Digital Signatures & Rights Management in the Acrobat Family of Products

Guides for administrators and advanced users:

Administration
Digital Signatures
Document Security
Contents

1 The Basics ................................................................................................................................................. 13
  1.1 Why are You Here? .................................................................................................................................. 13
    1.1.1 Because You Care About Information Assurance ................................................................. 13
    1.1.2 Because You're Technical ........................................................................................................... 14
    1.1.3 Because You're Lost? .................................................................................................................... 14
  1.2 What's new for 9.1? ............................................................................................................................. 15
  1.3 Basic Concepts ...................................................................................................................................... 17
    1.3.1 What is Security and Information Assurance? ........................................................................... 17
    1.3.2 What is Trust? ............................................................................................................................... 17
    1.3.3 What is a Trusted Identity? ........................................................................................................... 18
    1.3.4 What is a Digital ID? ..................................................................................................................... 19
    1.3.5 Digital ID Storage Mechanisms ................................................................................................. 21

Administration Guide................................................................................................................................. 23

2 Predeployment Installer Tuning .................................................................................................................. 24
  2.1 Getting Started with the Wizard ......................................................................................................... 24
  2.2 Application Defaults ............................................................................................................................. 25
    2.2.1 Signing Environment Defaults ................................................................................................... 25
    2.2.2 Signature Validation Defaults .................................................................................................... 26
    2.2.3 Document Security Defaults ...................................................................................................... 27
  2.3 Wizard Setup ....................................................................................................................................... 28
  2.4 Basic Installer Tuning ........................................................................................................................... 29
  2.5 Advanced Installer Tuning for Power Users ......................................................................................... 32

3 Migrating and Sharing Security Settings ................................................................................................... 35
  3.1 Security Setting Import and Export ................................................................................................... 35
    3.1.1 Exporting Security Settings to a File ......................................................................................... 36
    3.1.2 Importing Security Settings from a File .................................................................................... 37
    3.1.3 Importing Security Settings from a Server ............................................................................... 38
  3.2 Sharing Settings & Certificates with FDF .......................................................................................... 39
    3.2.1 FDF Files and Security ............................................................................................................... 40
    3.2.2 Exporting Application Settings with FDF Files ......................................................................... 41
      3.2.2.1 Distributing a Trust Anchor or Trust Root ......................................................................... 42
      3.2.2.2 Setting the Certificate Trust Level ..................................................................................... 45
      3.2.2.3 Exporting Your Certificate ................................................................................................ 45
      3.2.2.4 Emailing Your Certificate ................................................................................................ 46
      3.2.2.5 Saving Your Digital ID Certificate to a File ................................................................. 47
      3.2.2.6 Requesting a Certificate via Email .................................................................................... 48
      3.2.2.7 Emailing Server Details ................................................................................................... 49
      3.2.2.8 Exporting Server Details ................................................................................................ 51
    3.2.3 Importing Application Settings with FDF Files .......................................................................... 52
      3.2.3.1 Responding to an Email Request for a Digital ID ........................................................... 52
      3.2.3.2 Importing Someone's Certificate ................................................................................... 54
4 Post Deployment Infrastructure Integration ................................................................. 66

4.1 LDAP Directory Server Integration ............................................................................ 66
  4.1.0.1 Configuring an LDAP Server to Look Up Certificates ........................................ 67
  4.1.0.2 Configuring Clients to Search a Directory Server for Certificates .................. 67

4.2 Setting up Signature Validation .................................................................................... 69

4.3 Digital ID Management ................................................................................................. 70
  4.3.1 PKCS#11 Integration ............................................................................................ 70
  4.3.2 .......................................................................................................................... 70
  4.3.3 Lifecycle Management ......................................................................................... 70

4.4 OIDs and Certificates ................................................................................................ 71
  4.4.1 What is an Object Identifier? (OIDs) .................................................................... 71
  4.4.2 Adobe OIDs ....................................................................................................... 71
  4.4.3 Security OIDs .................................................................................................... 72
  4.4.4 Certificate Practice Statement OIDs ................................................................. 72
  4.4.5 Ubiquity OIDs ................................................................................................. 73
  4.4.6 X.509 Extension OIDs ..................................................................................... 74

5 Registry and plist Settings .............................................................................................. 75

5.1 Setting Basic Client and Workflow Preferences ......................................................... 75
  5.1.1 Preventing End-User Modification ...................................................................... 75
  5.1.2 Importing Security Settings ............................................................................... 78
  5.1.3 Digital ID Management ....................................................................................... 80
    5.1.3.1 Specifying extended certificate information .................................................. 80
    5.1.3.2 Allowing Self Sign Digital IDs ...................................................................... 80
    5.1.3.3 PKCS#11 Configuration .............................................................................. 81
    5.1.3.4 Roaming ID Configuration ............................................................................ 81
    5.1.3.5 Setting Digital ID Defaults ......................................................................... 84
    5.1.3.6 Digital ID Import and Export Paths ........................................................... 86
  5.1.4 Certificate Management ....................................................................................... 87
    5.1.4.1 Windows Integration ................................................................................... 87
    5.1.4.2 Trusted Identity List Configuration ............................................................ 88
    5.1.4.3 Custom Certificate Preferences ................................................................... 89
  5.1.5 Custom Security Handler Preferences ................................................................. 93
  5.1.6 ASPKI Service Provider Interfaces ...................................................................... 95
    5.1.6.1 Revocation Checker Providers .................................................................... 95
    5.1.6.2 Signature Validation Directory Providers .................................................. 96
    5.1.6.3 Roaming ID Authentication Mechanisms .................................................. 97
  5.1.7 FDF Import and Export ....................................................................................... 97
  5.1.8 Security Settings Console .................................................................................... 98
  5.1.9 Displaying All Chains in the Certificate Viewer ................................................ 99
  5.1.10 Configuring Password Caching ....................................................................... 99
  5.1.11 Turning on FIPS Mode .................................................................................... 100

5.2 Content Security ........................................................................................................ 101
  5.2.1 Configuring Certificate Handling for Certificate Security .................................. 101
6 Digital Signatures: Technical Overview .................................................................................. 132

6.1 Signature Technical Details .................................................................................................. 132
  6.1.1 How Signing Happens ........................................................................................................ 133
  6.1.2 What’s Different with XML Forms? .................................................................................... 134
  6.1.3 Signature Validation ............................................................................................................. 135
  6.1.4 Signature Format ................................................................................................................ 135
  6.1.5 Signatures and Permissions ................................................................................................ 135
    6.1.5.1 Locking a Document During Signing .............................................................................. 137

6.2 Differences Between Signature Types ................................................................................. 138
  6.2.1 Standard PDF and Static XML Forms ................................................................................. 138
    6.2.1.1 Approval Signature ........................................................................................................ 138
    6.2.1.2 Certification Signature (DocMDP) ................................................................................ 139
    6.2.1.3 Signature with Field Restrictions (FieldMDP) ............................................................. 139
  6.2.2 Requiring FIPS-Compliant Encryption Algorithms .......................................................... 139
  6.2.3 Setting up Security Policy Favorites .................................................................................. 140
  6.2.4 Configuring LiveCycle Right Management Servers ....................................................... 140
  6.2.5 Security Envelopes ............................................................................................................ 144

5.3 Digital Signatures .................................................................................................................. 104
  5.3.1 Revocation Checking Preferences ..................................................................................... 104
    5.3.1.1 Configuring OCSP Revocation Checking ...................................................................... 105
    5.3.1.2 Configuring CRL Revocation Checking ....................................................................... 108
    5.3.1.3 Certificate Chain Building ........................................................................................... 110
  5.3.2 Signature Creation Preferences ........................................................................................ 111
    5.3.2.1 Changing the Default Hashing Algorithm ................................................................. 111
    5.3.2.2 Specifying a Signing Format ......................................................................................... 112
    5.3.2.3 Configuring Revocation Checking for Signing ............................................................. 113
    5.3.2.4 Embedding Revocation Data in Signatures ................................................................. 113
    5.3.2.5 Requiring a Pre-Signing Digest Comparison ................................................................. 114
    5.3.2.6 Requiring Signature Property Retrieval ....................................................................... 115
    5.3.2.7 Setting up Timestamp Servers .................................................................................... 116
  5.3.3 Signing Workflow Preferences ........................................................................................ 118
    5.3.3.1 Requiring Preview Mode to Sign ................................................................................ 119
    5.3.3.2 Specifying Signature Appearances ............................................................................. 119
    5.3.3.3 Preconfiguring Location and Contact Information ....................................................... 119
    5.3.3.4 Specifying Signing Reasons ......................................................................................... 120
    5.3.3.5 Certification Signature Preferences ............................................................................ 121
    5.3.3.6 Controlling Signature Warnings: Review ................................................................. 122
    5.3.3.7 Controlling Signature Warnings: Fonts ...................................................................... 123
  5.3.4 Signature Validation Preferences ..................................................................................... 124
    5.3.4.1 Setting Validation Preferences that Map to UI .......................................................... 124
    5.3.4.2 Certificate Chain Validation Method ........................................................................... 125
    5.3.4.3 Controlling Signature Status Icon Behavior .............................................................. 126
    5.3.4.4 Logging Certificate Validation Data ............................................................................ 127
    5.3.4.5 Using Embedded Validation Data .............................................................................. 128
    5.3.4.6 Revocation Checking Constraints ............................................................................. 129
  5.3.5 Document Status Dialog (deprecated) ........................................................................... 129
  5.3.6 Examine Document .......................................................................................................... 130

Digital Signatures .......................................................................................................................... 131
7 Certificate Processing .......................................................................................................................... 144

7.1 Path Discovery: Building the Chain ...................................................................................................... 144
  7.1.1 Cross-Certificates .......................................................................................................................... 146
    7.1.1.1 Chain Building FAQs ................................................................................................................ 148

7.2 Path Validation: Checking a Chain’s Health ......................................................................................... 150

7.3 Establishing Certificate Trust .............................................................................................................. 150

7.4 Certificate Revocation Checking ......................................................................................................... 150
  7.4.1 Revocation Types .......................................................................................................................... 152
  7.4.2 Revocation Checking Order of Precedence .................................................................................. 152
  7.4.3 Timestamp Revocation Checking .................................................................................................. 153
  7.4.4 Revocation Checking Default Behavior ......................................................................................... 154
    7.4.4.1 OCSP Default Behavior .............................................................................................................. 154
    7.4.4.2 CRL Processing Default Behavior ............................................................................................. 155
    7.4.4.3 MSCAPI Default Behavior ........................................................................................................ 157
  7.4.5 Interaction of iReqRevCheck Settings ............................................................................................. 158
    7.4.5.1 Revocation Checks During Signature Creation ........................................................................ 158
    7.4.5.2 Revocation Checks During Signature Validation ...................................................................... 159
  7.4.6 Customized Revocation Check Example ......................................................................................... 160
  7.4.7 Revocation Tab GUI and Strings ................................................................................................... 163

8 Authoring Signable Documents .............................................................................................................. 170

8.1 Best Practices for Signed Documents that will Change .................................................................... 170

8.2 Setting up the Signing Environment .................................................................................................. 170
  8.2.1 Setting Signing Preferences .......................................................................................................... 171
    8.2.1.1 Requiring Preview Mode ........................................................................................................... 171
    8.2.1.2 Changing the Default Signing Method ......................................................................................... 172
    8.2.1.3 Embedding Signature Revocation Status ................................................................................... 173
    8.2.1.4 Allowing Signing Reason ........................................................................................................... 174
    8.2.1.5 Showing Location and Contact Details ....................................................................................... 174
    8.2.1.6 Enabling Document Warning Review ....................................................................................... 174
    8.2.1.7 Requiring Document Warning Review Prior to Signing ........................................................... 175
    8.2.1.8 Enabling a Warnings Comment or Legal Attestation ................................................................. 175
  8.2.2 Signature Appearance Configuration ............................................................................................... 176
    8.2.2.1 Custom Signature Appearance .................................................................................................. 176
    8.2.2.2 Custom Watermark and Backgrounds ....................................................................................... 178
    8.2.2.3 Editing or Deleting a Signature Appearance ............................................................................. 179
  8.2.3 Timestamps for Signing .................................................................................................................. 179

8.3 Working with Signature Fields ............................................................................................................ 181
  8.3.1 Creating a Blank Signature Field ................................................................................................... 181
  8.3.2 Specifying General Field Properties ............................................................................................ 182
  8.3.3 Customizing Field Appearances ................................................................................................... 183
  8.3.4 Changing the Default Field Appearance ...................................................................................... 184
  8.3.5 Cut, Copy, and Paste Signature Fields ............................................................................................ 184
  8.3.6 Arranging Signature Fields ............................................................................................................ 185
12.1.1 What Makes a Signature Valid? ........................................................................................................... 232
  12.1.1.1 Authenticity Verification ........................................................................................................... 232
  12.1.1.2 Document Integrity Verification ........................................................................................................... 233
12.2 Setting up Your Environment for Signature Validation ........................................................................ 234
  12.2.1 Validating Signatures Automatically ................................................................................................. 234
  12.2.2 Method, Revocation Checking, and Validation Time Preferences .................................................. 235
  12.2.3 Using Root Certificates in the Windows Certificate Store .............................................................. 236
  12.2.4 Validating Signatures with Timestamps and Certificate Policies .................................................. 237
12.3 Validating Signatures Manually ............................................................................................................ 238
  12.3.1 Validating Signatures with Adobe Reader ................................................................................................. 239
  12.3.2 Validating a Single Signature in Acrobat ............................................................................................... 239
  12.3.3 Validating All Signatures in Acrobat ........................................................................................................... 239
  12.3.4 Validating an Problematic Signature (trusting a signer on-the-fly) ................................................... 240
  12.3.5 Validating Signatures for other Document Versions ............................................................................ 243
  12.3.6 Validating Signature Timestamps ........................................................................................................... 243
    12.3.6.1 When Timestamps Can’t be Verified ........................................................................................................... 246
12.4 Status Icons and Their Meaning ............................................................................................................ 246
  12.4.1 Signature Status Definitions ............................................................................................................. 246
  12.4.2 Document Status Definitions ............................................................................................................. 246
12.5 Troubleshooting a Signature or Document Status ................................................................................ 247
  12.5.1 Troubleshooting an Identity Problem ....................................................................................................... 247
    12.5.1.1 Troubleshooting Digital ID Certificates ................................................................................................. 248
    12.5.1.2 Displaying the Signer’s Certificate ........................................................................................................... 249
    12.5.1.3 Verifying the Identity of Self-Signed Certificates .................................................................................... 250
    12.5.1.4 Checking Certificate Revocation Status ........................................................................................... 251
    12.5.1.5 Exporting a Certificate Other than Yours to a File ........................................................................... 252
  12.5.2 Troubleshooting a Document Integrity Problem ................................................................................ 252
    12.5.2.1 LiveCycle Dynamic Forms and the Warning Triangle .................................................................... 253
    12.5.2.2 Viewing and Comparing Changes and Versions ............................................................................. 254
    12.5.2.3 Viewing a List of Post-Signing Modifications ..................................................................................... 254
    12.5.2.4 Comparing a Signed Version to the Current Version ........................................................................ 255
12.6 Document Behavior After Signing ...................................................................................................... 256
  12.6.1 JavaScript and Dynamic Content Won’t Run ...................................................................................... 257
  12.6.2 Certifying a Document is Prevented ...................................................................................................... 257
  12.6.3 Form Field Fill in, Signing, and/or Other Actions Don’t Work .......................................................... 257
13 Document Integrity and Preview Mode .................................................................................................... 258
  13.1 Preview Mode and Signing Workflows ..................................................................................................... 258
  13.2 Preview Mode and Validation (View Signed Version) .......................................................................... 259
  13.3 PDF Signature Reports ........................................................................................................................... 259
  13.4 Signature Report Error Codes .................................................................................................................. 261
Document Security (Rights Management) .................................................................................................. 264
14 Document Security Basics .......................................................................................................................... 265
  14.1 Security Method Basics .......................................................................................................................... 265
    14.1.1 Choosing a Security Method Type ......................................................................................................... 266
    14.1.2 Security Policies .................................................................................................................................. 267
    14.1.3 Security Methods and Encryption ......................................................................................................... 267
    14.1.3.1 Encryption Workflow .......................................................................................................................... 267
B.12 Changes Across Releases ............................................................................................................ 326
B.12.1 Digital ID Related Files and Storage Mechanisms..................................................................................... 326
B.12.2 Encryption Algorithms ........................................................................................................................................ 327
B.12.3 Algorithms Used During Digital Signature Creation ................................................................................ 329
B.12.4 Seed Values ............................................................................................................................................................. 329
B.11.1 Digital ID Management ....................................................................................................................................... 324
B.11.2 Certificate Processing and Viewing Enhancements ................................................................................ 324
B.11.3 Signature Enhancements ................................................................................................................................... 324
B.11.4 Document Security Enhancements ............................................................................................................. 325
B.11.5 Application Environment Preference Improvements .................................................................................. 325
B.10 What’s New for 8.1 ................................................................................................................................................ 322
B.9 What’s new for 9.0 ......................................................................................................................................................... 322
B.9.1 Enhanced security ................................................................................................................................................. 319
B.9.2 Security setting import and export ................................................................................................................ 319
B.9.3 Signatures and signing workflows ................................................................................................................. 319
B.9.4 FIPS ............................................................................................................................................................................. 321
B.9.5 Certificate Security .................................................................................................................................................. 321
B.9.6 Reader Enablement ................................................................................................................................................. 321
B.9.7 Password security .................................................................................................................................................... 321
B.9.8 Adobe Reader usage rights and feature enablement ......................................................................................... 322
B.9.9 Security and Encryption for of PDF packages (portfolios) ............................................................................... 322
B.9.10 Signing of PDF packages (portfolios) ................................................................................................................ 322
B.9.11 Attachments ........................................................................................................................................................... 322
B.9.12 Product renaming ............................................................................................................................................... 322
B.8 What’s new for 9.1 ......................................................................................................................................................... 322
B.7 What’s new for 9.2 ......................................................................................................................................................... 319
B.6 Security workflow for encryption-permissions .............................................................................................. 318
A.5 Signature Status and Troubleshooting Guide ................................................................................................. 317
A.2 Signing and the Byte Range ................................................................................................................................. 314
A.1 Signature Creation Workflow ............................................................................................................................... 313
A.6 One Page Keys and Guides ................................................................................................................................. 312
A.5.1 Managing your ALCRMS Account ..................................................................................................................... 308
A.5.2 Creating an ALCRMS Security Policy .............................................................................................................. 308
A.5.3 Applying ALCRMS Security ............................................................................................................................. 309
A.5.4 Refreshing the Security Policy List ................................................................................................................ 309
A.5.5 Synchronizing a Document for Offline Use ................................................................................................. 310
A.5.6 Revoking a Document ......................................................................................................................................... 310
A.5.7 View a Document’s Audit History ................................................................................................................ 310
A.4 Revocation Checking Process ................................................................................................................................. 316
A.3 Signatures, Permissions, and the PDF Language ................................................................................................. 315
A.2 Signing and the Byte Range ................................................................................................................................. 314
A.1 Signature Creation Workflow ............................................................................................................................... 313
A. One Page Keys and Guides ................................................................................................................................. 312

Appendices ..................................................................................................................................................................... 311
### C How tos: Certificate Trust and Trusted Identities

- **C.13 Adding Someone to Your Trusted Identity List** .......................................................... 333
  - C.13.1 Adding a Certificate From a Signature........................................................................ 333
  - C.13.2 Requesting a Digital ID via Email............................................................................. 334
  - C.13.3 Importing a Certificate From a File............................................................................ 334
  - C.13.4 Searching for Digital ID Certificates ....................................................................... 335
- **C.14 Certificate Trust Settings** .......................................................................................... 337
  - C.14.1 Setting Certificate Trust............................................................................................ 338
  - C.14.3 Using Certificates for Certificate Security (Encryption)......................................... 341
- **C.15 Using Directory Servers to Add Trusted Identities** .................................................. 341
  - C.15.1 Manually Configuring a Directory Server............................................................... 342
  - C.15.2 Editing Directory Servers Details............................................................................ 343
  - C.15.3 Deleting a Directory Server.................................................................................... 343
  - C.15.4 Specifying a Default Directory Server....................................................................... 344
  - C.15.5 Importing and Exporting Directory Server Settings................................................... 344
- **C.16 Managing Contacts** .................................................................................................. 344
  - C.16.1 Viewing and Editing Contact Details........................................................................ 344
  - C.16.2 Emailing Certificate or Contact Data........................................................................ 345
  - C.16.3 Saving Certificate or Contact Details to a File......................................................... 346
  - C.16.4 Associating a Certificate with a Contact................................................................. 346
  - C.16.5 Changing a Trusted Identity’s Certificate Association............................................. 346
  - C.16.6 Deleting Contacts and Certificates........................................................................ 347

### D How tos: Using Digital IDs

- **D.17 Working with Digital IDs** ......................................................................................... 349
  - D.17.1 Registering a Digital ID for Use in Acrobat.............................................................. 349
  - D.17.2 Digital ID Management and the Security Settings Console.................................... 350
  - D.17.3 Setting Identity Information..................................................................................... 350
- **D.18 Generic ID Operations** ............................................................................................ 351
  - D.18.1 Specifying Digital ID Usage..................................................................................... 351
  - D.18.2 Sharing (Exporting) a Digital ID Certificate............................................................. 352
  - D.18.3 Viewing All of Your Digital IDs................................................................................ 352
  - D.18.4 Customizing a Digital ID Name................................................................................ 353
  - D.18.5 Viewing Digital ID Certificates in the Certificate Viewer........................................... 353
- **D.19 Managing PKCS#12 Digital ID Files** ........................................................................ 354
  - D.19.1 Logging in to a Digital ID File.................................................................................... 355
  - D.19.2 Adding a Digital ID from a PKCS#12 File................................................................. 355
  - D.19.3 Adding and Removing Digital ID Files from the File List....................................... 356
  - D.19.4 Changing an ID File’s Password............................................................................... 356
  - D.19.5 Changing a PKCS#12 File’s Password Timeout...................................................... 357
  - D.19.6 Logging in to PKCS#12 Files..................................................................................... 358
  - D.19.7 Creating a Self-Signed Digital ID............................................................................. 358
  - D.19.8 Deleting a PKCS#12 Digital ID................................................................................ 361
- **D.20 Managing Windows Digital IDs** .............................................................................. 362
  - D.20.1 Finding a Digital ID in a Windows Certificate Store File......................................... 362
This guide describes the technical details of the Acrobat family of products’ security-related features such as digital signatures and rights management (encryption and permissions).

1.1 Why are You Here?

1.1.1 Because You Care About Information Assurance

Many users have business and other reasons to care about information assurance more than the average end user. You might work in an enterprise setting as an administrator or workflow architect. Your concern might be configuring, deploying, and managing clients across your organization, or you may be responsible for creating secure end-to-end workflows on a network. In all of those cases, you should be concerned with both the application’s runtime security options as well as its packaged security features.

It is obvious that digital signatures and content security through encryption and permissions are features designed to help you protect content and control its use. What’s sometimes missed however, is that these features are exercised in an application that runs on a machine interacting with other files, machines, and users via a network. Ideally, all of a workflow’s components should be secure, and tuning applications, machine, servers, and users (through education) increases the security of the environment in which signed and encrypted documents exist.

Tip: Runtime security is an essential component of information assurance. For more details, see the document *Application Security in the Acrobat Family of Products*.

Figure 1 Information assurance components

![Diagram of Information Assurance Components](image-url)
1.1.2 Because You’re Technical

While this document does contain some end user “how to use the feature” content, it is primarily a technical document which provides in depth details not found in end user help or in the SDK. The primary focus here is to provide details that help enterprise users, admins, and other business users set up and maintain secure PDF workflows. Potential audiences might include:

- Administrators who configure, deploy, and maintain clients on many machines in an enterprise environment.
- Developers who need registry level detail to augment SDK information about creating custom plug-ins and handlers that use Acrobat’s security features.
- End users that need advanced knowledge of Acrobat’s and Adobe Reader’s security features.

1.1.3 Because You’re Lost?

This guide provides technical details that are probably not of interest to the casual user. It is also not a developer document, and developers should refer to the SDK and its associated references and APIs. As you peruse this document, keep in mind that there are numerous resources out there, including forums and even video tutorials. Adobe is aggressively revamping many of its learning resources as Web 2.0 matures, and it may be that one of these sites would prove equally, if not more, useful:

- Admin and end user documents:
  - http://www.adobe.com/support/acrobat: A fabulous resource that is rapidly evolving into the primary location for tutorials, guides, videos, blogs, and other help.
  - http://www.adobe.com/support/livecycle
  - http://www.adobe.com/support/reader
- White papers/data sheets: http://www.adobe.com/security
- www.acrobatusers.com

**Figure 2 Resource roadmap**

---

**SECURITY TOPIC**

**RESOURCE ROADMAP**

<table>
<thead>
<tr>
<th>SECURITY TOPIC</th>
<th>RESOURCE ROADMAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security &amp; Information Assurance</td>
<td><a href="http://www.adobe.com/security">www.adobe.com/security</a></td>
</tr>
<tr>
<td>AcrobatUsers.com</td>
<td><a href="http://www.acrobatusers.com/topics/security">www.acrobatusers.com/topics/security</a></td>
</tr>
<tr>
<td>Acrobat Help &amp; Support</td>
<td><a href="http://www.adobe.com/support/">www.adobe.com/support/</a> +product</td>
</tr>
<tr>
<td>Developer Connection</td>
<td><a href="http://www.adobe.com/devnet/">www.adobe.com/devnet/</a> +product</td>
</tr>
</tbody>
</table>
Note: Table 1 shows a partial list of the documentation residing at the above locations.

<table>
<thead>
<tr>
<th>Document</th>
<th>Audience</th>
<th>For information about</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acrobat SDK Documentation Roadmap</td>
<td>Developers</td>
<td>A guide to the documentation in the Adobe Acrobat SDK.</td>
</tr>
<tr>
<td>Acrobat and PDF Library API Reference</td>
<td>Developers</td>
<td>A description of the APIs for Acrobat and Adobe Reader® plug-ins, as well as for PDF Library applications.</td>
</tr>
<tr>
<td>JavaScript for Acrobat API Reference</td>
<td>Developers</td>
<td>A listing of the Acrobat JavaScript APIs.</td>
</tr>
<tr>
<td>Developing Acrobat Applications with JavaScript</td>
<td>Developers</td>
<td>Additional detail about the Acrobat JavaScript APIs.</td>
</tr>
<tr>
<td>PDF Reference 1.x</td>
<td>Developers</td>
<td>A detailed description of the PDF language.</td>
</tr>
<tr>
<td>FDF Data Exchange Specification</td>
<td>Developers</td>
<td>A object-level FDF file description. The files can be generated programmatically and used to share security-related data.</td>
</tr>
<tr>
<td>PDF Signature Build Dictionary Specification</td>
<td>Developers</td>
<td>Build properties for the PDF Reference’s signature dictionary which provides interoperability details for 3rd party handlers.</td>
</tr>
<tr>
<td>Digital Signature Appearances</td>
<td>Developers &amp; administrators</td>
<td>Guidelines for creating signatures programmatically.</td>
</tr>
<tr>
<td>Guidelines for Developing CSPs for Acrobat on Windows</td>
<td>Developers &amp; administrators</td>
<td>Guidelines for developing a Cryptographic Service Provider for use with Acrobat® on the Windows® platform.</td>
</tr>
<tr>
<td>Enhanced Security in Adobe Acrobat 9 and Adobe Reader 9</td>
<td>Administrators &amp; end users</td>
<td>X-domain configuration specifically and other aspects of the enhanced security feature generally.</td>
</tr>
<tr>
<td>Digital Signatures in the PDF Language</td>
<td>Anyone needing an overview</td>
<td>A generic description of how signature work in PDF.</td>
</tr>
<tr>
<td>Digital Signatures in Acrobat</td>
<td>Anyone needing an overview</td>
<td>A description of how signatures are implemented in Acrobat.</td>
</tr>
</tbody>
</table>

1.2 What’s new for 9.1?

Tip: Looking for information about earlier versions? See Appendix B, “What’s Changed Across Releases”.

The following changes and enhancements appear in this release:

- Long term signature validation enhancements:
  - By default, certificate revocation information is embedded in the signature. This provides the ability to verify signatures using embedded revocation information even after the end entity, intermediate CA, and root certificates have expired.
  - Timestamp signatures include revocation information.
  - Ability to add certificate and revocation information post signing. This results in the following behaviors:
The Basics

What's new for 9.1?

- **Certified documents with no-changes-allowed**: Adding validation information after signing invalidates the certification in Acrobat 9.0 and earlier.

- **Signed documents**: Adding validation information after signing in Acrobat 9.0 will show unsigned changes if no subsequent signature is added. Documents with validation information added after signing in pre 9.0 versions will show as “Valid with modifications.”

  **Note**: Certified documents configured for no-changes-allowed that have validation information added post signing will show invalid certification in for earlier versions of Acrobat and Adobe Reader.

- A user-interface item **Show timestamp warnings in the Document Message Bar** has been added. *This feature is currently not implemented.*

- The default signature digest algorithm is changed to SHA-256. If the cryptograph service provider cannot accommodate SHA-256, then the application uses SHA-1. For example, XP pre service pack 3 does not support SH-256 if the Windows CSP is used.

- In Acrobat 9.0, signing a certified document with the Lock After Signing option selected caused the certification to be invalidated. This problem is corrected in Acrobat 9.1.

- Signature verification time: The validation time new default is **The time at which the signature was created**. This value does not update after upgrading to 9.1 if the user has modified the setting for 9.0.

- You can apply an mouse or pen-driven signature by choosing **Advanced > Sign and Certify > Apply Ink Signature**.

  **Tip:** This command is unrelated to digital signatures and may be disabled for some documents. It may, however, cause confusion for some users. Organizations may wish to educate users on when to sign with one or the other.

- Script changes prevent signing: It is no longer possible to sign (or certify) a document in the same session when a document script change was made. You must first save, close, and reopen the document. This change affects form authors and developers.

  It should also be noted that scripts that modify scripts on the form can no longer be used in conjunction with signatures. This restriction is due to security vulnerability problems. Certified forms could never modify scripts and are unaffected. Uncertified forms that modified scripts during a fill-in and signing process must be changed so that the effects of the script updates can be achieved in other ways.

- Signing and certifying with Adobe-provided Reader Extensions certificates no longer permitted: Prior to Acrobat 9.1, it was possible to use a Reader Extensions certificate provided by Adobe to apply usage rights to a document for signing or certifying. Acrobat and Adobe Reader 9.1 no longer support using those certificates for signing or certifying. If a document was signed or certified using such a certificate in a previous version of Acrobat or in another product such as LiveCycle ES or Interactive Forms based on Adobe software, the certification will be shown as invalid in Acrobat and Adobe Reader 9.1 and later, although it will still show as valid in earlier versions of Acrobat and Adobe Reader.

- Macintosh OS X keychain support for software and hardware credential (private key) usage. The Macintosh Keychain Store is equivalent of the Microsoft Windows Certificate Store, and supporting it allows the user to use their Keychain Store credentials to:
  
  - Sign a document.
  
  - Encrypt a document using a certificate associated with a Keychain credential.
  
  - Decrypt documents.
PKCS#12 and ArcotID credential provisioning for encrypted document delivery.

- Signature validation performance improvements.
- XML data signature improvements.
- Additional localized search and redact patterns.

**Known issues**

Invalid Signature state in Reader 9.1: Forms containing scripts that assign an illegal value, such as a string, to a numeric field cause invalid signature status in some cases.

- If the document was signed by an earlier version of Acrobat and verified in Reader 9.1, Reader 9.1 shows the signature as invalid.
- If the document is signed by Reader 9.1, future versions of Reader show the signature as invalid. The workaround is to remove the offending script. Future versions of Reader will throw an exception so that such script is easier to find.

### 1.3 Basic Concepts

You’re going to have a hard time understanding most of the content in this document without understanding how Acrobat defines or uses “trust,” “trusted identities,” and “digital IDs.” Trust me.

#### 1.3.1 What is Security and Information Assurance?

As shown in Figure 1, security in the context of a living workflow includes all of the workflow’s components as well as the proper exercising of Acrobat’s security features by those participating in that workflow. Information assurance includes all the products, services, features, policies, and procedures that allow the reliable exchange of electronic information.

Adobe helps organizations protect sensitive information by enabling confidentiality, privacy, authentication, integrity, non repudiation, and availability.

#### 1.3.2 What is Trust?

The concept of “trust” may mean different things in different contexts. In Acrobat security workflows, trust can mean the following:

- **Trusting participants in your workflows:** For content security and signature workflows, you will need to trust those with whom you are sharing your documents. “Trusting an identity” means that you accept that someone’s certificate actually represents a particular person or organization. It is official recognition on your part of the ownership and origin of the digital ID; that is, that the digital ID represents a specific entity.

- **Setting certificate trust levels:** Once you’ve created a trusted identities list, you will likely need to allow and disallow certain operations. You do this by associating (setting) trust levels with each trusted identity’s certificate. Trust levels define privileges that allow documents signed or certified by that identity to execute privileged operations on YOUR machine--things that cannot otherwise be done by documents you otherwise just open and display--for example, playing multimedia or...
executing JavaScript. Providing trust to a certificate should only be done if you want documents created or signed by the trusted identity to have higher levels of access to your machine.

1.3.3 What is a Trusted Identity?

Digital signature and certificate security workflows both rely on certificates. Participants in signing workflows share their certificates ahead of time or embed them in a document. Participants in certificate security workflows must share their certificates ahead of time. Both operations involve importing other people's certificates into your Trusted Identities list. When a person's certificate information appears in the Trusted Identity Manager, they become a trusted identity.

Groups of people that share documents with certificate security or digital signatures are in essence a community of trusted identities that share their certificates to make those features work. You will add people to your trusted identity list and others will add you to theirs:

- When you sign document, the document recipient can validate your signature by validating the certificate embedded in the document. Conversely, you need access to a document sender's certificate to validate their signature.
- You encrypt a document with the document recipient's public key so that they can decrypt it with their corresponding private key. Conversely, others need your certificate to encrypt documents for you.

The Acrobat family of products provide tools for selecting and interacting with the certificates of document recipients you trust. For example, Acrobat’s user interface prompts authors to select one or more recipients when applying certificate security. Because it is often the case that a document will be sent or received from numerous individuals, it is expedient to create a list of trusted identities ahead of time. In large organizations, an administrator may do this for you; otherwise, you will use Acrobat’s Trusted Identity Manager to store your trusted identities’ contact information and certificates.

Getting someone’s contact information and certificate involves searching for (or having sent to you) the digital ID data in the requisite format. Some common ways of getting the data include the following:

- **Import the data from an .acrobatsecurity file.** Configuration details can be imported from a security settings file as described in Chapter 3, "Migrating and Sharing Security Settings".
The Basics

What is a Digital ID?

- **Extract the data from an FDF file.** Double-clicking on an FDF file causes Acrobat to automatically import the information.
- **Search a server directory.** Users can add directory servers containing contact information and certificates. Sometimes administrators preconfigure these directories.
- **Use the data embedded in a signed document.** The Certificate Viewer’s **Add to Trusted Identities** button adds a certificate to the trusted identities list and allows setting its trust level.

![Figure 3  Digital ID: Managing trusted identities]

From within the Manage Trusted Identities dialog, users import and manage the certificates and certificate owner data for document recipients they wish to trust. A contact will occasionally be associated with multiple certificates. Therefore, contacts and certificates are in some respects managed independently of each other. It is also possible to create a group from any number of contacts so that security can be applied to all group members with a single action. Users manage contacts, groups, and certificates by choosing **Advanced (Acrobat)** or **Document (Reader) > Manage Trusted Identities** and opening the Trusted Identities Manager.

![Figure 4  Manage Trusted Identities menu item]

1.3.4 What is a Digital ID?

A digital ID is like a driver’s license or passport or other “certified by some entity” paper identification. It proves your identity to people and institutions that you communicate with electronically. These IDs are an essential component of digital signatures and certificate security. In signing and certificate security workflows, you will be asked to select a digital ID from a list of previously installed digital IDs, since they are a required for signing, certifying, and applying certificate encryption to PDFs.
You can get a digital ID from a third-party provider, or you can create a self-signed digital ID. Self-signed digital IDs may be adequate for many situations. However, to prove your identity in most business transactions, you may need a digital ID from a trusted third-party provider, called a certificate authority. Because the certificate authority is responsible for verifying your identity to others, choose one that is trusted by major companies doing business on the Internet.

A digital ID consists of two main parts: a certificate and a private key. A certificate consists of your identity information (name, date, serial number, etc.) and a public key that are bound together and signed by a trusted or untrusted certificate authority. The certificate sometimes includes a reference to the certificate issuer’s certificate, thereby creating what is known as a “certificate chain.”

Digital IDs operate by using a key pair: data encrypted with one key can only be decrypted by the other corresponding key. When you sign PDF documents, you use the private key to apply your digital signature. You distribute the certificate that contains your public key to those who need to validate your signature or encrypt information for you. Only your private key can unlock information that was encrypted using your public key, so be sure to store your digital ID in a safe place.

Some users have multiple digital IDs for different purposes; for example, to sign documents in different roles or using different certification methods. Digital IDs are usually password protected and can be stored on your computer in password protected file, on a smart card or hardware token, in the Windows certificate store, or on a signing server (for roaming IDs). Acrobat applications can access digital IDs from any of these locations.

Users exchange their digital ID’s certificate so that they can validate signatures and encrypt documents for each other. Shared certificates can be physically sent in a file or made available over a network. The private key is never shared and is used to decrypt documents. There are several ways to share certificates:

- **Physical sharing**: Certificates can be physically shared in a file sent via email or located in a shared directory. They can be imported, exported, and otherwise managed with the Trusted Identity Manager. For details, see Chapter C, “How tos: Certificate Trust and Trusted Identities”.

- **Network sharing**: Certificates can be stored on a central server. The Trusted Identity Manager can be used to search for certificates on LDAP directory servers. Adobe applications provide tools for configuring and managing directory servers. For details, see “Using Directory Servers to Add Trusted Identities” on page 341.
1.3.5 Digital ID Storage Mechanisms

A digital ID’s certificate and private key need to be stored in a secure location. There are several file types and file locations where these items could be stored (Table 2). The digital ID data in these files is provided to the application via digital ID service providers (sometimes called Cryptographic Service Providers or CSPs). A service provider is simply a storage mechanism and code that makes the data available to the application.

In most cases, the digital ID is stored on a local or networked file. Common locations include the Windows Certificate Store which is accessible by Adobe applications and other Windows applications and the Acrobat store which is used only by the Acrobat family of products. Some IDs may exist only on external hardware such as a smart card connected to the computer.

The Acrobat family of products can access a digital ID from the following storage mechanisms:

- **Windows Certificate Store:** A local store (file location) provided by Windows that can import and export various file formats and that can be used by both Windows programs and Acrobat products.
- **PKCS#12 files:** A common file format residing on your hard drive that is used on both Windows and Macintosh.

  **Tip:** PKCS refers to a group of Public Key Cryptography Standards authored by RSA Security

- **PKCS#11 devices:** External devices such as a USB token or smart card that store digital ID data.
- **Roaming ID servers:** A network server. The private key is known only to the server. The server sends the certificate and its public key to users on demand. Users can import and export the certificate and its public key from Acrobat, but they never have install the private key on a local machine.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>5.x</th>
<th>6.x</th>
<th>7.x</th>
<th>8.x</th>
<th>9.x</th>
</tr>
</thead>
<tbody>
<tr>
<td>.acrobat</td>
<td>An XML format encapsulated in a PDF which stores security settings for import and export.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>security</td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PKCS#12:</td>
<td><strong>Personal Information Exchange Syntax Standard:</strong> Specifies a portable, password protected, and encrypted format for storing or transporting certificates.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.pfx (Win),</td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.p12 (Mac)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Basics

Digital ID Storage Mechanisms

Table 2 Digital ID-related file types

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>5.x</th>
<th>6.x</th>
<th>7.x</th>
<th>8.x</th>
<th>9.x</th>
</tr>
</thead>
<tbody>
<tr>
<td>.fdf</td>
<td>An Adobe file data exchange format used for importing and exporting settings and certificates (usually PKCS#12 files).</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td>.p7b, .p7c</td>
<td><strong>Certificate Message Syntax (CMS):</strong> Files with .p7b and .p7c extensions are registered by the Windows OS. Acrobat products can import and export these files.</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Certificate and public key only</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td>.cer</td>
<td><strong>Certificate format:</strong> A Microsoft format for digital IDs usually stored in the Windows Certificate Store.</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Certificate and public key only</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td>.apf</td>
<td><strong>Adobe Profile Files (Legacy):</strong> Not used after Acrobat 5. Files can be upgraded by double clicking them.</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
</tbody>
</table>
Administration Guide

Tuning the Installer
Migrating and Sharing Settings
Registry/Plist Level Configuration
Predeployment Installer Tuning

Because of Acrobat’s rich security features, users as well as administrators may require a way to copy settings across a number of machines or simply upgrade between application versions without losing their application’s configuration. There are two main mechanisms for doing just that:

- **Security setting import/export**: Whether a user wants to upgrade or an administrator wants to propagate settings across already installed applications, Acrobat import/export feature allows a user to select what to export to a secure file. For details, see “Security Setting Import and Export” on page 35.

- **Customizing the installer**: In enterprise settings, it is common to customize the installer for the Acrobat family of products prior to deploying clients on multiple machines. For details, see “Getting Started with the Wizard” on page 24.

- The following examples suggest using the Adobe’s Customization Wizard 8 and Acrobat’s export feature, but other deployment mechanisms are possible.

2.1 Getting Started with the Wizard

The Adobe Customization Wizard 9 for Acrobat provides a graphical user interface that allows you to customize the product’s Windows installer. It provides administrators with the ability to tune the installer when business needs can benefit from customized document security and digital signature workflows. The wizard saves deployment time and reduces complexity by eliminating scripts and manual registry edits. Preconfigured applications enable standardizing the user experience by providing consistent security settings across the enterprise. Tuned installer changes run whether the installation is pulled from a server or pushed silently to users’ desktops.

You can use wizard to customize Adobe product installers in the following ways:

- Customize digital signatures settings for signing and validation.
- Customize document security settings.
- Edit the Adobe application’s registry and installer tables.
- Configure a connection to an Adobe LifeCycle Rights Management Server.
- Lock settings so they cannot be altered by users.

While the wizard is interoperable across the entire Acrobat family of products, not all security features are available in all products. This document assumes the administrator is tuning an Acrobat Professional 8 installer which includes the full range of security features.

**Note:** The wizard is typically capable of tuning any member of the Acrobat family. Other Acrobat products include Adobe Reader, Adobe Acrobat Standard, and Adobe Acrobat Pro, etc. Product names may change across releases.

After reading “Wizard Setup” on page 28, refer to one of the following:

- “Basic Installer Tuning” on page 29
- “Advanced Installer Tuning for Power Users” on page 32

24
Tip: The wizard is capable of many other operations relevant to non-security application features. For details, refer to the wizard's documentation. While this document only describes the application’s security features, you should configure all of the application’s options at the same time. For more information about the wizard, refer to http://www.adobe.com/products/acrobat/deployment.html.

A note about migration and upgrading

Registry settings are not automatically migrated. You should always export and save those registry settings you want to appear in the new installation. You can import those settings and use them in a tuned installer.

2.2 Application Defaults

Before deploying an application across the enterprise, you should familiarize yourself with the application defaults as well as what is configurable via the user interface and the application preferences (the registry in the case of Windows). Knowledge about what is customizable is a prerequisite to tuning the out-of-the-box behavior prior to deployment. For details, see the following:

- “Signing Environment Defaults” on page 25
- “Signature Validation Defaults” on page 26
- “Document Security Defaults” on page 27

2.2.1 Signing Environment Defaults

Administrators can configure preferences to control the signing workflow, when and how a signature is created, the conditions for successful signature creation, and so on. For example, these preferences tell the application where to look for Windows certificates, control signature appearances, and provide control over message digest algorithms. The default settings are listed in Table 1.

Table 1 Default signing settings

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI?</th>
<th>Default</th>
<th>Registry Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Signing Method</td>
<td>No</td>
<td>Adobe Default Security</td>
<td>Adding additional methods, changing the default, and locking down the setting.</td>
</tr>
<tr>
<td>OCSP revocation checking</td>
<td></td>
<td></td>
<td>See signature validation defaults.</td>
</tr>
<tr>
<td>CRL revocation checking</td>
<td></td>
<td></td>
<td>See signature validation defaults.</td>
</tr>
<tr>
<td>Chain building</td>
<td>No</td>
<td>Don't follow URIs or require valid certificates</td>
<td>Allowing following URLs, care about certificate validity for RSA signatures, specifying LDAP servers, require certain policy OIDs.</td>
</tr>
<tr>
<td>Chain building with policies</td>
<td>No</td>
<td>See RFC 3280</td>
<td>Controlling certificate path processing logic via certificate policy extensions.</td>
</tr>
<tr>
<td>Hashing algorithm</td>
<td>No</td>
<td>SHA1</td>
<td>Specifying a different algorithm.</td>
</tr>
<tr>
<td>Signing format</td>
<td>No</td>
<td>adbe.pkcs7.detached</td>
<td>Specifying a different format.</td>
</tr>
</tbody>
</table>
2.2.2 Signature Validation Defaults

Administrators can configure preferences to control the signature validation workflow, when and how a signature is validated, the conditions for a signature status of Valid, and so on. For example, the preferences allow you to customize revocation checking, control document warnings, and specify when and where status icons appear. The default settings are listed in Table 2.

Table 2 Default signature validation settings

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI?</th>
<th>Default</th>
<th>Registry Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message digest comparison</td>
<td>No</td>
<td>Off</td>
<td>For RSA signatures, requiring a comparison between the pre and post signed message digests and aborting the signing process if they are not identical.</td>
</tr>
<tr>
<td>Enable password caching</td>
<td>No</td>
<td>On</td>
<td>Controlling whether certain passwords are cached to disk.</td>
</tr>
<tr>
<td>Signature logo/watermark</td>
<td>No</td>
<td>Acrobat logo</td>
<td>None. Using a custom logo is possible but does not occur via the registry.</td>
</tr>
<tr>
<td>View documents in preview document mode when signing</td>
<td>Yes</td>
<td>Off</td>
<td>Requiring preview mode.</td>
</tr>
<tr>
<td>Include signature’s revocation status when signing</td>
<td>Yes</td>
<td>Off</td>
<td>Embedding revocation status in signature to aid long term validation.</td>
</tr>
<tr>
<td>Show reasons when signing</td>
<td>Yes</td>
<td>Off</td>
<td>Turning the field on and off, customizing the reason’s list, and locking down the settings.</td>
</tr>
<tr>
<td>Show location and contact information when signing</td>
<td>Yes</td>
<td>Off</td>
<td>Turning the field on and off and providing default location and contact data.</td>
</tr>
<tr>
<td>Enable Reviewing of Document Warnings</td>
<td>Yes</td>
<td>When certifying</td>
<td>Setting any of the valid values.</td>
</tr>
<tr>
<td>Prevent Signing Until Document Warnings Are Reviewed</td>
<td>Yes</td>
<td>Never</td>
<td>Setting any of the valid values.</td>
</tr>
<tr>
<td>Timestamp servers</td>
<td>Yes</td>
<td>None</td>
<td>Configuring one or more servers and setting one as the default so that it applies a timestamp to each digital signature.</td>
</tr>
<tr>
<td>Roaming ID servers</td>
<td>Yes</td>
<td>None</td>
<td>Configuring roaming ID servers to enable user access to their digital ID from any location, setting an authentication method, etc.</td>
</tr>
<tr>
<td>Document warnings</td>
<td>Yes</td>
<td>Show when certifying, don’t require review</td>
<td>Setting when to show warnings, specifying whether or not warning review is required by the signer, and turning off and on font warnings.</td>
</tr>
<tr>
<td>Windows integration</td>
<td>Yes</td>
<td>Windows store is not accessible</td>
<td>Controlling access to MSCAPI via Acrobat so that users can find and set trust levels for certificates in the Windows store, and locking down the settings.</td>
</tr>
<tr>
<td>Signature Appearance</td>
<td>Yes</td>
<td>Text only</td>
<td>Specifying a custom appearance.</td>
</tr>
</tbody>
</table>

Table 1 Default signing settings

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI?</th>
<th>Default</th>
<th>Registry Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCSP revocation checking</td>
<td>No</td>
<td>If possible, check, fail if bad</td>
<td>Controlling if checks are done, how failure affects the signer’s ability to sign, whether online checks should be done, required certificates, and data and time criteria, request parameters, and so on.</td>
</tr>
</tbody>
</table>
2.2.3 Document Security Defaults

Administrators can configure preferences to control document security workflows, available security methods and handlers, security policies, and so on. For example, the preferences allow you to customize create a list of preconfigured security policy favorites, set up policy servers, and force revocation checking on certificates used for encryption. The default settings are listed in Table 3.

Table 3 Default document security settings

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI?</th>
<th>Default</th>
<th>Registry Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revocation checking</td>
<td>No</td>
<td>Don't check</td>
<td>Controlling if checks are done and whether encryption has a dependency on check success.</td>
</tr>
<tr>
<td>Enable password caching</td>
<td>No</td>
<td>On</td>
<td>Controlling whether certain passwords are cached to disk.</td>
</tr>
<tr>
<td>Nested verification sessions</td>
<td>No</td>
<td>5</td>
<td>Limiting the number of nested verification sessions</td>
</tr>
<tr>
<td>Clock skew</td>
<td>No</td>
<td>65 seconds</td>
<td>Limiting the amount of time signing can be after validation.</td>
</tr>
<tr>
<td>Validate on document open</td>
<td>Yes</td>
<td>On</td>
<td>Turning the field on and off and locking down the setting.</td>
</tr>
<tr>
<td>Verification time</td>
<td>Yes</td>
<td>Signing time if used</td>
<td>Adding additional methods, changing the defaults, and locking down the settings.</td>
</tr>
<tr>
<td>Hiding status icon for valid signatures</td>
<td>Yes</td>
<td>Show</td>
<td>Hiding the signature status icon for valid signatures.</td>
</tr>
<tr>
<td>Using revocation data embedded in a signature</td>
<td>Yes</td>
<td>On</td>
<td>Specifying when to use embedded revocation information as well as whether to make it available via JavaScript.</td>
</tr>
<tr>
<td>Default Signature Verification Method</td>
<td>Yes</td>
<td>Adobe Default Security</td>
<td>Adding additional methods, changing the default, and locking down the setting.</td>
</tr>
<tr>
<td>Windows integration</td>
<td>Yes</td>
<td>Windows store is not accessible</td>
<td>Controlling access to MSCAPI via Acrobat so that users can find and set trust levels for certificates in the Windows store, and locking down the settings.</td>
</tr>
</tbody>
</table>
### Table 3 Default document security settings

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI?</th>
<th>Default</th>
<th>Registry Customization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe LifeCycle Rights Management Servers</td>
<td>Yes</td>
<td>Allow configuration</td>
<td>Setting up one or more servers, choosing a default, specifying the protocol, and locking the settings.</td>
</tr>
<tr>
<td>Security policies</td>
<td>Yes</td>
<td>1 certificate and 1 password policy</td>
<td>Preconfiguring a list of policy names, IDs, preferred handlers, and marking policies as favorites.</td>
</tr>
<tr>
<td>Default Security Method</td>
<td>Yes</td>
<td>Adobe Default Security</td>
<td>Adding methods, changing the default, and locking down the setting.</td>
</tr>
<tr>
<td>Windows integration</td>
<td>Yes</td>
<td>Windows store is not accessible</td>
<td>Controlling access to MSCAPI via Acrobat so that users can use certificates in the Windows store for encryption.</td>
</tr>
</tbody>
</table>

### 2.3 Wizard Setup

To prepare for customizing the Adobe application’s installer:

1. Download the Adobe Customization Wizard 9, and install it according to its instructions.

2. Copy all of the installer files of the application you are going to tune on the same machine. You will need access to the .msi file during the tuning process.

3. Collect any needed files. For example, a company logo for signature backgrounds, .acrodata files listing LDAP servers and trusted identities, exported registry keys, and so on. In most cases, you can use many of the files from an already installed administrator-configured application.
2.4 Basic Installer Tuning

The wizard provides a user interface for modifying the application’s .msi installer file. You do not need to preinstall and configure the Acrobat application. However, there are several limitations to this method:

- **Files and Folders**: Many of the template files won’t exist unless you have installed the application and configured it. For example, the easiest way to create a directory server file is with the application’s GUI.

- **Registry**: The registry directories do not exist until the code is exercised from an installed application. While you could build the paths and add the values manually, simply configuring an installed application and using it as a template does most of the work for you.

- **Other options**: The wizard does not provide an UI for every configurable option, yet a large part of the application’s behavior is modifiable. Installing, configuring, and using the application is the
quickest way to understand how and why the installer should be modified to suit your enterprise environment.

**Tip:** To take full advantage of the wizard’s capabilities, advanced users should skip this section and proceed to “Advanced Installer Tuning for Power Users” on page 32.

To perform basic installer tuning for security features:

1. Click on CustWiz.exe.
2. Choose **File > Open Package**.
3. Browse to <application name>.msi file you would like to customize.
4. Choose **Open**.
5. Click on **Files and Folders** in the left-hand panel.

**Files and Folders** opens a directory of files on the source computer that enables copying existing files to the destination system. You can add or remove files from the installation list and modify file records. Copy preconfigured files from an existing installation from the top window to the appropriate folder in the bottom window under **Destination Computer**.

1. Copy files with the .acrodata extension. Paths vary by application, version, and user. For example, on Windows XP, drag and drop all acrodata files from C:\Documents and Settings\<user name>\Application Data\Adobe\Acrobat\<application version>\Security to CommonAppDataFolder\Adobe\<application>\<version>\Replicate\Security. In addition to directories.acrodata and addressbook.acrodata which can be set through the user interface (described below), files could include:
   - **addressbook.acrodata**: Populates the Trusted Identity Manager with certificate data. This file can be used to distribute a trust anchor for signature validation. Set now or at **Step 7**.
   - **directories.acrodata**: Populates the directory server list in the Security Settings Console. This file creates connections to an LDAP server used as a certificate repository. Set now or at **Step 7**.
   - **Appearances.acrodata**: Stores signature appearances. Unless there is a shared company appearance, do not use this file.
   - **security-policy.acrodata**: Populates the security policy list in the Security Policy Manager. This file can be used to distribute common document security policies.

2. Copy other needed files to some custom or existing location. For example, expand the Destination Computer directory ProgramFileFolder\Adobe\Acrobat \x.x\Acrobat and create a new folder called Security. You could copy SignatureLogo.pdf here (a signature watermark file) or other files such as drivers for PKCS#11 devices. The tuned installer will copy these files to the matching location in the installed application.

**Note:** Only .acrodata files can be placed in CommonAppDataFolder.
6. Click on **Registry** in the left-hand panel.

Registry opens a directory of registry keys on the source computer that can be dragged to the destination system. You can add, modify, or remove registry keys and registry key strings. If you need to modify the registry, you should probably not be using these basic instructions and should instead refer to “Advanced Installer Tuning for Power Users” on page 32.

7. Click on **Security** in the left-hand panel.

Security sets the directory server, trusted identity, and examine document options. Skip these steps if you have already copied the *directories.acrodata* and *addressbook.acrodata* files using **Files and Folders** described above. If you do not have these files, refer to “Advanced Installer Tuning for Power Users” on page 32.

1. Choose **Set** next to the **Directory Servers**. The wizard automatically looks in the default location: C:\Documents and Settings\<user name>\Application Data\Adobe\<application name>\<application version>\Security.

   **Note:** These files must be NEWER than the same files on the destination computer or they will not be copied to the new machine.

2. **Browse** to *directories.acrodata*. 
3. Choose **Open**.

4. Repeat the above steps for the Trusted Identities field and add *addressbook.acrodata*.

   **Note:** For application security details such as enhanced security configuration, refer to *Application Security for the Acrobat Family of Products*.

8. Click on **Digital Signatures** in the left-hand panel.

Digital Signature specifies the security options (verification, creation, reasons) for digital signatures. You can prevent end-users from changing many of these settings through the user interface by choosing the **Disallow** and **Prevent** check boxes.

   **Note:** At least one reason for signing must be used when using the wizard.

**Figure 3  Wizard: Digital signature panel**

9. This completes basic security feature configuration. Configure other preferences as desired, save the project, and exit the wizard.
2.5 Advanced Installer Tuning for Power Users

The wizard’s user interface (GUI) only provides basic options for security configuration. Moreover, using the wizard’s GUI to set security preferences will result in overwriting existing registry settings that you might drag and drop from your template application. For these reasons, power users—those with extensive experience modifying the registry and who are familiar with Acrobat’s advanced features—may want to utilize a more manual process. A robust and feature rich deployment workflow involves the following:

1. Install the new application. Upgrading an existing installation will result in the migration of some application settings and files.

2. Configure the application via its user interface which sets many registry preferences.

3. Modify the registry as needed. There are dozens of registry preferences that you can use to control workflows, certificate processing, data handling, communications with others servers, and so on. Many enterprise deployments have specific business requirements that would benefit from customizing how Acrobat behaves in document security and digital signature workflows. Administrators should familiarize themselves with the available options and customize client installations accordingly. Review the following sections to learn how and why to modify settings:
   - “Setting Basic Client and Workflow Preferences” on page 75
   - “Content Security” on page 101
   - “Digital Signatures” on page 104

4. Lock down features so that settings can not be changed by end users. For details, see “Preventing End-User Modification” on page 75.

5. Use the wizard to configure settings other than those under Security and Digital Signatures (beyond the scope of this document).

   **Tip:** If you need to manually edit the registry, you may not want to use the wizard’s user interface to set the Security and Digital Signature preferences.

6. Use the wizard to drag and drop the configured template registry to the installer.

7. Set up file deployment:
   1. Click on Files and Folders to open a directory of files on the source computer that enables copying existing files to the destination system. You can add or remove files from the installation list and modify file records. Copy preconfigured files from an existing installation from the top window to the appropriate folder in the bottom window under Destination Computer.

   **Note:** Only .acrodata files can be placed in CommonAppDataFolder. These files must be NEWER than the same files on the destination computer or they will not be copied.

   2. Copy acrodata files. Paths vary by application, version, and user. For example, on Windows XP, drag and drop all acrodata files from \Documents and Settings\<user name>\Application Data\Adobe\Acrobat\<application version>\Security to CommonAppDataFolder\Adobe\<application>\<version>\Replicate\Security. Acrodata files could include:
**addressbook.acrodata**: Populates the Trusted Identity Manager with certificate data. This file can be used to distribute a trust anchor for signature validation.

**directories.acrodata**: Populates the directory server list in the Security Settings Console. This file can be used to configure connections to an LDAP server used as a certificate repository.

**Appearances.acrodata**: Stores signature appearances. Unless there is a shared company appearance, do not use this file.

**security-policy.acrodata**: Populates the security policy list in the Security Policy Manager. This file can be used to distribute common document security policies.

3. Copy other needed files to a custom or existing directory. For example, expand the Destination Computer directory ```ProgramFileFolder\Adobe\Acrobat <application version>\Acrobat\``` and create a new folder called Security. Some common files that may be distributed this way include:

- **SignatureLogo.pdf**: To create a watermark or logo on each employee signature, create the logo and manually copy the file to the Documents and Settings Directory under their username.

- **PKCS#11 drivers**: If you use smart cards, you can distribute drivers that end users can import via the Security Settings Console.

- **Custom security handlers**: If you have developed a custom handler for signing or document security, install it in the application’s plugin directory.

**Note**: The tuned installer will copy these files to the matching location.

You are now ready to deploy the application as described in the Wizard’s documentation. Before deployment, verify that you have configured the following:

- Security infrastructure (service providers and preferences for digital ID and certificate management)
- Signing environment
- Signing workflow
- Signature creation
- Signature validation
- Document security
- Feature lock down settings
- Registry configuration: The new registry is copied to Wizard’s destination computer
- File copying: Needed files are copied to Wizard’s destination computer
Migrating and Sharing Security Settings

Security settings can be complex, and more importantly, they are often critical components of digital signature and document security workflows. For this reason, it’s often necessary to migrate and even share settings across one or more machines. There are two methods available:

- **Security Setting Import and Export**: One of Acrobat 9.0’s major new security features includes the ability to import and export security settings via .acrobatsecuritysettings files, thereby enabling easier version upgrades as well as configuration of multiple machines. The security settings import/export features offer several advantages over FDF files:
  - Most document security and digital signature related settings can be encapsulated in an `acrobatsecuritysettings` file whereas FDF could only transport one setting type and a time and could not encapsulate registry settings at all.
  - One file can be used instead of many files.
  - Trust can be assigned to imported on the fly, thereby simplifying workflows.
  - Security. Files can be signed and encrypted.
  - Files can be used to backup and restore settings, to distribute settings in a workgroup or enterprise, and to send specific information to another user.

- **Sharing Settings & Certificates with FDF**: FDF files are useful for importing and exporting a specific type of setting such as trust anchors, timestamps, directory servers, and so on.

### 3.1 Security Setting Import and Export

Acrobat 9.0 introduces a new feature that helps users and organizations migrate existing security settings through version upgrades and across multiple machines. Unlike FDF files, the new .acrobatsecuritysettings file supports the import and export of all settings including digital ID data, trust, server details, signing preferences, and so on. Settings can only be exported from Acrobat but settings can be imported by both Acrobat and Adobe Reader.

Acrobat now provides an interface that allows users to export their settings to a secure file for sharing or later import. Any workflow consists of two phases: creation of the security settings (export) and installation of the security settings (import).

- **Export**: To create the security settings file for export, simply choose **Advanced > Security > Export Security Settings**.

- **Import**: To install previously saved security settings, choose **Advanced > Security > Import Security Settings**, or open the settings file in Acrobat or Reader.

All security settings are represented as XML embedded in an empty PDF document. It contains elements that describe security settings for each of the supported security components, including encryption, digital signatures, usage rights, trusted identities, security servers, and so on. The format provides capabilities for different levels of the security settings management.

The PDF has an .acrobatsecuritysettings extension. On export, the file is certified with an invisible signature and optionally encrypted so that only authorized users can access it and verify its
trustworthiness before installation. At the installation time, the certification signature status is displayed at the top of the Import Security Settings dialog box.

Note: When updating from version 8 to version 9, the installer will move most settings from 8 into 9 automatically. This can save considerable time during an upgrade. Some settings such as PKCS#11 modules require installation and cannot automatically be moved.

3.1.1 Exporting Security Settings to a File

Settings can only be exported from Acrobat.

1. Choose **Advanced > Security > Export Security Settings**.

2. Check or uncheck the settings you would like to export.

Note: Whether you export or import settings via an FDF file or an .acrobatsecurity setting file, the actual settings are the same. Details about each individual setting are found in the FDF section as well as elsewhere in this document.

3. Choose **OK**.

Figure 4 Security settings: Export dialog

4. When the detailed Export Security Settings dialog appears, review the settings again.

5. If you would like to include or exclude any settings, highlight the setting and choose the **Include/Exclude Setting** button.
6. Choose **Export**.

7. Choose an encryption method. Encrypting the file ensures that the settings can’t be viewed by anyone other than the intended recipients.

**Figure 5  Security settings: Encryption method**

```
Encrypt Security Settings

Encrypt Security Settings using:
- Password Security
- Certificate Security
- None

OK  Cancel
```

8. Follow the dialog instructions which will vary with your choice of the document security method (password security or certificate security).

9. Choose **OK**.

10. You will be required to certify the file by signing it with a certification signature. When the certification workflow begins, choose **OK**.

11. Sign and save the file.

### 3.1.2 Importing Security Settings from a File

Settings can be imported by both Acrobat and Adobe Reader.

To import security settings:

1. Choose **Advanced > Security > Import Security Settings**.


3. Choose **Open**.

4. acrobatsecuritysettings files must be certified and are therefore signed. You can verify the signer’s identity by choosing the **Signature Properties** in the Document Message Bar and reviewing the signer’s details.

**Figure 6  Security settings: Document message bar**

```
To import the security settings shown in this document, click Import.
Certified by three <three@anotherone.com>, three certificate issued by three.
```

5. Review the settings carefully.
**Note:** Whether you export or import settings via an FDF file or an .acrobatsecurity setting file, the actual settings are the same. Details about each individual setting are found in the FDF section as well as elsewhere in this document.

**Caution:** The settings in the imported file will overwrite your current settings. Be sure to verify you’re getting the correct settings and that they are coming from a trusted source.

**Figure 7 Security settings: Import from a file panel**

6. Choose **Import**.

7. If the settings you imported included Digital IDs, you must log into each such ID to complete its installation. If there were Digital IDs then a dialog appears asking if you’d like to open the Security Settings Console and log in to the digital IDs you just imported. Choose **Yes** or **No**.

**Note:** For security reasons, acrobatsecuritysettings files do not carry the digital ID passwords. Before you can use any of the digital IDs you just imported, you must log in to each ID. You can do it now or later.

**Figure 8 Security setting import: Success dialog**
3.1.3 Importing Security Settings from a Server

If your organization distributes security settings periodically, you can set up Acrobat to regularly check for updates to these policies. Server-based security is set up by an administrator who provides the URL from which to get security updates. Once the application is configured, Acrobat will periodically poll the server (the default time is every three months) via http or https.

1. Choose **Edit > Preferences > Security**.
2. Check **Load security settings from a server**.
3. Enter the server address in the URL field.
4. Select a signing certificate if any. The .acrobatsecurity file will be signed with a certified signature. In order to install the file, you will need validate the signature.
5. Specify how often you want to check for security updates.
6. Select **Ask Before Installing** to be notified prior to installing new settings.
7. When the acrobatsecurity file opens, follow the instructions as described in “Importing Security Settings from a File” on page 37.

**Figure 9  Security setting preferences for server import**

3.2 Sharing Settings & Certificates with FDF

**Note:** The first time you receive settings, you may not have the certificate of the signer of the settings file. This will result in some additional dialogs asking if you are sure you trust the source of the settings file. Once installed, the settings file should include the proper certificate so these additional questions will be avoided in subsequent updates.

Acrobat and Adobe Reader support the use of FDF files to exchange data between the Acrobat family of client and server products. FDF files use a .fdf extension, and like .pdf, it is registered by Adobe so that the required application is used to open these files via a browser or file explorer. Acrobat provides the following FDF features:

- Import and export of digital ID certificates.
- Import and export of server settings for an Adobe LiveCycle Rights Management Server, LDAP directory servers, roaming credential servers, and timestamp servers.
Whether the file is located on a network or emailed, FDF file recipients simply double click on a FDF file to import its data automatically via the FDF import wizard, thereby eliminating the need for error prone, manual configuration.

FDF files provide individuals and businesses with many opportunities for streamlining workflows. For example:

- Alice wants to email her certificate to Bob and wants Bob to reply with his certificate. Alice chooses Request Contact in the Trusted Identity Manager. The workflow generates and emails an FDF file that can contain her certificate, a request for Bob’s certificate, and Alice’s return email address.

- Alice needs to encrypt documents for a number of people in her organization. An administrator sends her an FDF file that contains a large group of contacts. When Alice opens the FDF file, she is walked through the FDF Data Exchange UI wizard so that she can import these contacts into her Trusted Identities list.

- A server wants a copy of Bob’s certificate so that the server can encrypt documents for Bob. The server generates an FDF file that contains a certificate request and a return URL address. When Bob downloads the FDF file from the server, he is walked through the FDF Data Exchange UI wizard where he can respond by allowing his certificate to be returned.

- A company needs to distribute its trusted certificate to customers so that they can verify that the company’s documents are authentic. A server or administrator creates an FDF file that contains the trusted certificate and posts it on a Web server that hosts a Web page with a link to the file. When customers download the file, they are asked whether they wish to add this certificate to the Trusted Identity list and are given the ability to set the certificate’s trust level.

For more information, refer to the following:

- FDF Files and Security
- Importing Application Settings with FDF Files
  - “Responding to an Email Request for a Digital ID” on page 52
  - “Importing Someone’s Certificate” on page 54
  - “Importing Multiple Certificates” on page 55
  - “Importing Timestamp Server Settings” on page 57
  - “Importing Directory Server Settings” on page 59
  - “Importing Adobe LiveCycle Rights Management Server Settings” on page 60
  - “Importing Roaming ID Account Settings” on page 61
  - “Importing a Trust Anchor and Setting Trust” on page 63
- Exporting Application Settings with FDF Files
  - “Distributing a Trust Anchor or Trust Root” on page 42
  - “Setting the Certificate Trust Level” on page 45
  - “Exporting Your Certificate” on page 45
  - “Emailing Your Certificate” on page 46
  - “Saving Your Digital ID Certificate to a File” on page 47
  - “Requesting a Certificate via Email” on page 48
  - “Emailing Server Details” on page 49
  - “Exporting Server Details” on page 51
3.2.1 FDF Files and Security

FDF files are data exchange files. Like acrobatsecurity files, they help you move certificate, server, and other data from one machine to another. This data transfer usually involves some mechanism such as data injection into a PDF form field, installing files, executing a script, and so on. These actions represent a potential security risk, and in some environments that risk may be unacceptable. Acrobat therefore provides a new security feature that, when turned on, disables some FDF functionality unless those FDF files originate from a specifically privileged file, folder, or server.

The new feature is called Enhanced Security and may be enabled or disabled by choosing Edit > Preferences > Security (Enhanced). Table 4 lists the high level rules defining FDF behavior.

**Tip:** If you need to configure your environment for enhanced security or need to troubleshoot FDF workflows that may not be working as expected when enhanced security is on, see Enhanced Security.

<table>
<thead>
<tr>
<th>Action</th>
<th>FDF location</th>
<th>PDF location</th>
<th>8.x behavior</th>
<th>9.x behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opening a target PDF</td>
<td>local</td>
<td>local</td>
<td>PDF opens and no authentication required.</td>
<td>Same.</td>
</tr>
<tr>
<td>Opening a target PDF</td>
<td>local</td>
<td>http server</td>
<td>PDF opens</td>
<td>User authorization required unless trusted via enhanced security feature.</td>
</tr>
<tr>
<td>Opening a target PDF</td>
<td>https server</td>
<td>http server</td>
<td>PDF opens and no authentication required.</td>
<td>Same.</td>
</tr>
<tr>
<td>Opening a target PDF</td>
<td>https server</td>
<td>local</td>
<td>Blocked</td>
<td>Http hosted FDFs cannot open local files.</td>
</tr>
<tr>
<td>Data injection</td>
<td>n/a</td>
<td>n/a</td>
<td>Allowed</td>
<td>Allowed if:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Data returned via a form submit with url#FDF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• FDF has no /FDF key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Cross-domain policy permits it.</td>
</tr>
<tr>
<td>Data injection</td>
<td>server</td>
<td>browser</td>
<td>Allowed</td>
<td>Allowed if:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Link to PDF contains #FDF=url.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• FDF has no /FDF key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Cross-domain policy permits it.</td>
</tr>
<tr>
<td>Data injection</td>
<td>server</td>
<td>Application</td>
<td>Allowed</td>
<td>Allowed if:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• PDF makes EFS POST/GET and FDF sends data in https response to same PDF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Cross-domain policy permits it.</td>
</tr>
<tr>
<td>Data injection</td>
<td>Varied</td>
<td>Varied</td>
<td>Allowed</td>
<td>Authorization required if enhanced security is on and document is not set as a privileged location.</td>
</tr>
<tr>
<td>Script injection</td>
<td>Any</td>
<td>Any</td>
<td>Allowed</td>
<td>Injection is blocked unless if enhanced security is on and FDF is not in a privileged location.</td>
</tr>
</tbody>
</table>
3.2.2 Exporting Application Settings with FDF Files

FDF files can be created by administrators, end users, and even a server. It is a good idea to sign FDF files so that recipients of the file can establish a level of trust for the contents of the FDF file. For example, when an FDF file is signed, the **Accept the level of trust specified by the signer for all contacts in this file** checkbox becomes enabled, thereby allowing the importer to accept the level of trust you have specified.

**Note:** Recipients won’t be able to validate your signature unless you have previously sent them your digital ID certificate or your certificate was issued by someone they already trust.

**Figure 10 Signing an FDF file**

3.2.2.1 Distributing a Trust Anchor or Trust Root

Distributing a trusted certificate from Acrobat involves wrapping one or more certificates in an FDF file and making it available to other users via email, a network directory, or a Web site. Recipients simply click on the file or a link to the file to open the Acrobat wizard which downloads and/or installs the certificate.

**Certificate Chains and Trust Anchors /Roots**

Certificates usually exist as part of a hierarchy or “chain” of certificates, and part or all of the chain can be wrapped in an FDF file. The bottom-most and end user certificate (yours) is called an “end entity” (EE) certificate. The top-most certificate, (the root) is typically belongs to a trusted Certificate Authority (CA). Certificates in between the end entity and root certificates are sometimes called “intermediate certificates” (ICAs) and are issued by the CA or ICAs underneath the CA. Acrobat enables users to specify one or more of the certificates in a chain as trusted for specific operations. Thus, an EE certificate could have one or more trust anchors (trusted ICAs) that chain up to a the top-most CA certificate which is the primary trust anchor or “trusted root.”

A typical chain might include your certificate, your company’s ICA, and a root CA. Certificates inherit trust from certificates on the root end of the chain. For example, if the root certificate is trusted, then any certificates chaining to the that root will also be trusted. Some organizations have their own root CA or use an ICA certificate that is issued by an external CA and make these the trust anchors for their employees.
It is a common practice to trust certificates as high up in the chain as is reasonable since revocation checking starts at the chain bottom and continues until it reaches a trust anchor. Revocation checking occurs until reaching a certificate that is absolutely trusted by you or your organization. It also allows users to trust other certificates that chain up to the same root. The trust anchor is often an ICA for example, since if the root is issued by a company such as VeriSign, it might not be wise to make it a trust anchor as that tells Acrobat to trust the millions of certificates that chain up to VeriSign.

Distributing and installing ICA or CA trust anchors to a user or group of users allows them to:
- Distribute certified or signed documents to partners and customers.
- Help document recipients validate the signatures of document authors.

**Exporting a Trust Anchor**

When Acrobat exports a certificate, it automatically exports other selected certificates in that certificate’s chain and includes them in the FDF file.

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Manage Trusted Identities**.
2. Choose **Certificates** in the **Display** drop-down list.
   In addition to this method, you can also display the certificate from any signature or certificate security method workflow where a **Show Certificate** or **Certificate Details** button appears, such as the Signature Properties dialog.
3. Select the certificate (Figure 12).
   **Note:** In the unlikely event that you can sign the FDF file with a signature the recipient can validate (they will use a different certificate than the one you are exporting), set the certificate’s trust level before exporting it. For details, see “Setting the Certificate Trust Level” on page 45
   **Tip:** You could just choose **Export** and bypass the following two steps. However, exporting the certificate from the Certificate Viewer allows you to see the entire certificate chain where you can select all or part of it.
5. Select a certificate in the chain that appears in the left-hand window.
6. Choose **Export**.

7. Choose one of the following:
   - **Email the data to someone**: Emailing the data automatically creates an FDF file that other Adobe product users can easily import.
   - **Save the exported data to a file**: Acrobat FDF Data Exchange. FDF is a format recognized by the Acrobat family of products.

8. Choose **Next**.
9. **(Optional)** If the Identity Information dialog appears, enter your email address and any other information. If you have already configured your identity details, this screen may not appear. For details, see “Setting Identity Information” on page 350.

10. **Do not sign** if the certificate you use to sign uses the same trust anchor or you are distributing. Since recipients do not have this certificate yet, they will not be able to validate your signature.

    **Note:** Signing the FDF will only be useful if you have a digital ID that the recipient has already trusted (uses a trust anchor OTHER than the one you are currently distributing). The FDF file recipients must also already have that digital IDs certificate so that they can validate your signature without relying on the certificate you are currently sending. This workflow is uncommon, but it does allow recipients to automatically inherit your predefined trust settings for the certificate embedded in the file.

11. Choose **Next**.

12. Continue with the workflow until the trusted root is emailed or placed in a directory where your intended recipients can find it.

**Providing Instructions to the Trusted Root Recipients**

For details, see “Importing a Trust Anchor and Setting Trust” on page 63.

3.2.2.2 **Setting the Certificate Trust Level**

    **Note:** This section is only relevant for trust anchor’s in FDF files that are signed with a trusted signature. This is an unlikely scenario, since the trust anchor distributor is probably using the same trust anchor that is being distributed and the recipient doesn’t have it yet. Most users will likely need to manually set the imported certificate’s trust level.

When distributing a trusted root in a signed file that the FDF recipient can validate, set the certificate trust level:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Manage Trusted Identities**.

2. Choose **Certificates** in the **Display** drop-down list.
3. Highlight the needed certificate.

4. Choose **Edit Trust**.

5. Display the Trust tab.

6. Set the trust level as described in “Importing a Trust Anchor and Setting Trust” on page 63.

### 3.2.2.3 Exporting Your Certificate

You can use FDF files to export your certificate so that others can import it into their list of trusted identities. This enables them to encrypt documents for you and validate your signature for documents that you digitally sign.

- Before users receiving your signed document can validate your signature, they must receive the your certificate or one above it in the trust chain.
- Before users can encrypt a document for you with certificate encryption, they must have access your certificate.

Certificates can be emailed or saved to a file for later use. There are two ways to export a certificate:

- To export a certificate from the list in the Security Settings Console, refer the following:
  - “Emailing Your Certificate” on page 46
  - “Saving Your Digital ID Certificate to a File” on page 47
- To export any certificate displayed in the Certificate Viewer, choose **Export** on the Summary tab.
3.2.2.4 Emailing Your Certificate

If you do not have an email program on your machine, save the data to a file as described in “Saving Your Digital ID Certificate to a File” on page 47 and then send the file as an attachment using your web-based email program.

To email a digital ID certificate:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**.

2. Select **Digital IDs** in the left-hand tree.

3. Highlight an ID in the list on the right. If you have more than one, choose the one that is appropriate for the usage context. For example, send your company-issued ID to those you do business with.

4. Choose **Export**.

5. Choose **Email the data to someone** (Figure 13).

![Figure 13 Digital ID: ID export options](image)

6. Choose **Next**.

7. Enter the recipient’s email address and any other optional information.
8. Choose Email.

9. When the email program opens, send the email.

   **Note:** Some email problems only queue messages to be sent. You may need to start your email client program to cause the message to actually send.

### 3.2.2.5 Saving Your Digital ID Certificate to a File

To save a digital ID certificate to a file:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**.

2. Select **Digital IDs** in the left-hand tree.

3. Highlight an ID in the list on the right.

4. Choose **Export**.

5. Choose **Save the exported data to a file** (Figure 13).

6. Choose a file type:
   - **Acrobat FDF Data Exchange**: FDF files enable the easy exchange of data between any Acrobat family of products.
- **Certificate Message Syntax - PKCS#7**: Save the file as a PKCS7 file. Use this format when the data will be imported into a non-Adobe store such as the Macintosh key store or Windows Certificate Store.

7. Choose **Next**.

8. Browse to a file location and choose **Save**.

9. Choose **Next**.

10. Review the data to export and choose **Finish**.

### 3.2.2.6 Requesting a Certificate via Email

When you request digital ID information from someone, the application automatically attaches to the email an FDF file containing your contact information and certificate.

To request a certificate from someone:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Manage Trusted Identities**.

2. Choose **Request Contact**.

**Figure 15  Emailing a certificate request**

![Email a Request](image)
3. Confirm or enter your identity so that the recipient can identify you. The identity panel is prepopulated if the information has been previously as described in “Setting Identity Information” on page 350.

4. Choose **Include My Certificates** to allow other users to add your certificate to their list of trusted identities.

5. Choose whether to email the request or save it as a file.

6. Choose **Next**.

7. Select one or more digital IDs to export. Highlight contiguous IDs by holding down the Shift key. Highlight non-contiguous IDs by holding down the Control key.

   **Figure 16 Certificates: Selecting a digital ID for export**

<table>
<thead>
<tr>
<th>Name</th>
<th>Issuer</th>
<th>Expires</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ben</td>
<td>Ben</td>
<td>2010.08.10 23:08:02 Z</td>
</tr>
<tr>
<td>Fred Smith</td>
<td>Fred Smith</td>
<td>2004.12.08 11:20:18 Z</td>
</tr>
<tr>
<td>Joe Smith</td>
<td>Joe Smith</td>
<td>2005.11.12 18:36:23 Z</td>
</tr>
<tr>
<td>Johnny Rotten</td>
<td>Johnny Rotten</td>
<td>2005.02.08 20:36:56 Z</td>
</tr>
<tr>
<td>Rose ValidTestCA</td>
<td>CDS QE CA</td>
<td>2009.01.26 06:00:00 Z</td>
</tr>
</tbody>
</table>

8. Choose **Select**.

9. The next step varies depending on whether you chose to email the ID:

   - **If you chose Email**: Enter the person’s email address in the Compose Email dialog and choose **Email**. Send the email message when it appears in the launched email application with the certificate request attached.
   - **If you chose Save as file**: Choose a location for the certificate file Export Data As dialog. Choose **Save**, and then choose **OK**. Tell the intended recipient(s) where to find the file.

### 3.2.2.7 Emailing Server Details

Adobe LiveCycle Rights Management Server, directory server, roaming credential server, and timestamp server details can be exported to an FDF file for distribution to one or more people. Server information sent via an email resides in an attached FDF file. To send directory server details in an email:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**.

2. Select a server category from the left-hand list.

3. Select a server from the right-hand panel.
4. Choose Export.

Figure 17 Security Settings menu items

5. Choose Email the exported data to email the FDF file.

Figure 18 Digital ID Directory servers: Export destination

6. Choose Next.

The Identity panel (Figure 19) will not appear if the information has been previously configured. For details, see “Setting Identity Information” on page 350.
7. Choose **Sign** and complete the signing workflow (**Figure 29**). Sign FDF files so that recipients of the file can easily trust the file and its contents.

8. Choose **Next**.

9. Enter the email information.

10. Choose **Next**.

11. Review the export details.

12. Choose **Finish**.

### 3.2.2.8 Exporting Server Details

Adobe LiveCycle Rights Management Server, directory server, roaming ID, and timestamp server details can be exported to an FDF file for distribution to one or more people. Server information can be written to a file and saved to any location.

To save server details to a file:

1. Choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**.

2. Select a server category from the left-hand list.
Migrating and Sharing Security Settings

3. Select a server from the right-hand panel.

4. Choose Export.

5. Choose **Save the exported data to a file** to save the data in an FDF file that can be shared (Figure 18).

6. Choose **Next**.

   The Identity panel (Figure 19) will not appear if the information has been previously configured. For details, see “Setting Identity Information” on page 350.

7. Choose **Sign** and complete the signing workflow (Figure 29). Sign FDF files so that recipients of the file can easily trust the file and its contents.

8. Choose **Next**.

9. Browse to a location in which to save the file.

10. Choose a file name and choose **Save**.

11. Choose **Next**.

12. Review the export details.

13. Choose **Finish**.

### 3.2.3 Importing Application Settings with FDF Files

There are several ways to import Acrobat and Adobe Reader data from an FDF file:

- By choosing **File > Open**.
- Double clicking on an FDF file (.fdf)

**Tip:** The first two options above automatically invoke the simplest workflow.

- For FDF digital ID information, importing it into the Trusted Identity Manager.
- For FDF server settings, importing it with the Security Settings Console.

#### 3.2.3.1 Responding to an Email Request for a Digital ID

There may be times when someone else needs your digital ID to verify your signature or encrypt a file for you to decrypt (for example, when applying certificate security). To do either, they need access to the public part of your digital ID so that it can be added to their trusted identities list. One way someone can get your ID is to request it in an email.

To request your certificate, a user will simply choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Manage Trusted Identities** and then choose **Request Contact**. Acrobat automatically attaches an FDF file with their public certificate to an email that requests your digital ID. The workflow is essentially a digital ID “trade” that allows two users to exchange digital IDs. You must have a digital ID before responding to the request.

To respond to an email digital ID request:
1. Double click the attached FDF file.

2. Choose **Email your Certificate**.

**Figure 21  Emailing your certificate**

3. Choose a digital ID from the list of existing digital IDs.

**Note:** If you do not have a digital ID or choose **Cancel**, an alert appears that says “A certificate was not selected for export.” Exit the workflow and get a digital ID.

**Figure 22  Selecting a digital ID**

4. Choose **Select**.

5. Review the email details. You can edit the To, Subject, and Body fields (**Figure 23**).

6. Choose **Email**.

7. Send the email through your mail application.
3.2.3.2 Importing Someone’s Certificate

You can use an FDF file to import someone’s certificate into your list of trusted identities. This enables you to validate their signature and encrypt documents with their public key so only that intended recipient can open it.

**Tip:** Importing this information ahead of time enables you to configure your trusted identities list before needing to validate a signature or encrypt a document for someone.

To add someone’s certificate to your list of trusted identities:

1. Click on the FDF file or from Acrobat or Adobe Reader choose **File > Open**. The digital ID certificate may be sent directly from Acrobat as an email attachment or may reside in a networked directory.

2. Review the sender's information when the Import Contact dialog appears.

**Note:** If the file is signed, then the Import Contact dialog will also have a Signature panel as shown in Figure 25.
3. Choose **Set Contact Trust**.

4. When the Import Contact Settings dialog appears, configure the Trust and Policy Restrictions. For details, see "Importing a Trust Anchor and Setting Trust" on page 63.

5. Choose **Certificate Details**.

6. Choose the Details tab.

7. In the Certificate data panel, scroll to MD5-digest and SHA-1 digest and note the fingerprint numbers.

8. Contact the certificate's originator and verify the fingerprints are correct.

9. Choose **OK**.

10. Choose **OK**.

11. Choose **Close**.

### 3.2.3.3 Importing Multiple Certificates

You can use an FDF file to import multiple certificates or a company-wide address book into your list of trusted identities. This enables you to encrypt a document using the public key of the intended recipient so that only they can open it.

**Tip:** Importing this information ahead of time enables you to configure your trusted identities list before needing to validate signature or encrypt a document to those identities. Administrators can create a company-wide address book and can export it to an FDF file for distribution throughout a company via a network or email.

To add multiple certificate to the trusted identities list all at once:

1. Click on the FDF file or from Acrobat or Adobe Reader choose **File > Open**. The digital ID certificate may be sent directly from Acrobat as an email attachment or may reside in a networked directory.
2. If the FDF file is signed, the signature can be validated, AND a trust level has been specified by the sender, check or uncheck **Accept the level of Trust specified by the signer for all Contacts in this file**.

   **Note:** The box is disabled if either of the above conditions are not met. If the FDF is signed by someone you trust but their signature has a status of UNKNOWN, you may be able to simply add the sender to your list of trusted identities. To do so, choose **Signature Properties > Show Certificate** > select the **Trust tab** > and choose **Add to Trusted Identities**.

   - If the checkbox is selected, all contacts associated with this certificate will receive the level of trust that was set by the user that signed the FDF file.
   - If the checkbox is not selected, no trust level will be set for these certificates. The certificate cannot be used for many actions (such as providing a valid timestamp or encrypting) until a trust level is set as described in the user documentation.

3. Choose **Add Contacts to List of Trusted Identities**.

4. If there are multiple contacts in the file, the Choose Contacts to Import dialog appears. Remove those that are not wanted and highlight the rest.

5. Choose **Import**.

6. Choose **OK** in the confirmation dialog.
3.2.3.4 Importing Timestamp Server Settings

In enterprise settings, servers do not usually have to be manually configured. Timestamp server administrators often export the server information to an FDF file which is emailed or made available on a network. Users can import (add) directory server settings through the Security Settings user interface or simply by double clicking on the FDF file containing the data.

To import the server settings:

1. Locate the FDF file: find the file in an email or on the local file system and double click on it.
   The FDF can also be imported through the Security Settings Console by choosing **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**, selecting **Time Stamp Servers** in the left-hand list, and choosing **Import**.

2. Review the sender’s details. Note the following:
   - If the FDF is unsigned, no Signature panel appears in the import dialog.
   - If the FDF is signed, you can use the **Signature Properties** button to find out more information about the sender and the validity of the signature.
3. Review the timestamp server list. Note the following behavior:
   - If there is more than one server listed, all of the servers will be imported even though only one is highlighted.
   - At import time, you will be asked if you want to make the highlighted server the default server.
   - **Note:** If there is more than one server and you do not want to import all of them, highlight those that should not be imported and select **Remove**.

4. Choose **Import**.
   A dialog appears asking if the first (or only) server in the server list should be used as the default.

5. Choose **Yes** or **No**.
   - If **No** is selected, a default timestamp server must be set before timestamps can be used. To set a default timestamp server, choose **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings** > **Time Stamp Servers**, select a server, and choose **Set Default**.

6. After the import completes, choose **OK**.
   The settings are automatically imported and should now appear in your list of Time Stamp Servers.
3.2.3.5 Importing Directory Server Settings

In enterprise environments, administrators often set up user machines or export the configuration details to an FDF file which is emailed or made available on a network. In the latter case, you can import the server settings through the Security Settings Console or simply by double clicking on the FDF file containing the data.

To add server settings from a file:

1. Locate the FDF file: find the file in an email or on the local file system and double click on it.

   The FDF can also be imported through the Security Settings Console by choosing Advanced (Acrobat) or Document (Adobe Reader) > Security Settings, selecting Directory Servers in the left-hand list, and choosing Import.

2. Review the sender's details. Verify the signature properties if needed (Figure 29).

   Note: If the FDF is unsigned, the Signature panel will display Not signed and the Signature Properties button will be disabled.

   Figure 29 Digital ID Directory servers: Importing

   ![Image of FDF file import dialog]


4. If a confirmation dialog appears, choose OK.

   This dialog will not appear if Do not show this message again was previously selected.

5. Choose Close.

   The settings are automatically imported and should now appear in the Directory Servers list in the Security Settings Console.
3.2.3.6 Importing Adobe LiveCycle Rights Management Server Settings

In enterprise settings, administrators often set up user machines or export the configuration details to an FDF file which is emailed or made available on a network. In the latter case, you can import the server settings through the Security Settings Console or simply by double clicking on the FDF file containing the data.

To import the server settings:

1. Locate the FDF file: find the file in an email or on the local file system and double click on it.
   The FDF can also be imported through the Security Settings Console by choosing **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**, selecting **Adobe LiveCycle Rights Management Servers** in the left-hand list, and choosing **Import**.

2. Review the sender’s details. Note the following:
   - If the FDF is unsigned, no Signature panel appears in the import dialog.
   - If the FDF is signed, you can use the **Signature Properties** button to find out more information about the sender and the validity of the signature.

3. Choose **Log In**.
   **Tip:** You must identify yourself to the server before you will be allowed to import these settings. The Import button does is disabled until you log in.
4. Choose **OK**.

5. Choose **Import**.

6. If you do not already have a default Adobe LiveCycle Rights Management Server, a dialog appears asking whether or not you want to make this your default server, choose **Yes** or **No**.

7. Choose **OK**.

   The settings are automatically imported and should now appear in the Adobe LiveCycle Rights Management Servers list in the Security Settings Console.

### 3.2.3.7 Importing Roaming ID Account Settings

In enterprise settings, administrators often set up user machines or export the configuration details to an FDF file which is emailed or made available on a network. In the latter case, you can import the server settings through the Security Settings Console or simply by double clicking on the FDF file containing the data.

To import the server settings:

1. Locate the FDF file: find the file in an email or on the local file system and double click on it.

   The FDF can also be imported through the Security Settings Console by choosing **Advanced** (Acrobat) or **Document** (Adobe Reader) > **Security Settings**, selecting **Roaming ID Accounts** in the left-hand list, and choosing **Import**.

2. Review the sender’s details. Note the following:

   - If the FDF is unsigned, no Signature panel appears in the import dialog.
   
   - If the FDF is signed, you can use the **Signature Properties** button to find out more information about the sender and the validity of the signature.
3. Choose **Import**.

4. Verify the roaming ID account name and server URL.

   **Figure 33** Roaming ID server name and URL

   ![Image of roaming ID server settings]

   **Contents:**
   
   Roaming ID Accounts
   
   cygnet1 (https://cygnet1.acrot.com:9761)

   - Name: My RC Server
   - URL: https://cygnet1.acrot.com:9761

5. Choose **Next**.

6. Enter a user name and password.

   **Tip:** The topmost portion of this dialog is customizable and server-dependant. The fields will remain the same, but the branding will vary.

   **Figure 34** Logging in to a roaming ID server

   ![Image of roaming ID server login]

7. Choose **Next**.
8. After the confirmation that you have downloaded the roaming ID(s) appears, choose Finish.
The server settings and associated certificates are automatically imported and will now appear in the Roaming ID Accounts list in the Security Settings Console.

Figure 35 Downloaded roaming ID certificates

3.2.3.8 Importing a Trust Anchor and Setting Trust

Users occasionally need to import a trust anchor so that certificates that chain up to that anchor will also be trusted. This is particularly true in large organizations, and system administrators often distribute a trust anchor so that everyone within that organization can trust everyone else at the same level for signature workflows. For more information about trust anchors, see “Distributing a Trust Anchor or Trust Root” on page 42.

To import a certificate that will be used as a trust anchor:

1. Open the FDF with one of the following methods:
   - Click on the FDF file. It may be an email attachment or a file on a network or your local system.
   - In Acrobat or Adobe Reader choose File > Open, browse to the FDF file, and choose Open.

   **Note:** It is unlikely that you will receive a signed FDF file containing a trusted root. However, if you do, simply check Accept the level of trust specified by the signer for all contacts in this file and then choose Close. Skip the rest of the steps.

2. For unsigned FDF files containing a trusted root (the most likely case), choose Set Contact Trust.

3. Import the certificates.

4. Do one of the following:
   - If you already have the certificate:
     2. Choose Certificates in the Display drop down list.
     3. Select the certificate.
     4. Choose Edit Trust.

   - If the certificate is in a signature:
     1. Right click and choose Signature Properties.
2. Choose Show Certificate.

3. Select the Trust tab.

4. Choose Add to Trusted Identities.

   **Tip:** If Add to Trusted Identities is disabled, the identity is already on your Trusted Identities list. To change the trust settings, you must use the first method above.

5. On the Trust tab, select the trust options. In enterprise settings, an administrator should tell you which trust settings to use.

   **Note:** During an import action, recipients of the distributed trust anchor may be able to inherit its trust settings. Once you've verified the sender, you usually want to accept these settings so you can use the certificate the way the sender intended.

**Figure 36 Certificate trust settings**

- **Use this certificate as a trusted root:** Makes the certificate a trust anchor. The net result is that any certificates which chain up to this one will also be trusted for signing. At least one certificate in the chain (and preferably only one) must be a trusted root (trust anchor) to validate signatures and timestamps.

  **Tip:** There is no need to make end entity certificates trust anchors if they issued by a certificate holder whose certificate you have configured as a trust anchor. It is best practice to trust the topmost certificate that is reasonable to trust because revocation checking occurs on every certificate in a chain until that anchor is reached. For example, in a large organization, it is likely you would want to trust your company's certificate. If that certificate was issued by VeriSign, you would not want to make...
VeriSign a trusted root unless you wanted to trust every certificate that chains up to VeriSign.

- **Signed documents or data**: Trusts the certificate for approval signatures.
  
  **Tip**: This setting is disabled because if the certificate is set as a trust anchor. Trust anchors are automatically trusted for approval signatures.

- **Certified documents**: Trusts the certificate for certification signatures.
  - **Dynamic content**: Trusts multimedia and other dynamic content in certified documents. Selecting this option automatically adds documents that are certified with this certificate to the Trusted Documents list which is maintained by the Multimedia Trust Manager. For this reason, verify your application environment is configured correctly.
  
  **Tip**: This setting is disabled because if the certificate is set as a trust anchor. Trust anchors are automatically trusted for approval signatures.

- **Embedded high privilege JavaScript**: Trusts embedded scripts. Certificate settings do not override application-level settings, so even if JavaScript is enabled for a particular certificate, it may not execute unless the application’s preferences allow it. This option requires that the application environment be configured correctly.

- **Privileged system operations (networking, printing, file access, etc.)**: Some operations represent a security risk more serious than others. Acrobat considers the following operations potential threats to a secure application operating environment: Internet connections, cross domain scripting, silent printing, external-object references, and FDF data injection. If this checkbox is checked, documents that are certified with this certificate will allow these actions.

  **Tip**: This feature interacts with the Enhanced Security preferences which may be set by choosing **Edit > Preferences > Security (Enhanced)**. The application always takes the most permissive setting when determining what is allowed. For example, if the trust level for this certificate does not allow privileged operations but the certified file resided in a privileged location, then these operations will be permitted.

6. If you need to specify a policy restriction, do so. Most users only need to set policy restrictions at the request of their administrator. “Certificate Trust Settings” on page 337.

7. Choose **OK** twice.

8. Choose **Close**.
Deploying Acrobat or Adobe Reader in enterprise environments usually requires integrating the client installations with existing PKI and security infrastructure.

For details, refer to the following:
- “LDAP Directory Server Integration” on page 66
- “Setting up Signature Validation” on page 69
- “Digital ID Management” on page 70
- “OIDs and Certificates” on page 71
- “Lifecycle Management” on page 70

4.1 LDAP Directory Server Integration

Many enterprise companies use Lightweight Directory Access Protocol (LDAP) directory servers to enable LDAP-aware clients to retrieve email addresses, network services, software directories, and so on. The Acrobat family of products can use LDAP servers as x.509 public key certificate repositories that end users can search for needed certificates. Users may add certificates to their list of trusted identities for two reasons: so that they can encrypt a document for someone with that person’s certificate (use certificate security), and so that they can validate someone’s digital signature.

Both Acrobat and Adobe Reader ship with default server(s) which are listed in the Security Settings console (Figure 37):
- Versions 7.x:
  - VeriSign Internet Directory Service
  - GeoTrust Directory Service
  - IDtree Directory Service
- Version 8.x and later: VeriSign Internet Directory Service
While a user can search the default servers for certificates to add to their list of trusted identities, it is more likely that they will use LDAP servers to access their coworker’s certificates to encrypt documents for an internal workgroup or to find the trust chain of a signer’s certificate stored on the LDAP server during signature validation.

Therefore, administrators should configure their LDAP server and Acrobat/Adobe Reader installations as described below:

- “Configuring an LDAP Server to Look Up Certificates” on page 67
- “Configuring Clients to Search a Directory Server for Certificates” on page 67

4.1.0.1 Configuring an LDAP Server to Look Up Certificates

In order for Acrobat’s Trusted Identity Manager to search for certificates on an LDAP server, the server must be able to respond to a specific LDAP scheme. In addition to returning standard, generic information, the server must be able to respond to the request `userCertificate;binary` as defined by RFC 4523, Section 2.1.

**Note:** While Acrobat is itself case insensitive, LDAP servers may not be. Therefore, the LDAP server should be capable of handling the specific request that Acrobat sends. Verify your server uses the exact format of `userCertificate;binary`.

4.1.0.2 Configuring Clients to Search a Directory Server for Certificates

Administrators can configure directory servers in several ways:

- **Pre-installation (Recommended):** The administrator can tune the client installer with the Adobe Customization Wizard 8 prior to installing the application on various machines. The process involves configuring a directory server in your template application installation and using the wizard to copy the `directories.acrodata` file to the tuned installer.

- **Post-installation setting distribution:** The administrator can export server configuration details in an FDF file and distribute it across the organization. Users can automatically import the settings by double clicking on the file. This method is useful for configuration of an existing client installation. For details, refer to “Exporting Server Details” on page 51.

- **Post-installation instruction distribution:** The administrator can tell users how to configure the server manually.
To configure a directory server:

1. Choose **Advanced > Security Settings**.
2. Select **Directory Servers** in the left-hand list (Figure 37).
3. Choose **New**.

   *Figure 38  Digital ID Directory servers: Setting server details*

4. Configure the LDAP server settings in the Edit Directory Server dialog:
   - **Directory Name**: An arbitrary directory name.
   - **Access Type**: LDAP is the only type supported.
   - **Server Name**: The server name.
   - **Port**: The server port.
   - **Search Base**: A comma-separated list of name-value pairs used in the search. For example, `c=us,cn=Brown Trout,ou=Engineering` for country, common name, organizational unit.
   - **This server requires me to log on**: Check if the server requires a username and password.
   - **User name**: Leave blank.
   - **Password**: Leave blank.
   - **Timeout**: The number of seconds to keep trying to connect.
   - **Maximum Number of Records to Receive**: The number of records to return up to 10.

5. **Optional**: Make the server the default one to search by highlighting the directory server in the right-hand panel and choosing **Set Default**.

6. Choose **OK** if a confirmation dialog appears. A star appears next to the name of the selected server.
4.2 Setting up Signature Validation

For an end user to validate someone’s signature, they must trust the signer’s certificate for signing or some other certificate it chains up to. Trusting a certificate involves adding it to their trusted identity list in the Trusted Identity Manager and manually setting its trust level. Home users often exchange certificates as needed when using certificate security or add certificates on-the-fly directly from signatures in signed documents and then set trust levels on a per-certificate basis.

However, enterprise settings often require that their employees can easily validate each others’ signatures without any manual tasks. Because Acrobat will trust all end entity certificates for signing and certifying that chain up to a trust anchor (a certificate specifically trusted for those actions), administrators should preconfigure client installations or help their end users add a trust anchor.

Administrators can configure a trust anchor by the following:

- **Pre-installation (Recommended):** The administrator can tune the client installer with the Adobe Customization Wizard 8 prior to installing the application company-wide. The process involves adding a company certificate to the Trusted Identity Manager, setting it’s trust level, and using the wizard to copy the `addressbook.acrodata` file to the tuned installer.

To set up a trust anchor for distribution:

1. Set up a trust anchor in your template application as described in “Distributing a Trust Anchor or Trust Root” on page 42. This action creates the `addressbook.acrodata` file.

2. Add the `addressbook.acrodata` file to the installer as described in “Advanced Installer Tuning for Power Users” on page 32.

- **Post-installation setting distribution:** The administrator can export a trust anchor in an FDF file and distribute it across the organization. Users can automatically import the certificate by double clicking on the file. This method is useful for configuration of an existing client installation. For details, refer to “Distributing a Trust Anchor or Trust Root” on page 42.
4.3 Digital ID Management

4.3.1 PKCS#11 Integration

If your company uses PKCS#11 devices such as smart card readers or tokens, distribute the drivers or modules at deployment time. Most configuration must occur manually.

1. Set \texttt{cAdobe_P11CredentialProvider} as described in “PKCS#11 Configuration” on page 81.
2. Distribute the module file, such as \texttt{dkck201.dll} with instructions where to locate it.
3. Instruct users to restart Acrobat.

4.3.2 Lifecycle Management

Managing digital IDs throughout their lifecycle is currently beyond the scope of this document. However, common tasks for a PKI administrator include:

- Distributing Certificates
- Backing up Public Keys
- Escrowing Private Keys
- Renewing Keys
- Enroll new users
- Expiration and revocation of IDs

\textbf{Figure 40 Digital ID: Components}
4.4 OIDs and Certificates

4.4.1 What is an Object Identifier? (OIDs)

An object identifier (OID) is simply another name for an object. OIDs are a dot-separated series of numbers. Each dot-separated number has a specific meaning. In Acrobat, the series is made up of an immutable arc (1.2.840.113583) and subarcs such as:

\{iso(1) member-body(2) us(840) adbe(113583)\}

In the context of computer security, OIDs name nearly all X.509 certificate object types, including components of policies, distinguished names, CPSs, and so on. OIDs are associated with objects in data structures defined using the standard Abstract Syntax Notation number One (ASN.1) so that OIDs may be generated and processed by client and server software.

Tip: OIDs are like the Internet domain name space: organization that wish to create an OID gets a root OID assigned to them off of which they can create their own OIDs. There is a very large and relatively standardized set of OIDs that already exists. Most CAs do not create new OIDs. OIDs are typically attached to a certificate when it is created by a certificate authority using 3rd party software.

For example, certificates can be associated with a policy represented by a numeric string (the OID) that controls how Acrobat will behave. When Acrobat encounters a certificate, it processes the OIDs, and in this case it looks for a corresponding certificate policy and changes its behavior accordingly. Acrobat recognizes two kinds of certificate policy OIDs:

- **Acrobat-conformant OIDs**: These OIDs conform to the format described in this specification. There are a limited number of predefined OIDs, and each OID is associated with specific Acrobat behavior. These OIDs have a structure defined in X.208 from the International Telecommunications Union (ITU).

- **Arbitrary OIDs**: Any user-defined OID may be associated with a certificate.

4.4.2 Adobe OIDs

<table>
<thead>
<tr>
<th>Category</th>
<th>OID</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe-Arc</td>
<td>1.2.840.113583</td>
<td>All the Adobe-defined OIDs must be under this arc.</td>
</tr>
<tr>
<td>Acrobat</td>
<td>Acrobat-Arc.1</td>
<td>Acrobat OIDs.</td>
</tr>
<tr>
<td>Security</td>
<td>Acrobat.1</td>
<td>OIDs for managing PKCS#7 and PKCS#12 files.</td>
</tr>
<tr>
<td>CPS</td>
<td>Acrobat.2</td>
<td>Certificate Practice Statement OIDs subarc.</td>
</tr>
<tr>
<td>Ubiquity</td>
<td>Acrobat.7</td>
<td>Ubiquity OIDs subarc.</td>
</tr>
<tr>
<td>XS09-Extension</td>
<td>Acrobat.9</td>
<td>Contains all OIDs used to define Adobe proprietary X509 extensions. Note, an OID is defined under this arc, only if it doesn’t fall under any of the above arcs (e.g. Ubiquity Sub Rights extension is defined under Ubiquity arc whereas  extension is defined under this arc).</td>
</tr>
</tbody>
</table>
4.4.3 Security OIDs

### Table 4 Security OIDs

<table>
<thead>
<tr>
<th>Name</th>
<th>OID</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>Security.1</td>
<td>Acrobat 7 and earlier: Special casing for APF file encryption/decryption using algorithms not directly supported by BSafe.</td>
</tr>
<tr>
<td>DefaultSigningCredential</td>
<td>Security.2</td>
<td>Acrobat 6 and earlier: Sets the default signing credential when a P12 file contains more than one digital ID.</td>
</tr>
<tr>
<td>DefaultEncryptionCredential</td>
<td>Security.3</td>
<td>Acrobat 6 and earlier: Sets the default encryption credential when a P12 file contains more than one digital ID.</td>
</tr>
<tr>
<td>PasswordTimeout</td>
<td>Security.4</td>
<td>Set within the P12 file SecretBag along with the timeout value.</td>
</tr>
<tr>
<td>AuthenticDocumentsTrust</td>
<td>Security.5</td>
<td>Used in ExtendedKeyUsage X509 extension. The presence indicates that the particular credential can be used for CDS.</td>
</tr>
<tr>
<td>DynamicContentTrust</td>
<td>Security.6</td>
<td>NOT USED CURRENTLY.</td>
</tr>
<tr>
<td>UbiquityTrust</td>
<td>Security.7</td>
<td>Used in ExtendedKeyUsage X509 extension. The presence indicates that the particular credential can be used for CDS.</td>
</tr>
<tr>
<td>RevInfoArchival</td>
<td>Security.8</td>
<td>Used in PKCS#7 signature to identify the revocation information archival attribute.</td>
</tr>
<tr>
<td>X509-Extension</td>
<td>Security.9</td>
<td>Contains all OIDs used to define Adobe proprietary X.509 extensions. Note: an OID is defined under this arc only if it doesn't fall under any of the above arcs (e.g. Ubiquity Sub Rights extension is defined under Ubiquity arc whereas Timestamp extension is defined under this arc).</td>
</tr>
<tr>
<td>PPLKLiteCredential</td>
<td>Security.10</td>
<td>Used in digital IDs created in Acrobat/Reader v8.0 and later.</td>
</tr>
</tbody>
</table>

### Example 4.1: RevInfoArchival signature property format

```
adbe-revocationInfoArchival OBJECT IDENTIFIER ::= { adbe(1.2.840.113583) acrobat(1) security(1) 8 }

RevocationInfoArchival ::= SEQUENCE {
  crl [0] EXPLICIT SEQUENCE of CRLs OPTIONAL,
  ocsp [1] EXPLICIT SEQUENCE of OCSP Responses OPTIONAL,
  otherRevInfo [2] EXPLICIT SEQUENCE of OtherRevInfo OPTIONAL }

OtherRevInfo ::= SEQUENCE {
  Type OBJECT IDENTIFIER
  ValValue OCTET STRING
}
```

4.4.4 Certificate Practice Statement OIDs

### Table 5 CPS OIDs

<table>
<thead>
<tr>
<th>Name</th>
<th>OID</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthenticDocuments</td>
<td>CPS.1</td>
<td>CDS CPS OID. Present in the Certificate Policies X509 extension. Acrobat doesn't look at this OID.</td>
</tr>
</tbody>
</table>
4.4.5 Ubiquity OIDs

<table>
<thead>
<tr>
<th>Name</th>
<th>OID</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>CPS.2</td>
<td>Test CPS OID. Indicates that the given certificate has been issued for testing purposes only, and brings up the infamous dialog.</td>
</tr>
<tr>
<td>Ubiquity</td>
<td>CPS.3</td>
<td>Ubiquity CPS OID. Acrobat doesn't look at this OID.</td>
</tr>
<tr>
<td>Adhoc</td>
<td>CPS.4</td>
<td>Indicates that the certificate has been issued under an &quot;Adhoc&quot; policy, where the certificate isn't supposed to provide any kind of authentication.</td>
</tr>
</tbody>
</table>

Table 6 Ubiquity OIDs

<table>
<thead>
<tr>
<th>Name</th>
<th>OID</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>UbiquitySubRights</td>
<td>Ubiquity.1</td>
<td>Adobe proprietary X509 extension. Indicates the UB sub rights that can be enabled by this credential.</td>
</tr>
</tbody>
</table>

Example 4.2: Ubiquity Sub rights format

```
adobe-readerEnablement OBJECT IDENTIFIER ::=  { adbe(1.2.840.113583) acrobat(1) security(1) ubiquity(7) 1 }

UbiquityRights ::= SEQUENCE {
  version             INTEGER  { v1(1) }, -- extension version
  ubSubRights         UBSubRights,
  mode                DeploymentMode
}

UBSubRights ::= BIT STRING {
  FormFillInAndSave(0),
  FormImportExport(1),
  FormAddDelete(2),
  SubmitStandalone(3),
  SpawnTemplate(4),
  Signing(5),
  AnnotModify(6),
  AnnotImportExport(7),
  BarcodePlaintext(8)
  AnnotOnline(9),
  FormOnline(10),
  EFModify(11)
}

DeploymentMode ::= ENUMERATED {
  evaluation  (0), -- Eval cert. Docs are disabled when certificate is invalid.
  production  (1) -- Production cert. Docs remain valid for eternity.
}
```
4.4.6 X.509 Extension OIDs

<table>
<thead>
<tr>
<th>Name</th>
<th>OID</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time-stamp</td>
<td>X509-Extension.1</td>
<td>Adobe proprietary X.509 extension. Provides the server with the URL to signatures created using this credential. NOT EXPOSED TO EXTERNAL WORLD.</td>
</tr>
<tr>
<td>ArchiveRevInfo</td>
<td>X509-Extension.2</td>
<td>Adobe proprietary X.509 extension. Indicates that the signatures created using this credential should archive the relevant revocation info. NOT EXPOSED TO EXTERNAL WORLD.</td>
</tr>
</tbody>
</table>

**Example 4.3: Time-stamp X.509 extension format**

```plaintext
adbe-OBJECT IDENTIFIER ::= { adbe(1.2.840.113583) acrobat(1) security(1) x509Ext(9) 1 }

 ::= SEQUENCE {
    version INTEGER  { v1(1) }, -- extension version
    location GeneralName (In v1 GeneralName can be only uniformResourceIdentifier)
    requiresAuth boolean (default false), OPTIONAL
}
```

**Example 4.4: ArchiveRevInfo extension format**

```plaintext
adbe-archiveRevInfo OBJECT IDENTIFIER ::= { adbe(1.2.840.113583) acrobat(1) security(1) x509Ext(9) 2 }

ArchiveRevInfo ::= SEQUENCE {
    version INTEGER  { v1(1) }, -- extension version
}
```
This chapter describes the application preferences for the Acrobat family of products’ digital signature and document security features. Before continuing, you should know that:

- The tables and examples use the Windows registry. Most are applicable to Macintosh, Unix, and Linux systems.
- The root security directory for registry settings stored on a per-user basis are at: HKEY_CURRENT_USER\Software\Adobe\<application>\<version number>\Security\.
- The examples use Acrobat; other applications may provide different menu options.
- The security preferences folder does not appear in the registry until after the Acrobat product installation and a security feature is used. Subdirectories also appear as the code is exercised.

For details, about specific preferences, refer to the following:

- “Setting Basic Client and Workflow Preferences” on page 75
- “Content Security” on page 101
- “Digital Signatures” on page 104

**Caution:** Adobe strongly recommends that you do not make changes to the registry unless you are knowledgeable about editing and troubleshooting Microsoft Windows registry settings. Improper use of this feature can result in the corruption of critical system files.

### 5.1 Setting Basic Client and Workflow Preferences

#### 5.1.1 Preventing End-User Modification

While many lockable preferences have a matching corollary stored in HKCU, their editability via the user interface is controlled by their boolean lockdown counterpart in HKLM. When marked as uneditable, the application user interface item associated with that preference is disabled. The hierarchy within FeatureLockDown is typically the same as the one under the HKCU Security directory.

The Adobe Customization Wizard provides a UI for modifying some of these keys when tuning the client installer for Windows prior to deployment. However, because some lockable keys are not exposed in the wizard, it is often simpler to modify the keys manually and then use the Wizard to drag and drop the configured registry directories to the installer.

To lock down features:

1. Navigate to:
   
   - **7.x:** HKEY_LOCAL_MACHINE\SOFTWARE\Adobe\<product>\<version>\FeatureLockDown
   - **8.x and later:** HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\<product>\<version>\FeatureLockDown
64 bit Windows: HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Adobe\FeatureLockdown.

2. Create a directory path that mirrors the path in HKCU. For example, when configuring a digital signature key, create a cSecurity directory.

   **Tip:** The HKLM path usually mirrors the HKCU path, but is sometimes different. For details about all keys besides those listed here, refer to the Preference Reference for the Acrobat family of Products.

3. Add the requisite subdirectories. Use the same path used for the settings described in the rest of this chapter. For example, create a directory cPPKLite to set bAllowPasswordSaving"=dword:00000000. See Example 5.5.

4. **Predeployment installer tuning via the Wizard:** When you finish your other registry modifications, use the wizard UI to drag and drop the FeatureLockDown directory from My Computer to the Destination Computer.

   **Tip:** Most of the lock down keys are booleans indicating whether the preference is editable by end users. For more detail about a particular setting, refer to the Preference Reference for the Acrobat family of Products. When looking for a key, search for the key name without the data type prefix. For example, search for PrivKey and NOT bPrivKey since that setting is not a boolean in HKCU.

<table>
<thead>
<tr>
<th>Preference</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAllowPasswordSaving</td>
<td>Various</td>
<td>Caches passwords so they don’t have to be re-entered when accessing digital IDs, policies, and other features that use passwords. HKLM and HKCU</td>
</tr>
<tr>
<td>bPrivKey</td>
<td>Certificate handling</td>
<td>Prevents a user from changing the security handler used for signing and certificate security. HKLM and HKCU</td>
</tr>
<tr>
<td>bVerify</td>
<td>Signature validation</td>
<td>Prevents a user from changing the security handler used for the default signature verification method. HKLM and HKCU</td>
</tr>
<tr>
<td>bVerifyUseAlways</td>
<td>Signature validation</td>
<td>Qualifies the use of aVerify. HKLM and HKCU</td>
</tr>
<tr>
<td>bValidateOnOpen</td>
<td>Signature validation</td>
<td>Forces signature validation when a document opens. HKLM and HKCU</td>
</tr>
<tr>
<td>bReqRevCheck</td>
<td>Signature validation</td>
<td>Requires revocation checking to behave as specified. HKLM and HKCU</td>
</tr>
<tr>
<td>bReasons</td>
<td>Signing</td>
<td>Prevents users from modifying the reasons setting. v8.1: If locked and cReasons if empty, bAllowSigningReasons is 0 and read only. If locked and cReasons has values, then bAllowSigningReasons is true and read only. HKLM only</td>
</tr>
</tbody>
</table>
Table 8 Registry preferences: subject to feature lockdown

<table>
<thead>
<tr>
<th>Preference</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bSuppressStatusDialog</td>
<td>Signing</td>
<td>Deprecated since 8.0. Prevents the Document Status dialog from appearing when a certified document opens. HKLM and HKCU</td>
</tr>
<tr>
<td>bWinCacheSession Handles</td>
<td>Signing</td>
<td>(v 8.1 Windows only) Default: 1 Path: cPPKHandler Specifies whether to retain CSP handles when a user authenticates to a digital ID. If true, a user does not have to reauthenticate to use the ID unless they log out or the session ends. The impact of this preference will vary based on the CSP in use; however, the setting does not affect the Windows CSPs. HKLM only</td>
</tr>
<tr>
<td>bAllowInvisibleSig</td>
<td>Signing and</td>
<td>Prevents user from signing with an invisible certification signature. Disables the menu option in the signing menus. HKLM and HKCU</td>
</tr>
<tr>
<td></td>
<td>document security</td>
<td></td>
</tr>
<tr>
<td>bAllowAPSConfig</td>
<td>Document security</td>
<td>v 8.1 (Windows only) Default: 1 Prevents a LiveCycle Right Management Server from being configured by disabling the menu option in the Security Settings Console. HKLM only</td>
</tr>
</tbody>
</table>

Example 5.5: Lockdown keys

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cDefaultLaunchAttachmentPerms]"tBuiltInPermList"="version:1;ade:3;adp:3;app:3;asp:3;bas:3;bat:3;bz:3;bz2:3;chm:3;class:3;cmd:3;com:3;command:3;cpl:3;crt:3;csf:3;desktop:3;exe:3;fxp:3;gz:3;hex:3;hlp:3;hqx:3;hta:3;inf:3;ini:3;ins:3;isp:3;its:3;job:3;js:3;jse:3;ksh:3;lnk:3;lzh:3;mad:3;maf:3;mag:3;mag3:3;maq:3;mar:3;mas:3;mat:3;mau:3;mov:3;maw:3;mda:3;mdc:3;mdm:3;mdw:3;mdz:3;msc:3;msi:3;msp:3;mt:3;ocx:3;ops:3;pdf:3;pic:3;gif:3;pix:3;prf:3;prg:3;pst:3;rar:3;reg:3;scf:3;scr:3;scp:3;sea:3;shb:3;shs:3;sit:3;tar:3;tgz:3;ttmp:3;url:3;vbs:3;vbe:3;vbs:3;vsmacros:3;vss:3;vst:3;vsx:3;webloc:3;ws:3;wsc:3;wsp:3;wsh:3;zip:3;zlo:3;zoo:3;pdf:2;pdf:2"

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cDefaultLaunchURLPerms]"tSchemePerms"="version:1;shell:3;hcp:3;ms-help:3;ms-its:3;its:3;mk:3;mhtml:3;help:3;disk:3;afp:3;disks:3;telnet:3;ssh:3;javascript:1;vbscript:1;acrobat:2;mailto:2;file:2"

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cDocumentStatus]"bSuppressMessageBar"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cDigSig]"bValidateOnOpen"=dword:00000000

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cSecurity\cEDC]"bAllowAPSConfig"=dword:00000000

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Adobe Acrobat\8.0\FeatureLockDown\cSecurity\cHandlers]"bVerify"=dword:00000000
"bPrivKey"=dword:00000000
"bVerifyUseAlways"=dword:00000000
5.1.2 Importing Security Settings

One of Acrobat 9.0’s major new security features includes the ability to import and export security settings via an .acrobatsecuritysettings file, thereby enabling easier version upgrades as well as configuration of multiple machines.

The security settings import/export features offers several advantages over FDF files:

- Most document security and digital signature related settings can be encapsulated in an .acrobatsecuritysettings file whereas FDF could only transport one setting type and a time and could not encapsulate registry settings at all.
- One file can be used instead of many files.
- Trust can be assigned to imported on the fly, thereby simplifying workflows.
- Files can be signed and encrypted.

Use security settings files to backup and restore settings, to distribute settings in a workgroup or enterprise, and to send specific information to another user. Importing settings simply involves importing a file from a network (including automatically from a server) that has been exported from Acrobat and has then been made available from a trusted source.

**Figure 41 Security settings: Export dialog**

The following options are available:

- Specifying whether or not to poll a server for settings to import at regular intervals.
- Configuring whether or not the user should grant permission prior to installing new settings.
- Specifying a particular certificate so the signed settings will only be imported from a trusted source.
## Table 9 Registry preferences: Security setting import

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bAskBeforeInstalling | bool   | (v. 9.0) Default: 1  
Path: `<security root>\DigSig`  
Maps to GUI item: **Ask before installing** checkbox in Edit > Security > Security Settings panel.  
Specifies whether the settings should be imported silently or Acrobat should ask permission from the user. |
| bLoadSettingsFromURL | bool   | (v. 9.0) Default: 0; 10: 1  
Path: `<security root>\DigSig`  
Maps to GUI item: **Load security settings from a server** in Edit > Security > Security Settings panel.  
Specifies whether or not security settings should be automatically imported from a server at the specified time interval. |
| iCheckEvery         | int    | (v. 9.0) Default: 2419200  
Path: `<security root>\DigSig`  
Maps to GUI item: **Check every** radio buttons in Edit > Security > Security Settings panel.  
The polling interval to check the server for updated settings to import. Specifies the number of seconds it should wait between checks for updates. The default value is 30 days. The options are:  
604800: 1 week  
1209600: 2 weeks  
2419200: 1 month  
7257600: 3 months |
| iResourceId         | int    | (v. 9.0) Default: null  
Path: `<security root>\DigSig`  
An internally used number created by Acrobat when it first sets up the "resource" pointed to by the URL. It is not user customizable. |
| tLoadSettingsURL    | text   | (v. 9.0) Default: null  
Path: `<security root>\DigSig`  
Maps to GUI item: **URL** text box in Edit > Security > Security Settings panel.  
The server URL where the acrobatsecuritysettings file to import resides. |
| tLoadSettingsCERT   | cab    | (v. 9.0) Default: null (allows any certificate)  
Path: `<security root>\DigSig`  
Maps to GUI item: **Settings must be signed by** field in Edit > Security > Security Settings panel.  
Specifies a certificate that must be used to sign the imported security settings file. The value is a hexadecimal string corresponding to the SHA-1 hash of the certificate used to sign the settings file. |
| xdata               | string | (v. 9.0) Default: null  
Path: `<security root>\DigSig\cLastChecked`  
Binary data used for internal purposes. It is not set during installation or for tuning pre-deployment clients. It can safely be deleted in an existing environment. |
5.1.3 Digital ID Management

Digital ID preferences control or record the user’s personal digital ID behavior for certificate security and/or digital signatures. Some preferences are not customizable and are listed only for information purposes (Table 10). You can set the following:

- “Allowing Self Sign Digital IDs” on page 80
- “PKCS#11 Configuration” on page 81
- “Roaming ID Configuration” on page 81

5.1.3.1 Specifying extended certificate information

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cExtendedCertInfo</td>
<td>cab</td>
<td>(v 8.0) Default: null Path: &lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains a subkey for each certificate with extended information provided by attribute certificates. The subkeys take the form c{DIGEST} where {DIGEST} is a SHA-1 digest of the associated certificate’s public key encoded as hexadecimal. For example, cPubSec\cExtendedCertInfo\cAD6716326BDAC87628DFAD6716326. Each subkey contains the friendly name, related ID card, and associated attribute certificates.</td>
</tr>
<tr>
<td>cCertIssuerInfo</td>
<td>cab</td>
<td>(v 8.0) Default: null Path: &lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains a subkey for each certificate with extended information. The subkeys take the form c</td>
</tr>
</tbody>
</table>

5.1.3.2 Allowing Self Sign Digital IDs

By default, users can create self signed digital IDs. However, if you would like to prevent users from creating their own IDs, turn this feature off. Disabling this option prevents users from selecting Create a self-signed ID option in Add ID workflows.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bSelfSignCertGen</td>
<td>bool</td>
<td>(v 7.0) Default: 1 Path: &lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maps to GUI item: Create a self-signed digital ID for use with Acrobat.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Turns on and off the Create a self-signed ID option in Add ID workflows so that a user can create a self signed digital ID.</td>
</tr>
</tbody>
</table>
5.1.3.3 PKCS#11 Configuration

The key cAdobe_P11CredentialProvider\ contains a list of P11 modules the user has loaded by choosing Attach Modules in the Security Settings console. By specifying a valid path to a PKCS#11 DLL, modules can be pre-attached to installed clients. Because various errors appear as a result of a bad filename or pointing to a dll that is not a valid PKCS#11 module, test the settings and file before distributing them.

The following options are available:

- Preconfiguring the key when tuning the installer and distributing the module file or when modules are already installed.
- Setting the default browse path in which to look for additional modules.

Table 12 Registry preferences: PKCS#11

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| cModules      | Array of strings | (v. 7.0) Default: null  
Path: <security root>\cASPKI\cAdobe_P11CredentialProvider\  
Maps to GUI item: Attach Module in Security Settings console.  
Array of dynamic library paths to PKCS#11 modules. These may not necessarily be full paths but just something that the OS dynamic library loading functions will accept. For example, t0 may be a path to C:\WINDOWS\system32\dkck201.dll. |
| cP11Path      | ASPATH     | (v. 7.0) Default: null  
Path: <security root>\cPubSec  
Stores the last folder in which the user browsed for a P11 module. The next time the user goes to add a P11 module browsing starts in that folder. |
| cP11Credentials | CAB       | (v. 7.0) Default: null  
Path: <security root>\cPPKHandler  
Contains an array of subcabs for all known PKCS#11 digital IDs. The format is as follows:  
xCert: Binary value of the certificate  
xTokenKey: Binary value generated from the IDs PKCS#11 token. The binary value is generated with the following method: Initialize SHA-1 digest, add the digest the value of the token label, token manufacturer, token model, and token serial. Finish the SHA-1 digest operation. The resulting 20-byte value is the token key. |

5.1.3.4 Roaming ID Configuration

These preferences allow you to configure an application to use roaming IDs. While the needed configuration can be handled through the user interface by end users, you can set the following:

- **Specifying a Default Roaming ID Server:** When a user adds a roaming ID account through the GUI, a dialog asks for a friendly name and a server URL. If no other accounts have been configured and cDefaultServerInfo exists in the preferences (Table 13), its values populate both the friendly server name and URL fields in the Add a Roaming ID dialog.

- **Specify one or more authentication methods:** See “Roaming ID Authentication Mechanisms” on page 97.
Roaming ID Provider Persistent Storage

These preferences store roaming ID server data. Some values are provided by the user and some are provided by the server.

*Note:* These keys do not need to be customized and are provided for informational purposes only.

### Table 13  Registry preferences: Roaming ID server

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| tServerName   | text | (v. 8.0) Default: null  
*Path:* `<security root>`\cASPKI\cAdobe_RoamingID\cDefaultServerInfo  
A user friendly roaming ID server name. |
| tURL          | text | (v. 8.0) Default: null  
*Path:* `<security root>`\cASPKI\cAdobe_RoamingID\cDefaultServerInfo  
A roaming ID server URL. |

### Roaming ID Server-Set Preferences

The preferences in Table 15 are typically created as a result of communications with a roaming ID server.

*Note:* Whether or not you customize these settings is determined by the needs or your particular implementation.

### Table 14  Registry preferences: Roaming ID account

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| cAccounts     | cab  | (v 8.0) Default: null  
*Path:* `<security root>`\cPPKHandler\cRC<version>  
Contains entries for user accounts on roaming ID servers that the provider knows about. Every account is identified by a unique 9-character key such as cAB2CFEC0. |
| cRecentServerURLs | cab  | (v 8.0) Default: null  
*Path:* `<security root>`\cPPKHandler\cRC<version>  
Contains an array of roaming ID server URLs recently entered by the user. |

### Table 15  Registry preferences: Roaming ID user account (set by the server)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| tFriendlyName | text | (v 8.0) Default: null  
*Path:* `<security root>`\cPPKHandler\cRC<version>  
The friendly name created by the user for user interface display purposes. |
| tServerURL    | text | (v 8.0) Default: null  
*Path:* `<security root>`\cPPKHandler\cRC<version>\accounts\<accountname>  
The roaming ID server URL on which this account exists. |
### Registry and plist Settings

#### Digital ID Management

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cCredentials</td>
<td>cab</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;An array of certificates corresponding to digital IDs available through this account. The certificates are in the binary X.509 format.</td>
</tr>
<tr>
<td>cSAML_Assertion</td>
<td>cab</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;Holds an encrypted SAML assertion obtained during last successful authentication. Possession of this assertion is proof of a user's identity. Therefore, the assertion is encrypted using 256-bit AES algorithm in CBC mode. The encryption key is stored in Microsafe database that is protected by the OS login. There are two binary entries under the cSAML_Assertion cab: xEncryptedData contains the encrypted assertion, 'xIV' contains the initialization vector used by the AES encryption algorithm for this assertion.</td>
</tr>
<tr>
<td>tSAML_Assertion_Exp</td>
<td>text</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;Holds the time after which roaming ID provider will not attempt to use the SAML assertion stored in cSAML_Assertion. This time is calculated when assertion is first obtained and takes into account clock difference between the client machine and the server that generated the assertion. Time is represented in BER GeneralizedTime format without the type and length octets.</td>
</tr>
<tr>
<td>tSAML_Assertion_Source</td>
<td>text</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;Holds the URL of the authentication server from which the SAML assertion stored in cSAML_Assertion was obtained.</td>
</tr>
<tr>
<td>tSAML_Name_Value</td>
<td>text</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;SAML_NAME_&lt;Value, Format, Qualifier&gt; comprise the subject name identifier taken from the SAML assertion received during the account's last user authentication. The identifier is essentially a machine-readable user name that is unaffected by the choice of authentication mechanisms.</td>
</tr>
<tr>
<td>tSAML_Name_Format</td>
<td>text</td>
<td>Default: null&lt;br&gt;&lt;br&gt;<strong>Path:</strong> <code>&lt;security root&gt;</code>\cPPKHandler\cRC&lt;version&gt; &lt;br&gt;The value is provided by the server.&lt;br&gt;SAML_NAME_&lt;Value, Format, Qualifier&gt; comprise the subject name identifier taken from the SAML assertion received during the account's last user authentication. The identifier is essentially a machine-readable user name that is unaffected by the choice of authentication mechanisms.</td>
</tr>
</tbody>
</table>
Kerberos Authentication Mechanism

This option is only relevant if the ASSP-Kerberos SPI is selected as described in "Roaming ID Authentication Mechanisms" on page 97.

### Table 16 Registry preferences: Kerberos

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sServiceName</td>
<td>string</td>
<td>(v. 8.0) Default: ASSP. The administrator-specified roaming ID Kerberos service name. If the key is not present, the default value of ASSP is assumed. If the key is present and the value is empty string, Acrobat asks the roaming ID service for it's Kerberos service name. This method is not secure and enterprises are advised not to use this option.</td>
</tr>
</tbody>
</table>

5.1.3.5 Setting Digital ID Defaults

Most digital ID default values are set by the application when a user first uses an ID or manually specifies a default value in the Security Settings Console. Moreover, since user actions will overwrite some preconfigured value an administrator might provide, setting many of these properties is usually not worthwhile. However, it is possible and the following options are available:
- Specifying a default URL to obtain a new digital ID. This value is NOT overwritten by user actions.
- Listing a set of attribute certificates.
- Specifying a default signing ID. This value is end user-specific.
- Specifying a default encryption ID. This value is end user-specific.
- Customizing a default directory server used to locate certificates that can be imported into the Trusted Identity Manager.

**Caution:** Acrobat 9.0 users who configure a 3rd party security handler plugin may find that their non-default choice does not stick if the plugin calls `PSUNregisterHandler()`. That is, each time Acrobat restarts, the non-default security handler choice is lost. To fix the problem, change the plugin code to not call `PSUNregisterHandler()`.

### Table 17 Registry preferences: cPPKHandler

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| xDefEnrollmentURL | text | (v 7.0) Default: null  
Path: `<security root>`\cPubSec  
Maps to GUI item: Enroll at an online CA  
The destination URL when the user selects Enroll at an online CA while adding a new digital ID. unless a different URL is specified as seed-value on the signature field being signed). |
| cDigitalIDFiles   | cab  | (v 7.0) Default: null  
Path: `<security root>`\cPPKHandler  
Contains an array of subcabs for all application-known digital ID files. The format is as follows:  
cPath: The path of to the digital ID file.  
cCredentials: An array of certificates that have corresponding private keys in the file.  
cCertificates: An array of certificates that are in the file but do not have an associated private key (usually CA certs). Certificates are stored as binary data. |
| cACs              | cab  | (v 7.0) Default: null  
Path: `<security root>`\cPPKHandler  
Contains a set of attribute certificates as binary data. Each certificate is indexed with an integer 0 to N. The value is only set when a user imports attribute certificates. |
| tCredProvider     | text | (v 7.0) Default: null  
Path: `<security root>`\cPPKHandler\cCredSign  
Identifies credential service provider interface for the default signing digital ID. The value is set when a user opens the Security Settings Console and specifies a default signing ID. The value depends on the type of selected ID. For example, setting a self signed digital ID would result in a value of Adobe_FileCredentialProvider. See also xCertSHA1 |
| xCertSHA1         | text | (v 7.0) Default: null  
Path: `<security root>`\cPPKHandler\cCredSign  
Identifies the default signing digital ID by its SHA1 hash of the public key. The value is set when a user opens the Security Settings Console and specifies a default signing ID. See also tCredProvider |
5.1.3.6 Digital ID Import and Export Paths

The digital ID default path preferences point to the application security folder. For example, \Documents and Settings\<user name>\Application Data\Adobe\Acrobat\8.0\Security. The path is used when the user imports or exports an ID from the Security Settings Console. Since the application remembers the last accessed directory, if a user chooses a different directory, that action will overwrite the preconfigured value an administrator might provide.

The following options are available:

- Specifying a default path for exporting and importing digital ID certificates (does not include private keys).
- Specifying a default path for saving newly created digital ID files.

---

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tCredProvider</td>
<td>text</td>
<td>(v 7.0) Default: \Path: &lt;security root&gt;\cPPKHandler\cCredCrypt Identifies credential service provider interface for the ASPKI provider which exposes this digital ID. The value is set when a user opens the Security Settings Console and specifies a default signing ID. The value depends on the type of selected ID. For example, setting a self signed digital ID would result in a value of Adobe_FileCredentialProvider. See also xCertSHA1.</td>
</tr>
<tr>
<td>xCertSHA1</td>
<td>text</td>
<td>(v 7.0) Default: null \Path: &lt;security root&gt;\cPPKHandler\cCredCrypt Identifies the default encryption digital ID by its SHA1 hash of the public key. The value is set when a user opens the Security Settings Console and specifies a encryption signing ID. See also tCredProvider</td>
</tr>
<tr>
<td>bCustomPrefsCreated</td>
<td>bool</td>
<td>(v 7.0) Default: false \Path: &lt;security root&gt;\cPPKHandler Indicates whether a custom certificate specific preference (e.g. Identrus) has already been created and written to the registry. If true, it doesn’t get created again.</td>
</tr>
<tr>
<td>aDefDirectory</td>
<td>ASAtom</td>
<td>(v 7.0) Default: Adobe.PPKMS.ADSI.dir0 \Path: &lt;security root&gt;\cPPKHandler Maps to GUI item: Setting a default search directory affects the UI in two places: A star appears next to the default directory in the Security Settings Console and the directory is moved to the top of the directories drop down list in the Trusted Identities Manager’s Search for Recipients dialog. Default directory to use when searching for digital IDs. On Windows, the Adobe.PPKMS security handler provides access through the Microsoft Active Directory Script Interface (ADSI) to all the directories the user created in the Security Settings Console. These directories are named in the format of &lt;directory handler&gt; + &lt;index&gt;. For example, Adobe.PPKMS.ADSI.dir0, Adobe.PPKMS.ADSI.dir1, and so on. Unsupported for Linux and Macintosh.</td>
</tr>
</tbody>
</table>
5.1.4 Certificate Management

Signing and certificate security workflows require that users obtain and trust other people’s certificates. They will use those certificates to trust someone else’s signature and to encrypt documents for them. Administrators have several options for facilitating this process. For details, see:

- “Windows Integration” on page 87
- “Trusted Identity List Configuration” on page 88
- “Custom Certificate Preferences” on page 89

5.1.4.1 Windows Integration

While Acrobat has its own store, the Windows store may already contain needed certificates or your enterprise may simply be a Windows shop. Windows integration allows end users to search for and use certificates in the Windows Certificate Store.

End users can configure their application for Windows integration through the application’s Preference panel. Configuration options allow users to search the Windows store from the Trusted Identity Manager (through the Search button), set trust levels for any found certificate, and choose which certificates to use for encryption (once the certificate is located and added to the Trusted Identity Manager). If a user has a personal ID in the Windows store, it appears in the Security Settings Console automatically without any special configuration.

### Table 18 Registry preferences: Digital ID path

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cExportPath</td>
<td>path</td>
<td>(v 7.0) Default: The applications security folder. For example, C:\Documents and Settings&lt;username&gt;\Application Data\Adobe\Acrobat\Security Path: &lt;security root&gt;\cPubSec Default path for exporting credentials. Used by all security plugins.</td>
</tr>
<tr>
<td>cImportPath</td>
<td>path</td>
<td>(v 7.0) Default: The application security folder. For example, C:\Documents and Settings&lt;username&gt;\Application Data\Adobe\Acrobat\Security Path: &lt;security root&gt;\cPubSec Default path for importing credentials. Used by all security plugins.</td>
</tr>
<tr>
<td>cEmbeddedFilePath</td>
<td>path</td>
<td>(v 7.0) Default: The application security folder. For example, C:\Documents and Settings&lt;username&gt;\Application Data\Adobe\Acrobat\Security Path: &lt;security root&gt;\cPubSec Default path for embedded files. The path last chosen for extracting an embedded file from a WebBuy FDF. The first time an embedded file is extracted from an FDF the user is asked where to save it.</td>
</tr>
<tr>
<td>cProfilePath</td>
<td>path</td>
<td>(v 7.0) Default: The applications security folder. For example, C:\Documents and Settings&lt;username&gt;\Application Data\Adobe\Acrobat\Security Path: &lt;security root&gt;\cPubSec Default path for storing profile files such as PKCS#12 files. This is used both when creating new digital ID files and when browsing for existing files. Used by all security plugins.</td>
</tr>
</tbody>
</table>
Administrators can control whether clients can access MSCAPI through Acrobat so that users can find, use, and set trust levels for Windows certificates. The following options are available:

- Adding the Windows Certificate Store as a searchable repository with \texttt{bCertStoreImportEnable}.
- Setting separate trust levels for approval and certification signatures.
- Preventing end user modification of certificate trust levels.
- Tuning the service provider interface:
  - “Revocation Checker Providers” on page 95
  - “Signature Validation Directory Providers” on page 96

### Table 19 Registry preferences: MSCAPI provider

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{iMSStoreTrusted}</td>
<td>\texttt{int}</td>
<td>(v. 7.0) Default: 0 Path: &lt;security root&gt;\cASPKI\cMSCAPI\DirectoryProvider \texttt{Validating Signatures} and \texttt{Validating Certified Documents}. Controls whether or not certificates in the Windows Certificate Store are trusted for signing and certifying. 00: None. 60: \texttt{Validating Signatures} 62: \texttt{Validating Signatures} and \texttt{Validating Certified Documents}.</td>
</tr>
<tr>
<td>\texttt{bCertStoreImportEnable}</td>
<td>\texttt{bool}</td>
<td>(v 7.0) Default: 0 Path: &lt;security root&gt;\PPKHandler \texttt{Enable searching the Windows Certificate Store for certificates other than yours} If true, then users can import from MSCAPI certificate stores into their Trusted Identity Manager.</td>
</tr>
</tbody>
</table>

#### 5.1.4.2 Trusted Identity List Configuration

The trusted identity list contains all of a user’s imported certificates that they use for validating someone else’s signature or encrypting a document for them. The list is maintained and managed via the Trusted Identity Manager; however, administrators can preconfigure applications to use non-default list files, add certificates from the Windows, store, and so on.

The following options are available:

- Creating a custom filename/file for the trusted identity list.
- Specifying a non-default security handler to control Trusted Identity Manager functions. For details, see \texttt{aAddressBook} in Table 23.
- Adding the Windows Certificate Store as a searchable repository with \texttt{bCertStoreImportEnable}. For details, see Table 19.
- Turning off and on the ability to automatically download certificates sent by Adobe to users over the internet via \texttt{bLoadSettingsFromURL}. 

### 5.1.4.3 Custom Certificate Preferences

The `cCustomCertPrefs` directory provides the means to use certificate-specific settings to modify application behavior when it encounters a particular certificate. As the application builds a certificate chain, it compares the information it finds in the certificate with that in the registry to see if there is a match. If there is a match, the custom settings are used to override the application's default behavior. Certificates that chain up to a CA that match those configured here or that contain recognized extensions will use the preferences set in this directory.

You can use any of the preferences available in `\cASPKI` in your customized preference under `\cASPKI\cCustomCertPrefs`. Custom certificate preferences are specified differently than when they are used globally under `\ASPKI`. For example, The naming and path convention is always `c<key>:c<index>:<type>Value`, where the global preference would be `sSignCertOID` and the custom preference would be `cSignCertOID` (the data type is associated with the `Value` subkey rather than the key).

Certificates are identified by creating a hash of a unique identifier and appending it to `c` such as `c312E322E3834302E3131343032312E310000`. When the application finds in a certificate chain a hash that matches the hash in the registry, the custom preference is used. Locate custom preferences under one of the following methods:

- Base 16 encoded SHA1 public key hash of the CA certificate. A SHA1 hash of the public key is used instead of SHA1 hash of the certificate so that the preferences can survive cross certification where the SHA1 hash of the public key remains the same.
- Hex representation of an OID followed by two NULL characters.

### Identifying Certificates with a Hash of the Public Key

Adobe's Certificate Viewer provides an easy way to get the public key hash.

To do so:

1. Import the certificate into Acrobat.
2. Choose **Advanced > Manage Trusted Identities.**
3. Choose **Certificates** from the Display drop down list.
4. Highlight the certificate you will use (an ICA).

5. Choose **Show Certificate**.

6. Choose the Details tab.

   **Figure 42 Certificate Viewer details tab**

   ![Certificate Viewer details tab](image)

7. Highlight **SHA1 digest of public key**.

8. Copy the hash in the lower panel to the clipboard. This example uses `5D800FA2F49D4816FCA014BB9442665922BA8A77`.

9. Open the registry

10. Navigate to `HKEY_CURRENT_USER\Software\Adobe\<application>\<version>\Security\cASPKI\cASPKI\cCustomCertPrefs\`.

11. Right click on `cCustomCertPrefs`.

12. Choose **New > Key**.

13. Enter "c" followed by the public key hash you just created. For example: `c5D800FA2F49D4816FCA014BB9442665922BA8A77`.

   This registry key is now ready to be populated with custom preferences and should look like **Figure 43**.
Adding Certificate Preferences

On a Windows machine, certificate-specific preferences can be added by following the steps below:

1. Navigate to `HKEY_CURRENT_USER\Software\Adobe\<application name>\<version number>\Security\cASPKI\cASPKI\cCustomCertPrefs\<your ID hash>.

2. Add the needed containers and keys. Custom entries under cCustomCertPrefs are always a cab and the name is prepended with a “c.” For example, to set a timestamp server provider preferences, you would use Table 49. While the key list shows `sPassword` as a valid name, when it is used under `CustomCertPrefs`, the entry should be renamed to `cPassword`.

Requirements will vary based on your specific need:

- cAdobe_LTVProvider
- cAdobe_TSPProvider
- cAdobe_OCSPRevChecker
- cAdobe_CRLRevChecker
- cAdobe_ChainBuilder

Associating a Certificate Preference with a Chain Scope

`iStart` and `iEnd` can be used to specify for what parts of a certificate chain a custom certificate preference will apply. They are always used at the container level of c0, c1, c2, and so on. For example, Acrobat could be configured to search for acceptable policy OIDS only in the certificates that are the first, second, and third levels below the root CA (Figure 44).

Table 21 Registry preferences: Scoping certificate

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iStart</td>
<td>int</td>
<td>(v 7.0) Determines the start of the preference relevance depth relative to the certificate chain. By default, the preference starts at the current level.</td>
</tr>
<tr>
<td>iEnd</td>
<td>int</td>
<td>(v 7.0) Determines the end of the preference relevance depth relative to the certificate chain. By default the depth of the preference is MaxUns32</td>
</tr>
</tbody>
</table>
To specify a scope within a chain:

1. **Navigate to** `<root>\cCustomCertPrefs\<certificate public key hash>\cAdobe_ChainBuilder\cAcceptablePolicyOIDs\c0`.
2. **Highlight c0**, right click, and choose **New > DWORD**.
3. Enter the field names **iStart** and **iEnd**.
4. **Right click on a field** and choose **Modify**.
5. **Set the Value Data** field to specify the needed start or end range.
6. **Choose OK**.
7. **Restart the application**.

**A Custom Certificate Preference Example for Identrus Compliance**

Acrobat has two custom certificate preferences that enable Identrus compliance at `<security root>\cASPKI\cASPKI\cCustomCertPrefs\<hash of Identrus OID>`. You can use this as a template for a similar custom certificate preference:

- `c312E322E3834302E3131343032312E310000`
Whenever a credential that chains up to an Identrus CA is used for signing or signature validation, the application follows the Identrus rule set rather than the default rule set. The custom certificate preferences enable Acrobat to recognize Identrus certificates and process them as required by Identrus.

When Acrobat starts for the first time, it checks whether bCustomPrefsCreated is set. If not set to true, Acrobat writes out the Identrus rule set that is hard coded within Acrobat. If it’s already set, then writing out the Identrus rules is skipped.

**Note:** If the Identrus rules in Acrobat need to be changed without updating Acrobat, then create a custom installer (tuned with the wizard) that sets bCustomPrefsCreated to true and writes out the new Identrus rules. Acrobat will then first check whether custom preferences have been created. If they are already created, then Acrobat won’t write out the Identrus rules within Acrobat, and those written out by the custom installer will be respected by ASPKI.

An Identrus CA is identified by the certificate policy OID 1.2.840.114021.1.1 present in the production Identrus root CA. However, the certificate policy OID present in their test root CA is different, and in order to be able to test Acrobat against the test Identrus environments, Acrobat also uses the same Identrus rules for CAs that have the certificate policy OID 1.2.840.114021.1. Identrus certificates must contain one of the OIDs listed in AcceptablePolicyOIDs in Table 22. If the OID is not present, the certificate is deemed to be invalid and the signature will also be invalid.

### Table 22 Registry preferences: Identrus

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cAcceptablePolicyOIDs\c1</td>
<td>cab of strings</td>
<td>(v. 7.0) Default: &quot;iEnd&quot;=dword:00000002 and &quot;iStart&quot;=dword:00000002 The default chain scope in which to look for the policy OIDs.</td>
</tr>
<tr>
<td>cAcceptablePolicyOIDs\ c1\cValue</td>
<td>cab of strings</td>
<td>(v. 7.0) Default: The values described below. An array of strings containing the policy OIDs for a certificate to be considered acceptable.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>For ICA certificates:</strong> Set to 1.2.840.114021.1.6.1 and 1.2.840.114021.1.2.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>For EE certificates:</strong> Set to 1.2.840.114021.1.4.1, 1.2.840.114021.1.4.2, 1.2.840.114021.1.7.2, 1.2.840.114021.1.10.1, 1.2.840.114021.1.10.2, 1.2.840.114021.1.13.2, 1.2.840.114021.1.16.2, 1.2.840.114021.1.19.2, 1.2.840.114021.1.22.2, 1.2.840.114021.1.25.2, 1.2.840.114021.1.28.2, 1.2.840.114021.1.30.2</td>
</tr>
</tbody>
</table>

### 5.1.5 Custom Security Handler Preferences

Adobe.PPKLite is the default security handler used for performing private key functions, validating signatures, and signing and encrypting documents. This is represented in the user interface as Adobe Default Security in the Digital Signatures Advanced Preferences dialog on both the Verification and Creation tabs. Administrators can install custom handlers to perform these functions, in which case the drop down lists on these tabs will list the additional handlers. All entries in the cHandler folder are reset by the Digital Signature Preferences dialog’s Reset button.

Security handlers are Acrobat plugins. Information about creating plugins in general and security handlers in particular can be found in the Acrobat Software Development Kit (SDK) and its HFTs, header
files, and other API documentation. Because Acrobat’s Adobe.PPKLite is becoming more feature rich with each release, it is unlikely that you will need a custom security handler.

If a custom handler is used, you can specify the following:

- Separate handlers for signing/encryption and signature validation.
- The default method displayed in the drop-down list of handlers.
- Lock down the selections so they cannot be modified by end users.

### Table 23 Registry preferences: Security handler

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| aPrivKey      | ASAtom     | (v 7.0) Default: Adobe.PPKLite  

Path: <security root>\cHandlers  
Maps to GUI item: **Method to use When Signing and Encrypting Documents** on the Creation tab of the Digital Signature Advanced Preferences dialog.  
Used by DigSig and PubSec to store the handler that accesses private key functions. It is used for signing, decryption, and responding to an FDF file request to export contact information. The value should be set to Adobe.NoHandler if it is desired that the user be asked to select a handler.  
Subject to lockdown as described in “Preventing End-User Modification” on page 75.

| aVerify       | ASAtom     | (v 7.0) Default: Adobe.NoHandler  

Path: <security root>\cHandlers  
Maps to GUI items: The selected radio button determines the value:  
**Use the document-specified method, prompt if it is not available:** Adobe.NoHandler  
**Use the document-specified method, use the default method if it is not available:** Adobe.PPKLite  
**Always use the default method (overrides the document-specified method)**. Takes the value selected from Default Method for Verifying Signatures.  
Remembers the name of the preferred handler to use when verifying signatures. If this value is not set, then the handler used to verify signatures is the handler that matches the Filter attribute in the signature dictionary; if this handler is not available, then the user is prompted to select a handler. If this value is set then, its meaning is qualified by the value of aVerifyUseAlways.  
Subject to lockdown as described in “Preventing End-User Modification” on page 75.

| aVerifyUseAlways | ASAtom     | (v 7.0) Default: 0  

Path: <security root>\cHandlers  
Maps to GUI item: **Always use the default method (overrides the document-specified method)**  
Qualifies the use of aVerify. If true and aVerify is set to a handler name, then this handler is used to verify all signatures. If false, then the aVerify handler is used only to verify signatures when the handler specified by the signature dictionary Filter attribute is not present.  
Subject to lockdown as described in “Preventing End-User Modification” on page 75.
5.1.6 ASPKI Service Provider Interfaces

A service provider interface (SPI) is part of an architectural model that provides a programming interface for developing replaceable components and common services access. As a standalone PKI toolkit written in C++, ASPKI has no dependencies on the PDF Library or Acrobat and is designed to be portable and usable in different applications, including but not limited to Acrobat and GUI-less servers. It provides a number of SPIs which may be configured independently of each other.

The SPI preferences in `<security root>`\cASPKI\cSPIs are not exposed to end users and can only be set by an administrator that needs to fine tune the application’s certificate handling. For example, a company may not want to use Adobe or MSCAPI as a credential provider, might need to control revocation checking in a custom way, or might wish to prevent the use of PKCS#11 devices. In most deployments, the default behavior should be sufficient and you should not need to modify the settings. However, customization is possible as follows:

- “Revocation Checker Providers” on page 95
- “Signature Validation Directory Providers” on page 96
- “Roaming ID Authentication Mechanisms” on page 97

### 5.1.6.1 Revocation Checker Providers

The revocation checker provider provides revocation checking services. You can specify one or more revocation checking methods and choose whether to use the default methods or some MSCAPI-specific method.

The following options are available:

- Use one or both of Adobe’s revocation checking methods (CRL and OCSP).

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| aAddressBook  | ASAtom   | (v 7.0) Default: null  
Path: `<security root>`\cHandlers
Remembers a preferred handler for accessing Trusted Identity Manager functions including certificate data import from an FDF file. |
| aDirectory    | ASAtom   | (v 7.0) Default: null  
Path: `<security root>`\cHandlers
Remembers a preferred handler for directory functions (e.g. LDAP), including for importing directory information from an FDF data exchange file. |
| cDialogs: xSelHandler | 4 int keys | (v 7.0) Default: null  
Path: `<security root>`\cDigSig
The last on-screen coordinates of a handler’s digital ID selection dialog. It is a subkey containing 4 keys: Top, Bottom, Left, and Right.  
This preference could be used by 3rd party handlers or by someone invoking a non-signing digital ID selection dialog via JavaScript. |
- Use of the MSCAPI revocation checking plugin model as an alternative to Adobe mechanisms. For example, administrators may have standardized on MSCAPI or might prefer the MSCAPI method of using a CRL registry cache (Acrobat has its own cache).

**Note:** Acrobat’s default CRL cache location is C:\Documents and Settings\<user>\Application Data\Adobe\<application>\<version>\Security\CRLCache

### Table 24 Registry preferences: SPI for revocation checker providers

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iRevocationChecker</td>
<td>int</td>
<td>(v 7.0) Default: 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Path:</strong> <code>&lt;security root&gt;\cASPKI\cSPIs</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0: Use none of the registered providers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Use first registered provider.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: Use all registered providers.</td>
</tr>
<tr>
<td>cRevocationChecker</td>
<td>cab</td>
<td>(v 7.0) Default: Adobe_OCSPRevChecker, Adobe_CRLRevChecker</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Path:</strong> <code>&lt;security root&gt;\cASPKI\cSPIs</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>An array of text entries (t0-tn) containing the name of a registered provider:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Adobe_OCSPRevChecker:</strong> Adobe’s default OCSP method.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Adobe_CRLRevChecker:</strong> Adobe’s default CRL method.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>MSCAPI_RevocationChecker:</strong> Accesses MSCAPI revocation checking plugin framework.</td>
</tr>
</tbody>
</table>

### 5.1.6.2 Signature Validation Directory Providers

The directory provider SPI provides access to trust anchors and intermediate CAs used for signature validation. By default, certificates in all of the supported locations are used.

The following options are available:

- Preventing or allowing access to certificates in P12 files. End users must also be logged in to the file.
- Preventing or allowing access to certificates in the Trusted Identity Manager (AAB).
- Preventing or allowing access to certificates in the Window Certificate Store.
- Preventing or allowing access to self-signed certificates created by an Adobe application.

### Table 25 Registry preferences: SPI for directory providers

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iDirectoryProvider</td>
<td>int</td>
<td>(v 7.0) Default: 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Path:</strong> <code>&lt;security root&gt;\cASPKI\cSPIs</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0: Use none of the registered providers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Use first registered provider.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: Use all registered providers.</td>
</tr>
</tbody>
</table>
### 5.1.6.3 Roaming ID Authentication Mechanisms

The authentication mechanism provider pertains only to roaming IDs. It enables you to specify one or more authentication mechanisms. The mechanism must be supported by the roaming ID server with which the application communicates. The following features are available:

- Enabling multiple authentication mechanisms.
- Limiting the authentication mechanism to one specified type.
- Turning off authentication so that roaming IDs cannot be used.

### Table 25 Registry preferences: SPI for directory providers

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cDirectoryProvider</td>
<td>cab</td>
<td>(v 7.0) Default: All&lt;br&gt;Path: <code>&lt;security root&gt;</code>\cASPKI\cSPIs&lt;br&gt;An array of text entries (t0-tn) containing the name of a registered provider:&lt;br&gt;Adobe_FileCredentialDirectoryProvider: Provides access to PKCS#12 files.&lt;br&gt;AAB_DirectoryProvider: Provides access to the Trusted Identity Manager.&lt;br&gt;MSCAPI_DirectoryProvider: Provides access to the Windows Certificate Store.&lt;br&gt;Adobe_SelfSignedCredDirectoryProvider: Provides access to self signed certificates created by Acrobat.</td>
</tr>
</tbody>
</table>

### Table 26 Registry preferences: SPI for authentication mechanisms

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iAuthMechanisms</td>
<td>int</td>
<td>(v 8.0) Default: 2&lt;br&gt;Path: <code>&lt;security root&gt;</code>\cASPKI\cSPIs&lt;br&gt;0: Use none of the registered providers.&lt;br&gt;1: Use first registered provider.&lt;br&gt;2: Use all registered providers.</td>
</tr>
<tr>
<td>cAuthMechanisms</td>
<td>cab</td>
<td>(v 8.0) Default: The array contains all four values.&lt;br&gt;Path: <code>&lt;security root&gt;</code>\cASPKI\cSPIs&lt;br&gt;An array of text entries (t0-tn) where each entry contains the name of a registered provider:&lt;br&gt;PLAIN: A mechanism defined in RFC2595 consisting of a single message specifying the user's ID and password.&lt;br&gt;ASSP-Kerberos: A mechanism commonly used on Windows that passes a Single Sign On token and receives back a SAML assertion.&lt;br&gt;ASSP-ArcotID: A mechanism recognized by Arcot roaming ID servers.&lt;br&gt;ASSP-QnA: A mechanism that initiates a question-answer dialog between the user and server.</td>
</tr>
</tbody>
</table>

### 5.1.7 FDF Import and Export

The File Data Exchange Format (FDF) provides a format for easily importing and exporting certificate data and application settings. These settings appear in `<security root>`\cPubSec after a client uses the feature.
The default values are stored internally by the application and are not visible in the registry. An administrator can set the default behavior, but your configuration is subject to modification by end users via the user interface.

The following features are available:

- Specifying whether the default export behavior is to save or email the file.
- Specifying whether the default export behavior is to sign the file.
- Specifying whether the default certificate request behavior is to save or email the file.
- Enabling or disabling WebBuy FDF processing (deprecated).

**Example 5.6: FDF preferences**

```
[HKEY_CURRENT_USER\Software\Adobe\Adobe Acrobat\8.0\Security\cPubSec]
"bFDFRequestExcludeCert"=dword:00000000
"bFDFRequestSave"=dword:00000000
```

**Table 27 Registry preferences: FDF import and export**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bFDFExportSave      | bool   | (v 7.0) Default: 1
  **Path:** `<security root>\cPubSec`
  Maps to GUI item: **Save as** and **Email** radio buttons in export dialog.
  Persists whether user chose to save (1) or email (0) the FDF during export. |
| bFDFExportSign      | bool   | (v 7.0) Default: 0
  **Path:** `<security root>\cPubSec`
  Persists whether the user chose to sign the FDF during export. |
| bFDFRequestExcludeCert | bool   | (v 7.0) Default: 0
  **Path:** `<security root>\cPubSec`
  Similar to the **bFDFRequestSave**. False includes the user's certificate in all certificate requests. True excludes it. |
| bFDFRequestSave     | bool   | (v 7.0) Default: 0
  **Path:** `<security root>\cPubSec`
  When building a request for someone’s certificate, the user is asked whether they want to save the request as an FDF or email it directly. This flag is the cached answer to that question. |
| bWebBuyFDF          | bool   | (v 7.0) Default: 1
  Deprecated.
  **Path:** `<security root>\cPubSec`
  Enables WebBuy FDF file processing. |

### 5.1.8 Security Settings Console

Security Settings Console preferences persist information about the state of the console user interface. These preferences are user generated and implementation specific and are likely to change across application versions.
5.1.9 Displaying All Chains in the Certificate Viewer

By default, the Certificate Viewer builds and displays the trusted chain from the EE to the trust anchor. However, it is possible to show all found chains whether they are trusted or not. While most users do not need this information, it can be used for troubleshooting and verification. End users can turn this option on and off by using the Certificate Viewer’s checkbox **Show all certification paths found**. The following option is available:

- Showing all certification paths by default.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bShowAllChains   | bool   | (v. 7.0) Default: 0  
Path: <security root>\cPPKHandler  
If true, the Certificate Viewer shows all the chains; otherwise, it shows only the trusted chain. If there are no trusted chains, then all the chains are shown and this preference is ignored. |

5.1.10 Configuring Password Caching

By default, password caching is turned on so that users will not always have to enter a password when one is required. This feature affects Adobe Policy Server log in, signing with digital IDs in the Acrobat store (pfx or p12 files), changing password timeout policies, and creating new password security policies. For example, setting the option to false disables the menu option **Save password with the policy** when creating a new policy.

The following options are available:
Turning on FIPS Mode

This is a Windows only feature.

Acrobat and Reader can provide encryption via the Federal Information Processing Standard (FIPS) 140-2 mode. FIPS 140 is a cryptographic security standard used by the federal government and others requiring higher degrees of security. Through registry configuration it is possible to force Acrobat to use only FIPS 140-certified cryptographic libraries. Doing so only affects the production and not the consumption of PDF files, and it only affects encryption and digital signature workflows.

When the FIPS mode is on, encryption uses FIPS-approved algorithms provided by the RSA BSAFE as follows:

- v. 9.x and earlier: Crypto-C ME 2.1 encryption module with FIPS 140-2 validation certificate 608.
- v. 10: Crypto-C ME 3.0.0.1 encryption module with FIPS 140-2 validation certificate 1092
FIPS mode changes Acrobat’s default behavior as follows:

- FIPS-compliant algorithms are always used.
- Self-signed certificate creation is disabled. In FIPS mode, users cannot create self-signed certificates.
- Signing with non-FIPS supported algorithms results in an error message; that is, signing fails if the document hash algorithm (digest method) is set to MD5 or RIPEMD160.
- Password security is turned off. Users can apply certificate or Adobe LiveCycle Right Management Server security using the AES encryption algorithm to a document, but password encryption is disabled.
- When applying certificate security, the RC4 encryption algorithm is not allowed.
- Documents protected with non-FIPS compliant algorithms cannot be saved.

### Table 31 Registry preferences: FIPS mode

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFIPSMode</td>
<td>bool</td>
<td>(v 8.1) Default: 0 Path: HKEY_CURRENT_USER\Software\Adobe\Adobe Acrobat&lt;version&gt;\AVGeneral</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Turns FIPS mode on and off.</td>
</tr>
</tbody>
</table>

### 5.2 Content Security

Content security settings control certificate security, security policies, Adobe LiveCycle Right Management Server connections, and other features which affect application behavior in security-related workflows. For details, see the following:

- “Configuring Certificate Handling for Certificate Security” on page 101
- “Setting up Security Policy Favorites” on page 102
- “Configuring LiveCycle Right Management Servers” on page 102
- “Security Envelopes” on page 104
- “Configuring Password Caching” on page 99

#### 5.2.1 Configuring Certificate Handling for Certificate Security

Some application behavior for handling certificates in certificate security workflows is configurable. The following options are available:

- Configuring digital IDs used for decryption. For details, see “Setting Digital ID Defaults” on page 84.
- Configuring the certificate handling for encryption. For details, refer to “Trusted Identity List Configuration” on page 88 and “Windows Integration” on page 87.

### 5.2.2 Requiring FIPS-Compliant Encryption Algorithms

See “Turning on FIPS Mode” on page 100.
5.2.3 Setting up Security Policy Favorites

The keys at \Security\cPPKLite\cSP_Favorites contain an array of subkeys c0-cN where each index defines a favorite security policy. Both user and organizational policies can be favorites. Any policy marked as a favorite will appear in the user’s favorite’s list. End users make a policy a favorite by opening the Manage Security Policies dialog, highlighting the policy, and choosing Favorites. A star icon appears to the left of the policy name and the policy becomes available in the top level menu.

The following options are available:

- Specifying an non-default handler for a policy.
- Marking one or more policies as a favorite.
- Specifying policy names.

Table 32  Policy favorites preferences

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aHandler</td>
<td>ASAtom</td>
<td>(v 7.0) Default: Null&lt;br&gt;Path: &lt;security root&gt;\PPKLite\cSP_Favorites\c&lt;index&gt;&lt;br&gt;An ASAtom specifying which PDCrypt handler knows how to handle this security policy.</td>
</tr>
<tr>
<td>blsFavorite</td>
<td>bool</td>
<td>(v 7.0) Default: Null&lt;br&gt;Path: &lt;security root&gt;\PPKLite\cSP_Favorites\c&lt;index&gt;&lt;br&gt;Determines whether the referenced security policy is displayed as a favorite.</td>
</tr>
<tr>
<td>sls</td>
<td>string</td>
<td>(v 7.0) Default: Null&lt;br&gt;Path: &lt;security root&gt;\PPKLite\cSP_Favorites\c&lt;index&gt;&lt;br&gt;A string containing the security-policy.acrodata file key used to reference the policy that is being applied.</td>
</tr>
<tr>
<td>tName</td>
<td>text</td>
<td>(v 7.0) Default: Null&lt;br&gt;Path: &lt;security root&gt;\PPKLite\cSP_Favorites\c&lt;index&gt;&lt;br&gt;The security policy name.</td>
</tr>
</tbody>
</table>

5.2.4 Configuring LiveCycle Right Management Servers

The preferences in EDC (a legacy name) define Adobe LiveCycle Right Management Server connections. Users can specify servers through the Security Settings Console. However, administrators can preconfigure user machines to control the end user experience.

The following options are available:

- Setting a default server under cEDC (Table 33). The default server appears with a star icon in the Security Settings Console.
- Controlling whether to use HTTP or HTTPS with bAllowConnectViaHTTP (Table 33).
- Adding one or more servers in to the known server list cEDC\KnownServers (Table 34). These server definitions will appear in the Security Settings Console’s server list.
- Locking down the settings so that the server configuration dialog will not appear in the user interface, thereby preventing end users from adding servers or changing server settings. For details, see “Preventing End-User Modification” on page 75.
### Table 33: Registry preferences: LiveCycle Right Management Server

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAllowConnectViaHTTP</td>
<td>bool</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC</code>&lt;br&gt;If true, the server connection URI uses the format <a href="http://server:port/path">http://server:port/path</a>. If false, it uses the format <a href="https://server:port/path">https://server:port/path</a>. In either case, if the connection fails, an error message appears.&lt;br&gt;For more information, see <a href="#">Guidelines for Developing CSPs for Acrobat on Windows</a>.</td>
</tr>
<tr>
<td>tLastServerURL</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC</code>&lt;br&gt;The last APS server used to open a document and the server used for offline key synchronization. It is set automatically when opening a document.</td>
</tr>
<tr>
<td>tServerName</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC</code>&lt;br&gt;The Adobe Policy Server selected by the user as the default. This is set via the user interface in the Security Settings Console either by creating a new server (the first one is the default) or by selecting an existing one.</td>
</tr>
<tr>
<td>tServerURL</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC</code>&lt;br&gt;The default server’s URL.</td>
</tr>
</tbody>
</table>

### Table 34: Registry preferences: LiveCycle Right Management Server list

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bSavePassword</td>
<td>bool</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC\KnownServers</code>&lt;br&gt;Maps to GUI item: ALRMS settings in the Security Settings Console&lt;br&gt;Indicates whether the password has been cached for this server.</td>
</tr>
<tr>
<td>tName</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC\KnownServers</code>&lt;br&gt;Maps to GUI item: ALRMS settings in the Security Settings Console&lt;br&gt;The user defined name for this server.</td>
</tr>
<tr>
<td>tServer</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC\KnownServers</code>&lt;br&gt;Maps to GUI item: ALRMS settings in the Security Settings Console&lt;br&gt;The DNS server name (i.e. alrms.adobe.com). There is no scheme specified.</td>
</tr>
<tr>
<td>tLockboxId</td>
<td>text</td>
<td><strong>Path</strong>: <code>&lt;security root&gt;\EDC\KnownServers</code>&lt;br&gt;Set if <code>bSavePassword</code> is not 0 to look up the password in a user’s secure password cache. This is not directly exposed via the user interface.</td>
</tr>
</tbody>
</table>
5.2.5 Security Envelopes

These keys appear after a user creates a security envelope to deliver one or more documents securely. The keys in cMain remember the user choices such as the last search path for finding attachments and so on. An administrator could set a default value, but these values would be changed by user actions.

Table 35 Registry preferences: Security envelope

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tLastSearchPath</td>
<td>ASText</td>
<td>(v 7.0) Default: null Path: &lt;security root&gt;\cMain The last path used for selecting files to include in an eEnvelope.</td>
</tr>
<tr>
<td>tLastEnvelope</td>
<td>ASPATH</td>
<td>(v 7.0) Default: null Path: &lt;security root&gt;\cMain The ASPATH to the last envelope used for Secure PDF Delivery.</td>
</tr>
</tbody>
</table>

5.3 Digital Signatures

Signing is a complex process that could involve multiple signatures on a changing document. Moreover, support for multiple types of IDs, revocation checking methods, and so on mean that Acrobat is as feature rich as it is heavy with customizable preferences. For this reason, a basic understanding of the signing and signature validation process is prerequisite to customizing application behavior. Once you are ready to configure the application, refer to the following sections for details:

- “Revocation Checking Preferences” on page 104
- “Signature Creation Preferences” on page 111
- “Signing Workflow Preferences” on page 118
- “Signature Validation Preferences” on page 124
- “Configuring Password Caching” on page 99

**Note:** Signing and signature validation behavior is also subject to control via general preferences such as the digital ID and service provider interface preferences described in “Setting Basic Client and Workflow Preferences” on page 75.

5.3.1 Revocation Checking Preferences

Since revocation checking can occur both during signature creation and signature validation, revocation settings may affect both the user’s ability to sign and to validate signatures. A check can occur for the signing certificate as well as for the certificates associated with any revocation check responses. OCSP and CRL checking are both supported, and MSCAPI checks will use one or the other or both.
Interaction between OCSP and CRL Preferences

If a signature has the following chain CA | ICA | EE and OCSP revocation checking preferences are specified for the CA but CRL preferences are not, then Adobe_OCSPRevChecker and Adobe_CRLChecker behave as follows:

- While doing OCSP revocation checking for the ICA, the preferences specified for the CA are used. If the scope for the preferences is specified as infinite, then the CA preferences are also used for revocation checking the EE.
- While doing CRL revocation checking, if no preferences have been specified either for the CA or the ICA, then the preferences present at cASPKI:cAdobe_CRLRevChecker are used instead.

Tip: For more detail about how revocation checking affects signing and signature validation, see Chapter 7, “Certificate Processing”.

The following options are available:

- “Configuring OCSP Revocation Checking” on page 105
- “Configuring CRL Revocation Checking” on page 108
- “Certificate Chain Building” on page 110

5.3.1.1 Configuring OCSP Revocation Checking

OCSP revocation checking can occur both during signature creation and signature validation on both the signing certificate as well as for the certificates associated with any revocation check responses. The following options are available:

- “Specifying the Time and Method of OCSP Checks” on page 106
- “Specifying Certificates for Valid OCSP Responses” on page 107
- “Specifying Certificates for Valid OCSP Requests” on page 108

Tip: For more detail about how revocation checking affects signing and signature validation, see Chapter 7, “Certificate Processing”.

OCSP responders: Determining if they are authorized to do rev checks

RFC2560 defines three methods of determining whether the OCSP responder is authorized to perform OCSP revocation checking. Two methods are strictly defined and the third one is called "local configuration" which Acrobat defines by specifying a set of certificates. If OCSP response is signed with one of these certificates then the responder is considered authorized.

Rule 1 (Acrobat 8.0): defined the local configuration rule as follows by authorizing OCSP responses that come from responders specified by sURL.

Rule 2(Acrobat 9.0): If a custom certificate preference has a new "AuthorizedResponder" boolean entry with a value of true, and the certificate being checked for OCSP revocation as well as the OCSP response both chain up to the customer certificate, then the responder is authorized.

The order in which verifying OCSP responders occurs is as follows:

1. Local configuration rule #1 (A8 and A9).
2. Local configuration rule #2 (new in A9).
3. The two deterministic methods from RFC2560.

**Note:** The structure and location of the new AuthorizedResponder entry is the same as for SendNonce entry. However, while SendNonce may be specified under ASPKI or a custom certificate preference, AuthorizedResponder may be specified only under custom certificate preferences.

### Specifying the Time and Method of OCSP Checks

OCSP revocation checking preferences allow you to control when and how an OCSP check occurs. The following options are available:

- Specifying when to do revocation checking as well as the effect of a failed or bad response.
- Specifying when and where to go online to get a response.
- Specifying whether to include a nonce. Nonces are random generated numbers that are sent with a request and matched by a response. They improve security by assuring communication with an active, non-spoofed server.
- Using or ignoring a response's `thisUpdate` and `nextUpdate` times to control its validity. See RFC 2560 for details.
- Setting a limit on the amount of time difference between the local time and response's publish time.

#### Table 36 Registry preferences: OCSP method

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| iReqRevCheck    | int    | (v 7.0) Default: 2  
  **Path:** `<security root>\cASPKI\cAdobe_OCSPRevChecker`  
  Indicates whether revocation checks are required to succeed on the OCSP response. Interacts with other iReqRevCheck settings as described in “Interaction of iReqRevCheck Settings” on page 158.  
  0: Don’t do revocation checks.  
  1: Do a check IF certificate has AIA extension or responder info is in registry; don’t fail if the check fails.  
  2: Do a check IF certificate has AIA extension or responder info is in registry; all checks must succeed if there is data and a check occurs.  
  3: Require a check; it must succeed under all circumstances. |
| iURLToConsult   | int    | (v 7.0) Default: 0  
  **Path:** `<security root>\cASPKI\cAdobe_OCSPRevChecker`  
  Specifies how the revocation checker chooses which responder to use:  
  0: Use the AIA extension in the certificate.  
  1: Use the URL key in `sURL`.  
  2: Use the AIA extension in the certificate. If it is not present, use the URL key in `sURL`.  
  3: Use the OCSP request signer’s certificate AIA extension. Relevant only if `SignRequest` is 0. |
Specifying Certificates for Valid OCSP Responses

It is possible to require certain features for certificates used to sign OCSP responses. If a response does not meet the specified parameters, it is considered invalid and the signature status may be Unknown or Invalid. The following options are available:

- Allowing or disallowing the OCSPNoCheck extension.
- Requiring the presence of a public key hash extension (bRequireOCSPCertHash).

### Table 36 Registry preferences: OCSP method

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sURL                  | string      | (v 7.0) Default: null  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
The URL used to fetch OCSP responses. |
| bGoOnline             | bool        | (v 7.0) Default: 1  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
Specifies whether to go online to do revocation checking. Never used for Reader enabled signatures (UR3). |
| bSendNonce            | bool        | (v 7.0) Default: 1  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
If true, nonces are included in the OCSP request and expected to be present in the response and should match the request's nonce. If false, nonces are not sent. |
| iResponseFreshness    | int         | (v 7.0) Default: 525600 (1 year)  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
Specifies the amount of time in minutes after the response’s published thisUpdate time for which the response will be valid. After that time, the response will be invalid. |
| bIgnoreValidityDates  | bool        | (v 7.0) Default: 0  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
Specifies whether to ignore the response’s thisUpdate and nextUpdate times, thereby preventing any negative affect of these times on response validity.  
The value is set to true for ubiquity signatures created by enabling usage rights for Adobe Reader. |
| iMaxClockSkew         | int         | (v 7.0) Default: 5 minutes  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
The number of minutes the local machine time can vary from the response’s published time to account for a network delay, time synchronization issues, and so on. |

### Table 37 Registry preferences: OCSP certificate

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bAllowOCSPNoCheck     | bool        | (v 8.0) Default: 1  
Path: \<security root>\cASPKI\cAdobe_OCSPRevChecker  
Specifies whether the OCSPNoCheck extension is allowed in the response signing certificate. |
Specifying Certificates for Valid OCSP Requests

It is possible to require certain features for certificates used to sign OCSP requests. If a request does not meet the specified parameters, it is considered invalid and the signature status may be Unknown or Invalid. The following options are available:

- Specifying whether OCSP requests should be signed (bSignRequest).
- Requiring the presence of a particular OID in a request (sSignCertOID).

### Table 37 Registry preferences: OCSP certificate

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bSignRequest</td>
<td>bool</td>
<td>(v.7.0) Default: 0</td>
</tr>
<tr>
<td>sSignCertOID</td>
<td>string</td>
<td>(v.7.0) Default: null</td>
</tr>
</tbody>
</table>

Tip: For more detail about how revocation checking affects signing and signature validation, see Chapter 7, “Certificate Processing”.

### 5.3.1.2 Configuring CRL Revocation Checking

CRL revocation checking can occur both during signature creation and signature validation on both the signing certificate as well as for the certificates associated with any revocation check responses. The following options are available:

- “Specifying the Time and Method of CRL Checks” on page 108
- “Specifying Certificates for Valid CRL Checks” on page 109

### Specifying the Time and Method of CRL Checks

CRL revocation checking preferences allow you to control when and how a CRL check occurs. The following options are available:

- Specifying when to do revocation checking as well as the affect of a failed or bad response.
- Specifying when and where to go online to get a response.
- Setting a time limit for caching a response after which the application must get a new response.
- Specifying a LDAP server to query for CRLs.
Note: Querying an LDAP server can result in poor application performance depending on the quality of the network connection and the number of directories to search.

Table 39 Registry preferences: CRL

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iReqRevCheck</td>
<td>int</td>
<td>(v 7.0) Default: 1</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cAdobe_CRLRevChecker</td>
</tr>
<tr>
<td>Indicates whether revocation checks are required to succeed on the CRL response.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interacts with other iReqRevCheck settings as described in “Interaction of iReqRevCheck Settings” on page 158.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0: Don’t do revocation checks.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1: Do a check if responder details are in CRLDp certificate extension or the registry; don’t fail if the check fails.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2: Do a check if responder details are in CRLDp certificate extension or the registry; all checks must succeed if there is data and a check occurs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3: Require a check; it must succeed under all circumstances.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sURL</td>
<td>string</td>
<td>(v. 7.0) Default: null</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cAdobe_CRLRevChecker\cURLDP</td>
</tr>
<tr>
<td>The URL used to fetch CRL responses for an additional URL CRL Distribution point.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bAlwaysConsult</td>
<td>bool</td>
<td>(v. 7.0) Default: 0</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cAdobe_CRLRevChecker\cURLDP</td>
</tr>
<tr>
<td>Optional. Determines when the URL is used for an additional URL CRL distribution point. If false, the URL is only used when the certificate does not have a CRLDp extension.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bGoOnline</td>
<td>bool</td>
<td>(v. 7.0) Default: 1</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cAdobe_CRLRevChecker</td>
</tr>
<tr>
<td>Indicates whether it’s acceptable to go online to fetch a CRL. If false, only cached CRLs (on local disk or ones embedded with signature) are consulted. Internally set to false for ubiquity signatures in Reader enabled documents internally.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sLDAP</td>
<td>string</td>
<td>(v. 7.0) Default: Null</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cAdobe_CRLRevChecker</td>
</tr>
<tr>
<td>The LDAP server to get CRLs from in the form <a href="http://www.ldap.com">www.ldap.com</a>. Without the protocol prefix, as LDAP is assumed. All DN-based queries for CRLs will be directed to this server.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iMaxRevokeInfoCacheLifetime</td>
<td>int</td>
<td>(v 7.0) Default: 24 (hours)</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td>Maximum cache lifetime of the information (e.g. CRL) used to do revocation checking.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Specifying Certificates for Valid CRL Checks

It is possible to require certain features for certificates used to sign CRL responses. If a response does not meet the specified parameters, the response will be considered invalid and the signature status may be Unknown or Invalid. The following options are available:

- Specifying whether to ignore the response certificate’s times in the thisUpdate and nextUpdate extensions.
- Requiring the presence of the Authority Key Identifier extension.
5.3.1.3 Certificate Chain Building

The revocation checking process includes building the certificate chain so that each discovered certificate can be analyzed and processed as specified by other application preferences. Administrators do have some control over what certificates are used to build a chain. The following options are available:

- Controlling whether AIA extensions are followed.
- Requiring the use of valid RSA signatures on all certificates in a chain.
- Requiring the presence of specific policy OIDs in the specified chain scope for it to be valid.
- Pointing to an LDAP server for path discovery purposes. Querying an LDAP server can result in poor application performance depending on the quality of the network connection and the number of directories to search.

### Table 40 Registry preferences: CRL

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bIgnoreValidityDates | bool | (v. 7.0) Default: 0  
  Path: `<security root>\cASPKI\cAdobe_CRLRevChecker`  
  Specifies whether to ignore the response's `thisUpdate` and `nextUpdate` times, thereby preventing any negative affect of these times on response validity.  
  The value is set to true for ubiquity signatures created by enabling usage rights for Adobe Reader. |
| bRequireAKI         | bool | (v. 8.0) Default: 0  
  Path: `<security root>\cASPKI\cAdobe_CRLRevChecker`  
  Specifies whether the Authority Key Identifier extension must be present in a CRL. |

### Table 41 Registry preferences: Chain building

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| bFollowURIsFromAIA  | bool | (v. 7.0) Default: 0  
  Path: `<security root>\cASPKI\cAdobe_ChainBuilder`  
  If true, the chain builder is allowed to follow URIs in AIA certificate extensions so that certificates can be downloaded if they are not available locally.  
  The default does not allow phone-home capability. CRLdps and OCSP AIA extensions do allow following URIs because they require that the certificate chain up to a trust anchor. |
| bRequireValidSigForChaining | bool | (v. 8.0) Default: 0  
  Path: `<security root>\cASPKI\cAdobe_ChainBuilder`  
  If true, the chain builder will not build chains with invalid RSA signatures on certificates. Consider chain CA > ICA > EE where the CA's signature on an ICA is invalid. If this setting is true, the chain building will stop at the ICA and the CA will not be included in the chain. If this preference is false, the full 3-certificate chain is produced. This setting does not affect DSA signatures. |
5.3.2 Signature Creation Preferences

**Tip:** For a one page guide, see “Signature Creation Workflow” on page 313.

Signature creation preferences can control the type and strength of signature as well as what conditions will cause signing to fail. The available options allow you to configure signing workflows for varying degrees of security (Figure A.1). The following options are available:

- Preferences that control the signature strength and format:
  - “Changing the Default Hashing Algorithm” on page 111
  - “Specifying a Signing Format” on page 112
- Preferences that control what is included in a signature:
  - “Configuring Revocation Checking for Signing” on page 113
  - “Embedding Revocation Data in Signatures” on page 113
  - “Setting up Timestamp Servers” on page 116
- Preferences that control when to abort the signature creation process:
  - “Requiring a Pre-Signing Digest Comparison” on page 114
  - “Requiring Signature Property Retrieval” on page 115

5.3.2.1 Changing the Default Hashing Algorithm

The default algorithm used to create a message digest (document hash) during signing can be customized. In some enterprise situations, such as when FIPS compliance is required, you may need a more secure algorithm. Alternate hashing algorithms can be specified by name or OID as shown below. The algorithm used is displayed in the Hash Algorithm field of the Signature Property dialog Document tab.

Usage rules:
- MSCAPI supports different algorithms across versions. For example, early XP versions only supported SHA1 and MD5. The use of other algorithms will require that the signer use a digital ID that resides in a .pfx/.p12 file in the Acrobat cache.
- With XP SP3, MSCAPI supports SHA256 on certificates and some token devices.
- Pre 9.1: Acrobat uses SHA1 as the default.
9.1 and later: Acrobat uses SHA256 as the default, but will use SHA1 if the token does not support SHA256.

If using FIPS mode, do not use MD5 or RIPEMD160.

The following options are available:

- Specifying an alternate algorithm.

### Table 42 Registry preferences: Signing algorithm

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSignHash</td>
<td>ASAtom</td>
<td>(v 7.0) Default: SHA1 for 9.0 and earlier; SHA256 for 9.1 and later</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Path: <code>&lt;security root&gt;\cPubSec</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The hashing algorithm to use while signing. The ASAtom is a binary entry that uses the name of a supported algorithm:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MD5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- RIPEMD160</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SHA1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SHA256 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SHA384 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SHA512 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For an alternative, see tSignHash.</td>
</tr>
<tr>
<td>tSignHash</td>
<td>ASAtom</td>
<td>(v 7.0) Default: SHA1 for 9.0 and earlier; SHA256 for 9.1 and later</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Path: <code>&lt;security root&gt;\cPubSec</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>A text entry that contains the OID of the hashing algorithm:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 1.2.840.113549.2.5: MD5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 1.3.36.3.2.1: RIPEMD160</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 1.3.14.3.2.26: SHA1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 2.16.840.1.101.3.4.2.1: SHA256 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 2.16.840.1.101.3.4.2.2: SHA384 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 2.16.840.1.101.3.4.2.3: SHA512 (v. 8.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For an alternative, see aSignHash.</td>
</tr>
</tbody>
</table>

### 5.3.2.2 Specifying a Signing Format

The default format for creating the signature object that is embedded in a signed document is PKCS#7. The object contains the encrypted message digest, certificates, timestamps, and other information. It does not include the signature appearance and data outside of Contents in the signature dictionary. Format choices are limited so that a signature encoded by one handler can be unencoded (validated) by another handler. Providing a value for aSignFormat writes that value to the signature dictionary's SubFilter object. For details, see “Signature Interoperability” in the PDF Reference.

- **PKCS#1**: For signing PDF files using PKCS#1, the only recommended value of SubFilter is adbe.x509_rsa_sha1, which uses the RSA encryption algorithm and SHA-1 digest method. The certificate chain of the signer is stored in the Cert entry.

- **PKCS#7**: The value of Contents is a DER-encoded PKCS#7 binary data object containing the signature. The PKCS#7 object must conform to the PKCS#7 specification in Internet RFC 2315, PKCS #7: Cryptographic Message Syntax, Version 1.5. SubFilter can take one of the following values:
  - **adbe.pkcs7.detached**: No data is encapsulated in the PKCS#7 signed-data field.
5.3.2.3 Configuring Revocation Checking for Signing

Applying a signature to a document involves both creating a signature and then validating it. Despite the fact that end users see only one step (the signature appears with a status icon), there are actually two phases which an administrator can configure independently of the other. Revocation checking can occur during the initial signing phase to control whether or not a signature is created. The following option is available:

- Specifying when to do revocation checking as well as the affect of a failed or bad response.

  **Note:** Interacts with `bIsEnabled`. For more detail about how revocation checking affects signing and signature validation, see Chapter 7, “Certificate Processing”.

### Table 44 Registry preferences: Revocation checking (signing)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| iReqRevCheck | int  | (v 7.0) Default: 0  
  **Path:** `<security root>`\`\cASPKI\cASPKI\cSign  
  Indicates whether revocation checks are required to succeed to create the signature.  
  Interacts with other iReqRevCheck settings as described in “Interaction of iReqRevCheck Settings” on page 158.  
  **0:** Don't do revocation checks.  
  **1:** Do a check IF CRLDp or AIA information resides in the certificate or registry; don't fail if the check fails.  
  **2:** Do a check IF CRLDp or AIA information resides in the certificate or registry; all checks must succeed if there is data and a check occurs.  
  **3:** Require a check; it must succeed under all circumstances. |

5.3.2.4 Embedding Revocation Data in Signatures

Whether revocation checking information is stored in a signature varies by version. Storing such data in a signature enables offline revocation checking and a determination of whether a signer's certificate was valid at the time of signing.

### Table 43 Registry preferences: Signing format

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| aSignFormat | ASAtom | (v 7.0) Default: adbe.pkcs7.detached  
  **Path:** `<security root>`\`\cPubSec  
  The format to use when signing a document using public key cryptography when a format is not specified by a seed value, javascript parameter, or the PubSec Handler. Allowable values include:  
  - adbe.pkcs7.detached  
  - adbe.pkcs7.sha1  
  - adbe.x509.rsa_sha1 |
Setting `bIsEnabled` to 1 via the GUI or registry automatically sets `cSign\iReqRevCheck` to 2. The rationale is that if you choose to embed the revocation status, you probably want a status to embed. A consequence of this choice is that you must do a check and retrieve a good result; otherwise, no signature is created. In other words, signing with a revoked certificate is prevented when this setting is on.

The following options are available:

- Embedding revocation status in a signature.
- Specifying the embedded data cache size to limit the amount of cached data.

**Tip:** If you are setting up a signing workflow for both signers and signature validators, you may want to set `iUseArchivedRevInfo` so that document recipients will validate signatures based on the signer’s `bIsEnabled` setting.

### Table 45 Registry preferences: Long term validation

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bIsEnabled</code></td>
<td>bool</td>
<td>Maps to GUI item: <strong>Include signature’s revocation status when signing</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specifies whether the signature revocation status is included in the signature.</td>
</tr>
<tr>
<td><code>iMaxRevInfoArchiveSize</code></td>
<td>int</td>
<td>Maps to GUI item: <strong>Include signature’s revocation status when signing</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum size of the revocation archival info in kilobytes. An attempt is made to store as much revocation information as possible without exceeding the limit.</td>
</tr>
</tbody>
</table>

#### 5.3.2.5 Requiring a Pre-Signing Digest Comparison

When signing a PDF document, a message digest is created for the document and sent to the cryptographic module that performs the signing operation. Setting the registry entry `bEnforceSecureChannel` to 1 ensures the message digest sent to the cryptographic module is checked against the signed message digest that it returns. This flag ensures that intermediate layers of software between Acrobat and the cryptographic module do not tamper with the signing operation.

**Note:** When using a certificate that includes a DSA public key with omitted parameters, the test to detect signature validity is not performed. In these cases, setting `bEnforceSecureChannel` has no effect.

When this preference is turned on, a digest mismatch results in a warning dialog (Figure 47). The signature is removed from the document and the signing application aborts the signing process.
5.3.2.6 Requiring Signature Property Retrieval

Acrobat currently provides a signature property for timestamps. By default, retrieving a valid and trusted timestamp is not required, and property retrieval failure only results in creating a signature which uses the local time.

The following option is available:

- Requiring property retrieval in order to create a signature. If fetching a timestamp fails for any reason (bad URL, no network connection, etc.) the signature creation process is aborted and no signature is created.

### Table 47 Registry preferences: Signature property retrieval

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bReqSigPropRetrieval</td>
<td>bool</td>
<td>(v 7.0) Default: 0</td>
</tr>
<tr>
<td>Path</td>
<td></td>
<td>&lt;security root&gt;\cASPKI\cASPKI\cSign</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Indicates whether retrieving a signature property must succeed. For example,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>if a user configures a bad timestamp server URL and makes it the default, a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>signature is not created. An error appears.</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Make best effort, but success is not required. A signature is created.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Property retrieval must succeed. On failure, a signature is not created and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>an error dialog appears.</td>
</tr>
</tbody>
</table>

---

**Registry and plist Settings**

**Signature Creation Preferences**

**Figure 47 Secure channel error**

![Secure channel error](image)

### Table 46 Registry preferences: Message digest comparison

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bEnforceSecureChannel</td>
<td>bool</td>
<td>(v 8.0) Default: 0</td>
</tr>
<tr>
<td>Path</td>
<td></td>
<td>&lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This setting prevents signing when the original message digest and the signed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>message digest do not match. This error can be caused by a modification of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>original message digest, a modification of the signed message digest, or a mismatch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>between the private and public key used for signing. When this preference is on,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the user sees a warning dialog when the digest mismatch occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When using a certificate that doesn't include a public key (such as a DSA certificate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>with an omitted public key), the test to detect signature validity is not performed. Do not turn this setting on if such certificates are used.</td>
</tr>
</tbody>
</table>
5.3.2.7 Setting up Timestamp Servers

Timestamp servers are automatically used during signing when a server is configured and selected as a default. The default server is identified by a star in the Security Settings Console. `cPPKHandler` can be used to specify a list of servers and `cAdobe_TSPProvider` can specify the default server that is actually used during signing.

End users can overwrite preconfigured values by editing server settings in the Security Settings Console. Note that if a user sets a new default server, the console values will overwrite the values in `cAdobe_TSPProvider`. Values under `cPPKHandler` do not get written to `cAdobe_TSPProvider` unless that server is selected as the default.

The following options are available:

- Specifying a list of servers that will appear in the Security Settings Console. Preferences are represented as a list c0-cN and contain the server name, URL, and whether the authentication is required (Table 48).
- Setting a default server (Table 49).
- Specifying when to do revocation checking as well as the affect of a failed or bad response.
- Increasing security by choosing a more robust hashing algorithm. The algorithm must be supported by the timestamp server.
- Requiring signature property retrieval (a valid and trusted server URL) in order to create a signature. For details, see “Requiring Signature Property Retrieval” on page 115.

**Note:** See also “Signature Validation Preferences” on page 124.

---

**Table 48 Registry preferences: Timestamp server list**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| tServer | `cab` | (v 7.0) Default: null  
Path: `<security root>\cPPKHandler\cTimeStampServers\c<index>`  
The server URL. Describes a timestamp server. It is created from the Security Settings dialog with the timestamp Server configuration. |
| tName | `text` | (v 7.0) Default: null  
Path: `<security root>\cPPKHandler\cTimeStampServers\c<index>`  
The user-defined server name. This can be Unicode. |
| bAuthRequired | `bool` | (v 7.0) Default: null  
Path: `<security root>\cPPKHandler\cTimeStampServers\c<index>`  
Maps to GUI item: **This server requires me to log on**  
If true, indicates that the above timestamp server requires authentication. |
xLockboxId string (v 7.0) Default: null
Path: <security root>\cPPKHandler\cTimeStampServers\c<index>
Maps to GUI item: The preference is populated when the user checks *This server requires me to log* on and then enters a username and password.
If a timestamp server requires authentication, the authentication data is stored in a secure store such as Microsafe and is identified by this ID. The service provider needs to know what type of secure store the identifier names. Only used when ASPKI is running within the Acrobat environment.

### Table 49 Registry preferences: Default timestamp server

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| iReqRevCheck | int     | (v 7.0) Default: 2  
Path: <security root>\cASPKI\cAdobe_TSPProvider  
Indicates whether revocation checks on timestamps are required to succeed before signing. Failure does not affect signature creation or validation, it only results in defaulting to the local, machine time. 
Interacts with other iReqRevCheck settings as described in "Interaction of iReqRevCheck Settings" on page 158. The possible values include:  
0: Don’t do revocation checks.  
1: Do a check IF CRLDp or AIA information resides in the certificate or registry; don’t fail if the check fails.  
2: Do a check IF CRLDp or AIA information resides in the certificate or registry; all checks must succeed if there is data and a check occurs.  
3: Require a check; it must succeed under all circumstances. |
| sURL         | string  | (v 7.0) Default: null  
Path: <security root>\cASPKI\cAdobe_TSPProvider  
A timestamp server URL such as http://www.example.com/tsp. Because no default is specified, it must be configured for timestamping to work. Only the HTTP(s) protocol is supported. |
| bAuthRequired| bool    | (v 7.0) Default: null  
Path: <security root>\cASPKI\cAdobe_TSPProvider  
If true, indicates that the above timestamp server requires authentication. |
| sUser        | string  | (v 7.0) Default: null  
Path: <security root>\cASPKI\cAdobe_TSPProvider  
The server login username. Relevant only if bAuthRequired is true. Only username and password-based authentication is supported. |
| sPassword    | string  | (v 7.0) Default: null  
Path: <security root>\cASPKI\cAdobe_TSPProvider  
The server login password. Relevant only if bAuthRequired is true. |
5.3.3 Signing Workflow Preferences

Signing workflow preferences control what end users can see and do when they invoke the signing dialog. You can require certain actions, hide and display data fields, and change how the signing process is affected by content which might impact the users ability to see what they are signing. The following options are available:

- “Requiring Preview Mode to Sign” on page 119
- “Specifying Signature Appearances” on page 119
- “Preconfiguring Location and Contact Information” on page 119
- “Specifying Signing Reasons” on page 120
- “Certification Signature Preferences” on page 121
- “Controlling Signature Warnings: Review” on page 122
- “Controlling Signature Warnings: Fonts” on page 123

### Table 49 Registry preferences: Default timestamp server

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| xLockboxId  | string (v 7.0) Default: null | Path: <security root>\cASPKI\cAdobe_TSPProvider  
Maps to GUI item: The preference is populated when the user checks **This server requires me to log on** and then enters a username and password.  
If a timestamp server requires authentication, the authentication data is stored in a secure store such as Microsafe and is identified by this ID. The service provider needs know what type of secure store the identifier names. Only used when ASPKI is running within the Acrobat environment. |
| iHashAlgo   | int (v 7.0) Default: 2 | Path: <security root>\cASPKI\cAdobe_TSPProvider  
Identifies the hashing algorithm used to hash the timestamped data. The valid values are:  
0: MD5  
1: SHA1 (Default prior to 9.1)  
2: SHA256 (Supported beginning with 8.0 and the default since 9.1)  
For an alternative, see sHashAlgo which supports more options. |
| sHashAlgo   | string (v 8.0) Default: SHA1 pre 9.1; SHA256 9.1 and later | Path: <security root>\cASPKI\cAdobe_TSPProvider  
The hashing algorithm OID used to hash the data to be timestamped. The valid values are:  
- MD5: 1.2.840.113549.2.5  
- RIPEMD160: 1.3.36.3.2.1  
- SHA1: 1.3.14.3.2.26  
- SHA256: 2.16.840.1.101.3.4.2.1  
- SHA384: 2.16.840.1.101.3.4.2.2  
- SHA512: 2.16.840.1.101.3.4.2.3 |
| iSize       | int (v 7.0) Default: 4096 | Path: <security root>\cASPKI\cAdobe_TSPProvider  
ASPKI requires the signature property to predict the size (in bytes) so that enough space can be set aside. |
5.3.3.1 Requiring Preview Mode to Sign

Preview mode turns off (suppresses) content and dynamic behavior that could prevent the signer from seeing what they are signing. While the use of preview mode adds an extra step in the signing workflow, it turns off potentially bad content, checks the document for the presence of any PDF constructs that may cause problems with signature integrity, and provides a report about any found problems. For more information about preview mode, see Chapter 13, “Document Integrity and Preview Mode”.

The following option is available:

- Force the use of preview mode during signing.

Table 50 Registry preferences: Preview mode

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bPreviewModeBefore Signing</td>
<td>bool</td>
<td>(v 7.0) Default: 0&lt;br&gt;&lt;br&gt;Path: &lt;security root&gt;\cDigSig&lt;br&gt;Maps to GUI item: Preferences &gt; Security &gt; View documents in preview mode when signing&lt;br&gt;Specifies whether a signer is forced to use preview mode during signing. If true, preview mode is automatically invoked on a sign action. Users should read the document message bar text, view a report about any warnings, and then choose Sign Document.</td>
</tr>
</tbody>
</table>

5.3.3.2 Specifying Signature Appearances

The application remembers what signature appearance a signer used and stores its index number in iAPIndex. Because an end user’s appearance selection will overwrite any custom value here, customization by an administrator would server no useful purpose.

**Note:** This key is not customizable and is provided for informational purposes only.

Table 51 Registry preferences: Appearance

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iAPIndex</td>
<td>int</td>
<td>(v 7.0) Default: null.&lt;br&gt;&lt;br&gt;Path: &lt;security root&gt;\cPubSec&lt;br&gt;Remembers the last used signature appearance index.</td>
</tr>
</tbody>
</table>

5.3.3.3 Preconfiguring Location and Contact Information

The signing dialog has the capability of showing a location and contact information fields during a signing workflow. Field fill-in is optional. By default, the option is off, but end users and administrators can turn this option on. The location will appear in the Signature Properties dialog and in the Signature’s pane and may optionally appear in the signature appearance. The following options are available:

- Showing or not showing the **Contact** and **Location** fields in the signing dialog.
- Setting default contact information.
- Setting default location information.
Note: If the end user changes the field data in the signing dialog, those values will overwrite the registry-specified values.

### Table 52  Registry preferences: Signing information

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAllowOtherInfoWhenSigning</td>
<td>bool</td>
<td>(v 8.0) Default: 0 &lt;br&gt;Path: <code>&lt;security root&gt;\cPubSec</code> &lt;br&gt;Maps to GUI item: Preferences &gt; Security &gt; Advanced Preferences &gt; Creation tab &gt; Show location and contact information when signing &lt;br&gt;Specifies whether the location and contact information UI will appear during signing.</td>
</tr>
<tr>
<td>tContactInfo</td>
<td>text</td>
<td>(v 7.0) Default: null &lt;br&gt;Path: <code>&lt;security root&gt;\cPubSec</code> &lt;br&gt;Maps to GUI item: Contact field in the Sign dialog. &lt;br&gt;When bAllowOtherInfoWhenSigning is true (on), the signing dialog displays a location and contact field. User data is saved and reused during subsequent signing events.</td>
</tr>
<tr>
<td>tLocation</td>
<td>text</td>
<td>(v 7.0) Default: null &lt;br&gt;Path: <code>&lt;security root&gt;\cPubSec</code> &lt;br&gt;Maps to GUI item: Location field in the Sign dialog. &lt;br&gt;When bAllowOtherInfoWhenSigning is true (on), the signing dialog displays a location and contact field. User data is saved and reused during subsequent signing events.</td>
</tr>
</tbody>
</table>

### 5.3.3.4 Specifying Signing Reasons

The signing dialog has the capability of showing a signing reasons drop down list during a signing workflow. By default, the option is off, but end users and administrators can turn this option on. If a reason is used, it appears in the signature appearance, the Signature Properties dialog, and in the Signatures pane. The following options are available:

- Showing or not showing the Reasons field in the signing dialog.
- Changing the default reasons. Administrators can add, delete, and modify the reason list.
- Locking the reason list so that it can’t be modified by end users.

### Table 53  Registry preferences: Signing reason

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAllowReasonWhenSigning</td>
<td>bool</td>
<td>(v 8.0) Default: 0 &lt;br&gt;Path: <code>&lt;security root&gt;\cPubSec</code> &lt;br&gt;Maps to GUI item: Preferences &gt; Security &gt; Advanced Preferences &gt; Creation tab &gt; Show reasons when signing &lt;br&gt;Specifies whether the reason UI will appear during signing. The preference can be overridden by a document seed value set on a field. &lt;br&gt;8.1: Subject to lockdown as described in “Preventing End-User Modification” on page 75. If cReasons is locked and is empty, bAllowSigningReasons is 0 and read only (The UI is turned off). If cReasons is locked and has values, then bAllowSigningReasons is true and read only.</td>
</tr>
</tbody>
</table>
5.3.3.5 Certification Signature Preferences

These preferences only control certification signature behavior and have no effect on approval signature behavior. In addition to the general signature preferences described elsewhere in this document, the following options are available:

- **Preventing invisible signatures**: By default, users can sign with a visible or invisible signature. Prohibit invisible certification signatures by setting `bAllowInvisibleSig` to 0.

- **Legal attestations (warning comments)**: When certifying a document that contains dynamic content, a signer can choose a default warning comment from a list or create a custom one. You can prepopulate this list with custom comments with `cAttest`.

- **(Pre v. 8.0) Control certification based on document content**: For versions prior to 8.0, you can control certification rights based on the nature of the document content and whether it generates LegalPDF warnings. These preferences are deprecated in 8.0.

Table 54 Registry preferences: Certification signature

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `bAllowInvisibleSig` | bool | (v 7.0) Default: 1<br><br>Path: `<security root>\cDigSig`<br>Maps to GUI item: Certify with Invisible Signature<br>If true, invisible signatures are allowed. False disables the menu option, prevents signing and certifying with invisible signatures, and limits JavaScript support by signature fields.<br>Subject to lockdown as described in “Preventing End-User Modification” on page 75.
5.3.3.6 Controlling Signature Warnings: Review

The Sign dialog is capable of showing a **Review** button. The button invokes the PDF Signature Report which analyzes the document for the presence of any dynamic content that could adversely affect the integrity of signing workflows. If none is found, a dialog appears indicating that there are no problems. If content such as a comment or JavaScript is discovered, the PDF Signature Report appears with a list of any PDF constructs that may cause problems with signature integrity. The following options are available:

- Never showing or allowing the review of document warnings.
- Limiting warning review to certification workflows.
- Requiring warning review prior to applying an approval and/or certification signature.
- Always requiring review of warnings for every signature.

### Table 54 Registry preferences: Certification signature

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| cAttest               | cab of text   | (v 7.0) Default: null  
Path: <security root>cDigSig  
List of most recently used attestations regarding LegalPDF warnings in a document. Entries in this folder are named t0, t1, etc. The application may have one or more default strings such as "I have included this content to make the document more interactive." |
| bAllowCertNonGreen    | bool          | (v 7.0 ONLY) Default: 1  
Path: <security root>cDigSig  
If true, a certification signature may be applied to a document containing Legal PDF warnings. If false, then its not allowed and the author is informed of the reason. |
| bAllowSigCertOnly     | bool          | (v 7.0 ONLY) Default: 0  
Path: <security root>cDigSig  
Specifies whether any subsequent signers can sign a certified document containing LegalPDF warnings with additional approval signatures. In other words, the presence of any LegalPDF warning prevent any additonal signatures. |
| bAllowSigCertGreenOnly| bool          | (v 7.0 ONLY) Default: 0  
Path: <security root>cDigSig  
Specifies whether any subsequent signers can sign a certified document that does not contain LegalPDF warnings with additional approval signatures. In other words, the presence of any LegalPDF warning does not prevent any additonal signatures. |
5.3.3.7 Controlling Signature Warnings: Fonts

LegalPDF warnings have been replaced by PDF Signature Report errors in versions 8.0 and later. Both mechanisms provide similar warnings. The following option is available:

- Toggling warnings for true type and non-embedded fonts on and off.

### Table 55 Registry preferences: Document warning

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iShowDocumentWarnings</td>
<td>int</td>
<td>(v 8.0) Default: 1</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cPubSec</td>
</tr>
<tr>
<td>Maps to GUI item:</td>
<td></td>
<td>Preferences &gt; Security &gt; Advanced Preferences &gt; Creation tab &gt; Enable reviewing of document warnings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specifies whether a button to allow reviewing document warnings shows up on the signing dialog. Interacts with iRequireDocumentWarnings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0: Never</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Show when certifying only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: Always</td>
</tr>
</tbody>
</table>

| iRequireReviewWarnings      | int  | (v 8.0) Default: 0                                                          |
| Path:                       |      | <security root>\cPubSec                                                      |
| Maps to GUI item:           |      | Preferences > Security > Advanced Preferences > Creation tab > Prevent signing until document warnings are reviewed. |
|                             |      | Specifies whether the user is required to review document warnings before signing via the signing dialog. Interacts with iShowDocumentWarnings. |
|                             |      | 0: Never                                                                    |
|                             |      | 1: Show when certifying only                                                |
|                             |      | 2: Always                                                                  |

### Table 56 Registry preferences: Font warning

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bEnNonEmbFontLegPDFWarn</td>
<td>bool</td>
<td>(v 7.0) Default: 0</td>
</tr>
<tr>
<td>Path:</td>
<td></td>
<td>&lt;security root&gt;\cDigSig</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Turns on and off warnings about non-embedded fonts. A warning appears when the LegalPDF dictionary NonEmbeddedFonts attribute has a non zero value. Turning this value on causes a warning to appear in the PDF Signature Report which indicates the document contains unembedded fonts.</td>
</tr>
</tbody>
</table>

| bTrueTypeFontPDFSigQWarn    | bool | (v 7.0) Default: 0                                                          |
| Path:                       |      | <security root>\cDigSig                                                      |
|                             |      | Turns on and off warnings about True Type fonts.                           |
5.3.4 Signature Validation Preferences

Signature validation preferences control the display of status icons, logging, how validation occurs, and so on. Many of these preferences interact with the signing preferences and should be set accordingly. The following options are available:

- “Setting Validation Preferences that Map to UI” on page 124
- “Controlling Signature Status Icon Behavior” on page 126
- “Logging Certificate Validation Data” on page 127
- “Using Embedded Validation Data” on page 128
- “Revocation Checking Constraints” on page 129
- “Displaying All Chains in the Certificate Viewer” on page 99

**Tip:** For more detail about how revocation checking affects signing and signature validation, see Chapter 7, “Certificate Processing”.

5.3.4.1 Setting Validation Preferences that Map to UI

While users can configure general application signature validation preferences via the GUI, admins can preconfigure the application. The following options are available:

- Controlling whether all signatures are validated when a document opens.
- Specifying which time to use when validating a signature.
- Specifying when to do revocation checking as well as the affect of a failed or bad response.
- Using expired timestamps.
- Showing timestamp warnings in the Document Message Bar.

### Table 57 Signature validation preferences

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bValidateOnOpen</td>
<td>bool</td>
<td>(v 7.0) Default: 1&lt;br&gt;Path: &lt;security root&gt;\cDigSig&lt;br&gt;Maps to GUI item: Preferences &gt; Security &gt; Verify signatures when the document is opened&lt;br&gt;Specified whether to automatically validate all signatures on document open. Subject to lockdown as described in “Preventing End-User Modification” on page 75.</td>
</tr>
<tr>
<td>iSigVerificationTime</td>
<td>int</td>
<td>(v 7.0) Default: 1&lt;br&gt;Path: &lt;security root&gt;\cPPKHandler&lt;br&gt;Maps to GUI item: Preferences &gt; Security &gt; Advanced Preferences &gt; Verification tab &gt; Verify Time&lt;br&gt;Each radio button corresponds to a value below:&lt;br&gt;0: Always carry out the verification at current time&lt;br&gt;1: Use the signing time if it’s secure (e.g. timestamped), else use current time&lt;br&gt;2: Always use signing time</td>
</tr>
</tbody>
</table>

**Acrobat Family of Products**

124
5.3.4.2 Certificate Chain Validation Method

The application uses shell validation by default, but chain validation may be used when required. Compliance with the German signature law requires chain validation.

<table>
<thead>
<tr>
<th>Table 58 Registry preferences: Chain validation method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
</tr>
<tr>
<td>iValidityModel</td>
</tr>
</tbody>
</table>

**Path:** `<security root>\cASPKI\cASPKI\cProvider`

Specifies the validity model for validating signatures and certificates.

<table>
<thead>
<tr>
<th>0: PKIX shell model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Chain validity model</td>
</tr>
</tbody>
</table>

Chain validation is used to validate all or part of a certificate chain when any certificate chaining up to a CA certificate containing the qualified certificate policy extension (OID 1.3.36.8.1.1) or the validity model certificate extension OID (1.3.6.1.4.1.8301.3.5.1) with the value set to the chain model OID (1.3.6.1.4.1.8301.3.5.1).
5.3.4.3 Controlling Signature Status Icon Behavior

By default, when an application validates a signature it displays a signature status icon in the Signature Properties dialog, and in the Signatures Pane. You can customize status icon behavior for a particular enterprise requirement. For example, a blue i appears on a signature status icon based on certain rules when a document is changed after it was signed.

The following options are available:

- Turning on the icon for signature appearances with \( b\text{SigAPStatusIconDisable} \). This is off by default because displaying the signature status within the document represents a security vulnerability.
- Turning off the icon for signature appearances AND remove the Hide signature field validity icon when signature is valid from the user interface so the user cannot change the setting with \( i\text{DisplayValidIcon} \).
- Turning on the icon for valid signatures only with \( i\text{DisplayValidIcon} \).
- Turning off the blue i in the Signature Properties dialog, and Signatures Pane with \( b\text{ShowWarningForChanges} \).

Table 59 Registry preferences: Signature status icon

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b\text{SigAPStatusIconDisable} )</td>
<td>bool (v 7.0) Default: 0</td>
<td>Controls whether the signature status icon is displayed in the signature appearance on the document. If true, status icon is not displayed regardless of signature status. This setting overrides ( i\text{DisplayValidIcon} ) and ( b\text{ShowWarningForChanges} ).</td>
</tr>
<tr>
<td>( i\text{DisplayValidIcon} )</td>
<td>int (v 7.0) Default: 2 for versions 9.0 and later. 0 for older versions.</td>
<td>Controls whether the signature status icon is displayed in a signature appearance. 0: Always 1: Display except when the signature is valid 2: Never. This value disables ( b\text{ShowWarningForChanges} ) and removes the Hide signature field validity icon option from the GUI. This setting does not affect the icons in the Signatures Pane or in the Signature Properties dialog</td>
</tr>
</tbody>
</table>
5.3.4.4 Logging Certificate Validation Data

Versions 8.x and later enable logging certificate validation and revocation checking information. You can set both the logging level and log location (Table 60). The path must already exist for logging to take place. The following options are available:

- Specifying a logging path and filename.
- Setting a logging level.

Example 5.7: Chain building log file settings

```
[HKEY_CURRENT_USER\Software\Adobe\Adobe Acrobat\8.0\Security\cASPKI\cAdobe_ChainBuilder]
"iLogLevel"=dword:00000008
"sLogFilePath"=<BINARY path to existing directory for log file>
```

Example 5.8: Log file for troubleshooting certificate validation

```
20070207000213Z: ---------------------------
20070207000213Z: Chain builder: Starting chain validation. Chain length = 3
20070207000213Z: Processing Certificate: DN: ou=VeriSign Trust Network, ou=(c) 1998 VeriSign, Inc. - For authorized use only, ou=Class 2 Public Primary Certification Authority - G2, o=VeriSign, Inc., c=US Serial: 00B92F60CC89FA17A4609B85B706C8AAF
20070207000213Z: verification time = 20070207000213Z
20070207000213Z: Processing Certificate: DN: cn=Enterprise Services CA, ou=Class 2 OnSite Individual Subscriber CA, ou=Terms of use at https://www.verisign.com/rpa (c)01, ou=VeriSign Trust Network, o=Adobe Systems Incorporated Serial: 0C0DB7043D0427BEB152ECA02DC95903
20070207000213Z: verification time = 20070207000213Z
20070207000213Z: verification time = 20070207000213Z
20070207000213Z: Finished Chain Validation. TroubleFlags: 0
```
5.3.4.5 Using Embedded Validation Data

Administrators can control how embedded revocation information is used. The following options are available:

- Specifying when archived revocation data is used for revocation checking.

  **Tip:** If you are setting up a signing workflow for both signers and signature validators, you may want to set \( iUseArchivedRevInfo \) so that document recipients will validate signatures based on the signer’s \( bIsEnabled \) setting.

- Controlling whether or not revocation data is stored in a JavaScript object.

### Table 60 Registry preferences: Logging

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sLogFilePath | bin  | (v. 8.0) Default: null  
**Path:** \(<security root>\cASPKI\cAdobe_ChainBuilder\)  
Specifies the full path of the text log file. For example: C:\ASPKI.log. The parent folder must already exist. |
| iLogLevel  | int  | (v. 8.0) Default: null  
**Path:** \(<security root>\cASPKI\cAdobe_ChainBuilder\)  
Specifies the log level during chain building and validation. The supported levels are:  
1: fatal errors  
2: possible errors  
4: informational messages  
8: verbose information  
f: all messages |

### Table 61 Registry preferences: Signature validation

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| iUseArchivedRevInfo   | int  | (v. 7.0) Default: 2  
**Path:** \(<security root>\cASPKI\cAdobe_LTVProvider\)  
Indicates whether the revocation information archived with the signature is used for revocation checking.  
0: Never  
1: Use only if more recent info is not available.  
2: Always |
| bReturnRevInfoToUser  | bool | (v 7.0) Default: 0  
**Path:** \(<security root>\cPPKHandler\)  
If true, the revocation information is maintained within the \SignatureInfo\ object and can be retrieved through JavaScript. For more information, see the Acrobat JavaScript Reference. |
5.3.4.6 Revocation Checking Constraints

Signature validation can have dependencies on other keys. The following options are available:

- Requiring signature property verification such as timestamps. Signatures will not be valid if this key is true and timestamp verification does not succeed.
- Limiting the number of nested verification sessions to prevent looping.
- Limiting the amount of time the signing time can be after the validation time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `bReqSigPropVerification` | bool    | (v 7.0) Defaults: 0  
`Path: <security root>\cASP\cASPKI\cVerify`  
Indicates whether signature property verification must succeed or not. If it is required and fails, the signature is not validated. As of 8.0, the only property used is the timestamp URL. |
| `iMaxVerifySession` | int     | (v8.0) Default: 5  
`Path: <security root>\cASP\cASPKI\cVerify`  
Indicates the maximum number of nested verification session allowed. This is used to prevent the application from going into infinite loop verifying the OCSP and/or CRL signer certificates caused by incorrect OCSP and/or CRL certificate setup. |
| `iMaxClockSkew` | int     | (v 7.0) Default: 65 (minutes)  
`Path: <security root>\cPubSec`  
The maximum difference in minutes the signing time is allowed to be after the validation time for which the signature can still be valid.  
PubSec verifies that a document isn’t signed in the future by looking at the verifier’s system time and the time embedded in the signature dictionary. Whenever time comes into picture, there is always the possibility that the signer and verifier’s times are out of sync. MaxClockSkew accommodates such differences. |

5.3.5 Document Status Dialog (deprecated)

**Note:** These preferences are not in the security directory and are deprecated after 7.x.

For application versions prior to 8.0, these preferences control whether the application displays the document status dialog when a user opens a certified document. The dialog contains a **Do not show this dialog next time this document is opened** checkbox allows users to turn the dialog off for the currently viewed document when it is next opened.

The application stores a list of document IDs, and each ID is associated with the user’s choice and the last certifying signature status. When the application matches an opened document with an ID in the registry, The application respects the choice as long as the certifying signature’s validity status has not changed since the last time the document was opened.

Administrators can turn off the document status dialog for specific documents by distributing a preference which contains the signature status and suppress status for particular document IDs.
5.3.6 Examine Document

The Examine Document dialog box identifies hidden document information that might pose a risk to the integrity of security and signature workflows. Found content is listed and linked to in the Examine Document pane. Users can click on a link to view the content and check/uncheck items to mark them for removal. Checked items are removed when the user selects the **Remove** button. The following options are available:

- Examining a document each time it is closed.
- Examining a document each time it is emailed.

### Table 63 Registry preferences: Document status dialog

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sSuppressListOrder</td>
<td>string</td>
<td>(v 7.0) Default: null Path: DocumentStatus An indexe of the most recently accessed documents. This is initialized as the string &lt;abc...zABC...z&gt;. As certified documents are opened, an entry for them is created in cSuppressStatusDocList keyed by the first letter in this string. That letter is then moved to the end of this string (bcd...zABC...za).</td>
</tr>
<tr>
<td>cSuppressStatusDocList</td>
<td>cab</td>
<td>(v 7.0) Default: null Path: DocumentStatus Contains a list of entries for previously opened certified documents. Each entry here is a subkey of the form c&lt;some letter (a-zA-Z)&gt;), and each subkey contains iLastStatus and xDocID.</td>
</tr>
<tr>
<td>iLastStatus</td>
<td>int</td>
<td>(v 7.0) Default: null Path: DocumentStatus The certifying signature’s status the last time a document was opened.</td>
</tr>
<tr>
<td>xDocID</td>
<td>binary</td>
<td>(v 7.0) Default: null Path: DocumentStatus An application generated binary ID which is used to identify the document.</td>
</tr>
</tbody>
</table>

### Table 64 Registry preferences: Examine document

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAutoLaunchAtDocClose</td>
<td>int</td>
<td>(v 9.0) Default: 0 Path: &lt;security root&gt; Maps to GUI item: Preferences &gt; Document &gt; Examine document when closing document Automatically examines the document for hidden content when it is closed.</td>
</tr>
<tr>
<td>bAutoLaunchAtSendMail</td>
<td>int</td>
<td>(v 9.0) Default: 0 Path: &lt;security root&gt; Maps to GUI item: Preferences &gt; Document &gt; Examine document when sending document by email Automatically examines the document for hidden content when it is sent in an email.</td>
</tr>
</tbody>
</table>
Digital Signatures

Technical Details
Authoring Signable Documents
Signing and Certifying
Validating Signatures
Digital Signatures: Technical Overview

Acrobat’s digital signature capabilities allow document authors to set up a secure signing environment and create simple documents as well as complex forms with one or more fields. Authors can design documents with multiple signature fields each with unique behavioral characteristics and appearances. A signed field can lock other fields so that signed data can’t be changed, and authors can force certain signature fields to be a required part of a workflow. Attention to signature field design and configuration can help you make the document “do the right thing” when document recipients get it as well as control what they can and cannot do with a document.

What authors and signers can do is subject to administrator control. Applications can be configured pre or post-deployment to behave as needed during signing and signature validation workflows. Once you’ve read this chapter, refer to Chapter 5, “Registry and plist Settings”.

The following sections provide an overview of Adobe’s digital signatures implementation:

- “Signature Technical Details” on page 132
- “How Signing Happens” on page 133
- “Signatures and Permissions” on page 135
- “Differences Between Signature Types” on page 138
- “Standard PDF and Static XML Forms” on page 138
- “Dynamic XML Forms” on page 140

6.1 Signature Technical Details

Digital signature support in PDF uses two base technologies to check if:

- The document has been changed since it was signed (verification or document integrity check)
- The signers can be authenticated (validation or signature validation check).

If either of those two checks fail, then a variety of means are used to bring the situation to the attention of the person opening the signed document.

For the document integrity check, the base technology computes a message digest over the bytes of the PDF file, encrypts the digest with the private key, and stores that version in the file. During signature validation, the digest can be decrypted with the corresponding public key and checked against the stored value. Once a PDF is signed, subsequent attempts to change the file are either prohibited or restricted to become incremental updates to the original file. Since incremental updates are always additional bytes added to the end of the file, the message digest over the original file can still be computed. The Acrobat Save As function typically writes a new file, but if the file has been signed, Save As either does an incremental save or if there have been no changes since it was signed, it does a copy (bytes) file operation. This supports the common practice of copying a file using and does not invalidate the signature.

For the signature validation check, digital IDs following the ITU-T X.509 v3 standard are used to identify the signer. The ID’s certificate chain is included in the PKCS#7 object within the PDF file. When the
signed file opens, the last certificate in the chain is authenticated against a certificate authority stored on the machine. If the chain cannot be authenticated, then the authenticity of the signer is reported as unknown. Self-signed certificates provide their own certificate authority. These will also be trusted if a user directs Acrobat to include it in the trusted identities list the first time it is encountered.

The modification, detection and prevention technology (MDP) has been developed to allow more control over what can, should, and must not, change after a given signature has been applied. In some cases, an author can specify which prohibited changes invalidate the signature.

6.1.1 How Signing Happens

To understand the ensuing discussion about what is signed, it is helpful to have a high level understanding of what a byte range is as well as the signing process.

A byte range is an array of four numbers (named ByteRange) which is stored alongside the signature value. The four numbers are actually two byte sequences that define what is hashed. The first number in each pair is the offset in the file (from the beginning, starting from 0) of the beginning of a stream of bytes. The second number is the length of that stream.

Between the end of the first sequence and the beginning of the second is the location for the value of the /Contents key, which contains the actual signature value. For example, if a hash is calculated for bytes 0 through 839, and 960 through 1200, it would look like Figure A.2.

The signing process is as follows:

1. The document is converted to a stream of bytes.

2. The entire PDF file is written to disk, with a suitably-sized space set aside for the signature value which is populated with temporary, worst-case (largest) values in the ByteRange array.

3. The document is digested with a hash algorithm such as SHA-256, and a hash of the entire file is computed using the bytes specified by the real ByteRange value.
   
   Note: Except for the signature hole (signature value bytes), Acrobat always computes the hash for the entire PDF file, starting from byte 0 and ending with the last byte in the physical file. In the byte range, the first value must be 0, the last must be the offset of the last byte in the file and the other numbers identify the hole in which the signature will be placed.

4. The temporary ByteRange array is overwritten using the correct values. Because the byte offsets must not change, extra bytes following the new array statement are overwritten with spaces.

5. The hash value is encrypted with the signer’s private key using a supported RSA or DSA signature algorithm.

6. A signature object is generated. By default, it is a PKCS#7 object.

7. The signature object is placed in the file on disk, overwriting the placeholder /Contents value. Any space not used for the signature object is overwritten with spaces.

8. The PDF file is reloaded in Acrobat to ensure that the in-memory version matches the disk version. Registry/plist-level signature creation preferences can control the type and strength of signature as well as what conditions will cause signing to fail. The available options allow you to configure
signing workflows for varying degrees of security. Almost every phase of the signing workflow is configurable. Moreover, for any certificate in a signature creation workflow, revocation checking may be customized as described in Chapter 7, “Certificate Processing”.

**Tip:** For a one page guide, see “Signature Creation Workflow” on page 313.

### 6.1.2 What’s Different with XML Forms?

Signatures can vary with respect to the document components that are included in the message digest. That is, in certain scenarios some information may be included or excluded in the signed byte range. This is true for XML form documents where the signer can omit or include data by using a particular signing method or workflow.

XML form documents are usually forms created by Adobe® LiveCycle® Designer software. The forms may have a flow layout (dynamic) or a fixed layout (static). XML forms are represented as XML until they are saved in the PDF format for later consumption by Acrobat or Adobe Reader.


In many signing workflows, XML forms saved in a PDF format are signed as any other PDF. However, there can be differences. For example, Rendering components are never included in the signed byte range. It is also possible to use a data signatures to include form data but exclude everything else from the signed byte range.

XML form components are generally treated as follows:

- **Template components:** These components are part of the signed byte range except for data signatures. Template components consist of form field definitions (the actual form fields are in the <form> DOM).

- **Rendering components:** These components are part of the signed byte range except for:
  - Data signatures.
  - Certification signatures (both invisible and visible) in flowed, dynamic XML forms.

Form rendering components are like marks on a page—essentially what prints and what the user sees when they sign—a document “snapshot”. For most types of signatures and documents this snapshot is important because it provides a high level of assurance that Acrobat signs what the user saw when they signed. For example, form time may display in a document, but the time is “captured” when the document is signed. Not signing these components can significantly reduce file bloat in a complex file.

- **Data:** Existing data in a file at the time of signing is always part of the signed byte range. It is also possible to only sign the data with a data signature and not include anything else.

Signing data provides a “baseline” so that it is possible to determine what data was in the document at the time of signing. If a document has been certified with no changes allowed, then any post-certification data changes will make the signature status be invalid. It is more common that a document is certified with the option to permit form fill so that the entire document is protected while allowing users to enter new data. In this case, the signature remains valid, the change is noted in status, and users can determine the data’s state at the time of certification by choosing **View Signed Version**.

**Note:** LiveCycle Designer allows authors to create forms where only the data is signed and nothing else. Data signatures leverage a different underlying technology. For example,
when a workflow includes signing with a data signature, the data is sent as XML or captured in an XML data package (XDP) and sent to a server. The PDF format is not used, no pkcs#7 object is created, and so on. XML digital signatures conform to the W3C XML-Signature standard (http://www.w3.org/TR/xmldsig-core/).

6.1.3 Signature Validation

Signature validation may vary with the signature format. For example, a PKCS#1 signature is smaller than PKCS#7 signature and is thus restricted in the amount of information that it can contain. However, the relationship between signature validity and its signing certificate validity is the same.

For details, see Chapter 7, “Certificate Processing”.

6.1.4 Signature Format

PKCS#1 was the only format available to Acrobat version 4 when digital signatures was introduced. Acrobat 5 introduced the more versatile PKCS#7 format, although PKCS#1 remained the default setting. In Acrobat 5, the user had the option of setting the format via a preference setting. Acrobat 6 changed the default format to PKCS#7 and the user interface was removed for modifying the setting. However, Acrobat maintains the ability to process both formats so that users can force the use of PKCS#1 via the use of JavaScript commands.

6.1.5 Signatures and Permissions

Because signing is always associated with permissions, you should understand how to set permissions and the consequences of your decisions with respect to signature workflows. Acrobat provides the following features:

- Setting permissions at the field as well as the document level.
- Setting permissions at document/field authoring time as well as at signing time.

Acrobat provides two signature types: approval and certification. These two names appear in the product’s user interface and are used throughout the end user documentation. Acrobat users can sign with either kind of signature. Acrobat authors also have the capability of Reader enabling a document so that Adobe Reader users can apply an approval signature (they can’t certify). Reader enabling a document actually signs it with an invisible signature. These signature types use the same technology and differ only in their permissions.

**Tip:** For a one page guide, see “Signatures, Permissions, and the PDF Language” on page 315.

At the PDF language level, signatures vary in that they may or may not be hooked up to one or more permission handlers. Permission handlers are specified in the PDF by various dictionaries are the result of document author and signer actions (Table 1). Thus, a signature may have one or more of the permissions or restrictions:

- **DocMDP:** An entry which exists as the result of applying a certification signature and which sets document level restrictions.
- **FieldMDP:** An entry which exists as the result of a document author setting permissions (or causing them to be set as the result of a signing action) and which set field-level restrictions.
- **UR3**: An entry which exists as the result of a document author using Acrobat to enable document features for an Adobe Reader (such as signing).

### Table 1  Applied permission handlers as a result of user actions

<table>
<thead>
<tr>
<th>Adds field restrict.?</th>
<th>Reader enabled?</th>
<th>Signature type</th>
<th>Permission handlers</th>
<th>Message digest contents</th>
<th>Signing Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>No</td>
<td>Approval</td>
<td>Default handler</td>
<td>Everything</td>
<td>Acrobat</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Approval</td>
<td>Certification</td>
<td>DocMDP</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Approval</td>
<td>Default handler</td>
<td>Everything</td>
<td>Reader</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Approval</td>
<td>FieldMDP</td>
<td>Everything</td>
<td>Acrobat</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Visible certification</td>
<td>FieldMDP, DocMDP</td>
<td>Everything (except rendering for dynamic XML forms)</td>
<td>Acrobat</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Invisible certification</td>
<td>FieldMDP, DocMDP</td>
<td>Everything (except rendering for dynamic XML forms)</td>
<td>Acrobat</td>
</tr>
<tr>
<td>N/A</td>
<td>Yes</td>
<td>Reader enabling</td>
<td>UR3</td>
<td>Everything (except rendering for dynamic XML forms)</td>
<td>Acrobat</td>
</tr>
<tr>
<td>N/A</td>
<td>No</td>
<td>Yes</td>
<td>Data signature</td>
<td>N/A</td>
<td>Form data only</td>
</tr>
</tbody>
</table>

Signatures can be used to ensure that a signed document has not been altered since it was signed (it has “document integrity”). A document integrity check can be automatically performed by the application or manually by the signature validator. The signer’s identity may also be verified by checking the signer’s certificate data that is embedded it the signature. When both document integrity and the signer’s identity are verified, non-repudiation is also applicable.

- **Approval signature with the default handler:**
  
  An approval signature is a byte range signature over all PDF file contents. Approval signatures take a visual snapshot of the document at the time it was signed and thus provide a high level of document integrity:
  
  - If there have been incremental changes to a document after an approval signature has been applied, the signature’s status will be valid with a warning or invalid depending on the nature of the change.
  
  - If the signed contents have not been altered and there have been no incremental changes since the approval signature, the signature’s status will be valid.

- **Certification signature (DocMDP):**
  
  An certification signature is a byte range signature over all PDF file contents but with additional document protection rules (DocMDP rules) that determine which document properties are not permitted to change. A certification signature ensures that a document has not been modified contrary to the author’s intent. When a certification signature is applied, Acrobat prevents users from executing operations that would violate the document protection rules.
  
  - If there have been incremental changes to a document after an approval signature has been applied, the signature’s status will be valid with a warning or invalid depending on the nature of the change.
  
  - If any of the signed contents have been altered in ways not permitted by the signer OR if any document protection rules have been violated, the signature’s status will be invalid.
If the signed contents have not been altered AND no document protection rules have been violated, a certification signature’s status will be valid.

**Signatures in fields with restrictions (FieldMDP):**

Either an approval or certification signature may be used in a field with restrictions defined by FieldMDP. These signatures are also byte range signatures over all PDF contents, but with additional form field protection rules. An author can specify which document form fields are not permitted to change after a certain signature field is signed, thereby locking one or more fields and ensuring that form fields have not been modified contrary to the author’s intent.

- If any of the signed contents have been altered OR if any form field protection rules have been violated, a signature’s status is invalid.
- If the signed contents have not been altered AND no form field protection rules have been violated, the signature status is valid.

**Signatures created when Reader enabling (UR3):**

Adobe Reader users can not sign any PDF. However, an Acrobat author or an ALRMS user can create a PDF that “turns on” this capability for an Adobe Reader user. Acrobat adds Reader extension rules to the document and then signs them. When the document opens in Adobe Reader, the signature is validated, and the Reader extension rules are processed by Adobe Reader to unlock functionality that would otherwise be locked. Documents that activate Reader’s hidden signing capability are called “Reader-enabled.” Adobe Reader users can only apply an approval signatures. These signatures are a byte range signature over all the PDF content, but the signing is the result of embedded rules that determine which normally prohibited document functions are permitted. These rules ensure that document functionality has not been executed contrary to an application’s intent.

- If any changes to the signed contents violate the UR3 rules (discovered by the document modification analysis), a Reader-enabled signature’s status is invalid.
- If the signed contents have not been altered AND no document changes violate the UR3 rules (discovered by the document modification analysis), the signature’s status is valid.

### Table 2 Signature type support in XML forms

<table>
<thead>
<tr>
<th>Sig. Type</th>
<th>Static XML forms</th>
<th>Dynamic XML forms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.x</td>
<td>7.x</td>
</tr>
<tr>
<td>Approval</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Certification (DocMDP)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Field protection (FieldMDP)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Reader enabling (UR3)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

#### 6.1.5.1 Locking a Document During Signing

New for 9.0 is the ability to lock the document when applying an approval signature. Older viewers—earlier than 9.0—ignore and do not process the additional approval signature FieldMDP Lock state. To increase compatibility, Acrobat sets FieldMDP to lock all fields when the document is locked. Consequently, older viewers consider all fields locked when the document lock is applied using Acrobat 9.0. However, adding annotations is still possible.
Lock Document after Signing only appears when these three conditions are met:

- The author of the document did not apply any locking rules to the field (no FieldMDP rules are present).
- Seed values are either absent or they do not prevent locking.
- The field being signed is the last, unsigned field in the document.

### 6.2 Differences Between Signature Types

The following sections provide details about how Acrobat handles signing tasks with respect to a particular signature and document type. The following questions are answered:

- What is digested and signed?
- How is the signature verified?
- How are in-memory changes detected?
- How are prohibited in-memory changes prevented?

#### 6.2.1 Standard PDF and Static XML Forms

##### 6.2.1.1 Approval Signature

**What is digested and signed?**

The PDF file’s full byte range is saved to disk, digested, and signed.

**How is the signature verified?**

The byte range is digested and compared to the digest stored in the signature value. If they match, the signature is valid. If they do not match, the signature will be invalid. If the digests match, but there have been incremental changes since the signature was applied, these changes are analyzed and the result may be that the signature is considered invalid, valid, or valid with a form modified warning (blue i icon).

**How are in-memory changes detected?**

When a change is made, a notification is sent to the signature to put it into an unverified state.

When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

**How are prohibited in-memory changes prevented?**

An approval signature does not prohibit any in-memory document changes. However, any changes made after an approval signature will cause the signature status to indicate that changes have been made.
6.2.1.2 Certification Signature (DocMDP)

What is digested and signed?

The PDF file’s full byte range is saved to disk, digested, and signed (including the rule(s) data in DocMDP).

Note: For backward compatibility with Acrobat 6.x, a PDF object digest is also included.

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature value. If these digests do not match, the certification signature is invalid.

A run-time analysis of document components compares the opened and signed document versions. Changes that violate the DocMDP rule specified in the signed version of the document will cause the certification signature will be invalid; otherwise, the certification signature will be valid.

Note: In a static XML form, the run-time analysis is only done on PDF level components. In a dynamic XML form, the run-time analysis is done on XML packets and PDF-level components.

How are in-memory changes detected?

When a change is made, a notification is sent to the signature to put it into an unverified state.

When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

How are prohibited in-memory changes prevented?

Acrobat/XML form permissions are used to prevent changes that would invalidate the certification signature.

6.2.1.3 Signature with Field Restrictions (FieldMDP)

What is digested and signed?

The PDF file’s full byte range is saved to disk, digested, and signed (including the field rule(s) data in FieldMDP).

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature. If these digests do not match, the signature will be invalid.

A run-time analysis of form field components compares the opened and signed document versions. Form field changes that violate the FieldMDP rules specified in the signed version of the document will cause the FieldMDP signature to report an invalid state; otherwise, the FieldMDP signature will report a valid state.

Note: In a static XML form, the run-time analysis is done on PDF level form fields. This is in contrast to a dynamic XML form where the run-time analysis is done on XML form fields.
How are in-memory changes detected?

When a change is made, a notification is sent to the signature to put it into an unverified state.

When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

How are prohibited in-memory changes prevented?

Acrobat permissions are used to prevent changes that would invalidate the FieldMDP signature.

6.2.1.4 Signatures Created When Reader Enabling (UR3)

What is digested and signed?

The PDF file's full byte range is saved to disk, digested, and signed (including the Reader extension usage rules in UR3).

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature value. If these digests do not match, the signature will be invalid.

A run-time analysis of document components compares the opened and signed document versions. Changes that violate the UR3 rule specified in the signed version of the document will cause the signature to report an invalid state; otherwise, the signature is considered valid.

How are in-memory changes detected?

No effort is made to detect in-memory changes.

How are prohibited in-memory changes prevented?

Acrobat permissions are used to prevent changes that would invalidate the signature.

6.2.2 Dynamic XML Forms

6.2.2.1 Approval Signature

What is digested and signed?

The PDF file's full byte range is saved to disk, digested, and signed.

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature value. If they match, the signature is valid. If they do not match, the signature will be invalid. If the digests match, but there have been incremental changes since the signature was applied, the signature will show a warning.

How are in-memory changes detected?

When a change is made, a notification is sent to the signature to put it into an unverified state.
When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

**How are prohibited in-memory changes prevented?**

An approval signature does not prohibit any in-memory document changes. However, any changes made after an approval signature will cause the signature status to indicate that changes have been made since the signature was applied.

### 6.2.2.2 Certification Signature (DocMDP)

**Note:** For signed dynamic XML forms in Acrobat 8.0, everything is digested and signed for visible certification signatures. Everything but rendering components are digested and signed invisible certification signatures. For Acrobat 9.0, the rendering components are never signed.

**What is digested and signed for certification signatures?**

The PDF file’s byte range (including the form data but NOT including form rendering components) is saved to disk, digested, and signed (including the rule(s) data in DocMDP).

**Note:** For more information, see “What’s Different with XML Forms?” on page 134.

**How is the signature verified?**

The byte range is digested and compared to the digest stored in the signature value. If these digests do not match, the certification signature will be invalid.

A run-time analysis of document components compares the opened and signed document versions. Changes that violate the DocMDP rule specified in the signed version of the document will cause the certification signature to report an invalid state; otherwise, the certification signature will report a valid state.

**Note:** In a static XML form, the run-time analysis is only done on PDF level components. In a dynamic XML form, the run-time analysis is done on XML packets and PDF-level components.

**How are in-memory changes detected?**

When a change is made, a notification is sent to the signature to put it into an unverified state.

When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

**How are prohibited in-memory changes prevented?**

Acrobat permissions are used to prevent changes that would invalidate the certification signature.

### 6.2.2.3 Signature with Field Restrictions (FieldMDP)

FieldMDP rule(s) can apply to an approval or a certification signature.
What is digested and signed?

- For approval signatures with FieldMDP rule(s), the full byte range of the PDF file, including document snapshot and including the FieldMDP rule(s) data is saved to disk, digested and signed.

- For certification signatures with FieldMDP rule(s), the full byte range of the PDF file, excluding a document snapshot and including the DocMDP and FieldMDP rule(s) data is saved to disk, digested and signed.

Note: In Acrobat 8.0, rendering components were not digested for invisible certification signatures.

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature. If these digests do not match, the FieldMDP signature will be invalid.

For certification signatures, the certification signature is first verified as described in “Certification Signature (DocMDP)” on page 139.

A run-time analysis of form field components compares the opened and signed document versions. Form field changes that violate the FieldMDP rules specified in the signed version of the document will cause the FieldMDP signature to report an invalid state; otherwise, the FieldMDP signature will report a valid state.

Note: In a dynamic XML form, the run-time analysis is done on XML form fields. This is in contrast to a static XML form document where the run-time analysis is done on PDF level form fields.

How are in-memory changes detected?

When a change is made, a notification is sent to the signature to put it into an unverified state.

When validation is requested (which can be accomplished by clicking the signature field), the signature verification process is executed, updating the signature state.

How are prohibited in-memory changes prevented?

Acrobat form permissions are used to prevent changes that would invalidate the FieldMDP signature.

6.2.2.4 Signatures Created When Reader Enabling (UR3)

What is digested and signed?

The full byte range of the PDF file, excluding document snapshot and including the UR3 rule(s) data is saved to disk, digested, and signed.

How is the signature verified?

The byte range is digested and compared to the digest stored in the signature value. If these digests do not match, the signature will be invalid.

A run-time analysis of document components compares the opened and signed document versions. Changes that violate the UR3 rule specified in the signed version of the document will cause the signature to report an invalid state; otherwise, the signature will report a valid state.
How are in-memory changes detected?
No effort is made to detect in-memory changes.

How are prohibited in-memory changes prevented?
Acrobat permissions are used to prevent changes that would invalidate the signature.

XML digital signatures conform to the W3C XML-Signature standard (http://www.w3.org/TR/xmldsig-core).

The signature information is in the XML form datasets packet which is part of the PDF file. The payload is currently XML form data or a subset thereof which is defined by the XML signature definition. The payload is transformed according to the XML signature definition (there are many potential options) and then digested and signed.

There is no byte range in XML signatures. The payload is transformed according to the XML signature definition and then digested. It is then compared with signature value extracted (decrypted) from the signature. If they match, the signature is valid. If they do not match, the signature will be invalid. If the data has been modified (even incrementally) since it was signed, the signature is invalid.

Data signatures do not prohibit any in-memory document changes. However, any changes (incremental or otherwise) to the signed data since the signature was applied will result in an invalid signature.
Certificate Processing

Acrobat products’ certificate processing conforms to RFC 3280. Whenever an Acrobat product uses a certificate in a signing workflow, its processing includes these basic steps:

- **Path Discovery: Building the Chain**: The application builds the certificate chain by mapping the issuer name of the subordinate certificate to the subject name of the certificate higher up the chain.

- **Path Validation: Checking a Chain’s Health**: After the chain(s) is constructed, certificate health is checked for each link (certificate) in the chain. This process verifies it is signed with a valid signature, the constraints aren’t violated, that it has not expired, etc. For example, name, policy, and path length constraints are often imposed by one or more intermediate (ICA) certificates in a chain. Some checks include:
  - **Process basic fields**: Check for basic certificate fields and field values such as Signature, Validity, Subject, etc.
  - **Process field extensions**: Continue processing based on certificate data and application settings:
    - Certificate field extensions may contain data that controls application behavior. For example, if KeyUsage is specified, is the certificate being used in the correct context? Was it constructed with the intention of being used for signing and encryption?
    - Application settings may require using or ignoring certificate field extension data. For example, IgnoreBasicConstraints and RequireAKI can be used to override default certificate processing.
  
  **Note**: Path validation is subject to customization via application preferences. For details, see Chapter 5, “Registry and plist Settings”.

- **Establishing Certificate Trust**: The application examines the chain for a trust anchor. If an anchor is not found, the process exits and no revocation checking occurs.

- **Certificate Revocation Checking**: The revocation checking type and order occurs as described in “Revocation Checking Order of Precedence” on page 152.

  Application settings control when checks occur, the type of checks, and whether the user-initiated action will succeed or fail if the certificate is revoked or the check fails.

  **Note**: While most deployments will use the application’s default settings, much of the application’s certificate processing behavior is under administrative control. You can configure the registry so that certificates that aren’t RFC 3280-compliant can still be used, you can apply a requirement to a specified part of a chain (a scope or “section”), and you can fine-tune revocation checking for the end users certificate as well as any certificate involved in the process. For details, see Chapter 5, “Registry and plist Settings”.

### 7.1 Path Discovery: Building the Chain

Before Acrobat can get to the revocation check process and finish validating the certificate, it must first find and build a certificate chain that ends in a trusted root. The certificate graph builder service
provider interface (SPI) provides the path discovery, chain building, and path validation services. The default behavior of Acrobat products adheres to RFC 3280, RFC 2560, and other relevant standards.

**Note:** For signature workflows, at the very minimum, the signer’s certificate is included in the signature dictionary (SigDict) within the PDF file. This certificate is either in the top level of the SigDict in PKCS #1 signature or in the contents sub-dictionary of a PKCS #7 signature. Once Acrobat has decoded the signer’s certificate, it begins the chain building process by looking for the parent (issuing) certificate. It continues to search until it finds a self-signed certificate (a certificate without a parent because it was issued by itself).

Acrobat searches for parent certificates from the following locations:
- Certificates embedded in the PKCS#7 object in the SigDict (for signature workflows only).
- Any server listed in the `cAdobe_ChainBuilding` registry or plist entry.
- The Acrobat Address Book which the user accesses via the Manage Trusted Identities dialog.
- The Windows Certificate Store (Windows only).

Acrobat matches a certificate based on the following criteria:
- The Subject Key Identifier (SKI) in the found certificate matches the Authority Key Identifier (AKI) in the child certificate
- If the AKI is missing from the child’s certificate, then the Subject Distinguished Name in the found certificate matches the Issuer Distinguished Name in the child certificate

Acrobat continues this process in an iterative manner until all possibilities of finding a match have been exhausted.

As Acrobat builds the chain, it compares the information it finds in the certificate with that in the registry to see if there is a match. If there is a match, application settings override the application’s default behavior. Administrators should refer to Chapter 5, “Registry and plist Settings” to learn more about what is subject to customization.

**Figure 1 Simple certificate chain**

In many cases, the chain is a simple, linear path from the end entity (e.g. a signer using Acrobat) to the trust anchor or root. The root may or may not be the trust anchor, but in either case the end user should
explicitly decide which certificates will be trusted. A conservative approach is best, since trusting a root or some other certificate high up in a chain result sin trusting all certificates which chain up to that certificate.

7.1.1 Cross-Certificates

Acrobat products also discover and validate chains which incorporate cross-certificates. A cross-certificate is one issued by a CA that signs the public key for another CA’s certificate. Cross-certificates allow organizations to provide their end users with certificates that enable establishing trust between two or more CAs. Cross certificates enable chain building to your own trust anchor when you don’t (or can’t) trust the signer’s trust anchor.

In practice, one methodology includes a CA (such as CA1 Incorporated) sending an unsigned certificate and a signing request to another CA (such as CA2 Incorporated). CA2 signs the certificate and sends it back to CA1. CA2 is therefore the issuer and CA1 is subject. If the two CAs do this for each other, the issuer for CA1’s certificate is CA2, and the issuer for CA2 is CA1, and each organization will be able to validate the other’s signatures.

In these distributed trust architectures (also called mesh PKIs) where trust occurs across more than one certificate authority, cross-certification may be unilateral or bilateral as well as forward or reverse.

Note: Instruction on PKI infrastructure types and their setup is beyond the scope of this document. The purpose here is to describe Acrobat’s behavior during signature validation when that validation relies on building a certificate chain that uses cross certificates.

Cross Certificate Lifecycle

At a high level, cross-certificates are created and used as follows:

1. CA1 sends a certificate and a certificate signing request to CA2.
2. CA2 signs the certificate and sends it back. Note that the public key for the CA2 root (a self signed certificate) and the cross certificate are identical.
3. The cross certificate is installed on CA1 client machines. The cross certificate chains up to root CA1 and it’s issuer is CA1.
4. Root CA1 is explicitly trusted as a trust anchor.
5. If CA2 needs to validate signatures from CA1, the process is repeated in reverse so that bilateral (or bidirectional) cross certification can be set up.
Cross Certificate Usage

Cross-certification is useful in Acrobat workflows because it binds unrelated CAs so that document recipients can validate signatures from signers outside of their organization.

For example, when an employee of CA1 received a PDF signed by a certificate issued under CA2, opening it in Acrobat enables the possibility of signature validation. If the CA1 employee has installed the cross certificate and it chains up to their trust anchor, then Acrobat can build and validate the certificate chain. Naturally, if the other certificate tests pass, then the signature will be deemed valid.

Let’s take a hypothetical use case as an example: Adobe establishes a unilateral cross certification relationship with Bank.com. In this case, there is no reciprocal relationship. Adobe sends a certificate to Bank.com for signing and receives the signed certificate back. The cross certificate is included in Acrobat so that Acrobat users can validate all signatures created with certificates issued by Bank.com or by ICAs whose certificates chain up to Bank.com. However, Bank.com clients do not have to automatically accept all certificates that chain up to the Adobe root.

Now when a Bank.com user signs a document and sends it to another Acrobat user, cross certificates are used during chain building as follows:

1. When the Bank.com signer signs, Acrobat builds a signature object which contains all the certificate information in the signer’s chain. The data includes the root CA’s (Bank.com’s) public key.

2. When the document recipient tries to validate the signature, Acrobat builds the chain from signature object data, and the chain is built from the signer’s end entity certificate to the root (Figure 3).

   Tip: In order for the signature to be valid, Acrobat must be able to build a chain to a certificate the validator explicitly trusts.
3. Acrobat tries to build additional chains and discovers that the subject cn on one of it's certificates (the cross certificate signed by Bank.com) matches the issuer cn of the signer’s root certificate (the Bank.com root).

4. Acrobat builds another chain from the signer’s end entity certificate, through the cross certificate, and to the trusted root on the validator’s machine (the Adobe root).

5. Because a chain was built to a certificate the signature validator trusts, the signature is marked as valid.

**Figure 3 Cross certificate chain**

7.1.1.1 Chain Building FAQs

**Does Acrobat handle chain building differently when cross certificates are used?**

No. Acrobat builds as many chains as it can. If the issuer cn of one certificate matches the subject cn of another, the application simply tries to build another chain. If the cross certificate chains to a trust anchor, the signature may be valid.

**What happens if a cross certificate has an issue date after the signature date?**

Cross certificates must be valid at the time of signing. Chains that include a cross certificate that was issued after a signature was created will not be used.

**How does Acrobat handle cross certificate rollovers?**

It doesn’t for the same reasons as above. The *Valid From* date of the cross certificate must be prior to the signing date.

**Does the cross certificate need to be a trust anchor?**

No, but it must chain up to a trust anchor.
Can a signer embed revocation information for the cross certificate as well in the signature?

Yes, if they have access to the certificate. The signer can embed revocation information for any certificate on their machine, but cross certificates usually resides on the signature validator’s machine.

Does the presence of a cross certificate change the way revocation checking happens?

No. Revocation checking occurs after chain building and path validation and the entire process happens in the same way regardless of whether a cross certificate exists in the chain.

Can I examine the chains that Acrobat builds?

Yes. The Certificate Viewer displays all the chains. Open the viewer by right clicking on any signature and choosing Signature Properties. The Show Certificate button opens the viewer.

- Check Show all certification paths found to display all the chains.
- Highlight a certificate and examine the Details tab to learn about certificate data.
7.2 Path Validation: Checking a Chain’s Health

After a trust anchor is found, Acrobat proceeds to check the status of each certificate in the chain. Beginning with the top most certificate in the chain (the certificate farthest from the signer’s certificate) Acrobat verifies each certificate is valid. Checked items include but are not limited to:

- Is the certificate within its validity period?
Certificate Processing

Establishing Certificate Trust

- Is the signature over the certificate valid (has the certificate been tampered with)?
- Are the various constraints such as basic constraints, name constraints, path length constraints, and so on being met?
- Is the certificate being used in accordance with its key usage settings?

A failure in either decoding the certificate (i.e. the certificate was malformed) or in any of the other checks once the certificate has been decoded will prevent the final step of revocation checking, but Acrobat continues to check until all of the certificates in the chain have been examined.

7.3 Establishing Certificate Trust

Once the chain has been constructed, Acrobat then determines if any of the certificates in the chain have been assigned the role of a trust anchor. Starting from the bottom of the chain (the signer’s certificate), Acrobat first checks the user’s trusted identities list to see if the trust flag has been manually set. If trust cannot be established from the certificates in that list AND the user has selected the Trust all root certificates in the Windows Certificate Store preference, Acrobat checks the installed root certificates in Windows Certificate Store. Acrobat continues this process for each certificate in the chain until trust is either established or all of the certificates in the chain have been checked.

If trust cannot be established Acrobat stops trying to validate the signature.

7.4 Certificate Revocation Checking

After the chain is built, validated, and a trust anchor is found, Acrobat determines whether any of the certificates in the chain have been revoked. Of all of the steps this is the most convoluted and requires the most amount of computing overhead.

The first thing Acrobat has to do is decide if it needs to do revocation checking at all, so Acrobat looks in the registry/plist to determine if it needs to do any revocation checking as well as wether that check must succeed for the signature to be considered valid. Both of these are subject to user or administrator configuration.

Revocation checking begins with the bottom most certificate in the chain (the signer’s certificate). It continues to check each certificate successively by moving up the chain. Note that revocation checking does NOT occur for:

- Certificates found in the Windows Personal store, as these are under the direct control of the user.
- The root certificate in the chain, as these are not revokable by any standard means.
- A trust anchor or any certificate above a trust anchor.
- Self-signed certificates.

In any of the following steps a revocation response may be found. In the case where the response is considered invalid Acrobat must proceed in its quest to locate a valid response. The validity of the response is predicated on two things. First, is the response within its validity period based on the time
being used to perform the revocation check (i.e. current time vs. signing time)? Second, was the response issued by the appropriate certificate authority? If the response was issued by someone other that the CA that issued the certificate being checked or its delegate, the response is invalid. However, once a valid response is located Acrobat ceases to look further.

Acrobat will look for a valid revocation response from the following locations and in the following order:

1. The PKCS#7 object contained in the SigDict for an embedded OCSP response or embedded CRL.

2. The local CRL cache directory.

3. The registry for a custom certificate preference that directs Acrobat where to query for a response for that specific certificate.

4. The cAdobe_OCSPRevChecker registry entry to see if there are any global instructions for procuring an OCSP response.

5. The Authority Information Access (AIA) extension in the certificate for a URI from which an OCSP response can be procured.

6. The cAdobe_CRLRevChecker registry entry to see if there are any global instructions for downloading a CRL.

7. The CRL Distribution Point (CDP) extension in the certificate for a URI from which an CRL can be downloaded.

8. An LDAP lookup based on the DN of the issuer.

The first two items on the list are on the top because it is always more efficient to find a revocation response locally rather than having to go on to the internet. OCSP checks are done ahead of trying to download a CRL because OCSP processing is faster than having to download and parse a CRL. Finally, the LDAP lookup is a last, best effort at finding something after all of the normal methods have failed to return a valid response.

Almost all revocation checking behavior is subject to administrator customization through application settings such as the Windows registry. For example, revocation checking can be configured individually for each part of the workflow, including the check type, when it occurs, and what happens if a check can’t succeed and/or a certificate is revoked. For details, see the following:

- “Revocation Types” on page 152
- “Revocation Checking Order of Precedence” on page 152
- “Timestamp Revocation Checking” on page 153

### 7.4.1 Revocation Types

Acrobat uses one or more of its ASPKI service providers to specify available revocation checking methods. Access to specific providers is determined by customizable registry preferences. By default, Acrobat can make a request for the following check types in order to authenticate a public key:

- **Online Certificate Status Protocol (OCSP)**: OCSP defines a protocol for determining the revocation status of a digital certificate without requiring a CRL. Unlike CRL, OCSP obviates the need to frequently download updates to keep certification status lists current. Acrobat’s OCSP revocation checker adheres to RFC 2560.
Certificate Revocation List (CRL): CRL is a method that public key infrastructures use to maintain access to cached or networked lists of unexpired but revoked certificates. The list specifies revoked certificates, the reasons for revocation (optional), and the certificate issue date and issuing entities. Each list contains a proposed date for the next release. When a user asks for server access, the server allows or denies that access based on the CRL entry for that user. Acrobat’s CRL revocation checker adheres to RFC 3280 and NIST PKITS except for delta CRLs.

Other methods: Checking initiated through Windows Microsoft Crypto API (MSCAPI) can use either OCSP or CRL but is not limited to them.

7.4.2 Revocation Checking Order of Precedence

Tip: For a one page key, see “Revocation Checking Process” on page 316.

The revocation check type and order is subject to administrator control; however, the default behavior is as follows:

1. Get service providers: Make both OCSP and CRL service providers available. Configurable via the cRevocationCheckers registry preference. For details on configuring these settings, see “Registry and plist Settings” on page 75.

2. Check for embedded revocation information: Check if the signer embedded the revocation status in the signature; check OCSP first, then CRL. Configurable via iUseArchivedRevInfo.

3. Check local CRL cache: If embedded data is not used, check to see if a CRL is stored in the application’s local cache. For example, at C:\Documents and Settings\<user>\Application Data\Adobe\Acrobat\<version>\Security\CRLCache.

Check remote OCSP: iReqRevCheck preference specifies whether the check is required to succeed and what should happen if it doesn’t. The check occurs as follows:

- If there is a custom certificate preference, use those settings (e.g. look at iURLToConsult and use value in sURL).
- If there is no custom certificate preference, use registry settings in Adobe_OCSPRevChecker (e.g. look at iURLToConsult and use value in sURL).
- If there is no registry preference, use the AIA extension in the certificate.
- If no OCSP occurs or it fails, go to the next step and try to use a CRL.

4. Retrieve a remote CRL: iReqRevCheck preference specifies whether the check is required to succeed and what should happen if it doesn’t. The check occurs as follows:

- If there is a custom certificate preference, use those settings (e.g. if no sURL and sLDAP is set, search LDAP server).
- If no custom certificate preference, use Adobe_CRLRevChecker setting (e.g. if no sURL and sLDAP is set, search LDAP server).
- If no registry preference, use the CRLdp extension in the certificate.
- If all fail, revocation checking fails.

5. Stop checking if any response succeeds.
7.4.3 Timestamp Revocation Checking

Whether or not revocation checking occurs for timestamps depends on whether or not you are using secure time to verify signatures (Figure 5). If Secure time is not selected, then timestamps are effectively ignored.

![Figure 5 Timestamp UI preference](image)

Unless a timestamp is also timestamped (unlikely), revocation checking always occurs at the current time. The application simply asks: "Is this timestamp signature's certificate revoked now (at the verification time)." In all other respects, revocation checking for timestamps is similar to non-timestamp checks in that many of the default behaviors can be modified by changing the settings under Adobe TSPProvider.

Order of precedence for timestamps follows the non-timestamp order. However, because timestamps can be set via a field seed value, the order can be changed as follows (from most controlling to least controlling): Seed value > certificate > application preference. For example, when a certificate has an OID that says “use this timestamp server,” a signature field has a seed value that says “use this timestamp server,” and the application has a default timestamp server configured, the seed value takes precedence.

Unless otherwise configured, revocation checking occurs for timestamps as follows:

- The application ignores any timestamp time and uses the signer’s system time unless the verification time is set to the Secure time.
- If the revocation information is not embedded in the document, Acrobat goes online to check the status at the signer’s signing time.
- If the signature is timestamped and revocation information is embedded at the time of signing and:
  - The response has not expired; there is no need to check the status.
  - The response is expired, the application checks the status relative to the current time.

7.4.4 Revocation Checking Default Behavior

7.4.4.1 OCSP Default Behavior

The OCSP protocol specifies the syntax for server-client communication. OCSP provides a grace period to users with expired certificates so that they can have the possibility to access a server for certificate renewal.
Certificate Processing  Revocation Checking Default Behavior

Note: Further details regarding this protocol are outlined in RFC 2560. The following sections describe Acrobat’s implementation details of the OCSP based certificate revocation checking service that Acrobat’s Public Key Infrastructure Library (ASPKI) provides to its clients. Unless explicitly mentioned, ASPKI does not deviate from RFC 2560.

Whenever users access a server, OCSP simply asks for certificate status information. The server then responds with a status of Current, Expired, or Unknown. Acrobat’s ASPKI executes OCSP-based revocation checking by doing the following:

Obtaining an Authorized OCSP Responder

There are two methods of locating an OCSP responder:

- Application configuration in the registry either via a global setting or a custom certificate preference.
- Via the Authority Information Access (AIA) extension in the certificate. This extension includes a URI specifying an OCSP responder.

Note: Registry settings override certificate data. An administrator may specify an OCSP responder in the registry that value will override the AIA extension in the certificate. For details about revocation checking order, see “Revocation Checking Order of Precedence” on page 152.

Sending an OCSP Request to the Responder

By default, OCSP requests include a Nonce and are unsigned, but you can override this behavior with application settings. For details on configuring these application settings, “Registry and plist Settings” on page 75.

The format of a valid OCSP request is defined in RFC 2560. After obtaining an OCSP responder to use, ASPKI constructs an OCSP request for the certificate. If the responder is different from the responder outlined in the certificate’s AIA extension (if any), ASPKI attaches an OCSP service locator extension to the request. This allows for configurations where a particular OCSP responder routes OCSP requests to their intended responders.

OCSP Response Processing

Default response processing behaviors are subject to administrator modification via application settings. ASPKI processing steps include the following:

1. **Nonce verification**: If the response contains a Nonce, ASPKI verifies that the Nonce matches the one included in the request and rejects the response if this fails. ASPKI will not reject responses that do not contain any Nonce to allow supporting OCSP responders that sends out pre-generated responses.

2. **Response staleness checks**: Verifications follow section 4.2.2.1 of RFC 2560 closely. ASPKI verifies the revocation check time provided by the client is within the interval provided by the ThisUpdate and NextUpdate entries in the response. Clients can specify an upper limit for the time difference between ThisUpdate (the time at which the response is known to be valid) and the current time. This client specified limit applies only when ThisUpdate is earlier than the current time. The default value is 525600 minutes (1 year).

Note: RFC 2560 specifies that responses whose ThisUpdate entry is later than the current time should be considered unreliable. ASPKI, on the other hand, allows the client to set
the maximum clock skew that can be tolerated. This allows ASPKI to further process responses whose delivery might have been slightly delayed due to other factors such as network congestion.

ASPKI rejects the response if its NextUpdate entry is earlier than the current time. As indicated in section 2.4 of RFC 2560, the absence of this entry indicates that newer revocation information is always available from the server. In this situation, ASPKI only does the checks that are done to the ThisUpdate entry mentioned above to determine the staleness of the response.

3. **Responder authorization verification:** ASPKI follows section 4.2.2.2 of RFC 2560 when verifying whether a responder is authorized to provide revocation information for a certificate. While looking for the responder’s certificate, ASPKI searches the list of certificates embedded in the response, if any, and its directories. Verification fails only if all of the found certificates are not authorized. Clients can specify particular OCSP responders as authorized by trusting them for signing OCSP responses in their certificate directory (For an example, see “Customized Revocation Check Example” on page 160).

4. **Verifying responder’s signature:** Signature verification involves a revocation check for the responder’s certificate. Section 4.2.2.2.1 of RFC 2560 allows for this second revocation check to be ignored. The criterion for this are:
   1. Responder’s certificate is issued by the issuer of the certificate currently being checked.
   2. The responder certificate has OCSP-Signing as one of its extended key usages.
   3. The responder certificate has an OCSP-NoCheck extension. OCSP-NoCheck is sometimes embedded in the responder certificate as an extended key usage instead of a certificate extension. ASPKI currently accepts both extension placements.

   **Note:** You can also configure ASPKI to ignore this second revocation check if criterion 1 and 2 are met but not criterion 3. By default, if the responder’s certificate is trusted for OCSP signing but its revocation status is unknown, ASPKI will accept the responder’s signature as valid. The client can configure ASPKI to reject the signature in this event.

### 7.4.4.2 CRL Processing Default Behavior

**When is the revocation status of a public-key certificate (PKC) checked against a CRL?**

For all versions of Acrobat 6.0 and later, revocation checking will utilize a CRL if revocation checking is based on either the current time or the time of signing, and:

- The CRL is embedded in the signature content and is within its validity period (based on which time standard is being used).
- The CRL is cached locally and is within its validity period.
- A local copy can be obtained in real time via a distribution address listed in:
  - The preferences (Windows registry or Mac plist file).
  - A CRL Distribution Point (CDP) extension included within the PKC.

**When will a CRL be loaded/updated?**

For all versions of Acrobat 6.0 and later, an updated CRL will try to be obtained if either:

- The locally cached CRL is not within its validity period, or
The CRL is within its validity period but is not a valid indirect CRL or is otherwise corrupted.

**What is the caching behavior, if no CRL can be downloaded?**

For all versions of Acrobat 6.0 and later, only valid CRLs (those within the usage time frame and issued by the appropriate CA) will be used for the purpose of substantiating signature validity. If the cached CRL is not considered valid then Acrobat will not use it. If it cannot be updated because of connectivity issues, and there is no other mechanism for Acrobat to ascertain the revocation status, then the digital signature status will be problematic. The only exception is if the user explicitly deselects the **Require certificate revocation checking to succeed** preference.

**What is the caching behavior, if the CRL is corrupt/syntactically incorrect?**

For all versions of Acrobat 6.0 and later, Acrobat won't delete a bad CRL from the cache, but it will try to update it every time the digital signature is validated. If the previous CRL is in the cache and is considered valid, then Acrobat will not update a fresh copy. Acrobat only downloads another after the previous CRL is no longer usable. The new CRL overwrites the previous, cached copy. If the fresh copy of the CRL was corrupt and the user needed to rely on the previous copy (e.g. changing the validation time from the current time to the signing time) they would have lost the opportunity to use the older (but valid) CRL.

The only way to ensure that an older CRL will always be available to Acrobat is to embed it within the signature data at the time of signing.

**What is the caching behavior, if the CRL is obsolete?**

For all versions of Acrobat 6.0 and later, if the obsolete CRL is in the cache it will be replaced if a new CRL can be obtained. Only CRLs contained within the signed data (embed revocation status at signing) will always be available. If the obsolete CRL in the cache folder cannot be replaced (e.g. a lack of internet connectivity) it will remain untouched.

**Can CRL download being controlled manually? User driven?**

For all versions of Acrobat 6.0 and later, no. Even if the user could obtain the CRL from the CA (e.g. via an LDAP or Internet browser) and the manually copied it into the cache folder, Acrobat wouldn't use it because the file name wouldn't be in the correct format. When Acrobat downloads a CRL it names the saved file based on a hash of the issuer DN information. It then does a subsequent lookup for cached CRL files based on the matching file name.

**What is the behavior if there are more than one valid source for CRLs (LDAP and HTTP)?**

For all versions of Acrobat 6.0 and later, Acrobat will first look for a valid CRL locally. That is, it will first look in the signed data that is the digital signature. If there is no embedded CRL, or if the embedded CRL is outside of the validity period based on the validation time (i.e. current time is being used as opposed to signing time), then Acrobat will check in the local cache folder for an updated CRL. If there is no valid CRL in the cache folder (Acrobat considers an invalid or obsolete is CRL the same as no CRL) then it will try to retrieve the revocation status from an online source.

Note that Acrobat will first try to retrieve an OSCP response and only tries to retrieve a valid CRL if that effort fails. Skipping to the CRL download, the first thing Acrobat does is look in the registry for a URI the points to the download location. While checking the registry it also checks to see if it is supposed to use the registry entry in lieu of the CDP (an override to the CDP) or only if the CDP extension does not yield
to downloading a valid CRL. If Acrobat is going to use the CDP extension it will try the try the locations (regardless of protocol) based on the order of entry.

A properly formatted CDP consists of a sequence of “DistributionPoint”. The DistributionPoint is itself a sequence of up to three items: distributionPoint, reasons, or crlIssuer. The distributionPoint entry is itself a sequence of general names, which in this case would be one or more URLs that point to the location where the CRL can be retrieved. Acrobat will start with the first entry in the sequence and continue looking until either a valid CRL has been retrieved or the list has been exhausted. Once Acrobat finds a valid CRL it quits looking.

**Will Acrobat check online for valid CDPs even if local, valid CRL is cached?**

For all versions of Acrobat 6.0 and later, Acrobat will only check online for a valid CRL if the cached (be it embedded or in the cache folder) CRL is considered invalid or obsolete.

### 7.4.4.3 MSCAPI Default Behavior

Acrobat can invoke MSCAPI to do revocation checking if need be. In most cases, MSCAPI uses CRL or OCSP, but it can use SCVP/XKMS/DSS and other methods.

**Basic Revocation Checking**

MSCAPI provides basic revocation checking which supports CRLs. CRLdps with HTTP(S), LDAP, and file types are supported. On Windows XP and later, there is some (limited) support for ActiveDirectory lookups. CRLdps are checked in the order they are found in the extension. If any CRLdp types other than the supported types are used, they must be AFTER the supported types. If not, there may be a 45 second delay for each CRLdp, for each certificate as these are checked and fail.

For example, entrust RAs require a CRLdp containing the CA name to be added to each intermediate CA. This CRLdp will cause the delay if placed before the http CRLdp.

**Note:** For more information, see Certificate Revocation Lists (CRLs) and IIS 5.0 FAQs at [http://support.microsoft.com/default.aspx?scid=KB;en-us;q289749](http://support.microsoft.com/default.aspx?scid=KB;en-us;q289749)

The basic revocation handler caches CRLs downloaded during signature validation in the Temporary Internet Files directory. These files are shared with Internet Explorer. CRLs may also be installed directly by the end user or system administrator from Internet Explorer. These CRLs must be installed in the Intermediate CA store to be effective.

**Other Revocation Handlers**

MSCAPI provides a mechanism for adding revocation providers and extending the basic functionality with other types. MSCAPI will query all providers until a definitive answer is reached or no providers are left to query. This may cause revocation checking to return differing results on systems where different revocation handlers have been installed.

7.4.5 Interaction of iReqRevCheck Settings

Depending on your application settings, revocation checking may occur when a signature is created or validated and on revocation check responses (if any). In other words, both signing certificate and any revocation check response certificates may be subject to revocation checking.

The application settings such as iReqRevCheck for cSign, cCRL, cOCSP, cVerify, and cAdobe_TSPProvider which affect a signature-related workflow; therefore, administrators should understand the default application behavior as well as the net result of proposed customizations before modifying these settings.

7.4.5.1 Revocation Checks During Signature Creation

Table 4 is provided as a guide to the most common scenarios.

At a high level, the general rules for signing workflows include the following:

- If iReqRevCheck for both cSign and OCSP/CRLRevCheck are 0 or 1, a signature always results.
- If iReqRevCheck for either cSign or OCSP/CRLRevCheck is 2 or 3, bIsEnabled is 1 (embed certificate data), and the certificate is revoked, no signature is created.
- If iReqRevCheck for either cSign or OCSP/CRLRevCheck is 2 or 3, bIsEnabled is 0 (don’t embed certificate data), and the certificate is revoked, an invalid signature is created.
- If iReqRevCheck for either cSign or OCSP/CRLRevCheck setting is 3 and a check doesn’t occur, no signature is created.

### Table 4 Registry setting’s impact on signature status during signature creation

<table>
<thead>
<tr>
<th>Signing certificate revocation checking</th>
<th>Response certificate revocation checking</th>
<th>Result based on cAdobe_LTVProvider setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>cSign\iReqRevCheck</td>
<td>Do rev check?</td>
<td>Revoked?</td>
</tr>
<tr>
<td>0</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>1</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Revoked</td>
</tr>
</tbody>
</table>
### Table 5: Registry setting’s impact on signature status during signature creation

<table>
<thead>
<tr>
<th>Signing certificate revocation checking</th>
<th>Response certificate revocation checking</th>
<th>Result based on cAdobe_LTVProvider setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>cSign\iReqRevCheck</td>
<td>Do rev check?</td>
<td>Revoked?</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
</tbody>
</table>

### 7.4.5.2 Revocation Checks During Signature Validation

Table 5 is provided as a guide to the most common scenarios. At a high level, the general rules for signature validation workflows include the following:

- If iReqRevCheck for both cVerify and OCSP/CRLRevCheck are 0 or 1, the signature status is valid.
- If iReqRevCheck for both cVerify and OCSP/CRLRevCheck is 1, 2, or 3 and the certificate is revoked, the signature status is invalid.
7.4.6 Customized Revocation Check Example

Most of the application’s revocation checking behavior is customizable. Modification is usually just a matter of creating and setting a few keys as described in Chapter 5, “Registry and plist Settings”. The following example presents a more complex scenario in which a number of behaviors are configured inside of a custom certificate preference.

Table 5 Registry setting’s impact on signature status during signature validation

<table>
<thead>
<tr>
<th>Signing certificate revocation checking</th>
<th>Response certificate revocation checking</th>
<th>Signature status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>1</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>1</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>No Check</td>
<td>Unknown</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>2</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
<tr>
<td>3</td>
<td>Check</td>
<td>Not revoked</td>
</tr>
</tbody>
</table>
Specifying an Alternate OCSP Responder

Caution: These instructions are only provided as an example. Adobe strongly recommends that you do not make changes to the registry unless you are knowledgeable about editing and troubleshooting Microsoft Windows registry settings. Improper use of this feature can result in the corruption of critical system files.

You can set an alternate OCSP responder to use when OCSP information is not present in the certificate or in the event that information is incorrect or out-of-date. This example demonstrates how to do two things:

- Redirect revocation checking on a Windows machine to a different OCSP responder than the one specified by the certificate. In this case, the administrator configures the application to ignore the AIA extension in the certificate and to use the custom certificate preferences instead.
- Place the instructions in a custom certificate preference so that they are only followed when a particular, matching certificate is used during revocation checking.

To specify an OCSP responder:

1. Navigate to `HKEY_CURRENT_USER\Software\Adobe\Adobe Acrobat\<version>\Security\cASPKI\cASPKI\cCustomCertPrefs\<SHA1 hash of public key or hex representation of an OID followed by two NULL characters>`.
2. Highlight the new key, right click, and choose New > Key.
3. Create a new container named `cAdobe_OCSPRevChecker`.
4. Create four new subkeys under `cAdobe_OCSPRevChecker`:
   - `cSendNonce`
   - `cSignRequest`
   - `cURLToConsult`
   - `cURL`
5. Under each of the four new keys, add a subkey named `c0` (Figure 6).
Figure 6  Revocation checking registry entry

6. Highlight `cSendNonce` and right click.

7. Choose New > DWORD Value and name it `bValue`.

8. Set Value Data to one of the following:
   - 0: False
   - 1: True (Default)

9. Highlight `cSignRequest` and right click.

10. Choose New > DWORD Value and name it `bValue`.

11. Set Value Data to one of the following:
    - 0: False (Default)
    - 1: True

12. Highlight `cURLToConsult` and right click.

13. Choose New > DWORD Value and name it `iValue`.

14. Set Value Data to one of the following:
    - 0: Use the certificate’s AIA extension (Default).
    - 1: Use the URL key in `cURL`
    - 2: Use the AIA in the certificate. If it is not present, use the URL key in `cURL`
    - 3: Use the AIA in the certificate to be used to sign the OCSP request. This setting requires that `SignRequest` is true.

15. Highlight `cURL` and right click.

16. Choose New > Binary Value and name it `sValue`.
17. Double click sValue to launch the Edit Binary Value dialog (Figure 7). Type the URL into the right side of the display (Click in the right column to get the cursor in place).

18. After entering the URL, type a zero on the end of the hex stream. This is a NULL terminator that appears as a dot at the end of the URL string. This is NOT a period.


Figure 7 Adding an OCSP URL in the binary editor
![Image of adding an OCSP URL in the binary editor]

7.4.7 Revocation Tab GUI and Strings

The Certificate Viewer’s Revocation tab displays information about whether revocation checking occurred, the type of check, and its result. The Revocation tab contains several fields which may contain revocation details: The summary string area and the four detail areas in the Detail panel (Figure 8).

Figure 8 UI revocation strings
![Image of UI revocation strings]

There are potentially four different strings that could appear in each of the first three fields. As shown in the following tables, these strings are determined by the following factors:

- **Certificate status**: Valid, invalid, unknown, or problem.
- **Type of revocation check**: CRL, OCSP, or MSCAPI.
- **Special conditions**: If there is a special condition the default string in the summary and details 1 field is replaces as shown in Table 10. The special conditions include:
There are path validation errors such as policy constraints, invalid times, or name constraints.

The certificate does not chain up to a trusted root.

The certificate is self-signed.

OCSP no check: The CA specifically states that no revocation checking should be done.

The user has specifically trusted the certificate. No revocation checks are done for such certificates.

**Note:** If there is a problem with the revocation check, a fourth detail field displays: Click Problems to view the problems that were encountered while performing this revocation check.

### Table 6  Revocation checking: GUI strings for CRL check

<table>
<thead>
<tr>
<th>Sig state</th>
<th>String</th>
<th>String for special condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>Default: The selected certificate is valid.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Invalid</td>
<td>Default: The selected certificate has been revoked.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Unknown</td>
<td>Default: Could not determine whether the selected certificate is still valid.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td>Default: Problem determining whether the selected certificate is still valid.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Valid: CRL/OCSP response in signature</td>
<td>Default: The selected certificate is considered valid because it does not appear in the Certificate Revocation List (CRL) that is embedded in the document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Valid: CRL/OCSP outside of signature</td>
<td>Default: The selected certificate is considered valid because it does not appear in a Certificate Revocation List (CRL).</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Invalid: CRL response is outside of doc</td>
<td>Default: The selected certificate has been revoked and appears in a Certificate Revocation List (CRL).</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Invalid: CRL response is embedded in doc</td>
<td>Default: The selected certificate has been revoked and appears in a Certificate Revocation List (CRL) that is embedded in the document.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Unknown</td>
<td>Default: The selected certificate does not provide information on how its revocation status can be verified. It cannot be determined whether this certificate is valid or whether it has been revoked.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td>Default: An attempt was made to determine whether the certificate is valid by checking whether it appeared in any Certificate Revocation Lists (CRLs).</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition: See Table 10.</strong></td>
<td></td>
</tr>
</tbody>
</table>
**Table 6  Revocation checking: GUI strings for CRL check**

<table>
<thead>
<tr>
<th>Sig state</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any except “Unknown” or “Problem”</td>
<td><strong>Default:</strong> The CRL was signed by %SIGNER% on %DATE1% and is valid until %DATE2%. There are no special conditions for this field.</td>
</tr>
<tr>
<td>If CRL does not have “valid until” info.</td>
<td><strong>Default:</strong> The CRL was signed by %SIGNER% on %DATE1%. There are no special conditions for this field.</td>
</tr>
<tr>
<td>Any</td>
<td><strong>Default:</strong> Click Signer Details to get more information on the source of the revocation information. There are no special conditions for this field.</td>
</tr>
</tbody>
</table>

**Table 7  Revocation checking: GUI strings for OCSP check**

<table>
<thead>
<tr>
<th>Sig state</th>
<th>String</th>
</tr>
</thead>
</table>
| Valid | **Default:** The selected certificate is valid.  
**String for special condition:** See Table 10. |
| Invalid | **Default:** The selected certificate has been revoked.  
**String for special condition:** See Table 10. |
| Unknown | **Default:** Could not determine whether the selected certificate is still valid.  
**String for special condition:** See Table 10. |
| Problem | **Default:** Problem determining whether the selected certificate is still valid.  
**String for special condition:** See Table 10. |
Table 7 Revocation checking: GUI strings for OCSP check

<table>
<thead>
<tr>
<th>Sig state</th>
<th>String</th>
<th>String for special condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid: OCSP response in signature</td>
<td>Default: The selected certificate is considered valid because it has not been revoked, as verified using an Online Certificate Status Protocol (OCSP) response that was embedded in the document.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Valid: OCSP outside of signature</td>
<td>Default: The selected certificate is considered valid because it has not been revoked, as verified in real-time using the Online Certificate Status Protocol (OCSP).</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Invalid: OCSP response is outside of doc</td>
<td>Default: The selected certificate has been revoked, as verified in real-time using the Online Certificate Status Protocol (OCSP).</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Invalid: OCSP response is embedded in doc</td>
<td>Default: The selected certificate has been revoked, as verified using the Online Certificate Status Protocol (OCSP) response that was embedded in the document.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Unknown</td>
<td>Default: The selected certificate does not provide information on how its revocation status can be verified. It cannot be determined whether this certificate is still valid or whether it has been revoked.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Problem</td>
<td>Default: An attempt was made to determine whether the certificate is valid by doing a revocation check using the Online Certificate Status Protocol (OCSP).</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Any except “Unknown” or “Problem”</td>
<td>Default: The OCSP Response was signed by %SIGNER% on %DATE1% and is valid until %DATE2%. There are no special conditions for this field.</td>
<td></td>
</tr>
<tr>
<td>If CRL does not have “valid until info.”</td>
<td>Default: The OCSP Response was signed by %SIGNER% on %DATE1%. There are no special conditions for this field.</td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td>Default: Click Signer Details to get more information on the source of the revocation information. There are no special conditions for this field.</td>
<td></td>
</tr>
<tr>
<td>Sig state</td>
<td>String</td>
<td>String for special condition</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Valid</td>
<td><strong>Default:</strong> The selected certificate is valid.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>Invalid</td>
<td><strong>Default:</strong> The selected certificate has been revoked.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>Unknown</td>
<td><strong>Default:</strong> Could not determine whether the selected certificate is still valid.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td><strong>Default:</strong> Problem determining whether the selected certificate is still valid.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>Valid: OCSP response in signature</td>
<td><strong>Default:</strong> N/A.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Valid: OCSP outside of signature</td>
<td><strong>Default:</strong> The selected certificate is considered valid because it has not been revoked, according to Microsoft’s cryptographic revocation verification services.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Invalid</td>
<td><strong>Default:</strong> The selected certificate has been revoked, according to Microsoft’s cryptographic revocation verification services. Microsoft’s revocation services do not provide details regarding how revocation was performed.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Unknown</td>
<td><strong>Default:</strong> The selected certificate does not provide information on how its revocation status can be verified. It cannot be determined whether this certificate is valid or whether it has been revoked.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Problem</td>
<td><strong>Default:</strong> An attempt was made to determine whether the certificate is valid by doing a revocation check using Microsoft revocation services. There were problems encountered while performing this check. Problem details and the mechanism used to perform revocation checks are not available from Microsoft revocation services.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td>Any except “Unknown” or “Problem”</td>
<td><strong>Default:</strong> The OCSP Response was signed by %SIGNER% on %DATE1% and is valid until %DATE2%.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>If CRL does not have “valid until” info.</td>
<td><strong>Default:</strong> The OCSP Response was signed by %SIGNER% on %DATE1%.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
<tr>
<td>Any</td>
<td><strong>Default:</strong> Click Signer Details to get more information on the source of the revocation information.</td>
<td>See Table 10.</td>
</tr>
<tr>
<td></td>
<td><strong>String for special condition:</strong> See Table 10.</td>
<td></td>
</tr>
</tbody>
</table>
### Table 9 Revocation checking: Revocation info button text

<table>
<thead>
<tr>
<th>Condition</th>
<th>Button text</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRL check</td>
<td>CRL Signer Details...</td>
</tr>
<tr>
<td>OCSP check</td>
<td>OCSP Signer Details...</td>
</tr>
<tr>
<td>MSCAPI</td>
<td>No button shown</td>
</tr>
<tr>
<td>Path validation error</td>
<td>No button shown</td>
</tr>
<tr>
<td>Certificate does not chain to trusted root</td>
<td>No button shown</td>
</tr>
<tr>
<td>Certificate is self-signed</td>
<td>No button shown</td>
</tr>
</tbody>
</table>

### Table 10 Revocation checking: Strings for special conditions

<table>
<thead>
<tr>
<th>Field</th>
<th>Sig state</th>
<th>Path validation error</th>
<th>Untrusted chain</th>
<th>Self-signed</th>
<th>OCSP no check</th>
<th>Cert is trust anchor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Revocation checks not performed.</td>
<td>Revocation checks not performed.</td>
</tr>
<tr>
<td>Invalid</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Unknown</td>
<td>Revocation checks not performed.</td>
<td>Revocation checks not performed.</td>
<td>Revocation checks not performed.</td>
<td>NA</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"Acrobat Family of Products"
### Table 10 Revocation checking: Strings for special conditions

<table>
<thead>
<tr>
<th>Field</th>
<th>Sig state</th>
<th>Path validation error</th>
<th>Untrusted chain</th>
<th>Self-signed</th>
<th>OCSP no check</th>
<th>Cert is trust anchor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid: CRL/OCSP response in signature</td>
<td>NA</td>
<td>NA</td>
<td>Self-signed and root certificates do not provide a mechanism for revocation checking to be done.</td>
<td>The selected certificate is considered to be valid because it has OCSP No-Check extension (see the Details Tab for details).</td>
<td>The selected certificate is either a Trusted Root or a certificate above the Trusted Root in the certificate chain (see the Trust Tab for details). No revocation checks are done for such certificates, they are inherently considered trustworthy.</td>
<td></td>
</tr>
<tr>
<td>Valid: CRL/OCSP outside of signature</td>
<td>NA</td>
<td>NA</td>
<td>Self-signed and root certificates do not provide a mechanism for revocation checking to be done.</td>
<td>The selected certificate is considered to be valid because it has OCSP No-Check extension (see the Details Tab for details).</td>
<td>The selected certificate is either a Trusted Root or a certificate above the Trusted Root in the certificate chain (see the Trust Tab for details). No revocation checks are done for such certificates, they are inherently considered trustworthy.</td>
<td></td>
</tr>
<tr>
<td>Invalid: CRL response is outside of doc</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Invalid: CRL response is embedded in doc</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Unknown</td>
<td>There were errors encountered while building the certificate chain to a trusted root Certificate. Revocation checks</td>
<td>The selected certificate does not chain up to a trusted root certificate (see the Trust Tab for details). The result</td>
<td></td>
<td>Self-signed and root certificates do not provide a mechanism for revocation checking to be done.</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>
There are no special conditions for these fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sig state</th>
<th>Path validation error</th>
<th>Untrusted chain</th>
<th>Self-signed</th>
<th>OCSP no check</th>
<th>Cert is trust anchor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detail 2 and 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 10 Revocation checking: Strings for special conditions
Acrobat’s digital signature capabilities allow authors to set up a secure signing environment and create simple documents and complex forms with one or more fields. Document authors can design documents with multiple signature fields each with unique behavioral characteristics and appearances. A signed field can lock other fields so that signed data can’t be changed, and authors can force certain signature fields to be a required part of a workflow. Attention to signature field design and configuration can help you make the document “do the right thing” when someone receives it as well as control what that person can and cannot do with it.

For more information, refer to the following:
- “Best Practices for Signed Documents that will Change” on page 170
- “Setting up the Signing Environment” on page 170
- “Working with Signature Fields” on page 181
- “Authoring Signable Forms” on page 186

### 8.1 Best Practices for Signed Documents that will Change

Some workflows require that someone enter form data, provide annotations and comments, or sign a document. When the form author has authorized or the application allows such changes, the changes are not flagged as problematic or warnings. In general, the goal should be to design documents and workflows so that both the signature status and document status are always valid.

Best practices for signed documents that will change vary by role:
- **Document author:** Form fields can be ordered, named, and associated with behaviors that limit changes in signing workflows.
- **First signer**: Use a certification signature for the first signature in a document and set **Permitted Changes after Certifying** as needed. The specified actions should not result in a warning triangle to appearing on signatures.
- **Signature validators and subsequent signers**: View the signed version of the document and look at the signature’s status icon in the Signature’s pane as well as the document status icon in the message bar. If there are any issues or problems, read the text. You may also wish to view the document the PDF signature report, view modifications, and so on.

### 8.2 Setting up the Signing Environment

A number of preferences control how your application, the document, and the signature will behave in signing workflows. These preferences tell the application where to look for Windows certificates, control signature appearances, enable the use of preview mode, and so on.
Tip: Participants in signing workflows (both document authors and signers) should review their application settings and configure their environment. Some preferences control authoring, some only have to do with signing, and some impact both.

Table 11 Signing environment preferences

<table>
<thead>
<tr>
<th>Importance</th>
<th>Description</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recommended</td>
<td>A number of preferences affect the signing workflow and resulting signature. All users should review these settings.</td>
<td>Setting Signing Preferences</td>
</tr>
<tr>
<td>Optional</td>
<td>Create default user information ahead of time to save time later.</td>
<td>Setting Identity Information</td>
</tr>
<tr>
<td>Optional</td>
<td>Use the default signature appearance or create your own.</td>
<td>Signature Appearance Configuration</td>
</tr>
<tr>
<td>Optional</td>
<td>To sign with a timestamp, configure a timestamp server.</td>
<td>Timestaps for Signing</td>
</tr>
</tbody>
</table>

8.2.1 Setting Signing Preferences

Preferences options vary by platform and application as follows:

1. Choose one of the following:
   - Acrobat (Windows): Edit > Preferences > Security
   - Acrobat (Macintosh): Acrobat > Preferences > Security
   - Adobe Reader (Windows): Edit > Preferences > Security
   - Adobe Reader (Macintosh): Adobe Reader > Preferences > Security

2. Set your preferences as described in the following sections:
   - “Requiring Preview Mode” on page 171
   - “Changing the Default Signing Method” on page 172
   - “Embedding Signature Revocation Status” on page 173
   - “Allowing Signing Reason” on page 174
   - “Showing Location and Contact Details” on page 174
   - “Enabling Document Warning Review” on page 174
   - “Requiring Document Warning Review Prior to Signing” on page 175
   - “Enabling a Warnings Comment or Legal Attestation” on page 175

8.2.1.1 Requiring Preview Mode

In general, everyone within a signing workflow needs to know that what they are viewing is actually what is signed or being signed. Preview mode automatically checks document integrity and generates a report that itemizes any content that could potentially prevent the signer from knowing what they are signing.

Preview mode provides several benefits:
   - It checks the document for elements that may prevent a signer from seeing what they are signing.
• It suppresses document elements that may prevent a signer from knowing what they are signing.
• It generates a report about those elements. For details, see Chapter 13, “Document Integrity and Preview Mode”.

To use preview mode automatically:

1. Choose one of the following:
   • Acrobat (Windows): Edit > Preferences > Security
   • Acrobat (Macintosh): Acrobat > Preferences > Security
   • Adobe Reader (Windows): Edit > Preferences > Security
   • Adobe Reader (Macintosh): Adobe Reader > Preferences > Security

2. Set View documents in preview document mode when signing.

   Figure 9  Preview document mode preference

8.2.1.2 Changing the Default Signing Method

In some enterprise situations administrators may require a method other than Adobe Default Security. For example, non-Adobe plugins may be used in business environments that require support of biometrics, signature escrow, alternative methods of private key access, and so on. In those cases, administrators may preconfigure Acrobat to use an alternate plugin or provide user training on how to choose the right one.

Third party plugins include:

• **Entrust® plug-in for Acrobat 4 and 6**: This plugin interfaces to the Entrust Entelligence desktop application and provides the same functionality that is provided by Adobe’s plugin. Businesses that use Entrust for PKI deployment may require the Entrust plug-in.

• **SignCube® plug-in for Acrobat 7**: The SignCube plugin is used to create signatures recognized as valid under the German Digital Signature Law.

• **CIC**: The Communication Intelligence Corporation® Plugin (CIC) is used by some banks and insurance companies to provide an electronic version of handwritten signatures. This plugin limits users’ ability to use encryption.

To change the default signing method:
1. Choose one of the following:
   - Acrobat (Windows): Edit > Preferences > Security
   - Acrobat (Macintosh): Acrobat > Preferences > Security

2. Choose **Advanced Preferences**.

3. Choose the Creation tab (**Figure 10**).

4. Under **When Verifying**, check one of the last two radio buttons so that the **Default Method for Verifying Signatures** drop down list becomes active.

5. Select a signing method.

   **Note:** Do not change the signing method unless instructed to do so by your administrator.

### 8.2.1.3 Embedding Signature Revocation Status

1. Choose one of the following:
   - Acrobat (Windows): Edit > Preferences > Security
   - Acrobat (Macintosh): Acrobat > Preferences > Security
   - Adobe Reader (Windows): Edit > Preferences > Security
   - Adobe Reader (Macintosh): Adobe Reader > Preferences > Security

2. Choose **Advanced Preferences**.

3. Choose the Creation tab (**Figure 10**).

   **Figure 10** Signature creation preferences

4. *(Recommended)* Set Include signature’s revocation status when signing.

   Embedding the signing certificate’s revocation status in a document allows recipients to validate certificates (signatures) while offline and speeds up the revocation checking process. Moreover, if a certificate is revoked or expired at some time after signing, embedded revocation information
enables the application to determine if a certificate was valid at the time of signing so that the signature status will remain valid.

**Note:** Revocation checking occurs immediately after signing as well as during signature validation. If the revocation status is not embedded in a signature, the application looks in the locally stored certificate revocation list cache. If it is not there, the application goes online to complete the check.

### 8.2.1.4 Allowing Signing Reason

Turning this option on results in a **Reasons** field appearing in the signing dialog. The signer can then choose a default reason such as “I have reviewed this document” or create a new one.

1. Choose one of the following:
   - Acrobat (Windows): **Edit > Preferences > Security**
   - Acrobat (Macintosh): **Acrobat > Preferences > Security**
   - Adobe Reader (Windows): **Edit > Preferences > Security**
   - Adobe Reader (Macintosh): **Adobe Reader > Preferences > Security**

2. Choose **Advanced Preferences**.

3. Choose the Creation tab (Figure 10).

4. Set **Show reasons when signing**.

### 8.2.1.5 Showing Location and Contact Details

When this option is turned on, the **Location** and **Contact Info** fields appear in the signing dialog.

1. Choose one of the following:
   - Acrobat (Windows): **Edit > Preferences > Security**
   - Acrobat (Macintosh): **Acrobat > Preferences > Security**
   - Adobe Reader (Windows): **Edit > Preferences > Security**
   - Adobe Reader (Macintosh): **Adobe Reader > Preferences > Security**

2. Choose **Advanced Preferences**.

3. Choose the Creation tab (Figure 10).

4. Set **Show location and contact information when signing**.

### 8.2.1.6 Enabling Document Warning Review

Enabling document warning review allows signers to check document integrity prior to signing. The document can be analyzed to determine if it contains any content that could adversely impact the integrity of the signing process. For example, a document could contain JavaScript that could change a data field before or after a signature is applied. Setting an option here affects what appears in the **Prevent Signing Until Document Warnings Are Reviewed** drop-down list.
8.2.1.7 Requiring Document Warning Review Prior to Signing

If document warning reviews are critical to your signing workflow, you can require them.

**Note:** Enable Reviewing of Document Warnings and Prevent Signing Until Document Warnings Are Reviewed settings function in tandem and should be set together. Setting both these options to Always results in the highest degree of assurance that the signing process is not adversely impacted by malicious content.

1. Choose one of the following:
   - Acrobat (Windows): Edit > Preferences > Security
   - Acrobat (Macintosh): Acrobat > Preferences > Security
   - Adobe Reader (Windows): Edit > Preferences > Security
   - Adobe Reader (Macintosh): Adobe Reader > Preferences > Security

2. Set Prevent Signing Until Document Warnings Are Reviewed. Select from the following:
   - Never: Signing can continue without a document warning review.
   - When certifying a document: Signers must choose Review to apply a certification signature.
   - Always: Signers must always choose Review when signing.

8.2.1.8 Enabling a Warnings Comment or Legal Attestation

For certified documents that contain multimedia, comments, or other dynamic content, it often beneficial to add a warnings comment or legal attestation that states that the content is OK and permitted by the author. If document warnings are enabled, then the signer can review the warnings
and either choose from Acrobat’s default comment “I have included this content to make the document more dynamic,” or create a custom comment.

To enable warnings comments on certified documents:

1. Set **Enable Reviewing of Document Warnings** to **When certifying a document** or **Always** as described in “Enabling Document Warning Review” on page 174.

### 8.2.2 Signature Appearance Configuration

The signing workflow allows you to select from a list of signature appearances. You can use the default appearance created from your name or create one or more custom appearances and store them for later use. An appearance consists of three main components, and each can be separately customized:

- **Signature**: The text or graphic that identifies the signer on the left hand side of the appearance. Photos and scanned signatures are common. When using a graphic, make the background transparent if a watermark should be visible in the underlying layer.

- **Watermark or background**: A background image or text that is automatically applied to each signature. By default, this is the Adobe PDF logo.

- **Signature details**: Signature data that the signer want to include in the appearance.

**Figure 11 Custom signature appearance**

### 8.2.2.1 Custom Signature Appearance

Users generally customize one or more signature appearances and store them for later use. Available signatures are listed in the appearance panel (Figure 12).

**Note:** If you have created a watermark file as described in “Custom Watermark and Backgrounds” on page 178, the watermark should automatically appear in all of your signature appearances.

To create a new signature appearance:

1. Choose **Edit > Preferences** (Windows) or **Acrobat > Preferences** (Macintosh).
2. Choose **Security** in the left-hand list.
3. In the Appearance panel, choose **New**.
4. Configure the signature appearance options:
   - **Title**: Any arbitrary title used to identify the appearance.
   - Set the graphic options in the Graphic panel
     - **No graphic**: No graphic is used.
     - **Imported graphic**: Choose File > Browse, select a file and choose OK.
     - **Name**: Your text name will appear instead of a graphic. The name is extracted from the signing certificate.

   **Note**: By default, the signature watermark is the Adobe PDF logo but it can be customized. To avoid obscuring a background, use line art with a transparent background.

   - Set the text options in the Configure Text panel
     - **Name**: The name associated with the certificate.
     - **Date**: The date signed.

   **Note**: Signature appearances can only display local (computer) time, and it will likely differ from that in the Date/Time tab on the Signature Properties dialog when a timestamp server is used.

   - **Location**: The location associated with the identity configured in Acrobat.
   - **Reason**: The reason for signing.
   - **Distinguished name**: A name with details such as country, organization, organizational unit, and so on.
   - **Labels**: A label for each of the items above. For example, Reason.
   - **Logo**: The logo or graphic used as a background watermark.

   - Set the text options in the Text Properties panel
     - **Text Direction**: Choose a direction appropriate for the signer’s language.
     - **Digits**: If languages are installed that use digits other than 1234567890, the drop-down list will be populated with alternate choices. Choose a digit set appropriate for the signer’s language.

5. Choose OK.
8.2.2.2 Custom Watermark and Backgrounds

A watermark is a partially transparent graphic or logo that appears “behind” a signature. By default, the watermark is the Adobe PDF logo. Line (vector) art that is simple and unobtrusive often works best.

1. Import a logo or create a new one in a program such as Adobe Illustrator (used in the example below).

2. Set a low transparency level and flatten the transparency:
   1. Select all and group the objects if there is more than one.
   2. Choose Window > Transparency and slide the transparency slider to some low value such as 20%.
   3. Choose Object > Flatten Transparency. Leaving the Raster/Vector balance at 100%.
   4. Save the file to a PDF file.

3. Open the PDF file in Acrobat.

4. Crop the page and remove white space.

   **Note:** The method varies across product versions. For example, for 8.x, choose Document > Crop Page and check Remove White Margins.

5. Save the file as SignatureLogo.pdf in:
8.2.2.3 Editing or Deleting a Signature Appearance

To edit a signature appearance:

1. Choose **Edit > Preferences** (Windows) or **Acrobat > Preferences** (Macintosh).
2. Choose **Security** in the left-hand list.
3. Highlight an appearance in the Appearance panel.
4. Choose **Edit** or **Delete**.
5. Edit the appearance.
6. Choose **OK**.

8.2.3 Timestamps for Signing

Signature times tell you that a document and signature existed prior to the indicated time. All signatures are associated with the signer machine’s local time, but they may also include a timestamp time provided by a timestamp server if one is configured. Because a user can set that time forward or back, a local time is less reliable than a timestamp time. Local times are labelled as such in the Date/Time and Summary tabs of the Signature Property dialog (Figure 14).

**Note:** Because signature appearances only display local time, the appearance time will be different from the timestamp time shown in the Date/Time tab of the Signature Properties dialog.

**Figure 14 Timestamps: Local, machine time**

Like signatures, timestamps are provided by someone (a timestamp authority) who uses certificates to confirm their identity. Before you can validate a timestamp, you must explicitly trust the timestamp authority’s certificate. Timestamp certificate status appears in the Date/Time and Summary tabs of the Signature Property dialog:

- Untrusted timestamp certificates appear as follows:

**Figure 15 Timestamps: Untrusted stamp**

- Trusted timestamps that have been added to the Trusted Identities list and have been explicitly trusted for signing appear as follows:
Timestamps are usually provided by third-party timestamp authorities such as GeoTrust. Because timestamp authorities may charge for their services, Acrobat does not automatically set a default timestamp server if multiple servers are listed. Users must manually specify which timestamp server to use as the default.

**Configuring Acrobat to use a Timestamp Server**

To use a timestamp when you sign, configure your application to use a timestamp server, set it as the default, and set trust the certificate of the timestamp authority. The timestamp server is always used if a default timestamp server is specified.

To manually set up a timestamp server:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Choose **Timestamp Servers**.
3. Choose **New**.
4. Enter the server settings:
   - **Name**: The server name.
   - **Server URL**: The server URL.
   - **Username**: The login username if required.
   - **Password**: The login password if required.
5. Choose **OK**.
6. Make this server or some other server the default by choosing **Set Default**. Timestamping cannot occur unless a default server selected.
8.3 Working with Signature Fields

Signature fields are a type of form field, and both Acrobat and Adobe Reader ignore whether they are authored with Forms Designer or Acrobat. Digital signatures behave uniformly irrespective of the authoring mechanism.

For details about customizing one or more fields, see the following:

- Specifying General Field Properties
- Customizing Field Appearances
- Changing the Default Field Appearance
- Cut, Copy, and Paste Signature Fields
- Arranging Signature Fields
- Creating Multiple Copies of a Signature Field
- Authoring a Document with Multiple Fields
- Locking Fields Automatically After Signing
- Unlocking a Field Locked by a Signature
- Making a Field a Required Part of a Workflow
- Specifying a Post-Signing Action

8.3.1 Creating a Blank Signature Field

Signatures and related information are stored in a signature field embedded on the page. A signature field is an Acrobat form field. Signature fields are automatically created at the time of signing, but it is also possible to create empty signature fields for later signing.

To create a signature field:

1. Choose **Forms > Add or Edit Fields.**
2. Select the **Add New Field** button.

3. Choose **Digital Signature**.

4. Click and drag where the field should appear. The Digital Signature Properties dialog appears.

5. For simple signature fields, choose **Close**.

   By default, field names are numbered sequentially starting with “Signature1” and contain the default tooltip “Unsigned signature field (click to sign).”

   **Figure 18 Signature field: Default appearance**

6. If you would like to set field properties, choose the **Show All Properties** link in the popup box (for details, see “Specifying General Field Properties” on page 182); otherwise, exit the forms editor.

### 8.3.2 Specifying General Field Properties

A signature field’s general properties include name, tooltip, display behavior, and so on. For example, fields are numbered sequentially and are associated with a generic tooltip. However, the field can be given a unique name, provided with tooltip instructions for an eventual signer, and configured to display only in the Signatures tab and not in the document.

**Note:** You cannot edit these properties during signing workflows. An author must create a blank signature field and edit the properties before initiating the signing process. Moreover, invisible field properties cannot be edited.

To change a field’s general properties.

1. Create a new field.

   **Note:** For existing fields, place them field in edit mode by selecting **Forms > Add or Edit Fields** and then double click on them OR right click and choose **Properties**.

2. Display the General tab.

3. Configure the options:
   - **Name**: Any arbitrary name.
   - **Tooltip**: Any arbitrary text. It is required to make the document accessible to the visually impaired.
   - **Form Field**: Set the field display properties.
8.3.3 Customizing Field Appearances

Field border properties, fill color, fonts, and so on can be individually specified. These properties are NOT editable during signing workflows. An author must create a blank signature field and edit the properties before initiating the signing process. Invisible field properties cannot be edited.

To change a signature field's appearance:

1. Create a new field.

   **Note:** For existing fields, place them field in edit mode by selecting **Forms > Add or Edit Fields** and then double click on them OR right click and choose **Properties**.

2. Display the Appearance tab.

3. Configure the appearance options.

4. Edit the properties on the other tabs or choose **Close**.
8.3.4 Changing the Default Field Appearance

The default appearance of a blank signature field is a light blue box with no borders that performs no action on signing. These defaults can be changed globally so that all future signature fields will have a custom appearance and action.

To change signature field defaults:

1. Only the attributes on the appearance and Actions tab can be set as defaults for future fields. Customize a field as described in the following.
   - Customizing Field Appearances
   - Specifying a Post-Signing Action

2. Choose Close.

3. Right click on the field.


8.3.5 Cut, Copy, and Paste Signature Fields

The forms field context menu provides a number of editing items, including options for cutting, copying, pasting, and deleting.

To perform an edit action on a field:

1. Place the field in edit mode by selecting Forms > Add or Edit Fields.

2. Right click on the field.

3. Choose Edit.

4. Choose Cut, Copy, Paste, or Delete (Figure 21).
8.3.6 Arranging Signature Fields

While you can drag and drop fields anywhere, the field context menu provides a number of options for arranging multiple fields such as aligning, centering, and distributing fields.

To arrange multiple fields:

1. Place the fields in edit mode by selecting Forms > Add or Edit Fields.
2. Drag a rectangle around the fields to arrange.
3. Right click.
4. Choose Align, Center, or Distribute and use the submenus to arrange the fields (Figure 21).

8.3.7 Creating Multiple Copies of a Signature Field

Once a field is configured, multiple copies of the field can be placed on the same page.

To create multiple copies of a field:

1. Place the field in edit mode by selecting Forms > Add or Edit Fields.
2. Right click on the field.
3. Choose Create Multiple Copies (Figure 21).
4. Configure the copy options, including:
   - The number of fields down and across.
   - The overall field size.
   - The overall position.
5. Choose OK.

**Tip:** Acrobat automatically names the fields by numbering them. Providing unique and intuitive names helps signers and other document recipients navigate and interact with the document.
8.4 Authoring Signable Forms

Many documents that require signatures are forms. Some forms may have multiple signatures fields, with different signers providing data in certain other form fields. In such cases, document design, field layout, and even field appearance may contribute to the ease with which the form can be integrated into an efficient business process.

For example, it is often useful to lock the form fields associated with a particular signature field once it is signed. This eliminates the need to examine the signed document version to see if the value of a field was changed between that signed version and the current version. For more information, see the following:

- “Authoring a Document with Multiple Fields” on page 186
- “Locking Fields Automatically After Signing” on page 187
- “Making a Field a Required Part of a Workflow” on page 188
- “Specifying a Post-Signing Action” on page 189
- “Unlocking a Field Locked by a Signature” on page 191

8.4.1 Authoring a Document with Multiple Fields

Documents commonly have multiple form fields, and one or more signature fields are often used to verify or approve the data in preceding fields. In these cases, proper document layout and field design may be a critical aspect of usability. When designing a complex form for signing, consider using the following field properties:

- **Layout**: Design the form so that form data precedes a signature. If there is more than one signature field, make sure end users can understand which signature fields are associated with specific data.
• **Appearance**: Signature fields can look similar to other form fields, but it may be desirable to customize their appearance so they can be more readily distinguished. For details, see “Customizing Field Appearances” on page 183.

• **Names and tooltips**: Intuitive field names and tooltips facilitate authoring and signing in the following ways:
  - Help the author choose which fields should be read only in the Signed tab of the Digital Signature Properties dialog as well as what field to call when JavaScript is used to customize a document.
  - Help signers find fields and understand how to use the form. For details, see “Specifying General Field Properties” on page 182.
  - Make it easier for signature validators to identify which fields have changed since the names may be used in the Signature pane and elsewhere.

• **Locking behavior**: Consider which fields should become read-only after signing. Locking certain fields helps prevent document changes that could cause a signature to become invalid. For details, see “Locking Fields Automatically After Signing” on page 187.

### 8.4.2 Locking Fields Automatically After Signing

Form authors can designate which form fields should be locked after any other field is signed. Both signed and unsigned signature fields can be configured to become read-only after they are signed. By setting post-signing, field locking properties, authors can prevent data changes to any combination of form or signature fields. Two common use cases for automatic locking include:

- Preventing users other than the document author from clearing or re-signing a field.
- Preventing users from changing form data after the document has been signed.

To automatically lock one or more fields after signing:

1. Create a new signature field.

   **Note**: For existing fields, place them field in edit mode by selecting **Forms > Add or Edit Fields** and then double click on them OR right click and choose **Properties**.

2. Display the Signed tab.
3. Choose **Mark as read only**.

When a field with field locking is signed, both a normal document signature and an object hash of the locked fields are produced and included in the document. When the signature is validated, the viewing application validates the bytes of the PDF file AND compares the object hash in the signature to the object hash from the objects in memory. This allows the application to detect prohibited, changes.

4. Use the drop-down list to select from the following:
   - **All fields**: All signature fields will be read only after signing.
   - **All fields except these**: All signature fields except those specified will be read only after signing. Choose **Pick** and select the field to exclude.
   - **Just these fields**: Only the specified signature fields will be read only after signing. Choose **Pick** and select the field to include.

5. Choose **Close**.

### 8.4.3 Making a Field a Required Part of a Workflow

Certain workflows may require a signature. For example, after a signature field is signed, form fields may be prepopulated or additional fields may appear. It is also common for form designers to require signing before the document can be emailed or submitted to a server for processing.

To require a signature:

1. Place the field in edit mode by selecting **Forms > Add or Edit Fields**.
2. Right click on the field and choose **Properties**.
3. Check **Required** on the General tab.
4. Choose **Close**.

Users can still open, close, save, and send the document without any indication that the field is required until the document author sets up a check for the required flag. For example, the check could be as
simple as emailing the document. In this case, the author would add an action to the button to submit
the document and configure a URL. If the document recipient clicks on the field and then cancels the
signing process, an alert will appear. Server-side and other JavaScript checks are also possible.

**Figure 24 Required field not signed alert**

![](image)

### 8.4.4 Specifying a Post-Signing Action

JavaScript actions can be associated with a signature field so that an action occurs whenever the user
interacts with the field in some predefined way. However, documents are usually signed to protect,
guarantee, and or attest to the signed content. Signers usually want to know that the document they
are seeing is the document they are signing, and document recipients usually need to know that the
document they are viewing is the same as the document that was signed. For this reason, adding
actions to a signature field is inadvisable. Field actions change the underlying bytes of a PDF and could
adversely affect document security as well as content integrity.

**Caution:** Using this feature is NOT recommended since such actions are contrary to the
secure and trusted nature of most signing workflows. Adding actions will result in a
legal warning about the legal integrity of the document.

To associate an action with a field.

1. Create a new field.

   **Note:** For existing fields, place the form in edit mode by selecting **Forms > Add or Edit Fields** and
   then double click on them OR right click and choose **Properties**.

2. Display the Action tab.
3. Configure the options:

- **Select Trigger**: Choose a type of action.
  - **Mouse Up**: The user clicks on the field and releases.
  - **Mouse Down**: The user clicks on the field.
  - **Mouse Enter**: The cursor enters the field.
  - **Mouse Exit**: The cursor exits the field.
  - **On Focus**: The user hovers over or tabs to the field.
  - **On Blur**: The user stops hovering over or tabs away from the field.
- **Select Action**: See Table 5.

4. Choose **Add**.

5. Follow the action instructions that appear in the action dialog.

6. Optional: Move actions **Up**, **Down**, **Edit**, or **Delete** actions as necessary.

7. Choose **Close**.

### Table 5 Actions that can be associated with a signature field

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute a Menu Item</td>
<td>Executes a specified menu command as the action.</td>
</tr>
<tr>
<td>Go to 3D View</td>
<td>Changed the view to the 3D view specified by the form author.</td>
</tr>
<tr>
<td>Go to a Page View</td>
<td>Jumps to the specified destination in the current document or in another document.</td>
</tr>
</tbody>
</table>
Unlocking a Field Locked by a Signature

When a signature field’s properties specify that signing will automatically lock other fields, those fields cannot be edited until they are unlocked. Since it was a signature that locked the fields in the first place, unlocking the fields simply involves clearing the signature.

**Tip:** Unlocking a field is the same thing as clearing the signature field. Only the signer clear the signed field.

To unlock a field:

1. Right click on the signature field.
2. Choose **Clear Signature**.

Other fields can now be edited.
Controlling Signing with Seed Values

Acrobat’s seed value feature helps authors control document behavior once it has been routed to the signer. Seed values can be used to embed certificate requirements and other instructions in signature fields. When a signer signs a custom, “seeded” field, the author-specified behaviors are automatically invoked and enforced.

A seed value specifies an attribute and attribute value. The author can make the seed value a preference or a requirement. For example, you can use seed values to limit a user’s choices when signing a particular signature field. For details about what seed values can be used to control, see the following:

- Forcing a Certification Signature
- Giving Signers the Option to Lock a Document
- Forcing Signers to Use a Specific Signature Appearance
- Adding Custom Signing Reasons
- Specifying Timestamps for Signing
- Specifying Alternate Signature Handlers and Formats
- Specifying a Signature Hash Algorithm
- Embedding Revocation Information in a Signature
- Specifying Certificate Properties for Signing
  - Specifying Signing Certificates Origin
  - Specifying Certificates by Key Usage
  - Specifying Certificates by Policy
  - Specifying a URL When a Valid Certificate is not Found
- Custom Workflows and Beyond

9.1 Seed Value Basics

How you add a seed value to a document varies across products:

- **Acrobat**: JavaScript calls must be used because no direct user interface is provided. The JavaScript function `signatureSetSeedValue` sets properties that are used when signing signature fields. The properties are stored in the signature field and are not altered when the field is signed, the signature is cleared, or when `resetForm` is called. This method (and JavaScript generally) can be executed with a batch process, by dropping the script in Acrobat’s JavaScript subdirectory, menu events, Acrobat’s JavaScript debugger, and other methods.

- **LiveCycle Designer**: Set seed values via the signature field Properties panel.

When setting seed values, keep in mind the following:
Seed values should not be set on signed documents and cannot be set on certified documents after the document is certified. They are primarily used to configure fields on documents that are not yet signed.

Setting a seed value often causes Acrobat to not display or use its default settings. For example, default reasons are stored in a registry list, and specifying signing reasons with a seed value overrides that list.

Seed value properties include those listed in Table 6. Note that certspec and timeStampspec are objects that have multiple properties.

9.1.1 Changes Across Releases

Each Acrobat release results in support for additional seed values as shown in Table 4.

9.1.2 Supported Seed Values

Note: The examples in this document demonstrate the simplest case. For more information, refer to the Acrobat JavaScript Scripting Guide and JavaScript for Acrobat API Reference.

Table 6  Seed values: Object properties and descriptions

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certspec</td>
<td>object</td>
<td>A seed value CertificateSpecifier Object. For details, see “Specifying Certificate Properties for Signing” on page 205.</td>
</tr>
<tr>
<td>digestMethod</td>
<td>array of strings</td>
<td>(Acrobat 8.0) An array of acceptable digest methods to use while signing. These are only applicable if the digital ID contains RSA public and private keys. If they contain DSA public/private keys, then the value is always SHA1. Valid values include: MD5, SHA1 (default), SHA256, SHA384, SHA512, and RIPEMD160.</td>
</tr>
<tr>
<td>filter</td>
<td>string</td>
<td>The language-independent name of the signature handler to be used when signing.</td>
</tr>
<tr>
<td>flags</td>
<td>number</td>
<td>A set of bit flags controlling which of the following properties are required. The value is the logical OR of the following values, which are set if the corresponding property is required: 1: filter 2: subFilter 4: version 8: reasons 16: legalAttestations (Acrobat 8.0) 32: shouldAddRevInfo (Acrobat 8.0) 64: digestMethod (Acrobat 8.0) 128: lockDocument 256: appearanceFilter Usage: 1 specifies filter, 3 specifies filter and sub-filter, and 11 specifies filter, sub-filter, and reasons. If this field is not present, all properties are optional. For more details, refer to the PDF Reference.</td>
</tr>
</tbody>
</table>
Controlling Signing with Seed Values

Supported Seed Values

Table 6 Seed values: Object properties and descriptions

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>real</td>
<td>See version (above). (Optional) The minimum required capability of the signature field seed value dictionary parser. A value of 1 specifies that the parser must be able to recognize all seed value dictionary entries specified in PDF 1.5. A value of 2 specifies that it must be able to recognize all seed value dictionary entries specified in PDF 1.7 and earlier. A value of 3 specifies that it must be able to recognize all seed value dictionary entries specified in PDF 1.7-ADBE-3 and earlier. The Ff (flags above) entry indicates whether this is a required constraint. For more details, refer to the PDF Reference.</td>
</tr>
<tr>
<td>legalAttestations</td>
<td>array of strings</td>
<td>(Acrobat 7.0) A list of legal attestations that the user can use when creating an MDP (certified) signature.</td>
</tr>
<tr>
<td>mdp</td>
<td>string</td>
<td>(Acrobat 7.0) The modification, detection, and prevention (MDP) setting to use when signing the field. Values include: allowNone default defaultAndComments While allowAll is a legal value, it cancels out the effect of mdp and no certification signature can be used for this field.</td>
</tr>
<tr>
<td>reasons</td>
<td>array of strings</td>
<td>A list of reasons that the user is allowed to use when signing. (Acrobat 8.0) If this array contains a single empty string and reasons are marked as required using the flags variable, Acrobat will not allow a signing reason. If this array is empty and reasons are marked as required, an exception will be thrown.</td>
</tr>
<tr>
<td>shouldAddRevInfo</td>
<td>boolean</td>
<td>(Acrobat 8.0) The default value is false. If true, the application does certificate and chain revocation checking and embeds the information in the signature. If true and the flag is set to require these actions, any failure in these actions results in signing failure. Only relevant if subFilter is adbe.pkcs7.detached or adbe.pkcs7.sha1. If the subFilter is adbe.x509.rsa_sha1 and adding revocation information is required, the signing operation fails.</td>
</tr>
<tr>
<td>subFilter</td>
<td>array of strings</td>
<td>An array of acceptable formats to use for the signature. Refer to the Signature Info object’s subFilter property for a list of known formats.</td>
</tr>
<tr>
<td>timeStampSpec</td>
<td>object</td>
<td>(Acrobat 7.0) A seed value timeStamp specifier object. It uses the url and flags properties to specify a timestamp server. For details, see “Specifying Timestamps for Signing” on page 201</td>
</tr>
<tr>
<td>version</td>
<td>number</td>
<td>The minimum required version number of the signature handler to be used to sign the signature field. Valid values are 1 and 2. (Acrobat 8) This must be set to 2 if this seed value object contains any Acrobat 8-specific content that is marked as required.</td>
</tr>
</tbody>
</table>
Authors sometimes use JavaScript to set seed values for signature fields. When Acrobat’s JavaScript console is used for JavaScript execution, the JavaScript debugger must be enabled.

**Tip:** If you do not intend to set seed values with JavaScript through Acrobat’s JavaScript debugger, skip this section.

To enable the JavaScript debugger:
1. Choose **Edit > Preferences** (Windows) or **Acrobat > Preferences** (Macintosh).
2. Choose **JavaScript** in the left-hand category list.
3. Check **Enable JavaScript**.
4. Check **Enable JavaScript debugger after Acrobat is restarted**.
5. Restart Acrobat.

To set seed values with the console (JavaScript debugger) in Acrobat, do the following:
1. Choose **Ctrl + J**.
2. Use the **View** drop-down list and select **Console**.
3. Enter the requisite JavaScript.
4. Highlight the JavaScript. If you do not highlight the JavaScript, only the last line of code is executed.
5. Press Control + Enter simultaneously or select the Enter key on the numeric keypad.

   **Tip:** When the JavaScript is executed correctly, the debugger returns “undefined.”

6. Save the document, and test the field.

**Figure 26 Seed values: JavaScript debugger**

![JavaScript Debugger Image]

9.2 Forcing a Certification Signature

By default, signature fields can be signed with an approval or certification signature at the time of signing. However, it is possible to constrain a signature field such that only a certification signature can be used.

Certification signatures are always associated with modification detection and prevention settings (and an mdp property) that control what types of changes can be made to a document before the signature becomes invalid. Changes are stored in the document as incremental saves beyond the original version of the document that was covered by the certifying signature. The mdp seed value allows you to control what behavior the signer can allow after signing (Figure 27).

**Note:** If a document is already signed, fields with the mdp property specified will NOT invoke the certifying workflow. No error is given. Do not use mdp unless you are sure the requisite field will be the first one signed.

**Figure 27 Seed value: Forcing mdp selection during certification**

![Seed Value Image]
Controlling Signing with Seed Values

Forcing a Certification Signature

MDP has one of the following four values:

- **allowAll**: Do not use allowAll unless you want to force an approval signature since this value results in MDP not being used for the signature and therefore doesn’t force a certifying signature.

- **allowNone**: Document changes invalidate the signature and lock the author’s signature. allowNone bypasses any custom legalAttestations because no document changes can occur and the user does not therefore need to be warned about malicious content. Do not use with legalAttestations.

- **default**: Allow form field fill-in if fields are present in the document as well as additional signatures. Other changes to the document invalidates the signature.

- **defaultAndComments**: Allow form field fill-in if fields are present in the document and allows annotations (comments) to be added, deleted, or modified as well as additional signatures. Other changes to the document invalidates the signature. Note that annotations can be used to obscure portions of a document and thereby affect the visual presentation of the document.

To force a certifying signature for a particular field:

1. Create a signature field with an intuitive name and tooltip.

2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.1).

3. Set the mdp value:
   - **allowNone**: Do not use with legalAttestations.
   - **default**: Allow form field fill-in, including signing.
   - **defaultAndComments**: Allow form field fill-in, signing, and comments.

4. Add legalAttestations if you would like to customize user choices.

Signers can view warnings about potentially malicious content (content that could change the appearance of a signed document) during signing. The Review button in the signing dialog runs the PDF/SigQ Conformance Checker which reports on rich content. Signers can then enter a **Warnings Comment** in the drop-down list indicating why that content is OK.

When specifying custom legal attestations, keep the following in mind:

- Since certified document warnings only appear in certifying workflows, only use legalAttestations if you also use mdp. For details, see “Forcing a Certification Signature” on page 196.

- Customizing legal attestations overrides and removes default choices for the signer.

- Custom text is viewable in the user interface during signing when the signer chooses **Review** in the signing dialog.
5. Highlight the JavaScript and choose **Control + Enter** or choose the **Enter** key on the numeric keypad.

When someone signs the field, the certifying workflow is invoked and only the specified mdp settings will be available (Figure 27).

6. Run the JavaScript, save the document, and test the field.

**Example 9.1: Seed value: mdp**

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue(
{
    mdp: "defaultAndComments",
    legalAttestations: ["Approved by Management", "Signed by Procurement"]
})
```

### 9.3 Giving Signers the Option to Lock a Document

While certifying a document allows authors and signers to restrict certain document features, for example, subsequent signing and filling out forms, these permissions are set at the document level and cannot become more restrictive as signatures are applied. Acrobat 9 provides a seed value that adds a **Lock Document** checkbox to the signing dialog when the configured field is selected for signing. This allows any signer to completely lock the document from further changes.

The author is given the choice of requiring their seed value to be honored by the signing application or to be an optional recommendation. This is provided through the use of the required flags bitfield to indicate whether or not the seed value is optional or mandatory. The options are as follows:
Controlling Signing with Seed Values

Giving Signers the Option to Lock a Document

- **true**: A value of true indicates that the desired action is that the document should be locked at the time of signing. If the Ff entry indicates that LockDocument is not a required constraint, the user can choose to override this at the time of signing. Otherwise the document must be locked after signing.

- **false**: A false value indicates that the document should not be locked after signing. By default, the user is given the option of locking after signing. Again, the required flag determines whether this is a required constraint.

- **auto**: Default. The auto value allows the consuming application to decide whether or not to present the lock UI for the document and whether to honor the required flag based on the properties of the document.

**Example 9.2: Seed value: lockDocument**

```javascript
f = this.getField("mySigFieldName");
f.signatureSetSeedValue({lockDocument:<true/false/auto>});
```

//Set the setting as required
f.signatureSetSeedValue({lockDocument:<true/false/auto>, flags:0x80});

**Figure 29 Sign Document dialog: With Lock Document checkbox added**
9.4 Forcing Signers to Use a Specific Signature Appearance

Enterprises and other structured work environments sometimes provide users with predefined signature appearances. These appearances are then used for specific signing purposes. For example, the appearance may identify the signer’s organizational affiliation or a particular task or workflow that pertains to the signed document.

Authors in such environments can specify which signature appearance is required for any given signature field. As with other seed values, a flag bit is used to indicate whether or not the field is a recommendation or mandatory. A signature field is correlated with a specified name which is then used to match a given appearance. The string name must exactly match the name of a signature appearance for it to be selected.

Example 9.3: Seed value: signatureAppearance

```javascript
f = this.getField(<Field Name>);
f.signatureSetSeedValue({AppearanceFilter:"Example Appearance Name"});

//Set the setting as required
f.signatureSetSeedValue({AppearanceFilter:"Example Appearance Name", flags: 0x100});
```

9.5 Adding Custom Signing Reasons

Acrobat predefines several common signing reasons such as “I am approving this document.” However, the author can specify custom reasons and make those reasons required or optional. When custom reasons are marked as required, users cannot enter any new reasons as the field becomes read-only. When those reasons are flagged as optional, signers can choose one of the provided reasons or create a new one by typing in the Reason field. Specifying a signing reason will remove all of the default reasons from the reason drop-down list.

**User interface impact:** Note that end users have a user interface preferences that allows them control whether or not the reason’s field appears. The preference interacts with the reasons flag as shown in Table 7, and the logic is as follows:

- The document author has control over whether the UI appears and the required flag overrides user-specified settings.
- When a flag makes the field optional, end users can enter custom reasons.

To specify custom signing reasons:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.4).
3. Add the reasons. The reason list is an array in the format of ["one", "two", "three"].
4. Enter a flag value to indicate whether the value is required or not.
   - If a reason is not required, signers can add their own custom reason while signing.
If the predefined reasons are required, signers are prevented from saving a document with their own reason (Figure 30).

5. Run the JavaScript, save the document, and test the field.

### Table 7 Reason field behavior

<table>
<thead>
<tr>
<th># of Reasons</th>
<th>UI Pref</th>
<th>Flag</th>
<th>Reason Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (empty array)</td>
<td>off</td>
<td>Required</td>
<td>Reason field does not appear in UI.</td>
</tr>
<tr>
<td>0 (empty array)</td>
<td>on</td>
<td>Required</td>
<td>Reason field does not appear in UI.</td>
</tr>
<tr>
<td>0 (empty array)</td>
<td>off</td>
<td>Optional</td>
<td>Reason field does not appear in UI.</td>
</tr>
<tr>
<td>0 (empty array)</td>
<td>on</td>
<td>Optional</td>
<td>Display the default list.</td>
</tr>
<tr>
<td>1 or more</td>
<td>off or on</td>
<td>Required</td>
<td>Display the custom reasons in a read-only field.</td>
</tr>
<tr>
<td>1</td>
<td>off</td>
<td>Optional</td>
<td>Reason field does not appear in UI.</td>
</tr>
<tr>
<td>2 or more</td>
<td>on</td>
<td>Optional</td>
<td>Display the custom drop-down list and let the user enter a custom reason.</td>
</tr>
<tr>
<td>2 or more</td>
<td>off</td>
<td>Optional</td>
<td>Reason field does not appear in UI.</td>
</tr>
</tbody>
</table>

**Example 9.4: Seed value: Custom signing reason**

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue{
  { reasons: ["This is a reason", "This is a better reason"],
    flags: 8
  }
}
```

**Figure 30 Seed value: Reason not allowed error**

---

### 9.6 Specifying Timestamps for Signing

Timestamps originating from a timestamp authority’s timestamp server are often associated with signatures. If it is critical in your workflow to acquire a secure timestamp with a digital signature, it can be controlled at the document level instead of relying on the signer’s Acrobat configuration. Adding a seed value to the signature field with the timestamp server authority settings overrides the corresponding application level settings, if any. Use the `timeStampspec` specifier object’s `url` and `flags` properties to specify a timestamp server.
To specify a timestamp server:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.5).
3. Provide a URL for the `timeStampspec` object.
   
   **Tip:** Timestamp seed value settings override the end users’ application settings, if any.

4. Enter a flag value to indicate whether the value is required or not.
   - If it is required, the field is automatically timestamped on signing. If the application cannot find the server, an error appears (Figure 31).
   - If it is not required, the field will be automatically timestamped on signing if the application can find the server. If it cannot find the server, the signature is not timestamped and no error appears.

5. Run the JavaScript, save the document, and test the field.

**Example 9.5: Timestamp server seed value**

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue({
    timeStampspec: {
        url: "http://153.32.69.130/tsa",
        flags: 1
    }
});
```

**Figure 31 Time stamp server error**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>string</td>
<td>URL of the timeStampspec server providing a RFC 3161-compliant timeStamps.</td>
</tr>
<tr>
<td>flags</td>
<td>number</td>
<td>A flag controlling whether the timestamp is required (1) or not required (0). The default is 0.</td>
</tr>
</tbody>
</table>
Organizations may choose to use alternate signature technologies or implementations (signature handlers), provided by third party software developers. For example, a corporation may have deployed Entrust Entelligence® to all their desktops and may choose to use the Entrust signature plug-in with Acrobat. Two seed values allow authors to specify which signature handler and format to use. By using a standard format, interoperability across multiple signature handlers is possible.

Filter also allows authors to control what handler version is required. For example, for Acrobat 6.x, the PPKLite version is 0. For Acrobat 7.x, the PPKLite version is 1. Therefore, specifying a version of 1 prevents signers from signing when their application is older than Acrobat 7.0. Custom handlers can use any version as required.

**User interface impact**: Specifying a non-default handler can result in a different user interface and workflow during signing.

Seed values for specifying handlers and signature types are the following:

- **filter**: filter is the internal name of a signature handler. Signature handlers perform a number of functions including signature validation. While Acrobat ships with a default handler (Adobe PPKLite), custom or third-party handlers such as those from Entrust and VeriSign may be used. The Acrobat SDK describes how to write a custom handler (Adbe.DocSign).

  **Tip**: filter is often used in conjunction with version when a minimum filter version is required.

- **subfilter**: subfilter is the internal name of the signature format, such as adbe.pkcs7.detached intended to be verifiable by signature handlers other than the one that created it. Signature handlers need to be able to understand the signature type (or format).

  **Tip**: Since it is possible that different handlers might be used for signing and validating, filter and subfilter are used together to assure that signing workflows with different components are interoperable. These properties are identical to those in the signature dictionary. For more information, refer to the *PDF Reference*.

To specify a signature handlers and format type:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method ([Example 9.6](#)).
3. Specify a filter.
4. If filter is specified, you may use the optional version as follows:
   - PPKLite for Acrobat 6.X: 0
   - PPKLite for Acrobat 7.x: 1
   - Custom handlers: Any.
5. Enter the handler name and subfilter type. Third parties may define their own subfilters but should follow the naming convention recommended in the *PDF Reference*. The *PDF Reference* defines the following standard subfilter values:
   - adbe.x509.rsa_sha1
   - adbe.pkcs7.detached
6. Run the JavaScript, save the document, and test the field.

Example 9.6: Seed value: Specifying signature components
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue(
    {
        filter: "Entrust.PPKEF",
        subfilter: "adbe.x509.rsa_sha1
    }
)

9.8 Specifying a Signature Hash Algorithm

When a signer’s digital ID contains RSA public and private keys, it is possible to specify alternative signature hash algorithms. The default algorithm is SHA1, and the alternatives are listed in Table 6.

User interface impact: Once a document is signed, the signature’s hash algorithm can be viewed by right clicking on a signature, choosing Show Signature Properties, and displaying the Document tab. The algorithm is displayed in the Hash Algorithm field.

Caution: If a signer may be using FIPS mode, do NOT specify MD5 or RIPMD160.

To specify a non-default algorithm:

1. Create a signature field with an intuitive name and tooltip.

2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.7).

3. Specify the digestMethod. This can be an array of comma-separated items such as ['RIPEMD160', 'SHA384'].

4. Run the JavaScript, save the document, and test the field.

Example 9.7: Hash algorithm seed value
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue({
    digestMethod: ['SHA384']
});
9.9 Embedding Revocation Information in a Signature

Users (signers) have the option to embed certificate revocation status in a signature by turning on Include signature’s revocation status when signing in their preferences. However, the default value is false (revocation information is not embedded), and document authors may need to force embedding of revocation information regardless of the users application settings. Embedding the signing certificate’s revocation status in a document allows recipients to validate certificates (signatures) while offline and speeds up the revocation checking process. Moreover, if a certificate is revoked or expired at some time after signing, embedded revocation information enables the application to determine if a certificate was valid at the time of signing so that the signature status will remain valid.

Note: Only relevant if subFilter is adbe.pkcs7.detached or adbe.pkcs7.sha1. If the subFilter is adbe.x509.rsa_sha1 and adding revocation information is required, the signing operation fails.

To force embedding of certificate revocation information in a signature:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.7).
3. Set shouldAddRevInfo to true.
4. Run the JavaScript, save the document, and test the field.

Example 9.8: Hash algorithm seed value

// Obtain the signature field object:
var f = this.getField("mySigFieldFilename");

f.signatureSetSeedValue({
  shouldAddRevInfo: true
});

9.10 Specifying Certificate Properties for Signing

Certificate seed values are commonly used to restrict signing to particular certificates such as those issued by particular certificate authorities or containing numbers that specify certain policies with “object identifiers” or “OIDs.” Authors specify which certificate signers must use by setting the certSpec object’s properties (Table 9). These can be preferences or requirements. If a certificate cannot be found that matches a required certificate seed value, a URL can be provided to allow the signer to get more information such as how to obtain an appropriate certificate.

Certificate specification can be used to streamline workflows. When one certificate is allowed, the digital ID dialog is bypassed and the signer is directed to sign and save immediately. Signing fails if the selected certificate is not an exact match. It is also often expedient to provide a URL value so that users are directed to a help page or some location where a digital ID can be obtained.
Controlling Signing with Seed Values

Specifying Certificate Properties for Signing

Figure 32 Seed value: Specifying certificates for signing

Table 9 Seed values: certSpec properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flags</td>
<td>number</td>
<td>A set of bit flags controlling which of the following properties of this object are required. The value is the logical OR of the following values, which are set if the corresponding property is required:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: subject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: issuer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4: oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8: subjectDN (Acrobat 8 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16: issuerDN (Acrobat 8 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32: keyUsage (Acrobat 8 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>64: url (Acrobat 8 and later)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this field is not present, all properties are optional.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Usage: 1 specifies subject, 3 specifies subject and issuer, and 6 specifies issuer and oid. That is, values can be added. If this field is not present, all properties are optional.</td>
</tr>
<tr>
<td>issuer</td>
<td>array of</td>
<td>One or more issuers that are acceptable for signing. The issuer can be a root or intermediate root certificate. Access to the physical, DER-encoded certificate is required. It is identified by a path to a discrete file in the format of &quot;/c/test/root.cer&quot;.</td>
</tr>
<tr>
<td></td>
<td>certificate</td>
<td>objects</td>
</tr>
</tbody>
</table>
Controlling Signing with Seed Values

### Specifying Certificate Properties for Signing

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyUsage</td>
<td>array of integers</td>
<td>(Acrobat 8.0) Integers in HEX or decimal that specify the <strong>keyUsage</strong> extension that must be present in the signing certificate. Each integer is constructed as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>There are two bits used for each keyUsage type (defined in RFC 3280) starting from the least significant bit:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- digitalSignature(bits 2,1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- nonRepudiation(4,3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- keyEncipherment(6,5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- dataEncipherment(8,7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- keyAgreement(10,9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- keyCertSign(12,11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cRLSign(14,13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- encipherOnly(16,15)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- decipherOnly(18,17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The value of the two bits have the following semantics:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>00</strong>: The corresponding keyUsage is not allowed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>01</strong>: The corresponding keyUsage is required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>10 and 11</strong>: The state of the corresponding keyUsage doesn’t matter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, if it’s required that keyUsage must require <strong>digitalSignature</strong> and the state of all other’s doesn’t matter, then the corresponding integer would be 0x7FFFFFFFFD. That is, to represent digitalSignature, set 01 for bits 2 and 1 respectively, and set 11 for all other keyUsage types.</td>
</tr>
<tr>
<td>oid</td>
<td>array of strings</td>
<td>One or more policy OIDs that must be present in the signing certificate’s policy. The OID is part of the value of the certificate’s certificate policy field. This property is only applicable if the issuer property is present. <strong>oid</strong> and <strong>issuer</strong> can be used together to specify a certificate that has the selected policy.</td>
</tr>
<tr>
<td>subject</td>
<td>array of certificate objects</td>
<td>One or more subjects that are acceptable for signing. The subject property identifies specific individuals (as certificate owners) that can sign. Access to the physical, DER-encoded certificate is required. It is identified by a path to a discrete file in the format of [&quot;/c/test/root.cer&quot;].</td>
</tr>
<tr>
<td>subjectDN</td>
<td>array of certificate objects</td>
<td>(Acrobat 8.0) Each object specifies a subject distinguished name (DN) acceptable for signing. More than one DN may be specified, but a signing certificate must satisfy at least one of the DNs by containing all the attributes specified in the matching DN. DN attribute restrictions are specified by adding them as properties. The properties’ key names can either be the corresponding attributes’ friendly names or OIDs (as defined in RFC 3280). The properties’ value must be of type string. For more information about the various attributes and their types, refer to RFC 3280.</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td>A URL that can be used to enroll for a new certificate if a matching one is not found, such as <a href="https://aardvark.corp.example.com/">https://aardvark.corp.example.com/</a>. Works in conjunction with urlType (if present). A degenerate use of this property is when the URL points to a Web service that is a digital ID store such as a roaming ID server. In that case, the URL indicates that as long as the signer has a digital ID from that Web service, it is acceptable for signing.</td>
</tr>
</tbody>
</table>
Authors can limit potential signers to individuals or groups as follows:

- **subject** limits potential signers to only those specified individuals. Signers could be limited to one or more people.
- **issuer** limits signers to those with certificates that chain up to a common, shared issuer. For example, all of a company’s employees may use the company’s certificate as an intermediate certificate and that certificate could be used as the issuer.
- **subjectDN** limits signers to those with certificates that match all the attributes of one of the listed DNs. For example:
  - `{cn:"Alice", ou:"Engineering", o:"Acme Inc"}`. For details about the friendly names of DN attributes (cn, o, ou, and so on), refer to the RDN Object in the *JavaScript for Acrobat API Reference*.
  - `{cn:"Joe Smith", ou:"Engineering", 2.5.4.43:"JS"}`, where OID 2.5.4.43 is used to carry out matching for the “Initials" attribute.

The following is sample code to define the above DN:

```javascript
var subjectDN = {cn:"Joe Smith", ou:"Engineering"};
subjectDN["2.4.5.43"] = "JS";
```

Attributes whose value is of type `DirectoryString` or `IA5String` can be specified as shown in the example above, whereas all other value types, e.g. `dateOfBirth` whose value is of type `GeneralizedTime`, the value needs to be specified as a hex encoded binary string.

To specify a certificate:

1. Create a signature field with an intuitive name and tooltip.
2. Get the required certificates and install them in some accessible location.
   - **Tip:** They must be in a `.cer` files in a DER format.
3. Create the JavaScript that gets the field object and uses the seed value method. Use `security.importFromFile` to get the DER-encoded certificates from their installed location (*Example 9.9*).
4. Add the `subject` and `issuer` properties to the `certspec` object.
5. Enter a flag value to indicate whether the value is required or not. Either or both the `subject` and `issuer` may be required.
6. Run the JavaScript, save the document, and test the field.
Controlling Signing with Seed Values

Example 9.9: Certificate issuer and subject seed value

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

var mySubjectCert = security.importFromFile("Certificate", "/C/Temp/nebwhifflesnit_DER.cer");
var myIssuerCert = security.importFromFile("Certificate", "/C/Temp/nebsCompany_DER.cer");

f.signatureSetSeedValue(
   {
      certspec: {
         subject: [mySubjectCert],
         issuer: [myIssuerCert],
         flags: 3
      }
   }
)
```

9.10.2 Specifying Certificates by Key Usage

Acrobat’s default signature handler allows signing with certificates where the **Key usage** field is *Sign transaction* or *Sign document*. However, the **keyUsage** seed value allows you to override the default behavior and limit signing to those certificates where the **keyUsage** is set to any value defined in RFC 3280 (see Table 9). While the seed value could be used to require or disallow any of RFC 3280 **keyUsage** values, the two most common cases allow or disallow *digitalSignature* (bits 2,1) (displayed as *Sign transaction* in Acrobat’s Certificate Viewer) or *nonRepudiation* (4,3) (displayed as *Sign document* in Acrobat’s Certificate Viewer). However, any combination of uses may be set.

To restrict signing to a certificate with a particular **keyUsage**:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.10).
3. Specify the **keyUsage** value in HEX:
   1. Specify 00, 01, 10, or 11 for each of the **keyUsage** values beginning with the least significant bit (the last one in the list in Table 9). For example:
      - *digitalSignature* is disallowed and non repudiation is required, and other values don’t matter: 111111111111110100. Convert to HEX: 3FFF4
      - *digitalSignature* is required and non repudiation is disallowed, and other values don’t matter: 111111111111110001. Convert to HEX: 3FFF1
   2. Remove the 3 and prepend the HEX value with 0x7FFF so it is in the correct HEX 32-bit format such as 0x7FFFFFF1.
   4. Enter a flag value to indicate whether the value is required or not. Set 32 if **keyUsage** is required and there are no other **certspec** properties.
   5. Run the JavaScript, save the document, and test the field.
Example 9.10: Certificate key usage seed value

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue({
  certspec: {
    keyUsage: [0x7FFFFFF1], //Set KeyUsage to "digitalSignature"
    flags: 32 //Require keyUsage
  },
});
```

9.10.3 Specifying Certificates by Policy

For legal reasons, policies are often associated with certificates. One way policies are identified is through an object identifier (OID), a unique series of numbers in the certificate policies' field that identifies the policy. Since an OID is always used with the issuer, authors can use this seed value pair when a company issues different certificates with different policies and it is necessary to restrict signing to certificates associated with a certain policy.

To restrict signing to a certificate containing a specific policy:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.11).
3. Specify the issuer.
4. Specify the oid. A policy OID is part of the value of the certificate's certificate policy field (Figure 33).
5. Enter a flag value to indicate whether the value is required or not. A value of 6 is recommended since issuer and oid must be specified together.
6. Run the JavaScript, save the document, and test the field.

Figure 33 Policy OID
Example 9.11: Certificate policy seed value

```javascript
var myIssuerCert = security.importFromFile("Certificate", "/C/Temp/nebsCompany_DER.cer");

// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue(
{
  certspec: {
    issuer: [myIssuerCert],
    oid: ["2.16.840.1.1.113733.1.7.23.2"],
    flags: 6
  }
})
```

9.10.4 Specifying a URL When a Valid Certificate is not Found

When a valid certificate is not found, users can be redirected to a URL during the signing workflow. The URL may be to a server with a certificate repository; or, more likely, the URL may be a link to a Web page describing how to obtain a new or valid certificate.

To specify a certificate with a specific policy:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.12).
3. Specify a certificate as described in one of the previous sections. Use `issuer` and/or `subject`.
4. Specify the URL. The URL can point to a certificate server or to instructions for getting a certificate.
5. Run the JavaScript, save the document, and test the field.

Example 9.12: Alternate certificate URL seed value

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

var mySubjectCert = security.importFromFile("Certificate", "/C/Temp/nebwhifflesnit_DER.cer");

f.signatureSetSeedValue(
{
  certspec: {
    subject: [mySubjectCert],
    url: "https://aardvark.corp.example.com/",
  }
})
```

9.10.5 Restricting Signing to a Roaming ID

Fields can be required to be signed with roaming IDs by specifying the `certspec url` and `urlType` properties. By providing the roaming ID server URL and the ASSP protocol as arguments, only roaming
IDs associated with the specified server will appear in the signing dialog’s digital ID drop-down list when a user attempts to sign the field.

To require signing only with a roaming ID:

1. Create a signature field with an intuitive name and tooltip.
2. Create the JavaScript that gets the field object and uses the seed value method (Example 9.13).
3. Specify the roaming ID server URL.
4. Specify ASSP as the URL type.
5. Run the JavaScript, save the document, and test the field.

Example 9.13: Roaming ID seed value

```javascript
// Obtain the signature field object:
var f = this.getField("mySigFieldName");

f.signatureSetSeedValue(
{
    certspec: {
        urlType: "ASSP",
    }
}
)
```

9.11 Custom Workflows and Beyond

Advanced document and workflow customization is beyond the scope of this document. However, keep in mind that Acrobat’s security APIs allow users many opportunities for customization. Document developers can easily create custom signing menu items, automate tasks, and perform other operations beyond those described in the preceding seed value sections.

For example, Example 9.14 performs a number of operations that would simplify signing operations in an enterprise setting. The script adds a Request Employee Signature to the toolbar and set up a number of automatic actions. When a user selects the menu item, a signature field with predefined properties is automatically created in the needed document location, and the field’s seed values are set.

**Note:** For more information, refer to the online Acrobat JavaScript Scripting Guide, JavaScript for Acrobat API Reference, PDF Reference, and the Acrobat SDK.

Example 9.14: Automating signing tasks

```javascript
//**************************************************************************
//File: seedValue.js
//Purpose: Demo how to set certificate constrictions into a signature field
//Steps: 1. Add a menu item under Tools, called Request Employee Signature
//      2. Add a signature and text field (for display) to the current open file
//      3. Set seed value
//      3.1 Wrap certificate object
//      3.2 Set seed value to the added signature field
```
Controlling Signing with Seed Values

Custom Workflows and Beyond

// reason: "I am approving this document"
// certSpec:
//   issuer: Example/MyCompanyCA (the root)
//   oid: 2.5.29.16 the oid of Example/MyCompanyCA
//   url: https://my.corp.example.com/
//   flag: 2 set limits on issuer
// 4. Display seed value added to the sig field to the added text field.
//**************************************************************************
// 1. Add a Tools menu item called Request Employee Signature
app.addMenuItem
({
cName: "Request Employee Signature",
cParent: "Tools",
cExec: "setSeedValues()",
cEnable: "event.rc = (event.target != null);",
nPos: 0
});

//Run function when menu item "Request Employee Signature" is clicked
function setSeedValues(){
//modify the following according needs
 var sigfieldName = "aSigField";
 var myReasons = ["I am approving this document"];
 var myIssuer;
 var oids = ["2.5.29.16"];
 var url = "https://seneca.corp.example.com/";
 var certSpecFlag = 2;//constricts on issuer, 6: issuer + OID, 1: users to
 sign, 7: issuer + oid + user
 var svFlag = 0; //no restrictions
 try{
 //2. add a sig field called "aSigField" and a text field
 var field = this.addField(sigfieldName,"signature",0,[180,640,352,680]);
 field.borderStyle = border.s;
 field.fillColor = color.ltGray;
 //a text field to display what seed values set to the sig field
 var textField = this.addField("aText", "text",0, [110,360,500,550]);
 textField.borderStyle = border.s;
 textField.fillColor = color.yellow;
 textField.multiline = true;
 textField.display = display.hidden;  //hiden form screen and print
 textField.setAction("MouseUp", "event.target.display = display.
 hidden;"); //click field, field disappears
 //3. set seed value
 //3.1 set up issuer's certificate object
 var myIssuerDN = {CN:"Enterprise Services CA", OU:"VeriSign Trust
 Network", O:"Example Systems Incorporated");
 var mykeyUsage = ["kDigitalSignature","kCRLSign"];  
 var myMD5Hash = "BF70 913F F8D6 D60A 47FE 8253, 3081 5DB4";
 var mySHA1Hash = "6b e8 46 06 39 f5 65 18 48 b2 f8 3a b1 46 3f 56 02 be 06
 c3";
 var myserialNumber = "3e 1c bd 28";
```javascript
var mysubjectCN = "Example Root CA";
var mysubjectDN = {CN:"Example Root CA", OU: "Example Trust Services", O: "Example Systems Incorporated", C: "US"};
var myusage = {endUserSigning: true};
var ExampleRootCertBinary = "308204A130820389A00302010202043E1CBD28300D06092A864886F70D01010505003069310440512D9E9B47DB42A57C1FC2A648B0D7BE92694DA4F62957C5781118DC8751CA13B2629D4F2B32BD31A5C1FA52AB0588C8";

// var myIssuer = {binary:ExampleRootCertBinary, issuerDN:myissuerDN, keyUsage: mykeyUsage, MD5Hash:myMD5Hash, SHA1Hash:mySHA1Hash, serialNumber:myserialNumber, subjectCN: mysubjectCN, subjectDN:mysubjectDN, usage:myusage};
var myIssuer = security.importFromFile("Certificate", "/c/test/root.cer"); // if import from an external reference

// 3.2 set up seed value
field.signatureSetSeedValue({reasons:myReasons, certspec: { issuer: myIssuer}, /*oid:oids, */url:url, flags:certSpecFlag}, flags:svFlag);

// 4. Display seed value added to the signature field to the new text field
var result = "";
var w = field.signatureGetSeedValue();
for (i in w)
    result += (i + " = " + eval("w." +i) + "\n");

var z = w.certspec;
for (i in z)
    result += (i + " = " + eval("z." +i) + "\n");

textField.value = result + "** Click on me to make me disappear **";
textField.display = display.show; // display what seed values were set

} catch (e){
    app.alert("setSeedValues(): " + e);
}
```
Like a conventional, handwritten signature, digital signatures identify the signer. However, digital signatures also enhance security because they store information about the signer as well as the signed document. For example, signatures can be used to verify signed content has not been altered, confirm the signer’s identity and to prevent the signer from denying their own signature. Before signing, review the Signing Basics and then refer to the following:

- “Signing With a Certification Signature” on page 216
- “Signing with an Approval Signature” on page 222

10.1 Signing Basics

10.1.1 Before You Sign . . .

Before signing, do the following:

- **Configure the signing application**: Both authors and signers should configure their application environment. For details, see “Setting up the Signing Environment” on page 170.
- **Obtain a digital ID**: Get a digital ID from your own organization, a 3rd-party provider or create a self-signed one.
- **Finish editing the document**: Sign only after making final changes. Post-signing changes may impact signature validity.
- **Pick a signature type**: Learn about approval and certification signatures so you know which to use.

10.1.2 Signature Types

A document can contain certification and/or approval signatures. Which signature type you need depends on the intent of both the author and the signer. Signature types include the following:

- **Certification Signature**: A certification signature provides a higher level of document control than an approval signature. Because it must be the first signature in a document, certification menu options are disabled if another signature is already present. Certified documents that have not been invalidated by illegal changes may display a blue ribbon icon next to the digital signature (Figure 36). Use certification signatures for the following:
  - When you as the document author want to attest to the document contents.
  - When you want to restrict the actions of future document recipients.
  - For documents that will be signed multiple times. You can specifically permit additional signatures so that the status of existing signatures is not impaired as signatures are added.
  - When you as the document’s author or creator are placing the logical "seal of authenticity" on the document; thereby declaring it an official document for you or your organization.
Approval Signature: An approval signature is any signature that was applied without choosing Certify Document. Any signature other the first one must be an approval signature. Use approval signatures for the following:

- For any signature other than the first.
- When you do not need to attest to the document content.
- When you do not need to restrict what a document recipient can do with the document.
- When you are approving a document or form for further processing; for example, a purchase order.

10.1.3 Signing User Interface

Signing features are accessible in several ways which vary depending on whether a document already contains signature fields or signatures:

- Pull down menus: Pull down menus provide menu items for signing, certifying, and working with signed documents. Items are enabled and disabled based on the current state of the document and what the author has allowed.
- Right click menus: For signed documents, right clicking on any signature in the Signatures pane or in the document displays a context menu. Menu items allow you to clear or validate a signature as well as to view the signature’s properties.
- Click on a signature field: Clicking on a signature field automatically invokes the Sign Document dialog.

10.2 Signing With a Certification Signature

Certifying a document enables the first signer attest to its contents and specify the types of changes permitted for the document to remain certified. Certification helps document authors and recipients determine that documents are legitimate and tamper-proof, thereby enabling trustworthy online transactions and more secure communications.

For example, suppose that a government agency creates a form with signature fields. When the form is complete, the agency certifies the document and allows users to change only form fields and sign the document. Users can fill in the form and sign the document, but if they remove pages or add comments, the document does not retain its certified status. Certifying a document helps ensure that it is not altered without the author’s approval.

Certified documents display the following (Figure 34):

- Blue ribbon icon: An icon appears next to the digital signature and in the Signature tab.
- Document restrictions: The Signature tab displays the certifier-specified restrictions.
- Explicitly trusted but potentially dangerous content: A list appears in the Signature tab, if any.
- Certification attestation: Depending on user preferences (see , a signer-specified reason for signing may appear in the Signature tab and the signature appearance.

Before certifying, be aware of the following:
• Certification locks certain document elements and limits what a document recipient can do with it.
• Because certification is designed to carry more legal weight than an uncertified document, greater attention to the content and process is typically warranted.
• Certification signatures are automatically validated even if the application preference to automatically validate signatures is turned off.

**Document Locking**

Certification limits what a recipient can do with a document. Some actions are locked automatically, and some are locked by the certifier. For example, during certification the signer can choose from the following options:

• No changes allowed
• Form fill-in and digital signatures
• Annotations, form fill-in, and digital signatures

General editing, adding or removing pages, and so on are automatically prevented. Any changes that are explicitly locked by the certifier or automatically prohibited by the application invalidate the certifier's signature and revokes the document's certification.

**Document Defensibility**

Acrobat has a notion of a document’s defensibility which is defined by the features that appear in the legal attestation dictionary, described in Section 8.7.4 of the *PDF Reference* manual. Note that aside from when a signer is certifying, Acrobat does not actively inform the user about the document’s legal defensibility.

In any case, a document’s legal defensibility improves if it does not contain content that threatens the signer’s ability to see what they are signing as well as their ability to certify that what the document recipient sees is the same as that which was certified. Such content includes JavaScript, multimedia, and so on. It is the certifier’s responsibility to either remove that content or attest to the fact that such content should be retained.

Hazardous content is revealed to users in two ways:

• Acrobat helps the signer identify such content by scanning the document during the certification signing process. The signer is given the option to embed an attestation in the document about that content that explains why it is present. This behavior is unique to certification signatures and does not apply to approval signatures.

• Document recipients use the **View Document Integrity Properties** button to launch the same content scanning process that was automatically launched when the certifier signed. If the document is certified, the process generates a report that includes the certifier’s attestation, if any. Content that has been explicitly trusted by the certifier also appears on the signature tab under **Trusted Content** (Figure 34).
Legal Attestations and Warnings Comments

For documents with dynamic content, signer’s may want to add a legal attestation or comment indicating the included content has been reviewed is specifically permitted. A legal attestation can only be added on certified documents and during signing. When this option is enabled by the signer’s application settings, the Certify Document dialog displays a Review button which invokes the PDF Signature Report dialog. The dialog display a Warnings Comment field that allows the signer to choose from a default comment or to create a custom comment.

The Enable Reviewing of Document Warnings and Prevent Signing Until Document Warnings Are Reviewed settings function in tandem and should be set together. Setting both these options to Always results in the highest degree of assurance that the signing process is not adversely impacted by malicious content. For details, see “Setting up the Signing Environment” on page 170.

10.2.1 Certification Workflow for Documents with Multiple Signers

Certification allows document authors to define what changes are legal (possible), and it allows the recipient to identify whether a document’s problematic features (content that could change the document appearance) originated with the certifier or not. More importantly, this gives the recipient the assurance that if these features in the document are indeed malicious, the certifier can be proven to be at fault. The recommended workflow for defensible signatures can then be described as follows:

1. The document author adds the requisite form fields and any other document customizations. Preventing certain future actions (e.g. to form fill in and signing) can be accomplished ahead of time via JavaScript or during signing.

2. The document is signed with a certification signature. If there is problematic content in the document, the certifier chooses Review and adds a Warnings Comment explaining why the content is OK.

3. The document is routed to the next person in the workflow.

4. The document recipient manually validates the certification signature if the application is not set up to validate signatures automatically.

5. Document integrity is verified by right clicking on the certification signature and then choosing Show Signature Properties > Legal tab > View Document Integrity Properties. This action invokes the PDF Signature Report dialog which displays a list of problematic content as well as the...
certifier’s comment about that content (if any). For example, a certifier might state why they have added a link to a corporate web site, JavaScript, or some other item.

**Note:** The certifier’s warning comment is not viewable via preview mode.

6. The recipient decides whether or not to continue modifying and signing the document based on the list of warnings and certifier’s warning comment (if any).

7. The recipient modifies the document if it is permitted by the certifier (for example, filling in a form).

8. The recipient signs with an approval signature and forwards it to the next recipient (if any).

### 10.2.2 Setting up a Document for Certification

**Authoring Form Fields**

When a certified document contains more than one form field, field properties such as locking, placement, naming, tooltips, and even appearance should be specified in ways which help the recipients understand the form and easily determine whether or not data changes have invalidated the signature and certification. For details, see “Authoring Signable Forms” on page 186.

**Using Seed Values to Individual Form Fields**

You can customize the way a certified document behaves for signers by giving form fields additional features with seed values. For example, you can preconfigure custom signing reasons or limit signing to only those with certificates with predefined characteristics. Certifying a Document

**Before continuing:**

- Configure your application as described in “Setting up the Signing Environment” on page 170.
- Prepare the document for certification as described in “Setting up a Document for Certification” on page 219.

1. Initiate the certification process by doing one of the following:
   - Right clicking on a signature field and choosing **Certify with a Visible Signature** (Figure 38).
   - Choosing a menu item:
     - Advanced > Sign & Certify > Certify with Visible Signature
     - Advanced > Sign & Certify > Certify without Visible Signature

   **Note:** Selecting a field results in signing that field. When a field is not preselected, choosing one of these menu items invokes a dialog which asks you what field you would like to sign.

2. (Optional): If your application is configured to automatically enter preview mode during signing, do the following:
   1. Review the text in the Document Message Bar at the top of the document.
   2. Choose **View Report** to invoke the PDF Signature Report dialog. Acrobat checks to see if the document contains dynamic content that could adversely impact the integrity of the document. In general:
      - If no errors are listed, the document can always be signed.
• If only errors with a green check appear, then the document contains rich content that can be suppressed in preview mode. These documents can be signed safely in preview mode.

• If any error with a red X appears, then the document contains rich content that can not be suppressed in preview mode. The document should not be signed by signers who are concerned that such content may affect how the document appears to the signer or document recipient.

3. Review the warnings and decide whether the document is OK to sign. Choose Close when done.


3. Configure the Certify Document dialog:
• Digital ID: Select a digital ID. The digital ID selected as the default for signing is automatically selected. For details about changing the default, see “Specifying Digital ID Usage” on page 351.

• Password: Enter a password if the selected digital ID requires it.

• Appearance: Select an appearance or use the default one.

• Reason: If the application is configured to display the Reason for Signing Document field, choose an item from the list or enter a new reason.

• Location and Contact Info fields: If desired, fill in these optional fields.

4. Set Permitted Changes after Certifying:
• No changes allowed: Prevents users, JavaScript, and other hazardous content from changing the document. Since potentially hazardous content is prevented from interacting with the document, that content will not appear in the Signature pane’s Trusted Content list.

• Form fill-in and digital signatures: Limits user interaction to adding data to form fields, including signatures.

• Annotations, form fill-in, and digital signatures: Limits user interaction to adding data to form fields, signing, and commenting.

  Tip: If the document contains form fields, specify the settings that make the most sense for the intended workflow. Keep in mind a signature field is a form field. For details, see “Making a Field a Required Part of a Workflow” on page 188.

5. If the Review button appears on the dialog (an application setting), choose Review. This action invokes the PDF Signature Report again and enables adding a warnings comment or legal attestation (another application setting).

  Tip: If you have already reviewed the warnings and don’t need to add a comment, skip this step.
6. If there are any document warnings in the PDF Signature Report, do the following:
   • Review the warnings and determine whether it is acceptable to certify the document as is. If not remove the problematic content and start over.
   • If the content is ok, enter a **Warnings Comment** for the document recipient. Select the default or enter a custom comment. A comment should tell the reader why the content is there and that you have approved it.

7. Choose **OK**.

8. Choose **Sign**.

9. Save the document.

10.2.3 Certifying a Dynamic Form

This information pertains only to LiveCycle Forms Designer which ships with Acrobat.
Because certification is designed to provide a higher level of assurance about document integrity as well as define the boundaries of permitted changes, certifying a dynamic form requires special procedures. Dynamic forms can contain behaviors that prevent certification. Dynamic forms may need to be configured to support certification.

To configure a dynamic form for certifying: Choose File > Form Properties and display the Defaults tab. In the Scripting panel, set Preserve Scripting Changes to Form When Saved to Manual. When the form is subsequently opened in Acrobat or Adobe Reader (with signing rights), certification will be possible.

10.2.4 Why Can’t I Certify?

In order to certify a document, the certifying signature must be the first one in the document and there must be no restrictions on the document that prevents certifying. Either of these two situations may arise if you are not the author of the document. When a document is ineligible for certification, the certification user interface items are disabled.

In order to certify the document, clear existing signatures, remove the restrictions if you have permission to do so, or save the document under a new name so that you are the document author.

10.3 Signing with an Approval Signature

Documents may be signed with simple approval signatures. When a document is part of a workflow where document elements do not need to be locked, use an approval signature. PDFs can be signed in Acrobat, in Reader (in special cases), or in a browser.

10.3.1 Signing Documents in Acrobat

You can create a new signature field or sign an existing one:

- If your document already contains a signature field, simply click on it and follow the instructions or choose a Sign Document menu item.

- If you want to create a new field on-the-fly as part of the signing process, choose a Place Signature menu item. You may want to read Chapter 8, “Authoring Signable Documents” if you would like to control how the document behaves once it is signed.

Before continuing:

- Configure your application as described in “Setting up the Signing Environment” on page 170.
To sign a document with an approval signature:

1. Initiate the approval signing process by doing one of the following:
   - Sign an existing field:
     - Click on a signature field.
     - Right click on a signature field and choose **Sign Document** (Figure 38).
   - Sign a new field by choosing **Advanced > Sign & Certify** and then selecting **Approval**.

   **Tip:** Certification signature menu items are disabled if the document has already been signed.

2. Choose **View Report** to invoke the PDF Signature Report dialog. Acrobat checks to see if the document contains dynamic content that could adversely impact the integrity of the document. In general:
   - If no errors are listed, the document can always be signed.
   - If only errors with a green check appear, then the document contains rich content that can be suppressed in preview mode. These documents can be signed safely in preview mode.
   - If any error with a red X appears, then the document contains rich content that can not be suppressed in preview mode. The document should not be signed by signers who are concerned that such content may affect how the document appears to the signer or document recipient.

3. Review the warnings and decide whether the document is OK to sign. Choose **Close** when done.

4. Choose **Sign Document**.
4. Configure the Sign Document dialog:
   - **Digital ID**: Select a digital ID. The digital ID selected as the default for signing is automatically selected. For details about changing the default, see “Specifying Digital ID Usage” on page 351. If the desired digital ID is not listed, select **Refresh ID List** if your digital ID is on a hardware device which you recently connected, or select **New ID** to create or install a new digital ID now.
   - **Password**: Enter a password if the selected digital ID requires it.
   - **Appearance**: Select an appearance or use the default one.
   - **Reason**: If the application is configured to display the **Reason for Signing Document** field, choose an item from the list or enter a new reason.
   - **Location** and **Contact Info** fields: If desired, fill in these optional fields.

5. Choose **Sign**.

6. Save the document.

### 10.3.2 Signing in a Browser

To sign a document in a browser, the document must contain an empty signature field:

1. Click on any signature field or choose **Pen Icon > Sign Document** on the Tasks toolbar, and then follow the steps described in **Signing Documents in Acrobat**.

2. To retain a copy of the signed document, choose the **File > Save A Copy**.
### 10.3.3 Clearing One or More Signatures

Clearing a signature field deletes the signature but leaves the empty field. Not all signatures can be cleared. You may be prevented from deleting the signature in the following cases:

- You cannot delete someone else’s signature.
- If the author of a signature field has marked it to become read-only after it is signed, it can only be cleared by the author.

To clear all signature fields in a document, do one of the following:

- Choose **Advanced > Sign & Certify > Clear All Signatures**.
- In the Signatures tab, choose **Options > Clear All Signatures**.

To clear a single signature field:

1. Right click on a signature.
2. Choose **Clear Signature Field**.
Personalizing Your Signature Appearances

Personalized signature appearances allow you to provide additional information about yourself, affiliations, or company. Home users often modify the default appearance as a matter of personal taste, while enterprise users are sometimes provided with a company-specific style and logo.

Every aspect of the visible appearance is subject to customization, and the options range from using logos, a replica of your handwritten signature, photos, images, and text. A typical appearance consists of three components, and each can be separately customized:

- **Signature**: A graphic that identifies the signer on the left-hand side of the appearance, such as a photo or scanned signatures. Transparent backgrounds allow a watermark to be visible in the underlying layer.
- **Signature details**: Signature data that the signer wants to appear to the right of the signature.
- **Watermark or logo**: An image that appears behind the signature. The default is the PDF logo.

You can create any number of appearances ahead of time for later use, and the signing workflow allows you to select one from your library appearances. However, you can also just use the default appearance created from your name or create one on-the-fly at signing time.

When you sign a PDF file, the appearance becomes part of the signed document. It is not part of the signature.

**PDF/A compliance**

PDF/A is a subset of PDF intended as a file format suitable for the long-term archiving of electronic documents. Appearances in PDF/A-compliant documents must be must conform to the following:

- Embed fonts. PDF/A requires font embedding because font metrics and glyph compliments change over time. Long term archiving can not rely on anything not included in the document.
- Colors should use DeviceRGB when signing RGB/Office docs or else with associated ICC profiles.

### 11.1 Customizing a Signature Appearance

Users generally customize one or more signature appearances and store them for later use. Available signatures are listed in the appearance panel at signing time (Figure 40).

To customize a new signature appearance:

1. Choose **Edit > Preferences** (Windows) or **Acrobat > Preferences** (Macintosh).
2. Choose **Security** in the left-hand list.
3. In the Appearance panel, choose **New** or **Edit**.
4. Configure the signature appearance **Title**: Any title used to identify the appearance. These are useful for identifying which appearance to use for particular workflows.

![Figure 41  Signature appearance: Title](image)

5. Set the graphic options in the Configure Graphic panel
   - **No graphic**: No graphic is used.
   - **Imported graphic**: Choose *File > Browse*, select a file and choose *OK*.
   - **Name**: Your text extracted from the signing certificate.

![Figure 42  Signature appearance: Import graphic](image)

6. Set the text fields to display in the appearance in the Configure Text panel:
   - **Name**: The name associated with the certificate.
Personalizing Your Signature Appearances

Customizing a Signature Appearance

- **Date**: The date signed. Signature appearances can only display local (computer) time, and it will likely differ from that in the Date/Time tab on the Signature Properties dialog when a timestamp server is used.

- **Location**: The location associated with the identity configured in Acrobat.

- **Reason**: The reason for signing.

- **Distinguished name**: A name with details such as country, organization, organizational unit, and so on.

- **Labels**: A label for each of the items above. For example, Reason.

**Figure 43 Signature appearance: Text fields**

7. **Logo**: The logo or graphic used as a background watermark. The default watermark is the Adobe PDF logo. Your watermark will be shared automatically in all of your signature appearances.

**Figure 44 Signature appearance: Logo or watermark**

8. Set the text direction and character set in the Text Properties panel:

- **Text Direction**: Choose a direction appropriate for the signer’s language.
9. Choose **OK**.

**Figure 45 Signature appearance: Text properties**

- **Digits:** If languages are installed that use digits other than 1234567890, the drop-down list will be populated with alternate choices. Choose a digit set appropriate for the signer’s language.

### 11.2 Changing the Logo (Watermark or Background)

A watermark is a partially transparent graphic or logo that appears “behind” a signature. By default, the watermark is the Adobe PDF logo. Line (vector) art that is simple and unobtrusive often works best.

1. Import a logo or create a new one in a program such as Adobe Illustrator.
2. Illustrator instructions: Set a low transparency level and flatten the transparency:
   1. Select all and group the objects if there is more than one.
   2. Choose **Window > Transparency** and move the slider to some low value such as 20%.
   3. Choose **Object > Flatten Transparency**. Leaving the Raster/Vector balance at 100%.
   4. Save the file as a PDF.
3. Open the PDF file in Acrobat.
4. Crop the page and remove white space.
   
   **Note:** The method varies across product versions. For example, for 8.x, choose **Document > Crop Page** and check **Remove White Margins**.
5. Save the file as **SignatureLogo.pdf** in:
   - **Windows:** C:\Documents and Settings\<user>\Application Data\Adobe\Acrobat\<version>\Security.
   - **Macintosh:** \Users\<user name>\Library\Application Support\Adobe\Acrobat\<version>\Security.
11.3 Creating appearances from handwritten signatures

You can use your handwritten signature as the graphic displayed by your signature appearance as follows:

1. Scan your signature at a relatively high DPI; for example, 300.
2. Convert the image to vector drawing.
3. Make the background transparent.
4. Save the image in the desired format. A wide variety of formats is supported.

Figure 46 Handwritten signature appearance

Sandra Sample
When you receive a signed document, you may want to validate its signature(s) in order to verify who the signer was, when they signed it, and what was actually signed. Depending on how you have configured your application, validation may occur automatically.

However, understanding both how to validate a signature manually as well as what signature components are analyzed during the validation process can facilitate trouble-free workflows and mitigate signature status problems. Participants in signing workflows may also want to configure their environment to streamline the validation process and control what kinds of content in signed documents can be run on their machine.

The following sections provide validation details:

- “Signature Validity Basics” on page 232
- “Setting up Your Environment for Signature Validation” on page 234
- “Validating Signatures Manually” on page 238
- “Status Icons and Their Meaning” on page 246
- “Troubleshooting a Signature or Document Status” on page 247
- “Document Behavior After Signing” on page 256

12.1 Signature Validity Basics

As part of the signature validation process, Acrobat and Adobe Reader verify the signer’s identity as well as the document’s integrity.

12.1.1 What Makes a Signature Valid?

Signature validity is determined by checking the signature’s digital ID certificate status (is it valid and trusted?) and document integrity (has it changed since being signed):

- Authenticity verification confirms that the signer’s certificate or that one of its parent certificates exists in the validator’s list of trusted identities and that the signing certificate is valid at that point in time according to the user’s Acrobat or Reader configuration (time of signing, secure timestamp time, or current time).
- Document integrity verification confirms that the signed content hasn’t changed after it was signed or that it has only changed in ways specifically permitted by the signer.

12.1.1.1 Authenticity Verification

Authenticity verification starts with a signer obtaining a digital ID that includes an X.509 certificate. The validator must add that certificate (or have previously added one of its issuing certificates) to their trusted identities list. Either the signer’s certificate or one of its issuing certificates must then be explicitly trusted for signing, thereby making it a trust anchor used during signature validation. At
validation time, the certificate is processed and analyzed to see if it’s valid. That is, Acrobat performs a revocation check and other relevant operations before determining what the signature status will be.

**Figure 47 Internal Document Signature components**

12.1.1.2 Document Integrity Verification

In signing workflows, document integrity refers to whether or not what was signed has changed after signing in a way that violates any document rules. That is, what the signer signed should be reproducible and viewable on the document recipient’s end. At a high level, the Acrobat family of products therefore implements signatures as follows:

- Each signature captures what the document looked like at the signing point in time.
- Only very limited changes are possible after a signature is applied. At most, form field values, additional signatures, and annotations can be changed or added.
- View Signed Version shows exactly what was signed. The signature panel lists post-signing changes.
- A certification signature can tighten the rule to allow less changes than form fields, additional signatures, and annotations.

To verify if a document has changed after signing (has integrity), Acrobat or Adobe Reader must have a way to uniquely identify what was signed. To do this, it uses a *message digest*. A message digest is a number which is created algorithmically from a file and which uniquely represents that file. If the file changes, the message digest changes. Sometimes referred to as a *checksum* or *hash*, a message digest is simply a unique number created at signing time that identifies what was signed and is then embedded in the signature and the document for later verification.

During the act of signing, the application creates a message digest and then encrypts that digest with the signer’s private key. The digest is embedded in the document along with the signature appearance. Every time a document is signed, a new digest is created. Thus, each signature is only valid for a specific version of the document.
Because the application stores and numbers a document version for each signature, signature validators can determine what was actually signed. When you validate a signature, a new message digest is created and compared to the digest that was embedded in the document at signing time. If the two digests are not identical the signature is invalid.

Both signers and signature validators should understand the following about the relationship between signatures and document versions:

- Every time a document is signed, the document’s state at the point of signing is stored in the PDF.
- Versions are incrementally numbered beginning with “1.”
- A document with 10 signatures will have 10 versions.
- A signature applies to a version (e.g. signature X with version X and signature Y with version Y, etc.).
- When you open a document in Acrobat or Adobe Reader, the current version always displays.

Note: To learn more about how each signature results in a new version of the document, refer to http://www.adobe.com/devnet/acrobat/pdfs/DigitalSignaturesInPDF.pdf.

12.2 Setting up Your Environment for Signature Validation

Document recipients should configure their environment to handle incoming documents in a way that enhances workflow efficiency or meets some business need. While Adobe Acrobat and Adobe Reader provide default options, customizing the environment often provides a better user experience. In large, enterprise environments, your environment may be preconfigured by a system administrator.

Options include the following:

- **Validating Signatures Automatically**: By default, validation occurs automatically. If signatures should not be validated automatically when a document opens, turn this option off.
- **Method, Revocation Checking, and Validation Time Preferences**: Accept the defaults or configure plugin usage, time display, automatic revocation checking, and other settings.
- **Using Root Certificates in the Windows Certificate Store**: If you would like to trust and use certificates in the Windows Certificate Store for signature validation, turn this option on. Trusting all of these certificates is not recommended.
- **Certificate Trust Settings**: Specify whether a certificate should be a trust anchor, trusted for signing, and trusted for certain behaviors in certified documents.

12.2.1 Validating Signatures Automatically

By default, signatures are automatically validated. However, you may want to turn it off because:

- You don’t care whether the signatures are valid.
- The desktop cannot be configured to validate the signature.
- To gain a small increase in application speed when it opens a document. The difference may be negligible depending on the number of signatures and whether a system administrator has customized revocation checking.

To configure automatic signature validation:
Validating Signatures Method, Revocation Checking, and Validation Time Preferences

1. Choose Edit > Preferences (Windows) or Acrobat (or Adobe Reader) > Preferences (Macintosh).

2. Choose Security in the left-hand list.

3. Check Verify signatures when document is opened.

12.2.2 Method, Revocation Checking, and Validation Time Preferences

To set advanced digital signature preferences:

1. Choose Advanced Preferences.

2. Display the Verification tab.

Verification tab options let you specify the validation plugin, default validation methods, whether or not certificate revocation checking is automatic, what time is associated with a validated signature, and whether or not a status icon appears with the signature.

Figure 48 Signature verification preferences

3. Select the signature validation method (use the default setting unless instructed to change it by a system administrator):
   - Use the document-specified method, prompt if it is not available.
   - Use the document-specified method, use the default method if it is not available.
   - Always use the default method (overrides the document-specified method).

Tip: Don’t change this setting unless instructed to do so by a system administrator. Signatures are created and validated by plugins. These options specify which plugin is used. Both Acrobat and Adobe Reader provide a default plugin for signing documents.
Validating Signatures Using Root Certificates in the Windows Certificate Store

and verifying signatures. While the signing and verification usually use the same plugin, this is not always the case. However, a signature always “knows” what plugin is required to verify it.

4. If you have installed a non-default plugin, select your preferred method for verifying signatures.

5. Check or uncheck **Require that certificate revocation checking be done whenever possible during signature validation**.

   This option checks certificates against a list of revoked certificates during validation, either with the Online Certificate Status Protocol (OCSP) or the Certificate Revocation List (CRL). If this option is not selected, the revocation status for approval signatures is ignored. *Revocation checking always occurs for certificates associated with certification signatures.*

   **Note:** Signature verification is similar to credit card validation. OCSP checking is like making a phone call to verify the card number. CRL checking is like checking the card numbers against a list.

6. In the Verification Time panel, select a time verification method:
   - **Current time**: The digital signature validation time.
   - **Secure time**: The secure timestamp server time if one is present and trusted, otherwise the current time.
   - **Creation time**: The signature creation time. This may be synonymous with the secure time if a timestamp was used.

**Time verification changes from 9.0 to 9.1**

With 9.1, the default preferences assure signatures will be valid if the certificate used was valid at the time of signing rather than valid at the time of checking the validation. So by default, a signature will remain valid in the long term even after the certificate has expired. But it also means that, as well as trusting the signature, you are also trusting the document’s signing time.

**Tip:** It is possible for a signer to change their system date to a time when a certificate was valid and then sign the document, which would then lead to misleading results. Hence, it is suggested that signatures be configured to use time stamp servers.

In Acrobat 9.0, if the verification time preference is set to "The time at which the signature was created", then the CRL validity window had to include the time-of-signing in order to be used in signature validation. In Acrobat 9.1, this behavior is changed so that the CRL validity window overlaps the interval from the time-of-signing to the end of validity of the signing certificate. This change allows CRLs issued after the time-of-signing to be more consistently used. This change can result in a behavior change where signatures that could not be validated (but were in the problematic state) can now be successfully validated, yielding a valid or invalid signature status.

12.2.3 Using Root Certificates in the Windows Certificate Store

The Windows Certificate Store contains a store called “Trusted Root Certificate Authorities” that contains numerous root certificates issued by different certification authorities. Certificates are “root” certificates by virtue of being at the top of the certificate chain hierarchy. There are two common ways a certificate ends up in the Windows Certificate Store root directory:

- The computer manufacturer or Microsoft has put them there.
A company administrator has put them there as part of a company-wide program. Most home users should not trust all Windows root certificates by default because by trusting a root certificate you may be trusting all the content provided by the company that owns that certificate. Many root certificates ship with Windows, and users may have imported others as a result of some online action.

Enterprise users, on the other hand, should consult company policy to determine whether or not to trust all Windows root certificates for validating signatures or certifying documents. This information should come from an administrator, though your application may already be configured with the correct settings. A common reason to trust Windows roots is so the administrator can manage from a central location the certificates deployed on a network.

To use these certificates for signature validation:

1. Display the Windows Integration tab.

2. Specify the trust level for all root certificates in the Windows Certificates Store:
   - **Validating signatures**: Certificates will be trusted for approval signature validation.
   - **Validating certified documents**: Certificates will be trusted for certification signature validation.

3. Choose **OK**, and exit the preferences dialogs.

### 12.2.4 Validating Signatures with Timestamps and Certificate Policies

Certificate policies can be used with timestamps, but they can only be verified on the client end, not on the server end. That is, a timestamped signature can not be sent with CRL request with a specific policy OID; however, the client can require that the server response include a specified policy constraint. If the timestamp server returns a response that doesn’t include a matching policy OID, then the client would reject the timestamp and its status would be invalid. The user interface shows the following:
Validating Signatures Manually

- The signature could be valid, but it's validated at the current time. The Signature pane shows the appropriate icon.
- The timestamp is invalid. The Summary tab of the Signature Properties dialog shows a red X

The require a timestamp to be associated with a particular certificate policy:

1. Configure your application to validate signatures using Secure Time as described in “Method, Revocation Checking, and Validation Time Preferences” on page 235.

2. Configure a policy constraint for a trust anchor in your trusted identities list:
   1. Choose Advanced > Managed Trusted Identities.
   2. In the Display drop down list, choose Certificates.
   3. Select the timestamp server's certificate that will be used as a trust anchor.
   4. Choose Edit Trust.
   5. Choose the Policy Restrictions tab.
   6. Enter a certificate policy OID.
   7. Choose OK.

   **Note:** If the timestamp server returns a response with a policy not specified by the client, the timestamp signature will be invalid due to an invalid policy constraint.

12.3 Validating Signatures Manually

Unless the application is configured to do otherwise, signatures are validated automatically when a document opens. If they are not validated or if a signature needs to be revalidated, you can validate one or more signatures manually.

Validating a signature allows you to verify the signer’s identity and determine whether the displayed document is identical to what was signed (or that only allowed changes were made):

- Identity verification confirms the signer's certificate or one of its parent certificates exists in the list of trusted identities and is not expired or revoked.
- Document integrity verification confirms that the signed content hasn’t changed since signing or that it has only changed in ways specifically permitted by the signer. Signatures can be validated one at a time or all at once.

Before validating a signature, it is a good idea to understand what a signature is and how signature status is indicated. For details, see the following:

- “What Makes a Signature Valid?” on page 232
- “Status Icons and Their Meaning” on page 246
12.3.1 Validating Signatures with Adobe Reader

The process for validating one or more signatures in Adobe Reader is similar to Acrobat. However, the top level menu item is labelled Document instead of Advanced. Therefore, the validation paths are as follows:

- **Document > Sign > Validate All Signatures**
- Click on a signature in the document or the Signatures pane, right click, and choose **Validate Signature**.

12.3.2 Validating a Single Signature in Acrobat

Signatures can be validated one at a time or all together as described in “Validating All Signatures in Acrobat” on page 239. Signature validity can be determined by viewing its associated icon. A green check mark indicates the signature is valid without reservations. Other icons indicate there may be a problem.

There are several ways to verify a signature manually:

- Right click on any signature in the Signatures pane or in the document, and choose **Validate Signature**.
- Highlight a signature in the Signatures tab, and choose **Advanced > Sign & Certify > Validate Signature** or open the Signature Properties dialog and choose **Validate Signature**.

![Figure 50 Signatures tab: Validate signature](image)

12.3.3 Validating All Signatures in Acrobat

All signatures in a document may be validated simultaneously. This feature is particularly useful if the auto-validate option has been turned off.

To validate all signatures:

1. Choose **Advanced > Sign & Certify > Validate All Signatures**.
2. If a dialog appears asking if all signatures should be validated, choose OK. This dialog does not appear if you have previously checked Do not show this message again.

Figure 51 Validate all signatures dialog

![Validate all signatures dialog](image)

3. If a dialog appears confirming all signatures have been validated, choose OK. This dialog does not appear if you have previously checked Do not show this message again.

Figure 52 Signature validation confirmation

![Signature validation confirmation](image)

### 12.3.4 Validating an Problematic Signature (trusting a signer on-the-fly)

If a signer’s digital ID certificate has not been explicitly trusted, the signer is untrusted and the signer’s signature validity will be problematic. When a signer has not been trusted ahead of time, you can trust their certificate for signing and certifying directly from the signature. After their ID (contact information and certificate) is added to your list of trusted identities, the signature can be validated.

To add an someone’s certificate a list of trusted identities:

1. Display the Signature Properties dialog by right clicking on any signature in the document or the Signatures tab and choosing Show Signature Properties.

2. Choose the Summary tab (Figure 53).
3. Choose **Show Certificate**.

   Adding an unverified digital ID certificate to the trusted identity list could pose a security threat. This is particularly true for self-signed IDs that are not issued by a third-party certificate authority. For details, see "Verifying the Identity of Self-Signed Certificates" on page 250.

4. When the Certificate Viewer appears, choose the Trust tab (Figure 54).

5. Choose an item in the left-hand certificate path field. There may be one or more certificates which make up a certificate chain.

   **Tip:** If the bottom-most certificate on the chain is selected, then only that certificate will be trusted. If the top-most certificate is selected, then any certificates having that certificate as a root will be trusted. For example, if the root certificate is issued by VeriSign and it is trusted, then other certificates having VeriSign's certificate as the root (also issued by them) will also be trusted. It is a best practice to trust the topmost certificate that you are willing to trust. Revocation checking starts at the bottom of a chain (begins with the end entity), and once it reaches a trusted root revocation checking stops.
6. Choose **Add to Trusted Identities**.

7. When asked if the certificate should be trusted, choose **OK**.

8. When the Import Contact Settings dialog appears, configure its trust levels. For details, see “Certificate Trust Settings” on page 337.
   
   The Policy Restrictions tab will not appear if there are no policies associated with this certificate.

9. Choose **OK**.

10. Choose **OK**.

11. Choose **Close**.

12. Right click on the signature and choose **Validate Signature**.

   **Tip:** The yellow triangle icon on the signature will not change until the signature is revalidated.
Validating Signatures for other Document Versions

When you open a document, the latest version is always displayed. You can see whether the signature associated with earlier signed versions of the document are valid simply by opening the Signature pane and viewing the status icon and text.

Documents with multiple signatures contain the elements needed to reconstruct any previous version of itself as it existed at the time of signing. In other words, Acrobat and Adobe Reader “remembers” that version A is signed, that changes were made to version B, and so on. Therefore, it may be necessary to view the signed version in order to see what content was actually signed. Viewing the signed version allows you to check if the signature is valid for a particular document version.

To view the signed version of a document.

1. Right click on the signature you want to validate in the document or in the Signatures tab.
2. Choose View Signed Version. The application opens the signed version of the document.
3. Revalidate the signature if necessary.

Tip: For more about versioning, see “Document Integrity Verification” on page 233.

Validating Signature Timestamps

If you know a signature is timestamped or your workflow requires timestamps, read the following sections. At a high level, the rules are as follows:

- You can configure Acrobat to use timestamps by setting the verification preferences as described in “Method, Revocation Checking, and Validation Time Preferences” on page 235.
- If set, the secure timestamp server time is used if one is present and trusted, otherwise the current time is used.
- Timestamp validity does not affect signature validity. A signature can be valid even if the timestamp server’s certificate is invalid or expired.
- The signature validation time appears in the Date/Time tab of the Signature Properties dialog.

Local Time versus Timestamp Time

Signature times tell you that a document and signature existed prior to the indicated time. All signatures are associated with the signer machine’s local time, but they may also include a timestamp time if the signer’s application was configured to use a timestamp server. Because users can set their machine time forward or back, local time is less reliable than a timestamp time. Local times are labelled as such in the Date/Time and Summary tabs of the Signature Property dialog (Figure 56).

Note: Because signature appearances only display local time, the appearance time will be different from the timestamp time shown in the Date/Time tab of the Signature Properties dialog.

Figure 56 Timestamps: Local, machine time
What is a timestamp?

A timestamp is like a signature inside of a signature. Like signatures, timestamps are provided by someone (a timestamp authority) who uses a certificate to confirm their identity. A timestamp’s certificate must be valid (not revoked by the issuer) and trusted (by you) for the timestamp to be valid. Timestamp certificate status appears in the Date/Time and Summary tabs of the Signature Property dialog:

- Untrusted timestamp certificates appear as follows:

  **Figure 57  Timestamps: Untrusted stamp**

  ![Signature is timestamped but the timestamp could not be verified.]

- Trusted timestamps that have been added to the Trusted Identities list and have been explicitly trusted for signing appear as follows:

  **Figure 58  Timestamps: Trusted stamp**

  ![Signature is timestamped.
  Date: 2015/06/13 13:17:13 -0700]

How Do I Validate a Timestamp in a Signature?

Validating a timestamp is the process by which you check to see if the timestamp was applied and that its certificate is valid. In order to validate a timestamp, you need to manually verify:

- **The timestamp was applied**: If a timestamp fails for some reason (the server cannot be found, the network is down, etc.), timestamping fails silently.
- **The timestamp certificate is trusted**: If a timestamp was applied, the certificate must be trusted by adding to your trusted identities list.

To verify that a signature has been properly timestamped:

1. Display the Signature Properties dialog by right clicking on any signature in the document or the Signatures tab and choosing **Show Signature Properties**.

2. Choose the Date/Time tab. Timestamp status is indicated by the icon and associated text:

   - **Warning triangle**: Timestamping failed or a timestamp is not present and the local time is used. Verify the signer used a timestamp.
   - **Magnifying glass**: A timestamp may have been used but you have not yet trusted the timestamp certificate. Proceed to the next step.
   - **Clock**: A timestamp with a trusted certificate was used.
Figure 59  Timestamps: Date/Time tab

Note: The following steps add a timestamp certificate to your list of trusted identities.

3. Choose Show Certificate.

4. When the Certificate Viewer appears, choose the Trust tab.

5. Choose an item in the left-hand certificate path field. There may be one or more certificates which make up a certificate chain.

   Tip: If the bottom-most certificate on the chain is selected, then only that certificate will be trusted. If the top-most certificate is selected, then any certificates having that certificate as a root will be trusted. For example, if the root certificate is issued by VeriSign and it is trusted, then other certificates having VeriSign’s certificate as the root (also issued by them) will also be trusted. It is a best practice to trust the topmost certificate that you are willing to trust. Revocation checking starts at the bottom of a chain (begins with the end entity), and once it reaches a trusted root revocation checking stops.

6. Choose Add to Trusted Identities.

7. When asked if the certificate should be trusted from within the document, choose OK.

Figure 60  Revalidate signatures warning

8. When the Import Contact Settings dialog appears, configure the its trust levels. For details, see “Certificate Trust Settings” on page 337.

   The Policy Restrictions tab will not appear if there are no policies associated with this certificate.

9. Choose OK.

10. Choose OK.

11. Choose Validate Signature on the Date/Time tab of the Signature Properties dialog. The icon should change to a clock if the check is successful.
12.3.6.1 When Timestamps Can’t be Verified...

If a signature is timestamped but cannot be verified, open the Trusted Identity Manager and verify:

- A certificate is associated with the timestamp server. Verify the timestamp authority’s certificate is in the certificate list.
- The trust level of the certificate is set. Choose a certificate and verify that the trust level is set for signing. The certificate must either be a trust anchor or be issued by someone whose certificate you have specified as a trust anchor.

12.4 Status Icons and Their Meaning

By default, signatures are validated automatically when a document opens. You can change this behavior as described in “Validating Signatures Automatically” on page 234. Signature and document status’ are represented by status icons and text both in the document, on the Signatures pane, on the Document Message Bar, and in the Signature Properties dialog, and elsewhere.

**Note:** For a higher level of assurance, do not rely solely upon the visual inspection of status icons. Review the Signature Properties dialog for revocation and trust information as well as the signer’s certificate details.

12.4.1 Signature Status Definitions

To determine a signature’s status, the application checks the signature’s digital ID certificate status (is it valid) and document integrity (has it changed since being signed).

The rules for determining signature status are as follows:

- **Valid** signatures used a valid and trusted certificate and the document has not changed or has changed in ways specifically permitted by the author.
- **Problematic** signatures are associated with certificates that cannot be validated or lack a trust relationship with the signer.
- **Unknown** signatures indicate that the signature validity state has not been checked.
- **Invalid** signatures either have an invalid certificate or the document has changed in ways specifically prohibited by the author.

12.4.2 Document Status Definitions

**Tip:** For a one page key, see “Signature Status and Troubleshooting Guide” on page 317.

In addition to the individual status for each signature, the Document Message Bar displays the document’s overall status. The document status is essentially a summation or “rollup” of all the signature status’ AND the effect of document changes after the last signature.

For example, a form might have two valid signatures and a valid document status. However, when some someone types into the form’s text box, both the signature status’ get the blue “i” information icon indicating that something has changed after the signatures were added. The document message bar
now shows a yellow triangle indicating that there are unsigned changes in the document. If the form is signed again, the overall document status changes back to valid as indicated by the green check.

The rules for determining a document’s status are as follows:

- If there is only one signature and the document hasn’t changed since it was signed, then the document status is identical to the signature status.
- The status is flagged as problematic if there are unsigned changes following the last approval signature.
- The status is unknown (magnifying glass) if the authenticity verification check could not complete.
- Like a signature status, if either the authenticity verification or document integrity check fails, the overall document status is invalid (red x).

12.5 Troubleshooting a Signature or Document Status

**Note:** In enterprise settings, a system administrator may have configured your application to behave differently than described below.

Ideally, signature validation should result in the display of a green check icon for approval signatures or a blue ribbon icon for certification signatures. Icons always appear in the Document Message Bar, the Signature Pane, and in the Signature Properties dialog.

In addition to the signature and document status icons and text, key tools for troubleshooting signatures include the following:

- **Signatures pane**: Displays all of the signatures, status, change history, and links to signed versions.
- **Signature Properties dialog**: The dialog provides five tabs that display signature information and buttons for performing document validation tasks. It also provides a Show Certificate button for invoking the Certificate Viewer.
- **Certificate Viewer**: The viewer provides certificate-specific information and buttons for performing certificate validation tasks. Together, the Signature Properties dialog and Certificate Viewer should provide you with enough information to either successfully validate a signature or reject the document as insecure.

12.5.1 Troubleshooting an Identity Problem

If the signature status or overall document status indicates that there is a problem with verifying the authenticity of the signer, you may need to verify the signer and/or decide whether to trust that signer.

**Note:** Trust does not happen automatically. For a signature to be trusted, your application must be configured for that trust. That configuration could be the result of actions by Adobe, your administrator, or you.

To troubleshoot authenticity problems, open the signature panel and expand the information for the problematic signature. Read what it says the problem is, and then take one or more of the following actions:
1. If the status is unknown (displays a magnifying glass) and the icon shows a magnifying glass, it is possible signature validation did not occur.
   - Validate the signature(s) as described in “Validating Signatures Manually” on page 238.
   - Verify you have an internet connection and the application is configured properly.
   - Since the problem may not be with your application, try again later.

2. If the status is problematic (displays a warning triangle), do the following:
   - Verify the signer is in your trusted identity list and that you have configured their certificate or one of their certificate’s issuing certificates as a trust anchor.
   - Verify you have trusted the signer’s certificate for signing and (if necessary) certifying. The Certificate Viewer’s Trust tab indicates the certificates trust level (right click on a Signature, choose Show Signature Properties and then Show Certificate). Specify the certificate’s trust settings as described in “Certificate Trust Settings” on page 337.
   - Verify that a revocation check occurred. Open the Certificate Viewer’s Revocation tab (right click on a Signature, choose Show Signature Properties and then Show Certificate). Check the following:
     - If revocation checking occurred, Problems encountered is active and you can select the button to view a description of the problems.
     - If revocation checking did not occur at all, Check revocation is active and you can select the button to check revocation manually.
     - If online revocation checking is required, it may have failed as a result of no online access or an application problem.

3. If the status is invalid (displays a red X), the signer’s certificate is invalid. Do the following:
   - Contact the signer. The signer may need to get a new digital ID and re-sign a new document.
   - Policy restrictions on a trust anchor can result in signature invalidity. If you have set a policy restriction, determine if that is the problem remove the restriction.

4. If you still cannot pinpoint the problem, or you need help with some of the steps above, read the following:
   - “Troubleshooting Digital ID Certificates” on page 248
   - “Displaying the Signer’s Certificate” on page 249
   - “Verifying the Identity of Self-Signed Certificates” on page 250
   - “Checking Certificate Revocation Status” on page 251
   - “Exporting a Certificate Other than Yours to a File” on page 252

12.5.1.1 Troubleshooting Digital ID Certificates

Someone becomes your trusted identity when you import their valid digital ID certificate and set a specific trust level for that certificate. You can set trust levels ahead of time if you have access to those certificates. If you do not have access to those certificates, simply validate and trust certificates “on-the-fly” as you receive individual documents. As shown in Table 8, the Certificate Viewer provides six tabs with functionality for working with and verifying digital ID certificates.
### 12.5.1.2 Displaying the Signer’s Certificate

When a certificate is displayed in the certificate viewer, you can check certificate validity, trust settings, associated policies, and other details that help you establish the owner’s identity. The Certificate Viewer provides six tabs that displays certificate data and allows you to manage that certificate (Table 8).

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Manage Trusted Identities**.
2. Choose **Certificates** in the **Display** drop-down list.

   **Note:** In addition to this method, you can also display the certificate from any signature or certificate security method workflow where a **Show Certificate** or **Certificate Details** button appears.

3. Select the certificate.

4. Choose **Show Certificate**. The Certificate Viewer displays the certificate. (Figure 61). The following details are available:

   - **Left hand panel**: The certificate chain.
   - **Bottom area**: A description of the certificate, path validity statement, path validation time, and sometimes the type of validation.
   - **Summary tab**: Owner, issuer, validity period, intended usage. An **Export** button allow users to export the certificate to a file.
   - **Details tab**: Lists all the certificate fields (extensions) and their values.
   - **Revocation tab**: Indicates whether a revocation check occurred and the result. Allows users to initiate a manual check and analyze problems.

<table>
<thead>
<tr>
<th>Tab</th>
<th>What it shows</th>
<th>What you can do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Signer and Issuer information, validity dates, and intended usage.</td>
<td>Export the certificate to a file.</td>
</tr>
<tr>
<td>Details</td>
<td>Certificate data such as subject, issuer, used algorithms, public key, and so on.</td>
<td>The data can be used in a variety of ways such as using the digests to verify the certificate’s origin.</td>
</tr>
<tr>
<td>Revocation</td>
<td>Shows certificate validity status of a revocation check and provides an explanation.</td>
<td><strong>Signer Details</strong>: Open the certificate in the Certificate Viewer. The button is only active if the revocation check was successfully completed. <strong>Problems encountered</strong>: View revocation checking problems. The button is only active if revocation checking occurred but failed. <strong>Check revocation</strong>: Enables manual revocation checking. The button is only active if no checking occurred AND a check is possible.</td>
</tr>
<tr>
<td>Trust</td>
<td>Lists the user-specified certificate trust settings.</td>
<td>Add the certificate to the Trusted Identity list.</td>
</tr>
<tr>
<td>Policies</td>
<td>Lists policy OIDs associated with this certificate, if any. Describes the policy.</td>
<td>View policy details.</td>
</tr>
<tr>
<td>Legal Notice</td>
<td>Displays a generic legal disclaimer, the certificate issuer’s policy statement, issuer notice, and link to the policy, if any.</td>
<td>If an issuer policy is used, the policy can be displayed.</td>
</tr>
</tbody>
</table>
12.5.1.3 Verifying the Identity of Self-Signed Certificates

Certificates are usually issued by a trusted, third-party certificate authority such as VeriSign. However, anyone can set up a certificate authority or create a self-signed certificate purporting to be anyone else. For self-signed certificates or those issued by unknown or untrusted certificate authorities, it is prudent to verify the certificate owner’s identity before being added to their trusted identity list.

To verify the origin of the certificate:

1. Display the certificate in the Certificate Viewer:
   - If the certificate is embedded in a signature, right click on the signature, choose Show Signature Properties, display the Summary tab, and choose Show Certificate.
   - If the certificate is in an FDF file, double-click the attached file, and choose Certificate Details in the Import Contact Settings dialog.

   **Tip:** Double clicking a file other than an FDF will likely install the certificate in the Windows Certificate Store. If the file is .cer, .p7b, or some other format and you want to import
into the Acrobat certificate store, save the file locally and import it into the Trusted Identity Manager as described in “Sharing Settings & Certificates with FDF” on page 39.

2. Display the Details tab.

Figure 62 Certificates: Verifying originator

3. In the Certificate data panel, scroll to MD5-digest and SHA-1 digest, and note the numbers.

4. Contact the certificate's originator, and verify the MD5-digest and SHA-1 numbers are correct.

5. After the certificate is verified, display the Trust tab and add the certificate to the trusted identities list.

6. Specify certificate trust settings so that it can be used as a trusted root, to certify documents, and so on. For details, see “Certificate Trust Settings” on page 337.

12.5.1.4 Checking Certificate Revocation Status

Only the certificate issuer (a certificate authority) has the right to revoke a certificate. A certificate could be revoked because its security might be compromised, it could be invalid for some reason, or the owner of the ID might have left the company. Adobe applications check revocation status as part of its public key authentication.

To check a certificate’s revocation status:

1. Choose the Certificate Viewer’s Revocation tab.

2. Choose Check Revocation.

   The button is only active if revocation checking has not occurred and it is possible to do a check.

   Revocation checking is not possible for trusted roots and self-signed certificates.

3. Choose OK.
12.5.1.5 Exporting a Certificate Other than Yours to a File

Users in enterprise settings can send problem certificates to their system administrator or help personnel for troubleshooting. Certificates may be exported from the Trusted Identity Manager or from the Certificate Viewer.

To do export a certificate from the Trusted Identity Manager:

1. Choose Advanced (Acrobat) or Document (Reader) > Manage Trusted Identities.
2. Choose Certificates in the Display drop-down list.
3. Select the certificate.
4. Choose Show Certificate. The Certificate Viewer displays the certificate. (Figure 61).
5. Choose Export.
6. Email the certificate or save it to a file as described in “Emailing Your Certificate” on page 46.

12.5.2 Troubleshooting a Document Integrity Problem

If the signature status or overall document status indicates that there could be a document integrity problem, you need to figure out if and how the document changed during the workflow.

To check document integrity:

1. If the status is unknown and the icon shows a magnifying glass, it is possible signature validation did not occur.
   • Validate the signature(s) as described in “Validating Signatures Manually” on page 238.
   • Verify you have an internet connection and the application is configured properly.
   • Since the problem may not be with your application, try again later.

2. If the status is problematic (displays a warning triangle) shows a, the document has changed but those changes are legal because they have been allowed by the document author. To determine what changed, do any of the following:
   • Open the Signature pane and expand the signature tree to view the change history,
Validating Signatures

Troubleshooting a Document Integrity Problem

- Right click on a signature and choose **View Signed Version** or choose **Click to view this version** in the Signature pane to view the version that signed. Review the status for this version. For details, see Chapter 13, “Document Integrity and Preview Mode”.
- Open the **Form Fields Filled In, or Annotations Created or Modified** item to see which fields or annotations were changed or added.
- Review the document changes as described in “Viewing a List of Post-Signing Modifications” on page 254.
- Perform a page by page and line by line comparison of the problem version with any of the signed versions. For details, see “Comparing a Signed Version to the Current Version” on page 255.

3. Based on the discovered document changes, determine whether you should trust the document.

4. If the status is invalid (displays a red X), illegal changes have been made to the document, there is no way to undo those changes without further changing the document illegally. Do the following:
   - Contact the sender to resolve the problem. Secure the workflow so that illegal changes are prevented.
   - Policy restrictions on a trust anchor can result in signature invalidity. If you have set a policy restriction, determine if that is the problem remove the restriction.

5. If you still cannot pinpoint the problem, or you need help with some of the steps above, read the following:
   - “Troubleshooting Digital ID Certificates” on page 248

12.5.2.1 LiveCycle Dynamic Forms and the Warning Triangle

Document Integrity Checks for 9.0

Same as 8.1 except that changes to document behavior are detected and invalidate an approval signature: prior versions displayed a yellow triangle upon discovery of changes to document behavior.

Document Integrity Checks for 8.1

Acrobat 8.1 does not consider all scripts executed during document construction to potentially modify the document, and the detection of a script does not cause Acrobat to flag the document as changed. The application compares the digitally signed and current document versions to determine if the current version has been modified. If there is a change, then a warning triangle appears on the approval signature. For example, the following changes are detected:

- Changes to the value of an LiveCycle form field (including clearing it).
- Changes to the properties of an LiveCycle form field.

**Document Integrity Checks for 8.0**

Acrobat 8.0 considers all scripts executed during document construction to potentially modify the document even if it’s designed with a read-only query or some other “no change” action. The presence and detection of those scripts would trigger Acrobat to display the yellow warning triangle, thereby indicating that the document may have changed. Scripts that invalidate certification would be prevented from executing so it would not be possible for such scripts to invalidate certifying signatures.
12.5.2.2 Viewing and Comparing Changes and Versions

Document authors and recipients often need to know if a document has changed since it was signed. Acrobat keeps track of a document’s version number, stores previous document versions in their entirety, and enables users to compare document versions by work and page. When you open a document, the latest version is always displayed whether or not it is the signed version. Document recipients should always remember the following:

- Every time a document is signed, the entire document at the point of signing is stored in the PDF as an incrementally numbered version.
- A signature signs a document at a specific point in time; that is, signature X signs version X and signature Y signs version Y. You can view exactly what the document looked like at that point in time using View Signed Version.

For details, see the following:
- Chapter 13, “Document Integrity and Preview Mode”.
- “Viewing a List of Post-Signing Modifications” on page 254.
- “Comparing a Signed Version to the Current Version” on page 255.

12.5.2.3 Viewing a List of Post-Signing Modifications

Because it is possible to change a document without changing its appearance, the list of post-signing modifications is often a superset of what is visually displayed when comparing documents using Document Compare. Therefore, a thorough analysis of a signed document’s integrity includes viewing the document modifications list.

To view a list of post-signing modifications, open the Signature pane and view the change history. All changes are listed in chronological order.

Note that you can also do the following to get a more condensed list:

1. Right click on a signature and choose Show Signature Properties.
2. Choose the Document tab.
3. Choose Compute Modifications List. The list is capable of showing the following:
   - Fields (with field names) that have been created, modified, deleted, or filled in.
   - Annotations (comments) that have been created, modified or deleted.
   - Pages that were created, modified or deleted.
   - Spawned pages, deleted spawned pages, and modified spawned pages.
   - Attachments that have been added, deleted, or modified.
   - Miscellaneous: Some changes which occur in memory or cannot be explicitly listed are labelled miscellaneous.
12.5.2.4 Comparing a Signed Version to the Current Version

**Note:** The Compare feature is not available in Adobe Reader.

As you revise a document and save it to a different name or location, you can verify that you have the latest version by comparing it against an older version. If you're revising a document using comments you received during a review, you may need to view a previous version to make sure that you included all the revisions. As a reviewer, you may want to check the updated document against an older version to make sure that the author has incorporated all of your requested changes.

Document Compare does not compare comments or other annotations in the document.

To automatically compare a signed document version with the current view:

1. Right click on a signature and choose **Compare Signed Version to Current Version**.
2. When the two versions appear side by side, review the highlighted areas to review what was changed.

This method compares the two versions page by page. Compare completes by opening a temporary document that summarizes the differences. The first two pages summarize the changed, added, deleted, or moved pages, taking document A as the original and document B as the modified version (Figure 65).

The differences are displayed as follows:

- Even numbered pages (the pages on the left on the two page document view) are pages from document A.
- Odd numbered pages (the pages on the right on the two page document view) are pages from document B.
- Pages that were moved are not shown in the report.
- Any added page, which only exists in document B, is paired with a blank page in the report. Naturally, the added page will be on the right, and the blank page will be on the left in the two-page view.
- Any deleted page, which only exists in document B, is paired with a blank page in the report. Naturally, the deleted page will be on the left, and the blank page will be on the right in the two-page view.
Validating Signatures

12.6 Document Behavior After Signing

A document’s behavior will likely change after it has been signed. Some of it’s content may not work (multimedia may not play), some of the application’s menu items may be disabled so that you can’t use them, and so on. How a document behaves on your desktop could be the result of one or more factors:

- How the document was authored. Were restrictions or requirements placed on the signature fields?
- How a document was signed. Was an approval or certification signature used? Did the signer place restrictions placed on future permissible actions?

Figure 65 Compare: By page summary report

Figure 66 Compare: By page
Validating Signatures

12.6.1 JavaScript and Dynamic Content Won’t Run

High privilege JavaScript and dynamic content in documents will only run if you have explicitly trusted the sender’s digital ID certificate for such actions. Because scripts and dynamic content represent a security risk, Acrobat prevents some of those operations by default. For details, see “Certificate Trust Settings” on page 337.

12.6.2 Certifying a Document is Prevented

Only one certification signature is allowed in a document; therefore, it must be the first one.

12.6.3 Form Field Fill in, Signing, and/or Other Actions Don’t Work

Once a document is signed or certified, at most only form fill-in, additional signatures, and addition of annotations is allowed. All other operations on the document are disabled as they would invalidate the signature. A certified document may impose further restrictions.

Locking the document after signing also reduces the allowed actions on a document to almost none.
In general, documents that contain no dynamic content (and only recognizable PDF content) are safer to sign that documents with content that can impair one’s ability to see what they are signing and validation. At a high (and simplistic) level, static documents are good; dynamic documents are risky.

Since 8.1, Acrobat has defined PDF features that should be avoided for signable documents that should have a deterministic and repeatable visual rendering. Acrobat’s preview mode feature is designed to display that rendering to users during signing and signature validation.

Preview mode analyzes a document for signing best practices and generates a report and messages which indicate the presence of content that might violate those practices. Preview mode suppresses content that may alter the document’s appearance, thereby allowing you to view, sign, and validate the document in a static and secure state.

To mitigate the risk associated with signatures in complex documents, preview mode makes the document as static as possible and also generates a report about what behaviors could and could not be suppressed. Using this feature involves the following:

- Learning when and how to use preview mode during signing or signature validation.
- If preview mode or warnings are reviewed, analyzing the result to determine if a document should be trusted.

**Tip:** There is only a loose correlation between signature or document status and the information displayed in the PDF Signature report. The report is simply a tool designed to identify the presence of potentially malicious dynamic content that could affect the integrity of your signing workflow. Therefore, signature status and PDF signature report information may appear to be contradictory.

### 13.1 Preview Mode and Signing Workflows

While preview mode may not be needed in your workflow, using it offers a higher degree of assurance that signers and signature validators are viewing the same document and that the signed version is unlikely to change in signing workflows. Users should decide for themselves whether to sign and even trust signatures in documents that contain dynamic content. Therefore, preview mode is recommended for both signing and validation.

Preview mode can be invoked during any part of a workflow:

- **Before or during signing:** When the preference View documents in preview document mode when signing is turned on, preview mode is automatically invoked.
- **After signing:** Right click on any signature and choose View Signed Version.

Application behavior varies slightly depending on how preview mode was invoked. In general, however, PDF signature report information can be viewed on the document message bar (DMB) text or by choosing the View Report button on the DMB.
It may be prudent to not trust a document that generates errors for content that could not be suppressed. For other errors, you may wish to analyze the document for non-conforming content in order to evaluate whether you should trust the document.

13.2 Preview Mode and Validation (View Signed Version)

Acrobat and Adobe Reader store in signed documents a unique document version for every signature in the document. In other words, they “remember” that version A is signed, that changes were made to version B, and so on. When you open a document, the latest version is always displayed. However, it is sometimes useful to view the signed version in order to see what content was actually signed.

To view the signed version of a document.

1. Open the Signatures pane, and verify the signature status.

2. Right click on a signature and choose View Signed Version.
   View Signed Version is essentially a rollback feature that enables the signature validator to view the document version as it was at the point in time when it was signed.

3. Choose View Report in the Document Message Bar to view a report about the dynamic content that could and could not be suppressed (if any).

4. Right click on the signature and choose Show Signature Properties for more detail about the signature.

   Figure 67  Digital Signature Properties: Document Versioning panel

13.3 PDF Signature Reports

Signature workflows often require a document that has a deterministic and repeatable visual rendering that is consistent with the state of the document as signed. This variant does not concern itself with content in a document that is not rendered or does not affect rendering. Examples of such content include metadata and bookmarks.

The level of document’s conformance to signing best practices is determined by the presence or absence of certain content. There are three categories:

- “Content preview mode cannot suppress” on page 260
- “Content preview mode can suppress” on page 260
• “No external dependencies or dynamic content” on page 261

**Content preview mode cannot suppress**

Documents that contain content or behaviors which are dynamic or invisible and which cannot be suppressed in preview mode automatically invoke the PDF Signature Report dialog. For example, preview mode cannot suppress (eliminate from the document or make static) externally referenced images, multimedia content outside of the PDF file, and TrueType fonts. Such content is associated with a red X. Users can highlight an error to see more detail (Figure 68). The signer should decide whether or not to trust the document for signing. If you are not the author, contact the author for additional information.

![Figure 68 PDF Signature Report: Content which cannot be suppressed in preview mode](image)

**Content preview mode can suppress**

Preview mode can suppress certain types of dynamic content. When documents are signed in preview mode, the signed view of the document (with the dynamic content suppressed) is saved so that signature validators can use the View Signed Version feature to validate the signature and see what the signer saw when they were signing.

The **View Report** button opens a dialog that lists discovered rich content. Suppressed content is associated with a green check (Figure 70). If you are concerned about the presence of rich content even though it is suppressed in preview mode, review the error codes and descriptions in the PDF Signature Report dialog for more information.

![Figure 69 Document Message Bar: Suppressible rich content](image)
No external dependencies or dynamic content

For the highest level of document integrity insurance, do not allow dynamic content or any content with external dependencies. These documents are the safest to sign because they can be reliably displayed and do not require a special viewer or preview mode to be signed or to later view what was signed.

13.4 Signature Report Error Codes

As shown in the following tables, report errors categorize content as one of the following:

- **Dynamic features**: Presentations, user-launched multimedia, JavaScript, dynamic forms, and so on.
- **PDF content with variable rendering**: JavaScript, non-embedded fonts, and so on.
- **External content**: Hyperlinks, alternate images, linked files, and so on.
- **Uncategorized content**: Unrecognized or malformed PDF content.

<table>
<thead>
<tr>
<th>String</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document contains hidden behavior</td>
<td>1000</td>
<td>The document contains hidden actions that may not be intended or known by the end user. Actions include JavaScript actions (document open, save, etc.), playing multimedia, executing a menu item, and so on.</td>
</tr>
<tr>
<td>Comment or form field may silently change</td>
<td>1001</td>
<td>The document contains non-signature form fields. Such fields' visual appearances may change based on external variables.</td>
</tr>
</tbody>
</table>
### Table 9  dynamic feature warnings

<table>
<thead>
<tr>
<th>String</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document contains comments. Comments' visual appearances may change based on external variables.</td>
<td>1002</td>
<td>The document contains comments. Comments' visual appearances may change based on external variables.</td>
</tr>
<tr>
<td>Document contains named actions that may launch menu items without the user's knowledge.</td>
<td>1003</td>
<td>The document contains named actions that may launch menu items without the user's knowledge.</td>
</tr>
<tr>
<td>Presentations are not allowed since a presentation may contain animations or other elements that may change document appearance or behavior.</td>
<td>1004</td>
<td>Presentations are not allowed since a presentation may contain animations or other elements that may change document appearance or behavior.</td>
</tr>
<tr>
<td>XFA-based (dynamic forms) documents are not allowed since such forms could alter the document's appearance or behavior.</td>
<td>1005</td>
<td>The document contains a dynamic form</td>
</tr>
<tr>
<td>The document links to external PDFs on the Internet, file system, or network and it has no control over the nature of that linked content. Embedded Go-To actions must not refer to external hierarchies.</td>
<td>1006</td>
<td>Document may not open in the future</td>
</tr>
<tr>
<td>One or more form fields are associated with a 3D object, file attachment, multimedia, or other dynamic objects.</td>
<td>1007</td>
<td>Document contains hidden behavior</td>
</tr>
<tr>
<td>The document contains hidden actions that may not be intended or known by the end user. Actions include JavaScript actions (document open, save, etc.), playing multimedia, executing a menu item, and so on.</td>
<td>1008</td>
<td>Document contains hidden behavior</td>
</tr>
<tr>
<td>The document's content is divided into layers that can be silently displayed or hidden on the fly.</td>
<td>1009</td>
<td>Document contains hidden behavior</td>
</tr>
</tbody>
</table>

### Table 10  PDF Content with variable rendering

<table>
<thead>
<tr>
<th>String</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual elements may change based on external variables. For example, a logo may change color based on time or zoom level. No postscript XObjects allowed.</td>
<td>2004</td>
<td>Page content may silently change</td>
</tr>
<tr>
<td>Some or all of the content is encrypted and the encryption method is not available in standard Acrobat installations. For example, the document may be protected by the Adobe Policy Server. Document contain streams encrypted using crypt filter.</td>
<td>2006</td>
<td>Document may not open in the future</td>
</tr>
<tr>
<td>The document author has enabled image interpolation. No image interpolation is allowed.</td>
<td>2007</td>
<td>Page content may silently change</td>
</tr>
<tr>
<td>The document uses a PDF transfer function that interprets and replaces color. For example, it could replace black with white. Extended graphic state should not use the TR key</td>
<td>2009</td>
<td>Page content may silently change</td>
</tr>
<tr>
<td>The document uses a PDF transfer function that interprets and replaces color. For example, it could replace black with white. If present, the extended graphic state’s TR2 key must be set to default</td>
<td>2010</td>
<td>Page content may silently change</td>
</tr>
<tr>
<td>The document’s extended graphic state uses the FL key. The key is a number that indicates how much flatness tolerance should exist when drawing objects. Content may display differently from Acrobat to other applications</td>
<td>2011</td>
<td>Page content may silently change</td>
</tr>
<tr>
<td>Image XObject must not contain an alternate version</td>
<td>2012</td>
<td>Page content may silently change</td>
</tr>
</tbody>
</table>
Text appearance may silently change

2013 Document contains non-embedded fonts. When the document opens on a system that does not have the requisite fonts, Acrobat will replace them with some other font. Users should always turn on font-related warnings.

The non-embedded fonts warning is turned off by default. It can be turned on by setting the DigSig\bEnNonEmbFontLegPDFWarn preference to true. The disallowed font type warning is also turned off by default and can be turned on by setting the DigSig\bTrueTypeFontPDFSigQWarn preference to true. These are Windows registry or Mac plist settings. See the Security Administration Guide for more details.

Text appearance may silently change

2014 Disallowed font type: <font type>. True type and TrueType-based OpenType fonts are not allowed because they are programs and may change the document’s appearance based on external variables.

Table 11 External Content

<table>
<thead>
<tr>
<th>String</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document links to external content</td>
<td>3000</td>
<td>Document links to images not in the PDF. No external XObjects allowed.</td>
</tr>
<tr>
<td>Document links to external content</td>
<td>3001</td>
<td>Document links to images not in the PDF that are used as alternates. For example, an alternate, high resolution images might be specified for printing. Images must not contain an OPI alternate version.</td>
</tr>
<tr>
<td>Document links to external content</td>
<td>3002</td>
<td>Document contains external streams. The author has flagged some PDF bytes as a stream which may get data from an external source.</td>
</tr>
<tr>
<td>Document links to external content</td>
<td>3003</td>
<td>Document links to images not in the PDF that are used as alternates. For example, an alternate, high resolution images might be specified for printing. Form XObject must not contain an OPI alternate version.</td>
</tr>
</tbody>
</table>

Table 12 Uncategorized warnings

<table>
<thead>
<tr>
<th>String</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unrecognized PDF content</td>
<td>4000</td>
<td>Unrecognized PDF content: The document contains PDF content or custom content not supported by the current version of Acrobat. The document may have been created by a later version of Acrobat.</td>
</tr>
<tr>
<td>Page content may silently change</td>
<td>4001</td>
<td>Unrecognized drawing operator: The document contains PDF content or custom content not supported by the current version of Acrobat. The document may have been created by a later version of Acrobat.</td>
</tr>
<tr>
<td>PDF content contains errors</td>
<td>4002</td>
<td>Malformed drawing instructions: Syntax error. Page content violates the grammar for page content definition. For example, the instruction might specify drawing a square but the syntax for doing it is incorrect.</td>
</tr>
</tbody>
</table>
Document Security (Rights Management)

Security Basics
Password Security
Certificate Security
LiveCycle Policy-Based Security
Security methods provide a mechanism for users to specify document encryption and permission settings. You can encrypt all or part of a document and limit user actions such as only allowing form field fill-in or preventing printing. Each security method offers a different set of benefits, so familiarize yourself with the pros and cons of each type before proceeding (Table 1). Additionally, you can reuse your security settings by saving them as a policy.

To learn about security methods, see the following:

- “Security Method Basics” on page 265: You should understand the options available for specifying the security type, the encryption and permissions options, and whether or not your security settings should be saved as a policy.
- “Changing and Viewing Security Settings” on page 272: Security settings can be changed at any time by the document author.
- “Security Policies: Reusable Security Settings” on page 276: If you would like to reuse your settings, save them as a policy.

For details about security method types, see the following:

- Chapter 15, “Password Security”: Use password security if document recipients do not have digital IDs or it is too cumbersome to collect their certificates.
  
  **Tip:** Password security is unavailable if your application is operating in FIPS mode. Trying to save a document with password security applied results in an alert stating that this security method uses a non-FIPS compliant algorithm.

- Chapter 16, “Certificate Security”: Use certificate security if you can share digital ID certificates with workflow participants, need to configure different permissions for different users, or don’t want to rely on shared passwords.

- Chapter 17, “LiveCycle Rights Management Server Security”: If your company uses an Adobe LiveCycle Rights Management Server, use it to control document access and view audit trails.

### 14.1 Security Method Basics

Security is often added to documents to limit viewing, editing, printing, and other features to only those users that have the required password, a digital ID, or access to an Adobe LiveCycle Rights Management Server. Acrobat’s default security methods not only protect document content from unauthorized access, but also allow users to specify encryption levels and permission settings. At a high level, adding security to a document involves selecting a security type, configuring encryption and permissions, and then saving the document (Figure 1).

**Figure 1 Security method workflow**
Unless saved as a policy, security settings are document-specific and do not apply to other documents. Security can be added to a document through two main methods:

- Create new settings that may or may not be saved as a policy: Choose **Advanced > Security > Show Security Properties**, and then select and configure a method (Figure 2). Both certificate and ALCRMS security allows the user to save the settings as a policy.

**Figure 2  Security method selection**

- Create a new security policy with the Policy Manager: Choose **Advanced > Security > Manage Security Policies**. When the Policy Manager opens, choose **New**, select a policy type, and configure it (Figure 11).

### 14.1.1 Choosing a Security Method Type

While custom security handlers may be installed, the default security methods provide a wide range of robust options. There are a number of reasons to choose one security type over another, but all methods let the user specify encryption algorithms, what document components to encrypt, and what permissions should be granted to whom. Selecting a security type can involve an analysis of each method’s pros and cons (Table 1) or each security method’s basic features:

- **Password security**: Password security provides a simple way to share documents among users where sharing passwords is possible or when high levels of backward compatibility is required. Password policies do not require specifying any document recipients.

  **Tip:** Password security is unavailable if your application is operating in FIPS mode. Trying to save a document with password security applied results in an alert stating that this security method uses a non-FIPS compliant algorithm.

- **Public key certificate security**: Certificate security provides a high level of security, eliminates the need for password sharing, and allows assigning different permissions to different users whose identities can be verified and managed. Supported by Acrobat 6.0 and later.

- **Adobe LiveCycle Rights Management Server security**: These policies are stored on a server, and server access is required to use them. User access information is embedded in the document, so creating an ALCRMS policy involves specifying the document recipients from a list on the LiveCycle Server.
14.1.2 Security Policies

Most workflows allow users to save the settings as a policy, thereby creating a reusable library of preconfigured security methods. When a policy author sets an encryption levels and recipient permissions and then saves them, the policy can later be applied to any document by choosing Advanced > Security > Manage Security Policies and selecting a policy. Policies save time and ensure a consistently secure workflow. For details, see “Security Policies: Reusable Security Settings” on page 276.

14.1.3 Security Methods and Encryption

Encryption is used whenever a security method is added to a document. Security methods encrypt documents or parts of documents and are always involved in granting or denying permissions, thereby protecting content from unauthorized access and actions.

14.1.3.1 Encryption Workflow

The user workflow varies with the security method type as follows:

### Table 1 Security method pros and cons

<table>
<thead>
<tr>
<th>Method</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>Backward-compatible to Acrobat 3.0 for certain encryption levels. Simple and easily understood. Share documents by sharing the password. Supports passwords for document opening. Supports password protecting document permission settings.</td>
<td>Protection depends on password strength. Anyone who knows the password has document access. All users share the same permissions. Won’t work when the application is in FIPS mode.</td>
</tr>
<tr>
<td>Certificate</td>
<td>No password has to be remembered. Key is not susceptible to brute force discovery and resides only on the intended recipients machine. Can encrypt documents for specific people. Can use certificates issued by a trusted 3rd party certificate authority. Allows specifying different permission settings for users. Can leverage LDAP directories for recipients and group lists.</td>
<td>Users must have a digital ID. Organizations need to distribute and manage digital IDs. Compatible with Acrobat 5.0, but full support appears first in 6.0.</td>
</tr>
<tr>
<td>ALCRMS (server-based)</td>
<td>Centralized administration of security policies. Supports document auditing. Allows setting permissions for separate tasks such as opening, editing, and so on. Allows specifying different permission settings for users. Can leverage LDAP directories for recipients and group lists. Controls end-user offline access since authors can specify a validity time limit after which the document expires and is locked.</td>
<td>Requires a network connection. Requires an administrator and some infrastructure such as a LiveCycle Server.</td>
</tr>
</tbody>
</table>
Password security: The user is first asked to select a level of Acrobat backward compatibility. The selection automatically determines the encryption algorithm. Different document components can be encrypted based on the user’s selection. See Table 2.

Note: Password security is unavailable if your administrator has configured your application to operate in FIPS mode.

Certificate security: The user selects what document components to encrypt and then chooses the encryption algorithm.

ALCRMS security: The user selects what document components to encrypt. The algorithm is automatically applied by the server.

14.1.3.2 Choosing What to Encrypt

During the security method workflow, select a radio button in the Select Document Components to Encrypt panel to set the encryption options. You can encrypt all or part of a document based on your need for a specific security level and support for backward compatibility:

- All contents: Encrypts the document and its metadata (Acrobat 3 and later).
- All contents except metadata: Allows for document storage/retrieval systems and search engines to have access to the document metadata. A document open or a permissions password will required to access other document content.

Encrypting everything except the metadata allows continued access to Acrobat’s Catalog feature. By leaving the metadata unencrypted, users can catalog and index the metadata of encrypted documents, thereby making that data searchable (compatible with Acrobat 6 and later).

- Only file attachments: Allows full access to the document and encrypts only the file attachments. Permissions cannot be set on attachments. Using password security, a document open password is required for attachments (compatible Acrobat 7 and later).

Figure 3 Encryption configuration panel

14.1.3.3 Choosing an Algorithm

Acrobat continues to support more sophisticated encryption algorithms (Table 2). When configuring password security, users also have the opportunity to set the Acrobat compatibility level. Choosing to remain compatible with earlier versions of Acrobat results in the use of older algorithms and limits the document components which may be encrypted. For example, choosing a compatibility level of Acrobat 5.0 or earlier does not enable encrypting document contents but not metadata.

When you apply certificate security you will be asked to select between the 128-bit RC4 or 128 or 256 bit AES algorithms. The selection criteria should include the following:

- 256-bit AES is only compatible with Acrobat 9.0 and later. It is the most secure algorithm but results in the largest file size.
- **128-bit AES** is only compatible with Acrobat 7.0 and later. It is mandated for some U.S. government documents because it is more secure than RC4. AES has a bigger file size and adds up to 32 bytes per stream.

- **128-bit RC4** is compatible with Acrobat 6.0 and later as well as other non-Adobe and Adobe PDF clients such as Ghostscript® and Apple Preview® that have not implemented AES. RC4 has a smaller file size by about 32 bytes per stream.

  **Note:** RC4 is unavailable if your administrator has configured your application to operate in FIPS mode.

### Table 2 Encryption algorithm by security type and product version

<table>
<thead>
<tr>
<th>V.</th>
<th>Password Security</th>
<th>Certificate Security</th>
<th>LiveCycle Server</th>
</tr>
</thead>
</table>
| 9.0 | **Algorithms:** 128-bit RC4, 128 and 256 bit AES  
**Options:** All contents, all but metadata, only attachments. | **Algorithms:** 128-bit RC4, 128 and 256 bit AES  
**Options:** All contents, all but metadata, only attachments.  
Dropped support for .apf files. | **Algorithms:** 128 and 256 bit AES  
**Options:** All contents, all but metadata, only attachments. |
| 8.1 | Same as 8.0 except if FIPS mode is turned on, password security is not available. | Same as 8.0 except if FIPS mode is turned on, RC4 is not available. | Same as 8.0 |
| 8.0 | **Algorithms:** 128-bit RC4, 128-bit AES  
**Options:** All contents, all but metadata, only attachments. | **Algorithms:** 128-bit RC4, 128-bit AES  
**Options:** All contents, all but metadata, only attachments. | **Algorithms:** 128-bit AES  
**Options:** All contents, all but metadata, only attachments. |
| 7.0 | **Algorithms:** 128-bit RC4, 128-bit AES  
**Options:** All contents, all but metadata, only attachments. | **Algorithms:** 128-bit RC4, 128-bit AES  
**Options:** All contents, all but metadata, only attachments. | **Algorithms:** 128-bit AES  
**Options:** All contents, all but metadata, only attachments. |
| 6.0 | **Algorithms:** 40 & 128-bit RC4  
**Options:** All contents, all but metadata. | Added support for third-party certificates. | N/A |
| 5.0 | **Algorithms:** 40 & 128-bit RC4  
**Options:** All contents. | Self-sign p7b & apf files only. | N/A |
| 4.0 | **Algorithms:** 40-bit RC4  
**Options:** All contents. | | N/A |
| 4.0.5 | **Algorithms:** 40-bit RC4 (64-bit decrypt)  
**Options:** All contents. | | N/A |
| 3.0 | **Algorithms:** 40-bit RC4  
**Options:** All contents. | | N/A |
| 2.0 | **Algorithms:** 40-bit RC4  
**Options:** All contents. | | N/A |

### 14.1.4 Security Methods and Permissions

Permissions can be set whenever a security method is added to a document. Permission settings enable a document author to limit a document recipient’s activities and interaction with a document. For
example, restrictions can be placed on editing, copying, and printing. You set permissions by choosing
the desired options in the Permissions panel when applying a security method.

![Figure 4 Permissions panel](image)

### 14.1.4.1 Permissions Workflow

The workflow varies slightly with the security method type as follows:

- **Password security**: The permissions panel appears at the beginning of the workflow. Checking
  **Restrict editing and printing of the document** enables all the other fields. Only the password
  security method requires a permissions password. If the document has a permission and a
document open password, it can be opened with either password. The two passwords cannot be
  identical.

- **Certificate security**: The permissions panel appears at the end of the workflow. Permissions can be
  individually specified for different users by highlighting a specific recipient and choosing
  **Permissions**.

- **ALCRMS security**: The permissions are set ahead of time when the method is configured online.
  Permissions can be individually specified for different users by highlighting a document recipient
  and choosing **Permissions**. ALCRMS security provides the option of preventing a document
  recipient from saving and viewing the document offline, thereby storing a copy of the document on
  the local machine. This may not be desirable on public computers or when the computer is not
  secure.

### 14.1.4.2 Permissions Options

All of the security methods provide the following options:

**Note**: Adobe products enforce permissions restrictions. However, not all third-party products
fully support and respect these permissions. Encryption and therefore document
access would likely not be impaired, but Adobe cannot guarantee that individual
permissions settings will remain function. Recipients using such third-party products
might be able to bypass some of your restrictions.

Set the permissions as needed:

1. **Printing Allowed**:
   - **None**: Prohibits printing.
   - **Low Resolution**: Limits printing to 150-dpi resolution. Printing may be slower because
     each page is printed as a bitmapped image. This option is only available if a high
     encryption level (Acrobat 5 or Acrobat 6) is selected. Low resolution also inhibits users from
     printing a high quality document and using optical character recognition software to
     create a similar document with no security.
# Document Security Basics

## Associating Batch Processing with a Security Method

Acrobat can be configured to associate batch processes with a security method. When a batch process is associated with a security method, the method is invoked whenever a batch process is initiated.

To set the batch process security preference:

1. Choose **Edit > Preferences** (Windows) or **Acrobat > Preferences** (Macintosh).
2. Select **Batch Processing** in the left-hand tree.
3. Select a security method from the drop-down list.

   The security handler does not apply security to files. Instead, it determines how batch processing deals with files that are password-protected.

   - **If Don't Ask For Password** is selected, the batch sequence proceeds as if the files are not secure.

---

1. **High Resolution**: Allows printing at any resolution. For example, you can direct high-quality vector output to PostScript and other printers that support advanced high-quality printing features.

2. **Changes Allowed**: Limits page-level editing, commenting, and form field interaction.

   - **None**: Prevents users from changing the document, including filling in signature and form fields.
   - **Inserting, deleting, and rotating pages**: Lets users insert, delete, rotate pages, and create bookmarks and thumbnail pages. This option is only available if a high encryption level is selected.
   - **Filling in form fields and signing existing signature fields**: Lets users fill in forms and add digital signatures. This option doesn't allow users to add comments or create form fields.
   - **Commenting, filling in form fields, and signing existing signature fields**: Lets users fill in forms and add digital signatures and comments.
   - **Any except extracting pages**: Lets users change the document using any method listed in the **Changes Allowed** menu, except remove pages.

3. **Enable copying of text, images, and other content**: Allows file contents (excluding comments) to be copied. It also makes the content available to assistive technology devices such as screen readers. It also lets utilities that need access to the contents of a PDF file, such as Acrobat Catalog, get to those contents. This option is only available if a high encryption level is selected.

4. **Enable text access for screen reader devices for the visually impaired**: Only available if the option above is NOT checked. Lets visually impaired users read the document with screen readers. This option doesn't allow users to copy or extract the document's contents. This option is only available if a high encryption level is selected.
If Password Security is selected, batch processing pauses when it encounters secured files and prompts for a password.

4. Choose OK.

**Figure 5 Security methods for batch processing**

---

14.2 Changing and Viewing Security Settings

While anyone who can open a document can view its security methods, only those with permission can change those methods.

14.2.1 Viewing Document Encryption and Permission Settings

A document’s security settings specify an encryption level (algorithm), what components are encrypted, and permissions. The document may be subject to additional restrictions if it is signed or certified. For more information, see “Viewing Document Restrictions” on page 273.

To view a document’s encryption settings:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security** > **Show Security Properties**.

   **Tip:** You can also choose **File > Document Properties** or press **Control + D** and view the Security tab.

2. Choose **Show Details**. The settings are displayed in a dialog that varies with the security method type. The dialog does not update until a user saves and closes the document.
14.2.2 Viewing Document Restrictions

In addition to the encryption and permissions settings enforced by the document’s security method, a document may be subject to additional restrictions if it is signed or certified. A summary of all security methods and signature-related restrictions appears in the Document Restrictions Summary panel.

When a document has restricted features, any tools and menu items related to those features are disabled. Users who are restricted from using certain document features they think they need should contact the document author.
To view the document restrictions summary in the Document Properties dialog, choose **Advanced** (Acrobat) or **Document** (Reader) > **Security** > Show Security Properties.

**Tip:** You can also choose **Control + D** and view the Security tab.

**Figure 9 Document Property dialog**

---

**14.2.3 Viewing Security Settings in a Browser**

To view document security settings in a Web browser:

1. Click on the lock icon in the left-hand pane.
2. Choose **Permissions Details**.

**Figure 10 Security settings icon**

---

**14.2.4 Changing the Security Method Type**

Security settings cannot be changed on signed documents. Signature fields must first be cleared.
To change a document’s security method:


2. Choose a new security method from the drop-down list.

   **Note:** New settings do not appear in the user interface until the document is closed and reopened.

3. If the document is password protected, enter the document password.

4. Choose a security method and configure it. For details, see the following:
   - Chapter 15, "Password Security"
   - Chapter 16, “Certificate Security”
   - Chapter 17, “LiveCycle Rights Management Server Security”

5. Choose **OK**.

6. Save the document. New settings do not appear in the user interface until the document is closed and reopened.

### 14.2.5 Editing Security Method Settings

To change the security settings for an encrypted document:


2. In the Security panel, choose **Change Settings**. For details, see the following:
   - Chapter 15, "Password Security"
   - Chapter 16, “Certificate Security”
   - Chapter 17, “LiveCycle Rights Management Server Security”

   **Note:** New settings do not appear in the user interface until the document is closed and reopened.

3. Save the document.

### 14.2.6 Removing Document Security

Security can be removed from an open document by those with permissions to do so. You may be required to enter a password or have the requisite certificate to remove a policy.

To remove security settings from a document:

1. From the toolbar, choose **Advanced > Security > Remove Security**.

2. If prompted, type the permissions password.

3. When asked to confirm removal of the security settings, choose **OK**.

4. Choose **OK**.
14.3 Security Policies: Reusable Security Settings

Security policies provide a way to save and reuse security method settings. Saving your configured security method as a policy saves time and effort later. Policies are not embedded in a document. When a document is sent to someone, it contains the specified security settings, but the policy stays with the policy author.

**Note:** Policies can be applied to documents created with any version of Acrobat. However, specific policy settings may not be supported for documents created with some earlier versions.

**Figure 11 Security method selection from Policy Manager**

Policy settings include two main kinds of information:
- Encryption type, permission settings, and passwords (if any).
- Information about the individuals or groups that can open a document or change its security settings.

There are two categories of security policy sources. These policies can be displayed together or individually (Figure 11):
- **User policies:** User policies are created and applied by anyone. User password and certificate policies are stored locally while Adobe LiveCycle Rights Management Server policies are stored on the server. Policy authors can edit and delete the policies they create.
- **Organizational policies:** An organizational policy is created by an Adobe LiveCycle Rights Management Server administrator and is stored on a policy server. The server controls access to documents and auditing events as defined by the security policy. Only a policy administrator can edit, and delete organizational policies.
14.3.1 Creating Security Policies with Policy Manager

Policies can be created ahead of time or during the course of creating new security settings. When the Security Settings Console appears, simply choose **Save these settings as a policy** and enter a policy name and optional description (Figure 12).

**Figure 12 Security policy: General settings**

To create a security policy ahead of time with Policy Manager:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose **New**.
3. Select a security method for the policy and configure in the appropriate section:
   - Chapter 15, “Password Security”
   - Chapter 16, “Certificate Security”
   - Chapter 17, “LiveCycle Rights Management Server Security”

**Figure 13 Policy security method selection**

14.3.2 Applying a Security Policy to a Document

Organization and user policies can be applied to any document by those who have permission to do so.
### 14.3.3 Viewing a Security Policy

To view a security policy:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose a security policy.
3. Choose **View**.
   
   The policy opens in read-only mode and cannot be edited.

### 14.3.4 Copying a Security Policy

Copying a policy is useful when a new policy is needed that is similar to an existing policy. The first policy is simply copied, edited, and saved under a new name.

To copy a security policy:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose a security policy.
3. Choose **Copy**.
4. Change the policy’s settings as described in one of the following sections:
   - Chapter 15, “Password Security”
   - Chapter 16, “Certificate Security”
   - Chapter 17, “LiveCycle Rights Management Server Security”

### 14.3.5 Editing a Security Policy

Existing policies can be edited. For example, if a document is distributed to a group of users and the owners want to revoke permission for others to open it, the owner can change the policy.
To edit a security policy:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose a security policy.
3. Choose **Edit**.
4. Change the policy’s settings as described in one of the following sections:
   - Chapter 15, “Password Security”
   - Chapter 16, “Certificate Security”
   - Chapter 17, “LiveCycle Rights Management Server Security”

### 14.3.6 Making a Security Policy Favorite

When a policy is selected as a favorite, a star appears next to that policy and the policy is then listed on the security menu. Any new policy you create is automatically made a favorite.

To make a security policy favorite:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose a security policy.
3. Choose **Favorite**.
4. Choose **Close**.
   
   A star appears next to the selected policy. “Favorited” policies appear in the security menu (Figure 14).

![Figure 14 Security policy: Favorites list](image)

### 14.3.7 Refreshing the Security Policy List

If the policies are available via a server, refresh the security policy list to ensure that you have access to the most up-to-date server policies.

2. Choose **Refresh**.
3. When the login screen appears, enter a username and password.

4. Choose OK.

### 14.3.8 Deleting a Security Policy

A user can delete any policy that they created. It is not possible to delete organizational policies created by an administrator.

To delete a security policy:

1. Choose **Advanced > Security > Manage Security Policies** (Figure 11).
2. Choose a security policy.
3. Choose **Delete**.
4. Choose **Yes** at the confirmation dialog.
5. Choose **Close**.

### 14.4 Envelopes

You can add security to one or more documents by embedding them in an encrypted envelope, called a security envelope. Envelopes are simply PDF files with attachments. This method is especially useful if you want to send a secure file attachment without modifying or encrypting the attached files. When someone opens the envelope, they can extract the attachments and save them to disk. The saved files are identical to the original file attachments and are no longer encrypted when saved.

For example, suppose that you want to send several documents, including non-PDF documents, to your accountant, but you don’t want anyone else to view the documents. You can embed these documents as file attachments in a security envelope, encrypt the security envelope so that only your accountant can open the attachments, and then email it. Anyone can open the envelope, view its cover page, and even view a list of the contents of that envelope, but only your accountant can view the embedded attachments and extract them to the computer.
Embed file attachments in security envelopes for secure transit.

1. Choose the **Advanced > Security > Create Security Envelope**.

2. Choose **Add File To Send**.

3. Browse to the documents you want to attach and choose **Open**.
   - Select any PDFs in the list that you don’t want to include and choose **Remove Selected Files**.

4. Choose **Next**.

5. Select an envelope template.

6. Choose **Next**.

7. Select whether to deliver the envelope now or later. In most cases, you will want to choose **Send the envelope later** so you can view it and fill out its form fields before sending.
   - **Note:** Templates sometimes contain form fields (such as **To** and **From**) that you can fill in before sending. If you choose to send now and a dialog asks if you really want to send before filling in these fields, choose **Yes** or **No** to continue.

8. Choose **Next**.

9. Choose **Next** OR apply a security policy. Security policies are optional.
   - Select **Show All Policies**, and then select a security policy from the list of available policies (or create a new policy if needed).
Tip: Follow the on-screen instructions to complete the security envelope. If prompted, provide your identity information.

10. Choose **Finish**.

11. Type an email address in the message that appears and choose **Send**, or save the security envelope to send later.
Password Security

Acrobat users can perform any task in this section. Adobe Reader users can only view encrypted documents and cannot encrypt them for others.

Password security provides a simple method for sharing encrypted documents by sharing passwords. Like all security methods, password security can enforce document restrictions on operations such as opening, printing, and editing. Since password security does not provide the ability to specify different permissions for different users, everyone that can open the document will have the same permissions.

Document protection has a dependency on password strength. Acrobat 9.0 now allows full Unicode pass phrases up to 128 characters in length (an actual limit of 128 UTF-8 bytes). Acrobat 8.x and earlier limits passwords to 32 characters maximum and almost entirely to the Latin alphabet (strictly, PDFDocEncoding). Password security encryption levels may also be set to be backward-compatible to Acrobat 3.0.

**Note:** Password security is unavailable if your administrator has configured your application to operate in FIPS mode.

**Figure 16 Password security workflow**

Password security also provides separate options for opening the document and setting user permissions; therefore, password security uses two kinds of passwords: a Document Open password and a Permissions password.

- **Document open password:** Required to open a password-protected document.
- **Permissions password:** Required to change permissions such as those for copying and editing.
15.1 Creating Password Security Settings

Configure and add password security to a document by either creating a policy which can be saved and reused or by creating them once and discarding them. For details, see:

- “Creating a Reusable Password Security Policy” on page 284
- “Creating Password Security for One-Time Use” on page 286

15.1.1 Creating a Reusable Password Security Policy

2. Choose **New**.
3. Choose **Use passwords**.
4. Choose **Next**.
5. Enter a policy name and optional description.
6. Check or uncheck **Save passwords with the policy**.

**Tip:** You can save the password with the policy so that it’s automatically used, or you can have Acrobat prompt you for the policy each time you apply it.

**Figure 17 Security policy: General settings**

7. Choose **Next**.
8. Configure the security settings dialog:
1. **Compatibility**: The compatibility options determine what encryption options will be available. Compatibility with earlier versions of Acrobat may mean all document contents will have to be encrypted. Set the Acrobat compatibility level as follows:

- **Acrobat 3.0 and later**: Encryption uses the 40-bit RC4 encryption algorithm. This setting forces the encryption of strings and streams only and limits other features.
- **Acrobat 5.0 and later**: Encryption uses the 128-bit RC4 encryption algorithm. This setting allows the accessibility option to be selected independently of the copy option, restricts printing to 150-bit dpi, and expands the set of **Changes Allowed** options.
- **Acrobat 6.0 and later**: Encryption uses the 128-bit RC4 algorithm. This setting allows the option of leaving the document metadata unencrypted while the remainder of the document is encrypted. All of the options for Acrobat 5.0 and later are also available.
- **Acrobat 7.0 and later**: Encryption uses the 128-bit AES algorithm. When selected, the option of only encrypting the file attachments is available as well as all of the previous options.
- **Acrobat 9.0 and later**: Encryption uses the 256-bit AES algorithm. Password length can be up to 64 characters.

2. Configure the **Select Document Components to Encrypt** panel as described in “Choosing What to Encrypt” on page 268.

9. If you would like to control who can open the document, provide a Document Open password. You must provide a document open, a permissions password, or both. If you only need to create a permissions password, skip to Step 8.

- Check **Require a password to open the document**.
- Enter a password.

10. If you would like to use password-based permissions, check **Use permissions password to restrict editing of security settings**. Otherwise, skip to Step 12.

   Document authors can set a permissions password that allows users to change the document’s permissions. Only a holder of the permissions password will be able to change the permissions. The Permission password can also be used to open the document even if there is a separate Document Open password.

   **Tip**: Adobe recommends that permission passwords and document open password always be used together. The permissions password is used to change permissions and is NOT needed to gain access to the features the author is permitting. Thus, holders of the permissions password are essentially “owners” of the document and can do anything to it that the author could do.

   **Caution**: Adobe products enforce permissions restrictions. However, not all third-party products fully support and respect these permissions (the encryption would not be violated). Recipients using such third-party products might be able to bypass some of your restrictions.

   Set the permissions as needed:

   1. **Printing Allowed**:

      - **None**: Prohibits printing.
Password Security

Creating Password Security for One-Time Use

- **Low Resolution**: Limits printing to 150-dpi resolution. Printing may be slower because each page is printed as a bitmapped image. This option is only available if a high encryption level (Acrobat 5 or Acrobat 6) is selected. Low resolution also inhibits users from printing a high quality document and using optical character recognition software to create a similar document with no security.

- **High Resolution**: Allows printing at any resolution. For example, you can direct high-quality vector output to PostScript and other printers that support advanced high-quality printing features.

2. **Changes Allowed**: Limits page-level editing, commenting, and form field interaction.

- **None**: Prevents users from changing the document, including filling in signature and form fields.
- **Inserting, deleting, and rotating pages**: Lets users insert, delete, rotate pages, and create bookmarks and thumbnail pages. This option is only available if a high encryption level is selected.
- **Filling in form fields and signing existing signature fields**: Lets users fill in forms and add digital signatures. This option doesn't allow users to add comments or create form fields.
- **Commenting, filling in form fields, and signing existing signature fields**: Lets users fill in forms and add digital signatures and comments.
- **Any except extracting pages**: Lets users change the document using any method listed in the **Changes Allowed** menu, except remove pages.

3. **Enable copying of text, images, and other content**: Allows file contents (excluding comments) to be copied. It also makes the content available to assistive technology devices such as screen readers. It also lets utilities that need access to the contents of a PDF file, such as Acrobat Catalog, get to those contents. This option is only available if a high encryption level is selected.

4. **Enable text access for screen reader devices for the visually impaired**: Only available if the option above is NOT checked. Lets visually impaired users read the document with screen readers. This option doesn't allow users to copy or extract the document's contents. This option is only available if a high encryption level is selected.

11. Choose **OK**.

12. Reenter the Document Open and/or Permissions passwords (if any) when asked to confirm it and choose **OK**.

13. Choose **Finish**.

**15.1.2 Creating Password Security for One-Time Use**

Use this method if you:

- Need to make the document backward-compatible to Acrobat 3.0.
Password Security

Creating Password Security for One-Time Use

- Do not need to save the settings as a policy.

To apply password security to the current document:

3. Check or uncheck Save passwords with the policy.
4. Choose OK.
5. Set the compatibility level to control what encryption options will be available. Compatibility with earlier versions of Acrobat may require encrypting all document contents. Compatibility levels include:
   - Acrobat 3.0 and later: All document contents are encrypted with the 40-bit RC4 algorithm. This option provides the most limited set of permission setting options.
   - Acrobat 5.0 and later: Encryption uses the 128-bit RC4 encryption algorithm. This setting allows the accessibility option to be selected independently of the copy option, restricts printing to 150-bit dpi, and expands the set of Changes Allowed options.
   - Acrobat 6.0 and later: Encryption uses the 128-bit RC4 algorithm. This setting allows the option of leaving the document metadata unencrypted while the remainder of the document is encrypted. All of the options for Acrobat 5.0 and later are also available.
   - Acrobat 7.0 and later: Encryption uses the 128-bit AES algorithm. When selected, the option of only encrypting the file attachments is available as well as all of the previous options.
   - Acrobat 9.0 and later: Encryption uses the 256-bit AES algorithm. When selected, the option of only encrypting the file attachments is available as well as all of the previous options.
6. Configure the Select Document Components to Encrypt panel as described in “Choosing What to Encrypt” on page 268.
7. If you would like to control who can open the document, provide a Document Open password. You must provide a document open, a permissions password, or both. If you only need to create a permissions password, skip to Step 8.
   1. Check Require a password to open the document.
   2. Enter a password.
8. If you would like to use password-based permissions, check Restrict editing and printing of the document. Otherwise, skip to Step 11.
   Document authors can set a permissions password that allows users to change the document’s permissions. Only a holder of the permissions password will be able to change the permissions. Permission password can also open the document even if there is a separate Document Open password.

   **Tip:** Adobe recommends that permission passwords and document open password always be used together. The permissions password is used to change permissions and is NOT needed to gain access to the features the author is permitting. Thus, holders of the permissions password are essentially “owners” of the document and can do anything to it that the author could do.
9. Set the permissions as described in “Permissions Options” on page 270.

10. Choose **OK**.

11. Reenter the Document Open and/or Permissions passwords (if any) when asked to confirm it and choose **OK**.

12. If an alert appears indicating the changes won’t be applied until the document is saved, choose **OK**.

**Figure 18 Security settings require “save” alert**

13. Save the document. New settings do not appear in the user interface until the document is closed and reopened.

### 15.2 Opening a Password-Protected Document

You must know the Document Open or Permissions password to open the document.

To open a password protected document:

1. Open the document.
2. Enter the password.
3. Choose **OK**.

**Figure 19 Password prompt**

### 15.3 Removing Password Security

To remove password security settings from a document:
1. Open the document and supply the required password.

2. Do one of the following:

3. When asked to confirm that you would like to remove security, choose OK.

4. Save the document to have the change take effect.

15.4 Changing Document Collection Passwords

To change security settings for a collection of documents:


2. Do one of the following:
   - Select an existing sequence such as Password or Set Security to No Changes, and then choose Run Sequence.
   - To change the security options, define a new batch-processing sequence or edit an existing one.

15.5 Password Recovery

Caution: There is no way to recover a lost password from a document. Keep a backup copy that is not password-protected.
Acrobat users can perform any task in this section. Adobe Reader users can only view encrypted documents and not encrypt them for others.

If you share documents that require high security, you may need certificate security. Businesses use certificate security because a public key infrastructure (PKI) enables central management by an administrator. The administrator can set up an LDAP directory server for providing certificate access, create custom certificates for specialized workflows, and so on. Where secure PDFs do not have to be compatible with Acrobat versions prior to 6.0, certificate security has several advantages:

- **Different users can have different permission settings**: Unlike password-based security which applies permissions equally to everyone, certificate security allows authors to specify permissions for individuals. For example, it is possible to give employees the ability to sign documents and fill in form fields while giving only managers the ability to add comments or delete pages. Permissions are useful for distributing documents to users that need varied document access and usage rights.

- **Superior security attributes**: No password has to be remembered or shared as the public and private keys to encrypt and decrypt documents reside only on the machines of those participating in secure workflows. These keys are less susceptible to brute force discovery than passwords.

**Note**: Participants in a certificate security workflow must have a digital ID and cannot use Acrobat versions prior to 6.0.

**Figure 20  Certificate security workflow**

![Certificate security workflow diagram](image)
16.1 Setting up the Certificate Security Environment

If you’re going to use certificate security, consider doing the following:

- Configuring Acrobat to use certificates in the Windows Certificate store as well as those in the Acrobat store (which is on by default).
- Choosing which certificate to use for encryption for those contacts who have provided you with more than one.
- Setting up a group and a reusable security policy to simplify your workflows.

16.1.1 Accessing the Windows Certificate Store

The Windows Certificate Store contains certificates used by Windows applications. For example, when signing outgoing emails in Outlook, the digital ID comes from the Personal certificate store in Windows. The trusted certificates stored in the Windows Trusted People certificate store are used by Windows applications to validate signed emails from other people.

If you want to use certificates in the Windows Certificate Store to encrypt the document for the certificate owner, add the Windows store to the certificate search path. This allows you to search Windows directories when applying certificate security. By default, the Windows Certificate Store is not included in the application search path. Once the option is manually turned on, the Windows store will appear in the Search for recipients dialog Directories drop-down list.

To enable searching for certificates in the Windows Certificate Store:

1. Choose Edit > Preferences.
2. Choose Security in the left-hand panel.
3. Choose Advanced Preferences.
4. Display the Windows Integration tab.
5. Check Enable searching the Windows Certificate Store for certificates other than yours.
6. Choose OK.
7. Choose OK.

**Tip:** The checkboxes related to trust are only used for signature validation.
Figure 21  Windows integration

The Windows Certificate Store will now appear in Search for Recipients dialog's directory list. The dialog can be invoked from two locations:

- From a certificate security workflow: Set the encryption settings, choose Next, and then choose Search.
- From the Trusted Identity Manager: Choose Add Contacts, and then choose Search.

16.1.2 Selecting a Certificate to Use for Encryption

Because you encrypt a document for someone with the public key in their certificate, you must first explicitly choose their certificate for encryption. Each contact in your Trusted Identity list should be associated with at least one certificate. If there is only one certificate, Acrobat automatically selects it as the one to use for encryption. If more than one certificate is associated with the contact, you can select which one to use as the default encryption certificate.

**Note:** To use a certificate for encryption must have encryption usage rights. A warning dialog appears during the encryption process if the selected certificate cannot be used.

To set a default certificate for encryption:

1. Choose Advanced > Manage Trusted Identities.
2. Choose a contact in the left-hand list.
3. Choose Details.
4. Highlight a certificate in the certificate list.
5. Choose Use for encryption (Figure 22). The lock icon moves to the selected certificate.
6. Choose OK.
16.2 Working with Groups of Contacts

Contacts can be added to a group so that all group members can easily share a predefined set of permissions and restrictions. For example, it is possible to create a certificate-based security policy that applies to an entire group. Administrators and home users can create a group and export the group’s details to an FDF file that is then sent to individual users. This feature makes it easy to manage permissions for a large number of people.

Note: Importing a group imports the contacts (all group members), but not the group. If desired, create a new group from those newly imported contacts.

16.2.1 Creating a Group

Individual users and administrators create a group using the same method.

To create a group:

1. Choose Advanced > Manage Trusted Identities.
2. Choose New Group.
3. Enter a group name (Figure 23).
4. Add contacts.
5. Choose OK.

16.2.2 Adding or Removing Group Contacts

To add or remove group members:

1. Choose Advanced > Manage Trusted Identities.
2. Double-click on a group or highlight the group and choose Details.

3. Add or remove a contact:
   - **Adding a contact**: Choose Add, select a contact from the contact list, and choose OK twice.
   - **Removing a contact**: Select a contact, choose Remove, and choose OK.

16.2.3 Deleting a Group

To delete a group:

1. Choose Advanced > Manage Trusted Identities.
2. Choose a group in the left-hand list.
3. Choose Delete.
4. Choose OK.

16.3 Creating Certificate Security Settings

When adding security to a document, you either create a policy which can be reused or creating the once and discard them. For details, see:

- “Creating a Reusable Certificate Security Policy” on page 295
- “Creating Certificate Security for the Current Document” on page 298
To create a certificate security policy:

2. Choose **New**.
3. Select **Use public key certificates**.
4. Choose **Next**.
5. Enter a policy name and optional description.

6. Configure the **Select Document Components to Encrypt** panel as described in “Choosing What to Encrypt” on page 268.

7. Check or uncheck **Ask for recipients when applying this policy**.
   - If checked, you will not be asked in the next step to select the recipient certificates. Because the policy will not be associated with any recipients you will select them when you apply the policy.
   - If unchecked, you will be asked to select certificates now so that the document recipients will be identified in the policy.

8. Choose the encryption algorithm:
   - **128-bit RC4**: Compatible with Acrobat 6.0 and later as well as other non-Adobe and Adobe PDF clients such as Ghostscript and Apple Preview that have not implemented AES. RC4 has a smaller file size by about 32 bytes per stream.
   - **128-bit AES**: Compatible with Acrobat 7.0 and later. It is mandated for some U.S. government documents because it is more secure than RC4. AES adds up to 32 bytes per stream.
   - **256-bit AES**: Compatible with Acrobat 9.0 and later. Provides the highest level of encryption.

   **Note:** The RC4 encryption algorithm is unavailable if your administrator has configured your application to operate in FIPS mode.

9. Choose **Next**. If you checked **Ask for recipients when applying this policy**, choose **Finish**. Otherwise, go to the next step.
10. **The Digital ID Selection dialog may not appear. If it does not appear, go to the “add document recipients to the recipient list” step:** The digital ID selection dialog only appears if you have no digital IDs suitable for encryption or more than one. If you only have one digital ID suitable for encryption, then this dialog does not appear (for example, one ID is set as the default for encryption in the Security Settings Console). If the dialog does appear, select your digital ID that you will use to access this document in the future.

   **Tip:** While it is possible to apply certificate security without selecting your digital ID, doing so leaves you off of the recipient list and permanently locks you out of the document. If the required digital ID does not appear in the list, choose Add Digital ID and follow the steps described in “Registering a Digital ID for Use in Acrobat” on page 349.

11. If you have more than one digital ID, choose the digital ID persistence level.
   - Ask me which digital ID to use next time
   - Use this digital ID until I close the application
   - Always use this digital ID

   **Note:** This option will not appear for users with only one digital ID.

12. Choose OK.

13. Add document recipients to the recipient list. You will be encrypting the document with each recipient’s public key so that they can decrypt it. Choose from the following:
   - **Search** lets you search preconfigured directories for certificates on remote servers as well as in your local Trusted Identity list. For details about searching for certificates, see “Searching for Digital ID Certificates” on page 335. Highlight one or more found digital IDs and choose OK.
   - **Browse** lets you search your computer for certificate files stored locally. Highlight one or more found digital IDs and choose OK.
Tip: Business users may need to search their company LDAP directory server. For details, see “Using Directory Servers to Add Trusted Identities” on page 341.

Figure 27 Adding recipients to a document with certificate security

14. If you want to specify document permissions, do the following. Otherwise, skip to Step 16.

1. Highlight one or more recipients. Different permissions can be set for different recipients. Select multiple recipients from the list by using the Control or Shift keys.

2. Choose Permissions.

3. When an alert appears stating that non-Adobe products may not respect these settings, choose OK.

4. Check Restrict printing and editing of the document and security settings.

Caution: Adobe products enforce permissions restrictions. However, not all third-party products fully support and respect these permissions. Encryption and therefore document access would likely not be impaired, but Adobe cannot guarantee that individual permissions settings will remain function. Recipients using such third-party products might be able to bypass some of your restrictions.

Set the permissions as needed:

1. **Printing Allowed**:
   - **None**: Prohibits printing.
   - **Low Resolution**: Limits printing to 150-dpi resolution. Printing may be slower because each page is printed as a bitmapped image. This option is only available if a high encryption level (Acrobat 5 or Acrobat 6) is selected. Low resolution also inhibits users from printing a high quality document and using optical character recognition software to create a similar document with no security.
   - **High Resolution**: Allows printing at any resolution. For example, you can direct high-quality vector output to PostScript and other printers that support advanced high-quality printing features.
2. **Changes Allowed**: Limits page-level editing, commenting, and form field interaction.

- **None**: Prevents users from changing the document, including filling in signature and form fields.
- **Inserting, deleting, and rotating pages**: Lets users insert, delete, rotate pages, and create bookmarks and thumbnail pages. This option is only available if a high encryption level is selected.
- **Filling in form fields and signing existing signature fields**: Lets users fill in forms and add digital signatures. This option doesn't allow users to add comments or create form fields.
- **Commenting, filling in form fields, and signing existing signature fields**: Lets users fill in forms and add digital signatures and comments.
- **Any except extracting pages**: Lets users change the document using any method listed in the Changes Allowed menu, except remove pages.

3. **Enable copying of text, images, and other content**: Allows file contents (excluding comments) to be copied. It also makes the content available to assistive technology devices such as screen readers. It also lets utilities that need access to the contents of a PDF file, such as Acrobat Catalog, get to those contents. This option is only available if a high encryption level is selected.

4. **Enable text access for screen reader devices for the visually impaired**: Only available if the option above is NOT checked. Lets visually impaired users read the document with screen readers. This option doesn't allow users to copy or extract the document's contents. This option is only available if a high encryption level is selected.

15. Choose **OK**.

16. Choose **Next**.

17. Choose **Finish**.

**16.3.2 Creating Certificate Security for the Current Document**

This workflow allows you to save the settings as a policy or discard them after they are applied. Use this method if you:

- Have access to the document recipient's digital IDs.
- Do not need to save the settings as a policy.

To apply certificate security to the current document:


2. Select **Certificate Security** from the **Security Method** drop-down list.

3. Choose one of the following:

Acrobat Family of Products

- **Save these settings as a policy**: Choosing to save the settings as a policy activates the **Policy name** and **Description** fields. Once the wizard is completed, the settings are saved as a policy and added to the policy list in the Advanced > Security menu and the Policy Manager. If you are creating a policy, enter a policy name and optional description.

- **Discard these settings after applying**: Choosing to discard the settings deactivates the **Policy name** and **Description** fields and no settings are saved.

4. Configure the Select Document Components to Encrypt panel:

   **Note**: Adobe products enforce permissions restrictions. However, not all third-party products fully support and respect these permissions. Encryption and therefore document access would likely not be impaired, but Adobe cannot guarantee that individual permissions settings will remain function. Recipients using such third-party products might be able to bypass some of your restrictions.

Set the permissions as needed:

1. **Printing Allowed**:
   - **None**: Prohibits printing.
   - **Low Resolution**: Limits printing to 150-dpi resolution. Printing may be slower because each page is printed as a bitmapped image. This option is only available if a high encryption level (Acrobat 5 or Acrobat 6) is selected. Low resolution also inhibits users from printing a high quality document and using optical character recognition software to create a similar document with no security.
   - **High Resolution**: Allows printing at any resolution. For example, you can direct high-quality vector output to PostScript and other printers that support advanced high-quality printing features.

2. **Changes Allowed**: Limits page-level editing, commenting, and form field interaction.

   - **None**: Prevents users from changing the document, including filling in signature and form fields.
   - **Inserting, deleting, and rotating pages**: Lets users insert, delete, rotate pages, and create bookmarks and thumbnail pages. This option is only available if a high encryption level is selected.
   - **Filling in form fields and signing existing signature fields**: Lets users fill in forms and add digital signatures. This option doesn't allow users to add comments or create form fields.
   - **Commenting, filling in form fields, and signing existing signature fields**: Lets users fill in forms and add digital signatures and comments.
   - **Any except extracting pages**: Lets users change the document using any method listed in the Changes Allowed menu, except remove pages.

3. **Enable copying of text, images, and other content**: Allows file contents (excluding comments) to be copied. It also makes the content available to assistive technology devices such as screen readers. It also lets utilities that need access to the contents of a PDF file, such as
Certificate Security

Creating Certificate Security for the Current Document

Acrobat Catalog, get to those contents. This option is only available if a high encryption level is selected.

4. **Enable text access for screen reader devices for the visually impaired**: Only available if the option above is NOT checked. Lets visually impaired users read the document with screen readers. This option doesn't allow users to copy or extract the document's contents. This option is only available if a high encryption level is selected.

5. Choose the encryption algorithm:

   - **128-bit RC4**: Compatible with Acrobat 6.0 and later as well as other non-Adobe and Adobe PDF clients such as Ghostscript and Apple Preview that have not implemented AES. RC4 has a smaller file size by about 32 bytes per stream.

   - **128-bit AES**: Compatible with Acrobat 7.0 and later. It is mandated for some U.S. government documents because it is more secure than RC4. AES has a bigger file size and adds up to 32 bytes per stream.

   - **256-bit AES**: Compatible with Acrobat 9.0 and later. Provides the highest level of encryption.

   **Note**: The RC4 encryption algorithm is unavailable if your administrator has configured your application to operate in FIPS mode.

6. Choose **Next**.

7. **The Digital ID Selection dialog may not appear. If it does not appear, go to the “add document recipients to the recipient list” step**: The digital ID selection dialog only appears if you have no digital IDs suitable for encryption or more than one. If you only have one digital ID suitable for encryption, then this dialog does not appear (for example, one ID is set as the default for encryption in the Security Settings Console). If the dialog does appear, select your digital ID that you will use to access this document in the future.

   **Tip**: While it is possible to apply certificate security without selecting your digital ID, doing so leaves you off of the recipient list and permanently locks you out of the document.

If the required digital ID does not appear in the list, choose **Add Digital ID** and follow the steps described in “Registering a Digital ID for Use in Acrobat” on page 349.
8. If you have more than one digital ID, choose the digital ID persistence level.
   - Ask me which digital ID to use next time
   - Use this digital ID until I close the application
   - Always use this digital ID
   
   Note: This option will not appear for users with only one digital ID.

9. Choose OK.

10. Add document recipients to the recipient list. You will be encrypting the document with each recipient’s public key so that they can decrypt it. Choose from the following:
   - Search lets you search preconfigured directories for certificates on remote servers as well as in your local Trusted Identity list. For details about searching for certificates, see “Searching for Digital ID Certificates” on page 335. Highlight one or more found digital IDs and choose OK.
   - Browse lets you search your computer for certificate files stored locally. Highlight one or more found digital IDs and choose OK.

   Tip: Business users may need to search their company LDAP directory server. For details, see “Using Directory Servers to Add Trusted Identities” on page 341.
11. If you want to specify document permissions, do the following. Otherwise, skip to Step 13.

1. Highlight one or more recipients. Different permissions can be set for different recipients. Select multiple recipients from the list by using the Control or Shift keys.

2. Choose Permissions.

3. When an alert appears stating that non-Adobe products may not respect these settings, choose OK.

4. Check Restrict printing and editing of the document and security settings.

   Note: Adobe products enforce permissions restrictions. However, not all third-party products fully support and respect these permissions. Encryption and therefore document access would likely not be impaired, but Adobe cannot guarantee that individual permissions settings will remain function. Recipients using such third-party products might be able to bypass some of your restrictions.

Set the permissions as needed:

1. **Printing Allowed:**
   - **None:** Prohibits printing.
   - **Low Resolution:** Limits printing to 150-dpi resolution. Printing may be slower because each page is printed as a bitmapped image. This option is only available if a high encryption level (Acrobat 5 or Acrobat 6) is selected. Low resolution also inhibits users from printing a high quality document and using optical character recognition software to create a similar document with no security.
   - **High Resolution:** Allows printing at any resolution. For example, you can direct high-quality vector output to PostScript and other printers that support advanced high-quality printing features.
2. **Changes Allowed**: Limits page-level editing, commenting, and form field interaction.

<table>
<thead>
<tr>
<th>Changes Allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>None: Prevents users from changing the document, including filling in signature and form fields.</td>
</tr>
<tr>
<td>Inserting, deleting, and rotating pages: Lets users insert, delete, rotate pages, and create bookmarks and thumbnail pages. This option is only available if a high encryption level is selected.</td>
</tr>
<tr>
<td>Filling in form fields and signing existing signature fields: Lets users fill in forms and add digital signatures. This option doesn’t allow users to add comments or create form fields.</td>
</tr>
<tr>
<td>Commenting, filling in form fields, and signing existing signature fields: Lets users fill in forms and add digital signatures and comments.</td>
</tr>
<tr>
<td>Any except extracting pages: Lets users change the document using any method listed in the <strong>Changes Allowed</strong> menu, except remove pages.</td>
</tr>
</tbody>
</table>

3. **Enable copying of text, images, and other content**: Allows file contents (excluding comments) to be copied. It also makes the content available to assistive technology devices such as screen readers. It also lets utilities that need access to the contents of a PDF file, such as Acrobat Catalog, get to those contents. This option is only available if a high encryption level is selected.

4. **Enable text access for screen reader devices for the visually impaired**: Only available if the option above is NOT checked. Lets visually impaired users read the document with screen readers. This option doesn’t allow users to copy or extract the document’s contents. This option is only available if a high encryption level is selected.

12. Choose **OK**.

13. Choose **Next**.

14. Choose **Finish**.

### 16.3.3 Applying a Certificate Security Policy

If your certificate security settings already exist in a policy, apply those settings with Policy Manager:


2. Select a policy that uses certificate security.

3. Choose **Apply to Document**.

4. Save the document. New or changed settings do not appear in the user interface until the document is closed and reopened.
16.3.4 Applying a Certificate Security to a Group

Certificate security may be applied to more than one individual at a time. To apply security for multiple people, use the Search for Recipients dialog:

1. Configure certificate security as described in “Creating Certificate Security Settings” on page 294. When you are prompted to add document recipients to the recipient list, choose Search.

2. Select the Search Directories:
   - Check Search all directories to search all the directories you have configured in the Security Settings Console.
   - Uncheck Search all directories to search a specific directory. If you are just searching for individuals in your Trusted Identities list:
     1. Choose Trusted Identities from the Directories drop-down list.
     2. Select a group from the Groups drop-down list. All members of this group will appear in the Search Results field.

3. Enter a search name or email address.
4. Highlight one or more of displayed individuals.
5. Choose OK and continue configuring the security settings.

   **Figure 30  Searching for group contacts**

16.3.5 Opening a Certificate-Protected Document

Password protected documents require that a user know the document open password to open it. If a permissions password has been set, that password can also be used to open the document.
To open a password protected document:

1. Open the document.

2. Enter the password associated with the digital ID used to encrypt the document.

3. Choose OK.

Figure 31 Opening an encrypted document: With certificate security
Adobe LiveCycle Rights Management Server (ALCRMS) security is only available to users with access to an Adobe LiveCycle Rights Management Server.

**Tip:** This document provides a cursory overview of the ALCRMS features. For information on configuring your application to use an Adobe LiveCycle Rights Management Server, log in to the server and use the help system.

ALCRMS’s server-based security system provides a Web-based user interface for dynamic document control through the use of policies stored on a server. The policies enable centralized document management and event auditing. The documents that use those policies can reside anywhere. ALCRMS policies not only enable reusing security settings, but they also let the author expire and revoke documents irrespective of how many copies were created or distributed. You can also maintain accountability and audit who opens protected documents.

ALCRMS can be configured to run with LDAP, ADS, and other enterprise systems so that user lists can be leveraged from an organization’s existing information.

Using server-based security policies includes the following steps:

1. **Application configuration:** A system administrator configures the machine or the end user can do it manually. Server settings can also be sent via an .acrobatsecurity or an FDF file thereby enabling the end user to automatically import the requisite settings from a secure file. The administrator manages accounts, sets up organizational policies, and maintains the server.

2. **Policy configuration:** Reviewing the list of preconfigured organizational policies or creating a new user policy.

3. **Apply the policy and publish the document:** You apply a policy to the document with Acrobat’s Policy Manager which can be accessed via the security main menu or through the Document Properties dialog. The policy server generates a license and an encryption key. Acrobat embeds the license in the document and encrypts it using the encryption key. You distribute the document or tell others where to find it.

4. **Viewing a document that has a policy applied:** When users try to open the document, they must authenticate their identities. The document is decrypted and opens with whatever permissions are specified in the policy.

5. **Auditing events and modifying access:** You audit document and usage history by logging in to the ALCRMS. You can modify access rights and user policies.

### 17.1 Configuring Servers

In most cases if you have access to an ALRM server then you’re administrator will set up the server for you. There are three ways to set up an ALRM server:

- "Importing ALCRMS Settings from an FDF file" on page 307
17.1.1 Importing ALCRMS Settings from an FDF file

Adobe LiveCycle Rights Management Server settings can be distributed via FDF files. Both users and administrators can import and export server settings in the same way as timestamp and directory server information is imported and exported. For details, see “Migrating and Sharing Security Settings” on page 35.

If you need to configure the server settings manually, refer to “Configuring ALCRMS Settings Manually” on page 307.

17.1.2 Importing ALCRMS Settings with a Security Settings Import

In addition to using FDF files, an administrator may export the requisite security settings and provide you with a file to import. In this case, you will import any settings in the file according to the administrators instructions.

17.1.3 Configuring ALCRMS Settings Manually

Your server administrators will provide you with server connection details. Once these details are obtained, configure Acrobat to use the server.

To connect to a Adobe LiveCycle Rights Management Server:

2. Select Adobe LiveCycle Rights Management Servers in the left-hand panel.
3. Choose New.
4. Enter the server settings:
   - **Name**: The server name.
   - **Server Name**: The server URL.
   - **Server Port**: The server port.
   - **Username**: The login username if required.
   - **Password**: The login password if required.
5. Choose Connect to this Server.
6. Choose OK.
17.1.4 Managing your ALCRMS Account

To manage your ALCRMS Account:


2. If prompted, enter a username and password and choose **OK**.

3. Manage your account as described in the Adobe LiveCycle Rights Management Help documentation.

17.2 Working with Documents and ALCRMS Policies

17.2.1 Creating an ALCRMS Security Policy

ALCRMS policies are created using the server’s web interface. However, it is possible to launch that interface directly from Acrobat. Once the policy is created, return to Acrobat and choose **Finish** to add the policy to the policy list.

To create an ALCRMS user security policy:


2. Choose **New**.

3. Select **Use Adobe LiveCycle Rights Management**.

4. Log in to the server.

5. Navigate to the Policies page.

6. Enter a policy name and optional description.
7. Configure the **Validity period** panel. A document’s validity period determines how long it will be accessible. When a recipient opens a document with an expired validity period, an alert appears stating that the document is locked (Figure 33).

![Figure 33 Validity period expired alert](image)

8. Choose **Yes** or **No** to **Audit Documents**. Auditing tracks events such as printing, modifying, viewing, closing, form filling, and signing documents.

![Figure 34 Audit alert for ALCRMS security](image)

9. Set the **Auto-Offline lease period** to specify how long the document can be viewed offline before a user must synchronize with Adobe LiveCycle Rights Management Server.

10. Choose **Save**.

11. Exit the Web console and return to Acrobat.

12. Choose **Finish**.

### 17.2.2 Applying ALCRMS Security

Your ALCRMS policies will appear in Acrobat policy list.

To apply an ALCRMS policy:


2. Highlight a policy.

3. Choose **Apply to Document**.

### 17.2.3 Refreshing the Security Policy List

To refresh the list of available ALCRMS policies:

2. If prompted, enter a username and password and choose **OK**.
   An updated list of policies will appear in Policy Manager.

### 17.2.4 Synchronizing a Document for Offline Use

Synchronizing a document for offline use allows you to get the latest version so that you can access it when you are not connected to the network. To synchronize a document:

1. Choose **Advanced > Security > Adobe LiveCycle Rights Management > Synchronize for Offline**.

2. If prompted, enter a username and password and choose **OK**.

### 17.2.5 Revoking a Document

To revoke a document so that it cannot be viewed by anyone:


2. If prompted, enter a username and password and choose **OK**.

3. Enter the revocation details as described in the Adobe LiveCycle Rights Management Help documentation.

4. Choose **OK**.

### 17.2.6 View a Document’s Audit History

To view a document’s audit history:

1. Open the document you would like to track.

2. Choose **Advanced > Security > Adobe LiveCycle Rights Management > View Audit History**.

3. If prompted, enter a username and password and choose **OK**.

4. View the audited events as described in the Adobe LiveCycle Rights Management Help documentation.
Appendices

Quick Keys
What’s Changed Across Releases
Supported Standards
Etc.
One Page Keys and Guides

The following diagrams provide an overview of security related workflows and processes:

- Signature Creation Workflow
- Signing and the Byte Range
- Signatures, Permissions, and the PDF Language
- Revocation Checking Process
- Signature Status and Troubleshooting Guide
- Security workflow for encryption-permissions
A.1 Signature Creation Workflow

**Signature Creation Workflow and Configuration Options**

From *Digital Signatures and Rights Management in Acrobat and Adobe Reader*

The workflow is configurable through the GUI and registry settings.\(^1\)

Supports 3\(^{rd}\) party handlers.

Create signature appearances with the UI or programmatically. Contents are defined by the SigAP dictionary included in the form field.\(^2\)

The signature appearance is included in the message digest.

The DSA/RSA algorithm is specified by the signer’s digital ID private key in a P12/PFX file, smart card/token, roaming ID server, or Windows store.

Revocation checking is configurable for any certificate in the chain, including the signers, the ICA, any timestamp CA, etc.\(^3\)

If not time stamped or time stamping fails, use the local time.\(^3\)

The signature object is built in memory until all the revocation checking, timestamp, and certificate data is retrieved.

The signature object is encoded using the Distinguished Encoding Rules (DER). The DER object is hex encoded and padded with zeros to make the signed byte range match the size that was set aside when the signing process began.

Dictionary contents are customizable through UI and registry settings such as cReasons, cContactInfo, etc.\(^3\)

**References**

1. PDF Reference
2. Digital Signature Build Dictionary Specification
3. Digital Signatures and Rights Mgmt in Acrobat
4. Digital Signature Appearances (in SDK)
A.2 Signing and the Byte Range

A byte range is an array of four numbers (named ByteRange) which is stored alongside the signature value. The four numbers are actually two byte sequences that define what is hashed. The first number in each pair is the offset in the file (from the beginning, starting from 0) of the beginning of a stream of bytes. The second number is the length of that stream.

In many signing workflows, XML forms saved in a PDF format are signed as any other PDF. However, there can be differences. For example, Rendering components are never included in the signed byte range. Data signatures include form data but exclude everything else from the signed byte range.

Data signatures do not use byte range signing.

1: The document is converted to a stream of bytes.
2: The entire file is written to disk with a reserved space populated with temporary, worst-case (largest) values in the ByteRange array.
3: The document is digested with a hash algorithm, and a hash is computed using the bytes specified by the real ByteRange value.
   
   \textit{Except for the signature hole (signature value bytes), Acrobat always computes the hash for the entire PDF file, starting from byte 0 and ending with the last byte in the physical file. In the byte range, the first value must be 0, the last must be the offset of the last byte in the file and the other numbers identify the hole in which the signature will be placed.}

4: The temporary ByteRange array is overwritten using the correct values. Because the byte offsets must not change, extra bytes following the new array statement are overwritten with spaces.
5: The hash value is encrypted with the signer's private key using a supported RSA or DSA signature algorithm. A signature object is generated. By default, it is a PKCS#7 object.
6: The signature object is placed in the file on disk, overwriting the placeholder /Contents value. Any space not used for the signature object is overwritten with spaces.
7: The PDF file is reloaded in Acrobat to ensure that the in-memory version matches the disk version.

\textit{Registry/plist-level signature creation preferences can control the type and strength of signature as well as what conditions will cause signing to fail. The available options allow you to configure signing workflows for varying degrees of security. Almost every phase of the signing workflow and revocation checking is configurable.}
A.3 Signatures, Permissions, and the PDF Language

**Signatures and Permissions**

At the PDF language level, a signature may be hooked up to one or more permission handlers. Permissions may be specified by TransformMethod (FieldMDP, DocMDP, or UR3) which use a TransformParams array to specify signature characteristics and document permissions or by /Lock in the field dictionary. Permissions can be:

* Set by the author via seed values or with field properties that restrict user actions after signing.
* Set by the signer when certifying to allow no changes; form fill-in & signing; or form fill-in, signing, & annotations.
* Set by the signer using an approval signature under certain conditions.
* Document or field-level.
* A grantor of rights such as signing with an approval signature to Adobe Reader users via UR3.
* A cause of invalid signatures if permissions are violated.

<table>
<thead>
<tr>
<th>TransformMethod</th>
<th>Sig Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A (none)</td>
<td>Approval</td>
<td>Any number allowed. Can lock document during signing under certain conditions. Adobe Reader users can only sign when usage rights are enabled via UR3.</td>
</tr>
<tr>
<td>FieldMDP</td>
<td>Approval</td>
<td>Signer can lock document when signing if the field is last unsigned field and it contains no no seed values which prohibit locking or other locking rules.</td>
</tr>
<tr>
<td>FieldMDP</td>
<td>Both</td>
<td>Authors set permissions via the form field’s Digital Signature Properties dialog.</td>
</tr>
<tr>
<td>DocMDP</td>
<td>Certification</td>
<td>Set during certification. First signature only. By default, FieldMDP present. MDP seed value set on field will force use of certification signature.</td>
</tr>
<tr>
<td>UR3</td>
<td>Approval</td>
<td>Acrobat authors grant rights (e.g. signing with approval signatures) to Reader users.</td>
</tr>
</tbody>
</table>

![Signature Diagram](signature-diagram.png)

FieldMDP TransformParams

- /Actions/ (All | Inc. | Exc.)
- /Fields/[Field names]
- /P 1

DocMDP TransformParams

- Allow sign, annots, etc.
- /P (1)

UR3 TransformParams

- /V /2,2
- "Document/[FullSave]
- "Form/[form field rights]
- "Signature/[Modify]
- "Annots/[annot rights]
- "EF/[embedded file rights]
- "FormEX/[form field rights]

1-6 define usage rights
A.4 Revocation Checking Process

**Revocation Checking Order of Precedence:** Check type and order is configurable via application.

1. **Get service providers:** Make both OCSP and CRL service providers available. Configurable via the cRevocationCheckers preference.

2. **Check for embedded revocation information:** Check if the signature embedded the revocation status in the signature; check OCSP first. CRL. Configurable via iUse Archived RevInfo.

3. **Check local CRL cache:** If embedded data is not used, check whether the CRL is stored in the application’s local cache. For example, at C:\Documents and Settings\<user>\Application Data\Adobe\Adobe\<Security\CRLCache.

4. **Check remote OCSP:** iReqRevCheck preference specifies whether check is required to succeed and what should happen if it doesn’t. check occurs as follows:
   - If there is a custom certificate preference, use those settings (e.g., at iURLToConsult and use value in sURL).
   - If no custom certificate preference, use Adobe_OCSPRevCheck setting (e.g., look at iURLToConsult and use value in sURL).
   - If no registry preference, use the AIA extension in the certificate.
   - If no OCSP occurs or it fails, go to the next step and try to use a
**A.5 Signature Status and Troubleshooting Guide**

**Document status appears in the Document Message Bar.**

### Signature Status and Message Bar

- **Certified**
  - First signature is a certification signature.
  - Document has not changed or only contains permitted changes.

- **Valid**
  - First signature is a certification signature.
  - Document has not changed or only contains permitted changes.

- **Problematic**
  - Unverified, Certificate validation problem.
  - Unsigned changes after this signature.

- **Invalid**
  - Signature check not executed.
  - Illegal changes made, document corrupted, or policy restrictions violated.

**Sig Status**

- **Certified**
- **Valid**
- **Problematic**
- **Invalid**

**Identity Check**

- **Certified**
  - First signature is a certification signature.
- **Valid**
  - Document has not changed.
- **Problematic**
  - Unauthorized changes exist.
- **Unknown**
  - Check has not executed.

**Document integrity check**

- **Certified**
  - Document has not changed or only contains permitted changes.
- **Valid**
  - Document has not changed.
- **Problematic**
  - Unauthorized changes exist.
- **Unknown**
  - Integrity check has not executed.

**Status depends on two checks:**

1. **Signer's Identity:** Verifies the signer's certificate is trusted (in the validator's list of trusted identities) and valid at the time specified by Acrobat/Reader configuration: signing time, timestamp time, or current time.
2. **Document integrity:** Verifies the signed content hasn't changed or that it has only changed in ways permitted by the signer.

**There are two types of signatures:**

- **Certification:** Certifies the document. Only one allowed per document and it must be the first one. Can lock the document or specify allowed actions such as signing, form fill in, and commenting.
- **Approval:** Signs but doesn't certify. Any number allowed.
A.6 Security workflow for encryption-permissions

**OVERVIEW STEPS:**
1: Select a method
2: Create a policy?
3: Choose content to encrypt
4: Set permissions
5: Review settings
6: Save work

**CHOOSE METHOD:**
Each method can be saved as a reusable policy.

**CHOOSE COMPONENTS:**
Backward compatibility depends on the selected algorithm and the encrypted components.

**SET PERMISSIONS**
Certificate and ALCRM allow setting permissions at the user level.

**Pros:**
- No password to remember.
- Key not susceptible to brute force discovery and resides only on the recipients machine.
- Can encrypt documents for specific people.
- Can use certificates from trusted 3rd party.
- Specifies different permissions for users.
- Leverages LDAP for recipients & groups.
- Users must have a digital ID.
- Requires distributing/managing digital IDs.
- Full support appears first in 6.0.

**Pros:**
- Backward-compatible to Acrobat 3.0 only for certain encryption levels.
- Simple and easily understood.
- Share documents by sharing the password.
- Different open & permission password.
- Password strength is critical.
- Users share the same permissions.
- Disabled when in FIPS mode.

**Pros:**
- Centralized policy administration.
- Document auditing.
- Allows setting permissions for separate tasks such as opening, editing, and so on.
- Can specify different permissions for users.
- Leverages LDAP for recipients & groups.
- Offline control: Can specify a validity time limit after which document expires and is locked

**Cons:**
- Requires a network connection, an administrator, and a LiveCycle Server.
What’s Changed Across Releases

This document provides guidelines for deploying and configuring Adobe Acrobat and Adobe Reader for use in digital signature and document security workflows.

B.7 What’s new for 9.2

B.8 What’s new for 9.1

B.9 What’s new for 9.0

B.9.1 Enhanced security

Refer to the documents in the Application Security Library. This feature interacts with certified documents.

B.9.2 Security setting import and export

Acrobat 9 provides a more robust and detailed mechanism for importing and exporting security settings than was provided by FDF files. With Acrobat 9, all settings can be migrated to new machines, saved during upgrades, or distributed via a server. The new feature includes the following

- A new XML format for storing security settings which are saved in an empty PDF.
- Separate site and personal settings.
- A secure way of communicating and installing site-wide security settings.
- A user interface for selecting which settings are imported and exported.
- A mechanism for automatically installing or updating security settings from a server.

B.9.3 Signatures and signing workflows

Note: For additional detail, refer to “Adobe Acrobat 9 Digital Signatures, Changes and Improvements”.
What's Changed Across Releases

Signatures and signing workflows

- **User interface changes**: Numerous user interface improvements throughout, including the Signature Properties dialog and the Certificate Viewer. The Digital Signature toolbar was moved from the Advanced Editing toolbar to the Forms menu.

- **Indication of overall signature validity state**: Changes the Document Message Bar to specify the overall validity and integrity of the document. You can use this information to make a quick decision about the document.

- **Revision tracking in Signature panel**: Describes the changes made to each revision of the document. The changes to a revision include all changes made to the document between signing. This information is displayed in the signature panel for an Acrobat form.

- **Clearing signatures**: Prior to Acrobat 9.0, a signature that was not specifically protected by field locking could be cleared by anybody while protected fields could only be cleared by somebody with the private key that applied the signature. With version 9.0, all signatures can only be cleared by somebody with the private key that applied the signature.

- **Change in status of forms that take multiple signatures**: Allows additional signatures to be applied to a document, without changing earlier signatures to a “Valid with subsequent changes” status (green check with a yellow warning triangle).

- **Allowed and disallowed changes**: Refines the definition of the kinds of changes that can be made to a certified or signed document without invalidating the signatures applied to the document. Disallowed changes invalidate the signatures on a document.

- **Individual signature status icons removed from signature fields**: Removes the signature status icon from the signature appearance. The Document Message Bar is a better source of information about overall signature validity and integrity of a document, and the Signature panel’s list of signatures is a better source of information about individual signature status.

- **Individual signature status icon meanings**: Replaces Acrobat 8 icons with new icons that are compatible with the new overall signature validity state icons

- **Lock document after last signature**: Allows the last signer of the document to lock the document, which prevents further changes

- **Certification requirement for XML forms**: Adds requirement to certification of forms created with LiveCycle Designer ES

**Form field and form behavior changes**:

- **Document lock on signing**: A default signing action locks all form fields in a form. Users who want to create a multi-signature workflow must take explicit action to not lock fields they want subsequent fillers to use. This affects only forms created with Acrobat 9.0 and later.

- **Add a new signature and rollup status and icon “form modified during progressive form fill-in and signing.”**: This is similar to in concept (but not the same as) the current “green check + yellow triangle” signature status. It is an advisory warning for users to look more closely at the form and does not invalidate signatures.

- **Signatures followed by a form field change are marked with a “form modified” status and icon rather than a valid signature status and icon, unless the signature included field locking (MDP+). In that case it can be marked with a “valid signature” status and icon.**

- **If all signatures are valid or form modified, then the rollup status is “form modified.” The document message bar advises the user that the form has gone through a multi-step fill-in workflow and that they should inspect the document history in the signature panel.**

- **These same rules apply for both uncertified and certified documents. (That is, signature invalidation on modified field values and “form modified” signature state for multi-step workflows apply to both types of documents.)**
Post-signing changes warning: For Acrobat 8.x, the “document modified after signing” warning (the green check with yellow triangle icon) was triggered by any document change of any kind. Now the warning icons and text have changed. If a document is changed after the last signature, a warning is shown indicating that there are unsigned changes in the document. This could happen in the course of normal workflow as a form is filled in. When signed, the warning is no longer shown. The warning could also indicate an attempt to tamper with the document after it was signed.

- Changes not listed above result in invalid signature status.
- The “document modified after signing” warning is eliminated and never shown.
- The “form modified warning” is shown for any form field or data change that follows a non-field-locking (MDP+) signature.
- The “form modified warning” is displayed when needed for documents signed with a certification signature as well as an approval signature.

Certification of XML dynamic forms: For signed dynamic XML forms in Acrobat 8.0, everything is digested and signed for visible certification signatures. Everything but rendering components are digested and signed invisible certification signatures. For Acrobat 9.0, the rendering components are never signed.

B.9.4 FIPS

The FIPS-mode encryption module has been updated and changed to RSA BSAFE Crypto-C Micro Edition (ME) 2.1.0.3 cryptographic module. The FIPS validation status for RSA BSAFE Crypto-C ME 2.1.0.3 is available at http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2007.htm#865. For details, see “Turning on FIPS Mode” on page 100.

B.9.5 Certificate Security

- AES 256-bit encryption algorithm can be used.

B.9.6 Reader Enablement

- AES 256-bit encryption algorithm can be used.

B.9.7 Password security

For documents with password security the following changes have been made:

- AES 256-bit encryption algorithm can be used.
- If AES 256-bit encryption is used, then the password may use Unicode.
- Acrobat 8 compatibility allows a programmatic call with a null password for owner permission to succeed if the owner password really is null. Acrobat 9 does not. For Acrobat 9, if the owner password is null but the user password is not null, you must ask for owner permission with the non-null user password.
- Acrobat 9.0 now allows full Unicode pass phrases up to 128 characters in length (an actual limit of 128 UTF-8 bytes). Acrobat 8.x and earlier limits passwords to 32 characters maximum and almost entirely to the Latin alphabet (strictly, PDFDocEncoding).
B.9.8 Adobe Reader usage rights and feature enablement

Acrobat users can author documents that enable Reader users to take advantage of features that are otherwise unavailable. For example, a Reader user can sign a form field or save a form with their data when the author has enabled those features for that document. Support for earlier versions of this feature which are based on UB1 are disabled. Only UB3 is allowed and error messages have been improved.

B.9.9 Security and Encryption for of PDF packages (portfolios)

- The Modify and Secure Portfolio menu only applies encryption to the cover sheet not child documents.
- Failure to allow form fill-in results in blocking the recipient’s ability to modify the portfolio’s child documents.
- Encryption must be set at the child document level.
- Reader enablement rights must be set at the child document level.

B.9.10 Signing of PDF packages (portfolios)

Packages have been renamed as “portfolios.” The changes are as follows:

- The PDF portfolio’s user interface provides the standard signing mechanism. The signature is placed on the portfolio cover sheet (choose View > Portfolio > Cover sheet). The cover sheet may be signed in the traditional way either with an approval or certification signature.
- Signing or certifying a portfolio results in locking down its children (the assembled documents) while they reside within the portfolio.
- Certified documents that are placed inside of a portfolio will behave as if they are uncertified. Once removed from the portfolio, the contained certified document regains its certified-specific behaviors.
- You can sign but you can’t certify a document once it is placed inside a portfolio.

B.9.11 Attachments

Files with the file extensions .pkg and .jar have been added to the default attachment blacklist.

B.9.12 Product renaming

The Adobe LiveCycle Policy Server has been renamed to Adobe LiveCycle Rights Management Server.

B.10 What’s New for 8.1

- FIPS mode: Acrobat and Reader can provide encryption via a Federal Information Processing Standard (FIPS) 140-2 mode. When the FIPS mode is on, encryption uses FIPS-approved algorithms provided by the RSA BSAFE Crypto-C Micro Edition version 2.1 encryption module. FIPS mode is
only supported on Windows and can only be turned on by editing the registry. For details, see “Turning on FIPS Mode” on page 100.

- **digital ID authentication caching**: The `bWinCacheSessionHandles` registry preference has been added to specify whether to retain cryptographic service provider (CSP) handles when a user authenticates to a digital ID. When enabled, users can access their ID without reentering their password unless they log out or the session ends. For example, smart card users won’t have to enter their PIN with each use when `bWinCacheSessionHandles` is set to 1. The impact of this preference will vary based on the CSP in use. The setting does not affect Windows CSPs.

- **Refreshing the digital ID list**: A Refresh IDs button in the signing dialog allows users to refresh the list of available digital IDs after the signing process has already been initiated. For example, if a smart card user initiates the signing workflow without first attaching their card reader, their digital ID will not appear in the Digital ID drop down list. Refresh IDs allows users to attach their reader, insert their card, and refresh the list without having to restart the signing process.

- **Encryption level name changes**: The word “High” was removed from “High (128-bit RC4/AES).” For example, the levels are now just “128-bit AES.”

- **Document integrity checking and script detection**: During signature validation, the detection of a script in a PDF does not cause Acrobat to flag the document as changed. Acrobat 8.1 compares the digitally signed and current versions of the document to determine if the current version has been modified. Acrobat 8.0 considered all scripts executed during document construction as a document modification even if no changes were made (e.g. it performed a read-only query or some other “no change” action).

- **PDF portfolio security**: The following is a list of expected behaviors when a PDF package is secured:
  - If the cover sheet is signed with an approval or certification signature, then document recipients will not be able to edit and save any documents in the package.
  - If the cover sheet is signed or Reader-enabled, then the security method cannot be changed for any documents in the package.
  - If a package’s security settings do not allow changes, then that setting is inherited and it will not be possible to edit and save any child documents. For example, form fill-in and changing the security settings would not be allowed for any document in the package.
  - It is possible to validate a certification signature on the cover sheet when it is Reader-enabled.
  - If the permission to allow deleting, inserting, and extracting pages is turned on, form fill-in cannot be turned on for documents in the package.
  - If form fill-in is allowed on the cover sheet, then editing and saving is allowed for any document in the package.

**B.11 What’s New for 8.0**

Acrobat 8.0 ships with major improvements to its security features, including its user interface and core functionality. Nearly all of its features are more powerful and easier to use as a result of streamlined workflows and a redesigned user interface. All security-related menu items are accessible through a fewer number of clicks, and administrators now have more control over the end user experience via a set of expanded registry preferences.
For an overview, see the following:

- Digital ID Management
- Certificate Processing and Viewing Enhancements
- Signature Enhancements
- Document Security Enhancements
- Application Environment Preference Improvements

**B.11.1 Digital ID Management**

In addition to the top level menus that have been redesigned to improve usability, managing digital IDs through the Security Settings Console is now easier and less subject to user-error:

- The ID **Usage Options** menu allows users to individually specify which IDs to use for signing, certifying, and encrypting documents.
- The ID **Usage Options** menu enables users to associate a user-friendly name with an ID card that lists the ID’s basic details. The friendly name simplifies choosing an ID during signing.
- The **Remove ID** button only allows deleting self-signed digital ID created in Acrobat, thereby preventing accidental deletion of critical IDs. Users can only remove an ID in a .pfx file in the Acrobat store if it was created in Acrobat 8.0 or later.
- A **Manage Attribute Certificates** option enables users to associate an attribute certificate with an ID used for signing.

**B.11.2 Certificate Processing and Viewing Enhancements**

The application’s handling of certificates has been both extended and improved:

- **Attribute certificates** are now supported and can be managed via Security Settings Console.
- **Signature field customization**: Additional seed values have been added to allow authors to require certain certificate attributes in order to sign a field (Table 4).
- **Certificate Viewer** enhancements were introduced to improve usability and provide more features, including but not limited to the display of:
  - Attribute certificate data.
  - ISIS-MTT-required OIDs (in the **Certificate Data** drop down list on the Details tab).
  - The private key location, if known.
  - The validation model: *shell* or *chain* appears at the bottom of the Certificate Viewer.

**B.11.3 Signature Enhancements**

Many digital signature features have been added or improved. The changes significantly reduce the amount of effort it takes to sign while at the same time extending Acrobat’s signing capabilities:

- **Roaming IDs**: Users can access their roaming ID account on a remote server and sign from any location. Administrators can create custom workflows, manage IDs from a central location, and so on.
- **Signature field customization**: Additional seed values have been added to allow customization of signing workflows such as who can sign and what information is required (Table 4).
Streamlined signature workflow: Vast changes in a now user-friendly signing dialog provide more detail with fewer steps. All of the signing details can be selected on one dialog:

- Users can select from a list of available IDs, view an ID card that displays the digital ID's high level details, and click on the card to open it in the Certificate Viewer.
- A dynamic **Password** field that only appears when needed.
- Signature appearances display in a drop-down list.
- Depending on application settings, **Location**, **Contact**, and **Reasons** fields are available.

**Algorithms**: Support for more signing algorithms, including SHA256, SHA384, SHA512, RIPEMD160.

**Application settings**: The signing environment now has additional options. Users can turn on and off document warnings, force a warning review, turn off and on the **Location**, **Contact**, and **Reasons** fields, and require signing in preview mode. Administrators can preconfigure these settings via the registry and, in some cases, prevent end-user modification.

**Preview mode**: Preview mode suppresses dynamic content that could change the appearance of a signed document and analyzes the document for constructs that may be problematic for signing. The document message bar tells the user if the document is safe to sign, the level of PDF/SigQ compliance, and provides links to the PDF/SigQ Conformance Report dialog which lists potential problems. **View Signed version** also invokes the preview mode so that the signature validator can see what was signed by viewing the document in a “safe mode.”

**B.11.4 Document Security Enhancements**

Changes to the security features include:

- **Usability**: Top level menu items have been simplified to streamline workflows and improve usability.
- **New ID choices**: When adding a digital ID during a certificate security workflow, users can add IDs from PKCS#11 devices such as smart cards and tokens.
- **Reader enabling**: It is now possible to enable a document for signing and saving within Adobe Reader. While this is not solely a security feature, signatures provide protection against unknown document changes, and each document version (1 per signature) is always available for viewing.

**B.11.5 Application Environment Preference Improvements**

The user interface is redesigned for usability:

- **Trust Manager**: Trust settings now include simplified options for accessing resources that exist outside of a document, including attachments, Internet access, and external content. All of the sub-dialogs have been simplified, and advanced configuration details are described in the administration guide.
- **Multimedia Trust**: Options have been removed from the Trust Manager panel have been given a unique home. The options have been simplified and its interaction with the certification signatures has been made more intuitive (signature trust settings interact with the **Trusted Documents** and **Untrusted Documents** settings in Multimedia Trust.)
The Acrobat family of products introduces new features and enhancements with each release. The following technical details list only a fraction of the changes that occur with each new product release:

- Digital ID Related Files and Storage Mechanisms
- Encryption Algorithms
- Algorithms Used During Digital Signature Creation
- Seed Values
- Signature Appearances
- XML Form Support for Signatures
- Directory Servers
- FDF (Data Exchange) Files

### B.12.1 Digital ID Related Files and Storage Mechanisms

A digital ID may reside in a local or networked file or on some external hardware. There are several digital ID-related file types and storage mechanisms a user might encounter in signing and certificate encryption processes (Table 1). Digital IDs (both the certificate with the private key) and certificates (with a public key but no private key) are provided to Acrobat via digital ID service providers (one type of “cryptographic service provider” or CSPs).

In many cases, the digital ID is stored on a local or networked file. Common file locations include the Windows Certificate Store (where they can be used both by Acrobat and other Windows applications) and the default Acrobat cache (used only by the Acrobat family of products). Others IDs may reside on external PKCS#11 hardware that is plugged into the computer or on a roaming ID server. Regardless of location the application remembers the location when the ID is registered (imported) and is added to the Security Settings Console’s digital ID list.

The Acrobat family of products supports the following digital ID providers:

- **PKCS#12 files**: A common file format that contains the entire digital ID. It is used by Acrobat on Windows as well as Macintosh.
- **Windows Certificate Store**: A local store that can import and export various file formats and that can be used by both Windows MSCAPI-compliant programs including Acrobat products.
- **PKCS#11 devices**: External devices that store digital ID data. Users do not interact with a file.
- **Roaming ID servers**: The private key is known only to a remote server. The server sends the certificate and its public key to users after they authenticate. Users can import and export the certificate from Acrobat.
- **APF files**: A legacy format that is no longer used.
Table 1 Digital ID-related file types

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>5.x</th>
<th>6.x</th>
<th>7.x</th>
<th>8.x</th>
<th>9.x</th>
</tr>
</thead>
<tbody>
<tr>
<td>.acrobat security</td>
<td>An XML format encapsulated in a PDF which stores security settings for import and export.</td>
<td>Import</td>
<td>Export</td>
<td>Import</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PKCS#12: .pfx (Win), .p12 (Mac)</td>
<td><strong>Personal Information Exchange Syntax Standard:</strong> specifies a portable, password protected, and encrypted format for storing or transporting certificates.</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
</tr>
<tr>
<td>.fdf</td>
<td>An Adobe file data exchange format used for importing and exporting settings and certificates (usually PKCS#12 files).</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td>PKCS#7: .p7b, .p7c</td>
<td><strong>Certificate Message Syntax (CMS):</strong> Files with .p7b and .p7c extensions are registered by the Windows OS. Acrobat products can import and export these files.</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Certificate and public key only</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
</tr>
<tr>
<td>.cer</td>
<td><strong>Certificate format:</strong> A Microsoft format for digital IDs usually stored in the Windows Certificate Store.</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
<td>Export</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Certificate and public key only</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
</tr>
<tr>
<td>.apf</td>
<td><strong>Adobe Profile Files (Legacy):</strong> Not used after Acrobat 5. Files can be upgraded by double clicking them.</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td><strong>Contains:</strong> Digital ID (public and private keys)</td>
<td>Export</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
<td>Import</td>
</tr>
</tbody>
</table>

B.12.1.1 APF files

The ability to create an APF file (Acrobat Personal File) only existed in Acrobat 4 and 5. The file was a container for an Acrobat generated, self-signed digital ID plus an address book that held other peoples public keys for the purpose of applying Certificate Security (for signature validation).

Beginning with Acrobat 6, the ability to create an APF file was dropped because P12/PFX usage was introduced along with a separate address book for managing other peoples certificates. Acrobat 6 would import the certificates from an APF file and allow export of a digital ID to a P12 file. Acrobat 7 would still open the APF file and allow access to its contents, it no longer offered the ability to export your private key. Therefore, if you want to convert your APF file to a P12 file you can only use Acrobat 6.

Acrobat 9 does not support APF in any way, so if you need to use or access an APF file after Acrobat 9 is installed, you will have to uninstall A9, reinstall A6, import and then export the digital ID from the APF to a P12 file, and then reinstall A9.

B.12.2 Encryption Algorithms

Acrobat continues to support more sophisticated encryption algorithms when applying document security (Table 2). When configuring password security, users also have the opportunity to set the Acrobat compatibility level. Choosing to remain compatible with earlier versions of Acrobat results in the use of older algorithms and limitations in encryption options. For example, choosing a compatibility level of Acrobat 5.0 or earlier does not enable encrypting document contents separately from metadata.

When you apply certificate security you will be asked to select between one of the following algorithms:
What's Changed Across Releases

Encryption Algorithms

- 256-bit AES is only compatible with Acrobat 9.0 and later.
- **128-bit AES** is only compatible with Acrobat 7.0 and later. It is mandated for some U.S. government documents because it is more secure than RC4. AES has a bigger file size and adds up to 32 bytes per stream.
- **128-bit RC4** is compatible with Acrobat 6.0 and later as well as other non-Adobe and Adobe PDF clients that have not implemented AES. RC4 has a smaller file size by about 32 bytes per stream.

Acrobat and Adobe Reader 9 use the following algorithms:

- NLR (weak) encryption: RC4 with 40 bit key
- Normal encryption
  - RC4 with 40 and 128 bit key
  - AES128 with 128 bit key, CBC (cipher block chaining) mode
  - AES256 with 256 bit key, CBC (cipher block chaining) mode
- Password generation
  - CBC (cipher block chaining) encryption for AES
  - RC4 with 40 bit key (Acrobat 8 and earlier encryption)
  - AES256 for Acrobat 9 password generation ECB (electronic codebook) mode

<table>
<thead>
<tr>
<th>V.</th>
<th>Password Security</th>
<th>Certificate Security</th>
<th>Adobe Policy Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.0</td>
<td><strong>Algorithms</strong>: Added 256-bit AES</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The encryption module from RSA has been updated.</td>
<td><strong>Algorithms</strong>: Added 256-bit AES</td>
<td><strong>Algorithms</strong>: Added 256-bit AES</td>
</tr>
<tr>
<td>8.1</td>
<td>Same as 7.0 except if FIPS mode is turned on, password security is not available.</td>
<td>Same as 7.0. If FIPS mode is turned on, RC4 is not available.</td>
<td>Same as 7.0</td>
</tr>
<tr>
<td>8.0</td>
<td>Same as 7.0</td>
<td>Same as 7.0</td>
<td>Same as 7.0</td>
</tr>
<tr>
<td>7.0</td>
<td><strong>Algorithms</strong>: 128-bit RC4, 128-bit AES</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents, all but metadata, only attachments.</td>
<td><strong>Algorithms</strong>: Self-sign &amp; third-party</td>
<td><strong>Algorithms</strong>: 128-bit AES</td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents, all but metadata, only attachments.</td>
<td><strong>Algorithms</strong>: 128-bit RC4, 128-bit AES</td>
<td><strong>Options</strong>: All contents, all but metadata, only attachments.</td>
</tr>
<tr>
<td>6.0</td>
<td><strong>Algorithms</strong>: 40 &amp; 128-bit RC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents, all but metadata.</td>
<td><strong>Algorithms</strong>: Self-sign &amp; third-party</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents, all but metadata.</td>
<td><strong>Algorithms</strong>: 40 &amp; 128-bit RC4</td>
<td>N/A</td>
</tr>
<tr>
<td>5.0</td>
<td><strong>Algorithms</strong>: 40 &amp; 128-bit RC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents.</td>
<td><strong>Algorithms</strong>: Self-sign p7b &amp; apf files only.</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents.</td>
<td><strong>Algorithms</strong>: 40 &amp; 128-bit RC4</td>
<td>N/A</td>
</tr>
<tr>
<td>4.0</td>
<td><strong>Algorithms</strong>: 40-bit RC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>4.0.5</td>
<td><strong>Algorithms</strong>: 40-bit RC4 (64-bit decrypt)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong>: All contents.</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3.0</td>
<td><strong>Algorithms</strong>: 40-bit RC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2.0</td>
<td><strong>Algorithms</strong>: 40-bit RC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
B.12.3 Algorithms Used During Digital Signature Creation

Applying a digital signature involves the use of two algorithms: one for creating the message digest (a document hash) and one for encrypting the digest:

- **Message digest**: SHA1 is the default algorithm used to digest the data that is being signed. Specifying an alternate algorithm is possible through registry configuration.
- **Encryption algorithm**: That algorithm is selected based on the signing digital ID’s private key which specifies an algorithm and key length when the public-private key pair is generated.
- **Libraries**: CryptoC 6.1.1 and CryptoC 6.3.2 are statically linked libraries (not dlls).

<table>
<thead>
<tr>
<th>Ver.</th>
<th>Message Digest Algorithms</th>
<th>Encryption Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-5.0</td>
<td>MD5, SHA1</td>
<td>RSA: Up to 1024-bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSA: None</td>
</tr>
<tr>
<td>6.0</td>
<td>MD5, SHA1</td>
<td>RSA: Up to 4096-bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSA: None</td>
</tr>
<tr>
<td>7.0</td>
<td>MD5, SHA1, SHA256</td>
<td>RSA and DSA: Up to 4096-bit</td>
</tr>
<tr>
<td>8.0</td>
<td>MD5, SHA1, SHA256, SHA384, SHA512, RIPEMD160</td>
<td>RSA and DSA: Up to 4096-bit</td>
</tr>
<tr>
<td>9.0</td>
<td>MD5, SHA1, SHA256, SHA384, SHA512, RIPEMD160</td>
<td>RSA and DSA: Up to 4096-bit</td>
</tr>
</tbody>
</table>

MDS is used for Acrobat8 and earlier password generation and file encryption
SHA256 is used for Acrobat password generation and file encryption

### B.12.4 Seed Values

Signature field seed values enable document authors to limit a signer’s choices when signing a particular signature field. As shown in Table 4, document authors can create documents with behaviors and features that meet specific business needs, thereby enabling administrative control of signature workflows.

For more information, refer to the following (Go to [http://www.adobe.com/go/acrobat_security]):

- *Developing Acrobat Applications with JavaScript*
- *JavaScript for Acrobat® API Reference*

<table>
<thead>
<tr>
<th>Seed value</th>
<th>First support for seed value</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>certspec</td>
<td>Specifies that certain certificates must be used for a particular signature field.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td><code>6.0-7.x</code></td>
<td>Supports subject, issuer, and oid.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>8.x</code></td>
<td>Adds support for subjectDN, issuerDN, keyUsage, url, and urlType</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>filter</td>
<td>The language-independent name of the security handler to be used when signing.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
What's Changed Across Releases

### Signature Appearances

#### Table 4  Seed values: Changes across releases

<table>
<thead>
<tr>
<th>Seed value</th>
<th>First support for seed value</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>flags</td>
<td>A set of bit flags controlling which properties are required.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>8.0: 16: legalAttestations, 32: shouldAddRevInfo, and 64: digestMethod.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>legalAttestations</td>
<td>A list of legal attestations that the user can use when creating an MDP (certification) signature.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>mdp</td>
<td>Can be used to force a certification signature as well as to control permitted document changes.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>reasons</td>
<td>A list of reasons that the user is allowed to use when signing.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>8.0: Supports disabling signing reasons.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subFilter</td>
<td>An array of acceptable signature formats.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>timeStampspec</td>
<td>Specifies a timestamp server using the url and flags properties.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>version</td>
<td>The signature handler version to be used to sign the signature field. Valid values are 1 and 2.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>8.0: Must be set to 2 if this seed value object contains Acrobat 8-specific content marked as required.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>digestMethod</td>
<td>The algorithm used to created the message digest.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>6.0-7.x: MDS, SHA1.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8.0: Adds support for SHA256, SHA384, SHA512, and RIPEMD160.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>shouldAddRevInfo</td>
<td>Controls how the application does certificate and chain revocation checking.</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lockDocument</td>
<td>Allows the author to add a Lock Document checkbox to the signing dialog so a signer can lock the document at the time of signing.</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>AppearanceFilter</td>
<td>A text string naming the appearance required to be used when signing the signature field.</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

#### Table 5  Changes in signature appearances

<table>
<thead>
<tr>
<th>V.</th>
<th>Filename</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.0</td>
<td>No change</td>
<td>Signature status icon no longer appears with the appearance.</td>
</tr>
<tr>
<td>8.0</td>
<td>No change</td>
<td>No change.</td>
</tr>
<tr>
<td>7.x:</td>
<td>No change</td>
<td>No change.</td>
</tr>
<tr>
<td>6.0:</td>
<td>appearance.acrodata</td>
<td>One signature appearance file per OS login. Each page in the PDF corresponds to a signature appearance. Each page also contains private data that is hung off of the Page object. This data contains the settings for checkboxes and other options in the Configure Signature Appearance dialog.</td>
</tr>
<tr>
<td>5.05:</td>
<td>&lt;apf basename&gt;.pdf</td>
<td>One appearance file per apf file. 5.0 appearance files are forward compatible with 6.0.</td>
</tr>
</tbody>
</table>

---

**B.12.5 Signature Appearances**

The format has always been PDF, but appearance file names and behavior have evolved across releases.

---

*Acrobat Family of Products*
What's Changed Across Releases

XML Form Support for Signatures

**B.12.6 XML Form Support for Signatures**

Support for security features in dynamic XML forms authored with Designer has improved across versions. With version 8.0, there is full support for the following:

- **Approval signatures**: Sometimes referred to as signing.
- **Certification signatures**: Sometimes referred to as certifying.
- **MDP+**: modification-detection-protection (field locking).
- **Reader extensions**: Enabling usage rights so Reader users can sign documents with existing fields.

**Table 6 Signature type support in XML forms**

<table>
<thead>
<tr>
<th>Sig. Type</th>
<th>Static XML forms</th>
<th>Dynamic XML forms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.x</td>
<td>7.x</td>
</tr>
<tr>
<td>Approval</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Certification (DocMDP)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Field protection (FieldMDP)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Reader enabling (UR3)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**B.12.7 Directory Servers**

Acrobat products ship with preconfigured directory servers. The servers are used by the Trusted Identity Manager to locate certificates for import. Users can trust the imported certificates for signing and certifying documents as well as for encrypting documents prior to sending them to the certificate owner.

**Table 7 LDAP certificate repositories**

<table>
<thead>
<tr>
<th>Version</th>
<th>Default Directory Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.x</td>
<td>VeriSign Internet Directory Service</td>
</tr>
<tr>
<td></td>
<td>GeoTrust Directory Service</td>
</tr>
<tr>
<td></td>
<td>IDtree Directory Service</td>
</tr>
<tr>
<td>8.0</td>
<td>VeriSign Internet Directory Service</td>
</tr>
<tr>
<td>9.0</td>
<td>VeriSign Internet Directory Service</td>
</tr>
</tbody>
</table>

**B.12.8 FDF (Data Exchange) Files**

The FDF format and .fdf files enable the import and export of certificate data and security settings. End users can share application settings, and administrators can distribute trust anchors and server settings across their organization. Acrobat products can create the files as part of the export process, or files can be programatically generated by a custom program or script that writes a file conforming to the FDF Data Exchange Specification.

**Tip:** With 9.0, FDF is being deprecated and replaced with the security setting import/export feature.
### Table 8 FDF changes across releases

<table>
<thead>
<tr>
<th>First support for FDF feature</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import and export of Acrobat-generated self sign certificates.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>FDF file signing for origin verification.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Import and export of any supported certificate type (including those in the Windows store).</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Import and export of directory server settings.</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Import and export of Adobe LifeCycle Rights Management Server settings.</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Import and export of timestamp server settings.</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Import and export of roaming ID server settings.</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
As described in “Basic Concepts” on page 17, a digital ID consists of two main parts: a certificate with a public key and a private key. Participants in signing and certificate security workflows need to exchange the public part (the certificate) of their digital ID. Once you obtain someone’s certificate and add it to your trusted identities list, you can encrypt documents for them. If their certificate does not already chain up to a trust anchor that you have specified, you can set the certificate’s trust level so that you can validate the owner’s signature.

Tip: This feature requires a basic understanding of how Acrobat defines “trust” and “trusted identities.” For more information, see “Basic Concepts” on page 17.

Understanding what a trusted identity is and how trust levels are set can help you streamline workflows and troubleshoot problems. For example, you can add trusted identities ahead of time and individually set each certificate’s trust settings. In enterprise settings where certificates are stored on a directory server, you may also be able to search for certificates to expand your list of trusted identities.

The following sections provide details about managing trust and trusted identities in Acrobat and Adobe Reader.

C.13 Adding Someone to Your Trusted Identity List

You build a list of trusted identities by getting digital ID certificates from those who will be participating in signing and certificate security workflows. You get this information from a server, a file, or from a signed document. For signing workflows, you can get this information during the signature validation process. For certificate security workflows involving encryption, you must request the information ahead of time so you can encrypt the document with the document recipient’s public key.

C.13.1 Adding a Certificate From a Signature

When you receive a signed document from someone whose certificate is not in your trusted identity list AND does not chain up to a trust anchor (another certificate that is trusted), the signing certificate’s validity is unknown and a related icon and message appear in the document message bar. To validate the signature, you will need to trust one of the certificates in the certification chain. You could trust the signer (the end-entity certificate), one of the EE certificate issuer (an intermediate certificate), or the topmost certificate authority (the root).

Because revocation checking does not occur for certificates that are directly trusted (a trust anchor), it is best practice to trust a certificate other than the signer’s. That is, trust a certificate as high up in the chain as is practical for your signing workflows.

To add a certificate to your trusted identities list directly from a signature:

1. Right click on the signature and choose Show Signature Properties.
2. Choose Show Certificate.
3. When the Certificate Viewer appears, choose the Trust tab.

4. Choose **Add to Trusted Identities** (Figure 1).

5. Set the certificate trust settings as described in “Setting Certificate Trust” on page 338.

   **Figure 1 Certificate Viewer: Trust tab**
   
   ![Certificate Viewer: Trust tab]

---

**C.13.2 Requesting a Digital ID via Email**

Email requests for digital ID information use .acrobatsecurity or FDF files. For details, see Chapter 3, “Migrating and Sharing Security Settings”.

For details, see “Requesting a Certificate via Email” on page 48.

**C.13.3 Importing a Certificate From a File**

Acrobat and Adobe Reader are can export certificates to a file so that they can be shared as needed. To import certificates, follow the instructions described in Chapter 3, “Migrating and Sharing Security Settings”.

However, certificates may also exist in other file types such as .cer, .p7b, and so on. To import certificates from these file types:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Manage Trusted Identities**.

2. Choose **Add Contacts**.

3. Choose **Browse**.

4. Browse to the contact file location.

5. Select the file.
6. Choose **Open**.

**Figure 2 Importing digital ID data**

7. Choose **Import**.

8. Choose **OK** when the confirmation dialog appears.

### C.13.4 Searching for Digital ID Certificates

The search feature allows you to search a list of directories for certificates. If no directories have been previously specified, the **Search** button will NOT appear. The list of search servers in the Directories drop-down list is populated through three mechanisms:

- The default server settings that ship with Adobe Acrobat and Adobe Reader.
- The Windows Certificate Store if the user has turned on this option.
- User-specified directory servers the user has added in the Security Settings Console. For details, see “Using Directory Servers to Add Trusted Identities” on page 341.

**Tip:** Home users do not usually need to change the directory server list. Users in enterprise environments typically have the list preconfigured by their system administrator.
To search for a certificate so that you can add one or more people to your trusted identities list:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Manage Trusted Identities**.

2. Choose **Add Contacts**.

3. Choose **Search**.

4. Configure the search options:
   - Choose **Search all directories** or select a directory and optional group.
     
     Searching all directories may take some time. In a business environment, it is often expedient to just select the company's LDAP directory.
     
   - Enter a name and/or email address to search. This is an AND search. Using both fields only returns results that match both criteria.

5. Choose **Search**.

6. Select a name from the search results.

7. Choose **OK**.

8. If the desired entries are found, choose **Import**.

9. Choose **OK** when the confirmation dialog appears.
C.14 Certificate Trust Settings

Contacts in the trusted identities list should be associated with one or more certificates. Those certificate’s trust settings may be individually configured. Choosing to not trust a certificate does not prevent a document from displaying, but it will result in signatures having a problematic status. The status is represented by a yellow triangle in the Document Message Bar, Signatures pane, and the Signature Validation Status dialog (Figure 5). For each contact for whom you will encrypt a document with certificate security, one certificate can also be selected as the default for encryption.

Certificate trust settings have the following features:

- Trust settings are configured in the Trusted Identity Manager ahead of time, at the time of import, or directly from a signature.
Trust settings can be viewed in the Trusted Identity Manager by choosing Edit Trust or by choosing the Trust tab in the Certificate Viewer (Figure 6).

Certificates can be separately trusted for approval signatures and certification signatures.

Certificates can be individually configured to trust operations such as signing, certification, and allowing items such as dynamic content and JavaScript in certified documents. These settings interact with application environment settings.

**Figure 6 Certificate trust settings**

![Certificate trust settings](image)

### C.14.1 Setting Certificate Trust

Signers use their digital ID certificate to sign documents. In order for you to verify the validity of a signature, you must have explicitly trusted their certificate for signing or that certificate must chain up to a another certificate you have trusted (a trusted anchor). You can set trust ahead of time or when you are viewing a signature.

To trust a certificate for signing and certifying:

1. Do one of the following:
   - If you already have the certificate:
     2. Choose Certificates in the Display drop down list.
     3. Select the certificate.
     4. Choose Edit Trust.
   - If the certificate is in a signature:
     1. Right click and choose Signature Properties.
     2. Choose Show Certificate.
     3. Select the Trust tab.
4. Choose **Add to Trusted Identities**.

   **Tip:** If **Add to Trusted Identities** is disabled, the identity is already on your Trusted Identities list. To change the trust settings, you must use the first method above.

2. On the Trust tab, select the trust options. In enterprise settings, an administrator should tell you which trust settings to use.

   **Note:** During an import action, recipients of the distributed trust anchor may be able to inherit its trust settings. Once you’ve verified the sender, you usually want to accept these settings so you can use the certificate they way the sender intended.

---

**Figure 7 Certificate trust settings**

- **Use this certificate as a trusted root:** Makes the certificate a trust anchor. The net result is that any certificates which chain up to this one will also be trusted for signing. At least one certificate in the chain (and preferably only one) must be a trusted root (trust anchor) to validate signatures and timestamps.

  **Tip:** There is no need to make end entity certificates trust anchors if they issued by a certificate holder whose certificate you have configured as a trust anchor. It is best practice to trust the topmost certificate that is reasonable to trust because revocation checking occurs on every certificate in a chain until that anchor is reached. For example, in a large organization, it is likely you would want to trust your company’s certificate. If that certificate was issued by VeriSign, you would not want to make VeriSign a trusted root unless you wanted to trust every certificate that chains up to VeriSign.

- **Signed documents or data:** Trusts the certificate for approval signatures.
Tip: This setting is disabled because if the certificate is set as a trust anchor. Trust anchors are automatically trusted for approval signatures.

- **Certified documents**: Trusts the certificate for certification signatures.

- **Dynamic content**: Trusts multimedia and other dynamic content in certified documents. Selecting this option automatically adds documents that are certified with this certificate to the Trusted Documents list which is maintained by the Multimedia Trust Manager.

- **Embedded high privilege JavaScript**: Trusts embedded scripts. Certificate settings do not override application-level settings, so even if JavaScript is enabled for a particular certificate, it may not execute unless the application’s preferences allow it. This option requires that the application environment be configured correctly.

- **Privileged system operations (networking, printing, file access, etc.)**: Some operations represent a security risk more serious than others. Acrobat considers the following operations potential threats to a secure application operating environment: Internet connections, cross domain scripting, silent printing, external-object references, and FDF data injection. If this checkbox is checked, documents that are certified with this certificate will allow these actions.

Tip: This feature interacts with the Enhanced Security preferences which may be set by choosing Edit > Preferences > Security (Enhanced). The application always takes the least restrictive setting when determining what is allowed. For example, if the trust level for this certificate does not allow privileged operations but the certified file resided in a privileged location, then these operations will be permitted.

3. If you need to specify a policy restriction, do so. Most users only need to set policy restrictions at the request of their administrator.

4. Choose **OK** twice.

5. Choose **Close**.

### C.14.2 Setting Certificate Policy Restrictions

Policy restrictions are typically used in enterprise settings when configuring trust anchors. A restriction provides criteria the certificate chain must meet before a signing certificate can be used to create a valid signature. For example, a VeriSign certificate may be set as a trusted root, but a company may wish to only trust their own intermediate certificates (ICAs) that chain to VeriSign rather than all certificates that chain up to VeriSign. The company can issue an ICA with a certificate policy extension. By including that ICA in the certificate chain between all end entity certificates and VeriSign and requiring the presence of that extension, only company signers will be trusted.

Policies are represented by numbers called **object identifiers** (OIDs). OIDs are usually provided by your system administrator.

1. Select the Policy Restrictions tab if the Edit Certificate Trust dialog is displayed; otherwise, choose **Advanced** (Acrobat) or **Document** (Reader) > **Manage Trusted Identities**.

2. Choose **Certificates from the Display drop-down list**.

3. Highlight a certificate and choose **Edit Trust**.

4. Choose the Policy Restrictions tab and enter the restrictions:
Certificate Policies: Required. Enter the policy OID.
Description: Optional. Enter a meaningful description.

Figure 8 Policy restrictions

C.14.3 Using Certificates for Certificate Security (Encryption)

You only need to specify a certificate’s encryption usage if you are using certificate security. When more than one certificate is associated with the contact, you can select which one to use as the default encryption certificate. For details, see Chapter 16, “Certificate Security”.

C.15 Using Directory Servers to Add Trusted Identities

Businesses often use a centrally managed certificate repository such as an LDAP directory server. Directory servers are capable of returning X.509 public key certificates. These servers are searchable so that you can easily expand your list of trusted identities. Both Adobe Acrobat and Adobe Reader for Windows ship with default servers:

- Versions 7.x:
  - VeriSign Internet Directory Service
  - GeoTrust Directory Service
  - IDtree Directory Service
- Version 8.x and 9x:
  - VeriSign Internet Directory Service

Home users may never need to use directory servers. In most cases, needed certificates will be sent directly to you or will be embedded in a signature. However, enterprise users will likely use directory servers when their administrator has set up an LDAP server as part of a public key infrastructure. This allows the administrator to make the certificates available to teams and workgroups while managing them from a central location. The administrator usually preconfigures user machines, tells the user how to configure the server manually, or sends the server configuration details in a file as described in Chapter 3, “Migrating and Sharing Security Settings”.
C.15.1 Manually Configuring a Directory Server

Some companies store employee digital ID certificates on a networked LDAP server. To access those certificates, add the server to the list of directories used to locate those IDs.

**Tip:** In an ideal scenario, the server administrator supplies configuration details in a file as described in Chapter 3, “Migrating and Sharing Security Settings”.

To manually configure an identity directory:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Directory Servers in the left-hand list (Figure 9).
3. Choose New.
4. Configure the LDAP server settings in the Edit Directory Server dialog:
   - **Directory Name**: An arbitrary directory name.
   - **Access Type**: LDAP is the only type supported.
   - **Server Name**: The server name.
   - **Port**: The server port. 389 is the default port.
   - **Search Base**: A comma-separated list of name-value pairs used in the search. For example, \(c=us, cn=Brown.\) Trout, ou=example, \(dn=Acme\) Manufacturing for country, common name, organizational unit, and distinguished name.
   - **This server requires me to log on**: Check this box if the server requires username and password authentication to look up LDAP entries.
   - **User name**: The login username.
   - **Password**: The login password.
   - **Timeout**: The number of seconds to keep trying to connect.
   - **Maximum Number of Records to Receive**: The number of records to return.
5. Choose OK.
Figure 10  Digital ID Directory servers: Setting server details

C.15.2 Editing Directory Servers Details

Directory server details can be changed at any time.

To edit directory server information:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Directory Servers in the left-hand list (Figure 9).
3. Select a directory server from the right-hand panel.
4. Choose Edit.
5. Edit the information as described in “Manually Configuring a Directory Server” on page 342.
6. Choose OK.

C.15.3 Deleting a Directory Server

Previously configured directory servers can be removed from the server list at any time.

To delete a directory server:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Directory Servers in the left-hand list (Figure 9).
3. Select a directory server from the right-hand panel.
4. Choose Remove.
5. When a confirmation dialog appears, choose OK.
C.15.4 Specifying a Default Directory Server

A default server may be specified so that it is always used when searching for digital IDs.

To set default directory server:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Directory Servers in the left-hand list (Figure 9).
3. Select a directory server from the right-hand panel.
5. Choose OK if a confirmation dialog appears.

A star appears next to the name of the selected server.

Figure 11  Digital ID Directory servers: Setting defaults

C.15.5 Importing and Exporting Directory Server Settings

For details, refer to Chapter 3, “Migrating and Sharing Security Settings”.

C.16 Managing Contacts

Contacts are those people that will send you documents or receive documents from you. Each contact may be associated with one or more certificates. Like certificates, contacts can be added, removed, edited, and so on from the trusted identity list.

C.16.1 Viewing and Editing Contact Details

When a contact’s details change, it is possible to update them in the Trusted Identity Manager.

To change a contact’s details:

1. Choose Advanced (Acrobat) or Document (Reader) > Manage Trusted Identities.
2. Choose a contact in the left-hand list.
3. Choose **Details**.

![Figure 12 Contacts: Viewing details](image1)

**Figure 12 Contacts: Viewing details**

4. Edit the details.

5. Choose **OK**.

### C.16.2 Emailing Certificate or Contact Data

You can export certificate and contact data via email directly from the Trusted Identity Manager. Doing so allows other users to add that data their trusted identity list, thereby expanding the number of users that can participate in secure document workflows. For details, see “Emailing Your Certificate” on page 46.
C.16.3  Saving Certificate or Contact Details to a File

You can export certificate and contact data and save it to a file from the Trusted Identity Manager. Doing so allows you to email it later or locate it on a shared network directory. Other users can then add that data to their trusted identity list. For details, see “Saving Your Digital ID Certificate to a File” on page 47.

C.16.4  Associating a Certificate with a Contact

A certificate is usually already associated with a contact. However, in certain cases the two may need to be reassociated:

- Someone has provided you with new contact information.
- An old contact has sent you a certificate to be associated with old contact information.

To associate a certificate with a contact:

1. Choose Advanced (Acrobat) or Document (Reader) > Manage Trusted Identities.
2. Choose a contact in the left-hand list (Figure 12).
3. Choose Details.
4. Choose Associate Certificate (Figure 13).

Figure 14  Contacts: Selecting certificates

5. Select a certificate from the list.
6. Choose OK.
7. Choose OK.

C.16.5  Changing a Trusted Identity’s Certificate Association

Contacts in the Trusted Identity Manager only have value when they are associated with certificates. Therefore, removing a certificate association only makes sense when it is being replaced by another certificate. For example, someone in your trusted identities list may have replaced a compromised or expired certificate with a new one. In this case, simply replace the old certificate association with a new one.

1. Choose Advanced (Acrobat) or Document (Reader) > Manage Trusted Identities.
2. Choose a contact in the left-hand list (Figure 12).
3. Choose Details.

4. Choose a certificate from the list.

5. Choose Remove Association (Figure 15).

6. Choose a certificate from the list.

   **Note:** The certificate list is populated with the currently associated certificate and any unassociated certificates for the current contact. In other words, the list does not display all of a contact’s certificates, it displays only those that have no contact association.

7. Choose Associate Certificate.

8. Choose OK.

   **Figure 15 Edit Contact dialog**

---

**C.16.6 Deleting Contacts and Certificates**

It is possible to delete contact information independently from its certificate. The most common scenarios for deleting trusted identity information include the following:

- You no longer share documents with someone and can delete all of their contact and certificate data.
- The trusted identity’s contact information or certificate has changed and new data will be imported.

To delete a contact (and optionally a certificate):

1. Choose Advanced (Acrobat) or Document (Reader) > Manage Trusted Identities.

2. Choose Contacts from the Display drop-down list.

3. Choose a contact in the left-hand list (Figure 12).

4. Choose Delete.
5. Choose whether or not to delete the certificates along with contact. Once a certificate is deleted, it can no longer be used to validate someone’s signature or encrypt a document for them.

6. Choose **OK**.

![Figure 16 Contacts: Deleting](image)

**Deleting a Certificate**

To delete a certificate:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Manage Trusted Identities**.
2. Choose Certificates from the **Display** drop-down list.
3. Choose a certificate in the left-hand list (**Figure 13**).
4. Choose **Delete**.
5. Choose **OK**.
How tos: Using Digital IDs

The following sections provide details about managing digital IDs in Acrobat and Adobe Reader.

- “Working with Digital IDs” on page 349
- “Generic ID Operations” on page 351
- “Managing PKCS#12 Digital ID Files” on page 354
- “Managing Windows Digital IDs” on page 362
- “Managing Roaming ID Accounts and IDs” on page 362
- “Managing IDs Stored on Hardware Devices” on page 364

Tip: This feature requires a basic understanding of how Acrobat defines trust and “digital IDs.” For more information, see “Basic Concepts” on page 17.

D.17 Working with Digital IDs

D.17.1 Registering a Digital ID for Use in Acrobat

There are two ways to register a digital ID:

- In advance: You can set up the ID ahead of time for later use. To do so, choose Advanced (Acrobat) or Document (Reader) > Security Settings, selecting Digital IDs in the left-hand tree, and then choosing Add ID.
- On the fly: You can find or add IDs in signature and certificate security workflows. For example, when the Sign Document dialog appears, choose New ID from the Sign As drop down list.

For more information, refer to the following:

- Adding a Digital ID from a PKCS#12 File
- Finding a Digital ID in a Windows Certificate Store File
- Adding an ID that Resides on External Hardware
- Adding a Roaming ID Account to Get a Roaming ID

Figure 17 Add Digital ID dialog
.apf: No longer supported

Older application versions use a deprecated digital ID format with an .apf extension. .apf is not supported in 9.0. You must use Acrobat 8.x or earlier to use this type of ID.

D.17.2 Digital ID Management and the Security Settings Console

The Security Settings Console enables users to manage their own digital IDs. Choosing Advanced (Acrobat) or Document (Reader) > Security Settings opens a dialog for adding, removing, and setting the usage preferences for digital IDs stored on .pfx files, PKCS#11 modules and tokens, roaming ID servers, and the Windows Certificate Store.

**Tip:** You should always back up your private key if you have access to it. Without the key, encrypted documents cannot be decrypted and opened. To protect and back up private keys in an enterprise setting, administrators sometimes escrow private keys. If your digital ID is stored in a file on your local machine, consider copying it to a secure location.

**Figure 18 Security settings menu and manager**

D.17.3 Setting Identity Information

You can enter default identity (user) information that the application can automatically use as the defaults for workflows such as creating self-signed certificates and emailing certificate and server settings.

To create default user information:

1. Choose one of the following.
   - Acrobat (Windows): Edit > Preferences > Identity
   - Acrobat (Macintosh): Acrobat > Preferences > Identity
   - Adobe Reader (Windows): Edit > Preferences > Identity
   - Adobe Reader (Macintosh): Adobe Reader > Preferences > Identity

2. Configure the identity details. These details will appear in your signature appearance when you sign with a self-signed digital ID.

3. Choose OK.
D.18 Generic ID Operations

Once you have one or more digital IDs, you can edit, remove, and otherwise manage them from the Security Settings Console. To simplify workflows that use digital IDs, consider doing the following before using your ID:

- **Specifying Digital ID Usage**: Set an ID to automatically use each time one is required for signing or certificate encryption.
- **Sharing (Exporting) a Digital ID Certificate**: Since a digital ID’s certificate contains the public key required for validating your digital signature and encrypting documents for you, send it to those who participate in these kinds of workflows with you ahead of time.

Other operations also apply to all digital IDs irrespective of their format. For details, see:

- “Viewing All of Your Digital IDs” on page 352
- “Customizing a Digital ID Name” on page 353
- “Viewing Digital ID Certificates in the Certificate Viewer” on page 353

D.18.1 Specifying Digital ID Usage

If a digital ID is not specified for a particular task that requires one, a prompt will ask for a digital ID file. To avoid repeated prompts, specify a digital ID for signing and encryption. Different IDs may be used for signing and encryption.

When you specify ID usage, that ID is the first one in the list you’ll see when you’re asked to select an ID in a signing or encryption workflow. If you select a different ID, your usage option will change to the newly selected ID; that is, the last used ID becomes the new “default.”

To select a default digital ID file:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.

2. Select **Digital IDs** in the left-hand tree (Figure D.18.1).

3. Highlight an ID in the list on the right.
4. Choose **Usage Options**. A drop-down list appears.

![Figure 20 Usage options for a digital ID](image)

5. Choose one or more options: signing, certifying, and encrypting. A lock or pen icon (or both) will appear to the left of the digital ID based on this selection.

**Caution:** Invalid and expired IDs with a yellow caution triangle cannot be used.

### D.18.2 Sharing (Exporting) a Digital ID Certificate

Digital ID certificates must be distributed among participants in signing and certificate encryption workflows. Other users must have access to your certificate before:

- They can validate your signature if they are not already trusting a certificate above yours in the certificate chain. Note that a signature always includes the signer’s certificate, so validation can occur with the certificate embedded in the signature if it is not already on the validator’s machine.
- They can encrypt a document for you using certificate security.

Certificates can be emailed or saved to a file. For more information, see Chapter 3, “Migrating and Sharing Security Settings”.

**Tip:** To export a certificate displayed in the Certificate Viewer, choose **Export** on the Summary tab.

### D.18.3 Viewing All of Your Digital IDs

You can view all of your digital IDs in one list regardless of their type or location.

To view all of your IDs:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.

2. Select **Digital IDs** in the left-hand tree (Figure 18).

   All the IDs you have added appear in the right hand panel. The list includes all of the IDs that you can view separately under:

   - Digital ID Files
   - Roaming ID Accounts
   - Windows Digital IDs
   - PKCS#11 Modules and Tokens
D.18.4 Customizing a Digital ID Name

You can personalize a digital ID by providing a user-friendly name. This name appears in the ID drop-down list in workflows where you are asked to select an ID.

To provide a friendly name:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Digital IDs in the left-hand tree (Figure 18).
3. Highlight an ID in the list on the right.
5. Enter a name for the ID.

Figure 21  Personalizing an ID name

D.18.5 Viewing Digital ID Certificates in the Certificate Viewer

Your digital IDs appear in the Security Settings Console. From there, the Certificate Viewer can be used to display the time for which its certificate is valid and other details such as usage, a unique serial number, public key method, and so on (Figure 22).

To check certificate details:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Digital IDs in the left-hand tree (Figure D.18.1).
3. Highlight an ID in the list on the right.
4. Choose Certificate Details. The Certificate Viewer displays the certificate. (Figure 22). The following details are available:
   - Left hand panel: The certificate chain.
How tos: Using Digital IDs

Managing PKCS#12 Digital ID Files

- **Bottom area**: A description of the certificate, path validity statement, path validation time, and sometimes the type of validation.
- **Summary tab**: Displays the owner, issuer, validity period, and other details. The Intended Usage field tells you whether the certificate can be used for signing, encryption, or both. An Export button allow you to export the certificate to a file.
- **Details tab**: Lists all the certificate fields (extensions) and their values.
- **Revocation tab**: Indicates whether a revocation check occurred and the result. Allows users to initiate a manual check and analyze problems.
- **Trust tab**: Displays the certificate’s trust level. If it does not already exist in the trusted identities list, the Add to Trusted Identities is active. If the certificate is already on the Trusted Identities list and you want to change the trust level, see “Certificate Trust Settings” on page 337.
- **Policies tab**: Displays policy restriction information that must be met for a signature to be valid, if any.
- **Legal Notice tab**: Displays other certificate policies as well as a button which links to that policy, if any.

![Figure 22 Digital ID: Certificate viewer](image)

D.19 Managing PKCS#12 Digital ID Files

PKCS#12 digital ID files have several convenient features:

- Multiple IDs can be stored in a single, password-protected file.
- A file can contain both the public and private key.
- Passwords and password time-outs are user customizable.
D.19.1 Logging in to a Digital ID File

You will not usually need to log in to a digital ID file. Logging in means that Acrobat wants you to prove that you know the password to open the password-protected file containing the digital IDs. Since you likely supplied the password when you created your ID or obtained a new one, then you should be logged in.

However, you may need to log in for the following cases:
- You logged out of the file for some reason.
- You are importing an acrobatsecuritysettings file containing digital IDs.

To log in to a digital ID file:

D.19.2 Adding a Digital ID from a PKCS#12 File

If you need a digital ID does not appear in the digital ID list and you know it’s location, browse to it and add it. You can browse to PKCS#12 files (with .pfx or .p12 extensions) and Windows Certificate Store compatible files (with .cer and .der extensions).

**Note:** In enterprise settings, you may be instructed by your administrator to get a digital ID from a specific location or to customize Acrobat or Adobe Reader to work with software supplied by your organization.

To find a digital ID file:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Select Digital IDs in the left-hand tree (Figure 18).
3. Choose Add ID.
4. Select the My existing digital ID from and A file radio buttons (Figure 17).
5. Choose Next.
6. Choose Browse and browse to the digital ID file. PKCS#12 files may reside on a network or in some local location. For example,
7. Select the ID and choose **Open**.
8. Enter a password if one is required.
9. Review the digital ID list and choose **Finish**.

### D.19.3 Adding and Removing Digital ID Files from the File List

Adobe Acrobat and Adobe Reader only allow deletion of user-created self-signed digital IDs created with those applications. A file can have one or more IDs.

To delete or add an ID file:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Select **Digital IDs > Digital ID Files** in the left-hand tree (Figure 23).
3. Highlight a digital ID file in the right-hand panel.
4. Do one of the following:
   - Choose **Detach File**. The file is removed from the list but still remains on your file system.
   - Choose **Attach File**. Browse to the file, enter the file password, and choose **OK**.

   **Note:** Detaching a file does not remove it from your system, and it may be reattached later.

### D.19.4 Changing an ID File’s Password

Passwords and password time-outs are unique to PKCS#12 IDs. Since a file can contain multiple IDs, passwords and time-outs are configured at the file level rather than for individual IDs.

**Note:** If the file is read only, then the **Change Password** and **Password Timeout** options are disabled.

To change the password:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Highlight **Digital ID Files** in the left-hand tree (Figure 23).
3. Select a file in the right-hand panel (Figure 23).
4. Choose **Change Password**.
5. Enter the old password.
6. Enter a new password and confirm it.
7. Choose **OK**.

---

**Figure 24 Digital ID files: Password configuration**

---

D.19.5 Changing a PKCS#12 File’s Password Timeout

Passwords and password time-outs can only be set for PKCS#12 IDs. Since a file can contain multiple IDs, passwords and time-outs are configured at the file level rather than for individual IDs.

**Note:** If the is read only, then the **Change Password** and **Password Timeout** options are disabled.

To change the password timeout:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Highlight **Digital ID Files** in the left-hand tree (Figure 23).
3. Select a file in the right-hand panel (Figure 23).
4. Choose **Password Timeout**.

   **Tip:** The password timeout feature interacts with the Login/Logout feature as described in “Logging in to PKCS#12 Files” on page 358.

5. Configure the Password Timeout Policy dialog by specifying when a password prompt should appear:
   - **Always:** A password is always required each time the digital ID is used regardless of whether or not you are logged in to a file.
   - **After:** Choose a value from the drop-down list to set a time frame.
   - **Once per session:** A password is asked for only once while the application is open.
   - **Never:** The password is not usually required when using this ID and you are logged into the file.

6. Enter the password.
7. Choose **OK**.
D.19.6 Logging in to PKCS#12 Files

The digital ID Login feature provides access to the IDs in a particular file. Login behavior is dependant on the user-specified password timeout feature. If the user has specified a password timeout of *Never*, then the application never asks for a password when an ID is used for some process. For example:

- **Signing**: During signing workflows, you can sign with a digital ID without entering a password if you are logged into a file and the time-out is set to *Never*.
- **Batch processing**: In normal operation, batch sequences that require access to a digital ID invoke the user-interface’s authentication dialog. Because the dialog prompts for a password, the batch sequence is effectively stopped until a user intervenes. Logging in to a file provides the ID to the process without stopping it or requiring user input.

To enable sequences to run automatically and bypass normal user interface actions, do the following:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Select **Digital ID Files** in the left-hand tree (Figure 23).
   
   **Tip:** Verify the password timeout is set according to your own preferences. For details, see “Changing a PKCS#12 File’s Password Timeout” on page 357.
3. Select a file in the right-hand panel (Figure 23).
4. Do one of the following:
   - **Logout**: Highlight an ID in the list on the right and choose **Logout**.
   - **Login**: Highlight an ID in the list on the right and choose **Login**. Enter a password when prompted and choose **OK**.

D.19.7 Creating a Self-Signed Digital ID

**Note:** The option to create self-signed digital IDs is unavailable if your administrator has configured your application to prevent this operation.
Users can create a self-signed digital ID if they don’t wish to purchase an ID from a 3rd party certificate authority (CA) or are not given a company-provided ID. Self-signed IDs are usually considered less secure because the user has not been verified by a 3rd party CA. For self-signed IDs, you act as your own CA.

To create a self-signed digital ID:

1. Navigate to the Add Digital ID dialog as described in “Adding a Digital ID from a PKCS#12 File” on page 355.

2. Choose **A new digital ID I want to create now** (Figure 17).

3. Choose **Next**.

4. Select a digital ID format and storage location:
   - **New PKCS#12 Digital ID File**: Stores the IDs in a password protected file with a .pfx (Win) or .p12 (Mac) extension. The file is in a PKCS#12 standard format. The files can be copied, moved, and emailed. They are cross-platform, portable, and always password protected. This common format is supported by most security software applications, including web browsers. These files should always be backed up. On Windows XP, the default location is `C:\Documents and Settings\<username>\Application Data\Adobe\<application name>\<version>\Security\`.
   - **Windows Certificate Store**: (Windows only) Stores the ID in the Windows Certificate Store where it is also available to other Windows applications. The ID is protected by your Windows login. These IDs are easy to use and do not have to have file-level password protection. However, they are not portable and could be less secure if a file-level password is not specified.

5. Choose **Next**.
6. Configure the digital ID. The dialog is prepopulated if the Identity preferences have been previously configured:

   **Tip:** If you use non-Roman characters, choose **Enable Unicode Support** before continuing.

   - **Name:** The name that appears in the Signatures tab and in the signature field.
   - **Organizational Unit:** Optional. Appears in the signature and certificate.
   - **Organizational Name:** Optional. Appears in the signature and certificate.
   - **Email Address:** Optional. Appears in the signature and certificate.
   - **Country/Region:** Optional. Appears in the signature and certificate.
   - **Enable Unicode Support:** Optional. Use Unicode when your information cannot be adequately displayed with Roman characters.

   **Note:** Many applications do not support non-ASCII characters in certificates. Be sure to specify both an ASCII representation of the information as well as the Unicode representation of information you are supplying.

   - **Key Algorithm:** 2048-bit RSA offers more security than 1024-bit RSA, but 1024-bit RSA is more universally compatible. Use the 1024 bit key length if you are unsure.
   - **Use Digital ID for:** Select whether to use the digital ID for digital signatures, data encryption (certificate security), or both.

7. If a Windows digital ID was selected, choose **Finish**; otherwise, for a PKCS#12 ID do the following:

   1. Choose **Next**.
   2. Specify a file name and location for the digital ID file.
   3. Enter a password and confirm it.

   **Note:** Passwords are case-sensitive and must contain at least six characters.
D.19.8 Deleting a PKCS#12 Digital ID

Adobe Acrobat and Adobe Reader only allow deletion of user-created, self-signed digital IDs created by them. The methodology for deleting other types of IDs varies with the type of ID.

While the ID will be removed from the ID list, other ID’s in the container .pfx or p12 file will not be affected. Deleting the last, self-signed PKCS#12 ID in a .pfx or p12 file also deletes the digital ID file.

**Caution:** Because deleting an ID deletes its private key, operations that require that key will no longer be possible. If the file is used by other programs or you need it to open encrypted documents, do not delete it.

To delete a self-signed ID:

1. Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**.
2. Select **Digital IDs** in the left-hand tree (Figure D.18.1).
3. Highlight a self-signed ID in the list on the right that uses a digital ID file or Windows Certificate Store storage mechanism.
4. Choose **Remove ID**.
5. Choose **OK** when asked to proceed.

---

**Figure 28 Digital ID: PKCS#12 location and password**

**Figure 29 Digital ID: Deleting**
D.20 Managing Windows Digital IDs

For the Acrobat family of products, a “Windows digital ID” is an ID that resides in the Windows certificate store rather than the Acrobat store. Windows supports several formats listed in Table 2. These IDs are protected by your Windows login, are easy to use, and do require file-level password protection. However, they are not portable and are less secure when a file-level password is not specified.

The Windows store makes these IDs available to other Windows applications such as Acrobat and Adobe Reader. When an ID in the Windows store is registered with the application, it appears in the Security Settings Console. IDs in the Windows store are subject to the same operations as described in “Generic ID Operations” on page 351.

Figure 30 Windows digital ID menu

D.20.1 Finding a Digital ID in a Windows Certificate Store File

If you have a personal digital ID in the Windows store, it should appear in the Security Settings Console automatically without any special configuration. Acrobat products automatically find that ID. However, if there is a problem, you can browse to and add Windows Certificate Store compatible files (.cer and .p7b).

D.20.2 Deleting a Digital ID from the Windows Certificate Store

IDs that have been added to the Windows certificate store can only be deleted from the Security Settings Console if they are self-signed IDs created in Acrobat or Reader version 8.0 or later. Other IDs must be removed from the Windows store by using an application such as Internet Explorer. The store’s location in Internet Explorer may vary by version, but is typically found under Tools > Internet Options > Content tab > Certificates button.

D.21 Managing Roaming ID Accounts and IDs

A roaming ID is a digital ID that is stored on a server. The private key always remains on the server, but the certificate and its public key can be downloaded at the subscriber’s request to any location. Roaming IDs require an Internet connection.
Roaming IDs allow you to access and use your digital ID for signing or encryption from any machine that can access the server. You don't have to have your ID file with you or install it prior to use.

Roaming IDs can be centrally administered. When IDs expire, new ones can be issued and placed on the server rather than being distributed to each individual. Deployment and management therefore occurs in one location rather than on numerous client machines.

Depending on how the system is configured, users identify themselves (authenticate) to the server either with a username and password, Windows single sign-on, or by some 3rd party method such as ArcotID.

**Note:** Roaming IDs are only used for signing and cannot be used for certificate encryption. They are subject to the same operations as described in “Generic ID Operations” on page 351.

### D.21.1 Adding a Roaming ID Account to Get a Roaming ID

Roaming IDs are only available for those with roaming ID accounts on a roaming ID server. For connection details, contact your system administrator. Once you log in to your account, the IDs associated with that account will be automatically downloaded.

To install the roaming IDs certificate:

1. Verify you have an Internet connection.

   **Note:** If a roaming ID administrator has sent you an file with the account settings preconfigured, refer to Chapter 3, “Migrating and Sharing Security Settings”.

2. Do one of the following:
   - Navigate to the Add Digital ID dialog as described in “Adding a Digital ID from a PKCS#12 File” on page 355.
   - Choose **Advanced** (Acrobat) or **Document** (Reader) > **Security Settings**. Then expand the left-hand tree to **Roaming ID Accounts** and choose **Add Account** from the top menu (Figure 31).

3. Choose **Configure a roaming ID for use on this computer** (Figure 17).

4. Choose **Next**.

5. In the Add a Roaming ID dialog, enter an arbitrary server name. Choose a name that you can remember.

6. Enter the exact server URL.

7. Choose **Next**.

8. Enter your user name and password for this roaming ID server account.

9. Enter a server name and URL.

10. Choose **Next**.
Tip: Your server may require additional or different authentication steps. Follow the directions that appear in your workflow-specific dialogs.

11. Your certificate(s) will be automatically downloaded. Review the digital ID list and choose Finish.

D.21.2 Logging in to a Roaming ID Account

A roaming ID account is a user account on a roaming ID server containing one or more digital IDs. The login feature provides access to the IDs associated with the account. Depending on how the server administrator has set up the server, once you log in you may not be asked to supply a password again when you use that ID to sign.

To log in to a device:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Expand the left-hand tree to Roaming ID Accounts (Figure 31).
3. Select an account in the right-hand panel.
4. Choose Login.
5. Follow the instructions in the dialogs. The workflow varies by the roaming ID supplier as well as the authentication type.

Figure 31 Roaming ID Security Settings menu items

D.22 Managing IDs Stored on Hardware Devices

Smart cards, hardware tokens, and other devices are increasingly being used by businesses and individuals to carry digital IDs. These devices provide enhanced mobility, remote access to intranets and extranets, as well as strong security with public/private key cryptography and PIN access to the digital ID.

Note: Most devices comply with the Public Key Cryptography System 11 (PKCS#11) format devised by RSA.
The method for registering a digital ID on such a device with the application may vary. The manufacturer or your system administrator should provide detailed instructions. However, the steps below may be used as a general guide. IDs stored on a PKCS#11 device are subject to the same operations as described in “Generic ID Operations” on page 351.

D.22.1 Adding an ID that Resides on External Hardware

Digital IDs can reside on hardware such as a smart card or token with a USB interface. In these cases, the card is inserted into a smart card reader or the token is inserted directly into an USB port. Adobe products can be configured to look for and use IDs on these devices by adding the device’s module (software driver) to the module list. The module’s IDs are automatically registered with the application.

To register an ID that resides on external hardware:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.
2. Expand Digital IDs in the left-hand list (Figure 30).
3. Highlight PKCS#11 Modules and Tokens.

Figure 32 PKCS#11 Security Settings menu items

5. Browse to the device driver. On Windows, this could likely be C:\Windows\system32\<some dll>.dll. The exact path will be supplied by your system administrator or the maker of your device.
6. Choose Open.

The module and its IDs are automatically added to the list in the right-hand panel.

D.22.2 Changing Passwords

A card or token may contain multiple IDs. All of the IDs are password protected by a single password. This password is used to log in to a device and to sign.

1. Expand the tree under PKCS#11 Modules and Tokens.
2. Highlight any module.
3. A card or token label should appear in the right-hand panel. If there is more than one, select one.

4. Choose Change Password.

5. Enter the old password.

6. Enter a new password and confirm it.

7. Choose OK.

D.22.3 Logging in to a Device

Logging in provides access to the IDs on a particular device or smart card. In most cases login in is not required as it occurs on demand during signing or encryption/decryption.

PKCS#11 workflows vary by the device supplier. For example, additional passwords or PINs may or may not be required. The login interface may be provided by the Adobe application or by the device supplier.

To log in to a device:

1. Choose Advanced (Acrobat) or Document (Reader) > Security Settings.

2. Expand the tree under PKCS#11 Modules and Tokens.

3. Highlight any module.
4. A card or token label should appear in the right-hand panel. If there is more than one, select one.

5. Choose Login.

6. Enter a password.

7. Choose OK.

**Figure 35 PKCS#11 Security Settings menu items**
Acrobat’s features adhere to widely accepted standards as listed in Table 5.

Table 5 Standards support

<table>
<thead>
<tr>
<th>Reference</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIST PKITS “Public Key Interoperability Public Key Interoperability Test Suite Certification Path Validation”, Draft Version 0.7 June 30, 2003</td>
<td>Chain building and path validation, including cross certificates and multiple chains.</td>
</tr>
<tr>
<td>OIDS. ASN.1</td>
<td>Object Identifiers (OIDs)</td>
</tr>
</tbody>
</table>
Table 6 provides a comprehensive list of security terms and acronyms used in the fields of digital signatures and document security.

<table>
<thead>
<tr>
<th><strong>Table 6 Security Terms</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.apf</td>
<td>See Adobe Profile Files.</td>
</tr>
<tr>
<td>.cer</td>
<td>A Microsoft format for digital IDs often stored in the Windows Certificate Store. These IDs can be used by Windows programs as well as the Acrobat product family.</td>
</tr>
<tr>
<td>.p12</td>
<td>See PKCS#12.</td>
</tr>
<tr>
<td>.p7b</td>
<td>See PKCS#7.</td>
</tr>
<tr>
<td>.p7c</td>
<td>See PKCS#7.</td>
</tr>
<tr>
<td>.pfx</td>
<td>See PKCS#12.</td>
</tr>
<tr>
<td>AATL</td>
<td>See Adobe Approved Trust List</td>
</tr>
<tr>
<td>AC</td>
<td>See attribute certificate.</td>
</tr>
<tr>
<td>Acrobat’s Public Key Infrastructure Library</td>
<td>A standalone PKI toolkit written in C++ with the intention of being completely portable and usable in different applications, including but not limited to, Acrobat and GUI-less servers. ASPKI supports RFC 3280 and NIST compliant chain building and path validation, including support for cross certificates and multiple chains; multiple revocation protocols like CRL (RFC3280) and OCSP (RFC2560); time stamping (RFC3161); and embedded revocation information along with a signature to achieve signature archival.</td>
</tr>
<tr>
<td>AdES</td>
<td>See advanced electronic signatures</td>
</tr>
<tr>
<td>Adobe Approved Trust List</td>
<td>An Adobe program designed to facilitate trust in PDF signatures by downloading a list of trusted, high assurance root and ICA certificates to Acrobat and Reader v9.0 and above.</td>
</tr>
<tr>
<td>Adobe LiveCycle Rights Management Server</td>
<td>Adobe LiveCycle Enterprise Suite (ES) is a SOA J2EE-based (Java 2 Enterprise Edition) server software product from Adobe Systems Incorporated used to build applications that automate a broad range of business processes for enterprises and government agencies.</td>
</tr>
<tr>
<td>Adobe Policy Server</td>
<td>As of Acrobat 9, Adobe Policy Server is renamed to Adobe LiveCycle Rights Management Server.</td>
</tr>
<tr>
<td>Adobe Profile Files</td>
<td>Adobe’s legacy certificate format not used after Acrobat 5. The certificates are stored in .apf files. This format is not supported as of version 9.0.</td>
</tr>
<tr>
<td>advanced electronic signatures</td>
<td>A type of electronic signature described in the European Union Signature Directive. Differentiated from a Qualified Electronic Signature in that it may not use a QEC.</td>
</tr>
<tr>
<td>AIA</td>
<td>See authority information access.</td>
</tr>
<tr>
<td>AIIM</td>
<td>See Association for Information and Image Management.</td>
</tr>
<tr>
<td>ALCRMS</td>
<td>See Adobe LiveCycle Rights Management Server.</td>
</tr>
<tr>
<td>approval signature</td>
<td>A signature used to indicate approval of, or consent on, the document terms. Acrobat and Reader recognize both approval and certification signatures.</td>
</tr>
<tr>
<td>APS</td>
<td>See Adobe Policy Server.</td>
</tr>
</tbody>
</table>
# Glossary of Security Feature Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASPKI</td>
<td>See Acrobat’s Public Key Infrastructure Library.</td>
</tr>
<tr>
<td>Association for Information and Image Management</td>
<td>AILM is a non profit community that provides education, research, and best practices to help organizations find, control, and optimize their information as well as understand the challenges associated with managing documents and business processes.</td>
</tr>
<tr>
<td>attribute certificate</td>
<td>A file that contains the supplemental attributes and extension that is bound to a PKC, but does not itself contain any key data.</td>
</tr>
<tr>
<td>authority information access</td>
<td>An extension that is part of a PKC which contains information on how to access either PKC’s of issuing certificate(s) (least used) or where to access OCSP revocation information which is what this extension is primarily used for.</td>
</tr>
<tr>
<td>basic constraint</td>
<td>An extension within a public key certificate that defines whether or not the certificate has been issued to a CA.</td>
</tr>
<tr>
<td>CA</td>
<td>See certificate authority (CA).</td>
</tr>
<tr>
<td>CdES</td>
<td>See Cryptographic Message System Advanced Electronic Signatures.</td>
</tr>
<tr>
<td>CAPI</td>
<td>See MSCAPI.</td>
</tr>
<tr>
<td>CDS</td>
<td>See Certified Document Services (CDS).</td>
</tr>
<tr>
<td>CDS digital ID</td>
<td>A digital ID issued by a Certified Document Services provider.</td>
</tr>
<tr>
<td>CDS digital ID certificate</td>
<td>See CDS digital ID.</td>
</tr>
<tr>
<td>CEN</td>
<td>European Committee for Standardization.</td>
</tr>
<tr>
<td>certificate</td>
<td>That part of a digital ID that contains the public key. Certificates are shared among participants of signature and certificate security workflows in order to verify participant identities.</td>
</tr>
<tr>
<td>certificate authority (CA)</td>
<td>An entity which issues digital certificates for use by other parties. It is an example of a trusted third party. CAs are characteristic of many PKI schemes.</td>
</tr>
<tr>
<td>certificate revocation list (CRL)</td>
<td>CRL is a method that public key infrastructures use to maintain access to cached or networked lists of unexpired but revoked certificates. The list specifies revoked certificates, the reasons for revocation (optional), and the certificate issue date and issuing entities. Each list contains a proposed date for the next release. Acrobat’s CRL revocation checker adheres to RFC 3280 and NIST PKITS except for delta CRLs.</td>
</tr>
<tr>
<td>certification signature</td>
<td>A digital signature applied using an individual digital ID or organizational digital ID for the purpose of establishing the authenticity of a document and the integrity of a document’s content, including its appearance and business logic.</td>
</tr>
<tr>
<td>certified document</td>
<td>A document to which a certification signature has been applied.</td>
</tr>
<tr>
<td>Certified Document Services (CDS)</td>
<td>An Adobe program where commercial CA’s create a subordinate or ICA below that chains to the Adobe Root certificate. As the Adobe Root is automatically trusted by Acrobat and Reader v6.0 and above, signatures made with credentials that chain to it are also similarly trusted.</td>
</tr>
<tr>
<td>certify or certifying</td>
<td>The act of applying a certification signature to a document using the Acrobat “Certify” feature. Certifying helps establish document authenticity as well as the integrity of its content, including its appearance and business logic.</td>
</tr>
<tr>
<td>Click thru signature</td>
<td>A type of electronic signature where the signer is indicating their agreement with terms and indication to sign the document / process by clicking on a button, which might say “I accept.” Typically, an audit log of the event is kept for evidentiary purposes and authentication may or may not be required. Sometimes a server applied digital (certification) signature may be applied after the click thru process to protect the integrity of the document.</td>
</tr>
<tr>
<td>CMS</td>
<td>See Cryptographic Message Syntax.</td>
</tr>
</tbody>
</table>
### Table 6 Security Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRL</td>
<td>See certificate revocation list (CRL).</td>
</tr>
<tr>
<td>Cryptographic Message Syntax</td>
<td>A syntax is used to digitally sign, digest, authenticate, or encrypt arbitrary message content which is described in RFC 3369.</td>
</tr>
<tr>
<td>Cryptographic Message System</td>
<td>An EU Advanced Signature Format relying on CMS signatures, as described in ETSI TS 101 733.</td>
</tr>
<tr>
<td>Advanced Electronic Signatures</td>
<td>Application software that allows it to use MSCAPI to communicate with cryptographic module APIs such as PKCS#11 modules, PFX files, and so on.</td>
</tr>
<tr>
<td>CSP</td>
<td>See cryptographic service provider.</td>
</tr>
<tr>
<td>CSP</td>
<td>Certificate Service Provider. Alternate term for CA.</td>
</tr>
<tr>
<td>digest method</td>
<td>A hash algorithm used to create a one way hash of data. Acrobat supports six digest methods; MD5, SHA1, SHA256, SHA384, SHA 512, and RIPEMD 160.</td>
</tr>
<tr>
<td>digital ID</td>
<td>An electronic representation of data based on the ITU-T X.509 v3 standard, associated with a person or entity. It is often stored in a password-protected file on a computer or network, a USB token, a smart card, or other security hardware device. It can be used for digital signatures and certificate security. &quot;Digital ID&quot; is sometimes used interchangeably with &quot;certificate&quot;; however, a certificate is only one part of a digital ID which also contains a private key and other data.</td>
</tr>
<tr>
<td>digital signature</td>
<td>An electronic signature that can be used to verify the identity of the signer through the use of public key infrastructure (PKI) technology. Signers need a digital ID and an application capable of creating a signature.</td>
</tr>
<tr>
<td>digital signature algorithm</td>
<td>An encryption algorithm used to create a digital signature created by NIST as defined by FIPS 186.</td>
</tr>
<tr>
<td>digitally sign</td>
<td>To apply a digital signature using a digital ID.</td>
</tr>
<tr>
<td>document integrity</td>
<td>In signing workflows, document integrity refers to whether or not what was signed has changed after signing. That is, what the signer signed should be reproducible and viewable on the document recipient’s end. For the document recipient to validate a signature, it's important to determine to what document or what document version that signature applies. See message digest.</td>
</tr>
<tr>
<td>DSA</td>
<td>See digital signature algorithm.</td>
</tr>
<tr>
<td>EC</td>
<td>European Commission.</td>
</tr>
<tr>
<td>EE</td>
<td>See end entity certificate (EE).</td>
</tr>
<tr>
<td>EESSI</td>
<td>Electronic Exchange of Social Security Information.</td>
</tr>
<tr>
<td>eID</td>
<td>See electronic ID.</td>
</tr>
<tr>
<td>electronic ID</td>
<td>A broad term for any electronic ID. In the EU, it is commonly used to describe national-level identity cards.</td>
</tr>
<tr>
<td>electronic signature</td>
<td>Generic term. Generally defined as an electronic process which intrinsically links some tag (data, voice, image, key) to content that is being signed, is linked to the signer, and is generally capable of detecting changes in the document signed. A digital signature is a type of electronic signature, as are click thru and signature image.</td>
</tr>
<tr>
<td>embedded JavaScript</td>
<td>JavaScript that exists within a document rather than that which is executed from the JavaScript Console or through a batch process.</td>
</tr>
</tbody>
</table>
### Embedded Validation Response

Information from the digital ID issuer that was used to apply the digital signature and that indicates if the digital ID was valid when the signature was applied. If the digital ID was valid and no one has tampered with the document, the signature will have a status of VALID.

Once the digital ID expires or is revoked, it won’t be possible to determine if the signature was valid at the time it was applied unless there is an embedded revocation response.

### End Entity Certificate (EE)

The last element of a signing chain. By definition, an end entity certificate does not contain the basic constraint value CA, and is issued to an individual or a pseudo entity (e.g., a department or organization).

### ETSI

European Telecommunications Standards Institute.

### ETSI/ESI

ETSI/Electronic Signature & Infrastructure Technical Committee.

### EU

European Union.

### EU Experts Group

A generic term used widely within the EU. Of particular interest is the EUEG on Electronic Procedures.

### EU Signature Directive


### FIPS

Federal Information Processing Standards: These are publicly announced standards developed by the United States Federal government for use by all non-military government agencies and by government contractors. Many FIPS standards are modified versions of standards used in the wider community (ANSI, IEEE, IOS, etc.).

### FIPS 140

Federal Information Processing Standard 140: Standard which defines increasing levels of assurance for hardware and software based devices and applications for storing private keys. Level 1 is the lowest assurance, and level 4 is the highest, requiring devices to self-destruct and ‘zeroize’ themselves if they are compromised in any way.

### Hardware Security Module

While actually a generic term for any hardware device designed to securely store digital IDs, HSMs in common parlance are rack mounted servers or hardened PCI cards which are designed for higher security and higher volume cryptographic operations.

### Hardware Token

A hardware device (typically a smart card or USB device) that contains the user’s digital ID(s) and requires a password or other authentication method to access those IDs for the purpose of signing.

### HSM

See hardware security module.

### ICA

See intermediate certificate authority (ICA).

### IDABC

Interoperable Delivery of European eGovernment Services to public Administrations Businesses and Citizens individual digital ID: A digital ID issued to an individual to digitally sign as them self (e.g. John Smith) as opposed to an organization or other non-human entity.

### Individual Digital ID

A digital ID issued to individuals so that they can identify themselves during a digital signature process (e.g. John Smith) as opposed to an organization or other non-human entity.

### Intermediate Certificate Authority (ICA)

A type of CA characterized by the fact that the ICA’s certificate may itself be signed by a different ICA, all the way up to a ‘self-signed’ root certificate. Certificates in between the end entity and root certificates are sometimes called “intermediate certificates” (ICAs) and are issued by the CA or ICAs underneath the CA.

### ISO

International Standards Organization.

### ITC

Information Communication Technologies.
<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>long term validation</td>
<td>The validation of a digital signature after certificate(s) associated with the signature has expired.</td>
</tr>
<tr>
<td>LTV</td>
<td>See long term validation.</td>
</tr>
<tr>
<td>message digest</td>
<td>Before Acrobat or Adobe Reader can verify if a document the signed version of the document has changed or not (has integrity), it must first have a way to uniquely identify what was signed. To do this, it uses a message digest. A message digest is a number which is created algorithmically from a file and which uniquely represents that file. If the file changes, the message digest changes. Sometimes referred to as a checksum or hash, a message digest is simply a unique number created at signing time that identifies what was signed and is then embedded in the signature and the document for later verification.</td>
</tr>
<tr>
<td>MS</td>
<td>Member State (one of the EU countries)</td>
</tr>
<tr>
<td>MSCAPI</td>
<td>Windows Microsoft Crypto API (MSCAPI) is the API that the application uses to access cryptographic service providers such as PFX files and PKCS#11 files. MSCAPI is also used by the application anytime it uses a Windows security feature.</td>
</tr>
<tr>
<td>National Institute of Standards</td>
<td>A non regulatory agency of the United States Department of Commerce’s Technology Administration. The institute’s mission is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve quality of life.</td>
</tr>
<tr>
<td>NIST</td>
<td>See National Institute of Standards.</td>
</tr>
<tr>
<td>OCSP</td>
<td>See online certificate status Protocol (OCSP).</td>
</tr>
<tr>
<td>online certificate status Protocol (OCSP)</td>
<td>OCSP defines a protocol for determining the revocation status of a digital certificate without requiring a CRL. Unlike CRL, OCSP obviates the need to frequently download updates to keep certification status lists current. Acrobat’s OCSP revocation checker adheres to RFC 2560.</td>
</tr>
<tr>
<td>organizational digital ID</td>
<td>A digital ID issued to an organization or non-human entity (for example, the Adobe Public Relations Department). It can be used by an authorized employee / process to perform signing operations, at the desktop or server, on behalf of the company.</td>
</tr>
<tr>
<td>PAdES</td>
<td>See PDF Advanced Electronic Signatures.</td>
</tr>
<tr>
<td>PDF Advanced Electronic Signatures</td>
<td>A five part standard (ETSI TS 102 778) which describes how to use the digital signature features of the Portable Document Format (PDF) to meet EU signature requirements.</td>
</tr>
<tr>
<td>PKC</td>
<td>See public key certificate.</td>
</tr>
<tr>
<td>PKCS</td>
<td>A group of Public Key Cryptography Standards authored by RSA Security</td>
</tr>
<tr>
<td>PKCS#11</td>
<td>Public key Cryptography Standard #11: A Public key cryptography Standard published by RSA Laboratories defining an API, called Cryptoki, to devices which hold cryptographic information and perform cryptographic functions.</td>
</tr>
<tr>
<td>PKCS#11 device</td>
<td>External hardware such as a smart card reader or token. It is driven by a module (a software driver such as a .dll file on Windows).</td>
</tr>
<tr>
<td>PKCS#11 digital ID</td>
<td>An ID on a PKCS11# device. A device may contain one or more IDs.</td>
</tr>
<tr>
<td>PKCS#11 format</td>
<td>Cryptographic Token Interface Standard: An encryption format used by smart cards, tokens, and other PKCS#11-compatible devices. The ID is stored on the device rather than on the user’s computer.</td>
</tr>
<tr>
<td>PKCS#11 module</td>
<td>The software module that drives a PKCS#11 device.</td>
</tr>
<tr>
<td>PKCS#11 token</td>
<td>See PKCS#11 device.</td>
</tr>
</tbody>
</table>
## Glossary of Security Feature Terms

<table>
<thead>
<tr>
<th>Table 6 Security Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PKCS#12</strong></td>
</tr>
<tr>
<td><strong>PKCS#7</strong></td>
</tr>
<tr>
<td><strong>PKCS#9</strong></td>
</tr>
<tr>
<td><strong>PKI</strong></td>
</tr>
<tr>
<td><strong>point of single contact</strong></td>
</tr>
<tr>
<td><strong>Policy Server</strong></td>
</tr>
<tr>
<td><strong>Private key</strong></td>
</tr>
<tr>
<td><strong>privileged context</strong></td>
</tr>
<tr>
<td><strong>PSC</strong></td>
</tr>
<tr>
<td><strong>public key</strong></td>
</tr>
<tr>
<td><strong>public key certificate</strong></td>
</tr>
<tr>
<td><strong>public key infrastructure</strong></td>
</tr>
<tr>
<td><strong>QC</strong></td>
</tr>
<tr>
<td><strong>QEC</strong></td>
</tr>
<tr>
<td><strong>QES</strong></td>
</tr>
<tr>
<td><strong>QSCP</strong></td>
</tr>
<tr>
<td><strong>qualified certificate service provider</strong></td>
</tr>
<tr>
<td><strong>qualified electronic certificate</strong></td>
</tr>
<tr>
<td>Glossary of Security Feature Terms</td>
</tr>
<tr>
<td>------------------------------------</td>
</tr>
<tr>
<td><strong>Table 6 Security Terms</strong></td>
</tr>
<tr>
<td>qualified electronic signatures</td>
</tr>
<tr>
<td>roaming ID</td>
</tr>
<tr>
<td>root certificate</td>
</tr>
<tr>
<td>RSA</td>
</tr>
<tr>
<td>Secure Identity Across Borders Linked (STORK)</td>
</tr>
<tr>
<td>secure signature creation device</td>
</tr>
<tr>
<td>security restricted property or method</td>
</tr>
<tr>
<td>signature algorithm</td>
</tr>
<tr>
<td>signature image</td>
</tr>
<tr>
<td>SSCD</td>
</tr>
<tr>
<td>STF 364</td>
</tr>
<tr>
<td>STORK</td>
</tr>
<tr>
<td>SubCA</td>
</tr>
<tr>
<td>subordinate certificate authority</td>
</tr>
<tr>
<td>timestamp</td>
</tr>
<tr>
<td>TL</td>
</tr>
<tr>
<td>Trust (Trust service) Status List</td>
</tr>
<tr>
<td>trust anchor</td>
</tr>
<tr>
<td>trust list</td>
</tr>
</tbody>
</table>
### Glossary of Security Feature Terms

#### Table 6 Security Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS</td>
<td>See timestamp.</td>
</tr>
<tr>
<td>TSL</td>
<td>See Trust (Trust service) Status List.</td>
</tr>
<tr>
<td>TSP</td>
<td>A timestamp protocol which is described in RFC 3161.</td>
</tr>
<tr>
<td>X.509v3</td>
<td>In cryptography, X.509 is an ITU-T standard for a public key infrastructure (PKI) for single sign-on (SSO) and Privilege Management Infrastructure (PMI). X.509 specifies, amongst other things, standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm.</td>
</tr>
<tr>
<td>XAdES</td>
<td>See XML Advanced Electronic Signatures.</td>
</tr>
<tr>
<td>XML Advanced Electronic Signatures</td>
<td>An EU Advanced Electronic Signature Format relying on XML signatures, as described in ETSI TS 101 903.</td>
</tr>
</tbody>
</table>
Index

Index

.a 369
   .apf 369
   No longer supported 350
   .cer 369
   .p12 369
   .p7b 369
   .p7c 369
   .pfx 369

1 261
1000 261
1001 261
1002 262
1003 262
1004 262
1005 262
1006 262
1007 262
1008 262
1009 262

2 262
2004 262
2006 262
2007 262
2009 262
2010 262
2011 262
2012 262
2013 263
2014 263

3 263
3000 263
3001 263
3002 263
3003 263

4 263
4000 263
4001 263
4002 263

6 76
64 bit Windows 76

A 92
A Custom Certificate Preference Example for Identrus Compliance
A note about migration and upgrading 25

aAddressBook 95
AATL 369
AC 369, 370
AcceptablePolicyOIDsc 111
Access Type 68, 342
Accessing the Windows Certificate Store 291
Acrobat 71
Acrobat 3.0 and later 285, 287
Acrobat 5.0 and later 285, 287
Acrobat 6.0 and later 285, 287
Acrobat 7.0 and later 285, 287
Acrobat 9.0 and later 285, 287
Acrobat version, encryption algorithm, & security handler correlation 329
Actions that can be associated with a signature field 190
adbe.pkcs7.detached 112, 203
adbe.pkcs7.sha1 113, 204
adbe.x509.rsa_sha1 112, 203
Add Digital ID dialog 349
Adding a Certificate From a Signature 333
Adding a contact 294
Adding a Digital ID from a PKCS#12 File 355
Adding a Roaming ID Account to Get a Roaming ID 363
Adding an ID that Resides on External Hardware 365
Adding an OCSP URL in the binary editor 162
Adding and Removing Digital ID Files from the File List 356
Adding Certificate Preferences 91
Adding Custom Signing Reasons 200
Adding or Removing Group Contacts 293
Adding recipients to a document with certificate security 297, 302
Adding Someone to Your Trusted Identity List 333
Additional Revocation Checking Constraints 129
aDefDirectory 86
AdES 369
Adhoc 73
aDirectory 95
Adobe LiveCycle Rights Management Server security 266
Adobe OIDs 71
Adobe Profile Files 369
Adobe Reader usage rights and feature enablement 322
Adobe-Arc 71
Advanced Electronic Signatures 369
Advanced Installer Tuning for Power Users 32
aHandler 102
AIA 369
AllIM 369
ALCRMS 369
ALCRMS Server Configuration 308
Algorithms Used During Digital Signature Creation 329
All contents 268
All contents except metadata 268
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allowing Self Sign Digital IDs 80</td>
</tr>
<tr>
<td>Allowing Signing Reason 174</td>
</tr>
<tr>
<td>AllowOtherInfoWhenSigning i 120</td>
</tr>
<tr>
<td>Alternate certificate URL seed value 211</td>
</tr>
<tr>
<td>Always use this digital ID 296, 301</td>
</tr>
<tr>
<td>APF files 327</td>
</tr>
<tr>
<td>appearanceFilter 195</td>
</tr>
<tr>
<td>Application Defaults 25</td>
</tr>
<tr>
<td>Application Environment Preference Improvements 325</td>
</tr>
<tr>
<td>Application Preference Basics 75</td>
</tr>
<tr>
<td>Applied permission handlers as a result of user actions 136</td>
</tr>
<tr>
<td>Applying a Certificate Security Policy 303</td>
</tr>
<tr>
<td>Applying a Certificate Security to a Group 304</td>
</tr>
<tr>
<td>Applying a Security Policy to a Document 277</td>
</tr>
<tr>
<td>Applying ALCRMS Security 309</td>
</tr>
<tr>
<td>Approval Signature 138, 140, 216</td>
</tr>
<tr>
<td>approval signature 369</td>
</tr>
<tr>
<td>aPrivKey 94</td>
</tr>
<tr>
<td>ArchiveRevInfo 74</td>
</tr>
<tr>
<td>ArchiveRevInfo extension format 74</td>
</tr>
<tr>
<td>Arranging Signature Fields 185</td>
</tr>
<tr>
<td>aSignFormat 113</td>
</tr>
<tr>
<td>aSignHash 112</td>
</tr>
<tr>
<td>Ask me which digital ID to use next time 296, 301</td>
</tr>
<tr>
<td>ASPKI 369, 370</td>
</tr>
<tr>
<td>ASPKI Service Provider Interfaces 95</td>
</tr>
<tr>
<td>Associating a Certificate Preference with a Chain Scope 91</td>
</tr>
<tr>
<td>Associating a Certificate with a Contact 346</td>
</tr>
<tr>
<td>Associating Batch Processing with a Security Method 271</td>
</tr>
<tr>
<td>Association for Information and Image Management 370</td>
</tr>
<tr>
<td>Attachments 322</td>
</tr>
<tr>
<td>Audit alert for ALCRMS security 309</td>
</tr>
<tr>
<td>Authentication Mechanisms 97</td>
</tr>
<tr>
<td>AuthenticDocuments 72</td>
</tr>
<tr>
<td>AuthenticDocumentsTrust 72</td>
</tr>
<tr>
<td>Authenticity Verification 232</td>
</tr>
<tr>
<td>Authoring a Document with Multiple Fields 186</td>
</tr>
<tr>
<td>Authoring Form Fields 219</td>
</tr>
<tr>
<td>Authoring Signable Forms 186</td>
</tr>
<tr>
<td>Authority Information Access 370</td>
</tr>
<tr>
<td>Automating signing tasks 212</td>
</tr>
<tr>
<td>aVerify 94</td>
</tr>
<tr>
<td>aVerifyUseAlways 94</td>
</tr>
<tr>
<td>Basic Installer Tuning 29</td>
</tr>
<tr>
<td>Basic Revocation Checking 157</td>
</tr>
<tr>
<td>bAskBeforeInstalling 79</td>
</tr>
<tr>
<td>bAuthRequired 116, 117</td>
</tr>
<tr>
<td>bAutoLaunchAtDocClose 130</td>
</tr>
<tr>
<td>bAutoLaunchAtSendMail 130</td>
</tr>
<tr>
<td>bCertStoreImportEnable 88</td>
</tr>
<tr>
<td>bCustomPrefsCreated 86</td>
</tr>
<tr>
<td>Because You Care About Information Assurance 13</td>
</tr>
<tr>
<td>Because You’re Lost? 14</td>
</tr>
<tr>
<td>Because Your Technical 14</td>
</tr>
<tr>
<td>Before You Sign . . . 215</td>
</tr>
<tr>
<td>bEnforceSecureChannel 115</td>
</tr>
<tr>
<td>bEnNonEmbFontLegPDFWarn 123</td>
</tr>
<tr>
<td>Best Practices for Signed Documents that will Change 170</td>
</tr>
<tr>
<td>bFDFExportSave 98</td>
</tr>
<tr>
<td>bFDFExportSign 98</td>
</tr>
<tr>
<td>bFDFRequestExcludeCert 98</td>
</tr>
<tr>
<td>bFDFRequestSave 98</td>
</tr>
<tr>
<td>bFIPSMode 101</td>
</tr>
<tr>
<td>bFollowURIsFromAIA 110</td>
</tr>
<tr>
<td>bGoOnline 107, 109</td>
</tr>
<tr>
<td>blIgnoreValidityDates 107, 110</td>
</tr>
<tr>
<td>blsEnabled 114</td>
</tr>
<tr>
<td>blsFavorite 102</td>
</tr>
<tr>
<td>bLoadSettingsFromURL 79, 89</td>
</tr>
<tr>
<td>bPreviewModeBeforeSigning 119</td>
</tr>
<tr>
<td>bPrivKey 76</td>
</tr>
<tr>
<td>bReasons 76</td>
</tr>
<tr>
<td>bReqRevCheck 76</td>
</tr>
<tr>
<td>bReqSigPropRetrieval 115</td>
</tr>
<tr>
<td>bReqSigPropVerification 129</td>
</tr>
<tr>
<td>bRequireAKI 110</td>
</tr>
<tr>
<td>bRequireOCSPCertHash 108</td>
</tr>
<tr>
<td>bRequireValidSigForChaining 110</td>
</tr>
<tr>
<td>bReturnRevInfoToUser 128</td>
</tr>
<tr>
<td>bSavePassword 103</td>
</tr>
<tr>
<td>bSelfSignCertGen 80</td>
</tr>
<tr>
<td>bSendNonce 107</td>
</tr>
<tr>
<td>bShowAllChains 99</td>
</tr>
<tr>
<td>bShowWarningForChanges 127</td>
</tr>
<tr>
<td>bShowWarningForChanges and 126</td>
</tr>
<tr>
<td>bSigAPStatusIconDisable 126</td>
</tr>
<tr>
<td>bSignRequest 108</td>
</tr>
<tr>
<td>bSuppressStatusDialog 77</td>
</tr>
<tr>
<td>bTrueTypeFontPDFSigQWarn 123</td>
</tr>
<tr>
<td>bUseExpiredTimestamps 125</td>
</tr>
<tr>
<td>bValidateOnOpen 76, 124</td>
</tr>
<tr>
<td>bVerify 76</td>
</tr>
<tr>
<td>bVerifyUseAlways 76</td>
</tr>
<tr>
<td>bWebBuyFDF 98</td>
</tr>
<tr>
<td>bWinCacheSessionHandles 77</td>
</tr>
<tr>
<td>by ASPKI. 93</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>bAllowAPSConfig 77</td>
</tr>
<tr>
<td>bAllowCertNonGreen 122</td>
</tr>
<tr>
<td>bAllowConnectViaHTTP 103</td>
</tr>
<tr>
<td>bAllowInvisibleSig 77, 121</td>
</tr>
<tr>
<td>bAllowOCSPNoCheck 107</td>
</tr>
<tr>
<td>bAllowOtherInfoWhenSigning 120</td>
</tr>
<tr>
<td>bAllowPasswordSaving 76, 100</td>
</tr>
<tr>
<td>bAllowReasonWhenSigning 120</td>
</tr>
<tr>
<td>bAllowSigCertGreenOnly 122</td>
</tr>
<tr>
<td>bAllowSigCertOnly 122</td>
</tr>
<tr>
<td>bAlwaysConsult 109</td>
</tr>
<tr>
<td>Basic Concepts 17</td>
</tr>
<tr>
<td>basic constraint 370</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>CA 370</td>
</tr>
<tr>
<td>cAcceptablePolicyOIDsc1 93</td>
</tr>
<tr>
<td>cAcceptablePolicyOIDsc1cValue 93</td>
</tr>
</tbody>
</table>
### Acrobat Family of Products

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accounts</td>
<td>82</td>
</tr>
<tr>
<td>ACs</td>
<td>85</td>
</tr>
<tr>
<td>CAdES</td>
<td>370, 371</td>
</tr>
<tr>
<td>Can a signer embed revocation information for the cross certificate as well in the signature?</td>
<td>149</td>
</tr>
<tr>
<td>Can CRL download being controlled manually? User driven?</td>
<td>156</td>
</tr>
<tr>
<td>Can I examine the chains that Acrobat builds?</td>
<td>149</td>
</tr>
<tr>
<td>CAPI</td>
<td>370</td>
</tr>
<tr>
<td>Capturing Certificate Validation Data to a Log</td>
<td>127</td>
</tr>
<tr>
<td>cAttest</td>
<td>122</td>
</tr>
<tr>
<td>cAuthMechanisms</td>
<td>97</td>
</tr>
<tr>
<td>cCertIssuerInfo</td>
<td>80</td>
</tr>
<tr>
<td>cCredentials</td>
<td>83</td>
</tr>
<tr>
<td>cDialogs</td>
<td></td>
</tr>
<tr>
<td>xSelHandler</td>
<td>95</td>
</tr>
<tr>
<td>cDigitalIDFiles</td>
<td>85</td>
</tr>
<tr>
<td>cDirectoryProvider</td>
<td>97</td>
</tr>
<tr>
<td>CDS</td>
<td>370</td>
</tr>
<tr>
<td>CDS digital ID</td>
<td>370</td>
</tr>
<tr>
<td>CDS digital ID certificate</td>
<td>370</td>
</tr>
<tr>
<td>cEmbeddedFilePath</td>
<td>87</td>
</tr>
<tr>
<td>CEN</td>
<td>370</td>
</tr>
<tr>
<td>certificate</td>
<td>370</td>
</tr>
<tr>
<td>certificate authority (CA)</td>
<td>370</td>
</tr>
<tr>
<td>Certificate Chain Building</td>
<td>110</td>
</tr>
<tr>
<td>Certificate Chain Validation</td>
<td>125</td>
</tr>
<tr>
<td>Certificate Chains and Trust Anchors /Roots</td>
<td>42</td>
</tr>
<tr>
<td>Certificate issuer and subject seed value</td>
<td>209</td>
</tr>
<tr>
<td>Certificate key usage seed value</td>
<td>210</td>
</tr>
<tr>
<td>Certificate Management</td>
<td>87</td>
</tr>
<tr>
<td>Certificate policy seed value</td>
<td>211</td>
</tr>
<tr>
<td>Certificate Practice Statement OIDs</td>
<td>72</td>
</tr>
<tr>
<td>Certificate Processing and Viewing Enhancements</td>
<td>324</td>
</tr>
<tr>
<td>Certificate Revocation Checking</td>
<td>150</td>
</tr>
<tr>
<td>Certificate Revocation List (CRL)</td>
<td>370</td>
</tr>
<tr>
<td>Certificate Security</td>
<td>321</td>
</tr>
<tr>
<td>Certificate security workflow</td>
<td>290</td>
</tr>
<tr>
<td>Certificate Trust Settings</td>
<td>337</td>
</tr>
<tr>
<td>Certificate trust settings</td>
<td>64, 338, 339</td>
</tr>
<tr>
<td>Certificate Viewer</td>
<td>250</td>
</tr>
<tr>
<td>Trust tab</td>
<td>334</td>
</tr>
<tr>
<td>Certificate viewer</td>
<td>242</td>
</tr>
<tr>
<td>Certificate Viewer details tab</td>
<td>90</td>
</tr>
<tr>
<td>Certificate Viewer information</td>
<td>249</td>
</tr>
<tr>
<td>Certificates</td>
<td></td>
</tr>
<tr>
<td>Contact Information</td>
<td>55</td>
</tr>
<tr>
<td>Selecting a digital ID for export</td>
<td>49</td>
</tr>
<tr>
<td>Verifying originator</td>
<td>251</td>
</tr>
<tr>
<td>Certificates in the Trusted Identities list</td>
<td>45</td>
</tr>
<tr>
<td>Certification Signature</td>
<td>215</td>
</tr>
<tr>
<td>certification signature</td>
<td>370</td>
</tr>
<tr>
<td>Certification Signature (DocMDP)</td>
<td>139, 141</td>
</tr>
<tr>
<td>Certification Signature Preferences</td>
<td>121</td>
</tr>
<tr>
<td>Certification Workflow for Documents with Multiple Signers</td>
<td>218</td>
</tr>
<tr>
<td>certified document</td>
<td>370</td>
</tr>
<tr>
<td>Certified document indicators</td>
<td>218</td>
</tr>
<tr>
<td>Certified Document Services (CDS)</td>
<td>370</td>
</tr>
<tr>
<td>certify or certifying</td>
<td>370</td>
</tr>
<tr>
<td>Certifying a document</td>
<td></td>
</tr>
<tr>
<td>Document integrity warnings</td>
<td>221</td>
</tr>
<tr>
<td>Signature</td>
<td>221</td>
</tr>
<tr>
<td>Certifying a Document is Prevented</td>
<td>257</td>
</tr>
<tr>
<td>Certifying a Dynamic Form</td>
<td>221</td>
</tr>
<tr>
<td>certspec</td>
<td>193, 329</td>
</tr>
<tr>
<td>cExportPath</td>
<td>87</td>
</tr>
<tr>
<td>cExtendedCertInfo</td>
<td>80</td>
</tr>
<tr>
<td>Chain Building FAQs</td>
<td>148</td>
</tr>
<tr>
<td>Chain building log file settings</td>
<td>127</td>
</tr>
<tr>
<td>Changes Across Releases</td>
<td>193, 326</td>
</tr>
<tr>
<td>Changes Allowed</td>
<td>271, 286, 298, 299, 303</td>
</tr>
<tr>
<td>Changes in signature appearances</td>
<td>330</td>
</tr>
<tr>
<td>Changing a PKCS#12 File’s Password Timeout</td>
<td>357</td>
</tr>
<tr>
<td>Changing a Trusted Identity’s Certificate Association</td>
<td>346</td>
</tr>
<tr>
<td>Changing an ID File’s Password</td>
<td>356</td>
</tr>
<tr>
<td>Changing and Viewing Security Settings</td>
<td>272</td>
</tr>
<tr>
<td>Changing Document Collection Passwords</td>
<td>289</td>
</tr>
<tr>
<td>Changing Passwords</td>
<td>365</td>
</tr>
<tr>
<td>Changing the Default Field Appearance</td>
<td>184</td>
</tr>
<tr>
<td>Changing the Default Hashing Algorithm</td>
<td>111</td>
</tr>
<tr>
<td>Changing the Default Signing Method</td>
<td>172</td>
</tr>
<tr>
<td>Changing the Security Method Type</td>
<td>274</td>
</tr>
<tr>
<td>Check revocation</td>
<td>249</td>
</tr>
<tr>
<td>Checking Certificate Revocation Status</td>
<td>251</td>
</tr>
<tr>
<td>Choosing a certificate for encryption</td>
<td>293</td>
</tr>
<tr>
<td>Choosing a digital ID for certificate security</td>
<td>296, 301</td>
</tr>
<tr>
<td>Choosing a Security Method Type</td>
<td>266</td>
</tr>
<tr>
<td>Choosing an Algorithm</td>
<td>268</td>
</tr>
<tr>
<td>Choosing What to Encrypt</td>
<td>268</td>
</tr>
<tr>
<td>cimportPath</td>
<td>87</td>
</tr>
<tr>
<td>Clearing One or More Signatures</td>
<td>225</td>
</tr>
<tr>
<td>Click thru signature</td>
<td>370</td>
</tr>
<tr>
<td>cModules</td>
<td>81</td>
</tr>
<tr>
<td>CMS</td>
<td>370, 371</td>
</tr>
<tr>
<td>Comment or form field may silently change</td>
<td>261, 262</td>
</tr>
<tr>
<td>Compare</td>
<td></td>
</tr>
<tr>
<td>By page</td>
<td>256</td>
</tr>
<tr>
<td>By page summary report</td>
<td>256</td>
</tr>
<tr>
<td>Comparing a Signed Version to the Current Version</td>
<td>255</td>
</tr>
<tr>
<td>Compatibility</td>
<td>285</td>
</tr>
<tr>
<td>Configure the Select Document Components to Encrypt</td>
<td>285, 287, 295, 299</td>
</tr>
<tr>
<td>Configuring Acrobat to use a Timestamp Server</td>
<td>180</td>
</tr>
<tr>
<td>Configuring ALCRMS Settings Manually</td>
<td>307</td>
</tr>
<tr>
<td>Configuring an LDAP Server to Look Up Certificates</td>
<td>67</td>
</tr>
<tr>
<td>Configuring Certificate Handling in Certificate Security Workflows</td>
<td>101</td>
</tr>
<tr>
<td>Configuring Clients to Search a Directory Server for Certificates</td>
<td>67</td>
</tr>
<tr>
<td>Configuring CRL Revocation Checking</td>
<td>108</td>
</tr>
<tr>
<td>Configuring LifeCycle Rights Management Servers</td>
<td>102</td>
</tr>
<tr>
<td>Configuring OCSP Revocation Checking</td>
<td>105</td>
</tr>
<tr>
<td>Configuring Password Caching</td>
<td>99</td>
</tr>
<tr>
<td>Configuring Revocation Checking for Signing</td>
<td>113</td>
</tr>
<tr>
<td>Configuring Servers</td>
<td>306</td>
</tr>
<tr>
<td>Contacts</td>
<td></td>
</tr>
</tbody>
</table>
Index

Displaying the Signer's Certificate 249
Distributing a Trust Anchor or Trust Root 42
Document Behavior After Signing 256
Document contains hidden behavior 261, 262
Document contains links to external PDFs 262
Document Defensibility 217
Document Integrity 371
Document Integrity Checks for 8.0 253
Document Integrity Verification 233
Document links to external content 263
Document Locking 217
Document may not open in the future 262
Document may silently launch menu items 262
Document Message Bar
No dynamic content message 261
Suppressible rich content 260
Document open password 283
Document Property dialog 274
Document Security Defaults 27
Document Security Enhancements 325
Document security settings
ALCRMS security 273
Certificate security 273
Password security 273
Document Status Definitions 246
Document Status Dialog 129
Documentation related to security 15
Does Acrobat handle chain building differently when cross
certificates are used? 148
Does the cross certificate need to be a trust anchor? 148
Does the presence of a cross certificate change the way
revocation checking happens? 149
Downloaded roaming ID certificates 63
DSA 371
dynamic feature warnings 261
Dynamic form certification setting 222
Dynamic XML Forms 140
DynamicContentTrust 72

E

EC 371
Edit Contact dialog 345, 347
Editing a Security Policy 278
Editing Directory Servers Details 343
Editing or Deleting a Signature Appearance 179
Editing Security Method Settings 275
EE 371
EESSI 371
eID 371
Electronic ID 371
electronic signature 371
Emailing a certificate request 48
Emailing Certificate or Contact Data 345
Emailing Server Details 49
Emailing Your Certificate 46
Emailing your certificate 47, 53, 54
embedded JavaScript 371
embedded validation response 372

Embedding Revocation Data in Signatures 113
Embedding Revocation Information in a Signature 205
Embedding Signature Revocation Status 173
Enable text access for screen reader devices for the visually
impaired 271, 286, 298, 300, 303
Enabling a Warnings Comment or Legal Attestation 175
Enabling Document Warning Review 174
Enabling JavaScript to Set Seed Values 195
Encryption algorithm by security type and product version
269, 328
Encryption Algorithms 327
Encryption configuration panel 268
Encryption Workflow 267
end entity certificate (EE) 372
Enhanced security 319
Envelopes 280
Establishing Certificate Trust 150
ETSI 372
ETSI/ESI 372
EU 372
EU Experts Group 372
EU Signature Directive 372
Examine Document 130
Execute a Menu Item 190
Exporting a Certificate Other than Yours to a File 252
Exporting a Trust Anchor 43
Exporting Application Settings with FDF Files 41
Exporting Security Settings to a File 36
Exporting Server Details 51
Exporting Your Certificate 45
External Content 263

F

FDF (Data Exchange) Files 331
FDF changes across releases 332
FDF Files and Security 40
FDF Import and Export 97
FDF preferences 98
filter 193, 203, 329
Finding a Digital ID in a Windows Certificate Store File 362
FIPS 321, 372
FIPS 140 372
flags 193, 202, 206, 330
Forcing a Certification Signature 196
Forcing Signers to Use a Specific Signature Appearance 200
Form Field Fill in, Signing, and/or Other Actions Don't Work
257

G

Generic ID Operations 351
Getting Started with the Wizard 24
Giving Signers the Option to Lock a Document 198
Go to 3D View 190
Go to a Page View 190

H

hardware token 372
Index

Hash algorithm seed value 204, 205
Hidden 183
Hidden but printable 183
How are in-memory changes detected? 138, 139, 140, 141, 142, 143
How are prohibited in-memory changes prevented? 138, 139, 140, 141, 142, 143
How Do I Validate a Timestamp in a Signature? 244
How does Acrobat handle cross certificate rollovers? 148
How is the signature verified? 138, 139, 140, 141, 142
How Signing Happens 133
HSM hardware security module 372

I
iAPIIndex 119
iAuthMechanisms 97
ICA 372
iCheckEvery 79
iDABC 372
Identifying Certificates with a Hash of the Public Key 89
Identity preferences 351
iDirectoryProvider 96
iDisplayValidIcon 126
iEnd 91
iHashAlgo 118
iLastStatus 130
iLogLevel 128
iMaxClockSkew 107, 129
iMaxRevInfoArchiveSize 114
iMaxRevokeInfoCacheLifetime 109
iMaxVerifySession 129
Import Form Data 191
Importing a Certificate From a File 334
Importing a Trust Anchor and Setting Trust 63
Importing Adobe LiveCycle Rights Management Server Settings 60
Importing Adobe LiveCycle Server settings 60
Importing ALCRMS Settings from an FDF file 307
Importing ALCRMS Settings with a Security Settings Import 307
Importing and Exporting Directory Server Settings 344
Importing Application Settings with FDF Files 52
Importing digital ID data 335
Importing Directory Server Settings 59
Importing Multiple Certificates 55
Importing multiple certificates 56
Importing roaming ID Account Settings 61
Importing roaming ID server settings 62
Importing Security Settings 78
Importing Security Settings from a File 37
Importing Security Settings from a Server 38
Importing Someone’s Certificate 54
Importing Timestamp Server Settings 57
iMSStoreTrusted 88
individual digital ID 372
Information assurance components 13
Interaction between OCSP and CRL Preferences 105
Interaction of iReqRevCheck Settings 158
intermediate certificate authority (ICA) 372
Internal Document Signature components 233
iReqRevCheck 106, 109, 113, 117, 125, 158, 160
iRequireReviewWarnings 123
iResourceID 79
iResponseFreshness 107
iRevocationChecker 96
iShowDocumentWarnings 123
iSHS 99
iSigVerificationTime 124
iSize 118
ISO 372
issuer 206
iStart 91
iSVS 99
ITC 372
iURLToConsult 106
iUseArchivedRevInfo 128
iValidityModel 125

J
JavaScript and Dynamic Content Won’t Run 257

K
Kerberos Authentication Mechanism 84
keyUsage 207
Known issues 17

L
LDAP certificate repositories 331
LDAP Directory Server Integration 66
Legal Attestations and Warnings Comments 218
Legal Notice 249
legalAttestations 194, 330
Lifecycle Management 70
LiveCycle Dynamic Forms and the Warning Triangle 253
Local Time versus Timestamp Time 243
lockDocument 195
Lockdown keys 77
Locking a Document During Signing 137
Locking Fields Automatically After Signing 187
Log file for troubleshooting certificate validation 127
Logging in to a Device 366
Logging in to a Digital ID File 355
Logging in to a Roaming ID Account 364
Logging in to a roaming ID server 62
Logging in to an Adobe LiveCycle Rights Management Server 61
Logging in to PKCS#12 Files 358
Long term validation 373
LTV 373

M
Making a contact a trusted identity 57
Making a Field a Required Part of a Workflow 188
Making a Security Policy Favorite 279
Index

Manage Trusted Identities menu item 19
Managing Contacts 344
Managing IDs Stored on Hardware Devices 364
Managing PKCS#12 Digital ID Files 354
Managing Roaming ID Accounts and IDs 362
Managing Windows Digital IDs 362
Managing your ALCRMS Account 308
Manually Configuring a Directory Server 342
Maximum Number of Records to Receive 68, 342
mdp 194, 330
Message digest 373
message digest 233
Method, Revocation Checking, and Validation Time Preferences 235
Mouse Down 190
Mouse Enter 190
Mouse Exit 190
Mouse Up 190
MS 373
MSCAPI 373
MSCAPI Default Behavior 157

N
Name 182
National Institute of Standards (NIST) 373
NIST 373
No external dependencies or dynamic content 261

O
Obtaining an Authorized OCSP Responder 154
OCSP 373
OCSP Default Behavior 154
OCSP responders
   Determining if they are authorized to do rev checks 105
OCSP Response Processing 154
oid 207
OIDs and Certificates 71
On Blur 190
On Focus 190
Online Certificate Status Protocol (OCSP) 373
Only file attachments 268
Open a File 191
Open a Web Link 191
Opening a Certificate-Protected Document 304
Opening a Password-Protected Document 288
Opening an encrypted document
   With certificate security 305
   organizational digital ID 373
Organizational policies 276
Orientation 183
Other Revocation Handlers 157

P
PADES 373
Page content may silently change 262, 263
Password 68, 72, 342
Password caching
   Disabled UI item 100
Password prompt 288
Password Recovery 289
Password security 266, 321
Password security workflow 283
PasswordTimeout 72
Path Discovery
   Building the Chain 144
Path Validation
   Checking a Chain's Health 150
PDF Advanced Electronic Signatures (ETSI TS 102 778) 373
PDF content contains errors 263
PDF Content with variable rendering 262
PDF Signature Report
   Content which cannot be suppressed in preview mode 260
   Suppressed content 261
PDF Signature Reports 259
Permissions Options 270
Permissions panel 270
Permissions password 283
Permissions Workflow 270
Personalizing an ID name 353
PKCS 373
PKCS#11 373
PKCS#11 Configuration 81
PKCS#11 device 373
PKCS#11 digital ID 373
PKCS#11 format 373
PKCS#11 Integration 70
PKCS#11 module 373
PKCS#11 Security Settings menu items 365, 366, 367
PKCS#11 token 373
PKCS#12 374
PKCS#7 374
PKCS#9 374
PKI 374
Play a Sound 191
Play Media (Acrobat 5 Compatible) 191
Play Media (Acrobat 6 Compatible) 191
Policies 249
Policy favorites preferences 102
Policy OID 210
Policy restrictions 341
Policy security method selection 277
Policy Server 369, 374
Port 68, 342
Preconfiguring Location and Contact Information 119
Preference relevance keys 92
Presentation elements may change appearance 262
Preventing End-User Modification 75
Preview document mode preference 172
Preview Mode and Signing Workflows 258
Preview Mode and Validation (View Signed Version) 259
Printing Allowed 270, 285, 297, 299, 302
Private key 374
privileged context 374

Acrobat Family of Products

383
Index

Problems encountered 249
Product renaming 322
Providing Instructions to the Trusted Root Recipients 45
PSC 374
Public Key 374
Public Key Certificate 374
Public key certificate security 266
Public Key Infrastructure 374

Q
QC Qualified (Electronic) Certificate 374
QSCP 374
Qualified Certificate Service Provider 374

R
Read an Article 191
Read Only 183
Reader Enablement 321
Reason field behavior 201
reasons 194, 330
Refreshing the Security Policy List 279, 309
Registering a Digital ID for Use in Acrobat 349
Registry preferences
  Appearance 119
  Certification signature 121
  Chain building 110
  Chain validation method 125
cPPKHandler 85
CRL 109, 110
Default timestamp server 117
Digital ID 80
Digital ID path 87
Document status dialog 130
Document warning 123
Examine document 130
FDF import and export 98
FIPS mode 101
Font warning 123
Identrus 93
Kerberos 84
LifeCycle Rights Management Server 103
LifeCycle Rights Management Server list 103
Logging 128
Long term validation 114
Message digest comparison 115
MSCAPI provider 88
OCSP certificate 107, 108
OCSP method 106
Password caching 100
PKCS#11 81
Preview mode 119
Revocation checking (signing) 113
Revocation checking constraints 129
Roaming ID account 82
Roaming ID server 82
Roaming ID user account (set by the server) 82
Scoping certificate 91
Security envelope 104
Security handler 94
Security setting import 79
Security Settings Console 99
Self-signed digital ID 80
Signature property retrieval 115
Signature status icon 126
Signature validation 99, 128
Signing algorithm 112
Signing format 113
Signing information 120
Signing reason 120
SPI for authentication mechanisms 97
SPI for directory providers 96
SPI for revocation checker providers 96
subject to feature lockdown 76
Timestamp server list 116
Trusted Identity Manager 89
Registry setting’s impact on signature status during
signature creation 158
Registry setting’s impact on signature status during
signature validation 160
Removing a contact 294
Removing Document Security 275
Removing Password Security 288
Requesting a Certificate via Email 48
Requesting a Digital ID via Email 334
Required 183
Required field not signed alert 189
Requiring a Pre-Signing Digest Comparison 114
Requiring Document Warning Review Prior to Signing 175
Requiring FIPS-Compliant Encryption Algorithms 101
Requiring Preview Mode 171
Requiring Preview Mode to Sign 119
Requiring Signature Property Retrieval 115
Reset a Form 191
Resource roadmap 14
Responding to an Email Request for a Digital ID 52
Restricting Signing to a Roaming ID 211
Revalidate signatures warning 245
RevInfoArchival 72
RevInfoArchival signature property format 72
Revocation 249
Revocation Checker Providers 95
Revocation checking
  GUI strings for CRL check 164
  GUI strings for MSCAPI check 166
  GUI strings for OCSP check 165
  Revocation info button text 167
  Strings for special conditions 167
Revocation Checking Default Behavior 154
Revocation Checking Order of Precedence 152
Revocation Checking Preferences 104
Revocation Checking Process 316
Revocation checking registry entry 161
Revocation Checks During Signature Creation 158
Revocation Checks During Signature Validation 159
Revocation Tab GUI and Strings 163
Revocation Types 152
Revoking a Document 310
Index

roaming ID 375
Roaming ID Configuration 81
Roaming ID Provider Persistent Storage 82
Roaming ID Security Settings menu items 364
Roaming ID seed value 212
Roaming ID server name and URL 62
Roaming ID Server-Set Preferences 82
root certificate 375
Rules for opening a PDF via FDF 41
Run a JavaScript 191

S
Saving Certificate or Contact Details to a File 346
Saving Your Digital ID Certificate to a File 47
Search Base 68, 342
Searching for a document recipients 337
Searching for Digital ID Certificates 335
Searching for group contacts 304
Secure channel error 115
Secure Identity Across Borders Linked (STORK) 375
Secure Signature Creation Device 375
Secure time 236
Security 71
Security and Encryption of PDF packages (portfolios) 322
Security envelope 281
Security Envelopes 104
Security Method Basics 265
Security method pros and cons 267
Security method selection 266
Security method selection from Policy Manager 276
Security method workflow 265
Security Methods and Encryption 267
Security Methods and Permissions 269
Security methods for batch processing 272
Security OIDs 72
Security Policies 267
Reusable Security Settings 276
Security policy
Favorites list 279
General settings 277, 284, 295
security restricted property or method 375
Security setting import
Success dialog 38
Security Setting Import and Export 35
Security setting import and export 319
Security setting preferences for server import 39
Security settings
Document message bar 37
Encryption method 37
Export dialog 36, 78
Import from a file panel 38
Security Settings Console 98, 350
Security settings icon 274
Security settings menu and manager 350
Security Settings menu items 49
Security settings require “save” alert 288
Security Terms 369

Security workflow for encryption-permissions 318
Seed value
Custom signing reason 201
Forcing mdp selection during certification 196
lockDocument 199
mdp 198
Reason not allowed error 201
signatureAppearance 200
Specifying certificates for signing 206
Specifying signature components 204
Seed Value Basics 192
Seed Values 329
Seed values
certSpec properties 206
Changes across releases 329
Custom legal attestations 198
JavaScript debugger 196
Object properties and descriptions 193
timeStampspec properties 202
Select Document Components to Encrypt panel 268
Selecting a certificate chain for export 44
Selecting a Certificate to Use for Encryption 292
Selecting a digital ID 53
Sending an OCSP Request to the Responder 154
Server Name 68, 342
Set Layer Visibility 191
Set the compatibility level 287
Setting Basic Client and Workflow Preferences 75
Setting Certificate Trust 338
Setting Digital ID Defaults 84
Setting Identity Information 350
Setting Signing Preferences 171
Setting the Certificate Trust Level 45
Setting up a Document for Certification 219
Setting up Security Policy Favorites 102
Setting up Signature Validation 69
Setting up the Certificate Security Environment 291
Setting up the Signing Environment 170
Setting up Timestamp Servers 116
Setting up Your Environment for Signature Validation 234
Setting User Interface Preferences 124
Sharing (Exporting) a Digital ID Certificate 352
Sharing Settings & Certificates with FDF 39
sHashAlgo 118
shouldAddRevInfo 194, 330
Show/Hide a Field 191
Showing Location and Contact Details 174
slid 102
Sign Document dialog
With Lock Document checkbox added 199
Signature Algorithm 375
Signature appearance
Configuration 178
New button 177, 227
Signature Appearance Customizations 176
Signature Appearances 330
Signature Creation Preferences 111
Signature creation preferences 173
Index

Signature Creation Workflow 313
Signature Enhancements 324
Signature field
  Action properties 190
  Appearance properties 184
  Default appearance 182
  Edit options 185
  General properties 183
  Multiple copy options 186
  Signing properties 188
Signature field sign menu 223
Signature Format 135
Signature Image 375
Signature Properties
  Summary 241
Signature Report Error Codes 261
Signature Status and Troubleshooting Guide 317
Signature Status Definitions 246
Signature Technical Details 132
Signature type support in XML forms 137, 331
Signature Types 215
Signature Validation 135
Signature validation confirmation 240
Signature Validation Defaults 26
Signature Validation Preferences 124
Signature validation preferences 124
Signature Validity Basics 232
Signature verification preferences 235
Signature with Field Restrictions (FieldMDP) 139, 141
Signatures and Permissions 135
Signatures and signing workflows 319
Signatures Created When Reader Enabling (UR3) 140, 142
Signatures tab
  Validate signature 239
Signatures, Permissions, and the PDF Language 315
Signer Details 249
Signing a document
  Signature details 224
Signing an FDF file 42
Signing and the Byte Range 314
Signing Basics 215
Signing Documents in Acrobat 222
Signing Environment Defaults 25
Signing environment preferences 171
Signing in a Browser 224
Signing of PDF packages (portfolios) 322
Signing User Interface 216
Signing With a Certification Signature 216
Signing with an Approval Signature 222
Signing Workflow Preferences 118
Simple certificate chain 145
sLDAP 109, 111
sLogFilePath 128
sPassword 117
Specifying a Default Directory Server 344
Specifying a Post-Signing Action 189
Specifying a Signature Hash Algorithm 204
Specifying a Signing Format 112
Specifying a URL When a Valid Certificate is not Found 211
Specifying Alternate Signature Handlers and Formats 203
Specifying an Alternate OCSP Responder 161
Specifying Certificate Properties for Signing 205
Specifying Certificates by Key Usage 209
Specifying Certificates by Policy 210
Specifying Certificates for Valid CRL Checks 109
Specifying Certificates for Valid OCSP Checks 107
Specifying Certificates for Valid OCSP Requests 108
Specifying Digital ID Usage 351
Specifying extended certificate information 80
Specifying General Field Properties 182
Specifying preference relevance 92
Specifying Signature Appearances 119
Specifying Signing Certificates Origin 208
Specifying Signing Reasons 120
Specifying the Time and Method of CRL Checks 108
Specifying the Time and Method of OCSP Checks 106
Specifying Timestamps for Signing 201
SSCD 375
sServiceName 84
sSignCertOID 108
sSuppressListOrder 130
Standard PDF and Static XML Forms 138
Standards support 368
Status Icons and Their Meaning 246
STF 364 375
STORK 375
SubCA 375
subFilter 194, 330
subfilter 203
subject 207
subjectDN 207
Submit a Form 191
Summary 249
Supported Seed Values 193
sURL 107, 109, 117
sUser 117
Synchronizing a Document for Offline Use 310

T
tAddressBook 89
tContactInfo 120
tCredProvider 85, 86
Test 73
Text appearance may silently change 263
tFriendlyName 82
The document contains a dynamic form 262
This server requires me to log on 68, 342
Time stamp server error 202
Time verification changes from 9.0 to 9.1 236
Timeout 68, 342
Time-stamp 74
Timestamp 375
Timestamp Revocation Checking 153
Timestamp server seed value 202
Timestamp UI preference 153
Timestamp Usage for Signing 179
Time-stamp X509 extension format 74
Index

Timestamps
  Date/Time tab 245
  Entering server details 181
  Importing a server 58
  Importing server details from an FDF file 58
  Local, machine time 179, 243
  Trusted stamp 180, 244
  Untrusted stamp 179, 244
timeStampSpec 194, 330
TL 375
tLastEnvelope 104
tLastSearchPath 104
tLastServerURL 103
tLoadSettingsURL 79
tLocation 120
tLockboxId 103
tName 102, 103
Tooltip 182
Troubleshooting a Document Integrity Problem 252
Troubleshooting a Signature or Document Status 247
Troubleshooting an Identity Problem 247
Troubleshooting Digital ID Certificates 248
Trust 249
Trust (Trust service) Status List 375
trust anchor 375
Trust List 375
Trusted Identities
  Viewing revocation status 252
Trusted identities 21
Trusted Identity List Configuration 88
Trusting certificate from a document warning 242
Trusting Windows root certificates 237
TS 376
tSAML_Assertion_Expiration 83
tSAML_Assertion_Source 83
tSAML_Name_Format 83
tSAML_Name_Qualifier 84
tSAML_Name_Value 83
tSAML_NameQualifier 84
tSAML_NameValue 83
tServer 103
tServerName 82, 103
tServerURL 82, 103
tSignHash 112
TSL 376
TSP 376
tURL 82
Turning on FIPS Mode 100

Unrecognized PDF content 263
Untrusted signature 337
url 202, 207
urlType 208
Usage options for a digital ID 352
Use this digital ID until I close the application 296, 301
User name 68, 342
User policies 276
Using Certificates for Certificate Security (Encryption) 341
Using Directory Servers to Add Trusted Identities 341
Using Embedded Validation Data 128
Using Root Certificates in the Windows Certificate Store 236
Using Seed Values to Individual Form Fields 219

V
Validate all signatures dialog 240
Validating a Single Signature in Acrobat 239
Validating All Signatures in Acrobat 239
Validating an Problematic Signature (trusting a signer on-the-fly) 240
Validating Signature Timestamps 243
Validating Signatures Automatically 234
Validating Signatures for other Document Versions 243
Validating Signatures Manually 238
Validating Signatures with Adobe Reader 239
Validating Signatures with Timestamps and Certificate Policies 237
Validation period expired alert 309
Verifying the Identity of Self-Signed Certificates 250
version 194, 330
View a Document’s Audit History 310
Viewing a List of Post-Signing Modifications 254
Viewing a Security Policy 278
Viewing All of Your Digital IDs 352
Viewing and Comparing Changes and Versions 254
Viewing and Editing Contact Details 344
Viewing cross certificate chains in the Certificate Viewer 149
Viewing Digital ID Certificates in the Certificate Viewer 353
Viewing Document Encryption and Permission Settings 272
Viewing Document Restrictions 273
Viewing Security Settings in a Browser 274
Visible 183
Visible but doesn’t print 183

W
What happens if a cross certificate has an issue date after the signature date? 148
What is a Digital ID? 19
What is a timestamp? 244
What is a Trusted Identity? 18
What is an Object Identifier? (OIDs) 71
What is digested and signed for certification signatures? 141
What is digested and signed? 138, 139, 140, 142
What is Security and Information Assurance? 17
What is the behavior if there are more than one valid source for CRLs (LDAP and HTTP)? 156
What is the caching behavior, if no CRL can be downloaded? 156

Acrobat Family of Products 387
Index

What is the caching behavior, if the CRL is corrupt/syntactically incorrect? 156
What is the caching behavior, if the CRL is obsolete? 156
What is Trust? 17
What Makes a Signature Valid? 232
What’s Different with XML Forms? 134
What’s New for 8.0 323
What’s New for 8.1 322
What’s New for 9.0 319
What’s new for 9.1? 15

When is the revocation status of a public-key certificate (PKC) checked against a CRL? 155
When Timestamps Can’t be Verified... 246
When will a CRL be loaded/updated? 156
Why are You Here? 13
Why Can’t I Certify? 222
Will Acrobat check online for valid CDPs even if local, valid CRL is cached? 157
Windows digital ID menu 362
Windows Integration 87
Windows integration 292
Wizard
  Digital signature panel 32
  Files and Folders window 31
  Main user interface 29
Wizard Setup 28
Working with Digital IDs 349
Working with Documents and ALCRMS Policies 308
Working with Groups of Contacts 293
Working with Signature Fields 181

X

X.509 Extension OIDs 74
X.509v3 376
X509 Extension OIDs 74
X509-Extension 71
XAdES 376
xCertSHA1 85, 86
xdata 79
xDefEnrollmentURL 85
xDocID 130
xLockboxId 117, 118
XML Form Support for Signatures 331