[MS-RMSO]:
Rights Management Services System Overview

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

This document provides an overview of the Rights Management Services System Overview Protocol Family. It is intended for use in conjunction with the Microsoft Protocol Technical Documents,

publicly available standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

A Protocol Family System Document does not require the use of Microsoft programming tools or programming environments in order to implement the Protocols in the System. Developers who have access to Microsoft programming tools and environments are free to take advantage of them.

Abstract

Rights management allows individuals and administrators to encrypt and specify access permissions to various types of data, including documents and email messages. This helps prevent sensitive information from being accessed and used by unauthorized people. After permissions to content have been restricted by using rights management, the access and usage restrictions are enforced no matter where the information is, because the permissions are stored in the content itself.

Rights Management Services (RMS) enhances an organization's security strategy by providing protection of information through persistent usage policies. This also enables usage rights to be enforced after the information is accessed by an authorized recipient, such as restricting copying, printing, or forwarding. RMS also helps organizations enforce corporate policy governing the control and dissemination of confidential or proprietary information.

This document describes the intended functionality of the Rights Management Services System, how it interacts with systems or applications that create or consume rights protected content, and how it interacts with management clients that need to configure and manage the system.

Revision Summary

Date	Revision History	Revision Class	Comments
05/22/2009	0.1	Major	First Release.
07/02/2009	1.0	Major	Updated and revised the technical content.
08/14/2009	2.0	Major	Updated and revised the technical content.
09/25/2009	2.1	Minor	Updated the technical content.
11/06/2009	3.0	Major	Updated and revised the technical content.
12/18/2009	4.0	Major	Updated and revised the technical content.
01/29/2010	5.0	Major	Updated and revised the technical content.
03/12/2010	5.0.1	Editorial	Revised and edited the technical content.
04/23/2010	5.0.2	Editorial	Revised and edited the technical content.
06/04/2010	5.1	Minor	Updated the technical content.
07/16/2010	5.1	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	5.1	No change	No changes to the meaning, language, or formatting of

Date	Revision History	Revision Class	Comments
			the technical content.
10/08/2010	5.1	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	5.1	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	5.1	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	6.0	Major	Significantly changed the technical content.
03/25/2011	7.0	Major	Significantly changed the technical content.
05/06/2011	8.0	Major	Significantly changed the technical content.
06/17/2011	8.1	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction

This Protocol Family System Document (PFSD) is primarily intended to cover the Protocol Family as a whole. In conjunction with Member Protocol Technical Documents (TDs), which are intended to cover Member Protocols, it presents the rules for information exchange relevant to those Member Protocols and the Protocol Family that are used to interoperate or communicate with a Windows operating system in its various environments.

The Rights Management Services (RMS) system is information protection technology that works with RMS-enabled applications to help safeguard digital information from unauthorized use, online and offline, inside and outside of the firewall. RMS is designed for organizations that need to protect sensitive and proprietary information such as financial reports, product specifications, customer data, and confidential email messages. RMS can be used to help prevent sensitive information from intentionally or accidentally getting into the wrong hands.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

certificate chain
globally unique identifier (GUID)
NT LAN Manager (NTLM) Authentication Protocol
policy

The following terms are defined in [MS-RMPR]:

certificate
cloud service
consumer
content key
creator
license
offline publishing
online publishing
publishing license (PL)
rights policy template
service connection point (SCP)
security processor
use license

The following terms are specific to this document:

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. Note that in [RFC2119] terms, most of these specifications should be imperative, to ensure interoperability. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

Any specification that does not explicitly use one of these terms is mandatory, exactly as if it used MUST.

1.2 References

References to Microsoft Open Specification documents do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-ADSO] Microsoft Corporation, "Active Directory System Overview".

[MS-MWBE] Microsoft Corporation, "Microsoft Web Browser Federated Sign-On Protocol Extensions".

[MS-MWBF] Microsoft Corporation, "Microsoft Web Browser Federated Sign-On Protocol Specification".

[MS-RMPR] Microsoft Corporation, "Rights Management Services (RMS): Client-to-Server Protocol Specification".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2251] Wahl, M., Howes, T., and Kille, S., "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997, http://www.ietf.org/rfc/rfc2251.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part2-20030624

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XML1.0] Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C Recommendation, October 2000, http://www.w3.org/TR/2000/REC-xml-20001006

[XMLNS-2ED] World Wide Web Consortium, "Namespaces in XML 1.0 (Second Edition)", August 2006, http://www.w3.org/TR/2006/REC-xml-names-20060816/

[XMLSCHEMA1] Thompson, H.S., Ed., Beech, D., Ed., Maloney, M., Ed., and Mendelsohn, N., Ed., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[XPATH] Clark, J. and DeRose, S., "XML Path Language (XPath), Version 1.0", W3C Recommendation, November 1999, http://www.w3.org/TR/xpath

[XRML] ContentGuard, Inc., "XrML... eXtensible rights Markup Language", 2005, http://www.xrml.org/XrML_12.asp

If you have any trouble finding [XRML], please check here.

1.2.2 Informative References

[MS-DISO] Microsoft Corporation, "Domain Interactions System Overview".

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MS-NTHT] Microsoft Corporation, "NTLM Over HTTP Protocol Specification".

[MSDN-TaskSch] Microsoft Corporation, "Task Scheduler", http://msdn.microsoft.com/en-us/library/aa383614.aspx

2 Overview

Section 1, "Introduction", describes this Protocol Family System Document. This section introduces the system that is being documented.

2.1 System Summary

Data loss can lead to significant problems in an IT enterprise, with financial, legal, and regulatory compliance consequences - problems that can cause loss of credibility for an organization. The exchange of information has moved outside the traditional boundaries of a corporate network, including the use of mobile devices such as notebook computers, PDAs, and cellular phones. Typically, organizations secure information by using perimeter-based security methods, such as firewalls, access control lists, and encryption in transit. These methods help organizations control access to sensitive data, but authorized users are still free to do whatever they want with the information. Perimeter-based security methods simply cannot enforce business rules that control how people use and distribute the data outside the perimeter. Without an end-to-end solution in place to effectively control the use of digital information, no matter where it goes, sensitive information can too easily end up in the wrong hands, whether accidentally or maliciously.

Rights Management Services is used for restricting information access to only authorized users who are using trusted applications. RMS-protected content is encrypted and contains an associated usage **policy**, which defines the restrictions each user has when using the content. Usage policies work by assigning rights to users or groups of users. RMS defines and recognizes several rights by default - such as permission to read, copy, print, save, forward, and edit. Applications can also add custom rights.

The Rights Management Services system has three major entities: the **creator**, the **consumer**, and the server. The creator builds content and chooses an access policy for that content. When the RMS creator protects the content it is encrypted using a randomly generated **content key**. Both this key and the access policy are bound to the content in the form of a **publishing license**.

The consumer, upon receiving the document from the creator and opening it, supplies the server with the Publishing License and the consumer's identity. If the consumer is allowed access, according to the access policy in the **license**, the server issues the consumer a **use license** that specifies the access policy for the consumer and binds the content decryption key to the consumer's identity.

A client can play the role of a creator, a consumer, or both, depending on the type of implementation. The client is responsible for requesting **certificates**, licenses, and policies from the server. It is also responsible for enforcing authorization policies as they apply to protected information and encrypting or decrypting content as appropriate.

The server role in the Rights Management Services System is responsible for issuing certifications, keys, and authorization policies, and for signing these issued certificates and policies with keys it holds in escrow. It is also responsible for evaluating and issuing authorization policies based on identity credentials the client provides in protocol requests.

2.2 List of Member Protocols

The Rights Management Services (RMS) System implements the following protocol:

Rights Management Services (RMS): Client-to-Server Protocol, as specified in [MS-RMPR]. This protocol obtains and issues certificates and licenses used for creating and working with protected content.

2.3 Relevant Standards

The RMS System uses the following standards to allow interoperability with other external systems:

Hypertext Transfer Protocol, as specified in [RFC2616]. This protocol provides a standard for clients and servers to communicate. It defines how messages are formatted and transmitted, and what actions Web servers and client applications should take in response to various commands.

Extensible Markup Language, as specified in [XML1.0], [XMLNS], [XMLSCHEMA1], [XMLSCHEMA2], [XPATH]. This standard provides a format for describing structured data. This facilitates more precise declarations of content and more meaningful search results across multiple platforms.

eXtensible Rights Markup Language, as specified in [XRML]. This standard provides descriptions of usage rights and conditions for digital contents, together with message integrity and entity authentication in these descriptions.

SOAP, as specified in [SOAP1.1], [SOAP1.2/1], [SOAP1.2/2]. This standard provides a standard Internet protocol for exchanging structured information in a distributed environment.

Web Services Description Language, as specified in [WSDL]. This standard provides a general purpose XML language for describing the interface, protocol bindings, and the deployment details of network services.

Microsoft Web Browser Federated Sign-On Protocol, as specified in [MS-MWBF] and [MS-MWBE]. This standard allows integration with Active Directory Federation Services to support Federated Identities. It enables two-way collaboration between organizations when only one organization has an RMS server.

[MSDN-TaskSch] Microsoft Corporation, "Task Scheduler", http://msdn.microsoft.com/en-us/library/aa383614.aspx. This standard allows the creation of a Task Scheduler job that is used to retrieve RMS Policy templates from an RMS Server.

3 Foundation

This section describes the theoretical and practical information needed to understand this document and this system.

3.1 Background Knowledge and System-Specific Concepts

This section summarizes:

- Background knowledge required to understand this document.
- Concepts that are specific to this system, including:
- SOAP
- Cryptographic keys
- Directory services
- Federated Sign-On
- XrML
- RMS certificates and licenses

3.1.1 SOAP

SOAP is a simple XML-based protocol that enables applications to exchange information over HTTP. SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages. The Rights Management Services System uses SOAP for all client to server communications. For more information on SOAP, see [SOAP1.1], [SOAP1.2/1], and [SOAP1.2/2].

3.1.2 Cryptographic Keys

The Rights Management Services System uses both symmetric and asymmetric (also known as public-key) cryptography. Cryptography in RMS is used to protect various certificates, licenses and content to provide end users a seamless way to protect and unprotect content without knowledge of the underlying system.

Symmetric-key cryptography refers to encryption methods in which the key that was used to encrypt information is the same key that decrypts the information. In asymmetric cryptography there are two keys, a public key and a private key. The keys are mathematically related but it is not computationally feasible to determine one key with only the other. The public key may be freely distributed and is generally used to encrypt data. The private key is kept secret and is generally used for decrypting data.

3.1.3 Directory Services

RMS uses directory services, such as Active Directory, as a central repository for storing and retrieving identity and account information about RMS users and to enable the RMS client to discover the RMS server. The RMS System does not require Active Directory to be the directory service to be used. In scenarios where Active Directory is used as the directory service and the RMS server has joined a domain, the domain serves as the primary source of identity for the RMS server and RMS users. The domain, through the relevant security protocols, provides the basis for authentication within the domain, allowing principals within the domain to establish authenticated

connections with each other. Once authenticated, the domain provides authorization information in the form of additional identities representing groups, allowing authorization decisions to be made. RMS identifies users by e-mail addresses, which are stored in the directory. In scenarios where other directory services are used, separate authentication mechanisms may be used, such as anonymous authentication. See [MS-DISO] and [MS-ADSO] for more information on the Domain Interactions System and System and Active Directory respectively.

3.1.4 Federated Sign-On

Federated Sign-On allows enterprises to establish relationships for exchanging protected content with entities outside their directory infrastructure. This scenario is most suitable when one enterprise has RMS infrastructure and the other does not. For more information on Federated Sign-On see [MS-MWBE] and [MS-MWBF].

3.1.5 XrML

The eXtensible rights Markup Language (XrML) is a general-purpose, XML-based specification grammar for expressing rights and conditions associated with digital content, services, or any digital resource.

The RMS technology uses an XML vocabulary to express digital rights, the eXtensible rights Markup Language (XrML), version 1.2.1. XrML specifies a rights-expression language that trusted systems that are in a trusted environment can use to express digital information policies. You can apply XrML licenses to trusted information that is in any format, such as e-mail, office productivity tools, database contents, e-commerce downloads, line-of-business programs, and customer relationship management systems, to name a few. You can then enforce XrML licenses through any trusted rights management system that uses the XrML standard.

The rights to be managed are expressed in an XrML publishing license that is associated with the protected information. The publishing license is an expression of how the information owner wants it to be used, protected, or distributed. The publishing license and the user's identity are passed to the rights management services system, which builds a use license.

These licenses are easily interpreted and managed by various interoperable rights management systems because they all use the XrML standard. Managing information online by using licenses provides ease-of-access from any location. After the use license is downloaded, a client implementation will be able to access protected information both online and offline.

XrML supports an extensive list of rights. In addition, applications can define additional rights to meet particular needs. By defining additional rights, enterprises can build many business, usage, and workflow models to meet their specific requirements. For more information on XrML see [XRML].

3.1.6 RMS Certificates and Licenses

RMS uses specific XrML certificates to identify and trust different entities in the system. Licenses are also XrML certificates but are used to specify rights and conditions that govern content use. The following sections detail the certificates and licenses that are used by the RMS System.

3.1.6.1 Server Licensor Certificate

The Server Licensor Certificate (SLC) is the identity of the RMS Server and grants the role of a server that can issue certificates and licenses for working with protected content. The SLC grants the right to issue:

- Publishing licenses
- Use licenses
- Client licensor certificates
- Rights policy templates
- Rights account certificates

3.1.6.2 Security Processor Certificate

The Security Processor Certificate (SPC) is generated during activation and contains the public key corresponding to the SPC private key. The SPC represents the identity of a computer that can be used for working with protected content.

3.1.6.3 RMS Account Certificate

The RMS Account Certificate (RAC) represents the identity of a user who can access protected content.

3.1.6.4 Client Licensor Certificate

The Client Licensor Certificate (CLC) grants the role of a user who can publish protected content offline.

3.1.6.5 Publishing License

The publishing license (PL) defines usage policy for protected content and contains the content key with which that content is encrypted. The usage policy identifies all authorized users and the actions they are authorized to take with the content, along with any conditions on that usage. The publishing license tells the server what usage policies apply to a given piece of content and grants the server the right to issue use licenses based on that policy. The PL is created when content is protected.

3.1.6.6 Use License

The Use License (UL) authorizes a user to access a given protected content file and describes the usage policies that apply. The UL contains the symmetric content key for decrypting the content.

3.2 System Purposes

The Rights Management Services System provides the ability to secure information and restrict access to authorized users. It also provides the ability to enforce access policies and restrict information access to trusted applications. RMS provides an administration interface to manage the system and create access policies.

The purpose of the RMS System is to:

- Provide a protection mechanism to ensure that data, such as e-mail and documents, can be protected and consumed only by the intended recipients.
- Issue the requisite certificates necessary for securing and consuming data.
- Store and manage the certificates previously mentioned.

- Store and manage rights policy templates (also known as "templates"), which provide pre-determined policies for secured data.
- Log requests made to the RMS server.

Creators use the system for the following purposes:

- Securing data, which includes the access policies for the data.
- Confirming content consumer identity and supplying the consumer with decryption keys and access rights.

Consumers use the system for the following purpose:

- Obtaining licenses to access protected data, which includes the decryption key for the content and usage restrictions.

3.3 System Use Cases

This section specifies the major use cases of the Rights Management Services System and the rationale for their use.

3.3.1 Stakeholders and Interests Summary

The stakeholders in the RMS System are users, computers, applications, servers, and a service.

RMS User: A person who uses an RMS Client Application. The primary interests of an RMS User are to be able to protect and consume protected content.

Client Computer: A computer or device, such as a mobile phone, that hosts an RMS Client Application.

RMS Client Application: An application that acts as a client to an RMS Server. The application can be an end user-based or server-based application, and can perform RMS functions such as protecting content and providing access to protected content.

RMS Server: The component that provides RMS services such as issuing certificates and licenses.

RMS Administrator: A person who performs RMS administration in the enterprise and typically has full access to RMS Servers. The interests of the RMS Administrator are to enroll the RMS Server with the RMS Cloud Service for use in the system and configure the **service connection point (SCP)** in Active Directory.

RMS Cloud Service: A Web service run by Microsoft, available on the Internet, which provides enrollment services to RMS servers.<1>

Active Directory: The directory service that provides authentication and user identity services to the RMS Server. Active Directory also stores the SCP for use by RMS Client Applications and the RMS Server.

3.3.2 Supporting Actors and System Interests Summary

There are no other systems in which the RMS System is an actor.

3.3.3 Use Case Diagrams

The following table provides an overview for the groups of use cases which span the functionality of the Rights Management Services System. The sections which follow provide detailed descriptions of the use cases in each group. Each use case is described in detail in section 3.3.4.

Use case group	Use cases
Prepare RMS clients and servers to participate in the RMS system	Enroll RMS Server - RMS Server Configure SCP - RMS Administrator Bootstrap RMS Client - RMS Client Application
Protect content using RMS, consume content that has been protected using RMS	Acquire Templates - RMS Client Application Publish Protected Content Online - RMS Client Application Publish Protected Content Offline - RMS Client Application Consume Protected Content - RMS Client Application

The following diagrams illustrate the use cases in each use case group.

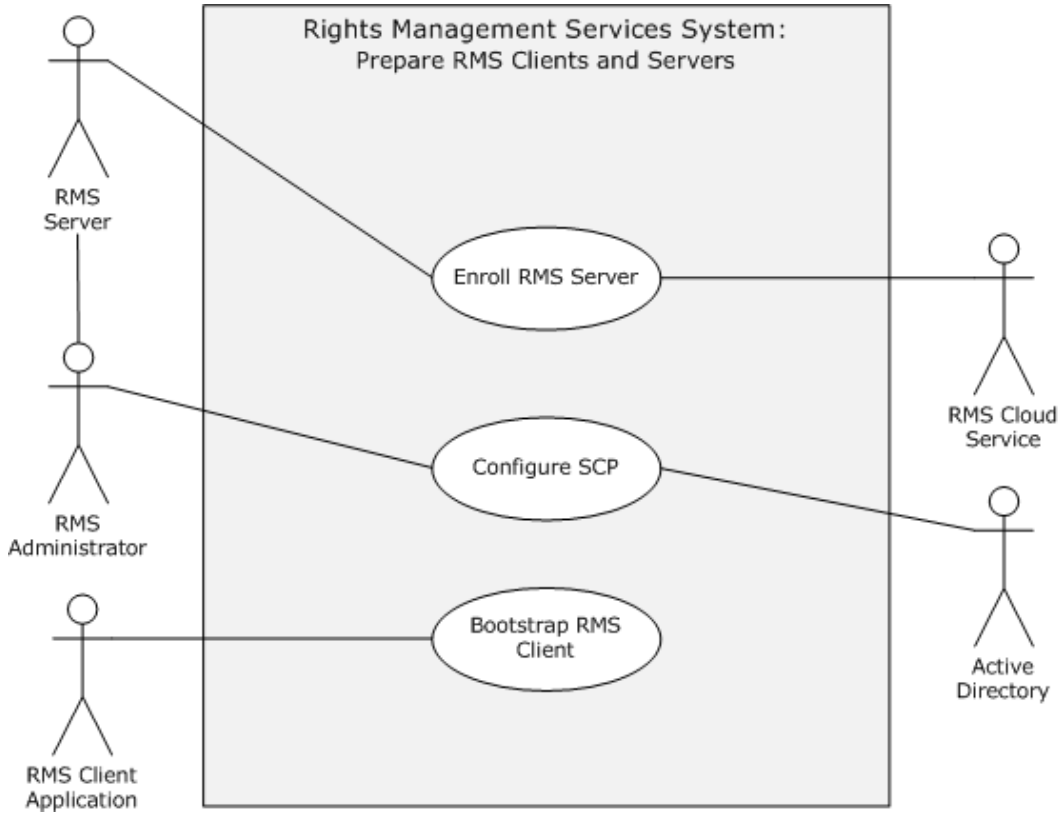

Figure 1: Preparing RMS clients and servers to participate in the RMS System

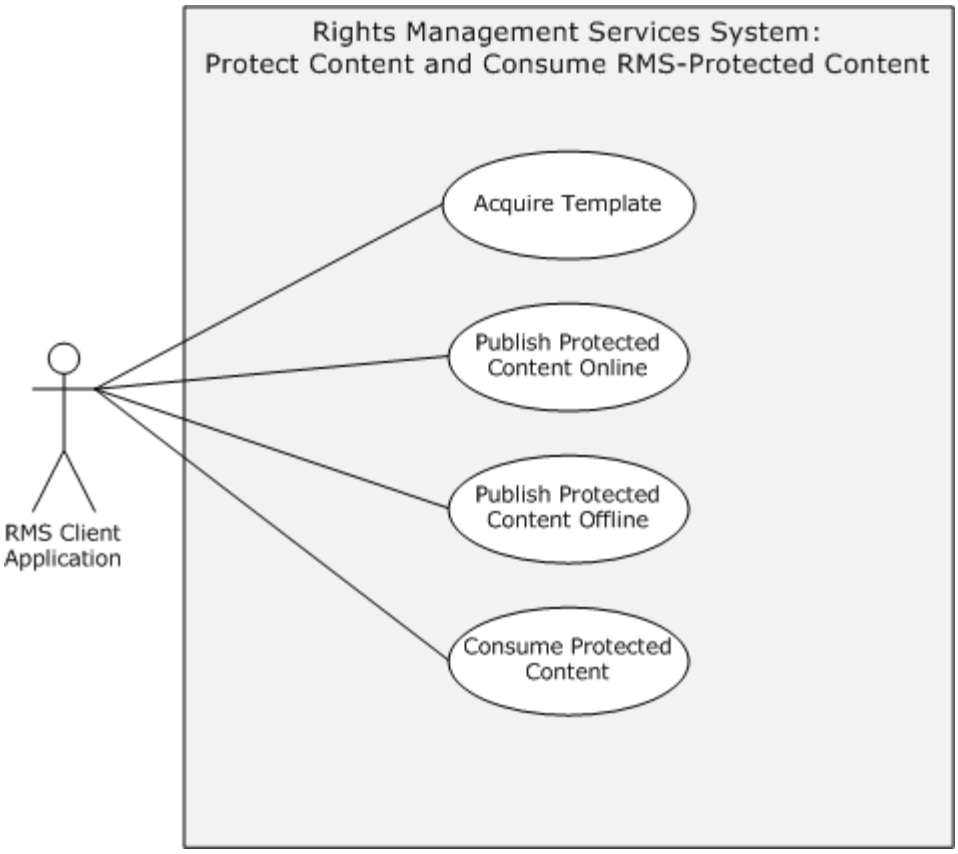

Figure 2: Protect content using RMS, consume content that has been protected using RMS

3.3.4 Use Case Descriptions

3.3.4.1 Enroll RMS Server - RMS Server

Goal: Enroll the server with the Microsoft Cloud Service so clients trust the server and send it requests.

Context of Use: An RMS Server has to perform enrollment before servicing any client requests. Servers perform enrollment by generating an enrollment request and sending it to the RMS Cloud Service. Server enrollment requests contain the public portion of the RMS Server's key pair and other enrollment information. Server enrollment requests can be made synchronously by the server directly contacting the RMS Cloud Service, or asynchronously by an RMS Administrator exporting the enrollment request and contacting the RMS Cloud Service from another computer.

Direct Actor: The direct actor of this use case is the RMS Server.

Primary Actor: The primary actor is the RMS Administrator.

Supporting Actors: The supporting actor is the RMS Cloud Service.

Stakeholders and Interests:

- RMS Server, as described in section 3.3.1

- RMS Administrator, as described in section 3.3.1
- RMS Cloud Service, as described in section 3.3.1

Preconditions:

- The server generates an asymmetric key pair for the certificate that will represent the server's identity.

Minimal Guarantees: If a properly formatted enrollment request is sent to the RMS Cloud Service, a server certificate will be generated, signed, and appended to the server enrollment **certificate chain**.

Success Guarantee: The minimal guarantee is the same as the success guarantee.

Trigger: The Administrator triggers this use case after RMS is installed on a server and when the server's certificate needs to be renewed.

Main Success Scenario:

The RMS Server makes an enrollment request to the RMS Cloud Service.<2>

The RMS Cloud Service returns a signed server certificate and the server enrollment certificate chain, which is then used by the server.

Extensions:

1a. Enrollment requests can also be made asynchronously by an RMS Administrator exporting the enrollment request and sending it to the RMS Cloud Service from another computer.

3.3.4.2 Configure SCP - RMS Administrator

Goal: Set a service connection point in Active Directory so that clients can locate the RMS Server.

Context of Use: In order for an RMS Client Application to contact an RMS Server it needs to have the URL of the server. As an alternative to configuring each client, the RMS Administrator can set an SCP in Active Directory. The clients can search for this SCP when they need the URL of an RMS Server.

Direct Actor: The direct actor of this use case is the RMS Administrator.

Primary Actor: The primary actor is the same as the direct actor.

Supporting Actors: The supporting actor is Active Directory.

Stakeholders and Interests:

- RMS Server, as described in section 3.3.1.
- RMS Administrator, as described in section 3.3.1.
- RMS Client Application, as described in section 3.3.1.

Preconditions: None.

Minimal Guarantees: None.

Success Guarantee: The URL of the RMS Server is specified in the service connection point in Active Directory.

Trigger: The Administrator triggers this use case after RMS is installed on a server.

Main Success Scenario:

- The RMS Administrator connects and authenticates to Active Directory.
- The RMS Administrator creates a serviceConnectionPoint in Active Directory containing the URL of the RMS Server.

Extensions: None.

3.3.4.3 Bootstrap RMS Client - RMS Client Application

Goal: Prepare an RMS Client Application to participate in the RMS System.

Context of Use: Client bootstrapping is a set of initialization steps that clients complete before either performing **offline publishing** or consuming content. During client bootstrapping, the Client Computer and RMS User are configured to participate in the RMS System. This involves various encryption key and certificate generations and exchanges.

Direct Actor: The direct actor of this use case is the RMS Client Application.

Primary Actor: The primary actor is the same as the direct actor.

Supporting Actors: None.

Stakeholders and Interests:

- RMS Client Application, as described in section 3.3.1
- RMS User, as described in section 3.3.1
- Client Computer, as described in section 3.3.1
- RMS Server, as described in section 3.3.1

Preconditions: None.

Minimal Guarantees: The RMS User and Client Computer are uniquely identified and receive the appropriate certificates to consume protected content.

Success Guarantee: The minimal guarantee, plus receiving the Client Licensor Certificate (CLC), which grants the ability to publish content.

Trigger: Typically this use case is triggered by an RMS Client Application needing to protect or consume protected content for the first time. This can be initiated by an RMS User using an RMS Client Application or by automation in an RMS Client Application.

Main Success Scenario:

The RMS Client Application generates a Security Processor Certificate (SPC) that is Client Computer-specific.

The RMS Client Application sends the SPC to the RMS Server and requests the user's RMS Account Certificate (RAC).

The RMS Server validates the SPC and identity of the user and sends the RMS Client Application the RAC.

To publish offline, the user needs to have a separate signing certificate that is bound to the user's identity in RMS. The client first finds the user's service location and then sends a request to the appropriate RMS Server to retrieve the CLC.

Extensions: None.

3.3.4.4 Acquire Templates - RMS Client Application

Goal: Retrieve the rights policy templates published by the RMS Server for use in publishing protected content.

Context of Use: Rights policy templates contain a pre-determined access policy that can be used by an RMS Client Application to assign rights when publishing protected content. Rights policy templates are published on the RMS Server and, in order to be used have to be retrieved by the RMS Client Application.<3>

Direct Actor: The direct actor of this use case is the RMS Client Application.

Primary Actor: In most cases the primary actor is the same as the direct actor. It is possible though that an RMS Client Application could allow an RMS User the ability to trigger this use case, making the RMS User the primary actor.

Supporting Actors: None.

Stakeholders and Interests:

- RMS User, as described in section 3.3.1
- RMS Client Application, as described in section 3.3.1
- RMS Server, as described in section 3.3.1
- Client Computer, as described in section 3.3.1

Preconditions:

- The RMS Client Application needs to be able to determine the location of the RMS Server by the means specified in [MS-RMPR] section 3.8.3.2.

Minimal Guarantees: The RMS Server returns the list of rights policy templates and the individual rights policy template when requested. The client SHOULD store individual templates in a local license store.

Success Guarantee: The success guarantee is the same as the minimal guarantee.

Trigger: This scenario can be triggered at any time, but generally an RMS Client Application triggers this use case at regular intervals to ensure it has the latest rights policy templates. An RMS Client Application may also allow an RMS User to trigger this use case.

Main Success Scenario:

An RMS Client Application requests the list of templates available from the RMS Server.

The RMS Client Application makes subsequent requests to the server for individual rights policy templates. The client SHOULD place the templates from the server in a local license store.

Extensions: None.

3.3.4.5 Publish Protected Content Online - RMS Client Application

Goal: Publish protected content by communicating directly with the RMS Server.

Context of Use: **Online publishing** allows publishing content by acquiring the public portion of the RMS Server's SLC, generating a publishing license (PL) for the content, and sending that license to the server to be signed. Online publishing does not require client bootstrapping.

Direct Actor: The direct actor of this use case is the RMS Client Application.

Primary Actor: In an end-user client application the primary actor is the RMS User. The primary actor can also be the RMS Client Application, for example in the case of server or automated applications.

Supporting Actors: None.

Stakeholders and Interests:

- RMS User, as described in section 3.3.1
- RMS Client Application, as described in section 3.3.1
- RMS Server, as described in section 3.3.1

Preconditions:

- If rights policy templates are used to protect the content they need to have already been retrieved per section 3.3.4.
- The RMS Client Application needs to be able to determine the location of the RMS Server by the means specified in [MS-RMPR] section 3.8.3.2.

Minimal Guarantees: The RMS Client Application receives a signed PL for the content.

Success Guarantee: The minimal guarantee, plus the PL contains the location of the RMS Server to retrieve a use license to consume the content. The PL allows the intended recipient to consume the content.

Trigger: An attempt to protect content using RMS, which can be trigged by an RMS User using an RMS Client Application, or by automation in an RMS Client Application.

Main Success Scenario:

The RMS Client Application sends a request to the RMS Server to retrieve the public portion of the server's SLC.

The application generates a PL for the content being protected, which contains the content key and usage policy.

The application encrypts content key and usage policy in the PL using the SLC public key.

The PL is sent to the RMS Server to be signed.

The RMS Server returns the signed PL for the content.

Extensions: None.

3.3.4.6 Publish Protected Content Offline - RMS Client Application

Goal: Protect content without making calls to an RMS Server.

Context of Use: Offline publishing gives an RMS Client Application the ability to publish content without making calls to the RMS Server. Unlike online publishing, client bootstrapping is required for offline publishing.

Direct Actor: The direct actor of this use case is the RMS Client Application.

Primary Actor: In an end user client application the primary actor is the RMS User. The primary actor can also be the RMS Client Application, for example in the case of server or automated applications.

Supporting Actors: None.

Stakeholders and Interests:

- RMS Client Application, as described in section 3.3.1
- RMS User, as described in section 3.3.1

Preconditions:

- The client needs to be bootstrapped with the RMS Server, per section 3.3.4.3, including having a CLC.
- If a template is used to publish offline, the client verifies that the key used to sign the template matches the key in the certificate used for publishing content.

Minimal Guarantees: The RMS Client Application protects the content and generates a Publishing License for it.

Success Guarantee: The minimal guarantee, plus the Publishing License contains the URL of the RMS Server that can issue a Use License for the content. The license for the protected content allows the intended recipient to consume the content.

Trigger: An attempt to protect content using RMS, which can be trigged by an RMS User using an RMS Client Application, or by automation in an RMS Client Application.

Main Success Scenario:

The RMS Client Application protects the content and generates a Publishing License for the protected content, which contains the usage policy and the content key.

The usage policy and content key in the license are encrypted using the server's SLC public key, which was retrieved from the CLC.

The license for the content is signed using the CLC.

Extensions: None.

3.3.4.7 Consume Protected Content - RMS Client Application

Goal: Remove protection from and consume content protected by RMS.

Context of Use: Upon receiving protected content, the RMS Client Application submits the Publishing License, RAC and application data to the server. The server validates the data received and returns a Use License that is used to unprotect the content for consumption.

Direct Actor: The direct actor of this use case is the RMS Client Application.

Primary Actor: In an end user client application the primary actor is the RMS User. The primary actor can also be the RMS Client Application, for example in the case of server or automated applications.

Supporting Actors: None.

Stakeholders and Interests:

- RMS User, as described in section 3.3.1
- RMS Client Application, as described in section 3.3.1
- RMS Server, as described in section 3.3.1

Preconditions:

- The client needs to be bootstrapped with the RMS Server, per section 3.3.4.3.
- The RMS Client Application needs to be able to determine the location of the RMS Server by the means specified in [MS-RMPR] section 3.8.3.2.

Minimal Guarantees: The RMS Server will evaluate the Publishing License, RAC, and application data. If it trusts all three it will return the appropriate license; if it does not trust all three it will return the appropriate fault code.

Success Guarantee: The RMS Client Application receives the license to consume the content and is able to remove the protection.

Trigger: An attempt is made to access RMS protected content using an RMS Client Application. This can be trigged by an RMS User using an RMS aware application, or by automation in an RMS Client Application.

Main Success Scenario:

The RMS Client Application sends the Publishing License that came with the content, the RAC and application data to the RMS Server.

The RMS server validates the data received. If it trusts the PL and RAC the server generates and sends a Use License for the RMS Client Application to consume the content.

The RMS Client Application receives the license to consume the content, which contains the protected symmetric content key and usage policies.

Extensions: None.

4 System Context

This section describes the relationship between this system and its environment.

4.1 System Environment

The Rights Management Services System requires network connectivity between clients and servers using the HTTP protocol, typically over port 80. Optionally HTTPS can be used, typically over port 443. The network needs to have Domain Name Services (DNS) with the RMS server registered in DNS. Clients may be located on private networks, such as an enterprise network managed by an IT department, or may communicate over the Internet.

An RMS client can be any device with the capability to connect to an RMS server using HTTP. Clients need to have the ability to persistently store certificates provided by the RMS server.

An RMS server MUST be a Web server capable of communicating with clients over HTTP.

4.2 System Assumptions and Preconditions

Given the environment described in section 4.1, the system has the following assumptions and preconditions:

- The RMS server requires a database and stored procedures to perform operations.
- The RMS server MUST have a directory available to authenticate users and retrieve the email address of user accounts. All user accounts and groups who use RMS to consume and publish content MUST have an email address that is configured in the directory.
- DNS registration is configured for the RMS servers.
- Each RMS Web service supports SOAP [SOAP1.1] over HTTP [RFC2616] over TCP/IP.

Member protocols supported by the system, as listed in section 2.2, may have additional assumptions and preconditions when that protocol is being used. Please see the relevant member protocol specification for details.

4.3 System Relationships

This section describes the relationships between the system and external components, system dependencies, and other systems influenced by this system.

4.3.1 Black Box Relationship Diagram

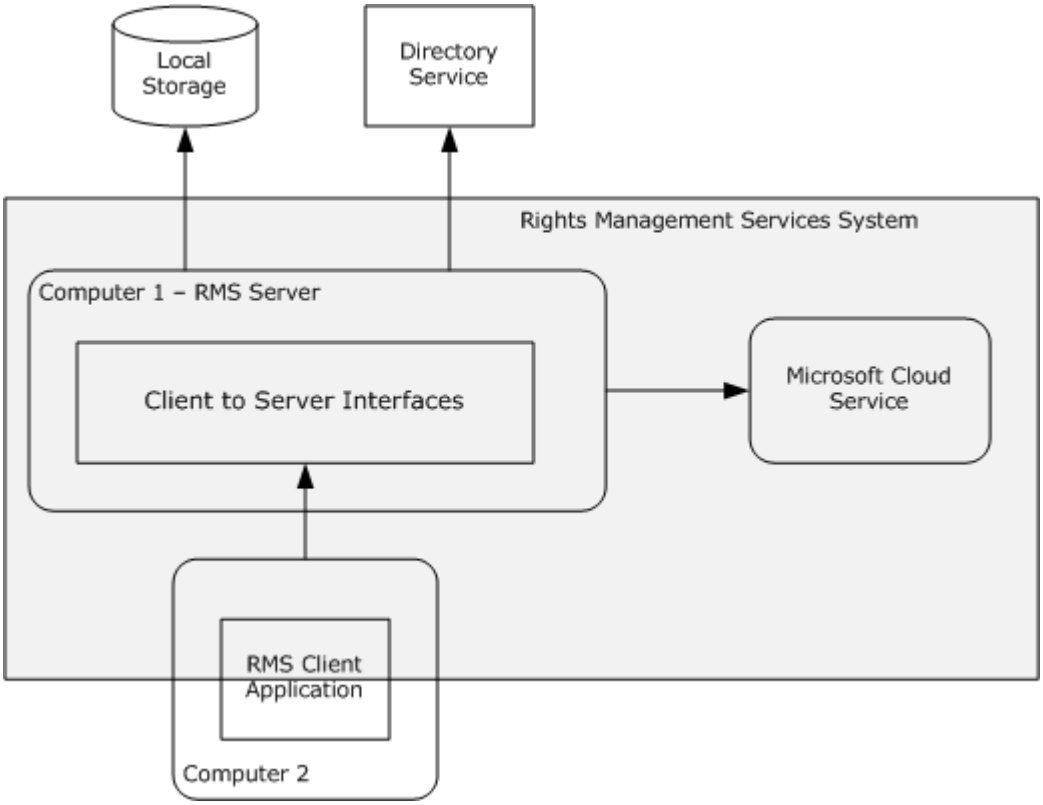

Figure 3: RMS black box diagram

As shown in the diagram, the RMS Server exposes a set of interfaces that RMS Client Applications contact to perform RMS actions. The RMS Server also sends requests to the Microsoft Cloud Service for enrollment of RMS servers.

The RMS server uses a local storage to maintain the RMS configuration, logging, and directory services databases. Additionally, certificates such as RMS Account Certificates are stored in the local storage. A directory service is also required to provide user authentication and other directory services.

4.3.2 System Dependencies

The RMS System depends on networking components and directory services to function.

The RMS System uses the SOAP messaging protocol, as specified in [SOAP1.1], for formatting requests and responses. It transmits these messages using the HTTP and/or HTTPS protocols. SOAP is considered the wire format used for messaging and HTTP and HTTPS are the underlying transport protocols. This requires that clients and servers have network connectivity and are properly configured to use TCP/IP. There is no specific requirement for the type of physical networking topology.

The system depends on the availability of a directory service to authenticate requests to the server (performed by the Web server) and provide the email address of the requesting user. The RMS

System uses the user's email address as a canonical identifier when specifying identities, rights, and policies.

4.3.3 System Influences

This system does not influence any other systems.

4.4 System Applicability

The RMS System is used to manage user access to protected information, such as documents, email, and files.

4.5 System Versioning and Capability Negotiation

The RMS System has client and server protocol versions 1.0, 1.0 SP1, 1.0 SP2, and 2.0.

The member protocols of the RMS System support limited capability negotiation via the VersionData type that is present on all protocol requests. On a request, the VersionData structure contains a MinimumVersion and MaximumVersion value indicating the range of versions the client is capable of understanding. On a response, the VersionData structure contains a MinimumVersion and MaximumVersion that the server is capable of understanding.

4.6 System Vendor-Extensible Fields

This system does not contain any vendor-extensible fields. All XML schema are considered nonextensible in the system protocols.

5 System Architecture

This section describes the basic structure of the system and the interrelationships among its parts, consumers, and dependencies.

5.1 Abstract Data Model

This section describes the conceptual data organization the system maintains to provide its functionality. The data model described in this section can be implemented in a variety of ways. This specification does not prescribe any specific implementation technique.

The Rights Management Services System uses two separate abstract data models, the Client Details ADM and Server Details ADM. The Client Details ADM contains the certificates and licenses required for RMS client applications to protect and consume content. The Server Details ADM maintains the certificates and data needed to service client requests and issue certificates and licenses to clients.

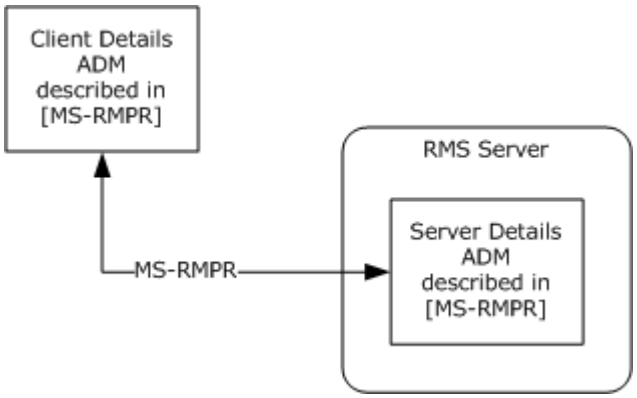

Figure 4: Abstract data model

5.1.1 Client Details ADM

The Client Details ADM in [MS-RMPR] section 3.8.1 describes the certificates and licenses a client stores in order to properly use the RMS System. The certificates and licenses all have chains that lead back to a CA certificate the system trusts. Descriptions of the licenses can be found in section 3.1.6 of this document.

5.1.2 Server Details ADM

The RMS Server ADM consists of the Server Licensor Certificate, which represents the server's identity, and a list of trusted security processor CA keys. The server SHOULD maintain a list of trusted SPC issuer keys so that it can determine whether to authorize client requests that involve a given SPC chain. See the Server Details ADM in [MS-RMPR] section 3.1.1.

5.2 White Box Relationships

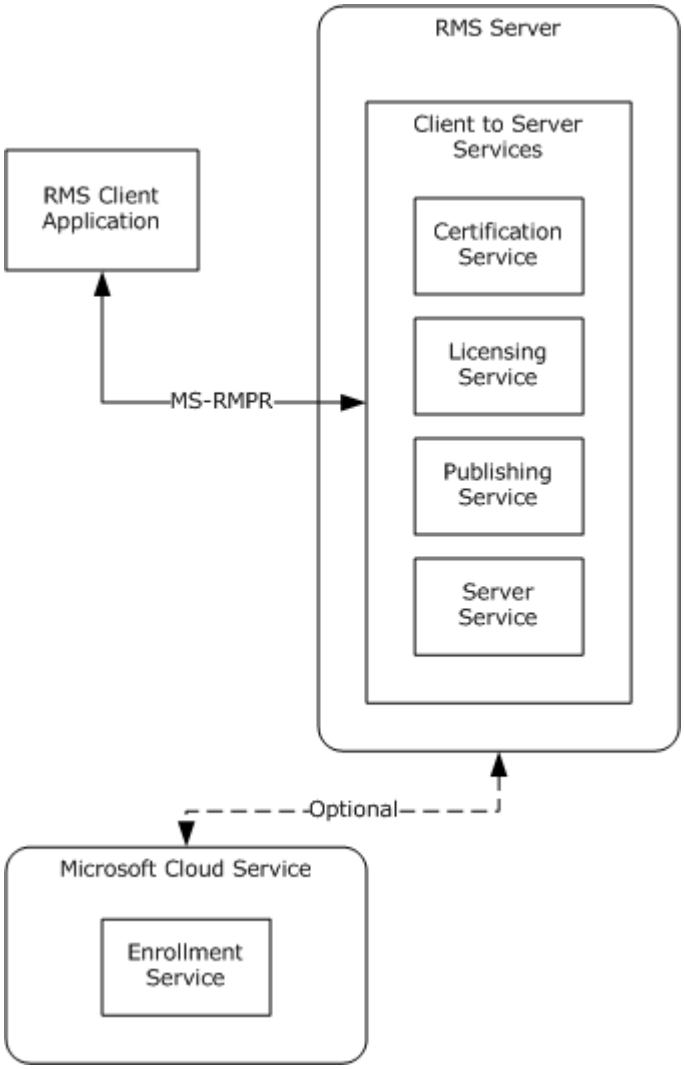

Figure 5: RMS white box diagram

The RMS System has one main service, the client to server service. The client to server services are the individual interfaces that RMS client applications call to perform RMS tasks such as bootstrapping, retrieving templates, publishing content, and licensing content.

5.3 Member Protocol Functional Relationships

5.3.1 Member Protocol Roles

The RMS Client to Server Protocol [MS-RMPR] is used for all client and server communication in relation to bootstrapping a client, consuming protected content, or protecting content.

5.3.2 Member Protocol Groups

The RMS System contains one protocol and no further grouping can be made.

5.4 System Internal Architecture

The Rights Management Services System uses the SOAP messaging protocol for all communication. The following sections detail the RMS client to server interfaces. Refer to [MS-RMPR] for full details on implementation of these interfaces.

5.4.1 Certification Service

The certification service provides RMS client applications the ability to bootstrap and receive the necessary certificates to participate in the RMS System. The Certification Service is the interface RMS client applications use in the use case Bootstrap RMS Client - RMS Client Application detailed in section 3.3.4.3.

5.4.1.1 Certify

As part of the bootstrapping process, an RMS client makes a request to the Certify Web method, which includes the client's Security Processor Certificate. The server validates the data it receives and if authorized returns the user's Rights Account Certificate. Full details of the Certification Service schema can be found in [MS-RMPR] section 3.3. Details of the Certify method can be found in [MS-RMPR] section 3.3.4.1.

5.4.2 Licensing Service

The licensing Web service provides two functions, issuing Use Licenses for protected content and providing rights policy templates to client applications that request them. Full details of the Licensing Service schema can be found in [MS-RMPR] section 3.4.

5.4.2.1 AcquireLicense

To consume protected content the client needs to acquire a Use License, which gives the user access to the content and includes the key to decrypt the content and the usage policies for the content. To acquire the Use License the client uses the AcquireLicense method, sending their RAC, Publishing License, and application data to the RMS server. After validating the RAC and PL in the request, the server returns the Use License for the content. Details of the AcquireLicense method can be found in [MS-RMPR] section 3.4.4.1.

5.4.2.2 AcquireTemplateInformation

To retrieve rights policy templates the client first makes a request to the AcquireTemplateInformation method. The server returns information about the available templates in the form of a list of GUIDs and hashes corresponding to the server templates. Details of the AcquireTemplateInformation method can be found in [MS-RMPR] section 3.4.4.2.

5.4.2.3 AcquireTemplates

After receiving the list of templates, the client determines which templates it needs to download from the server. The client uses the AcquireTemplates method with a list of rights policy template GUIDs and request templates corresponding to these GUIDs. Upon receiving the request the server will return the templates to the client. Details of the AcquireTemplates method can be found in [MS-RMPR] section 3.4.4.3.

5.4.3 Publishing Service

The Publishing Service has two functions, to sign Publishing Licenses generated during online publishing and to provide Client Licensor Certificates (CLCs), which are used in offline publishing. Full details of the Publishing Service schema can be found in [MS-RMPR] section 3.5.

5.4.3.1 AcquireIssuanceLicense

During online publishing, the RMS client application gets the Server Licensor Certificate (SLC) chain from the server (see details on GetLicensorCertificate in section 5.4.4.2), generates a symmetric content key and generates usage restrictions. The RMS Client Application generates a Publishing License (PL) which includes the content key and use restrictions. The content key and use restrictions are encrypted with the server's public key (from the SLC).

In online publishing, after the PL is created it MUST be signed by the server. The AcquireIssuanceLicense request is used to sign a PL during online publishing. The RMS client application makes an AcquireIssuanceLicense request to the server with the unsigned PL. The server signs the body of the PL and returns it to the RMS client application. Details of AcquireIssuanceLicense can be found in [MS-RMPR] section 3.5.4.1.

5.4.3.2 GetClientLicensorCert

During offline publishing the RMS client application uses the CLC to publish content without contacting the RMS server. The CLC contains the server's public key, which is used to encrypt the content key and use restrictions in the PL. The CLC private key is used to sign the body of the PL.

The GetClientLicensorCert method is used to obtain the CLC for the user. The user MUST already have a RAC and SPC to use this method. In the GetClientLicensorCert request the client submits a RAC chain and requests a CLC chain. When the server receives the request it performs signature validation on the RAC chain and verifies that it trusts the RAC. The server generates a CLC which contains a unique asymmetric key pair. The server encrypts the CLC private key with the public key of the RAC, so the RAC and SPC are required to access the signing key in the CLC. Details of GetClientLicensorCert can be found in [MS-RMPR] section 3.5.4.2.

5.4.4 Server Service

The Server Service has two purposes, to provide the service location for users and provide the Server Licensor Certificate (SLC) for use in online publishing. Full details of the Server Service schema can be found in [MS-RMPR] section 3.7.

5.4.4.1 FindServiceLocationsForUser

Depending on the RMS server configuration, different servers can be used for different functions for a given user. The client SHOULD use the FindServiceLocationsForUser request to discover the appropriate server for various services for a given user. The server uses the GetAuthenticatedAccount abstract interface to determine the authenticated domain account. When servicing FindServiceLocationsForUser requests, Microsoft Windows® implementations of the RMS server use the GetAuthenticatedAccount abstract interface to retrieve the domain account, which is authenticated using **NT LAN Manager (NTLM) Authentication Protocol** through Internet Information Services (IIS), as per [MS-NTHT]. The SOAP request does not encapsulate the authentication.

When the RMS client application makes a FindServiceLocationsForUser request, it includes a service type and requests its location. Given the authenticated domain account and the requested server type, the server determines the service location using the GetDirectoryForAccount and

GetServiceLocationForDirectory abstract interfaces and returns the URL to the RMS client application. Details of FindServiceLocationsForUser can be found in [MS-RMPR] section 3.7.4.2.

5.4.4.2 GetLicensorCertificate

During online publishing the RMS client application retrieves the SLC from the RMS server. The SLC public key is used to encrypt the symmetric content key and use restrictions in the publishing license.

The RMS client application makes a GetLicensorCertificate request to the server; no information is sent in the request. Upon receiving the request the server returns its SLC chain. Details of GetLicensorCertificate can be found in [MS-RMPR] section 3.5.4.2.

5.5 Failure Scenarios

The protocols of the RMS System are SOAP-based protocols that use HTTP 1.1 for transport. The protocols allow a server to notify a client of application-level faults by generating SOAP faults as specified in [SOAP1.1] section 4.4. In the SOAP fault, the <faultcode> element contains the type of exception being thrown. The <faultstring> element contains the text of the exception being thrown. For more information about the SOAP faults from the RMS server see [MS-RMPR] section 3.1.4.5.

Some common areas of failure are:

- Certificate or license format or arguments are invalid.
- Certificate or license has expired.
- Access is unauthorized.
- Email address is invalid or formatted incorrectly.
- Machine certificate (SPC) is not valid.
- A certificate or license has been tampered with (signature validation failed).
- VersionData element contains an invalid or unsupported version number.
- The user does not have rights in the publishing license.
- The certificate or license did not come from a trusted issuer.

The RMS System protocols will not be able to function if there are failures on the services it depends on, such as network connectivity, availability of DNS, or availability of a directory service.

6 System Details

This section contains the details that complete the descriptions in earlier sections of the document. These details are needed to understand and implement this system.

6.1 Architectural Details

This section provides a series of examples illustrating the use of the Rights Management Services System. The examples are:

- Protecting content using offline publishing
- Protecting content using online publishing
- Consuming protected content

Each example is described in detail in the following sections as a set of conceptual actions interchanged between the system and participants. These actions correspond to the interfaces described in section 5.4.

6.1.1 Protecting Content Using Offline Publishing

One of the most common tasks in RMS is publishing protected content using an RMS client application. This section describes the typical steps that can be completed for a user and computer that has not used RMS in the past to bootstrap the client, acquire a Client Licensor Certificate (CLC), acquire templates and, finally, publish the content offline. Note that templates are an optional component of RMS but are included in this example.

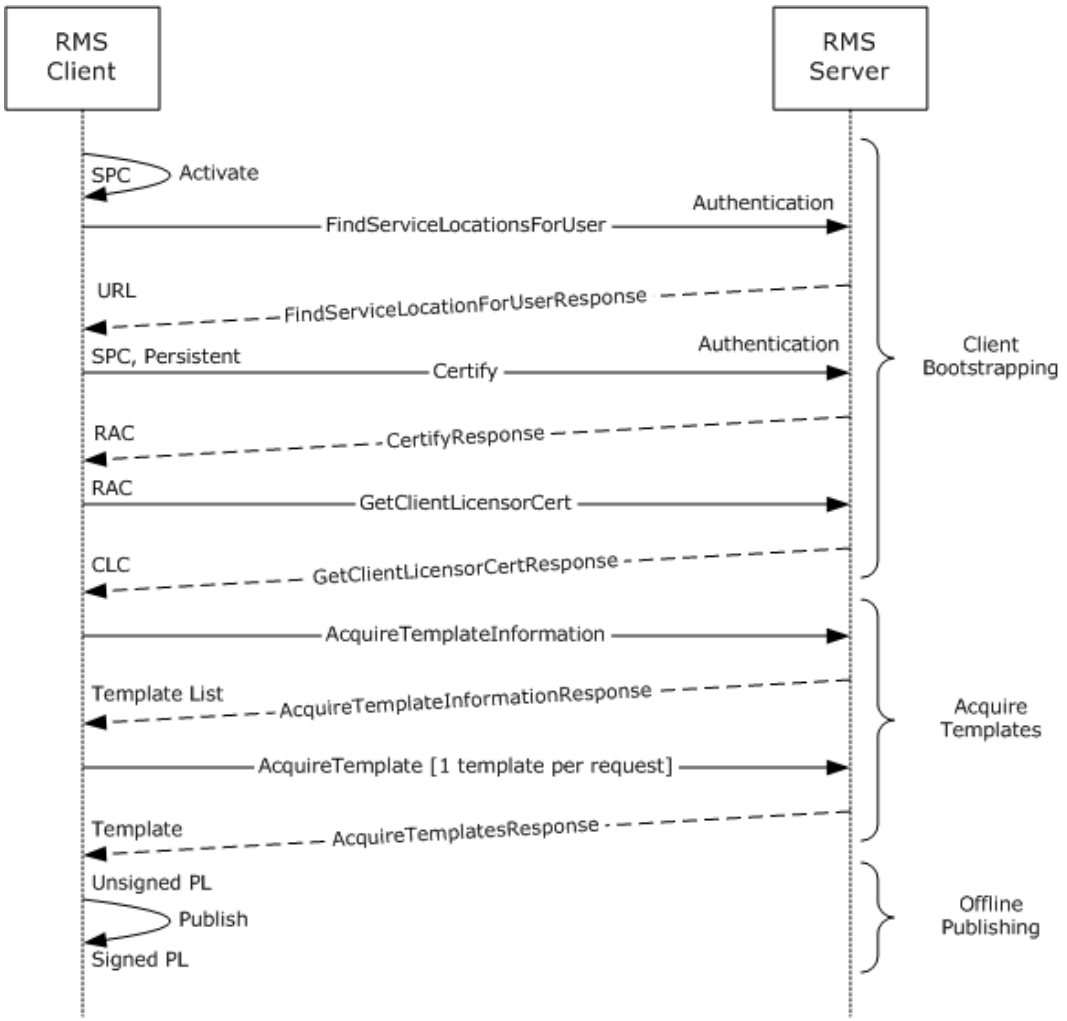

Figure 6: Message flow for protecting content using offline publishing

6.1.1.1 Client Bootstrapping

6.1.1.1.1 Activate the Computer

Before a computer or device can participate in the RMS System it MUST generate a Security Processor Certificate (SPC). In RMS protocol versions 1.0 SP1 and later, clients self-activate, generating their own SPC.<4> More details on activation can be found in [MS-RMPR] section 3.2.4.1.

6.1.1.1.2 Find Service Locations

Depending on the deployment topology of the servers in the network, different servers can be used for different functions for a given user. The client SHOULD make a FindServiceLocationsForUser request to the RMS server, and in return receive the URL of the server to make subsequent calls to for client bootstrapping. Details about the FindServiceLocationsForUser method can be found in [MS-RMPR] section 3.7.4.2.

6.1.1.1.3 Certify the User

After the computer is activated the user MUST be certified to participate in the RMS system. This is accomplished by binding an encryption key pair to both the user and the client computer by way of a RAC. The user MUST have a RAC to access protected content or to publish protected content offline. Full details of client certification can be found in [MS-RMPR] section 3.3.4.1.

6.1.1.1.4 Acquire a CLC

To publish offline, the user MUST have a signing key pair. The Client Licensor Certificate (CLC) binds a signing key pair to the user through their Rights Account Certificate (RAC). The CLC private key is encrypted by the RAC public key, the SPC and RAC are required to decrypt the signing key. The client uses the GetClientLicensorCert method to acquire a CLC from that server. More information can be found in [MS-RMPR] section 3.5.4.2, which details the GetClientLicensorCert method.

6.1.1.2 Acquire Templates

An RMS Client Application can request rights policy templates from an RMS server. First the RMS Client Application makes an AcquireTemplateInformation request to the server. The server returns information about the available templates in the form of a list of GUIDs and hashes corresponding to the server templates. The client then makes AcquireTemplate requests to the server to download each individual template. The client SHOULD maintain a local store of templates and when subsequent AcquireTemplateInformation requests are made the client SHOULD compare the server list against the list in the local store<5>. The client makes add/delete/edit updates to the templates in the local store. Through this process the client always keeps its templates in sync with the ones on the server.

6.1.1.3 Offline Publishing

Publishing offline is done entirely by the RMS client application and does not involve any interaction with the RMS server. When the client is used to protect content, it generates a PL that contains the usage policy and the content key, both of which are encrypted using the server's public key. The PL also contains a reference to a server that can be used to issue ULs from the PL.

During offline publishing, the usage policy and content key are encrypted using the server's public key from the ISSUER element of the CLC. The PL is signed using the CLC private key, and the resultant signed PL chain includes the PL, CLC, and SLC from the CLC chain. For more details on offline publishing see [MS-RMPR] section 1.3.5.

6.1.2 Protecting Content Using Online Publishing

Content can be published by using online publishing, where the client does not need to maintain certificates locally and use the server to sign Publishing Licenses (PLs). In online publishing the client computer is not required to be activated and the client user is not required to be bootstrapped. This example describes a typical scenario using online publishing, assuming rights policy templates are used in the publishing process.

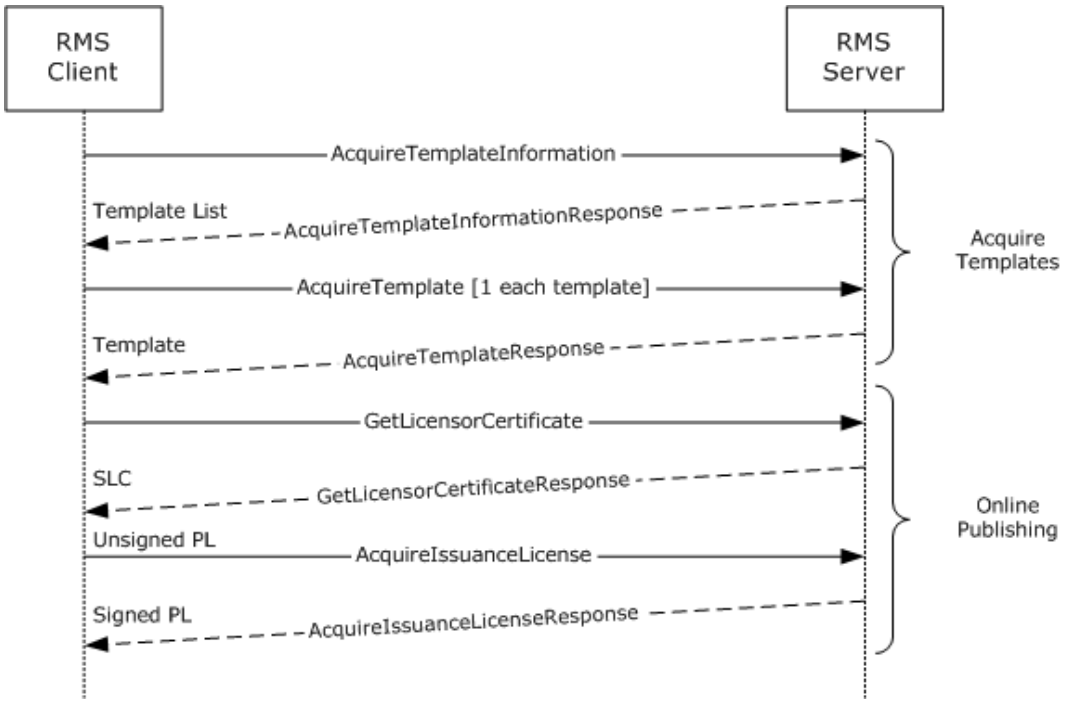

Figure 7: Message flow for protecting content using online publishing

6.1.2.1 Acquire Templates

An RMS Client Application can request rights policy templates from an RMS server. First the RMS Client Application makes an AcquireTemplateInformation request to the server. The server returns information about the available templates in the form of a list of GUIDs and hashes corresponding to the server templates. The client then makes AcquireTemplate requests to the server to download each individual template. The client SHOULD maintain a local store of templates and when subsequent AcquireTemplateInformation requests are made the client SHOULD compare the server list against the list in the local store<6>. The client makes add/delete/edit updates to the templates in the local store. Through this process the client always keeps its templates in sync with the ones on the server.

6.1.2.2 Online Publishing

6.1.2.2.1 Acquire the Server's Certificate

When publishing content, the RMS client application needs the public key from the server's SLC to encrypt the symmetric content key and usage rights in the PL. This way only the server can decrypt these elements, which it will do to generate Use Licenses. To get the server's SLC the RMS client application sends a request to the GetLicensorCertificate method. No custom data is sent in this request, and when the server receives the request it returns the SCL to the RMS client application.

6.1.2.2.2 Generate a Publishing License

After the RMS client application has the SLC and templates required for publishing, it generates a publishing license. The RMS server is not contacted during this step. The application includes the symmetric content key and use restrictions in the PL and encrypts them using the server's public key.

6.1.2.2.3 Sign the Publishing License

The final step in online publishing is to sign the PL. This is done by the RMS client application making an AcquireIssuanceLicense request to the RMS server. In the AcqurieIssuanceLicense request the unsigned PL is sent to the server, the server signs it and returns the signed PL.

6.1.3 Consuming Protected Content

When a user receives protected content they wish to consume, a series of steps have to happen to set them up in RMS and for the application to acquire a use license to open the protected content. This section describes the typical steps performed to consume content, assuming the computer and user have not used RMS in the past.

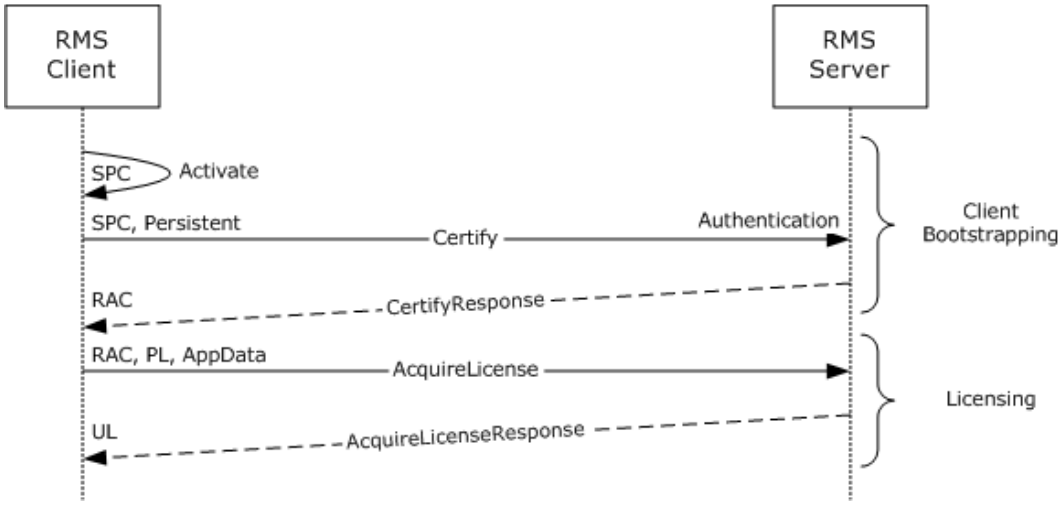

Figure 8: Message flow for consuming protected content

6.1.3.1 Client Bootstrapping

6.1.3.1.1 Activate the Computer

Before a computer or device can participate in the RMS system it MUST generate a Security Processor Certificate (SPC). In RMS protocol versions 1.0 SP1 and later, clients self-activate, generating their own SPC.<7> More details on activation can be found at [MS-RMPR] section 3.2.4.1.

6.1.3.1.2 Certify the User

To access protected content, the user needs a RAC that corresponds to the user's account. The client uses the Certify request to acquire a RAC. The server issues an asymmetric encryption key pair and identifies the user account in the RMS System. The client MUST have a valid SPC before calling Certify. Full details of client certification can be found in [MS-RMPR] section 3.3.4.1.

6.1.3.2 Licensing

The UL describes what usage policies apply to the user while accessing a particular protected content file. It also contains the content key encrypted with the user's RAC public key. The UL is the authorization token that allows a user to access protected content.

To retrieve the UL, the RMS client application issues an AcquireLicense request. In an AcquireLicense request the client sends the PL that came with the content along with the user's RAC and any application data to the RMS server. The server validates the RAC and PL. If validation succeeds and the user is granted access, the server generates a Use License and returns it to the client. More details on AcquireLicense can be found in [MS-RMPR] section 3.4.4.1.

6.1.4 Task Scheduler Job to Update RMS Templates from Server

Two tasks are added to the Task Scheduler to manage the synchronization between client and server to keep the **rights policy templates** up to date.

6.1.4.1 AD RMS Rights Policy Template Management (Automated)

This task automatically updates rights policy templates by querying an AD RMS Server. If authentication fails, the task fails silently and logs an Action Failed event. For more information on Task Scheduler, see [MSDN-TaskSch].

This task is scheduled to run at 3am daily. It is disabled by default, and can be enabled by Group Policy or manually through the Microsoft Windows® Task Scheduler, or through a command-line script. An example command line is: **schtasks.exe /change /tn "\microsoft\windows\Active Directory Rights Management Services Client\AD RMS Rights Policy Template Management (Automated)" /enable**.

This task MUST provide a way to keep track of the last-update time, as well as a way to control the frequency of updates. The task MAY use the Windows Registry to achieve these objectives.

- **Key**: HKEY_CURRENT_USER\Software\Policies\Microsoft\MSDRM\TemplateManagement

  Name: updateFrequency

  Type: DWORD

  Usage: The template update frequency is a numeric value that specifies the number of days since the last update after which the client SHOULD update its rights policy templates from the server. If this registry key does not exist, the default update frequency of 30 days applies.

- **Key**: HKEY_CURRENT_USER\Software\Policies\Microsoft\MSDRM\TemplateManagement

  Name: lastUpdatedTime

  Type: REG_SZ

  Usage: The value of this registry key is a UTC date and time that indicates the point in time when the last update took place. This value MUST be updated by the task.

- **Key**: HKEY_CURRENT_USER\Software\Policies\Microsoft\MSDRM\TemplateManagement

  Name: updateIfLastUpdatedBeforeTime

  Type: String

  Usage: The value of this registry key is a UTC date and time. If the last update was performed before the point in time specified in this key, the client is forced to update its rights policy templates.

6.1.4.1.1 Conditions to Trigger the Automated Task

The task performs a template acquisition when at least one of the following conditions is satisfied (evaluated in the following order):

- The **lastUpdatedTime** registry key is not found. This only occurs the first time the task is running, since the task updates that key at every template update.
- The value of the **lastUpdatedTime** key is greater than the current time. Since this probably means that the clock was changed, template acquisition is performed.
- The value of the **lastUpdatedTime** key precedes the current time by more than the number of days contained in the **updateFrequency** key, or if the **updateFrequency** key is not found, by more than 30 days.
- The **updateIfLastUpdatedBeforeTime** key exists and its value is later than the value of the **lastUpdatedTime** key.

6.1.4.1.2 Find Service Location

The task SHOULD make a **FindServiceLocationForUser** request to retrieve the Licensing Service, and in return receive the URL to make the **AcquireTemplateInformation** request. Details about the **FindServiceLocationsForUser** method can be found in [MS-RMPR] section 3.7.4.2.

6.1.4.1.3 Acquire Templates

Make a request to the **AcquireTemplateInformation** method, and store the templates in a store. After the templates are retrieved from the server, update the **lastUpdatedTime** registry key with the current time. Details about the **AcquireTemplateInformation** method can be found in section 5.4.2.2.

6.1.4.2 AD RMS Rights Policy Template Management (Manual)

This task supports non-silent template update, and is activated manually by the user through the Microsoft Windows® Task Scheduler.

This task does not determine whether or not it is time to run; it retrieves templates from the server regardless of when they were last updated.

The task performs the steps described in sections 6.1.4.1.2 and 6.1.4.1.3, with the only difference being that when sending the **FindServiceLocationForUser** request, if default authentication fails, instead of failing silently, the task provides a username-password prompt to the user to obtain credentials to connect to the server.

Details about the authentication methods used in the **FindServiceLocationsForUser** method can be found in [MS-RMPR] section 3.7.4.2.

6.2 Communication Details

6.2.1 Setting the Service Connection Point

The RMS Administrator sets the service connection point (SCP) in Active Directory using LDAP [RFC2251]. This section uses the following local variable:

ActiveDirectory_Connection: An ADConnection handle (see [MS-ADSO] section 6.2.2).

Setting the SCP involves the following tasks:

1. Invoke the "Initialize ADConnection" task ([MS-ADSO] section 6.2.6.1.1) to construct an **ADConnection** handle, with the following parameters:

   - *TaskInputTargetName*: NULL
   - *TaskInputPortNumber*: 389

   Store the created **ADConnection** handle in the **ActiveDirectory_Connection** variable.

2. Invoke the "Setting an LDAP Option on an ADConnection" task ([MS-ADSO] section 6.2.6.1.2) with the following parameters:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**
   - *TaskInputOptionName*: LDAP_OPT_PROTOCOL_VERSION
   - *TaskInputOptionValue*: 3

3. Invoke the "Establishing an ADConnection" task ([MS-ADSO] section 6.2.6.1.3) with the following parameter:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**

   If the *TaskReturnStatus* returned is not 0, setting the SCP fails.

4. Invoke the "Performing an LDAP Bind on an ADConnection" task ([MS-ADSO] section 6.2.6.1.4) with the following parameters:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**

   If the *TaskReturnStatus* returned is not 0, setting the SCP fails.

5. Invoke the "Perform an LDAP Operation on an ADConnection" task ([MS-ADSO] section 6.2.6.1.6) with the following parameters:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**
   - *TaskInputRequestMessage*: LDAP SearchRequest message ([RFC2251] section 4.5.1), as follows:
     - *baseObject*: EMPTY string
     - *scope*: baseObject
     - *filter*: (objectClass=*)
     - *attributes*: configurationNamingContext
     - *derefAliases*: neverDerefAliases
     - *typesOnly*: FALSE
   - *TaskOutputResultMessage*: Upon successful return from the task, this parameter will contain the results of the LDAP search.

   If the *TaskReturnStatus* returned is not 0, setting the SCP fails.

6. Invoke the "Perform an LDAP Operation on an ADConnection" task ([MS-ADSO] section 6.2.6.1.6) with the following parameters:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**
   - *TaskInputRequestMessage*: LDAP AddRequest message ([RFC2251] section 4.7), as follows:
     - *entry*: "CN=RightsManagementServices,CN=Services", prepended to the value of the configurationNamingContext attribute of the SearchResultEntry of the first LDAPMessage in the *TaskOutputResultMessage* of step 5.
     - *attributes*:
       - *objectClass*: container

   If the *TaskReturnStatus* returned is not 0, setting the SCP fails.

7. Invoke the "Perform an LDAP Operation on an ADConnection" task ([MS-ADSO] section 6.2.6.1.6) with the following parameters:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**
   - *TaskInputRequestMessage*: LDAP AddRequest message ([RFC2251] section 4.7), as follows:
     - *entry*: "CN=SCP", prepended to the entry used in step 6.
     - *attributes*:
       - *objectClass*: serviceConnectionPoint
       - *keywords*: 1.0, MSRMRootCluster
       - *serviceBindingInformation*: "[baseURL]/certification", where "[baseURL]" is the value of the **baseUrl** field of the **StoredConfiguration** element of the RMS Server ADM ([MS-RMPR] section 3.1.1.2.2).

   If *TaskReturnStatus* returned is not 0, setting the SCP fails.

8. Invoke the "Perform an LDAP Unbind on an ADConnection" task ([MS-ADSO] section 6.2.6.1.5) with the following parameter:

   - *TaskInputADConnection*: **ActiveDirectory_Connection**

If setting the SCP fails, clients will be unable to locate an RMS server using the SCP.

6.3 Transport Requirements

The system does not define a transport beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

6.4 Timers

The system does not define any timers.

6.5 Non-Timer Events

This system does not define any non-timer events beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

6.6 Initialization and Reinitialization Procedures

The system does not define any initialization requirements beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

6.7 Status and Error Returns

The system does not define any error handling requirements beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

7 Security

This section documents system-wide security issues that are not otherwise described in the Technical Documents (TDs) for the Member Protocols. It does not duplicate what is already in the Member Protocol TDs unless there is some unique aspect that applies to the system as a whole.

The system does not define any security requirements beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

8 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft Windows NT® operating system
- Microsoft Windows® 2000 operating system
- Windows® XP operating system
- Windows Server® 2003 operating system
- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system
- Windows Server® 2008 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 3.3.1: All versions of Microsoft's RMS server earlier than version 2 contacted the Microsoft enrollment service to sign the SLC key into the hierarchy. The RMS version 2 server ships with a shared enrollment private key and certificate chain. When the RMS version 2 server initializes, it generates its own unsigned SLC, signs it with this shared enrollment private key, and appends the certificate chain.

<2> Section 3.3.4.1: All versions of Microsoft's RMS server earlier than version 2 contacted the Microsoft enrollment service to sign the SLC key into the hierarchy. The RMS version 2 server ships with a shared enrollment private key and certificate chain. When the RMS version 2 server initializes, it generates its own unsigned SLC, signs it with this shared enrollment private key, and appends the certificate chain.

<3> Section 3.3.4.4: RMS 2.0 provides an interface for clients to retrieve rights policy templates. RMS versions prior to 2.0 do not provide this interface.

<4> Section 6.1.1.1.1: Support for the RMS Client version 1.0 has ended, and the Cloud Activation Service is no longer available for activation requests. Activate requests from RMS 1.0 clients will still be made to the RMS server, but activation requests from the RMS server to the Cloud Service will fail. This failure will result in the server returning a failure to the RMS client.

RMS: Client-to-Server protocol versions 1.0 SP1, 1.0 SP2, and 2.0 use self activation. Self activation continues to function as expected.

<5> Section 6.1.1.2: In Windows implementations, the client does not maintain the list of templates as part of client and server operations. A Scheduled Tasks job is provided that can be enabled and is run separately to maintain the local store of templates.

<6> Section 6.1.2.1: In Windows implementations, the client does not maintain the list of templates as part of client and server operations. A Scheduled Tasks job is provided that can be enabled and is run separately to maintain the local store of templates.

<7> Section 6.1.3.1.1: Support for the RMS Client version 1.0 has ended, and the Cloud Activation Service is no longer available for activation requests. Activate requests from RMS 1.0 clients will still be made to the RMS server, but activation requests from the RMS server to the Cloud Service will fail. This failure will result in the server returning a failure to the RMS client.

RMS: Client-to-Server protocol versions 1.0 SP1, 1.0 SP2, and 2.0 use self activation. Self activation continues to function as expected.

9 Change Tracking

This section identifies changes that were made to the [MS-RMSO] protocol document between the May 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Added explanatory statement regarding the removal of the publishing year from Microsoft Open Specification document references.	N	Content updated.

10 Index

R

S

T

U

V

W

X