key protection

The Architecture of a Tethered ERM System

The tethered ERM model is based on a license server ; client framework. The licensing server handles the cryptographic key, which is required to access the protected content. The use of the word tethered refers to the fact that, at some point, the client attempting to access protected content has to connect back to this licensing server to access this key.

Content access is authenticated via a license server: This means that the authentication mechanism is tied to the license server and not the content. The content thus being tied to the license server.

Overview of a tethered ERM system

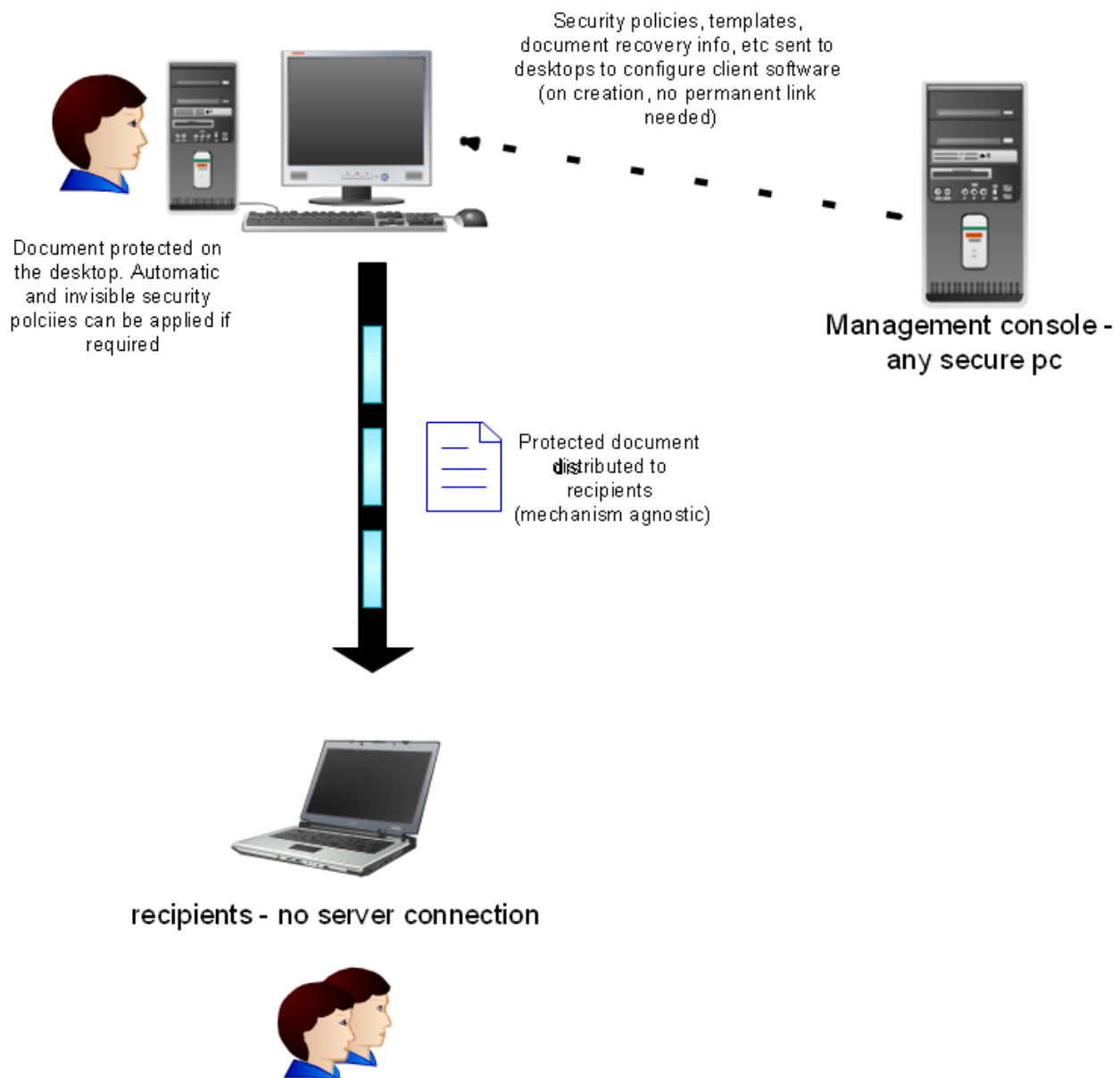


The Architecture of an Untethered ERM system

The untethered model does not require a license server to handle the cryptographic key; instead the key is distributed, securely, within the protected document itself. The effect of this is that the person protecting or accessing protected content does not need to log onto a server to download the encryption key.

The control of content access is not tied to a license server, *i.e.* the document itself contains the authentication mechanism requirements required to access it, thus retaining the document movement attributes.

Overview of an untethered ERM system



Closed or Open Systems: Affect on Sharing of Protected Content

Ideally, an ERM system should allow protected content to be shared in a seamless manner without the need for additional management overhead such as deciding who will be a member of the system and then managing those users. The protected content itself should be allowed to control access using the inherent access control mechanisms used when protecting content (if the user has the correct authentication token then they can access the content). In this way, such a system can evolve and extend, to naturally include users, as and when required, without the requirement of additional components.

An architecture based on a tethered model, is intrinsically a closed system. Anyone who needs to access documents protected by such a system must become a member of the system by being added into the license server list. The license server list is a repository of user identities and is often tied to LDAP or Active Directory. Membership of the system may be controlled exclusively by an administrator or can be made available for individuals to add themselves. How this is achieved can vary: For example, users within the enterprise can be added from those listed in Active Directory, whereas external users will generally require manual addition by the ERM administrator.

ERM systems that use an untethered method of content protection are inherently open and remove the need to, and potentially allow anyone, in any location, to share in the secured information (assuming they have the correct access control mechanism to do so). They do not require a centralised method of maintaining the users of the system. In this way, untethered ERM is much more of a flexible architecture and allows a more natural and fluid system to evolve, evoking the way that humans naturally work.

Installation and Setup Implications

Tethered System Components

Tethered ERM systems require the setup and management of a license server as this is the core component of the system. Typically, the policy or license server would contain a relational database that holds information on users, security policies, protected document identifiers, *etc.*

Such systems often have a management console that will create and manage security policies.

Coupled to the license server and management console, is a client based installation which performs the 'packaging' of the content.

Population of the database used to control the users within the ERM system on company internal users (employees) would be obtained, typically from LDAP or Active Directory lists. Information on others external to those lists, but who would need to share documentation, would be handled in another way, for example using a web-based password generation system.

Installation and configuration of the license server and configuration and maintenance of the related database is a complex part of the setup of a tethered system. The license server is the main constraining component of the system and this may affect

scalability unless carefully managed or controlled by the vendors design of the system.

Untethered System Components

In an untethered system the main component is the client installation. Untethered ERM systems do not have the license server component; however they should have a management console that will allow requirements, such as security policies, templates and document recovery, *etc.* to be created and managed.

Because the main component of the untethered system is the standalone client, and a license server is not required as part of the system, scalability is not affected and is essentially unlimited.

Integration with Third Party Products

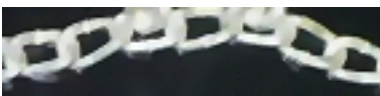
ERM systems are often a natural extension of other products such as online collaboration portals, content management systems, *etc.*

How the ERM system integrates with other products is an important consideration.

An untethered system with an appropriate API could be integrated deeply into the system to become a component of the system. This is because untethered systems do not require additional controlling layers such as a license server. Integration of such systems is comparable to adding a link into a chain.

Conversely, tethered applications can also be integrated into third party applications but their model ensures that the integration would not allow such a seamless approach. Integration of such a system would be comparable to adding a link to a chain which was then part of its own separate chain, *i.e.* creating a chain within a chain.

Untethered ERM integration



Tethered ERM integration



The implications of this are that integration of a tethered system adds extra layers into the third party product above and beyond those layers already in place.

Usability and Transparency

Any software product that is involved in data security, especially the security of end user generated information, must be as transparent as possible*. Transparency of operation, affords an application a number of benefits in terms of usability, which include:

- ✓ **End user ease of use in applying protection**– an ERM product that needs minimal or no end user intervention to apply protection is more likely to be used than a product that requires extra steps to apply the protection.
 - **Connectivity requirements**
 - Tethered systems require that (at some point) the end user applying the protection to the document must be online to allow the download of the license policy, used to protect the document. Tethered systems force standalone desktop applications such as Word, *etc* to act like server | client applications.
 - Untethered products do not require any connection to an external license server to apply protection: protection is applied directly on the desktop
 - **Automation of the protection process**
 - If the protection process can be fully automated and made invisible to the end user, thus removing end user intervention, then application of security can be fully controlled and enforced by the enterprise.
 - A system that requires access to a component outside of the normal desktop authoring environment (*i.e.* a license server) interrupts the transparent nature of the protection process. Although caching can be used as a 'fix' for this loss of transparency, the caching mechanism still requires both connectivity to the server, at some point, and the fore-knowledge to cache security policy licenses with expiration dates, *etc.*
 - **Secondary Logons**
 - Secondary logons to the ERM system are often used by tethered models and act to reduce the transparency of the ERM system. Extra logons to apply protection, or access protected documents, results in users not using the system correctly or avoiding the use of it altogether.

- ✓ **Recipient, ease of use** –recipients of protected documents may require access to those documents outside of the company firewall and with poor or no internet connectivity. Therefore, the requirement to log onto a license server, as required by tethered ERM systems, is greatly restricting in the use of ERM. Although some way forward has been made by tethered systems, to resolve this issue (namely caching of licenses) this does not improve transparency over all, as caching requires that users know they will be without a connection and a

* Transparency is also affected by the type of authentication used, *e.g.* use of a transparent mechanism such as digital certificates or a mechanism requiring network connection such as Active Directory. However a full look at authentication is outside the scope of this document.

high degree of management of the caching is required both on an end user basis and by the system administrator.

Conversely, untethered ERM systems do not require an end user to have network/internet connectivity to access a protected document (unless the document has date restrictions in which case there may be a requirement to check the time against an internet time server).

Server Components: Audit and Dynamic Change to Rights Restrictions

The major advantage seen to be gained by using a tethered system is the ability to:

- a. track document usage
- b. dynamically change/revoke user rights

Tethered systems perform these functions as an inherent part of their system setup. Because these systems use a server component to tether the protected content a consequence of this is that content can be audited and controlled as part of the protection process. However, this is also the approach that reduces transparency and adds issues in outside-firewall secured document movement.

Untethered systems do not have these functions as an inherent part of the system, but they can offer these functions as options within the system. Tracking can be performed on protected documents on the desktop with no connection required except to upload the tracking information (upload of the tracking data being the only point that tethering occurs). Similarly, changing rights restrictions can be performed by allowing those documents that require that function to become tethered. This dual approach to content protection allows the inherent fluid and transparent nature of an untethered system to continue.

Encryption Key Protection

Tethered systems store the cryptographic key encrypted on a license server. The key is downloaded when opening protected content.

Untethered systems store the cryptographic key within the protected content.

One of the fundamental problems faced when implementing any ERM solution is how to allow an authorised user to decrypt protected content while keeping the key (required for decryption) secret from the user.

Hiding the encryption key from the user is necessary to prevent the user from taking the key and using it to decrypt the protected content outside of the ERM system and so affording an unprotected copy of the content. However, keeping the key secret is a difficult problem.

It may appear that the tethered system affords a higher level of protection of the key, as it resides mainly on the server and may only be available on the local machine in memory while the content is open.

However, this view omits both the requirement of caching of licences, for off-line access, and the ease by which keys can be obtained from memory by an attacker with simple tools.

These factors mean that the same efforts to protect keys must be made in both tethered and untethered designs. Without hardware blackbox encryption, this means that extensive use of obfuscation must be used.

"The consensus among security professionals I have discussed the matter with is that if you want to do DRM without hardware support – another topic that is not for today – then there is little choice but to rely on some level of secrecy or obfuscation." says Professor Roger Needham FRS, Managing Director, Microsoft Research Limited, Cambridge

(See next page for overview)

✉ info@avocosecure.com

US: +1 415 839 9433

International: +44 207 851 6070

🌐 www.avocosecure.com



Overview

	Tethered	Untethered
Transparency in action	Transparency issues when protecting or accessing protected content	Inherently transparent system. Transparency may be affected by choice of authentication
Protection of encryption key	Requires strong protection	Requires strong protection
Ease of protecting documents offline	Caching needs to be performed to allow offline protection Some systems use a type of 'synchronisation' to attempt to make the caching process more transparent	No issues, protection can be performed anywhere at any time if allowed under security policies set
Ease of accessing protected documents offline	Unless caching is in place users would be unable to access a protected document offline	No issues in accessing protected content offline
Creating an audit of document usage	Inherent part of the system	Optional component of the system. Done locally then uploaded to a database when connected to the network
Changing access rights or restrictions dynamically	Inherent part of the system	Often available but document chosen to have this restriction would become tethered and so lose transparency and fluidity
Ease of integration into third party enterprise systems (e.g. content management systems, collaboration portals, etc.)	Can normally be integrated easily May involve extra server / database overhead	Becomes an inherent part of the system No extra server or database overhead
Scalability	Dependent on system as server piece can constrain scalability	Scalability not constrained

Related document

- ✓ Document DRM: Device Drivers vs. Plug-ins for Accessing Content Protected Files