Using the
ADOBE® LIVECYCLE® RIGHTS MANAGEMENT 9.5
PORTABLE PROTECTION LIBRARY
Contents

Chapter 1: Preface
What's in this document ................................................................. 1
Who should read this document ...................................................... 1
Related documentation ................................................................. 1

Chapter 2: Getting Started
Introduction .................................................................................. 2
Supported functionality ................................................................. 2
Prerequisites .................................................................................. 3
Client applications .......................................................................... 3
What you received .......................................................................... 4
Install the library ........................................................................... 4
Access the documentation ............................................................. 6
Considerations for building the Portable Protection Library ............... 6

Chapter 3: Samples
Considerations for building the sample application ......................... 8
Build the sample application .......................................................... 8
Execute the sample application ....................................................... 9

Chapter 4: Tutorial—LiveCycle Rights Management 9.5 Portable Protection Library
Introduction .................................................................................. 10
Guidelines ..................................................................................... 10
Authentication methods ................................................................. 11
Chapter 1: Preface

This document provides information about using the Adobe® LiveCycle® Rights Management 9.5 Portable Protection Library. The Portable Protection Library lets you programmatically modify documents to facilitate their automated protection.

What’s in this document

This document provides information about setting up and using the Rights Management 9.5 Portable Protection Library. This document also describes the classes, headers, libraries, and sample application that accompany the Portable Protection Library.

Who should read this document

This document is intended for C++ developers who are responsible for developing client applications that interact with Rights Management 9. In particular, this library is useful for developers working on data loss protection applications.

Related documentation

In addition to this document, the resources in the following table provide information relevant to Rights Management 9.5 Portable Protection Library.

<table>
<thead>
<tr>
<th>For information about</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rights Management 9.5 Portable Protection Library APIs</td>
<td>Navigate to the Documentation\API Reference folder after you extract the Portable Protection Library ZIP archive. The Rights Management 9.5 Portable Protection Library API Reference in this folder provides information about the relevant APIs.</td>
</tr>
<tr>
<td>Other Adobe services and products</td>
<td><a href="http://www.adobe.com">www.adobe.com</a></td>
</tr>
</tbody>
</table>
Chapter 2: Getting Started

Introduction

Adobe® LiveCycle® Rights Management 9.5 Portable Protection Library is a collection of C++ classes, headers, libraries, and APIs that let you programmatically modify documents for their automated protection. Portable Protection Library also lets you unprotect documents and review them for common use cases and workflows.

You can compile Portable Protection Library into other projects and products to facilitate low-latency operations that do not require an exchange of documents with the LiveCycle ES2 server. Server communication is minimal and restricted to activities such as authenticating the client and exchanging document-specific keys. Portable Protection Library is extensible and is designed for inclusion in products and solutions without the need for developers to understand the various file types supported.

The Portable Protection Library supports Adobe PDF, Microsoft® Word, PowerPoint, and Excel files.

Supported functionality

The Rights Management 9.5 Portable Protection Library supports offline access, encryption, and decryption functionalities. It lets you programmatically perform the following tasks:

- Secure PDF and MS Office 2003/2007 documents using a policy obtained from the LiveCycle ES2 server.
- Unprotect secured documents
- Revoke access to policy-protected documents
- Create and manage policies
- Switch policies

Documents can be protected unencrypted or using 256-bit or 128-bit AES encryption. The following authentication types are supported:

Kerberos authentication The user must be part of a Windows domain configured on the server.

User name/password authentication The user name and password are fetched using PPKClientCallbacks, whenever necessary.

Security Assertion Markup Language (SAML) authentication Sometimes, you may not be able to provide all end-user client machines with LiveCycle credentials or user accounts. In such cases, you could have one central LiveCycle client with an authenticated session. Once authenticated, the server would pass an SAML token identifying the session. This SAML token can be sent securely to the end-user client terminal that needs to be authenticated. In this way, you can centralize and control the authentication of end-user client terminals.
Prerequisites

Hardware requirements
An Intel® Pentium® 3 processor (or higher) and 512 MB (or more) of RAM are required for using the Portable Protection Library.

Software requirements
Portable Protection Library is currently available only for Microsoft® Windows®. You can build Portable Protection Library using Microsoft® Visual Studio® 2008 SP1 or Microsoft Visual Studio 2005 and use it with an existing LiveCycle ES2 server that has Service Pack 1 (9.0.0.1) or later applied.

The following table summarizes the software requirements for the Portable Protection Library.

<table>
<thead>
<tr>
<th>Client applications</th>
<th>32-bit and 64-bit Windows XP, Windows Vista, or Windows 7 operating systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>Microsoft Visual Studio 2005 or Visual Studio 2008 SP1</td>
</tr>
<tr>
<td>LiveCycle server</td>
<td>LiveCycle ES2 with Service Pack 1 (9.0.0.1) or later applied</td>
</tr>
<tr>
<td>Microsoft Office</td>
<td>Microsoft Office 2003 or 2007</td>
</tr>
<tr>
<td>Adobe Acrobat®</td>
<td>Acrobat 8 or higher</td>
</tr>
</tbody>
</table>

Client applications

Client applications in various businesses, such as corporate workplaces, government workplaces, and academic institutions, can use Rights Management Portable Protection Library to perform functions such as policy-protecting documents that are sent in and out of the organization.

Client applications use the client APIs provided by this library to interact with the LiveCycle server. A client application can be located within or outside an enterprise firewall.

The following illustration shows end users interacting with a client application to perform various tasks related to policy protection.
What you received

The Rights Management 9.5 Portable Protection Library is available as a ZIP file that you can obtain from your Adobe representative. This file contains the following items:

- A collection of C++ header files and DLL files that make up Rights Management 9.5 Portable Protection Library
- Product documentation, consisting of the following:
  - Getting Started (this document, PDF)
  - Rights Management 9.5 Portable Protection Library API Reference (HTML)
- Sample application that demonstrates the capabilities of the product, including creating policies, policy-protecting documents, and opening policy-protected documents.

Install the library

It takes only a few minutes to set up the Portable Protection Library. Explanation of the required installation steps are provided in the sections that follow.

To install the software, perform the following tasks:

1. Download and extract the ZIP file.
2. Check the system requirements.
3. Set up your development environment.
Download and extract the ZIP file

If you are reading this document, you have probably already downloaded and extracted the ZIP file. It is assumed that you saved a copy of the ZIP file locally after you downloaded the software.

The ZIP file contains the product software, documentation, and sample applications. The following folders are created after you extract the contents of the ZIP file:

- Documentation
- PPL Binaries and Samples

List of included classes

- PPKClientCallbacks
- PPKClientCallbacks::ApplicationInfo
- PPKClientCallbacks::CertificateInfo
- EDCClientException
- EDCCountedPtr<X>
- PDRLDateTime
- PDRLException
- PDRLPermission
- PDRLPolicy
- PDRLPolicyEntry
- PDRLPrincipal
- PDRLValidityPeriod
- PPKPublisher
- PPKPublisher::PolicyInfo
- PPKFile
- PPKFileFactory
- PPKPackager
- PPKPackager::PPKInspectResponse

For detailed information about these classes, refer to the Rights Management 9.5 Portable Protection Library API Reference.

Included header files

The following table lists the header files supplied as part of the Portable Protection Library.

<table>
<thead>
<tr>
<th>C_EDCException</th>
<th>C_PDRLPolicy.h</th>
<th>C_PDRLPolicyEntry.h</th>
</tr>
</thead>
<tbody>
<tr>
<td>C_PDRLPPrincipal.h</td>
<td>C_PDRLValidityPeriod.h</td>
<td>C_PPKClientCallbacks.h</td>
</tr>
<tr>
<td>C_PPKExports.h</td>
<td>C_PPKFile.cpp</td>
<td>C_PPKFile.h</td>
</tr>
<tr>
<td>C_PPKFileImpl.h</td>
<td>C_PPKPackager.h</td>
<td>C_PPKPublisher.h</td>
</tr>
<tr>
<td>EDCClientException.h</td>
<td>EDCCountedPtr.h</td>
<td>EDCExceptionHandler.h</td>
</tr>
<tr>
<td>PDRL.h</td>
<td>PDRException.h</td>
<td>PDRLPermission.h</td>
</tr>
</tbody>
</table>

Last updated 3/14/2011
**Set up your development environment**

Rights Management 9.5 Portable Protection Library supports Microsoft Visual Studio 2005 or 2008 SP1. Your projects must include the resources listed in “Download and extract the ZIP file” on page 5.

The samples provided with the Portable Protection Library demonstrate how you can configure project properties.

**View the configuration settings used in the sample applications**

- Open a solution and click Project > Properties to display the Property Pages dialog box.

**Access the documentation**

After extracting the ZIP file, you can view the accompanying product documentation. You can view PDF documents using either Adobe Reader®, Adobe Acrobat® Pro Extended, Acrobat Pro, or Acrobat Standard.

You can view the API Reference using a Web browser. The API Reference includes information that describes Portable Protection Library classes, libraries, headers, etc.

- Using a Web browser, navigate to Documentation\API Reference and open the index.html file.

  You can bookmark these pages in your Web browser.

**Considerations for building the Portable Protection Library**

Before building the library, ensure that:

- The LiveCycle ES2 server has Service Pack 1 or later applied
- Appropriate certificates are installed on the machine being used for protection
- Location of the PPL DLLs is added to the system path. Alternatively, the DLLs should be present in the folder that contains the .exe file.
Chapter 3: Samples

The Rights Management 9.5 Portable Protection library contains a sample application that you can use to become familiar with the Client API. You can open the sample application by opening the PPSSamples folder in Microsoft Visual Studio 2008 SP1 or Visual Studio 2005.

After you open this project, you can compile and run the sample application. This sample application consists of various methods that perform specific tasks.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>protect()</td>
<td>Protects PDF and a Word documents using policies defined on the LiveCycle ES2 server</td>
</tr>
<tr>
<td>switchPolicy()</td>
<td>Protects Word and PDF documents and, then, switches the policy for those documents</td>
</tr>
<tr>
<td>unProtect()</td>
<td>Protects and then unprotects PDF and Word files</td>
</tr>
<tr>
<td>revokeDocLicense()</td>
<td>Protects Word and PDF files and then revokes their licenses</td>
</tr>
<tr>
<td>inspectDocument()</td>
<td>Inspects a document and displays the information obtained</td>
</tr>
<tr>
<td>createSession()</td>
<td>Creates a session with the server. Ensure that the correct name and port number for your server are specified as arguments for this method.</td>
</tr>
<tr>
<td>main()</td>
<td>This method is the entry point for the application. It also displays the packagers and policies available on the server for your use.</td>
</tr>
</tbody>
</table>

Considerations for building the sample application

Before building the sample application provided with the Portable Protection Library, ensure the following:

- Relevant server and user credentials are specified in the PPSSample.cpp and Samples.cpp files
- Policies and users are created on the LiveCycle ES2 server exactly as they’re specified in Samples.cpp. Policies must bear the names Sample Policy and New Policy. Further, ensure that the only valid policy IDs existing on the LiveCycle ES2 server are specified while invoking APIs.

Build the sample application

2. Select File > Open Solution, navigate to the samples directory, and open PPSSample.sln.
3. Rebuild the solution.
Execute the sample application

1. At the command prompt, navigate to the directory containing your executable file.
2. Execute the sample application.
3. Follow on-screen instructions, as required.
Chapter 4: Tutorial—LiveCycle Rights Management 9.5 Portable Protection Library

Introduction

PPKPackager and PPKPublisher are the two main classes in the LiveCycle Rights Management 9.5 Portable Protection Library that you’ll use often. PPKPackager manipulates document formats to protect, unprotect, and inspect documents. PPKPublisher represents a LiveCycle ES2 user that has permissions to manage documents.

PPKClientCallbacks is an interface that you can implement. Typically, your implementations of the virtual functions in this class query the user for preferences, identification, or other available information, besides notifying the user of certain events. Instances of all classes are reference-counted and managed using the EDCCountedPtr class.

Guidelines

1. Instantiate an implementation of the PPKClientCallbacks interface.
2. Create an instance of the PPKPublisher class using the createSession method. This method takes the URL of the LiveCycle ES2 server and the PPKClientCallbacks instance. The Portable Protection Library calls the implemented functions when it needs to get credentials, preferences, and other required information from the publisher.
3. Create an instance of the PPKPackager class using the getPackager function. The Portable Protection Library can handle different file formats—each of which require a different packager. You must specify the type of packager that you need. Each instance of the packager class is associated with a file name.
4. Protect the document using the protectDoc function. This function takes an instance of PPKPublisher, specifying the LiveCycle ES2 user protecting the document. Other parameters are the ID of the policy to use, the name of the document to be registered on the server, and the file name of the protected document. To protect the document in place, leave the file name parameter empty.

Some other useful guidelines on using the Portable Protection Library are as follows:

- The inspectDoc function returns this protection status of a file. The information returned by inspectDoc includes the URL of the server used, the document ID, policy used for protection, etc.

- When a document is protected, its contents are encrypted. The contents can be decrypted (in place, if necessary) using the unProtect function.

Sometimes, you may want to decrypt a document temporarily—for example, to analyze its contents. In such a case, you call the unProtect function with the shouldLogEvent parameter set to false. This function decrypts the document for you without intimating the server, so that the audit trail is not cluttered with false entries. Once the temporary use case has been satisfied, it is the responsibility of the application developer to discard the decrypted content.
Your application may need to perform packaging operations in memory or on a remote file system. In such cases, you can override the default file-handling of the library by providing your own abstraction to the file system. To achieve this end, you will need to implement the PPKFileFactory and PPKFile interfaces. The file name, however, will still be restricted to the string type.

Your implementation of PPKFileFactory can choose how to interpret the file name. If PPKPackger::getPackager is called with an instance of your implementation of PPKFileFactory, an instance of PPKPackager is returned. This instance uses your implementation of PPKFile and PPKFileFactory for file handling operations.

**Authentication methods**

The server can authenticate a publisher in one of the following three ways:

**Kerberos Authentication**  The user must be part of a Windows domain configured on the server.

**Username / Password Authentication**  The user name and password are fetched using PPKClientCallbacks, whenever necessary.

**SAML authentication**  Sometimes, you may not be able to provide all end-user client machines with LiveCycle credentials or user accounts. In such cases, you could have one central LiveCycle client with an authenticated session. Once authenticated, the server would pass an SAML token identifying the session. This SAML token can be sent securely to the end-user client terminal that needs to be authenticated. In this way, you can centralize and control the authentication of end-user client terminals.

*Note: Kerberos is the first choice of authentication method. Once authenticated using any method, the user stays authenticated till the logout function is called or until the session expires.*