

Certification of Interactive Forms by Adobe using Server Certificates

Applicable Releases:

NetWeaver CE 7.1

NetWeaver CE 7.1 EhP1

Topic Area:

User Productivity

Capability:

User Interface Technology

Version 1.0

April 2009

© Copyright 2009 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice.

These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

SAP NetWeaver "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

Disclaimer

Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressly prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP's Support Services and may not be modified or altered in any way.

Document History

Document Version	Description
1.00	First official release of this guide

Typographic Conventions

Type Style	Description
<i>Example Text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation
Example text	Emphasized words or phrases in body text, graphic titles, and table titles
Example text	File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
Example text	User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

Icons





Icon	Description
	Caution
	Note or Important
	Example
	Recommendation or Tip

Table of Contents

1.	Business Scenario.....	1
2.	Prerequisites	1
3.	Creating and configuring server certificates.....	2
3.1	Creating a server certificate in the Key Storage Service.....	2
3.2	Export private key to a file and sign with Certification Authority. (optional)	5
3.3	Export the key pair to files	5
3.4	Import the public key to the ADS credential storage	6
3.5	Import the certificate to the ADS trusted anchors storage	8
3.6	Restart The PDF Manipulation Service	11
3.7	Applying a server certificate on Interactive Form	12
3.8	Validating server certificates on Interactive Form	14
3.9	Demo application	18

1. Business Scenario

In your business scenario you have a requirement to digitally sign an interactive form with a server credential. This guide will demonstrate how to configure the server credentials for form signing, how to apply these certificates on the interactive forms as well as how to validate these certificates with the use of the Adobe Document Services API.

2. Prerequisites

Prerequisite to this guide is to have the NetWeaver CE Application Server configured for the use of SSL. This is usually done by default during the application server installation or performed as a later step. Also it is required that the JVM is configured with the Java™ Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files, please refer to an SSL configuration guide for more information.

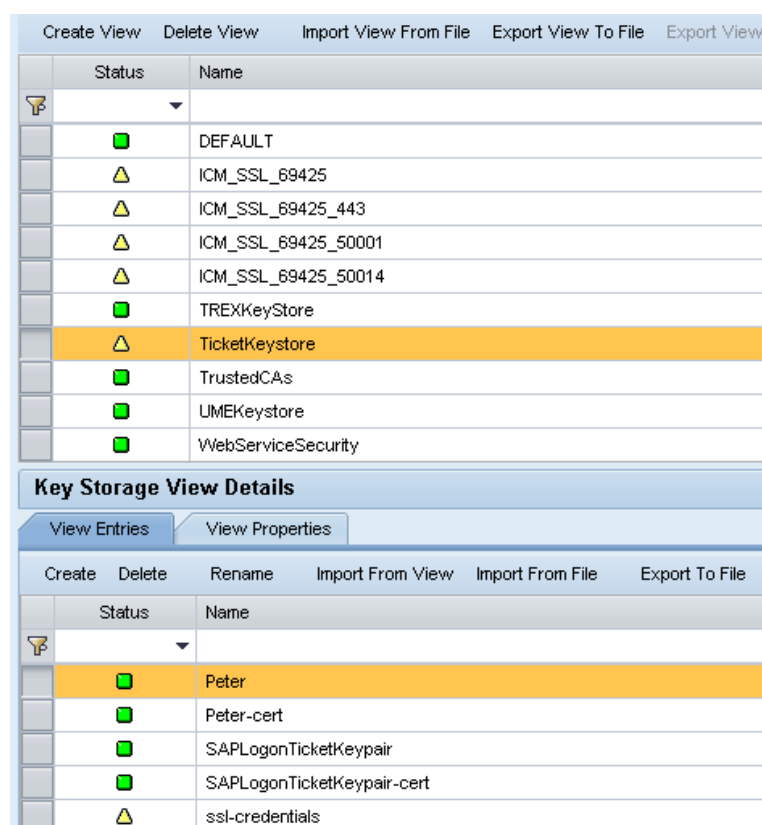
Adobe Document Services should be configured for secured access, please refer to the configuration guide: How to make use of ADS security prior to continuing with this document.

3. Creating and configuring server certificates.

3.1 Creating a server certificate in the Key Storage Service

1. Start and log in to the SAP NetWeaver Administrator. <http://server:port/nwa>
2. Navigate to *Configuration Management -> Security -> Certificates and Keys -> Key Storage* service

Or use the short link <http://server:port/nwa/key-storage>



Status	Name
	DEFAULT
	ICM_SSL_69425
	ICM_SSL_69425_443
	ICM_SSL_69425_50001
	ICM_SSL_69425_50014
	TREXKeystore
	TicketKeystore
	TrustedCAs
	UMEKeystore
	WebServiceSecurity

Key Storage View Details	
View Entries	
Status	Name
	Peter
	Peter-cert
	SAPLogonTicketKeypair
	SAPLogonTicketKeypair-cert
	ssl-credentials

3. Click on *Ticketkeystore* (You can use any other store or also you can import a certificate created by a certification authority).
4. To create the Public-Private-Key Pair press the button *Create*

- Fill out the subject properties.

The screenshot shows the 'Add New Key Storage Entry' dialog with 'Step 1: Entry Settings' selected. The 'Entry Settings' section contains the following fields:

- Entry Name: * (text field with value 'server')
- Algorithm: * (dropdown menu with value 'RSA')
- Key Length: * (dropdown menu with value '1024')
- Valid From: * (date field with value '4/14/2009')
- Valid To: * (date field with value '4/14/2029')
- Store Certificate: (checkbox checked)

At the bottom, there are buttons: Cancel, Back, Next, and Finish.

- Common Name suppose to be the name or ID of the server or user you are creating a key-pair (In this case we will call it "Server")
- The Entry Name is the name for identifying the key pair in the key store.
- Specify Validity period;
- Select "RSA" as secure algorithm
- Select Key Length - "1024" in this example
- Choose "Store Certificate" in order to store the public key as well.
- Press the button *Next*
- Fill out additional subject properties information for the certificate. You can extract all these additional information out from the certificate when validating it.
- The *commonName* presents the name of the certificate

The screenshot shows the 'Add New Key Storage Entry' dialog with 'Step 2: Subject Properties' selected. The 'Subject Properties' section contains a table with the following data:

Name	Object Identifier	Value
countryName *	2.5.4.6	US
stateOrProvinceName	2.5.4.8	CA
organizationName	2.5.4.10	SAP
localityName	2.5.4.7	Palo Alto
organizationalUnitName	2.5.4.11	RIG
commonName *	2.5.4.3	server

Below the table, there is a note: "Note: The properties with the asterisk symbol (*) in their name must have a value". At the bottom, there are buttons: Cancel, Back, Next, and Finish.

15. Press the button *Next...* {Note: According the new RCA regulations, self signed key pairs are not always accepted as credited identity. You have to always use Certification Authorities for signing self generated key-pairs. So, there are two possibilities... generating the key pairs and sending them to a CA for signing. Or since you have a *Credential* signed by a CA, use it to sign the newly created key pairs. }

Key Storage View Details

View Entries View Properties

Add New Key Storage Entry

Step 1 Step 2 Step 3 Step 4

Entry Settings Subject Properties Sign with Key Pair Preview and Create

Preview and Create Entry

Entry Name: ADSUser_TechEd2008
 Algorithm: RSA
 Key Length: 1024
 Valid From: 8/14/2008
 Valid To: 8/14/2015

Subject Properties

Name	Value
countryName *	US
stateOrProvinceName	Nevada
localityName	Las Vegas
organizationName	SAP
organizationalUnitName	TechEd 2008
commonName *	ADSUser2008

Issuer Info

Key	Value
organizationalUnitName	TechEd
countryName	US
commonName	localhost

Cancel Back Next Finish

Key Storage View Details

View Entries View Properties

Create	Delete	Rename	Import From View	Import From File	Export To File	Generate CSR Request	Import CSR Re
Status	Name	Entry Type	Algorithm				
■	SAPLoginTicketKeypair-cert	CERTIFICATE	DSA				
■	server	PRIVATE KEY	RSA				
■	server-cert	CERTIFICATE	RSA				
⚠	ssl-credentials	PRIVATE KEY	RSA				
⚠	ssl-credentials-cert	CERTIFICATE	RSA				

Entry Details

PRIVATE KEY entry

Creation date : Tue Apr 14 17:36:32 EDT 2009 (14 Apr 2009 21:36:32 GMT)
 Version: : PKCS#8 RSA
 Key Size : 1024 bits

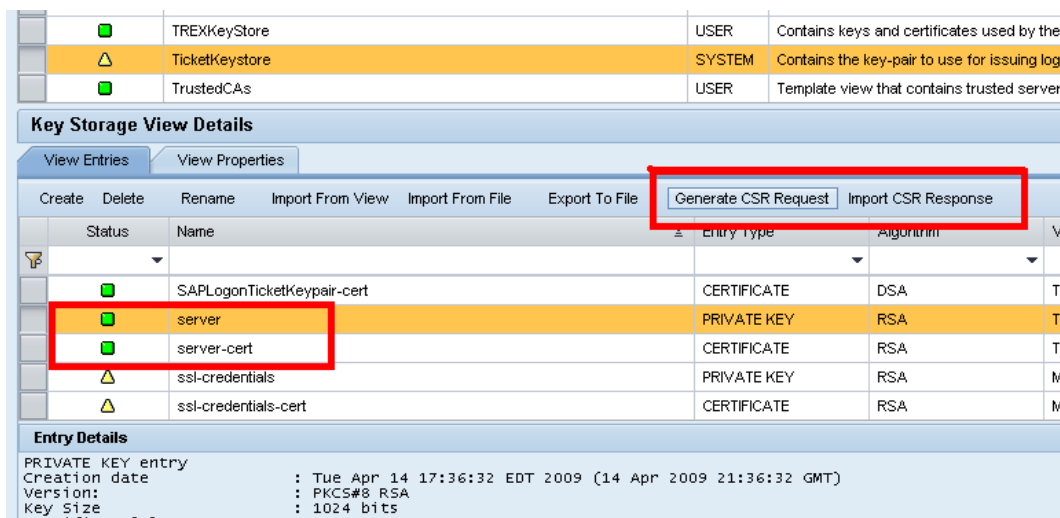
Certificate[0] -----
 Version : ver.3 X.509
 Algorithm : RSA
 Key Size : 1024 bits
 Subject name : CN=server,OU=RIG,L=Pa1o A1to,O=SAP,ST=CA,C=US
 Issuer name : CN=server,OU=RIG,L=Pa1o A1to,O=SAP,ST=CA,C=US
 Serial number : 2316886278
 Signature Algorithm : md5WithRSAEncryption (1.2.840.113549.1.1.4)

validity:
 not before : Tue Apr 14 17:28:00 EDT 2009 (14 Apr 2009 21:28:00 GMT)
 not after : Sat Apr 14 17:28:00 EDT 2029 (14 Apr 2029 21:28:00 GMT)

Public key fingerprint : 3C:A3:74:15:4B:54:F9:F9:84:E2:6E:17:97:21:9B:ED
 Certificate fingerprint(MD5): 85:17:F8:CF:4E:54:CD:37:36:11:A0:A4:4B:53:A1:B8
 Certificate extensions :
 [critical]
 [non critical]

3.2 Export private key to a file and sign with Certification Authority. (optional)

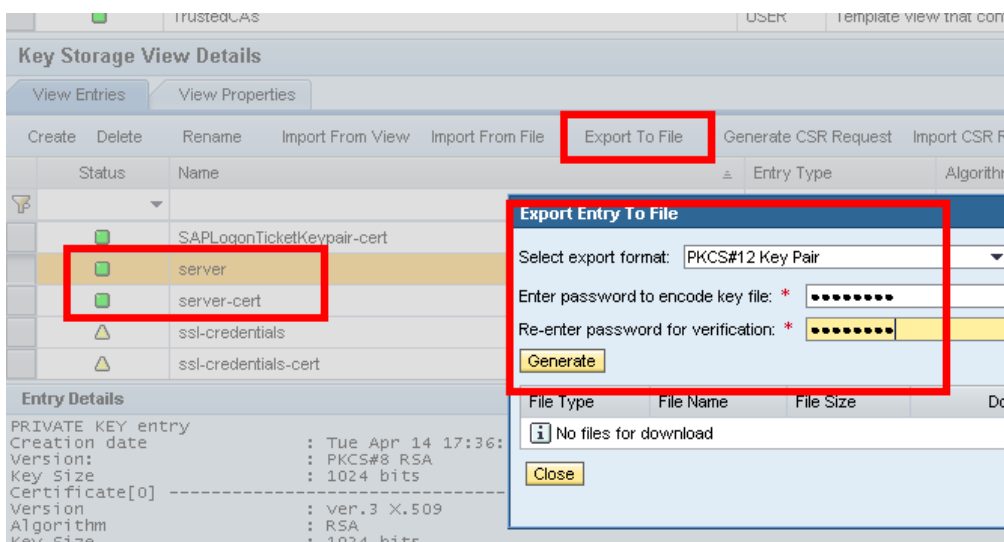
16. Select the created private key and click on “Generate CSR Request”



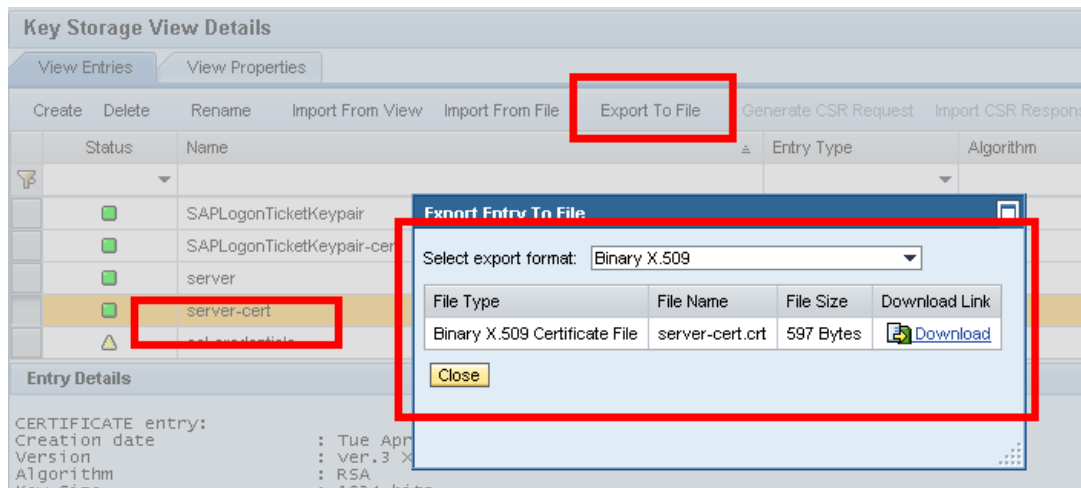
17. Store the file to the file system and send it to your Certification Authority for signing.
18. After getting the response back: Select the Cert you want to update and use the Import CRS Response button for importing
19. The CA Public Key (the CA Root Certificate) associated with this Private Key should be imported as well.

3.3 Export the key pair to files

20. Navigate to *Configuration Management -> Security -> Certificates and Keys -> Key Storage* tab section
21. Select the created credential (private key), press the *Export to File* button... and choose *PK12 format*, specify a password.



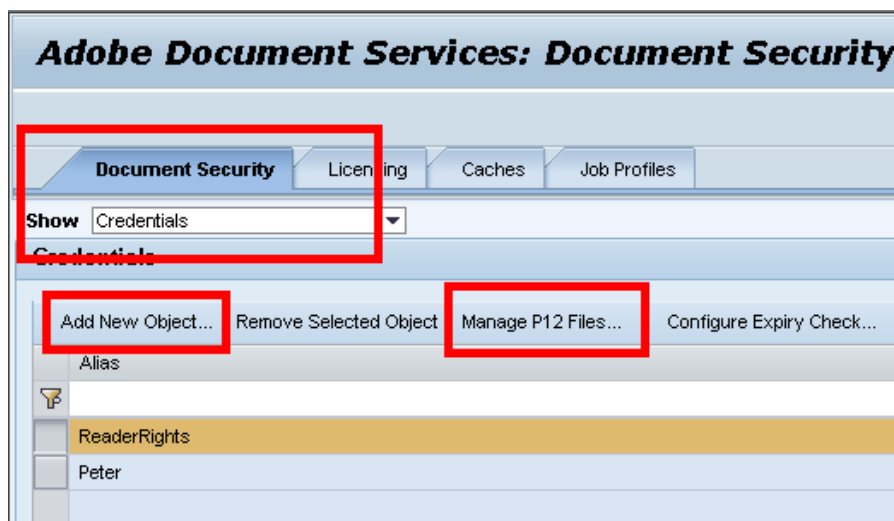
22. Click on Generate, Save the file locally to the file system by selecting the *Download* link, close the wizard.
23. Repeat the steps for the Public Key – certificate.
24. Choose *Binary X.509* format. Note: If you choose Base64 the ADS will NOT be able to validate the certificate!



25. Download the certificate.

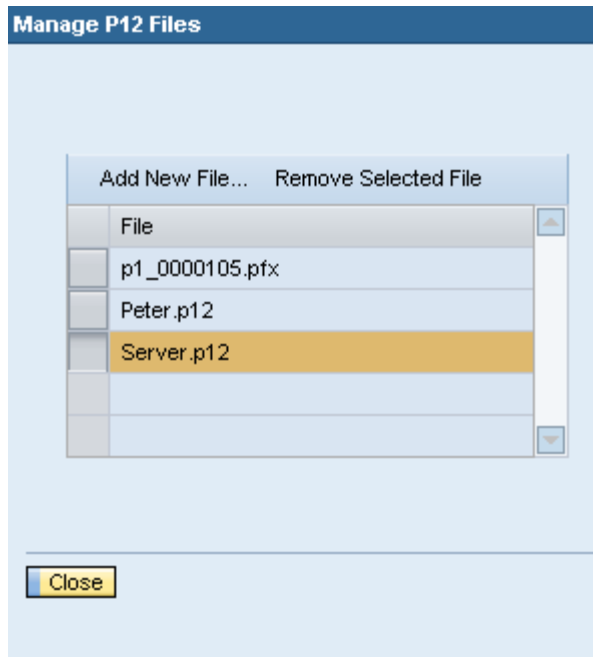
3.4 Import the public key to the ADS credential storage

26. Navigate to *Configuration Management* → *Infrastructure* → *Adobe Document Services*.
27. Select tab *Document security*. Choose *Credentials* from the drop down.

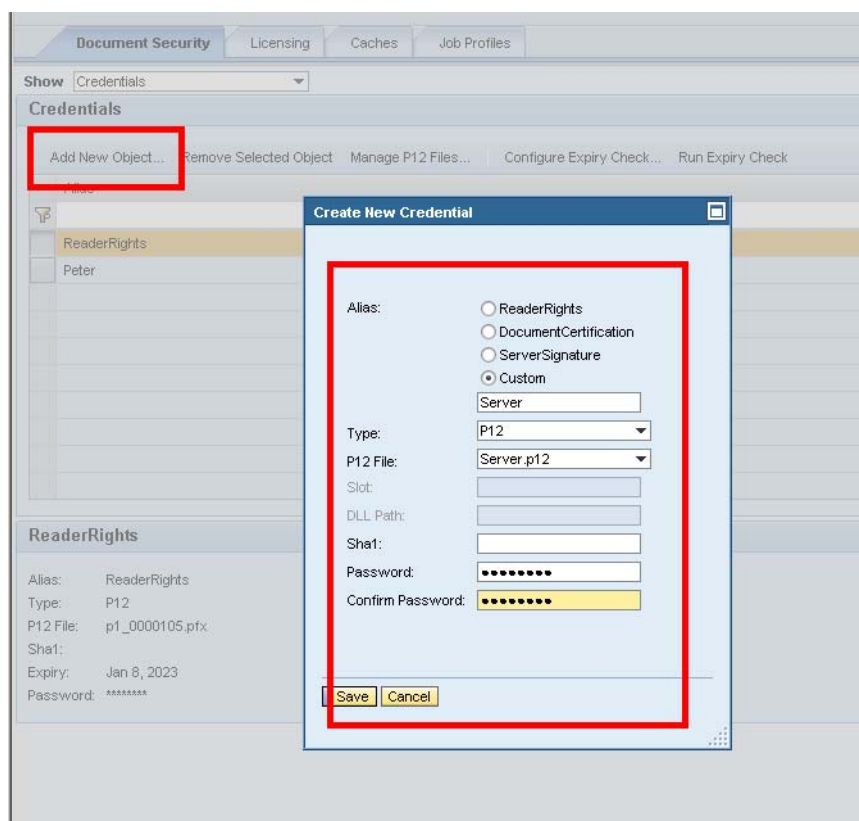


28. Select the *Manage P12 Files* button and click on *Add new file*
29. Navigate to the exported credential (pk12, PFX file).

30. Press the *Upload* button. Close the wizard

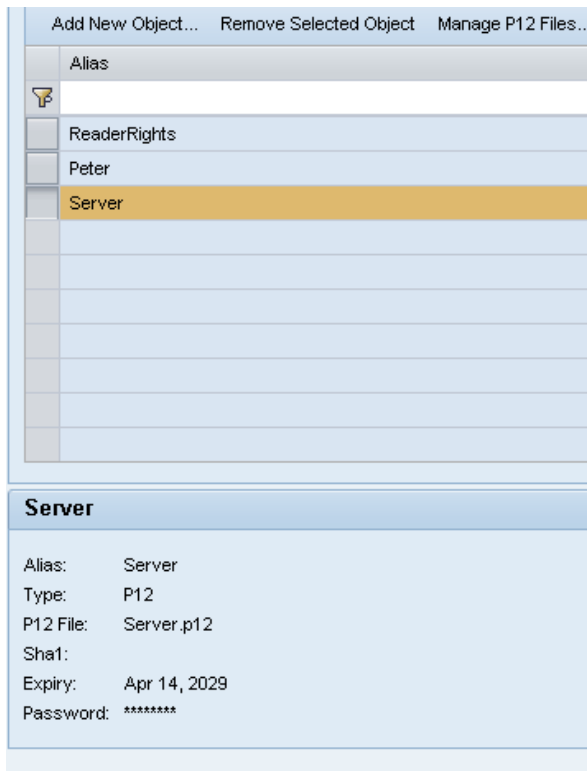


31. Click on *add new Object*



32. Select *Custom Alias* and provide a name. This alias name will be used for identifying the credential that need be applied on the Interactive Form. You can create many different aliases that are required by the business scenarios. Provide the alias names to the people that need to apply these certificates. A certificate can also be automatically applied with the Document API.

33. Choose the credential file. Provide the password and save the Credential alias.

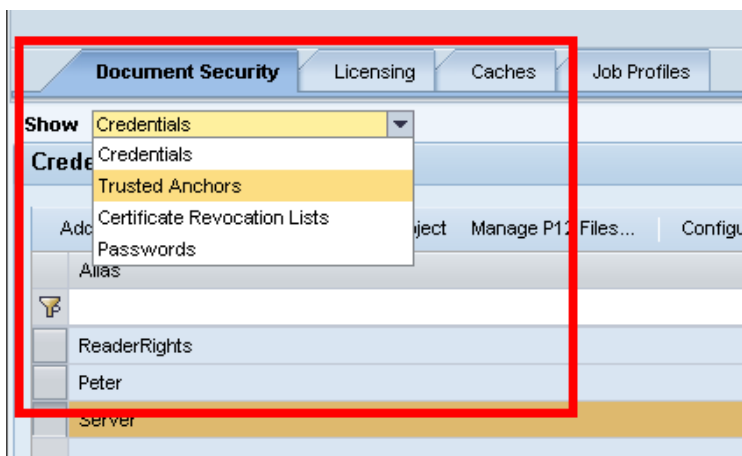


The screenshot shows the Adobe Document Security console. At the top, there are buttons: 'Add New Object...', 'Remove Selected Object', and 'Manage P12 Files...'. Below these is a list of credentials. The 'Server' credential is selected and highlighted in orange. Below the list, the details for the 'Server' credential are shown:

Alias:	Server
Type:	P12
P12 File:	Server.p12
Sha1:	
Expiry:	Apr 14, 2029
Password:	*****

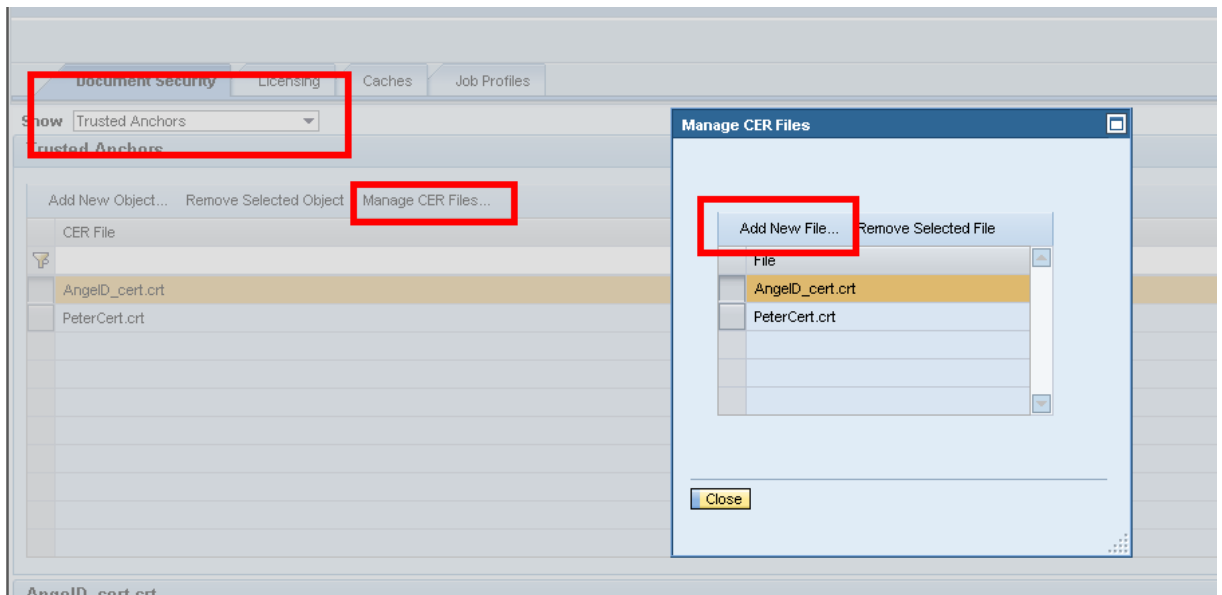
3.5 Import the certificate to the ADS trusted anchors storage

34. From the dropdown choose *Trusted Anchors*

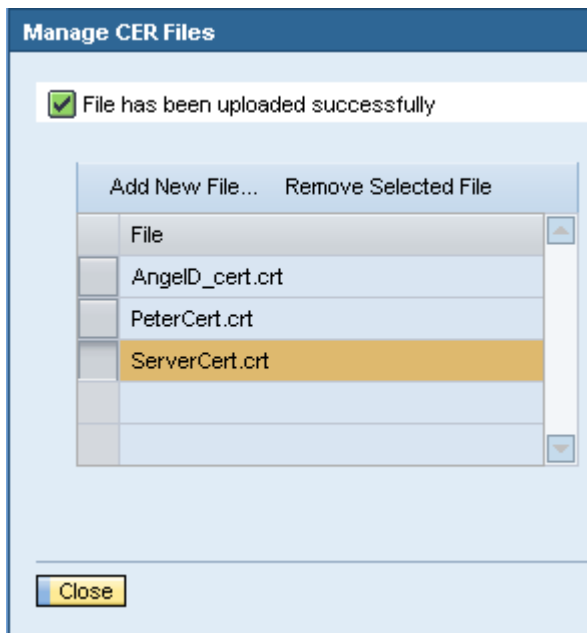


The screenshot shows the Adobe Document Security console. The 'Document Security' tab is selected. Below the tabs, there is a 'Show' dropdown menu. The dropdown menu is open, showing the following options: 'Credentials', 'Trusted Anchors', 'Certificate Revocation Lists', and 'Passwords'. The 'Trusted Anchors' option is highlighted. Below the dropdown menu, there is a list of credentials. The 'Server' credential is selected and highlighted in orange.

35. Search for *SecureConfigPort_Document...* in the Details About *SecureConfigPort_Document* destination section select *Edit* button

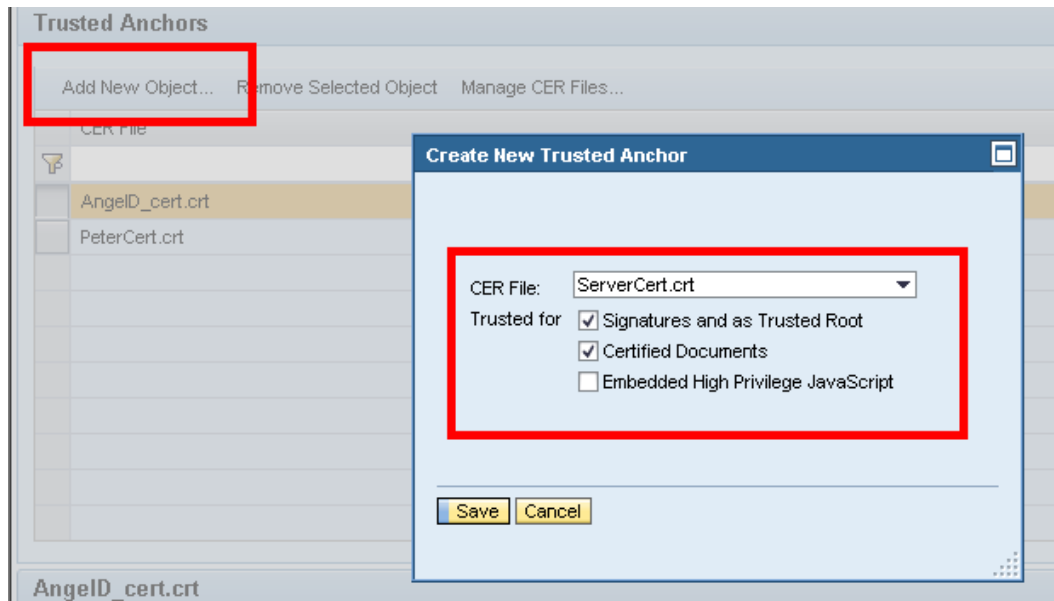


36. Select *Manage CER Files*
37. Choose *Add New File*.
38. Navigate the *certificate* file. Click *Upload*.



39. Close the wizard.

40. Select *Add new object*



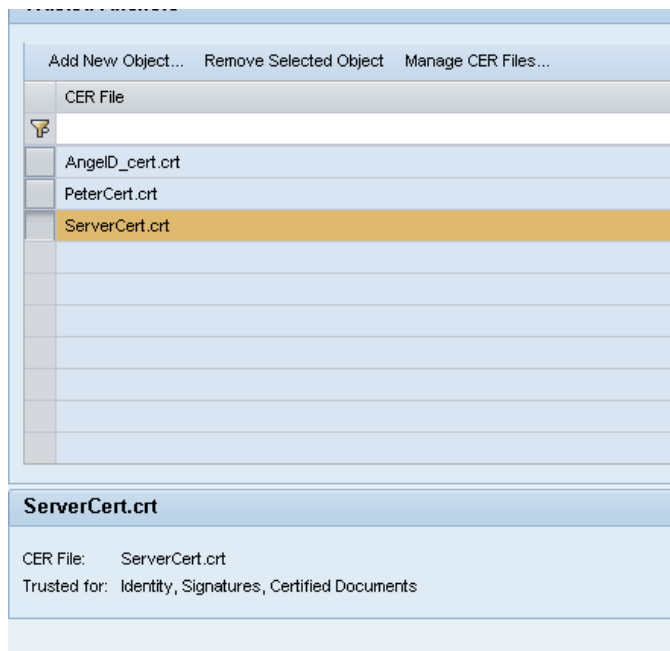
41. Choose the *CER File*

42. Select the checkbox what to trust this certificate for.

Signatures and as Trusted Root

Certificated Documents.

43. Click *Save*



A trusted anchor can be trusted for the following attributes

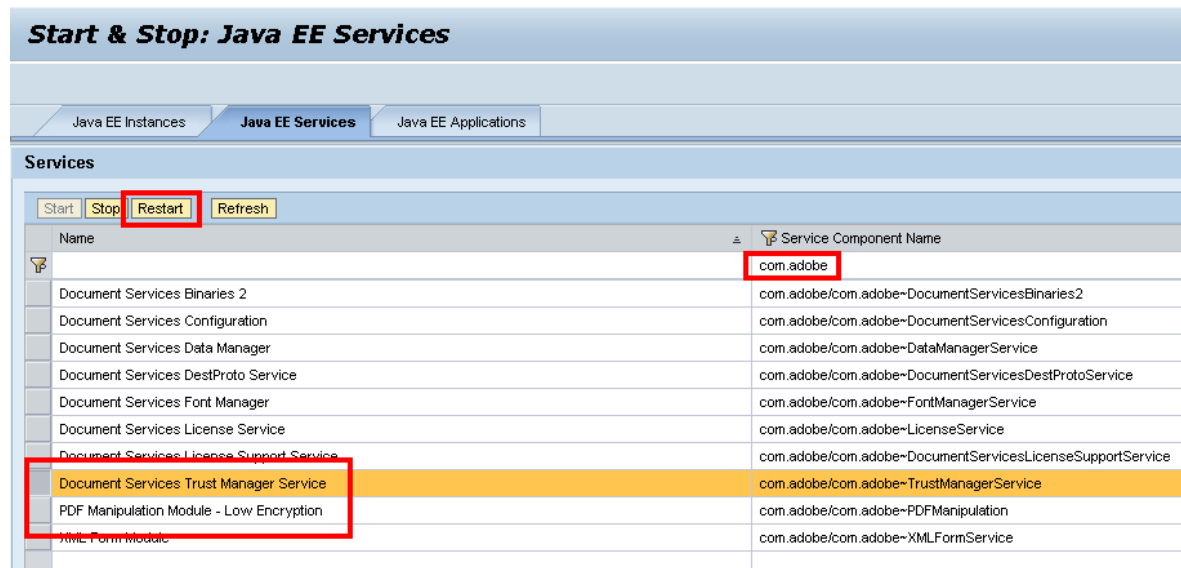
Trusted for	Description
Certified Documents	Documents signed with this signature as an author signature, or whose certificate chain includes this certificate, are considered trusted for certified documents
Embedded High Privilege Java Script	This option is available only if Certified Documents is already selected When this option is enabled, JavaScript embedded in the document is allowed to be executed(*)
Signatures and as trusted root	Documents signed with this signature, or whose certificate chain includes this certificate, are considered trusted for signed documents(**) This option is needed when the document must be signed and signature validated, in case only of a certifying document, only the element Certified Documents is required

Useful combinations of attributes assigned to a certificate.

Certified Document	Signatures and as CA trusted root	Description
X	–	Trust only children certificates for certifying
–	X	Trust certificate itself and children certificates if the certificate is not issued by a CA Trust children certificates for signing if public certificate is issued by a CA
X	X	Trust certificate itself and children certificates for signing and certifying

3.6 Restart The PDF Manipulation Service

44. Navigate to *Operations Management -> Systems -> Start & Stop ...* select the tab *JAVA EE Services*
45. In order these changes to take effect restart these 2 services:
PDF Manipulation Module
Document Services Trust Manager Service.



3.7 Applying a server certificate on Interactive Form

The certificate is applying on the interactive form with the help of the documentAPI.

All credentials are stored in the ADS key store (the steps we did before).

The identification of each credential is done with help of the alias under which the credential is stored.

The validation of the digital signature is done in the same way as the validation of client certificates.

46. Follow this example code for applying and verifying of a server credential.

In this example the alias is provided as an input field, also other properties that are set are reason and contact information.

The example has two actions: One for applying the certificate and one for validation.

Credential Alias:

Contact Info:

Reason:

47. Apply certificate method

```

/**
 * Method declared by application.
 */
/**
 * Demonstrates, how to apply certification on an existing PDF
document.
 */
try {

    // Get the alias name, provided at input field in this example.
    String credentialAlias =
wdContext.currentContextElement().getCredentialAlias();
    // Additional information that could be stored along with the credential
in the signature field.
    //Reason for signing
    String reason = wdContext.currentContextElement().getReason();
    //Locale
    String location = WDCClientUser.getCurrentUser().getLocale().toString();
    //Contact information
    String contactInfo = wdContext.currentContextElement().getContactInfo();

    //set the signature field permission.
    WDPDFDocumentCertificatePermission permission =
WDPDFDocumentCertificatePermission.CERT_DEFAULT_PERMISSION;

    /**
     * Get Interactive Form context. Because, we want to apply the
certification
     * on the PDF document represented by the interactive form UI element
on the
     * page.
     */
    IWDPDFDocumentAccessibleContext iFormContext =
WDPDFDocumentFactory.getDocumentHandler().getDocumentAccessibleContext();

    //Get the PDF from the InteractiveForm UI element
    byte[] pdf = wdContext.currentContextElement().getPdf();

    /**
     * Byte array based access.
     * Set the PDF to the Document Accessible Context
     */
    iFormContext.setPDF(pdf);

    /**
     * Get document context and then set certification on that. You need a
credential
     * to certify a PDF document. Please, consult your ADS administrator
for the
     * same. Default credential (DocumentCertification) will be used if you
do not specify one.
     */
    iFormContext.setCertification(credentialAlias, "SignatureField1",
reason, location, contactInfo, null, permission);

    IWDPDFDocument pdfDocument = iFormContext.execute();

```

```

        //Set the signed PDF back to the InteractiveForm UI element
        wdContext.currentContextElement().setPdf(pdfDocument.getPdf());

    } catch (Exception _e) {

wdThis.wdGetAPI().getComponent().getMessageManager().reportException(_e);
    }
    //@@end
}

```

3.8 Validating server certificates on Interactive Form

48. Application code for validating certificates.

```

//@@begin javadoc:verifyServerSignature()
/**
 * Method declared by application.
 */
//@@end
public void verifyServerSignature( ) {
    //@@begin verifyServerSignature()
    /**
     * This action demonstrates how to use PDF document API in stand alone
mode.
     * Assumed that, you have a PDF document in hand and you need to verify
whether
     * the document has been certified or not. Note, the difference between
the
     * API usage. You deal with an instance of type IWDPDFDocumentHandler
when
     * you want to handle the operation on your own. But, you deal with an
     * instance of IWDPDFDocumentInteractiveFormHandler, when you want to
deal
     * with the InteractiveForm UI element.
     */

    // Get PDF document handler.
    IWDPDFDocumentHandler pdfDocumentHandler =
WDPDFDocumentFactory.getDocumentHandler();

    /**
     * Get accessible context. The other available context under this mode
is
     * creation context. Since, we are going to deal with an existing PDF
document
     * here, we need to have an accessible context.
     *
     * @see onActionCreatePDFStandalone for the usage of creation context.
     */
    IWDPDFDocumentAccessibleContext accessibleContext =
pdfDocumentHandler.getDocumentAccessibleContext();

    // Set PDF stream.
    byte[] pdfStream = wdContext.currentContextElement().getPdf();

```

```

ByteArrayOutputStream outPDF = new ByteArrayOutputStream();
try {
    outPDF.write(pdfStream);
} catch (IOException e) {
    e.printStackTrace();
}

accessibleContext.setPDF(outPDF);

/*
 * We need to have the certificate in response. Hence we need to
instruct the
 * server to send us back the certificate and the related certification
 * information attached with the document. Add the GetCertification task
to
 * the server process. The same holds true for signatures also.
 */
accessibleContext.getTaskSetter().addGetCertificationTask();

/*
 * We need to verify whether the existing PDF document has been signed
or not.
 * To do so, we need to instruct the server process to send us back all
the
 * signatures attached with the document and its related info along with
the
 * document.
 */
accessibleContext.getTaskSetter().addGetSignatureTask();
/*
 * Execute. Note that IWDPDFDocument contains pdf binary (byte array)
and
 * data implicitly. Special task like GetCertification, GetSignature
needs to be
 * added explicitly.
 */
IWDPDFDocument pdfDocument = accessibleContext.execute();

// GetCertification
IWDPDFDocumentCertificate certificate = pdfDocument.getCertification();

//Get all signature fields
IWDPDFDocumentSignature[] signatures = pdfDocument.getSignature();

//Check all signature fields
if(null == certificate){
    wdThis.wdGetAPI().getComponent().getMessageManager().reportWarning(
        "This document has not been certified.");
} else {
    wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Cert
ificate Status: " + certificate.getStatus().toString());

    wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Cert
ificate Validity: " + certificate.isValid());
}

if(null == signatures){

    wdThis.wdGetAPI().getComponent().getMessageManager().reportWarning(

```

```

        "The application is not called though SSL or the document has not
        been signed.");
    }else{

        wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Numb
        er Of Signature Fields attached: " + signatures.length);

        wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Acce
        ssing signature details one by one: ");
        for(int i=0; i<signatures.length; i++){

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Fiel
            d: " + signatures[i].getField());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Stat
            us: " + signatures[i].getStatus().toString());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Sign
            er: " + signatures[i].getSigner());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Loca
            tion: " + signatures[i].getLocation());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Date
            : " + signatures[i].getDate());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Cont
            act Info: " + signatures[i].getContactInfo());

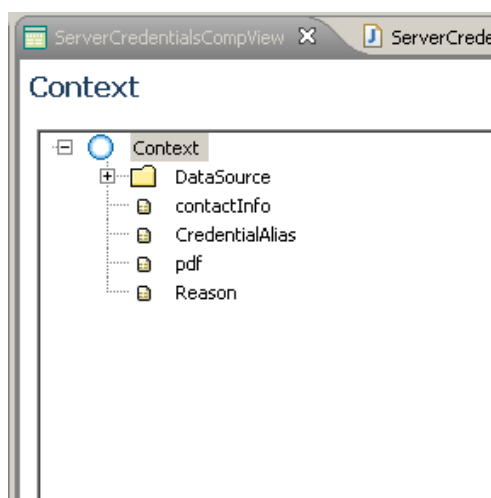
            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("Reas
            on: " + signatures[i].getReason());

            wdThis.wdGetAPI().getComponent().getMessageManager().reportSuccess("isVa
            lid: " + signatures[i].isValid());
        }
    }

    //@@end
}

```

49. Context elements: Certificate Alias name, pdf (binary), Reason, ContactInformation



50. Signature field name on form 'SignatureField1' – The same as the one in step 47 :
setCertification()

The screenshot displays the Adobe Acrobat Designer interface. On the left, the 'Tree' pane shows the form's structure, including a 'header' section with various text fields, a 'detail' section with a table, and a 'total' section. The 'SignatureField1' is highlighted in the 'total' section. On the right, a preview of the form is shown, featuring a table with columns 'Part No.' and 'Description', and a signature field labeled 'Authorized By (Signature)'.

```
iFormContext.setCertification(credentialAlias, "SignatureField1",
reason, location, contactInfo, null, permission);
```

3.9 Demo application

51. Executing the application via non-secured call.

Calling the setCertificate() method in a non secured mode will throw an exception pointing to the need of SSL connection:

ServerCredentials - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Reload Home Search Favorites Print Mail New Window New Tab

Address http://vmphlrig074:50000/webdynpro/dispatcher/demo.sap.com/ifba_server_signatures/ServerCredentials?SA Go Links

Purchase Order

P.O. Number **1234567890**
P.O. Date **Apr 22, 2009**

Ordered By
San Jose, CA

Deliver To
Palo Alto, CA

Phone Number:
Contact Name: Adobe

Phone Number:
Contact Name: SAP

Part No.	Description	Quantity	Unit Price	Amount
011-1234	Adobe Designer	3	\$200.00	\$600.00
Terms and Conditions			Total	\$600.00

Authorized By (Signature)

Credential Alias:
Contact Info:
Reason:

Certify Document

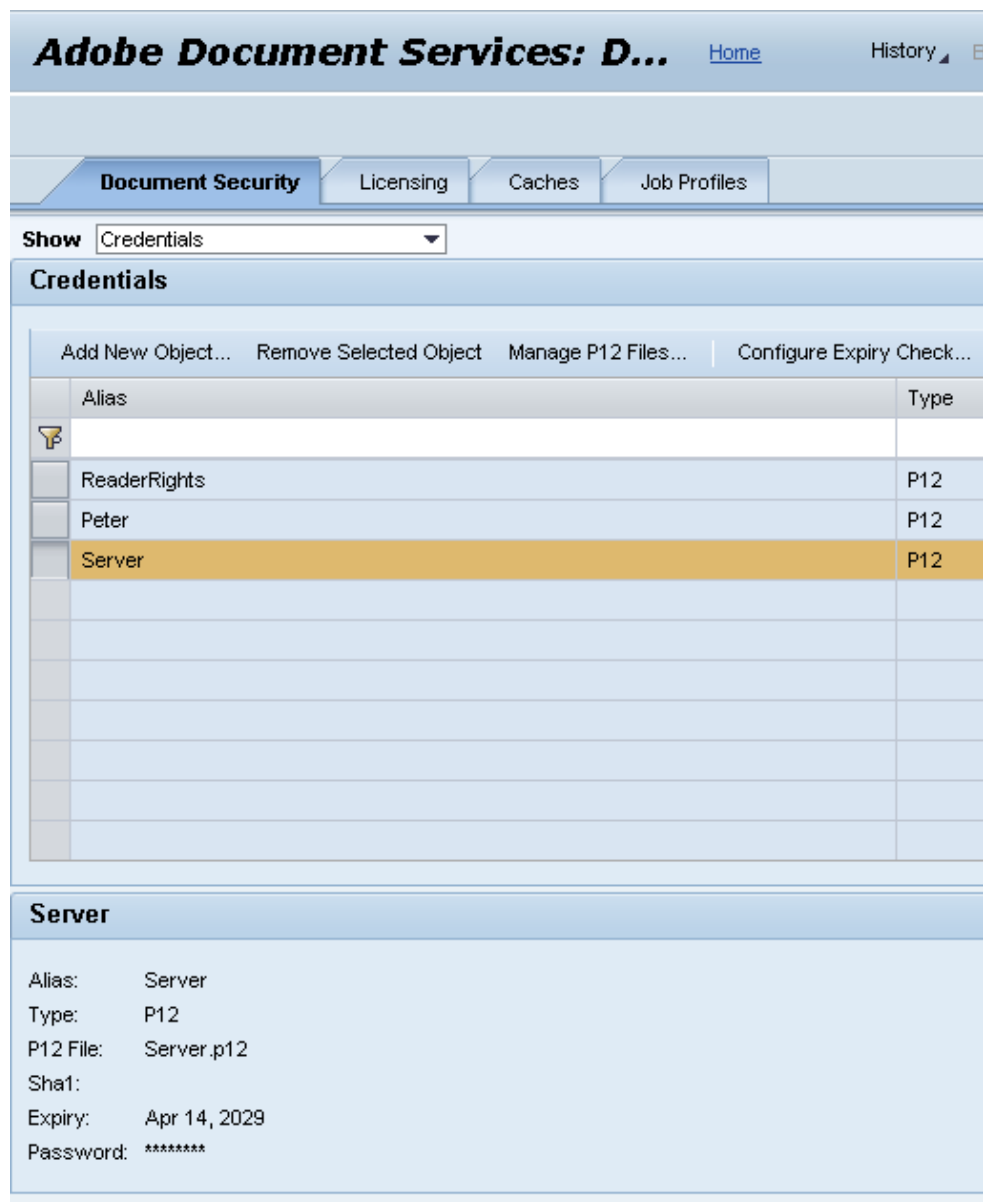
Verify Server Credential

Processing exception during a "Certify" operation. Request start time: Wed Apr 22 15:32:14 EDT 2009 com.adobe.ProcessingException: com.adobe.ProcessingException: <Certify> requires a secure connection. Please install the high encryption PDF Manipulation service and enable SSL. Exception Stack Trace: com.adobe.ProcessingException: com.adobe.ProcessingException: <Certify> requires a secure connection. Please install the high encryption PDF Manipulation service and enable SSL. at com.adobe.ads.request.ADSRequest.processOperations(Unknown Source) at com.adobe.ads.request.ADSRequest.process(Unknown Source) at com.adobe.AdobeDocumentServicesEJB.processRequest(Unknown Source) at com.adobe.AdobeDocumentServicesEJB.rpData(Unknown Source) at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39) at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at java.lang.reflect.Method.invoke(Method.java:585) at

Done Local intranet

52. Call the application with SSL.

The Alias that will be used is called *Server*:



The screenshot shows the Adobe Document Services: D... interface. The top navigation bar includes "Home" and "History". The "Document Security" tab is selected, with other tabs for "Licensing", "Caches", and "Job Profiles". A "Show" dropdown menu is set to "Credentials". Below this, the "Credentials" section contains a table with columns "Alias" and "Type". The table lists three entries: "ReaderRights" (P12), "Peter" (P12), and "Server" (P12). The "Server" entry is highlighted. Below the table, the "Server" section displays the following details:

Alias	Type
ReaderRights	P12
Peter	P12
Server	P12

Server details:

- Alias: Server
- Type: P12
- P12 File: Server.p12
- Sha1:
- Expiry: Apr 14, 2029
- Password: *****

Before applying the certificate:

ServerCredentials - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Reload Home Search Favorites RSS Mail Print Folders People

Address https://vmphlrig074:50001/webdynpro/dispatcher/demo.sap.com/ifba_server_signatures/ServerCredentials?S Go

Please fill out the following form. You can save data typed into this form. Highlight

Purchase Order

Ordered By
San Jose, CA

Phone Number:

Contact Name: Adobe

P.O. Number 1234567890

P.O. Date Apr 22, 2009

Deliver To
Palo Alto, CA

Phone Number:

Contact Name: SAP

Part No.	Description	Quantity	Unit Price	Amount
011-1234	Adobe Designer	3	\$200.00	\$600.00
Terms and Conditions			Total	\$600.00

Authorized By (Signature)

Credential Alias:

Contact Info:

Reason:

After applying the certificate:

Alias – Server

Contact info – test@sap.com

Reason: *PO Approved*

Click on *Certify Document*

ServerCredentials - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Reload

Address https://vmphlrig074:50001/webdynpro/dispatcher/demo.sap.com/ifba_server_

Purchase Order

P.O. N
P.O. I

Ordered By
San Jose, CA

Deliv
Palo J

Phone Number:

Phone

Contact Name: Adobe

Conti

Part No.	Description	Qu
011-1234	Adobe Designer	
Terms and Conditions		

server

Digitally signed by server
DN: cn=server, o=SAP, ou=IT, email=test@sap.com, c=US
Reason: PO Approved
Location: US
Date: 2008.04.23 10:00:00 -0700

Authorized By (Signature)

Credential Alias: Server

Contact Info: test@sap.com

Reason: PO Approved

Certify Document

Verify Server Credential

Signature field properties:

Reason – PO Approved ,

Location – Taken from the *User Locale* – see location in *setCertification()* method

The screenshot shows the 'Document' tab of the Signature Properties dialog. At the top, a warning icon and text state: 'Document was certified, validity is UNKNOWN.' Below this are tabs for 'Summary', 'Document', 'Signer', 'Date/Time', and 'Legal'. The 'Summary' tab is active. It contains the following fields: 'Signed by:' with the value 'server', a 'Show Certificate...' button, 'Reason:' with the value 'PO Approved', 'Date:' with the value '2009/04/22 13:05:03 -07'00'', and 'Location:' with the value 'en_US'. A 'Validity Summary' section contains three items: a blue ribbon icon with the text 'The changes that have been made to this document since it was certified are permitted by the Certifying party and do not invalidate the signature.', a warning icon with the text 'The signer's certificate is a self-signed certificate that is not trusted.', and another warning icon with the text 'Signature date/time are from the clock on the signer's computer.'

The *Signer's Contact* - test@sap.com under the Signer tab

The screenshot shows the 'Signer' tab of the Signature Properties dialog. It contains a warning icon and text: 'The signer's certificate is a self-signed certificate that is not trusted.' Below this is the 'Signed by:' field with the value 'server' and a 'Show Certificate...' button. An information icon and text state: 'Click Show Certificate for more information about the signer's certificate and its validity details, or to change the trust settings for the certificate or an issuer certificate.' A 'Validity Details' section contains three items: a warning icon with the text 'The signer's certificate is a self-signed certificate that is not trusted.', a green checkmark icon with the text 'Path validation checks were successful.', and a warning icon with the text 'Revocation checking was not performed.' At the bottom, the 'Signer's Contact Information:' field contains the value 'test@sap.com'. A warning icon and text at the very bottom state: 'When you directly trust a signer's certificate that is not issued by a root certificate authority that you trust, you should contact the signer to verify the certificate. Once you are confident that the signer is who he/she reports to be, then verify that the certificate is from the signer. For example, you can confirm the certificate's MDE direct with the signer. (Use the Certificate Viewer to view

Click on *Verify Server Credential*

ServerCredentials - Microsoft Internet Explorer

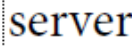
File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Reload Print Mail

Address https://vmphlrig074:50001/webdynpro/dispatcher/demo.sap.com/ifba_server_signatures/ServerCredent

Ordered by San Jose, CA	Delivered to Palo Alto, CA
Phone Number:	Phone Number:
Contact Name: Adobe	Contact Name: SAP

Part No.	Description	Quantity	Unit Price
011-1234	Adobe Designer	3	\$200.00
Terms and Conditions			Total



Authorized By (Signature)

Credential Alias:

Contact Info:

Reason:

Certify Document

Verify Server Credential

- ☒ Certificate Status: Valid
- ☒ Certificate Validity: true
- ☒ Number Of Signature Fields attached: 1
- ☒ Accessing signature details one by one:
- ☒ Field: SignatureField1
- ☒ Status: Valid
- ☒ Signer: server
- ☒ Location: en_US
- ☒ Date: Wed Apr 22 2009 16:05:03 GMT-0400 (Eastern Daylight Time)
- ☒ Contact Info: test@sap.com
- ☒ Reason: PO Approved
- ☒ isValid: true

53. The entire application project is attached to this document. Please look at the PDF attachments: demo.sap.com.ziparchive. Please change extension to .zip or de-archive with ZIP tool.

www.sdn.sap.com/irj/sdn/howtoguides